GY

中华人民共和国广播电视行业标准

GY/T 303.3—2018

智能电视操作系统
第3部分：应用程序编程接口

Smart TV operating system—
Part 3: Application programming interface

2018－07－06 发布　　2018－07－06 实施

国家广播电视总局　　发布

目　　次

前　言

GY/T 303《智能电视操作系统》已经或计划发布以下部分：

——第1部分：功能与架构；

——第2部分：安全；

——第3部分：应用程序编程接口；

——第4部分：硬件抽象接口；

——第5部分：功能组件接口；

——第6部分：可信执行环境接口；

——第7部分：符合性测试。

本部分为GY/T 303的第3部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分由全国广播电影电视标准化技术委员会（SAC/TC 239）归口。

本部分起草单位： 国家新闻出版广电总局广播科学研究院、国家广播电视网工程技术研究中心、华为技术有限公司、深圳市海思半导体有限公司、四川长虹网络科技有限责任公司、北京永新视博数字电视技术有限公司、深圳市茁壮网络股份有限公司、东方有线网络有限公司、创维数字技术股份有限公司、北京数码视讯科技股份有限公司、四川九州电子科技股份有限公司、中兴通讯股份有限公司、上海联彤网络通讯技术有限公司、北京赛科世纪科技股份有限公司、北京数字太和科技有限责任公司、上海兆芯集成电路有限公司、中国有线电视网络有限公司。

本部分主要起草人：盛志凡、白伟、同磊、咎元宝、赵良福、蒋艳山、付瑞、杨明磊、万乾荣、程伯钦、严海峰、张晶、陈亚东、李洪浩、袁宏伟、刘金晓、解伟、郭沛宇、马万铮、郭永伟、管丹东、何剑、赵学庆、王明敏、董进刚、宋勇立、丁送星、张定京、王颖、郭晓霞、王磊、曾品超、郭成、陶春、黄玲玲、梁志坚、杨波涛、李晓榕、方中华、孙明勇、汤新坤、贾庭兰、刘鹏、白鹤、谌颖、杨旭、李爽、刘江。

引　言

本部分的发布机构提请注意，声明符合本部分时，可能使用涉及本部分有关内容的相关授权的和正在申请的专利如下：

序号	标准章条号	专利名称
1	5	一种智能电视操作系统
2	5	一种智能电视系统
3	4.1.6、4.2.6、附录F、附录Q	一种在智能电视操作系统中支持全媒体播放的方法及智能电视终端
4	4.1.11、4.2.14、附录K、附录Y	一种用于智能操作系统的条件接收方法和系统
5	4.1.10、4.2.13、附录J、附录X	一种用于智能操作系统的数字版权管理（DRM）方法和系统
6	4.1.10、4.2.13、附录J、附录X	一种支持数字版权管理（DRM）的媒体网关/终端实现方法及其设备

本部分的发布机构对于该专利的真实性、有效性和范围无任何立场。

该专利持有人已向本部分的发布机构保证，他愿意同任何申请人在合理且无歧视的条款和条件下，就专利授权许可进行谈判。该专利持有人的声明已在本部分的发布机构备案，相关信息可以通过以下联系方式获得：

专利权利人	联系地址	联系人	邮政编码	电话	电子邮箱
国家新闻出版广电总局广播科学研究院	北京市西城区复兴门外大街2号	孟祥昆	100866	010-86098010	mengxiangkun@abs.ac.cn

请注意除上述专利外，本部分的某些内容仍可能涉及专利。本部分的发布机构不承担识别这些专利的责任。

智能电视操作系统
第3部分：应用程序编程接口

1 范围

GY/T 303的本部分规定了智能电视操作系统应用程序编程接口的相关技术要求和详细接口定义。

本部分适用于智能电视操作系统的应用开发和测试，包括JAVA应用和WEB应用两部分。

2 规范性引用文件

下列文件对于本部分的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本部分。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本部分。

GY/T 255—2012 可下载条件接收系统规范

GY/T 303.1—2016 智能电视操作系统 第1部分：功能与架构

GY/T 303.2—2016 智能电视操作系统 第2部分：安全

GB/T 4880.2—2000 语种名称代码 第2部分：3字母代码

GB/T 17975.1—2010 信息技术 运动图像及其伴音信息的通用编码 第1部分：系统

GB/T 20090.2—2006 信息技术 先进音视频编码 第2部分：视频

GB/T 28160—2011 数字电视广播电子节目指南规范

GB/T 28161—2011 数字电视广播业务信息规范

ISO/IEC 13522-1-2000 信息技术 多媒体和超媒体信息的编码(Information technology.Coding of multimedia and hypermedia information)

ISO/IEC 13818-1-2018 信息技术 移动图片和相关音频信息的普通编码 第1部分：系统（Generic Coding of Moving Pictures and Associated Audio Part1:Systems）

ISO/IEC 13818-6-1998 信息技术 移动图片和相关音频信息的普通编码 第6部分：DSM-CC的扩展（Generic Coding of Moving Pictures and Associated Audio Part6: Extension for Digital Storage Media Command and Control）

IETF RFC 791 互联网协议（Internet protocol）

IETF RFC 2373 IPv6寻址系统结构（IP Version 6 Addressing Architecture）

ECMA-262 ECMAScript语言规范（ECMAScript Language Specification）

3 缩略语

下列缩略语适用于本部分。

App 应用程序（Application）

AV 音视频（Audio Video）

CA 认证机构（Certification Authority）

CAS 条件接收系统（Conditional Access System）

CSS 样式级联表（Cascading Style Sheets）

DCAS 可下载条件接收系统（Downloadable Conditional Access System）
DTV 数字电视（Digital Television）
DRM 数字版权管理（Digital Rights Management）
DVB 数字视频广播（Digital Video Broadcasting）
ECM 授权控制信息（Entitlement Control Message）
EMM 授权管理信息（Entitlement Management Message）
EPG 电子节目指南（Electronic Program Guide）
ES 基本码流（Elementary Stream）
HTML 超文本标记语言（Hyper Text Markup Language）
HTTP 超文本传输协议（Hyper Text Transfer Protocol）
IPTV IP电视（IP Television）
JS 脚本语言（JavaScript）
MPEG 动态图像专家组（Moving Picture Experts Group）
OSD 屏幕叠加显示（On-Screen Display）
OTA 空中升级（Over The Air）
PID 包识别码（Packet Identifier）
TApp 可信应用（Trusted Application）
TEE 可信执行环境（Trusted Execution Environment）
TS 传送流（Transport Stream）

4 接口概述

4.1 接口概述

TVOS 应用框架层所能支撑的应用分为 JAVA 应用和 WEB 应用两大类：

——JAVA 应用是指采用 Java 语言开发的应用的统称；

——WEB 应用是指采用 HTML、JavaScript、CSS 等 Web 技术开发的应用的统称。

TVOS 应用框架层实现 JAVA 应用和 WEB 应用与功能组件模块的封装适配。本文档定义的应用程序编程接口为应用层的 JAVA 应用和 WEB 应用提供了统一的应用程序编程接口，为 TVOS 应用开发者开发 JAVA 应用和 WEB 应用提供参考。

本部分定义的接口符合 GY/T 303.1—2016 和 GY/T 303.2—2016 的相关要求。

4.2 TVOS JAVA 应用程序编程接口

4.2.1 TVOS JAVA 应用程序编程接口概述

TVOS JAVA应用程序编程接口以Java对象的方式提供调用接口，支撑应用实现电子节目指南、频道列表、电视节目播放等数字电视相关业务功能。JAVA应用程序编程接口共由11个功能单元组成，包括单向广播网络接入单元、广播协议处理单元、双向宽带网络接入单元、人机交互单元、AV设置单元、媒体处理单元、系统管理单元、应用引擎单元、多屏互动单元、DRM管理单元和DCAS管理单元。

4.2.2 单向广播网络接入单元

4.2.2.1 单向广播网络接入单元概述

单向广播网络接入单元用于实现单向广播网络接入功能，包括调谐的频率、调制方式、符号率等参数控制及信号强度和质量等信息的获取。该单元定义了调谐解调模块。Java接口详细定义见附录A。

4.2.2.2 调谐解调模块

调谐解调模块定义了调谐和解调用到的接口、类和异常，包括调谐参数定义、Tuner对象定义、调谐解调管理类等，通过这些定义实现了调谐解调功能。

调谐解调模块概要见表1。

表1 调谐解调模块概要

对象名	类型	说明	备注
DeliverySystemType	接口	DVB技术体系下的传送系统类型常量定义。	见A.2.1
TuningParameters	接口	调谐解调参数接口。	见A.2.2
TuningListener	接口	网络接口事件监听器接口,提供了网络接口相关事件处置的回调方法。	见A.2.3
DvbcTuningParameters	类	适用于DVB-C传送系统的调谐解调参数类。	见A.2.4
AbsssTuningParameters	类	适用于ABS-SS传送系统的调谐解调参数类。	见A.2.5
DtmbTuningParameters	类	适用于DTMB传送系统的调谐解调参数类。	见A.2.6
TunerEvent	类	调谐解调器事件类。	见A.2.7
TunerTuningEvent	类	网络接口开始调谐事件，继承TunerEvent类。	见A.2.8
TunerTuningOverEvent	类	结束调谐事件，继承TunerEvent类。	见A.2.9
Tuner	类	调谐解调控制接口。	见A.2.10
TunerManager	类	调制解调管理器类,用于跟踪连接到接收设备的广播网络接口，是调谐解调单元的入口类。	见A.2.11
TunerException	异常	网络接口异常。	见A.2.12
IncorrectLocatorException	异常	定位符格式不正确异常，继承TunerException类。	见A.2.13
StreamNotFoundException	异常	流未发现异常，继承TunerException类。由于传送流不在StreamTable中而导致对该传送流的引用无法被解析时，则抛出此异常。	见A.2.14
TuningParameterNotFoundException	异常	获取当前调谐解调参数失败或者DeliverySystemType 错误的时候，抛出异常。	见A.2.15

4.2.3 广播协议处理单元

4.2.3.1 广播协议处理单元概述

广播协议处理单元用于实现广播协议的处理，包括了MPEG对象定义模块、DVB对象定义模块、SECTION段过滤模块、URL封装模块、DVB定位符模块和广播协议处理模块。Java接口详细定义见附录B。

4.2.3.2 MPEG 对象定义模块

MPEG对象定义模块定义了MPEG-2体系下最基本的对象以及系统可能出现的异常。

本模块定义的最基本的MPEG-2对象有：

——传送流类（TransportStream）；

——基本流类（ElementaryStream）；

——广播业务类（Service）。

定义的MPEG-2异常有：

——广播内容未授权异常（NotAuthorizedException）；
——资源异常（ResourceException）；
——调谐解调异常（TuningException）。
MPEG对象定义模块概要见表2。

表 2　MPEG 对象定义模块概要

对象名	类型	说明	备注
NotAuthorizedInterface	接口	广播内容未授权报告接口，定义了失败原因常量，提供了查找失败原因的方法。	见B.2.1
TransportStream	类	MPEG-2 传送流类，代表一个 MPEG-2 传送流（TS），提供了获取传送流信息的方法。	见B.2.2
ElementaryStream	类	MPEG-2 基本流类，代表一个在传送流（TS）中承载的基本流（ES），提供了获取基本流信息的方法。	见B.2.3
Service	类	MPEG-2 业务类，代表一个在传送流中承载的 MPEG-2 业务，提供了获取业务信息的方法。	见B.2.4
NotAuthorizedException	异常	广播内容未授权异常，实现 NotAuthorizedInterface 接口，当访问无授权的加扰数据时抛出。	见B.2.5
TuningException	异常	调谐解调异常，当调谐解调失败时抛出。	见B.2.6
ResourceException	异常	资源异常，当因资源缺乏而无法进行操作时抛出。	见B.2.7

4.2.3.3　DVB 对象定义模块

DVB 对象定义模块定义了 DVB 体系下的 MPEG-2 基本对象：
——DVB 传送流类（DvbTransportStream）；
——DVB 基本流类（DvbElementaryStream）；
——DVB 广播业务类（DvbService）。
DVB 对象定义模块概要见表 3。

表 3　DVB 对象定义模块概要

对象名	类型	说明	备注
DvbElementaryStream	类	DVB 基本流类，代表一个在传送流（TS）中承载的符合 DVB 语义约束的 MPEG-2 基本流（ES），提供了获取 DVB 基本流信息的方法。	见B.3.1
DvbService	类	DVB 业务类，代表一个在传送流中承载的符合 DVB 语义约束的 MPEG-2 业务，提供了获取 DVB 业务信息的方法。	见B.3.2
DvbTransportStream	类	DVB 传送流类，代表一个符合 DVB 语义约束的 MPEG-2 传送流，提供了获取 DVB 传送流信息的方法。	见B.3.3

4.2.3.4　SECTION 段过滤模块

SECTION 段过滤模块提供了与 MPEG-2 段(section)过滤相关的类和方法。
SECTION 段过滤模块概要见表 4。

表 4 SECTION 段过滤模块概要

对象名	类型	说明	备注
SectionFilterListener	接口	段过滤事件监听器接口，提供了段过滤事件处置回调方法，由应用层实现。	见B.4.1
Section	类	MPEG-2 段类，描述了从传输流中过滤到的一个段。	见B.4.2
SectionFilterGroup	类	段过滤器组类，代表一个 MPEG-2 过滤器组，可以作为一个基本操作单元被激活和释放。	见B.4.3
SectionFilter	类	段过滤器类，该类为一组具有不同生命周期和缓存长度特点的段过滤器类的基类，提供了过滤器基本操作方法。	见B.4.4
SimpleSectionFilter	类	简单段过滤器类，继承 SectionFilter 类。	见B.4.5
TableSectionFilter	类	表段过滤器类，继承 SectionFilter 类。	见B.4.6
RingSectionFilter	类	循环段过滤器类，继承 SectionFilter 类。	见B.4.7
SectionFilterEvent	事件	段过滤事件类，一组段过滤事件类的基类。	见B.4.8
SectionAvailableEvent	事件	段数据可用事件，继承 SectionFilterEvent 类，报告过滤到了一个完整的段。	见B.4.9
VersionChangeDetectedEvent	事件	段过滤版本变更事件，继承 SectionFilterEvent 类。	见B.4.10
EndOfFilteringEvent	事件	段过滤结束事件，继承 SectionFilterEvent 类，报告段过滤结束。	见B.4.11
IncompleteFilteringEvent	事件	段过滤不完整事件，继承 EndOfFilteringEvent 类。	见B.4.12
TimeOutEvent	事件	段过滤超时事件，继承 EndOfFilteringEvent 类。	见B.4.13
FilterResourcesAvailableEvent	事件	过滤器资源可用事件，继承 ResourceStatusEvent 类。	见B.4.14
ForcedDisconnectedEvent	事件	段过滤器组与传送流强制断开事件，继承 ResourceStatusEvent 类。	见B.4.15
SectionFilterException	异常	段过滤异常的基类。	见B.4.16
ConnectionLostException	异常	连接丢失异常，继承 SectionFilterException 类。	见B.4.17
FilteringInterruptedException	异常	过滤中断异常，继承 SectionFilterException 类。	见B.4.18
FilterResourceException	异常	过滤器资源异常，继承 SectionFilterException 类。	见B.4.19
IllegalFilterDefinitionException	异常	非法过滤条件异常，继承 SectionFilterException 类。	见B.4.20
InvalidSourceException	异常	段数据源无效异常，继承 SectionFilterException 类。	见B.4.21
NoDataAvailableException	异常	段对象无可用数据异常，继承 SectionFilterException 类。	见B.4.22

4.2.3.5 URL 封装模块

URL 封装模块提供了 URL 封装的引用方法。

URL 封装模块概要见表 5。

表 5 URL 封装模块概要

对象名	类型	说明	备注
Locator	类	资源定位符类，将URL封装成定位符对象。	见B.5.1
InvalidLocatorException	异常	定位符无效异常。	见B.5.2

4.2.3.6 DVB 定位符模块

DVB 定位符模块提供了访问 DVB 广播业务及其内容的引用方法。

DVB 定位符模块概要见表 6。

表 6 DVB 定位符模块概要

对象名	类型	说明	备注
DvbLocator	类	DVB 定位符类，将 DVB 格式的 URL 封装成定位符对象。	见B.6.1
DvbNetworkBoundLocator	类	与网络绑定的 DVB 定位符类，此类对象唯一标识一个给定的实体和传送系统。	见B.6.2

4.2.3.7 广播协议处理模块

广播协议处理模块定义了与DVB广播协议处理相关的类和方法。

广播协议处理模块概要见表7。

表 7 广播协议处理模块概要

对象名	类型	说明	备注
SICommonInformation	接口	PSI/SI 公共信息接口，提供了获取 PSI/SI 公共特性的方法。	见B.7.1
SINetwork	接口	网络信息接口，提供了获取网络(network)信息的方法，每个 SINetwork 对象由 network_id 唯一标识。	见B.7.2
SIBouquet	接口	业务群信息接口，提供了获取业务群(bouquet)信息的方法，每个 SIBouquet 对象由 network_id、bouquet_id 唯一标识。	见B.7.3
SIService	接口	业务信息接口，提供了获取业务(service)信息的方法，每个 SIService 对象由 network_id、original_network_id、transport_stream_id 和 service_id 共同唯一标识。	见B.7.4
SITransportStream	接口	传送流信息接口，提供了获取传送流(transport_stream)信息的方法，每个 SITransportStream 对象由 network_id、original_network_id 和 transport_stream_id 共同唯一标识。	见B.7.5
SIElementaryStream	接口	基本流信息接口，提供了获取基本流(elementary_stream)信息的方法，每个 SIElementaryStream 对象是由 network_id、original_network_id、transport_stream_id、service_id 和 component_tag(或 elementary_PID)共同唯一标识。	见B.7.6
SIEvent	接口	节目事件信息接口，提供了获取事件(event)信息的方法，每个 SIEvent 对象由 network_id、original_network_id、transport_stream_id、service_id 和 event_id 共同唯一标识。	见B.7.7
SITime	接口	时间信息接口，提供了获取时间信息的方法，时间信息从 TDT 或 TOT 获得，每个 SITime 对象由 network_id 唯一标识。	见B.7.8
SIDescriptor	接口	描述符信息接口，提供了与描述符访问相关的方法。	见B.7.9
SIRequest	接口	PSI/SI 信息请求接口，描述了应用产生的一次 PSI/SI 信息检索请求，应用可以通过该对象取消该次请求。	见B.7.10
SIRetrieveListener	接口	SI 信息获取事件监听器，由应用程序实现。	见B.7.11
SIUpdateListener	接口	PSI/SI 表更新事件监听器，由应用程序实现。	见B.7.12

表 7（续）

对象名	类型	说明	备注
SIDescriptorTag	接口	描述符标签常量定义接口，取值详见 GB/T 28161—2011。	见B.7.13
SIRunningStatus	接口	广播业务(service)或节目(event)的运行状态常量定义接口，取值详见 GB/T 28161—2011。	见B.7.14
SIServiceType	接口	业务类型常量定义接口，取值详见 GB/T 28161—2011。	见B.7.15
SIStreamType	接口	流类型常量定义接口，取值详见 GB/T 17975.1—2010。	见B.7.16
SIDatabase	类	PSI/SI 信息数据库，提供了 DVB 模式下 PSI/SI 信息数据库的管理和操作方法，是应用获取 PSI/SI 信息的入口类。	见B.7.17
SIRequestFailureType	类	PSI/SI 信息请求失败的原因。	见B.7.18
SIRetrieveEvent	事件	PSI/SI 信息请求事件，是本包定义的一组跟 PSI/SI 信息请求相关的事件的基类。一次 PSI/SI 信息请求只会产生一个这样的事件。	见B.7.19
SISuccessRetrieveEvent	事件	PSI/SI 信息请求成功事件，继承 SIRetrieveEvent 类。	见B.7.20
SIFailureRetrieveEvent	事件	PSI/SI 信息请求失败事件，继承 SIRetrieveEvent 类。	见B.7.21
SIUpdateEvent	事件	PSI/SI 表更新事件。	见B.7.22
InvalidPeriodException	异常	当指定的日期无效时，抛出该异常。	见B.7.23

4.2.4 双向宽带网络接入单元

4.2.4.1 双向宽带网络接入单位概述

双向宽带网络接入单元用于实现与双向宽带网络接入控制相关的功能，包括双向网络的连接管理和数据操作等，定义了以太网管理模块和WiFi管理模块。Java接口详细定义见附录C。

4.2.4.2 以太网管理模块

以太网管理模块提供 DHCP 配置以及对以太网的管理控制的类和方法。

以太网管理模块概要见表 8。

表 8 以太网管理模块概要

对象名	类型	说明	备注
Listener	接口	以太网状态改变事件监听器。	见C.2.1
DhcpInfo	类	提供了获取 DHCP 配置信息的方法。	见C.2.2
EthernetManager	类	提供以太网管理的方法。	见C.2.3

4.2.4.3 WiFi 管理模块

WiFi 管理模块提供了与 WiFi 网络接口控制相关的类和方法。

WiFi 管理模块概要见表 9。

表 9 WiFi 管理模块概要

对象名	类型	说明	备注
ActionListener	接口	WiFi状态改变事件监听器，由应用层实现。	见C.3.1
WifiInfo	类	WiFi 连接信息类，提供了获取 WiFi 连接信息的方法。	见C.3.2
WifiManager	类	WiFi 管理器，提供 WiFi 无线网络的管理功能： a) 发现、扫描和管理当前可用的无线网络访问点(AP)； b) 管理当前活动的网络访问链接，如建立连接、断开连接、禁用连接、删除连接等。	见C.3.3
ScanResult	类	WiFi 扫描结果，描述了 WiFi 扫描发现的一个连接访问点信息。	见C.3.4

4.2.5 人机交互单元

4.2.5.1 人机交互单元概述

人机交互单元用于实现设备输入控制以及前面板显示控制功能，输入把遥控器、鼠标、键盘、前面板按键等输入设备发送的用户指令封装成按键消息，输出通过前面板或显示屏幕反馈信息；同时，还支持对语音输入的控制和实现。该单元定义了人机交互模块。Java接口详细定义见附录D。

4.2.5.2 人机交互模块

人机交互模块提供了人机交互相关的类和方法，包括用户输入和前面板输出。

人机交互模块概要见表 10。

表 10 人机交互模块概要

对象名	类型	说明	备注
UserInput	接口	用户输入消息定义。	见D.2.1
NgbKeyListener	接口	按键事件监听器接口，由需要监听 KeyEvent 的应用程序实现。	见D.2.2
NgbMouseListener	接口	鼠标事件监听器接口，由需要监听 MouseEvent 的应用程序实现。	见D.2.3
NgbVoiceListener	接口	语音事件监听器接口，由需要监听语音识别的应用程序实现。	见D.2.4
FrontPanel	类	前面板信息显示输出控制，包括 LED 指示灯和 LED 数码管显示控制。	见D.2.5
NgbInputManager	类	输入控制管理器，用于遥控器、按键、鼠标等事件的监听接收以及注入控制。	见D.2.6
NgbVoiceManager	类	语音相关控制管理器，用于语音识别功能的控制和实现。	见D.2.7
NgbInputEvent	事件	输入事件类，是输入事件的基类。	见D.2.8
KeyEvent	事件	按键事件类，继承 NgbInputEvent 类。	见D.2.9
MouseEvent	事件	鼠标事件类，继承 NgbInputEvent 类。	见D.2.10

4.2.6 AV 设置单元

4.2.6.1 AV 设置单元概述

AV设置单元用于实现对音视频参数的获取和设置功能，包括音频输出的端口状态、声道类型、全局音量和音量状态等，以及视频输出的端口状态、窗口匹配模块、亮度、对比度、饱和度、制式和透明度等，定义了AV设置模块。Java接口详细定义见附录E。

4.2.6.2 AV设置模块

AV设置模块提供了音视频输出参数设置相关的类和方法。

AV设置模块概要见表11。

表11 AV设置模块概要

对象名	类型	说明	备注
AudioSetting	类	音频输入/输出单元的各种参数设置。	见E.2.1
VideoSetting	类	视频输入/输出单元的各种参数设置。	见E.2.2

4.2.7 媒体处理单元

4.2.7.1 媒体处理单元概述

媒体处理单元用于实现媒体播放功能，包括播放、控制、语言选择等，定义了媒体处理模块。Java接口详细定义见附录F。

4.2.7.2 媒体处理模块

媒体处理模块定义了提供所有媒体功能的接口类以及相关的音、视频信息类。媒体处理模块用于支持DVB直播、VOD点播、IP直播、IP点播以及本地播放等媒体播放功能。

媒体处理模块定义的最基本的类有：

——媒体播放器类（MediaPlayer）；

——音频、视频、字幕等轨道类（TrackInfo）；

——音视频流信息类（MediaFormat）。

媒体处理模块概要见表12。

表12 媒体处理模块概要

对象名	类型	说明	备注
MediaPlayer	类	提供所有媒体功能的接口。支持包括DVB直播、VOD点播、IP直播、IP点播以及本地播放。	见F.2.1
TrackInfo	类	描述音频、视频、字幕等轨道信息。	见F.2.2
MediaFormat	类	使用HashMap存放音视频流等信息。	见F.2.3

4.2.8 系统管理单元

4.2.8.1 系统管理单元概述

系统管理单元用于实现外设设备管理、OTA升级管理、存储管理等与系统相关的功能，定义了系统管理模块、OTA升级模块、存储管理模块。Java接口详细定义见附录G。

4.2.8.2 系统管理模块

系统管理模块提供了配置参数访问、软硬件配置信息获取、外设管理和系统操作等的类和方法。

系统管理模块概要见表13。

表 13　系统管理模块概要

对象名	类型	说明	备注
PeripheralType	接口	TVOS 所支持的外设类型常量定义接口。	见G.2.1
Peripheral	接口	外设描述接口，提供了获取外设名称、状态、类型、ID 等方法。	见G.2.2
PeripheralListener	接口	外设事件监听器，由应用程序实现。	见G.2.3
PeripheralManager	类	外设管理器，是外设管理模块的入口类。	见G.2.4
DataConfig	类	配置数据访问类，提供了访问存储于接收终端 NVM 中的配置数据的方法。	见G.2.5
HardwareInfo	类	硬件信息描述类，提供了获取接收终端硬件参数信息的方法。	见G.2.6
SoftwareInfo	类	软件信息描述类，提供了获取接收终端软件参数信息的方法。	见G.2.7
SysTool	类	系统工具类，提供了系统待机、休眠以及重启等操作的方法。	见G.2.8
PeripheralEvent	事件	外设通知事件。	见G.2.9

4.2.8.3　OTA 升级模块

OTA 升级模块提供了 OTA 软件升级侦测与处理操作的类和方法。

OTA 升级模块概要见表 14。

表 14　OTA 升级模块概要

对象名	类型	说明	备注
OTAEventListener	接口	OTA 事件监听器，由应用程序实现。	见G.3.1
OTAManager	类	OTA 管理器，是 OTA 功能模块的入口类。	见G.3.2
OTAEvent	事件	OTA 事件。	见G.3.3

4.2.8.4　存储管理模块

存储管理模块提供了存储设备以及存储设备分区访问的类和方法。

存储管理模块概要见表 15。

表 15　存储管理模块概要

对象名	类型	说明	备注
Storage	接口	描述了存储设备信息，例如名称、大小、空闲状态、分区等。	见G.4.1
StorageEventListener	接口	存储事件监听器，由应用程序实现。	见G.4.2
StoragePartition	接口	描述了存储设备的分区信息，例如名称、大小、空闲状态、访问路径、分区类型等。	见G.4.3
StorageManager	类	提供了管理存储设备和存储设备分区的方法。	见G.4.4
StorageEvent	事件	与存储设备相关的存储事件。	见G.4.5

4.2.9　应用引擎单元

4.2.9.1　应用引擎单元概述

应用引擎单元用于实现频道搜索、电子节目指南获取、信息搜索等功能，定义了频道搜索模块、电子节目指南模块、信息搜索模块。Java接口详细定义见附录H。

4.2.9.2 频道搜索模块

频道搜索模块提供了与频道搜索相关的类和方法。

频道搜索模块概要见表 16。

表 16 频道搜索模块概要

对象名	类型	说明	备注
ChannelScanListener	接口	搜台过程事件的监听器，由应用程序实现。	见H.2.1
ChannelScanEngine	类	搜索引擎。频道搜索功能单元的入口类。	见H.2.2
ChannelScanEvent	事件	频道搜索事件，基类。	见H.2.3
ChannelScanFailureEvent	事件	频道搜索失败事件，继承 ChannelScanEvent 类。	见H.2.4
ChannelScanFinishEvent	事件	频道搜索结束事件，继承 ChannelScanEvent 类。	见H.2.5
ChannelScanNITSuccessEvent	事件	频道搜索成功解析 NIT 事件，继承 ChannelScanEvent 类。	见H.2.6
ChannelScanSuccessEvent	事件	频道搜索成功事件，继承 ChannelScanEvent 类。	见H.2.7

4.2.9.3 电子节目指南模块

电子节目指南（EPG）为终端用户提供了浏览广播业务相关信息的方法，例如业务名称、节目起止时间、内容梗概等，便于终端用户对业务进行快速检索和访问，该部分遵循 GB/T 28160—2011 的 EPG 节目信息的解析和呈现。

电子节目指南模块提供了获取 EPG 信息的类和方法。EPG 信息获取可采用缓存（Cache）机制，也可以在需要时临时装载。若采用缓存机制，应对 EPG 信息进行实时监控，以保证应用能提取到最新 EPG 信息。

电子节目指南模块概要见表17。

表 17 电子节目指南模块概要

对象名	类型	说明	备注
ProgramEvent	接口	描述了某个节目事件信息。	见H.3.1
ProgramEventFilter	接口	定义了应用从 EPG 模块查询节目事件信息所用的过滤器接口，由应用层实现。	见H.3.2
ProgramService	接口	描述了某个节目业务信息，是一组属于同一个业务的节目事件信息的封装。	见H.3.3
ProgramServiceFilter	接口	定义了应用从 EPG 模块查询节目业务信息所用的过滤器接口，由应用层实现。	见H.3.4
EPGUpdateListener	接口	EPG 信息更新事件监听器，由应用程序实现。	见H.3.5
EPGManager	类	EPG管理器，EPG信息获取的入口类。	见H.3.6
EPGUpdateEvent	事件	EPG信息更新事件。	见H.3.7

4.2.9.4 信息搜索模块

信息搜索模块提供了与全局搜索和匹配搜索相关的类和方法。“全局搜索”和“匹配搜索”的术语定义如下：

全局搜索 – 根据用户设置的搜索条件查找SI、PVR等内容，并返回有意义的搜索结果，用以提升用户体验；

匹配搜索 – 根据用户输入给出目前数据源中可以匹配的字符串，以缩短用户输入查询关键字的时间和降低用户输入查询关键字的难度。

信息搜索模块分为以下组件：

——搜索管理器(SearchManager) – 信息搜索模块的入口类；

——全局搜索会话(GlobalSearchSession) – 与全局搜索过程相关联的会话；

——匹配搜索会话(AutoCompleteSearchSession) – 与正在进行的匹配搜索过程相关联的会话。

信息搜索模块概要见表 18。

表 18　信息搜索模块概要

对象名	类型	说明	备注
AutoCompleteSearchListener	接口	匹配搜索事件监听器，由应用程序实现。	见H.4.1
AutoCompleteSearchResultItem	接口	描述了匹配搜索结果，提供了获取匹配搜索结果各种信息的方法。	见H.4.2
AutoCompleteSearchResultList	接口	描述了匹配搜索结果列表对象，提供了存取匹配搜索结果条目的方法。	见H.4.3
AutoCompleteSearchSession	接口	描述了一个匹配搜索会话,提供了匹配搜索会话控制方法。	见H.4.4
GlobalSearchListener	接口	全局搜索事件监听器，由应用程序实现。	见H.4.5
GlobalSearchResultItem	接口	描述了一个全局搜索结果对象，提供了访问搜索结果相关信息的方法。	见H.4.6
GlobalSearchResultList	接口	描述了全局搜索结果列表对象，提供了获取全局搜索结果条目的方法。	见H.4.7
GlobalSearchSession	接口	描述了一个全局搜索会话,提供了全局搜索会话控制方法。	见H.4.8
RetrieveDirection	接口	定义了搜索结果查找方向常量。	见H.4.9
SearchContentType	接口	内容类型常量定义。	见H.4.10
SearchCriteriaFlags	接口	过滤条件标记定义。	见H.4.11
SearchFields	接口	提供了设置搜索字段的接口。	见H.4.12
SearchHistoryItem	接口	描述了一条搜索历史记录，提供了获取搜索历史记录各种信息的方法。	见H.4.13
SearchHistoryList	接口	描述了搜索历史记录列表，提供了遍历搜索历史记录列表的功能。	见H.4.14
SearchStatus	接口	定义了搜索状态常量。	见H.4.15
SourceType	接口	搜索数据源常量定义。	见H.4.16
AutoCompleteSearchFilter	类	描述了一个自动匹配搜索过滤器，提供了匹配搜索过滤条件设置和获取方法。	见H.4.17
GlobalSearchFilter	类	描述了一个全局搜索过滤器，提供了全局搜索过滤条件设置和获取方法。	见H.4.18
SearchManager	类	全局搜索和匹配搜索的搜索管理器，是搜索功能模块的入口类。	见H.4.19
SortCriteria	类	定义了排序常量和排序方法。	见H.4.20
SearchException	异常	描述了全局搜索出现错误时的异常。	见H.4.21

4.2.10 多屏互动单元

4.2.10.1 多屏互动单元概述

多屏互动单元用于实现多屏互动局域网接口，定义了多屏互动模块。Java接口详细定义见附录I。

4.2.10.2 多屏互动模块

多屏互动模块可实现应用客户端在局域网内发现、连接、控制服务端设备功能。
多屏互动模块概要见表19。

表 19 多屏互动模块概要

对象名	类型	说明	备注
IMultiScreenService	接口	局域网环境中用于支撑多屏互动组件服务的功能接口。	见I.2.1
IMultiScreenCallBack	接口	多屏互动组件对外提供的远程访问接口。	见I.2.2

4.2.11 DRM 管理单元

4.2.11.1 DRM 管理单元概述

DRM管理单元用于实现DRM管理功能，定义了DRM管理模块。Java接口详细定义详见附录J。

4.2.11.2 DRM 管理模块

DRM管理模块概要见表20。

表 20 DRM 管理模块概要

对象名	类型	说明	备注
ChinaDrmManager	类	DRM模块类。	见J.2.1
ChinaDrmTeeRetVal	类	ChinaDrmManager类的ChinaDrm_SendCommandToTEE方法的返回类型类。	见J.2.2

4.2.12 DCAS 管理单元

4.2.12.1 DCAS 管理单元概述

DCAS管理单元用于实现DCAS应用相关的注册、过滤器设置、与TApp通讯等功能，定义了CAS解扰模块、CAS控制模块、CAS消息模块、CAS监听模块。该单元所定义的接口支持GY/T 255—2012。Java接口详细定义见附录K。

4.2.12.2 CAS 解扰模块

CAS解扰模块提供了DCAS终端软件平台解扰的应用程序编程接口。
CAS解扰模块概要见表21。

表 21　CAS 解扰模块概要

对象名	类型	说明	备注
CASModule	接口	用于表示请求解扰一组基本流的CASModule对象。	见K.2.1
CASDataUtils	接口	用来获取和设置CA信息，读写DCAS数据。	见K.2.2
CADescriptor	接口	提供了CA描述符的信息，给定业务的PMT中的可能提供CA描述符。此外CAT中也会有CA描述符的出现。	见K.2.3
CAServiceComponentInfo	接口	用于提取特定CA业务成分的信息，如ECM PID和用于装载控制字的DescramblerContext。	见K.2.4
CASPacketListener	接口	DCAS应用通过本接口来接收带外CAS Packets （如 EMMs)。	见K.2.5
CASSession	接口	提供CAS会话相关信息。	见K.2.6
CAStatus	接口	DCAS应用在每次DescramblerContext中的解扰状态发生变化时发送CAStatus，本状态用来指示解扰成功与否。如果所解扰成分中任何一个发生解扰失败，本状态必须汇报整个业务的解扰请求失败。当终端软件平台收到一个新的CAStatus,它应通过本段描述的CAS事件通知其他应用。	见K.2.7
CATListener	接口	DCAS应用需实现该接口,使用CAT中的CA描述符来过滤带内EMM。	见K.2.8
CATNotifier	接口	DCAS应用使用本方法注册用于获取CAT更新通知的监听器。	见K.2.9
CASModuleManager	类	用来注册所有被DCAS应用实现的CASModule。	见K.2.10
CASPermission	类	任一DCAS应用必须获取CASPersmission方可访问CASModuleManager，本机制用于确保只有被网络运营商授权的DCAS应用方可使用DCAS API。	见K.2.11

4.2.12.3　CAS 控制模块

CAS 控制模块提供了DCAS终端软件平台CAS 控制的应用程序编程接口。

CAS 控制模块概要见表 22。

表 22　CAS 控制模块概要

对象名	类型	说明	备注
DescramblerContext	接口	用来控制终端安全芯片解扰功能的组件。可以实例化多个DescramblerContext来使用不同密钥解扰多个码流的情形。	见K.3.1
ChipController	接口	用来控制终端安全芯片执行的组件。	见K.3.2
Key	类	一个基本密码密钥。	见K.3.3
CWKey	类	解扰密钥或控制字。	见K.3.4
CASTEEManager	类	与TEE中TA通信的功能接口。	见K.3.5

4.2.12.4　CAS 消息模块

CAS 消息模块提供了DCAS 消息监听的功能接口。

CAS 消息模块概要见表 23。

表 23　CAS 消息模块概要

对象名	类型	说明	备注
CASEventListener	接口	应被需要接收CAS事件的应用实现。	见K.4.1
CASAppInfo	接口	提供DCAS应用的信息。	见K.4.2
CASEventInfo	接口	提供CASEvent的信息。	见K.4.3
CASEventManager	类	应用使用CASEventManager注册监听器，来获取CAS事件。	见K.4.4

4.2.12.5　CAS 监听模块

CAS 监听模块提供了DCAS可分离安全设备的 CAS 监听的应用程序编程接口。

CAS 监听模块概要见表 24。

表 24　CAS 监听模块概要

对象名	类型	说明	备注
DetachableSecurityDevice	接口	用于应用注册可分离安全设备的监听器,来获取设备插拔状态。	见K.5.1
DetachableSecurityDeviceListener	接口	可分离安全设备的监听器,应被需要监听设备插拔状态的应用实现。	见K.5.2

4.3　TVOS WEB 应用程序编程接口

4.3.1　TVOS WEB 应用程序编程接口概述

TVOS WEB应用程序编程接口是对组件层各功能组件模块接口的JS封装，以JS对象的方式向WEB应用提供调用接口，支撑应用实现电子节目指南、频道列表、电视节目播放等数字电视相关的业务功能。WEB应用程序编程接口共由14个功能单元组成，包括单向广播网络接入单元、广播协议处理单元、双向宽带网络接入单元、人机交互单元、AV设置单元、媒体处理单元、应用管理单元、系统管理单元、消息管理单元、应用引擎单元、广播信息服务管理单元、多屏互动单元、DRM管理单元和DCAS管理单元。

TVOS WEB应用程序编程接口的定义符合ECMA-262的要求。

4.3.2　单向广播网络接入单元

4.3.2.1　单向广播网络接入单元概述

单向广播网络接入单元用于实现数字电视下的单向广播网络接入功能，包括调谐的频率、调制方式、符号率等参数控制及信号强度和质量等信息的获取，定义了调谐解调模块。JS接口详细定义见附录L。

4.3.2.2　调谐解调模块

调谐解调模块概要见表25

表 25　调谐解调模块概要

对象	说明	备注
DvbcTuningParameters	DVB-C 调谐解调参数对象。	见 L.2.3
AbsssTuningParameters	ABS-SS 调谐解调参数对象。	见 L.2.4
DtmbTuningParameters	DTMB 调谐解调参数对象。	见 L.2.5
DvbTune	频道调谐和解调对象。	见 L.2.6
DvbTunerInfo	Tunerid 与 Tuner 类型匹配对象。	见 L.2.7
DvbScan	频道搜索对象。	见 L.2.8

4.3.3　广播协议处理单元

4.3.3.1　广播协议处理单元概述

广播协议处理单元用于实现广播协议的处理，定义了DVB协议处理模块。JS接口详细定义见附录M。

4.3.3.2　DVB 协议处理模块

DVB协议处理模块概要见表26。

表 26　DVB 协议处理模块概要

对象	说明	备注
DvbBroadcast	DVB 广播通道对象。	见 M.2.3
DvbNetwork	DVB 网络对象。	见 M.2.4
DvbBouquet	DVB 业务群对象。	见 M.2.5
DvbTS	DVB 传送流对象。	见 M.2.6
DvbService	DVB 业务对象。	见 M.2.7
DvbVideoES	DVB 视频基本流对象。	见 M.2.8
DvbAudioES	DVB 音频基本流对象。	见 M.2.9
DvbOtherES	除视音频之外的其他 DVB 基本流对象。	见 M.2.10
DvbEvent	DVB 节目事件对象。	见 M.2.11

4.3.4　双向宽带网络接入单元

4.3.4.1　双向宽带网络接入单元概述

双向宽带网络接入单元用于实现与双向宽带网络接入控制相关的功能，包括双向网络的连接管理和数据操作等，定义了双向宽带网络设置模块。JS接口详细定义见附录N。

4.3.4.2　双向宽带网络设置模块

双向宽带网络设置模块概要见表27。

表 27　双向宽带网络设置模块概要

对象	说明	备注
Broadband	双向宽带网络对象。	见 N.2.2
Ethernet	以太网卡对象。	见 N.2.3
AP	无线热点对象。	见 N.2.4
IP	IP 对象。	见 N.2.5
Proxy	网络代理对象。	见 N.2.6

4.3.5　人机交互单元

4.3.5.1　人机交互单元概述

人机交互单元用于实现设备输入控制以及前面板显示控制功能，输入把遥控器、鼠标、键盘、前面板按键等输入设备发送的用户指令封装成按键消息，输出通过前面板或显示屏幕反馈信息，定义了用户输入单元模块和前面板输出模块。JS接口详细定义见附录O。

4.3.5.2　用户输入单元模块

用户输入单元模块概要见表28。

表 28　用户输入单元模块概要

对象	说明	备注
event	用户输入消息管理对象。	见 O.2.2

4.3.5.3　前面板输出模块

前面板输出模块概要见表29。

表 29　前面板输出模块概要

对象	说明	备注
FrontPanel	前面板对象。	见 O.3.1

4.3.6　AV 设置单元

4.3.6.1　AV 设置单元概述

AV设置单元用于实现对音视频参数的获取和设置功能，包括音频输出的端口状态、声道类型、全局音量和音量状态等，以及视频输出的端口状态、窗口匹配模块、亮度、对比度、饱和度、制式和透明度等，定义了音视频参数设置模块。JS接口详细定义见附录P。

4.3.6.2　音视频参数设置模块

音视频参数设置模块概要见表30。

表 30　音视频参数设置模块概要

对象	说明	备注
AudioSetting	音频参数设置对象。	见 P.2.1
VideoSetting	视频参数设置对象。	见 P.2.2

4.3.7　媒体处理单元

4.3.7.1　媒体处理单元概述

媒体处理单元用于实现媒体播放功能，包括播放、控制、事件处理、异常处理等，定义了媒体处理模块。JS接口详细定义见附录Q。

4.3.7.2　媒体处理模块

媒体处理模块概要见表31。

表 31　媒体处理模块概要

对象	说明	备注
MediaPlayer	媒体播放器对象。	见 Q.2.2

4.3.8　应用管理单元

4.3.8.1　应用管理单元概述

应用管理单元用于实现WEB应用的管理功能，包括应用查询、应用安装、应用卸载、应用更新等，定义了应用管理模块。JS接口详细定义见附录R。

4.3.8.2　应用管理模块

应用管理模块概要见表32。

表 32　应用管理模块概要

对象	说明	备注
widget	应用管理对象。	见 R.2.2

4.3.9　系统管理单元

4.3.9.1　系统管理单元概述

系统管理单元用于实现外设设备管理、OTA升级管理、存储管理等与系统相关的功能，定义了数据管理模块、外部存储设备管理模块、文件管理模块、多媒体文件模块、OTA软件升级模块、系统工具模块和软硬件信息模块。JS接口详细定义见附录S。

4.3.9.2　数据管理模块

数据管理模块概要见表33。

表 33　数据管理模块概要

对象	说明	备注
DataConfig	配置数据管理对象。	见 S.2.2

4.3.9.3　外部存储设备管理模块

外部存储设备管理模块概要见表34。

表 34　外部存储设备管理模块概要

对象	说明	备注
StorageDeviceManager	外部存储设备管理器对象。	见 S.3.2
StorageDevice	外部存储设备对象。	见 S.3.3
StoragePartition	外部存储分区对象。	见 S.3.4

4.3.9.4　文件管理模块

文件管理模块概要见表35表。

表 35　文件管理模块概要

对象	说明	备注
FileManager	目录和文件管理对象。	见 S.4.2
Directory	目录对象。	见 S.4.3
File	文件对象。	见 S.4.4

4.3.9.5　多媒体文件模块

多媒体文件模块概要见表36。

表 36　多媒体文件模块概要

对象	说明	备注
AudioFile	音频文件对象。	见 S.5.1
VideoFile	视频文件对象。	见 S.5.2
ImageFile	图片文件对象。	见 S.5.3

4.3.9.6　OTA 软件升级模块

OTA软件升级模块概要见表37。

表 37　OTA 软件升级模块概要

对象	说明	备注
Upgrade	OTA 升级控制对象。	见 S.6.2

4.3.9.7 系统工具模块

系统工具模块概要见表38。

表 38 系统工具模块概要

对象	说明	备注
Utility	用于完成事件消息的获取、字符串的打印等的工具对象。	见 S.7.1
GlobalVarManager	全局变量管理器对象。	见 S.7.2
Rectangle	矩形窗口对象。	见 S.7.3
SysTool	系统工具对象。	见 S.7.4

4.3.9.8 软硬件信息模块

软硬件信息模块概要见表39。

表 39 软硬件信息模块概要

对象	说明	备注
HardwareInfo	接收终端硬件参数信息对象。	见 S.8.1
SoftwareInfo	接收终端软件参数信息对象。	见 S.8.2

4.3.10 消息管理单元

4.3.10.1 消息管理单元概述

消息管理单元用于实现对消息的接收管理功能，包含按键消息和系统消息。该单元定义了消息管理模块。JS接口详细定义见附录T。

4.3.10.2 消息管理模块

消息管理模块概要见表40。

表 40 消息管理模块概要

对象	说明	备注
event	消息对象。	见 T.2.1

4.3.11 应用引擎单元

4.3.11.1 应用引擎单元概述

应用引擎单元用于实现频道管理、电子节目指南获取、预定提醒管理、信息搜索等功能，定义了电子节目指南模块、频道管理模块、预定提醒模块和信息搜索模块。JS接口详细定义见附录U。

4.3.11.2 频道管理模块

频道管理模块概要见表41。

表 41　频道管理模块概要

对象	说明	备注
ChannelManager	频道管理器对象。	见 U.2.2
Channel	频道对象。	见 U.2.3

4.3.11.3　电子节目指南模块

电子节目指南模块概要见表42表。

表 42　电子节目指南模块概要

对象	说明	备注
EPGManager	电子节目指南管理器对象。	见 U.3.2
ProgramEvent	节目事件对象。	见 U.3.3
ReferenceEvent	参考节目事件对象。	见 U.3.4
TimeShiftEvent	时移节目事件对象。	见 U.3.5

4.3.11.4　预定提醒模块

预定提醒模块概要见表43。

表 43　预定提醒模块概要

对象	说明	备注
OrderManager	预定管理器对象。	见 U.4.2
Order	预订对象。	见 U.4.3

4.3.11.5　信息搜索模块

信息搜索模块概要见表44。

表 44　信息搜索模块概要

对象	说明	备注
SearchManager	搜索管理器对象。	见 U.5.2
GlobalSearchSession	全局搜索会话对象。	见 U.5.3
AutoCompleteSearchSession	匹配搜索会话对象。	见 U.5.4
GlobalSearchFilter	全局搜索过滤器对象。	见 U.5.5
AutoCompleteSearchFilter	匹配搜索过滤器对象。	见 U.5.6
SortCritiria	排序机制对象。	见 U.5.7
GlobalSearchResultItem	搜索结果记录对象。	见 U.5.8
SearchHistoryItem	搜索历史记录对象。	见 U.5.9

4.3.12 广播信息服务管理单元

4.3.12.1 广播信息服务管理单元概述

广播信息服务管理单元用于实现广播信息服务的检测、接收和处理，支撑应急广播、信息服务、广告、OSD文本更新等相关业务。JS接口详细定义见附录V。

4.3.12.2 广播信息服务管理模块

广播信息服务管理模块实现监控BAT表、NIT表和过滤广播信息服务的Section段数据，同时通过DCAS管理单元相关接口获取终端对应的CA用户标识，并以此为判据，实现终端应急广播和广告等信息的精准接收。

广播信息服务管理模块概要见表45。

表 45 广播信息服务管理模块概要

对象	说明	备注
DthManager	广播信息服务管理对象。	见 V.2.2
Ad	广告对象。	见 V.2.3
AdService	广告服务对象。	见 V.2.4

4.3.13 多屏互动单元

4.3.13.1 多屏互动单元概述

多屏互动单元用于实现多屏互动局域网接口，定义了多屏互动模块。JS接口详细定义见附录W。

4.3.13.2 多屏互动模块

多屏互动模块概要见表46。

表 46 多屏互动模块概要

对象	说明	备注
MultiScreen	多屏互动控制对象。	见 W.2.1

4.3.14 DRM 管理单元

4.3.14.1 DRM 管理单元概述

DRM管理单元用于实现DRM管理功能，定义了DRM管理模块。JS接口详细定义见附录X。

4.3.14.2 DRM 管理模块

DRM管理模块概要见表47。

表 47 DRM 管理模块概要

对象	说明	备注
ChinaDrmManager	DRM 管理对象。	见 X.2.1

4.3.15 DCAS 管理单元

4.3.15.1 DCAS 管理单元概述

DCAS管理单元用于实现DCAS应用相关的注册、过滤器设置、与TApp通讯等功能，定义了EPG_DCAS模块和DCAS_APP模块。JS接口详细定义见附录Y。

4.3.15.2 EPG_DCAS 模块

EPG_DCAS模块概要见表48。

表 48 EPG_DCAS 模块概要

对象	说明	备注
EPG_DCAS	DCAS 信息返回对象。	见 Y.2.1

4.3.15.3 DCAS_APP 模块

DCAS_APP模块概要见表49。

表 49 DCAS_APP 模块概要

对象	说明	备注
JSDCAS.CASDescriptor	CAS 描述符对象。	见 Y.3.2
JSDCAS.CASEcmEvent	Ecm 消息对象。	见 Y.3.3
JSDCAS.CASEmmEvent	Emm 消息对象。	见 Y.3.4
JSDCAS.CASFilter	CAS 过滤器对象。	见 Y.3.5
JSDCAS.CASM	CAS 全局对象。	见 Y.3.6
JSDCAS.CASModule	CAS 模块对象。	见 Y.3.7
JSDCAS.CASModuleManager	CAS 模块管理器对象。	见 Y.3.8
JSDCAS.CASPacketEvent	CAS 数据包消息对象。	见 Y.3.9
JSDCAS.CASSession	CAS 解扰请求对象。	见 Y.3.10
JSDCAS.CASStatus	CAS 状态信息对象。	见 Y.3.11
JSDCAS.TeeController	CAS 与 TEE 通讯控制器对象。	见 Y.3.12
JSDCAS.TeeRetVal	CAS 与 TEE 通讯返回值对象。	见 Y.3.13

5 调用机制

TVOS应用框架层由JAVA应用框架和WEB应用框架两部分组成，JAVA应用框架和WEB应用框架分别形成了各自的应用程序编程接口供应用层的JAVA应用和WEB应用调用，使得这些应用完成各自的功能。

TVOS应用程序编程接口调用模型如图1所示。

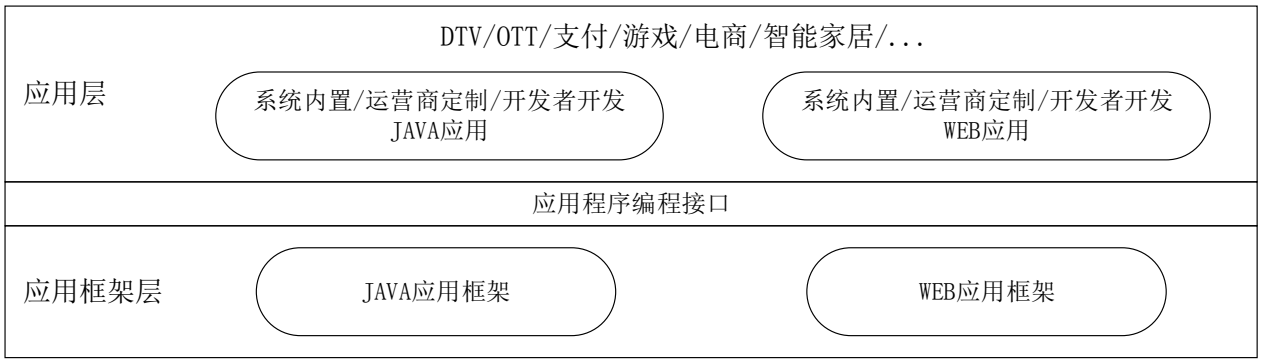


图 1　TVOS 应用程序编程接口调用模型

JAVA 应用在开发时通过引入包含了 TVOS JAVA 应用程序编程接口的 Jar 包来使用 TVOS JAVA 应用程序编程接口实现相应的功能。

WEB 应用在开发时通过引入 TVOS WEB 应用程序编程接口提供的各种内置 JavaScript 对象来实现相应的功能。

附 录 A
（规范性附录）
JAVA-单向广播网络接入单元

A.1 概述

本附录定义了单向广播网络接入单元的JAVA接口，主要是调谐解调模块，实现了调谐解调功能。

A.2 调谐解调模块

调谐解调模块定义了与调谐解调相关接口、类和异常。
调谐解调模块概要见表A.1。

表 A.1 调谐解调模块概要

接口	
DeliverySystemType	DVB技术体系下的传送系统类型常量定义。
TuningParameters	调谐解调参数接口。
TuningListener	网络接口事件监听器接口，提供了网络接口相关事件处置的回调方法，由应用层实现。
类	
DvbcTuningParameters	适用于DVB-C传送系统的调谐解调参数类。
AbsssTuningParameters	适用于ABS-SS传送系统的调谐解调参数类。
DtmbTuningParameters	适用于DTMB传送系统的调谐解调参数类。
TunerEvent	调谐解调器事件类，网络接口相关事件的基类。
TunerTuningEvent	网络接口开始调谐事件，继承TunerEvent类。
TunerTuningOverEvent	结束调谐事件，继承TunerEvent类。
Tuner	调谐解调控制接口。
TunerManager	调谐解调管理器类，用于跟踪连接到接收设备的广播网络接口，是调谐解调单元的入口类。
异常	
TunerException	网络接口异常，其他网络异常的基类。
IncorrectLocatorException	定位符格式不正确异常，继承TunerException类。
StreamNotFoundException	流未发现异常，继承TunerException类。对该传送流的引用无法被解析时，则抛出此异常。
TuningParameterNotFoundException	获取当前调谐解调参数失败或者DeliverySystemType 错误的时候，抛出异常。

A.2.1 接口org.ngb.broadcast.dvb.tuner.DeliverySystemType

原型：public interface org.ngb.broadcast.dvb.tuner.DeliverySystemType

描述：DVB技术体系下传送系统类型常量定义，包括了DVB-C、DTMB和ABS-SS这三种传送系统类型常量定义。

A.2.1.1 常量域——传送系统类型

A.2.1.1.1 SATELLITE_DELIVERY_SYSTEM

原型：public final static int SATELLITE_DELIVERY_SYSTEM = 0
描述：传送系统类型——ABS-SS。

A.2.1.1.2 CABLE_DELIVERY_SYSTEM

原型：public final static int CABLE_DELIVERY_SYSTEM = 1
描述：传送系统类型——DVB-C。

A.2.1.1.3 TERRESTRIAL_DELIVERY_SYSTEM

原型：public final static int TERRESTRIAL_DELIVERY_SYSTEM = 2
描述：传送系统类型——DTMB（即国标地面数字电视）。

A.2.1.1.4 UNKNOWN_DELIVERY_SYSTEM

原型：public final static int UNKNOWN_DELIVERY_SYSTEM = 0xFF
描述：未知的错误类型。

A.2.2 接口org.ngb.broadcast.dvb.tuner.TuningParameters

原型：public interface org.ngb.broadcast.dvb.tuner.TuningParameters
描述：调谐解调参数接口。

A.2.2.1 方法

A.2.2.1.1 getDeliverySystemType

原型：public int getDeliverySystemType()
描述：获取调谐解调参数适用的传送系统类型。
参数：无。
返回：int型，表示传送系统类型，取值详见org.ngb.broadcast.dvb.tuner.DeliverySystemType接口以及“传送系统类型”常量域定义。

A.2.3 接口org.ngb.broadcast.dvb.tuner.TuningListener

原型：public interface org.ngb.broadcast.dvb.tuner.TuningListener
描述：网络接口事件监听器接口，提供了网络接口相关事件处置的回调方法，由应用层实现。

A.2.3.1 方法

A.2.3.1.1 receiveNIEvent

原型：public abstract void receiveNIEvent(org.ngb.broadcast.dvb.tuner.TunerEvent anEvent)
描述：网络接口事件回调方法。
参数：anEvent - org.ngb.broadcast.dvb.tuner.TunerEvent对象，表示一个网络接口事件。
返回：无。

A.2.4 类org.ngb.broadcast.dvb.tuner.DvbcTuningParameters

原型：public class org.ngb.broadcast.dvb.tuner.DvbcTuningParameters implements org.ngb.broadcast.dvb.tuner.TuningParameters

描述：适用于DVB-C传输系统的调谐解调参数类。getDeliverySystemType()方法返回值固定为DeliverySystemType.CABLE_DELIVERY_SYSTEM。

A.2.4.1 常量域——调制方式

A.2.4.1.1 DVB_C_MOD_UNDEFINED

原型：public static final int DVB_C_MOD_UNDEFINED = 0

描述：DVB-C调制方式——未定义。

A.2.4.1.2 DVB_C_MOD_QAM16

原型：public static final int DVB_C_MOD_QAM16 = 1

描述：DVB-C调制方式——16QAM。

A.2.4.1.3 DVB_C_MOD_QAM32

原型：public static final int DVB_C_MOD_QAM32 = 2

描述：DVB-C调制方式——32QAM。

A.2.4.1.4 DVB_C_MOD_QAM64

原型：public static final int DVB_C_MOD_QAM64 = 3

描述：DVB-C调制方式——64QAM。

A.2.4.1.5 DVB_C_MOD_QAM128

原型：public static final int DVB_C_MOD_QAM128 = 4

描述：DVB-C调制方式——128QAM。

A.2.4.1.6 DVB_C_MOD_QAM256

原型：public static final int DVB_C_MOD_QAM256 = 5

描述：DVB-C调制方式——256QAM。

A.2.4.2 方法

A.2.4.2.1 DvbcTuningParameters

原型：public DvbcTuningParameters()

描述：构造方法，创建一个默认的DVB-C调谐解调参数对象。

参数：无。

A.2.4.2.2 DvbcTuningParameters

原型：public DvbcTuningParameters(int frequency, int modulation, int symbolRate)

描述：构造方法，根据输入参数创建一个DVB-C调谐解调参数对象。

参数：frequency - int型，表示DVB-C信号中心频率，单位为千赫（kHz）；

modulation - int型，表示DVB-C信号调制方式，取值详见
org.ngb.broadcast.dvb.tuner.DvbcTuningParameters类的“调制方式”常量域定义；
symbolRate - int型，表示DVB-C信号符号率，单位为千符号每秒（ksymbol/s）。

A.2.4.2.3 setFrequency

原型：public void setFrequency(int frequency)
描述：设置DVB-C信号调谐频率。
参数：frequency - int型，表示DVB-C信号中心频率，单位为千赫（kHz）。
返回：无。

A.2.4.2.4 getFrequency

原型：public int getFrequency()
描述：获取DVB-C信号调谐频率。
参数：无。
返回：int型，表示DVB-C信号调谐频率，单位为千赫（kHz）。

A.2.4.2.5 setModulation

原型：public void setModulation(int modulation)
描述：设置调制方式。
参数：modulation - int型，表示DVB-C信号调制方式，
取值详见org.ngb.broadcast.dvb.tuner.DvbcTuningParameters类的“调制方式”常量域定义。
返回：无。

A.2.4.2.6 getModulation

原型：public int getModulation()
描述：获取调制方式。
参数：无。
返回：int型，表示DVB-C信号调制方式，
取值详见org.ngb.broadcast.dvb.tuner.DvbcTuningParameters类的“调制方式”常量域定义。

A.2.4.2.7 setSymbolRate

原型：public void setSymbolRate(int symbolRate)
描述：设置DVB-C信号符号率。
参数：symbolRate - int型，表示DVB-C信号符号率，单位为千符号每秒（ksymbol/s）。
返回：无。

A.2.4.2.8 getSymbolRate

原型：public int getSymbolRate()
描述：获取DVB-C信号符号率。
参数：无。
返回：int型，表示DVB-C信号符号率，单位为千符号每秒（ksymbol/s）。

A.2.5 类org.ngb.broadcast.dvb.tuner.AbsssTuningParameters

原型：public class org.ngb.broadcast.dvb.tuner.AbsssTuningParameters implements org.ngb.broadcast.dvb.tuner.TuningParameters

描述：适用于ABS-SS传送系统的调谐参数类。getDeliverySystemType()方法返回值固定为SATELLITE_DELIVERY_SYSTEM。

A.2.5.1 常量域——极化方式

A.2.5.1.1 ABS_SS_POLAR_LINEAR_H

原型：public static final int ABS_SS_POLAR_LINEAR_H = 0

描述：ABS-SS极化方式——水平线极化。

A.2.5.1.2 ABS_SS_POLAR_LINEAR_V

原型：public static final int ABS_SS_POLAR_LINEAR_V = 1

描述：ABS-SS极化方式——垂直线极化。

A.2.5.1.3 ABS_SS_POLAR_CIRCULAR_L

原型：public static final int ABS_SS_POLAR_CIRCULAR_L = 2

描述：ABS-SS极化方式——左旋圆极化。

A.2.5.1.4 ABS_SS_POLAR_CIRCULAR_R

原型：public static final int ABS_SS_POLAR_CIRCULAR_R = 3

描述：ABS-SS极化方式——右旋圆极化。

A.2.5.2 方法

A.2.5.2.1 AbsssTuningParameters

原型：public AbsssTuningParameters()

描述：构造方法，创建一个默认的ABS-SS调谐解调参数对象。

参数：无。

A.2.5.2.2 AbsssTuningParameters

原型：public AbsssTuningParameters(int frequency, int symbolRate, int polarization)

描述：构造方法，根据输入参数创建一个ABS-SS调谐解调参数对象。

参数：frequency - int型，表示ABS-SS信号中心频率，单位为兆赫（MHz）；
symbolRate - int型，表示ABS-SS信号符号率，单位为千符号每秒（ksymbol/s）；
polarization - int型，表示ABS-SS信号极化方式。

A.2.5.2.3 setFrequency

原型：public void setFrequency(int freq)

描述：设置ABS-SS信号中心频率。

参数：freq - int型，表示ABS-SS信号中心频率，单位为兆赫（MHz）。

返回：无。

A.2.5.2.4 getFrequency

原型：public int getFrequency()
描述：获取ABS-SS信号中心频率。
参数：无。
返回：int型，表示ABS-SS信号中心频率，单位为兆赫（MHz）。

A.2.5.2.5 setPolarization

原型：public void setPolarization (int polarization)
描述：设置极化方式。
参数：polarization - int型，表示ABS-SS信号极化方式。
返回：无。

A.2.5.2.6 getPolarization

原型：public int getPolarization()
描述：获取ABS-SS信号极化方式。
参数：无。
返回：int型，表示ABS-SS信号极化方式。

A.2.5.2.7 setSymbolRate

原型：public void setSymbolRate(int symbolRate)
描述：设置ABS-SS信号符号率。
参数：symbolRate - int型，表示ABS-SS信号符号率，单位为千符号每秒（ksymbol/s）。
返回：无。

A.2.5.2.8 getSymbolRate

原型：public int getSymbolRate()
描述：获取ABS-SS信号符号率。
参数：无。
返回：int型，表示ABS-SS信号符号率，单位为千符号每秒（ksymbol/s）。

A.2.6 类org.ngb.broadcast.dvb.tuner.DtmbTuningParameters

原型：public class org.ngb.broadcast.dvb.tuner.DtmbTuningParameters implements org.ngb.broadcast.dvb.tuner.TuningParameters
描述：适用于DTMB传输系统的调谐解调参数类。getDeliverySystemType()方法返回值固定为TERRESTRIAL_DELIVERY_SYSTEM。

A.2.6.1 常量域——调制方式

A.2.6.1.1 DTMB_MOD_UNDEFINED

原型：public static final int DTMB_MOD_UNDEFINED = 0
描述：DTMB调制方式——未定义。

A.2.6.1.2 DTMB_MOD_QAM4

原型：public static final int DTMB_MOD_QAM4 = 1
描述：DTMB调制方式——4-QAM。

A.2.6.1.3 DTMB_MOD_QAM4_NR

原型：public static final int DTMB_MOD_QAM4_NR = 2
描述：DTMB调制方式——4-QAM-NR。

A.2.6.1.4 DTMB_MOD_QAM16

原型：public static final int DTMB_MOD_QAM16 = 3
描述：DTMB调制方式——16-QAM。

A.2.6.1.5 DTMB_MOD_QAM32

原型：public static final int DTMB_MOD_QAM32 = 4
描述：DTMB调制方式——32-QAM。

A.2.6.1.6 DTMB_MOD_QAM64

原型：public static final int DTMB_MOD_QAM64 = 5
描述：DTMB调制方式——64-QAM。

A.2.6.1.7 DTMB_MOD_QAM128

原型：public static final int DTMB_MOD_QAM128 = 6
描述：DTMB调制方式——128-QAM。

A.2.6.1.8 DTMB_MOD_QAM256

原型：public static final int DTMB_MOD_QAM256 = 7
描述：DTMB调制方式——256-QAM。

A.2.6.1.9 DTMB_MOD_QAM512

原型：public static final int DTMB_MOD_QAM512 = 8
描述：DTMB调制方式——512-QAM。

A.2.6.1.10 DTMB_BANDWIDTH_6M

原型：public static final int DTMB_BANDWIDTH_6M = 6000
描述：DTMB 频率带宽6M。

A.2.6.1.11 DTMB_BANDWIDTH_7M

原型：public static final int DTMB_BANDWIDTH_7M = 7000
描述：DTMB 频率带宽7M。

A.2.6.1.12 DTMB_BANDWIDTH_8M

原型：public static final int DTMB_BANDWIDTH_8M = 8000
描述：DTMB 频率带宽8M。

A.2.6.2 方法

A.2.6.2.1 DtmbTuningParameters

原型：public DtmbTuningParameters()
描述：构造方法，创建一个默认的DTMB调谐解调参数对象。
参数：无。

A.2.6.2.2 DtmbTuningParameters

原型：public DtmbTuningParameters(int frequency, int modulation)
描述：构造方法，根据输入参数创建一个DTMB调谐解调参数对象。
参数：frequency - int型，表示DTMB信号中心频率，单位为千赫（kHz）。
modulation - int型，表示DTMB 信号调制方式，取值详见org.ngb.broadcast.dvb.tuner.DtmbTuningParameters类的“调制方式”常量域定义。

A.2.6.2.3 setFrequency

原型：public void setFrequency(int frequency)
描述：设置DTMB信号中心频率。
参数：frequency - int型，表示DTMB信号中心频率，单位为千赫（kHz）。
返回：无。

A.2.6.2.4 setModulation

原型：public void setModulation (int modulation)
描述：设置DTMB信号调制方式。
参数：modulation - int型，表示DTMB信号调制方式，取值详见org.ngb.broadcast.dvb.tuner.DtmbTuningParameters类的“调制方式”常量域定义。
返回：无。

A.2.6.2.5 setBandWidth

原型：public void setBandWidth(int bandWidth)
描述：设置DTMB 频率带宽。
参数：bandWidth - int型，表示DTMB频率带宽。
返回：无。

A.2.6.2.6 getFrequency

原型：public int getFrequency()
描述：获取DTMB信号中心频率。
参数：无。
返回：int型，表示DTMB信号中心频率，单位为千赫（kHz）。

A.2.6.2.7 getModulation

原型：public int getModulation()
描述：获取DTMB信号调制方式。在成功调谐解调后，调用此方法返回DTMB信号的调制方式，否则返回值无意义。

参数：无。

返回：int型，表示DTMB信号调制方式，取值详见
org.ngb.broadcast.dvb.tuner.DtmbTuningParameters类的“调制方式”常量域定义。

A.2.6.2.8 getBandWidth

原型：public int getBandWidth()

描述：获取DTMB 频率带宽。在成功调谐解调后，调用此方法返回DTMB信号的频率带宽，否则返回值无意义。

参数：无。

返回：int型，表示DTMB频率带宽。

A.2.6.2.9 getCodingRatio

原型：public java.lang.String getCodingRatio()

描述：获取DTMB信号编码效率。在成功调谐解调后，调用此方法返回DTMB信号的编码效率，否则返回值无意义。

参数：无。

返回：java.lang.String对象，表示DTMB信号编码效率的字符串描述，可取值为“0.4”、“0.6”或“0.8”。

A.2.6.2.10 getPNMode

原型：public java.lang.String getPNMode()

描述：获取DTMB信号帧头模式。在成功调谐解调后，调用此方法返回DTMB信号的帧头模式，否则返回值无意义。

参数：无。

返回：java.lang.String对象，表示DTMB信号帧头模式,例如“PN945”。

A.2.7 类org.ngb.broadcast.dvb.tuner.TunerEvent

原型：public abstract class org.ngb.broadcast.dvb.tuner.TunerEvent extends
java.lang.Object

描述：调谐解调器事件类，本包定义的一组网络接口相关事件的基类。

A.2.7.1 方法

A.2.7.1.1 TunerEvent

原型：public TunerEvent(java.lang.Object tuner)

描述：构造方法。

参数：tuner – java.lang.Object对象，表示发出此事件的调谐器。

A.2.7.1.2 getSource

原型：public java.lang.Object getSource()

描述：获取产生该事件的对象。

参数：无。

返回：java.lang.Object对象，返回产生该事件的对象。

A.2.8 类org.ngb.broadcast.dvb.tuner.TunerTuningEvent

原型：public class org.ngb.broadcast.dvb.tuner.TunerTuningEvent
extends org.ngb.broadcast.dvb.tuner.TunerEvent
描述：网络接口开始调谐事件，继承org.ngb.broadcast.dvb.tuner.TunerEvent类。

A.2.8.1 方法

A.2.8.1.1 TunerTuningEvent

原型：public TunerTuningEvent(Object tuner)
描述：构造方法。
参数：tuner – java.lang.Object对象，表示开始调谐的调谐器对象。

A.2.9 类org.ngb.broadcast.dvb.tuner.TunerTuningOverEvent

原型：public class org.ngb.broadcast.dvb.tuner.TunerTuningOverEvent extends
org.ngb.broadcast.dvb.tuner.TunerEvent
描述：结束调谐事件，继承org.ngb.broadcast.dvb.tuner.TunerEvent类。

A.2.9.1 常量域——调谐结果

A.2.9.1.1 SUCCEEDED

原型：public final static int SUCCEEDED = 0
描述：锁频成功。

A.2.9.1.2 FAILED

原型：public final static int FAILED = 1
描述：锁频失败。

A.2.9.2 方法

A.2.9.2.1 TunerTuningOverEvent

原型：public TunerTuningOverEvent(java.lang.Object tuner, int status)
描述：构造方法。
参数：tuner – java.lang.Object对象，表示完成调谐的网络接口；
status – int型，表示标识调谐动作是否成功的状态码。
返回：无。

A.2.9.2.2 getStatus

原型：public int getStatus()
描述：获取调谐操作结束的状态码。
参数：无。
返回：int型，表示调谐结束的状态码，可取值为SUCCEEDED或FAILED。

A.2.10 类org.ngb.broadcast.dvb.tuner.Tuner

原型：public class org.ngb.broadcast.dvb.tuner.Tuner

描述：调谐解调控制器类。

A.2.10.1 方法

A.2.10.1.1 tune

原型：public void tune(org.ngb.broadcast.dvb.tuner.TuningParameters tuningParameters)
throws java.lang.IllegalArgumentException,
org.ngb.broadcast.dvb.tuner.TunerException

描述：异步方法，根据指定的调谐解调参数进行调谐解调。
——若产生异常，网络接口状态将保持不变，不产生事件；
——若未产生异常，该方法将会向网络接口事件监听器发送
org.ngb.broadcast.dvb.tuner.TunerTuningEvent和
org.ngb.broadcast.dvb.tuner.TunerTuningOverEvent事件。

参数：tuningParameters - org.ngb.broadcast.dvb.tuner.TuningParameters对象，表示调谐参数。方法会通过instanceof方法进一步判断输入参数的类型。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入调谐解调参数无效，则抛出此异常。
org.ngb.broadcast.dvb.tuner.TunerException – 若调谐失败，则抛出此异常。

A.2.10.1.2 getSignalIntensity

原型：public int getSignalIntensity()

描述：获取当前调谐频点的信号强度（信号电平）。

参数：无。

返回：int型，表示采用百分比形式表示的信号强度，取值范围0～100，0表示信号强度最弱，100表示信号强度最强。

A.2.10.1.3 getSignalQuality

原型：public int getSignalQuality()

描述：获取当前调谐频点的信号质量（信噪比）。

参数：无。

返回：int型，表示采用百分比形式表示的信号质量，取值范围0～100，0表示信号质量最差，100表示信号质量最好。

A.2.10.1.4 getCurrentTunningParam

原型：public org.ngb.broadcast.dvb.tuner.TuningParameters getCurrentTunningParam()
throws org.ngb.broadcast.dvb.tuner.TuningParameterNotFoundException

描述：获取当前调谐频点的调谐参数

参数：无。

返回：获取当前调谐频点的调谐参数，失败或者
org.ngb.broadcast.dvb.tuner.DeliverySystemType 错误的时候，抛出异常。
成功获取后需要判断org.ngb.broadcast.dvb.tuner.TuningParameters 实例的具体类型。

A.2.10.1.5 addNetworkInterfaceListener

原型：public void addNetworkInterfaceListener(org.ngb.broadcast.dvb.tuner.TuningListener listener)
描述：注册网络接口事件监听器。
参数：listener - org.ngb.broadcast.dvb.tuner.TuningListener对象，表示待注册的网络接口事件监听器。
返回：无。

A.2.10.1.6 removeNetworkInterfaceListener

原型：public void removeNetworkInterfaceListener(org.ngb.broadcast.dvb.tuner.TuningListener listener)
描述：注销网络接口事件监听器。
参数：listener - org.ngb.broadcast.dvb.tuner.TuningListener对象，表示待注册的网络接口事件监听器。
返回：无。

A.2.10.1.7 getCurrentTransportStream

原型：public org.davic.mpeg.TransportStream getCurrentTransportStream()
描述：获取当前广播网络接口调谐到的传送流。
参数：无。
返回：org.davic.mpeg.TransportStream对象，表示当前网络接口调谐到的传送流。若当前网络接口未调谐到任何传送流，则返回null。

A.2.10.1.8 getDeliverySystemType

原型：public int getDeliverySystemType()
描述：获取当前广播网络接口传送类型。
参数：无。
返回：int型，表示当前网络接口的传送类型。

A.2.11 类org.ngb.broadcast.dvb.tuner.TunerManager

原型：public class org.ngb.broadcast.dvb.tuner.TunerManager
描述：调制解调管理器类，用于跟踪连接到接收设备的广播网络接口，是调谐解调单元的入口类。

A.2.11.1 方法

A.2.11.1.1 getInstance

原型：public static org.ngb.broadcast.dvb.tuner.TunerManager getInstance()
描述：获取系统实现的TunerManager类的唯一实例。
参数：无
返回：org.ngb.broadcast.dvb.tuner.TunerManager类单例。

A.2.11.1.2 getNetworkInterfaces

原型：public org.ngb.broadcast.dvb.tuner.Tuner[] getNetworkInterfaces()
描述：获取接收设备所连接的所有网络接口对象数组。
参数：无

返回：Tuner对象数组。若无，则返回null。

A.2.12 异常org.ngb.broadcast.dvb.tuner.TunerException

原型：public class org.ngb.broadcast.dvb.tuner.TunerException extends java.lang.Exception
描述：网络接口异常，本包定义的一组其他网络异常的基类。

A.2.13 异常org.ngb.broadcast.dvb.tuner.IncorrectLocatorException

原型：public class org.ngb.broadcast.dvb.tuner.IncorrectLocatorException extends org.ngb.broadcast.dvb.tuner.TunerException
描述：定位符格式不正确异常，继承org.ngb.broadcast.dvb.tuner.TunerException类。

A.2.14 异常org.ngb.broadcast.dvb.tuner.StreamNotFoundException

原型：public class org.ngb.broadcast.dvb.tuner.StreamNotFoundException extends org.ngb.broadcast.dvb.tuner.TunerException
描述：流未发现异常，继承org.ngb.broadcast.dvb.tuner.TunerException类。由于传送流不在StreamTable中而导致对该传送流的引用无法被解析时，则抛出此异常。

A.2.15 异常org.ngb.broadcast.dvb.tuner.TuningParameterNotFoundException

原型：public class org.ngb.broadcast.dvb.tuner.TuningParameterNotFoundException extends org.ngb.broadcast.dvb.tuner.TunerException
描述：继承org.ngb.broadcast.dvb.tuner.TunerException类。获取当前调谐解调参数失败或者org.ngb.broadcast.dvb.tuner.DeliverySystemType 错误的时候，抛出异常。

附 录 B
（规范性附录）
JAVA-广播协议处理单元

B.1 概述

本附录定义了与广播协议处理相关的JAVA接口，包括了MPEG对象定义模块、DVB对象定义模块、SECTION段过滤模块、URL封装模块、DVB定位符模块和广播协议处理模块。

B.2 MPEG对象定义模块

MPEG对象定义模块定义了MPEG-2体系下最基本的对象以及系统出现的异常。
定义的最基本的MPEG-2对象有：
——传送流类（TransportStream）；
——基本流类（ElementaryStream）；
——广播业务类（Service）。
定义的MPEG-2异常有：
——广播内容未授权异常（NotAuthorizedException）；
——资源异常（ResourceException）；
——调谐解调异常（TuningException）。
MPEG对象定义模块概要见表B.1。

表 B.1 MPEG 对象定义模块概要

接口	
NotAuthorizedInterface	广播内容未授权报告接口，定义了失败原因常量，提供了查找失败原因的方法。
类	
TransportStream	MPEG-2 传送流类，代表一个 MPEG-2 传送流（TS），提供了获取传送流信息的方法。
ElementaryStream	MPEG-2 基本流类，代表一个在传送流（TS）中承载的基本流（ES），提供了获取基本流信息的方法。
Service	MPEG-2 业务类，代表一个在传送流中承载的 MPEG-2 业务，提供了获取业务信息的方法。
异常	
NotAuthorizedException	广播内容未授权异常，实现 NotAuthorizedInterface 接口，当访问无授权的加扰数据时抛出。
TuningException	调谐解调异常，当调谐解调失败时抛出。
ResourceException	资源异常，当因资源缺乏而无法进行操作时抛出。

B.2.1 接口org.davic.mpeg.NotAuthorizedInterface

原型：public interface org.davic.mpeg.NotAuthorizedInterface
描述：广播内容未授权报告接口，定义了失败原因常量，提供了查找失败原因的方法。

示例 1：

调用 getType()方法判断是业务还是基本流发生解扰故障：

——若 getType()方法返回 SERVICE，则调用 getService()方法获取不能被解扰的 org.davic.mpeg.Service 对象；

——若 getType()方法返回 ELEMENTARY_STREAM，则调用 getElementaryStreams()方法获取不能被解扰的 org.davic.mpeg.ElementaryStream 对象数组；

示例 2：

调用 getReason()方法获得不能被解扰的具体原因：

——POSSIBLE_UNDER_CONDITIONS，包括如下次要原因：

COMMERCIAL_DIALOG

MATURITY_RATING_DIALOG

TECHNICAL_DIALOG

FREE_PREVIEW_DIALOG

OTHER

——NOT_POSSIBLE，包括如下次要原因：

NO_ENTITLEMENT

MATURITY_RATING

TECHNICAL

GEOGRAPHICAL_BLACKOUT

OTHER

B.2.1.1 常量域——失败主要原因

B.2.1.1.1 POSSIBLE_UNDER_CONDITIONS

原型：public static final int POSSIBLE_UNDER_CONDITIONS = 0

描述：失败主要原因——在某些条件下可以访问。

B.2.1.1.2 NOT_POSSIBLE

原型：public static final int NOT_POSSIBLE = 1

描述：失败主要原因——不可能访问。

B.2.1.2 常量域——失败次要原因

B.2.1.2.1 COMMERCIAL_DIALOG

原型：public static final int COMMERCIAL_DIALOG = 1

描述：POSSIBLE_UNDER_CONDITIONS 的次要原因——需要进行有关付费的对话。

B.2.1.2.2 TURITY_RATING_DIALOG

原型：public static final int MATURITY_RATING_DIALOG = 2

描述：POSSIBLE_UNDER_CONDITIONS 的次要原因——需要进行有关年龄限制的对话。

B.2.1.2.3 TECHNICAL_DIALOG

原型：public static final int TECHNICAL_DIALOG = 3

描述：POSSIBLE_UNDER_CONDITIONS 的次要原因——需要进行有关技术支持的对话。

B.2.1.2.4 FREE_PREVIEW_DIALOG

原型：public static final int FREE_PREVIEW_DIALOG = 4
描述：POSSIBLE_UNDER_CONDITIONS 的次要原因——需要进行解释免费预览的对话。

B.2.1.2.5 NO_ENTITLEMENT

原型：public static final int NO_ENTITLEMENT = 1
描述：NOT_POSSIBLE 的次要原因——用户未被授权。

B.2.1.2.6 MATURITY_RATING

原型：public static final int MATURITY_RATING = 2
描述：NOT_POSSIBLE 的次要原因——因年龄原因不允许操作。

B.2.1.2.7 TECHNICAL

原型：public static final int TECHNICAL = 3
描述：NOT_POSSIBLE 的次要原因——因某种技术原因不允许操作。

B.2.1.2.8 GEOGRAPHICAL_BLACKOUT

原型：public static final int GEOGRAPHICAL_BLACKOUT = 4
描述：NOT_POSSIBLE 的次要原因——因地理原因不允许操作。

B.2.1.2.9 OTHER

原型：public static final int OTHER = 5
描述：POSSIBLE_UNDER_CONDITIONS 及 NOT_POSSIBLE 的次要原因——其他的原因。

B.2.1.3 常量域——MPEG-2对象

B.2.1.3.1 SERVICE

原型：public static final int SERVICE = 0
描述：发生错误的 MPEG-2 对象——业务对象被拒绝访问。

B.2.1.3.2 ELEMENTARY_STREAM

原型：public static final int ELEMENTARY_STREAM = 1
描述：发生错误的 MPEG-2 对象——基本流对象被拒绝访问。

B.2.1.4 方法

B.2.1.4.1 getType

原型：public int getType()
描述：获取发生未授权错误的 MPEG-2 对象类型。
参数：无。
返回：int 型，表示不能被解扰的 MPEG-2 对象类型，可取值为 SERVICE 或 ELEMENTARY_STREAM，详见 org.davic.mpeg.NotAuthorizedInterface 接口的“MPEG-2 对象”常量域定义。

B.2.1.4.2 getService

原型：public org.davic.mpeg.Service getService()
描述：获取不能被解扰的MPEG-2业务对象。
参数：无。
返回：Service对象，表示不能被解扰的业务。
——若getType()方法返回的类型为SERVICE，则此方法返回不能被解扰的业务对象；
——若getType()方法返回的类型为ELEMENTARY_STREAM，则此方法返回null。

B.2.1.4.3 getElementaryStreams

原型：public org.davic.mpeg.ElementaryStream[] getElementaryStreams()
描述：获取不能被解扰的基本流对象。
参数：无。
返回：ElementaryStream对象数组，表示不能被解扰的基本流。否则返回null。
——若getType()方法返回的类型为ELEMENTARY_STREAM，则此方法返回不能被解扰的基本流对象数组；
——若getType()方法返回的类型为SERVICE，则此方法返回null。

B.2.1.4.4 getReason

原型：public int[] getReason(int index) throws java.lang.IndexOutOfBoundsException
描述：获取无法被解扰的原因。
参数：index - int型，表示组件序号。
——若MPEG-2业务对象解扰失败，则该参数设为0；
——若MPEG-2基本流对象解扰失败，则index参数表示由getElementaryStreams()方法所返回的org.davic.mpeg.ElementaryStream对象数组的序号。
返回：int型数组，数组长度为2，int[0]表示主要原因，int[1]表示次要原因，取值详见org.davic.mpeg.NotAuthorizedInterface接口的“失败主要原因”和“失败次要原因”常量域定义。
异常：java.lang.IndexOutOfBoundsException - 若MPEG-2业务对象解扰失败，而index参数取非0值，则抛出此异常；若MPEG-2基本流对象解扰失败，而index超出了由getElementaryStreams()方法所返回的ElementaryStream数组的长度，则抛出此异常。

B.2.2 类org.davic.mpeg.TransportStream

原型：public class org.davic.mpeg.TransportStream
描述：MPEG-2传送流类，代表一个MPEG-2传送流（TS），提供了获取传送流信息的方法。一个传送流对象代表着可以通过特定网络接口进行访问的某个传送流，这表明传送流对象与网络接口之间有着隐含的关联关系。因此，若多个网络接口同时发送同一个传送流，也应当为每个接口发送的传送流构建单独的传送流对象。

B.2.2.1 方法

B.2.2.1.1 getTransportStreamId

原型：public int getTransportStreamId()
描述：获取传送流标识符（即获取PAT中的transport_stream_id字段信息）。
参数：无。

返回：int型，表示传送流的标识符。

B.2.2.1.2 retrieveService

原型：public org.davic.mpeg.Service retrieveService(int serviceId)
描述：根据指定的业务标识，获取传送流中承载的业务对象。
参数：serviceId - int 型，表示被请求业务的业务标识。
返回：org.davic.mpeg.Service 对象，表示被请求的业务。若指定的业务不存在，则返回 null。

B.2.2.1.3 retrieveServices

原型：public org.davic.mpeg.Service[] retrieveServices()
描述：获取传送流中承载的所有业务对象。
参数：无。
返回：Service 对象数组，表示传送流中承载的所有业务。若无业务对象，则返回 null。

B.2.3 类org.davic.mpeg.ElementaryStream

原型：public class org.davic.mpeg.ElementaryStream
描述：MPEG-2 基本流类，代表一个在传送流（TS）中承载的基本流（ES），提供了获取基本流信息的方法。若一个基本流被用于多个业务，则应实现该类的多个实例，即一个实例用于一个业务对象。
说明：本类定义的基本流信息来源于PMT。

B.2.3.1 方法

B.2.3.1.1 getPID

原型：public int getPID()
描述：获取承载该基本流的传送包的 PID 值。
参数：无。
返回：int 型，表示承载该基本流的传送包的 PID 值。

B.2.3.1.2 getAssociationTag

原型：public java.lang.Integer getAssociationTag()
描述：获取该基本流的 DSM-CC 关联标识。
参数：无。
返回： java.lang.Integer 对象，表示该基本流的 DSM-CC 关联标识，若无关联标识则返回 null。

B.2.3.1.3 getService

原型：public org.davic.mpeg.Service getService()
描述：获取该基本流所属的业务。
参数：无。
返回：org.davic.mpeg.Service 对象，表示包含该基本流的业务。

B.2.4 类org.davic.mpeg.Service

原型：public class org.davic.mpeg.Service
描述：MPEG-2 业务类，代表一个在传送流中承载的 MPEG-2 业务，提供了获取业务信息的方法。

B.2.4.1 方法

B.2.4.1.1 getServiceId

原型：public int getServiceId()
描述：获取业务的标识符（service_id）。
参数：无。
返回：int 型，表示该业务的业务标识符。

B.2.4.1.2 getTransportStream

原型：org.davic.mpeg.TransportStream getTransportStream()
描述：获取承载该业务的传送流对象。
参数：无。
返回：org.davic.mpeg.TransportStream 对象，表示承载该业务的传送流对象。

B.2.4.1.3 retrieveElementaryStream

原型：public org.davic.mpeg.ElementaryStream retrieveElementaryStream(int pid)
描述：根据指定的传送流 PID，获取该业务所包含的基本流对象。
参数：pid - int 型，表示承载该基本流的传送包的标识。
返回：ElementaryStream 对象，表示该业务所包含的基本流。若找不到指定的基本流对象，则返回 null。

B.2.4.1.4 retrieveElementaryStreams

原型：public org.davic.mpeg.ElementaryStream[] retrieveElementaryStreams()
描述：获取该业务中的所有基本流对象。
参数：无。
返回：org.davic.mpeg.ElementaryStream 对象数组，表示业务中的所有基本流，若无则返回 null。

B.2.5 异常org.davic.mpeg.NotAuthorizedException

原型：public class org.davic.mpeg.NotAuthorizedException extends java.langException implements NotAuthorizedInterface
描述：广播内容未授权异常，当访问无授权的加扰数据时抛出。

B.2.6 异常org.davic.mpeg.ObjectUnavailableException

原型：public class org.davic.mpeg.ObjectUnavailableException extends java.lang.Exception
描述：对象不可用异常，当对象无法访问时抛出。

B.2.7 异常org.davic.mpeg.ResourceException

原型：public class org.davic.mpeg.ResourceException extends java.lang.Exception
描述：资源缺乏异常，当因资源缺乏而无法进行操作时抛出。

B.3 DVB对象定义模块

DVB 对象定义模块定义了 DVB 体系下的 MPEG-2 基本对象：
——DVB 传送流类（org.davic.mpeg.dvb.DvbTransportStream）；

——DVB 基本流类（org.davic.mpeg.dvb.DvbElementaryStream）；
——DVB 广播业务类（org.davic.mpeg.dvb.DvbService）。
DVB 对象定义模块概要见表 B.2。

表 B.2 DVB 对象定义模块概要

类	
DvbElementaryStream	DVB 基本流类，代表一个在传送流（TS）中承载的符合 DVB 语义约束的 MPEG-2 基本流（ES），提供了获取 DVB 基本流信息的方法。
DvbService	DVB 业务类，代表一个在传送流中承载的符合 DVB 语义约束的 MPEG-2 业务，提供了获取 DVB 业务信息的方法。
DvbTransportStream	DVB 传送流类，代表一个符合 DVB 语义约束的 MPEG-2 传送流，提供了获取 DVB 传送流信息的方法。

B.3.1 类org.davic.mpeg.dvb.DvbElementaryStream

原型：public class org.davic.mpeg.dvb.DvbElementaryStream extends org.davic.mpeg.ElementaryStream
描述：DVB 基本流类，代表一个在传送流（TS）中承载的符合 DVB 语义约束的 MPEG-2 基本流（ES），提供了获取 DVB 基本流信息的方法。

B.3.1.1 方法

B.3.1.1.1 getComponentTag

原型：public java.lang.Integer getComponentTag()
描述:获取 DVB 基本流的组件标签(即获取 stream_identifier_descriptor 描述符中的 component_tag 字段值）。
参数：无。
返回：java.lang.Integer 对象，表示该 DVB 基本流的组件标签，若无组件标签（即 ES 流循环未携带 stream_identifier_descriptor 描述符），则返回 null。

B.3.2 类org.davic.mpeg.dvb.DvbService

原型：public class org.davic.mpeg.dvb.DvbService extends org.davic.mpeg.Service
描述：DVB 业务类，代表一个在传送流中承载的符合 DVB 语义约束的 MPEG-2 业务，提供了获取 DVB 业务信息的方法。

B.3.2.1 方法

B.3.2.1.1 retrieveDvbElementaryStream

原型：public org.davic.mpeg.dvb.DvbElementaryStream retrieveDvbElementaryStream(int componentTag)
描述：根据指定的组件标签获取该 DVB 业务所包含的 DVB 基本流。
参数：componentTag - int 型，表示 DVB 基本流的组件标签值。
返回:org.davic.mpeg.dvb.DvbElementaryStream 对象,表示 DVB 基本流对象。若由参数 componentTag 指定的 DVB 基本流不存在，则返回 null。

B.3.3 类org.davic.mpeg.dvb.DvbTransportStream

原型：public class org.davic.mpeg.dvb.DvbTransportStream extends
org.davic.mpeg.TransportStream

描述：DVB 传送流类，代表一个符合 DVB 语义约束的 MPEG-2 传送流，提供了获取 DVB 传送流信息的方法。

B.3.3.1 方法

B.3.3.1.1 getNetworkId

原型：public int getNetworkId()
描述：获取 DVB 传送流所在网络的网络标识。
参数：无。
返回：int 型，表示传送流所在网络的网络标识。

B.3.3.1.2 getOriginalNetworkId

原型：public int getOriginalNetworkId()
描述：获取 DVB 传送流的原始网络标识。
参数：无。
返回：int 型，表示传送流的原始网络标识。

B.4 SECTIOIN过滤模块

SECTION 过滤模块提供了与 MPEG-2 段(section)过滤相关的类和方法。
SECTION 过滤模块概要见表 B.3。

表 B.3 SECTION 过滤模块概要

接口	
SectionFilterListener	段过滤事件监听器接口，提供了段过滤事件处置回调方法，由应用层实现。
类	
Section	MPEG-2 段类，描述了从传输流中过滤到的一个段。
SectionFilterGroup	段过滤器组类，代表一个 MPEG-2 过滤器组，可以作为一个基本操作单元被激活和释放。
SectionFilter	段过滤器类，该类为一组具有不同生命周期和缓存长度特点的段过滤器类的基类，提供了过滤器基本操作方法。
SimpleSectionFilter	简单段过滤器类，继承 SectionFilter 类。
TableSectionFilter	表段过滤器类，继承 SectionFilter 类。
RingSectionFilter	循环段过滤器类，继承 SectionFilter 类。
事件	
SectionFilterEvent	段过滤事件类，一组段过滤事件类的基类。
SectionAvailableEvent	段数据可用事件，继承 SectionFilterEvent 类，报告过滤到了一个完整的段。
VersionChangeDetectedEvent	段过滤版本变更事件，继承 SectionFilterEvent 类。
EndOfFilteringEvent	段过滤结束事件，继承 SectionFilterEvent 类，报告段过滤结束。

表 B.3（续）

事件	
IncompleteFilteringEvent	段过滤不完整事件，继承 EndOfFilteringEvent 类。
TimeOutEvent	段过滤超时事件，继承 EndOfFilteringEvent 类。
FilterResourcesAvailableEvent	过滤器资源可用事件，继承 ResourceStatusEvent 类。
ForcedDisconnectedEvent	段过滤器组与传送流强制断开事件，继承 ResourceStatusEvent 类。
异常	
SectionFilterException	段过滤异常的基类。
ConnectionLostException	连接丢失异常，继承 SectionFilterException 类。
FilteringInterruptedException	过滤中断异常，继承 SectionFilterException 类。
FilterResourceException	过滤器资源异常，继承 SectionFilterException 类。
IllegalFilterDefinitionException	非法过滤条件异常，继承 SectionFilterException 类。
InvalidSourceException	段数据源无效异常，继承 SectionFilterException 类。
NoDataAvailableException	段对象无可用数据异常，继承 SectionFilterException 类。

SECTION 过滤模块各对象之间的关系见图 B.1。

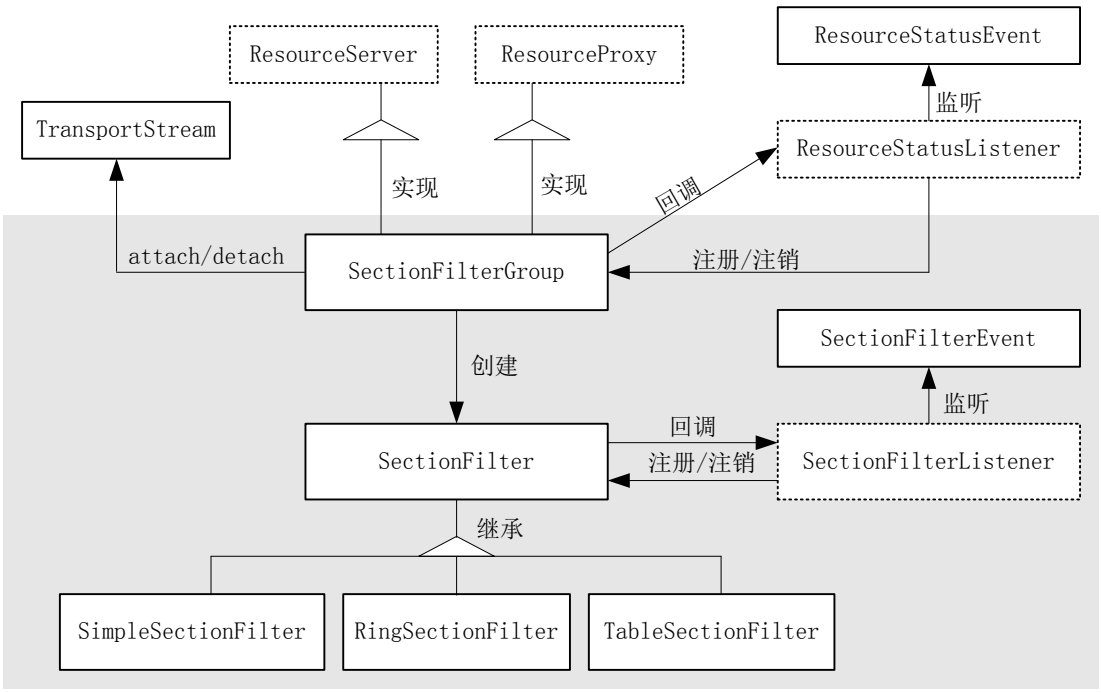


图 B.1　SECTION 过滤模块对象关系

SectionFilterGroup 对象和 TransportStream 对象之间连接关系状态转移图见图 B.2。

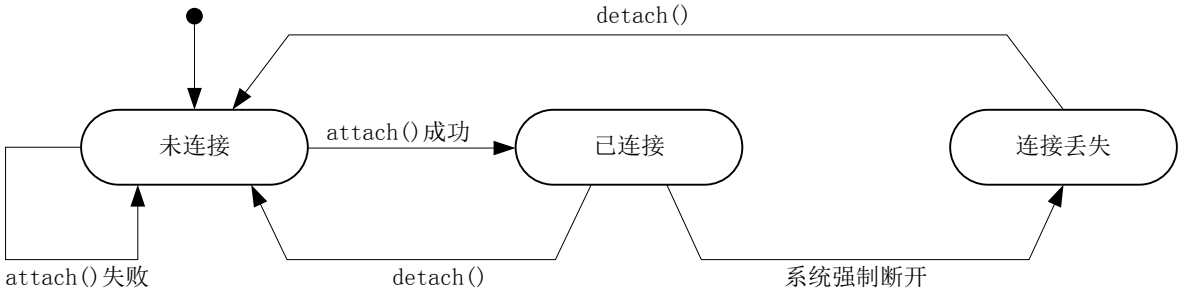


图 B.2 SectionFilterGroup 对象和 TransportStream 对象连接关系状态转移图

B.4.1 接口org.davic.mpeg.sections.SectionFilterListener

原型：public interface org.davic.mpeg.sections.SectionFilterListener
extends java.util.EventListener
描述：段过滤事件监听器接口，提供了段过滤事件处置回调方法，由应用层实现。

B.4.1.1 方法

B.4.1.1.1 sectionFilterUpdate

原型：void sectionFilterUpdate(org.davic.mpeg.sections.SectionFilterEvent event)
描述：段过滤事件处置回调方法。
参数：event - org.davic.mpeg.sections.SectionFilterEvent 对象，表示段过滤事件。方法实现者应通过 instanceof 方法进一步判断 event 参数的类型。
返回：无。

B.4.2 类org.davic.mpeg.sections.Section

原型：public class org.davic.mpeg.sections.Section implements Cloneable
描述：MPEG-2 段类，描述了从传送流中过滤到的一个段。用 clone()方法复制的段对象是一个独立的新对象。原来对象的状态变化不会影响新对象。用 clone()方法复制对象必须在没有抛出异常的情况下实现。

B.4.2.1 方法

B.4.2.1.1 clone

原型：public java.lang.Object clone()
描述：克隆 Section 对象。用 clone()方法复制的段对象是一个独立的新对象。原来对象的状态变化不会影响新对象。用 clone()方法复制对象必须在没有抛出异常的情况下实现。
重写：Object 类的 clone()方法。
参数：无。
返回： Section 对象。

B.4.2.1.2 current_next_indicator

原型：public boolean current_next_indicator()
throws org.davic.mpeg.sections.NoDataAvailableException

描述：获取段头 current_next_indicator 字段的值。
参数：无。
返回：boolean 型，取值 true 表示 current_next_indicator 字段为 1，false 表示 current_next_indicator 字段为 0。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若未过滤到段数据，则抛出此异常。

B.4.2.1.3 getByteAt

原型：public byte getByteAt(int index)
throws org.davic.mpeg.sections.NoDataAvailableException
,java.lang.IndexOutOfBoundsException
描述：获取段对象中所过滤到的段数据的指定字节。
参数：index－int 型，表示待检索字节在段中的位置。段中第一个字节（即 table_id 字段）的 index 值为 1。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若未过滤到段数据，则抛出此异常；
java.lang.IndexOutOfBoundsException - 若待检索字节已在段数据范围之外则抛出此异常。

B.4.2.1.4 getData

原型：public byte[] getData() throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段对象中所过滤到的段数据，包括段头。每次调用此方法将返回一个段数据的新拷贝。
参数：无。
返回：无。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若未过滤到段数据，则抛出此异常。

B.4.2.1.5 getData

原型：public byte[] getData(int index, int length)
throws org.davic.mpeg.sections.NoDataAvailableException,
java.lang.IndexOutOfBoundsException
描述：获取段对象中所过滤到的段数据的指定部分。每次调用此方法将返回一个段数据的新拷贝。
参数：index－int 型，表示待检索字节在段中的位置。段中第一个字节（即 table_id 字段）的 index 值为 1；
length - int 型，表示待检索数据从首字节开始后续连续字节数。
返回：无。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若未过滤到段数据，则抛出此异常；
java.lang.IndexOutOfBoundsException - 若待检索数据的某一部分已在段数据范围之外则抛出此异常。

B.4.2.1.6 getFullStatus

原型：public boolean getFullStatus()
描述：判断段对象是否包含有效数据。
参数：无。
返回：boolean 型，取值 true 表示段对象包含有效数据，false 表示无有效数据。

B.4.2.1.7 last_section_number

原型：public int last_section_number()
throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 last_section_number 字段的值。
参数：无。
返回：int 型，表示 last_section_number 字段的值。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.8 private_indicator

原型：public boolean private_indicator()
throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 private_indicator 字段的值。
参数：无。
返回：无。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.9 section_length

原型：public int section_length() throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 section_length 字段的值。
参数：无。
返回：int 型，表示段头 section_length 字段的值。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.10 section_number

原型：public int section_number() throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 section_number 字段的值。
参数：无。
返回：int 型，表示 section_number 字段的值。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.11 section_syntax_indicator

原型：public boolean section_syntax_indicator()
throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 section_syntax_indicator 字段的值。
参数：无。
返回：boolean 型，取值 true 表示 section_syntax_indicator=1，取值 false 表示 section_syntax_indicator=0。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.12 setEmpty

原型：public void setEmpty()
描述：设置段对象内的数据不再有效，一般由 org.davic.mpeg.sections.RingSectionFilter 使用，表明特定对象可被再次使用。
参数：无。
返回：无。

B.4.2.1.13 table_id_extension

原型：public int table_id_extension()
throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 table_id_extension 字段的值。
参数：无。
返回：int 型，表示 table_id_extension 字段的值。应用需要根据 table_id 的类型决定 table_id_extension 字段的具体语义。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.14 table_id

原型：public int table_id() throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 table_id 字段的值。
参数：无。
返回：int 型，表示段的 table_id 字段。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.2.1.15 version_number

原型：public short version_number()
throws org.davic.mpeg.sections.NoDataAvailableException
描述：获取段头 version_number 字段的值。
参数：无。
返回：short 型，表示 version_number 字段的值。
异常：org.davic.mpeg.sections.NoDataAvailableException - 若没有过滤到段数据，则抛出此异常。

B.4.3 类org.davic.mpeg.sections.SectionFilterGroup

原型：public class org.davic.mpeg.sections.SectionFilterGroup
implements org.davic.resources.ResourceProxy, org.davic.resources.ResourceServer
描述：段过滤器组类，代表一个 MPEG-2 过滤器组，可以作为一个基本操作单元被激活和释放。此类的目的是将独立应用间可能出现的资源死锁减至最小。

B.4.3.1 方法

B.4.3.1.1 SectionFilterGroup

原型：public SectionFilterGroup(int numberOfFilters)
throws java.lang.IllegalArgumentException
描述：构造方法，创建一个段过滤器组对象。
参数：numberOfFilters - int 型，表示段过滤器组需要占用的段过滤器的数量。
java.lang.IllegalArgumentException - 若参数 numberOfFilters 小于 1，则抛出此异常。

B.4.3.1.2 SectionFilterGroup

原型：public SectionFilterGroup(int numberOfFilters, boolean resourcePriority)
throws java.lang.IllegalArgumentException
描述：构造方法，创建一个段过滤器组对象。
参数：numberOfFilters - int 型，表示段过滤器组对象需要占用的段过滤器的数目；
resourcePriority - boolean 型，表示当过滤器资源不足时本过滤器组对象与 sections 包内其他对象之间拥有资源的相对优先级，取值 true 表示高优先级，false 表示低优先级。resourcePriority 参数的作用范围仅限于一个应用内。
异常：java.lang.IllegalArgumentException - 若参数 numberOfFilters 小于 1，则抛出此异常。

B.4.3.1.3 addResourceStatusEventListener

原型：public synchronized void addResourceStatusEventListener
(org.davic.resources.ResourceStatusListener listener)
描述：注册过滤器组的资源状态变化事件监听器，实现接口 org.davic.resources.ResourceProxy 中的 addResourceStatusEventListener()方法。若多次调用此方法，每一个监听器对象均可被告知每一次的资源变化。
参数：listener - org.davic.resources.ResourceStatusListener 对象，表示待注册的资源状态变化事件监听器。
返回：无。

B.4.3.1.4 removeResourceStatusEventListener

原型：public synchronized void removeResourceStatusEventListener
(org.davic.resources.ResourceStatusListener listener)
描述：注销过滤器组的资源状态变化事件监听器，实现接口 ResourceProxy 中的 removeResourceStatusEventListener()方法。若该监听器对象原来就不被通知，此方法不会产生其他影响。
参数：listener - org.davic.resources.ResourceStatusListener 对象，表示待注销的资源状态变化事件监听器。
返回：无。

B.4.3.1.5 attach

原型：public boolean attach(org.davic.mpeg.TransportStream stream,
org.davic.resources.ResourceClient client,
java.lang.Object requestData)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.sections.InvalidSourceException, org.davic.mpeg.TuningException
描述：将过滤器组对象与传送流关联。该过滤器组将试图获得在其建立时指定数量的段过滤器。对已

经连接的过滤器组再次调用 attach()方法直接返回 false。使用过滤、过滤器组时需要函数调用顺序是:过滤器组 attach, 过滤器 startFiltering, 过滤器 stopFiltering, 滤器组 detach。

参数：stream - org.davic.mpeg.TransportStream 对象，表示传送流对象；
client - org.davic.resources.esourceClient 对象，表示资源客户端，当过滤器资源失去时，需要通知的对象；
requestData - java.lang.Object 对象，表示资源管理器通知资源失去时所传递给资源客户端的应用专有数据。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 预留特定段过滤器失败，则抛出此异常；
org.davic.mpeg.sections.InvalidSourceException - 指定的传送流无效，则抛出此异常；
org.davic.mpeg.TuningException - 未调谐到指定的传送流，则抛出此异常。

B.4.3.1.6 detach

原型：public boolean detach()

描述：将过滤器组对象与传送流断开。过滤器组所占用的段过滤器将被释放。已经断开的过滤器组直接返回 false。

参数：无。

返回：boolean 型，成功时返回 true，失败时返回 false。

B.4.3.1.7 getClient

原型：public org.davic.resources.ResourceClient getClient()

描述：获取在 attach()方法中指定的 org.davic.resources.ResourceClient 对象，以便段过滤器从 SectionFilterGroup 中被去除时得到通知。实现接口 org.davic.resources.ResourceProxy 中的 getClient()方法。

参数：无。

返回：org.davic.resources.ResourceClient 对象，表示一个资源客户。若过滤器组并未与传送流关联，则返回 null。

B.4.3.1.8 getSource

原型：public org.davic.mpeg.TransportStream getSource()

描述：获取过滤器组对象当前关联的传送流。

参数：无。

返回：org.davic.mpeg.TransportStream 对象，表示本过滤器组所关联的传送流。若过滤器组并未与传送流关联，则返回 null。

B.4.3.1.9 newRingSectionFilter

原型：public RingSectionFilter newRingSectionFilter(int ringSize)
throws java.lang.IllegalArgumentException

描述：在父过滤器组中创建一个循环段过滤器。一旦启动，循环段过滤器
org.davic.mpeg.sections.RingSectionFilter 对象将使用父过滤器组创建时指定的段过滤器。

参数：ringSize - int 型，表示循环段过滤器所占用的 Section 对象数目。
返回：org.davic.mpeg.sections.RingSectionFilter 对象，表示一个循环段过滤器。
异常：java.lang.IllegalArgumentException - 若参数 ringSize 参数小于 1 则抛出此异常。

B.4.3.1.10 newRingSectionFilter

原型：public RingSectionFilter newRingSectionFilter(int ringSize, int sectionSize)
throws java.lang.IllegalArgumentException
描述：在父过滤器组中创建一个循环段过滤器。一旦启动，循环段过滤器
org.davic.mpeg.sections.RingSectionFilter 对象将使用父过滤器组创建时指定的段过滤器。
参数：ringSize - int 型，表示循环段过滤器所占用的 Section 对象数目；
sectionSize - int 型，表示段数据缓冲区长度，单位为字节。若待过滤的段其长度大于此值，多余的数据将被截掉，过滤继续，不对应用作出任何通告。
返回：org.davic.mpeg.sections.RingSectionFilter 对象，表示一个循环段过滤器。
异常：java.lang.IllegalArgumentException - 若 ringSize 参数小于 1 或 sectionSize 参数小于 1，则抛出此异常。

B.4.3.1.11 newSimpleSectionFilter

原型：public SimpleSectionFilter newSimpleSectionFilter()
描述：在父过滤器组中创建一个简单段过滤器。一旦启动，简单段过滤器
org.davic.mpeg.sections.SimpleSectionFilter 对象将使用父过滤器组创建时所指定的段过滤器。此种段过滤器拥有一个能容纳缺省长段(4096 字节)的缓冲区。
参数：无。
返回：org.davic.mpeg.sections.SimpleSectionFilter 对象，表示一个简单段过滤器。

B.4.3.1.12 newSimpleSectionFilter

原型：public SimpleSectionFilter newSimpleSectionFilter(int sectionSize)
throws java.lang.IllegalArgumentException
描述：在父过滤器组中创建一个简单段过滤器。一旦启动，简单段过滤器
org.davic.mpeg.sections.SimpleSectionFilter 对象将使用父过滤器组创建时指定的段过滤器。
参数：sectionSize - int 型，表示段数据缓冲区长度，单位为字节。若待过滤的段其长度大于此值，多余的数据将被截掉，过滤继续，不对应用作出任何通告。
返回：org.davic.mpeg.sections.SimpleSectionFilter 对象，表示一个简单段过滤器。
异常：java.lang.IllegalArgumentException - 若 sectionSize 参数小于 1，则抛出此异常。

B.4.3.1.13 newTableSectionFilter

原型：public TableSectionFilter newTableSectionFilter()
描述：在父过滤器组中创建一个表段过滤器。一旦启动，简单段过滤器
org.davic.mpeg.sections.SimpleSectionFilter 对象将使用父过滤器组创建时指定的段过滤器。表段过滤器的每一段均拥有容纳缺省的长段(4096 字节)的缓冲区。
参数：无。
返回：org.davic.mpeg.sections.TableSectionFilter 对象，表示一个表段过滤器。

B.4.3.1.14 newTableSectionFilter

原型：public TableSectionFilter newTableSectionFilter(int sectionSize)
throws java.lang.IllegalArgumentException

描述：在父过滤器组中创建一个表段过滤器。一旦启动，简单段过滤器org.davic.mpeg.sections.SimpleSectionFilter 对象将使用父过滤器组创建时指定的段过滤器。

参数：sectionSize - int 型，表示段数据滤缓冲区长度，单位为字节。 当第一个段被捕获，且表内段的总数已知时，所创建的每个段都将拥有这一长度的缓冲区。 若待过滤的段长度大于此值，多余的数据将被截掉，过滤继续，不对应用作出任何通告。

返回：org.davic.mpeg.sections.TableSectionFilter 对象，表示一个表过滤器。

异常：java.lang.IllegalArgumentException - 若 sectionSize 参数小于 1，则抛出此异常。

B.4.4 类org.davic.mpeg.sections.SectionFilter

原型：public abstract class org.davic.mpeg.sections.SectionFilter

描述：段过滤器类，该类为一组具有不同生命周期和缓存长度特点的段过滤器类的基类，提供了过滤器基本操作方法。若一个 org.davic.mpeg.sections.SectionFilterGroup 对象因为用户或资源回收等原因而与传送流取消关联，已经开始工作的段过滤器仍将持续工作。因此当org.davic.mpeg.sections.SectionFilterGroup 对象再次重新与传送流关联时，这些过滤器也将再次被激活。

B.4.4.1 方法

B.4.4.1.1 addSectionFilterListener

原型：public synchronized void addSectionFilterListener(org.davic.mpeg.sections.SectionFilterListener listener)

描述：注册段过滤事件监听器。

参数：listener - org.davic.mpeg.sections.SectionFilterListener 对象，表示待注册的段过滤事件监听器。

返回：无。

B.4.4.1.2 removeSectionFilterListener

原型：public synchronized void
removeSectionFilterListener(org.davic.mpeg.sections.SectionFilterListener
listener)

描述：注销段过滤事件监听器。

参数：listener - org.davic.mpeg.sections.SectionFilterListener 对象，表示待注销的段过滤事件监听器。

返回：无。

B.4.4.1.3 setTimeOut

原型：public void setTimeOut(long milliseconds)
throws java.lang.IllegalArgumentException

描述：设置过滤器超时时间。当超时发生时，产生的超时事件 org.davic.mpeg.sections.TimeOutEvent

将被发送到段过滤事件监听器，过滤停止。

——对 org.davic.mpeg.sections.SimpleSectionFilter，若在一定期限内没有段数据到达，则会产生此超时事件；

——对 org.davic.mpeg.sections.TableSectionFilter，若在一定期限内没有完整的表到达，则会产生此超时事件；

——对 org.davic.mpeg.sections.RingSectionFilter，若自上次成功过滤后经过一定周期未再成功过滤，则会产生此超时事件。

参数：milliseconds - long 型，表示超时期限，单位为毫秒。若将参数设置为 0，则超时作用将失效。设置超时只对后续操作有效。缺省值为 0。

返回：无。

异常：java.lang.IllegalArgumentException - 若超时参数 milliseconds 值为负，则抛出此异常。

B.4.4.1.4 startFiltering

原型：public boolean startFiltering(java.lang.Object appData, int pid)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.IllegalFilterDefinitionException,
org.davic.mpeg.sections.ConnectionLostException

描述：启动段过滤，过滤出由 pid 参数指定的传送包中所承载的所有段（section）。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 org.davic.mpeg.sections.SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。若应用不需要任何附加数据，此参数可以设为 null；
pid - int 型，表示待过滤的传送包标识(TS_PID)。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；
org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；
org.davic.mpeg.sections.IllegalFilterDefinitionException - 若是 org.davic.mpeg.sections.TableSectionFilter 调用该方法，则抛出此异常；
org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态，由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常。

B.4.4.1.5 startFiltering

原型：public boolean startFiltering(java.lang.Object appData, int pid, int tableID)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.ConnectionLostException,

org.davic.mpeg.sections.IllegalFilterDefinitionException

描述：启动段过滤，过滤出由 pid 参数指定的传送包中所承载的段（section），且表标识与 tableID 参数相匹配。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 org.davic.mpeg.sections.SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。 若应用不需要任何附加数据，此参数可以设为 null；

pid - int 型，表示待过滤的传送流包标识（TS_PID）；

tableID - int 型，表示待过滤的表标识（table_id）。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数 已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；

org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；

org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态， 由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常；

org.davic.mpeg.sections.IllegalFilterDefinitionException - 若 pid 或 tableID 参数为负数或者超出 MPEG-2 定义的范围时，则抛出此异常。

B.4.4.1.6 startFiltering

原型：public boolean startFiltering(java.lang.Object appData, int pid, int tableID,
byte[] posFilterDef, byte[] posFilterMask)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.sections.IllegalFilterDefinitionException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.ConnectionLostException

描述：启动段过滤，过滤出由参数 pid 指定的传送包中所承载的段（section），且表标识与 tableID 参数相匹配，段数据与给定的字节过滤器相匹配，过滤器字节数组的首字节对应段数据的第三个字节。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。若满足下列条件，表示段数据与给定的字节过滤器相匹配：

(posFilterDef[i] & posFilterMask[i]) == (段数据[3+i] & posFilterMask[i])

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 org.davic.mpeg.sections.SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。若应用不需要任何附加数据，此参数可以设为 null；

pid - int 型，表示待过滤的传送流包标识（TS_PID）；

tableID - int 型，表示待过滤的表标识（table_id）；

posFilterDef - byte 数组，表示段数据中位匹配的值；

posFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行

比较。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；

org.davic.mpeg.sections.IllegalFilterDefinitionException - 若定义了非法的过滤器，要么参数 posFilterDef 数组和 posFilterMask 数组长度不一致，要么其长度超出了系统过滤的能力，则抛出此异常；

org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；

org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态，由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常。

B.4.4.1.7 startFiltering

原型：public boolean startFiltering(java.lang.Object appData, int pid, int tableID,
byte[] posFilterDef, byte[] posFilterMask,
byte[] negFilterDef, byte[] negFilterMask)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.sections.IllegalFilterDefinitionException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.ConnectionLostException

描述：启动段过滤，过滤出由 pid 参数指定的传送包中所承载的段（section），且表标识与 tableID 参数相匹配， 段数据与给定的字节过滤器相匹配，过滤器字节数组的首字节对应段数据的第三个字节。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。若满足下列条件，表示段数据与给定的字节过滤器相匹配：

((posFilterDef[i] & posFilterMask[i]) == (段数据[3+i] & posFilterMask[i])) &&
((negFilterDef[i] & negFilterMask[i]) != (段数据[3+i] & negFilterMask[i]))

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。若应用不需要任何附加数据，此参数可以设为 null；

pid - int 型，表示待过滤的传送流包标识（TS_PID）；

tableID - int 型，表示待过滤的表标识（table_id）；

posFilterDef - byte 数组，表示段数据中位匹配的值；

posFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行比较；

negFilterDef - byte 数组，表示段数据中位匹配的值；

negFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行比较。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组

org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数 已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；

org.davic.mpeg.sections.IllegalFilterDefinitionException - 若定义了非法的过滤器，要么参数 posFilterDef 数组和 posFilterMask 数组长度不一致，要么其长度超出了系统过滤的能力，则抛出此异常；

org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；

org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组

org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态，由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常。

B.4.4.1.8 startFiltering

原型：public boolean startFiltering(java.lang.Object appData, int pid, int tableID,
int offset, byte[] posFilterDef, byte[] posFilterMask)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.sections.IllegalFilterDefinitionException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.ConnectionLostException

描述：启动段过滤，过滤出由 pid 参数指定的传送包中所承载的段（section），且表标识与 tableID 参数相匹配， 段数据与给定的字节过滤器相匹配，过滤器字节数组的首字节对应段数据的偏移量由参数 offset 指定。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。若满足下列条件，表示段数据与给定的字节过滤器相匹配：

(posFilterDef[i] & posFilterMask[i]) == (段数据[offset+i] & posFilterMask[i])

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 org.davic.mpeg.sections.SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。若应用不需要任何附加数据，此参数可以设为 null；

pid - int 型，表示待过滤的传送流包标识（TS_PID）；

tableID - int 型，表示待过滤的表标识（table_id）；

offset - int 型，表示过滤器字节数组首字节对应的段数据的偏移量，取值大于等于 3，小于 31；

posFilterDef - byte 数组，表示段数据中位匹配的值；

posFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行比较。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；

org.davic.mpeg.sections.IllegalFilterDefinitionException - 若定义了非法的过滤器，要么参数 posFilterDef 数组和 posFilterMask 数组长度不一致，要么其长度超出了系统过滤的能力，则抛出此异常；

org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；

org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组

org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态，由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常。

B.4.4.1.9 startFiltering

原型：public boolean startFiltering(java.lang.Object appData,
int pid, int tableID, int offset,
byte[] posFilterDef, byte[] posFilterMask,
byte[] negFilterDef, byte[] negFilterMask)
throws org.davic.mpeg.sections.FilterResourceException,
org.davic.mpeg.sections.IllegalFilterDefinitionException,
org.davic.mpeg.NotAuthorizedException,
org.davic.mpeg.sections.ConnectionLostException

描述：启动段过滤，过滤出由 pid 参数指定的传送包中所承载的段（section），且表标识与 tableID 参数相匹配，段数据与给定的字节过滤器相匹配，过滤器字节数组的首字节对应段数据的偏移量由参数 offset 指定。父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象必须先与传输流关联。若满足下列条件，表示段数据与给定的字节过滤器相匹配：
((posFilterDef[i] & posFilterMask[i]) == (段数据[offset+i] & posFilterMask[i])) &&
((negFilterDef[i] & negFilterMask[i]) != (段数据[offset+i] & negFilterMask[i]))

参数：appData - java.lang.Object 对象，表示应用提供的附加数据对象，作为此方法调用而产生的所有过滤器事件 org.davic.mpeg.sections.SectionFilterEvent 对象的一部分，该对象将被传递给注册的段过滤事件监听器，应用可将此对象用于应用之间的内部通信。 若应用不需要任何附加数据，此参数可以设为 null；

pid - int 型，表示待过滤的传送流包标识（TS_PID）；

tableID - int 型，表示待过滤的表标识（table_id）；

offset - int 型，表示过滤器字节数组首字节对应的段数据的偏移量，取值大于等于 3，小于 31；

posFilterDef - byte 数组，表示段数据中位匹配的值；

posFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行比较；

negFilterDef - byte 数组，表示段数据中位匹配的值；

negFilterMask - byte 数组，表示段数据中位匹配的掩码，即只有掩码值为‘1’的位才进行比较。

返回：boolean 型，成功时返回 true，失败时返回 false。

异常：org.davic.mpeg.sections.FilterResourceException - 调用本方法时，若父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 启动的段过滤器总数 已经与该过滤器组创建时所占用的段过滤器数目相等，则抛出此异常。无论父段过滤器组是否与 TS 流关联，本异常都适用；

org.davic.mpeg.sections.IllegalFilterDefinitionException - 若定义了非法的过滤器，要么参数 posFilterDef 数组和 posFilterMask 数组长度不一致， 要么其长度超出了系统过滤的能力，则抛出此异常；

org.davic.mpeg.NotAuthorizedException - 若待过滤的段被加扰且无解扰权限，则抛出此异常；

org.davic.mpeg.sections.ConnectionLostException - 若父过滤器组

org.davic.mpeg.sections.SectionFilterGroup 处于丢失连接的状态，由于缺少资源或段过滤器无法完成该方法的调用时，则抛出此异常。

B.4.4.1.10 stopFiltering

原型：public boolean stopFiltering()

描述：停止段过滤。如果所在的父过滤器组 org.davic.mpeg.sections.SectionFilterGroup 对象已连接到一个传送流，则与此过滤器对象相匹配的段过滤将停止。使用过滤、过滤器组时需要函数调用顺序是:过滤器组 attach, 过滤器 startFiltering, 过滤器 stopFiltering, 滤器组 detach。如果过滤器已经停止，再次调用 stopFiltering 会直接返回 false。

参数：boolean 型，成功时返回 true，失败时返回 false。

返回：无。

B.4.5 类org.davic.mpeg.sections.SimpleSectionFilter

原型：public class org.davic.mpeg.sections.SimpleSectionFilter extends org.davic.mpeg.sections.SectionFilter

描述：简单段过滤器类，继承自 org.davic.mpeg.sections.SectionFilter 类。简单段过滤器：仅使用一次，当找到与指定过滤条件相匹配的段时，该简单段过滤器即停止过滤，如同调用了 SectionFilter.stopFiltering()方法。

B.4.5.1 方法

B.4.5.1.1 getSection

原型：public Section getSection() throws org.davic.mpeg.sections.FilteringInterruptedException

描述：获取本过滤器所过滤到的段数据。此方法所获得的段，记述了最后一个满足过滤条件的 MPEG-2 的段。若简单过滤器正在过滤，此方法被阻塞，直到过滤结束。假如在此期间未重新调用 startFiltering()方法，重复调用本方法，均会返回相同的段对象。每启动一个新的过滤操作，都会产生一个新的段对象。旧的段对象将被删除，除非应用自己维护旧的段对象。所有访问前一个段对象的操作都会抛出 org.davic.mpeg.sections.NoDataAvailableException 异常。

参数：无。

返回：无。

异常：org.davic.mpeg.sections.FilteringInterruptedException - 若在发现匹配的段之前过滤操作停止，则抛出此异常。

B.4.6 类org.davic.mpeg.sections.TableSectionFilter

原型：public class org.davic.mpeg.sections.TableSectionFilter extends org.davic.mpeg.sections.SectionFilter

描述：表段过滤器类，继承自 org.davic.mpeg.sections.SectionFilter 类。表段过滤器：以最小的应用干预来接收完整的表。 表段过滤器启动时，将过滤符合给定过滤条件的第一个段。该段一旦被找到，将使用 last_section_number 字段来确定构成整个表所需要的段对象数。段对象数量被确定后，过滤器重新启动以接收表中所有的段。每接收到一个段，都会产生

org.davic.mpeg.sections.SectionAvailableEvent 事件。在接收到整个表之后，将会产生过滤结束 org.davic.mpeg.sections.EndOfFilteringEvent 事件。表中所有段的版本号应相同。如果某滤出段的版本号与第一个段不同，将产生 org.davic.mpeg.sections.VersionChangeDetectedEvent 事件，新捕获的段将被忽略，并继续对原版本的段进行过滤。

设置段过滤参数时需要注意：若约束条件过多则过滤器无法自动停止；而约束条件过宽则过滤产生不一致的结果（例如：使用 org.davic.mpeg.sections.TableSectionFilter 过滤部分段。当 API 发现过滤参数定义不完全，可能导致过滤器阻塞或 MPEG-2 不兼容的结果，将产生 org.davic.mpeg.sections.IncompleteFilteringEvent 事件，使过滤停止。

B.4.6.1 方法

B.4.6.1.1 getSections

原型：public Section[] getSections() throws
org.davic.mpeg.sections.FilteringInterruptedException

描述：获取本过滤器所过滤到的表的所有段数据。本方法将返回表的所有段对象数组。数组元素按段编号 section_number 排列。若直到调用本方法时某些段尚未过滤到，则其在数组中的相应记录被设置为 null。若没有任何段被滤出，此方法被阻塞，直到至少获得一个段或者过滤结束。假如在此其间未重新调用 startFiltering()方法，重复调用本方法，均会返回相同的数组。每启动一次新的过滤操作，都会有一个段对象数组被创建。旧的段数组对象将被删除，除非应用自己维护旧的段数组对象。所有访问前一个段数组对象的操作都会抛出 org.davic.mpeg.sections.NoDataAvailableException 异常。

参数：无。

返回：无。

异常：org.davic.mpeg.sections.FilteringInterruptedException - 若在发现匹配的段之前过滤操作停止，则抛出此异常。

B.4.7 类org.davic.mpeg.sections.RingSectionFilter

原型：public class org.davic.mpeg.sections.RingSectionFilter extends
org.davic.mpeg.sections.SectionFilter

描述：循环段过滤器类，继承自 org.davic.mpeg.sections.SectionFilter 类。

循环段过滤器：一旦启动后，无需停止和重新设置，用于捕获连续的 MPEG-2 段数据。一个 org.davic.mpeg.sections.RingSectionFilter 对象拥有预先规定数量的段对象。陆续捕获到的段数据将被依次装入这些段对象。当存在空的段对象时过滤将持续进行。如果应用希望过滤不间断，需要使用 Section 对象的 setEmpty 方法，在新的对象到达之前把段对象标记为空。如果过滤操作遇到非空的段对象，就会停止。每次 startFiltering 被调用时，将由数组的开始处进行段的过滤。循环段过滤器第一次被创建时，所有的段设为空，在此之后由应用程序将其清空。启动循环段过滤器将不再清空段。

B.4.7.1 方法

B.4.7.1.1 getSections

原型：public org.davic.mpeg.sections.Section[] getSections()

描述：获取本过滤器所过滤到的所有段数据。

参数：无。
返回：无。
返回：org.davic.mpeg.sections.Section 对象数组，表示本过滤器所过滤到的段数据。
org.davic.mpeg.sections.Section 对象数组始终充满数据，由应用程序负责检查哪些数据是有效的。在新的过滤数据未到之前，重复调用此方法将会得到相同的结果。

B.4.8 事件org.davic.mpeg.sections.SectionFilterEvent

原型：public class org.davic.mpeg.sections.SectionFilterEvent
extends java.util.EventObject
描述：段过滤事件，一组本包定义的段过滤事件的基类。

B.4.8.1 方法

B.4.8.1.1 getSource

原型：public java.lang.Object getSource()
描述：获取产生此事件的 SectionFilter 对象。
重写： java.util.EventObject 类的 getSource()方法。
参数：无。
返回： org.davic.mpeg.sections.SectionFilter 对象，表示产生此事件的源。

B.4.8.1.2 getAppData

原型：public java.lang.Object getAppData()
描述：获取通过 SectionFilter 对象的 startFiltering()方法传递的应用数据。
参数：无。
返回：java.lang.Object 对象，表示附加的应用数据。

B.4.9 事件org.davic.mpeg.sections.SectionAvailableEvent

原型：public class org.davic.mpeg.sections.SectionAvailableEvent
extends org.davic.mpeg.sections.SectionFilterEvent
描述：段数据可用事件，继承 org.davic.mpeg.sections.SectionFilterEvent 类，报告过滤到了一个完整的段。当一个符合过滤条件的段成功地从传送流中过滤完毕，由
org.davic.mpeg.sections.SimpleSectionFilter、
org.davic.mpeg.sections.TableSectionFilter 或
org.davic.mpeg.sections.RingSectionFilter 对象产生该事件。

B.4.10 事件org.davic.mpeg.sections.VersionChangeDetectedEvent

原型：public class org.davic.mpeg.sections.VersionChangeDetectedEvent
extends org.davic.mpeg.sections.SectionFilterEvent
描述：段过滤版本变更事件，继承 org.davic.mpeg.sections.SectionFilterEvent 类。由 org.davic.mpeg.sections.TableSectionFilter 产生此事件，报告接收到的段与前一个段版本不一致。每次过滤操作只可能产生一次该事件。具有不同版本的段将被丢弃。

B.4.10.1 方法

B.4.10.1.1 getOriginalVersion

原型：public int getOriginalVersion()
描述：获取段数据的旧版本号。
参数：无。
返回：int 型，表示段数据的旧版本号。

B.4.10.1.2 getNewVersion

原型：public int getNewVersion()
描述：获取段数据的新版本号。
参数：无。
返回：int 型，表示段数据的新版本号。

B.4.11 事件org.davic.mpeg.sections.EndOfFilteringEvent

原型：public class org.davic.mpeg.sections.EndOfFilteringEvent extends
org.davic.mpeg.sections.SectionFilterEvent
描述：段过滤结束事件，继承 org.davic.mpeg.sections.SectionFilterEvent 类，报告段过滤结束。当过滤结束时，由 org.davic.mpeg.sections.RingSectionFilter 和 org.davic.mpeg.sections.TableSectionFilter 对象产生该事件，org.davic.mpeg.sections.SimpleSectionFilter 过滤器结束时不产生此事件。

B.4.12 事件org.davic.mpeg.sections.IncompleteFilteringEvent

原型：public class org.davic.mpeg.sections.IncompleteFilteringEvent
extends org.davic.mpeg.sections.EndOfFilteringEvent
描述：段过滤不完整事件，继承自 org.davic.mpeg.sections.EndOfFilteringEvent 类。用于报告由 org.davic.mpeg.sections.TableSectionFilter 产生的过滤操作结束。当 API 发现段参数未被完全定义时产生此事件，这会导致阻塞过滤器或与 MPEG-2 不兼容的结果。

B.4.13 事件org.davic.mpeg.sections.TimeOutEvent

原型：public class org.davic.mpeg.sections.TimeOutEvent extends
org.davic.mpeg.sections.EndOfFilteringEvent
描述：段过滤超时事件，继承自 org.davic.mpeg.sections.EndOfFilteringEvent 类。段过滤操作超时时产生该事件：
——对于 org.davic.mpeg.sections.SimpleSectionFilter，若在规定的时间内没有过滤到段数据则产生此事件；
——对于 org.davic.mpeg.sections.TableSectionFilter，若在规定的时间内没有过滤到整个表则产生此事件；
——对于 org.davic.mpeg.sections.RingSectionFilter，若在最后一个段成功过滤后又超过了规定的时间，则产生此事件。

B.4.14 事件org.davic.mpeg.sections.FilterResourcesAvailableEvent

原型：public class org.davic.mpeg.sections.FilterResourcesAvailableEvent
extends org.davic.resources.ResourceStatusEvent
描述：过滤器资源可用事件，继承 org.davic.resources.ResourceStatusEvent 类。此事件表明对段过滤组有足够的段过滤器资源。例如：如果创建的一个段过滤器组具有 4 个过滤器，则事件发

生时至少有 4 个过滤器是可用的。注意，当应用试图再次连接段过滤器组时它们可能不再可用。故此事件对尝试再次连接段过滤器组的应用程序是一种有用的提示。此事件只在 org.davic.mpeg.sections.ForcedDisconnectedEvent 事件之后和应用再次成功地连接段过滤器组之前发生。

B.4.15 事件org.davic.mpeg.sections.ForcedDisconnectedEvent

原型：public class org.davic.mpeg.sections.ForcedDisconnectedEvent
extends org.davic.resources.ResourceStatusEvent

描述：段过滤器组与传送流强制断开事件，继承自 org.davic.resources.ResourceStatusEvent 类。报告原来可用的传送流不再可用，或者段过滤器资源已从连接的段过滤器组中去除。在第二种情况下，除产生本事件之外，org.davic.resources.ResourceClient 的 notifyRelease()方法也同时被调用。

B.4.15.1 方法

B.4.15.1.1 getSource

原型：public java.lang.Object getSource()

描述：获取产生该事件的 org.davic.mpeg.sections.SectionFilterGroup 对象。

重写：ResourceStatusEvent 类的 getSource()方法。

返回： SectionFilterGroup 对象，表示产生该事件的段过滤器组。

B.4.16 异常org.davic.mpeg.sections.SectionFilterException

原型：public class org.davic.mpeg.sections.SectionFilterException extends
java.lang.Exception

描述：段过滤异常，本包定义的一组段过滤异常的基类。

B.4.17 异常org.davic.mpeg.sections.ConnectionLostException

原型：public class org.davic.mpeg.sections.ConnectionLostException
extends org.davic.mpeg.sections.SectionFilterException

描述：连接丢失异常，继承 org.davic.mpeg.sections.SectionFilterException 类。此异常表明 org.davic.mpeg.sections.SectionFilterGroup 丢失了连接或资源，无法进行 startFiltering()方法调用。仅由处于丢失连接状态的 SectionFilterGroup 对象产生。

B.4.18 异常org.davic.mpeg.sections.FilteringInterruptedException

原型：public class org.davic.mpeg.sections.FilteringInterruptedException
extends org.davic.mpeg.sections.SectionFilterException

描述：过滤中断异常，继承 org.davic.mpeg.sections.SectionFilterException 类。此异常表明在过滤到数据之前过滤操作被中断。

B.4.19 异常org.davic.mpeg.sections.FilterResourceException

原型：public class org.davic.mpeg.sections.FilterResourceException
extends org.davic.mpeg.sections.SectionFilterException

描述：过滤器资源异常，继承 org.davic.mpeg.sections.SectionFilterException 类。此异常表明当 org.davic.mpeg.sections.SectionFilterGroup 处于连接或非连接状态时，资源不足，无法完成操作。

B.4.20 异常org.davic.mpeg.sections.IllegalFilterDefinitionException

原型：public class org.davic.mpeg.sections.IllegalFilterDefinitionException
extends org.davic.mpeg.sections.SectionFilterException
描述：非法过滤条件异常，继承 S org.davic.mpeg.sections.ectionFilterException 类。此异常表明过滤器的过滤条件定义无效。

B.4.21 异常org.davic.mpeg.sections.InvalidSourceException

原型：public class org.davic.mpeg.sections.InvalidSourceException
extends org.davic.mpeg.sections.SectionFilterException
描述：段数据源无效异常，继承 org.davic.mpeg.sections.SectionFilterException 类。

B.4.22 异常org.davic.mpeg.sections.NoDataAvailableException

原型：public class org.davic.mpeg.sections.NoDataAvailableException
extends org.davic.mpeg.sections.SectionFilterException
描述：段对象无可用数据异常，继承 org.davic.mpeg.sections.SectionFilterException 类。

B.5 URL封装模块

URL 封装模块提供了 URL 封装引用方法。
URL 封装模块概要见表 B.4。

表 B.4 URL 封装模块概要

类	
Locator	资源定位符类，将 URL 封装成定位符对象。
异常	
InvalidLocatorException	定位符无效异常。

B.5.1 类org.davic.net.Locator

原型：public class org.davic.net.Locator
描述：资源定位符类，将 URL 封装成定位符对象。

B.5.1.1 方法

B.5.1.1.1 Locator

原型：public Locator(java.lang.String url)
描述：构造方法，创建一个资源定位符对象。
参数：url - java.lang.String 对象，表示一个 URL 字符串。

B.5.1.1.2 hasMultipleTransformations

原型：public boolean hasMultipleTransformations()
描述：指示此定位符是否映射到多种传输相关的定位符格式。
参数：无。
返回：boolean 型，true 表示此定位符映射到多种传输相关的定位符格式，false 表示无。

B.5.1.1.3 toExternalForm

原型：public abstract java.lang.String toExternalForm()
描述：获取对应于定位符的 URL 字符串，若使用一个非空但无效的 URL 创建定位符实例，系统的行为取决于具体实现。
参数：无。
返回： java.lang.String 对象，表示本定位符对应的 URL 字符串。

B.5.1.1.4 toString

原型：public java.lang.String toString()
描述：获取 URL 字串。
重写： java.lang.Object 类的 toString()方法。
参数：无。
返回： java.lang.String 对象，表示 URL 字符串。

B.5.1.1.5 equals

原型：public boolean equals(Object obj)
描述：判断 obj 对象是否与本例相同。
重写： Object 类的 equals()方法。
参数：obj – Locator 对象，表示待比较的定位符。
返回：boolean 型，取值 true 表示 obj 对象与本实例相同，false 表示不相同。

B.5.2 异常org.davic.net.InvalidLocatorException

原型：public class org.davic.net.InvalidLocatorException extends java.lang.Exception
描述：定位符无效异常。当构造 Locator 对象的一个或多个参数无效时抛出此异常。

B.6 DVB定位符模块

DVB 定位符模块提供了访问 DVB 广播业务及其内容的引用方法。
DVB 定位符模块概要见表 B.5。

表 B.5 DVB 定位符模块概要

类	
DvbLocator	DVB 定位符类，将 DVB 格式的 URL 封装成定位符对象。
DvbNetworkBoundLocator	与网络绑定的 DVB 定位符类，此类对象唯一标识一个给定的实体和传送系统。

B.6.1 类org.davic.net.dvb.DvbLocator

原型：public class org.davic.net.dvb.DvbLocator extends org.davic.net.Locator

描述：DVB定位符类，将DVB格式的URL封装成定位符对象。

B.6.1.1 方法

B.6.1.1.1 DvbLocator

原型：public DvbLocator(int onid, int tsid) throws Org.davic.net.InvalidLocatorException

描述：构造方法，以“dvb://original_network_id.transport_stream_id”URL格式创建一个DVB定位符对象。

参数：onid - int型，表示原始网络标识(original_network_id)；
tsid - int型，表示传输流标识(transport_stream_id)。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.2 DvbLocator

原型：public DvbLocator(int onid, int tsid, int serviceid) throws
org.davic.net.InvalidLocatorException

描述：构造方法，以"dvb://original_network_id.transport_stream_id.service_id"URL格式创建一个DVB定位符对象。

参数：onid - int 型，表示原始网络标识(original_network_id)；
tsid - int 型，表示传输流标识(transport_stream_id)。若取值-1，则表示定位符不包含传输流标识(transport_stream_id)；
serviceid - int 型，表示业务标识(service_id)。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.3 DvbLocator

原型：public DvbLocator(int onid, int tsid, int serviceid, int eventid)
throws org.davic.net.InvalidLocatorException

描述：构造方法，以"dvb://original_network_id.transport_stream_id.service_id;event_id" URL格式创建一个DVB定位符对象。

参数：onid - int 型，表示原始网络标识(original_network_id)；
tsid - int 型，表示传输流标识(transport_stream_id)。若取值-1，则表示定位符不包含传输流标识(transport_stream_id)；
serviceid - int 型 ，表示业务标识(service_id)；
eventid - int 型 ，表示事件标识(event_id)。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.4 DvbLocator

原型：public DvbLocator(int onid, int tsid, int serviceid, int eventid, int componenttag)
throws org.davic.net.InvalidLocatorException

描述：构造方法，以"dvb://original_network_id.transport_stream_id.service_id.
component_tag;eventid"或"dvb://original_network_id.transport_stream_id.

service_id.component_tag"URL格式创建一个DVB定位符对象。

参数：onid - int 型，表示原始网络标识(original_network_id)；

tsid - int 型，表示传输流标识(transport_stream_id)。若取值-1，则表示定位符不包含传输流标识(transport_stream_id)；

serviceid - int 型，表示业务标识(service_id)；

eventid - int 型，表示事件标识(event_id)。若取值-1，则表示定位符不包含事件标识(event_id)；

componenttag - int 型，表示基本流组件标签（component_tag）。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.5 DvbLocator

原型：public DvbLocator(int onid, int tsid, int serviceid,
int eventid, int[] componenttags)
throws org.davic.net.InvalidLocatorException

描述：构造方法，以"dvb://original_network_id.transport_stream_id.service_id.component_tag{&component_tag};event_id"或"dvb://original_network_id.transport_stream_id.service_id.component_tag{&component_tag}"URL格式创建一个DVB定位符对象。

参数：onid - int 型，表示原始网络标识(original_network_id)；

tsid - int 型，表示传输流标识(transport_stream_id)。若取值-1，则表示定位符不包含传输流标识(transport_stream_id)；

serviceid - int 型，表示业务标识(service_id)；

eventid - int 型，表示事件标识(event_id)。若取值-1，则表示定位符不包含事件标识(event_id)；

componenttags - int 型数组，表示基本流组件标签(component_tag)数组。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.6 DvbLocator

原型：public DvbLocator(int onid, int tsid, int serviceid, int eventid,
int[] componenttags, java.lang.String filePath)
throws org.davic.net.InvalidLocatorException

描述：构造方法，以"dvb://original_network_id.transport_stream_id.service_id.component_tag{&component_tag};event_id/filepath"或"dvb://original_network_id.transport_stream_id.service_id.component_tag{&component_tag}/filepath" URL格式创建一个DVB定位符对象。

参数：onid - int 型，表示原始网络标识(original_network_id)；

tsid - int 型，表示传输流标识(transport_stream_id)。若取值-1，则表示定位符不包含传输流标识(transport_stream_id)；

serviceid - int 型 ，表示业务标识(service_id)；

eventid - int 型 ，表示事件标识(event_id)。若取值-1，则表示定位符不包含事件标识(event_id)；

componenttags - int 型数组，表示基本流组件标签(component_tag)数组；

filePath - java.lang.String 对象，表示文件路径字符串，包括起始的斜线。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围)，则抛出此异常。

B.6.1.1.7 DvbLocator

原型：public DvbLocator(java.lang.String url) throws org.davic.net.InvalidLocatorException

描述：构造方法，根据输入的字符串创建一个DVB定位符对象。

参数：url - java.lang.String对象，表示DVB URL字符串。

异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超出范围、不符合 DVB URL 格式等)，则抛出此异常。

B.6.1.1.8 getComponentTags

原型：public int[] getComponentTags()

描述：获取DVB定位符中包含的基本流组建标签数组信息。

参数：无。

返回：int型数组，表示基本流组建标签的数组，若定位符没有包含组件标签信息，数组长度为0。

B.6.1.1.9 getEventId

原型：public int getEventId()

描述：获取DVB定位符中包含的事件标识符信息。

参数：无。

返回：int型，表示事件标识，若无则返回-1。

B.6.1.1.10 getFilePath

原型：public java.lang.String getFilePath()

描述：获取DVB定位符中包含的文件名与路径信息。

参数：无。

返回：java.lang.String对象，表示路径字符串，包括开始的斜线。若定位符无路径信息，则返回null。

B.6.1.1.11 getOriginalNetworkId

原型：public int getOriginalNetworkId()

描述：获取DVB定位符中包含的原始网络标识符信息。

参数：无。

返回：int型，表示原始网络标识，若无则返回-1。

B.6.1.1.12 getTransportStreamId

原型：public int getTransportStreamId()

描述：获取DVB定位符中包含的传输流标识符信息。

参数：无。

返回：int型 ，表示传输流标识，若无则返回-1。

B.6.1.1.13 getServiceId

原型：public int getServiceId()

描述：获取DVB定位符中包含的业务标识符信息。
参数：无。
返回：int型，表示业务标识，若无则返回-1。

B.6.1.1.14 toExternalForm

原型：public java.lang.String toExternalForm()
描述：获取对应于定位符的URL字符串，若使用一个非空但无效的URL创建定位符实例，系统的行为取决于具体实现。实现Locator类的toExternalForm()方法。
参数：无。
返回： java.lang.String对象，表示本定位符对应的URL字符串。

B.6.2 类org.davic.net.dvb.DvbNetworkBoundLocator

原型：public class org.davic.net.dvb.DvbNetworkBoundLocator
extends org.davic.net.DvbLocator
描述：与网络绑定的DVB定位符类，此类对象唯一标识一个给定的实体和传送系统。例如：一个业务可能在卫星与地面网络中传输，DVBLocator对象可能相同，但DvbNetworkBoundLocator对象不同。

B.6.2.1 方法

B.6.2.1.1 DvbNetworkBoundLocator

原型：public DvbNetworkBoundLocator(org.davic.net.dvb.DvbLocator unboundLocator, int networkId)
throws org.davic.net.InvalidLocatorException
描述：构造方法，创建一个与网络绑定的DVB定位符对象。
参数：unboundLocator－org.davic.net.dvb.DvbLocator对象，表示未与传输网络绑定的DVB定位符；
networkId - int型，表示网络标识。
异常：org.davic.net.InvalidLocatorException - 若参数不能标识一个有效的定位符(例如：数值标识超过范围)，则抛出此异常。

B.6.2.1.2 getNetworkId

原型：public int getNetworkId()
描述：获取网络标识符。
参数：无。
返回：int型，表示网络标识。

B.7 广播协议处理模块

广播协议处理模块定义了与DVB广播协议处理相关的JAVA接口。
广播协议处理模块概要见表B.6。

表 B.6 广播协议处理模块概要

接口	
SICommonInformation	PSI/SI 公共信息接口，提供了获取 PSI/SI 公共特性的方法。
SINetwork	网络信息接口，提供了获取网络(network)信息的方法，每个 SINetwork 对象由 network_id 唯一标识。
SIBouquet	业务群信息接口，提供了获取业务群(bouquet)信息的方法，每个 SIBouquet 对象由 network_id、bouquet_id 唯一标识。
SIService	业务信息接口，提供了获取业务(service)信息的方法，每个 SIService 对象由 network_id、original_network_id、transport_stream_id 和 service_id 共同唯一标识。
SIEvent	节目事件信息接口，提供了获取事件(event)信息的方法，每个 SIEvent 对象由 network_id、original_network_id、transport_stream_id、service_id 和 event_id 共同唯一标识。
SITransportStream	传送流信息接口，提供了获取传送流(transport_stream)信息的方法，每个 SITransportStream 对象由 network_id、original_network_id 和 transport_stream_id 共同唯一标识。
SIElementaryStream	基本流信息接口，提供了获取基本流(elementary_stream)信息的方法，每个 SIElementaryStream 对象是由 network_id、original_network_id、transport_stream_id、service_id 和 component_tag(或 elementary_PID)共同唯一标识。
SITime	时间信息接口，提供了获取时间信息的方法，时间信息从 TDT 或 TOT 获得，每个 SITime 对象由 network_id 唯一标识。
SIDescriptor	描述符信息接口，提供了与描述符访问相关的方法。
SIRequest	PSI/SI 信息请求接口，描述了应用产生的一次 PSI/SI 信息检索请求，应用可以通过该对象取消该次请求。
SIRetrieveListener	SI 信息获取事件监听器，由应用程序实现。
SIUpdateListener	PSI/SI 表更新事件监听器，由应用程序实现。
SIDescriptorTag	描述符标签常量定义接口，取值详见 GB/T 28161—2011。
SIRunningStatus	广播业务(service)或节目(event)的运行状态常量定义接口，取值详见 GB/T 28161—2011。
SIServiceType	业务类型常量定义接口，取值详见 GB/T 28161—2011。
SIStreamType	流类型常量定义接口，取值详见 GB/T 17975.1—2010。
类	
SIDatabase	PSI/SI 信息数据库，提供了 DVB 模式下 PSI/SI 信息数据库的管理和操作方法，是应用获取 PSI/SI 信息的入口类。
SIRequestFailureType	PSI/SI 信息请求失败的原因。
事件	
SIRetrieveEvent	PSI/SI 信息请求事件，是本包定义的一组跟 PSI/SI 信息请求相关的事件的基类。一次 PSI/SI 信息请求只会产生一个这样的事件。
SISuccessRetrieveEvent	PSI/SI 信息请求成功事件，继承 SIRetrieveEvent 类。
SIFailureRetrieveEvent	PSI/SI 信息请求失败事件，继承 SIRetrieveEvent 类。
SIUpdateEvent	PSI/SI 表更新事件。
异常	
InvalidPeriodException	当指定的日期无效时，抛出该异常。

B.7.1 接口org.ngb.broadcast.dvb.si.SICommonInformation

原型：public interface org.ngb.broadcast.dvb.si.SICommonInformation
描述：PSI/SI 公共信息接口，提供了获取 PSI/SI 公共特性的方法。

B.7.1.1 常量域——PSI/SI对象类型

B.7.1.1.1 SI_BOUQUET

原型：public static final int SI_BOUQUET = 0
描述：SI 信息识别——业务群信息。

B.7.1.1.2 SI_NETWORK

原型：public static final int SI_NETWORK = 1
描述：SI 信息识别——网络信息。

B.7.1.1.3 SI_SERVICE

原型：public static final int SI_SERVICE = 2
描述：SI 信息识别——业务信息。

B.7.1.1.4 SI_TS

原型：public static final int SI_TS = 3
描述：SI 信息识别——传送流(TS)信息。

B.7.1.1.5 SI_ES

原型：public static final int SI_ES = 4
描述：SI 信息识别——基本流(ES)信息。

B.7.1.1.6 SI_EVENT

原型：public static final int SI_EVENT = 5
描述：SI 信息识别——事件(event)信息。

B.7.1.1.7 SI_TIME

原型：public static final int SI_TIME = 6
描述：SI 信息识别——时间信息。

B.7.1.2 方法

B.7.1.2.1 getType

原型：public int getType()
描述：获取实现本接口的 SI 对象类型。
参数：无。
返回：int 型，表示实现本接口的 SI 对象类型，取值详见
org.ngb.broadcast.dvb.si.SICommonInformation 接口的“PSI/SI 对象类型”常量域定义。

B.7.1.2.2 getSIDatabase

原型：public org.ngb.broadcast.dvb.si.SIDatabase getSIDatabase()
描述：获取实现本接口的SI对象所属的PSI/SI数据库。
参数：无。
返回：SIDatabase对象，表示实现本接口的SI对象所属的PSI/SI数据库。

B.7.2 接口org.ngb.broadcast.dvb.si.SINetwork

原型：public interface org.ngb.broadcast.dvb.si.SINetwork
extends org.ngb.broadcast.dvb.si.SICommonInformation
描述：网络信息接口，提供了获取网络(network)信息的方法，每个SINetwork对象由network_id唯一标识。接收终端有可能会同时接入多个广播网络，例如同时接入国标地面无线网络和有线网络，通过network_id区分网络对象。

B.7.2.1 方法

B.7.2.1.1 getNetworkID

原型：public int getNetworkID()
描述：获取网络标识（即NIT的network_id字段）。
参数：无。
返回：int型，表示单向广播网络对象的唯一标识（network_id）。

B.7.2.1.2 getNetworkName

原型：public java.lang.String getNetworkName()
描述：获取网络名称全称。
返回：java.lang.String对象，表示网络名称。

B.7.2.1.3 getShortNetworkName

原型：public java.lang.String getShortNetworkName()
描述：获取网络名称简称。
返回： java.lang.String对象，表示网络名称简称，若无网络名称简称则返回null。
示例：
“[0x86]Asterix[0x87] Digital Satellite TV Network”
网络名称全称：“Asterix Digital Satellite TV Network”。
网络名称简称：“Asterix”。

B.7.2.1.4 getServicesLocators

原型：public org.ngb.broadcast.dvb.si.DvbNetworkBoundLocator[] getServicesLocators()
描述：获取属于该网络的所有业务的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator对象数组，表示该网络下所有业务的定位符。若无，则返回的数组长度为0。

B.7.2.1.5 getService

原型：public org.ngb.broadcast.dvb.si.SIService getService

(org.davic.net.dvb.DvbLocator locator)

描述：根据指定的业务定位符，获取属于该网络的业务对象。

参数：locator - org.davic.net.dvb.DvbLocator 对象，表示业务的定位符。

返回：org.davic.net.dvb.si.SIService 对象，表示该网络下的业务对象。

B.7.3 接口org.ngb.broadcast.dvb.si.SIBouquet

原型：public interface org.ngb.broadcast.dvb.si.SIBouquet
extends org.ngb.broadcast.dvb.si.SICommonInformation

描述：业务群(bouquet)信息接口，每个 org.ngb.broadcast.dvb.si.SIBouquet 对象由 network_id 和 bouquet_id 共同唯一标识，即具有相同 bouquet_id 的业务群若出现在多个网络中，本部分规定应为这样的业务群创建多个 org.ngb.broadcast.dvb.si.SIBouquet 对象实例，通过 network_id 进行区分。

B.7.3.1 方法

B.7.3.1.1 getNetwork

原型：public org.ngb.broadcast.dvb.si.SINetwork getNetwork()

描述：获取本 org.ngb.broadcast.dvb.si.SIBouquet 对象所属的网络。

参数：无。

返回：org.ngb.broadcast.dvb.si.SINetwork 对象，表示本 org.ngb.broadcast.dvb.si.SIBouquet 对象所属的网络实例。

B.7.3.1.2 getNetworkID

原型：public int getNetworkID()

描述：获取业务群所属的网络标识。

参数：无。

返回：int 型，表示网络标识（network_id）。

B.7.3.1.3 getBouquetID

原型：public int getBouquetID()

描述：获取业务群标识（即 BAT 表的 bouquet_id 字段）。

参数：无。

返回：int 型，表示业务群标识（bouquet_id）。

B.7.3.1.4 getBouquetName

原型：public java.lang.String getBouquetName()

描述：获取业务群名称全称。

参数：无。

返回： java.lang.String 对象，表示业务群名称，若无可用信息，则返回 null。

B.7.3.1.5 getShortBouquetName

原型：public java.lang.String getShortBouquetName()

描述：获取业务群名称简称。

参数：无。

返回： java.lang.String 对象，表示业务群名称简称，若业务群名称简称不存在，则返回 null。

B.7.3.1.6 getService

原型：public SIService getService(org.davic.net.dvb.DvbLocator locator)
描述：根据指定的业务定位符，获取业务群中指定的业务。
参数：locator — org.davic.net.dvb.DvbLocator 对象，表示指定业务的定位符。
返回：SIService 对象，表示该业务群下的业务对象。若不存在，则返回 null。

B.7.3.1.7 getServicesLocators

原型：public DvbNetworkBoundLocator[] getServicesLocators()
描述：获取属于该业务群的所有业务的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator 对象数组，表示该业务群下所有业务的定位符。若无，则返回的数组长度为 0。

B.7.4 接口org.ngb.broadcast.dvb.si.SIService

原型：public interface org.ngb.broadcast.dvb.si.SIService
extends org.ngb.broadcast.dvb.si.SICommonInformation
描述：业务(service)信息接口，每个 org.ngb.broadcast.dvb.si.SIService 对象由 network_id、original_network_id、transport_stream_id 和 service_id 四要素共同唯一标识。在同一个网络中，一个 org.ngb.broadcast.dvb.si.SIService 对象由 original_network_id、transport_stream_id 和 service_id 三要素唯一确定，但一个业务有可能属于多个网络，本部分规定应为这样的业务创建多个org.ngb.broadcast.dvb.si.SIService对象,通过network_id进行区分，即在接收多网络信号的应用场景下，采用四要素唯一确定一个业务。

B.7.4.1 方法

B.7.4.1.1 getServiceName

原型：public java.lang.String getServiceName()
描述：获取业务名称全称。
参数：无。
返回： java.lang.String 对象，表示业务名称，若无可用信息，则返回 null。

B.7.4.1.2 getShortServiceName

原型：public java.lang.String getShortServiceName()
描述：获取业务名称简称。
参数：无。
返回： java.lang.String 对象，表示业务名称简称，若业务名称简称不存在，则返回 null。
示例：
The [0x86]**P**[0x87]ay [0x86]**M**[0x87]ovie [0x86]**C**[0x87]hannel
业务名称全称——“The Pay Movie Channel”。
业务名称简称——“PMC”。

B.7.4.1.3 getServiceProviderName

原型：public java.lang.String getServiceProviderName()
描述：获取业务提供者名称全称。
参数：无。
返回： java.lang.String 对象，表示广播业务提供者名称，若无可用信息，则返回 null。

B.7.4.1.4 getShortServiceProviderName

原型：public java.lang.String getShortServiceProviderName()
描述：获取业务提供者名称简称。
参数：无。
返回： java.lang.String 对象，表示业务提供者名称简称，若无简称信息，则返回 null。

B.7.4.1.5 getServiceType

原型：public int getServiceType()
描述：从 SDT 表的 service_descriptor 描述符中获取广播业务类型(service_type)。
参数：无。
返回：short 型，表示广播业务类型。取值见 SIServiceType 接口的“业务类型”常量域定义。

B.7.4.1.6 getChannelNumber

原型：public int getChannelNumber()
描述：获取业务的逻辑频道号。由系统自行决定逻辑频道号的获取方式（与运营商有关）。
参数：无。
返回：int 型，表示业务的逻辑频道号。
注：逻辑频道号在同一个网络里面唯一。

B.7.4.1.7 getDvbLocator

原型：public org.davic.net.dvb.DvbNetworkBoundLocator getDvbLocator()
描述：获取业务对象的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator 对象，表示此业务对象的定位符。

B.7.4.1.8 getNetworkID

原型：public int getNetworkID()
描述：获取业务对象所属的网络标识。
参数：无。
返回：int 型，表示网络标识（network_id）。

B.7.4.1.9 getOriginalNetworkID

原型：public int getOriginalNetworkID()
描述：获取业务对象所属的原始网络标识。
参数：无。
返回：int 型，表示业务对象所属的原始网络标识。

B.7.4.1.10 getTransportStreamID

原型：public int getTransportStreamID()
描述：获取业务对象所属的传送流标识。
参数：无。
返回：int 型，表示业务对象所属的传送流标识。

B.7.4.1.11 getServiceID

原型：public int getServiceID()
描述：获取业务标识。
参数：无。
返回：int 型，表示业务标识。

B.7.4.1.12 getNetwork

原型：public org.ngb.broadcast.dvb.si.SINetwork getNetwork()
描述：获取本 org.ngb.broadcast.dvb.si.SIService 对象所属的 SINetwork 对象。
参数：无。
返回：SINetwork 对象，表示本 SIService 对象所属的 SINetwork 对象。

B.7.4.1.13 getTransportStream

原型：public org.ngb.broadcast.dvb.si.SITransportStream getTransportStream()
描述：获取本 org.ngb.broadcast.dvb.si.SIService 对象所属的 org.ngb.broadcast.dvb.si.SITransportStream 对象。
参数：无。
返回：org.ngb.broadcast.dvb.si.SITransportStream 对象，表示本 org.ngb.broadcast.dvb.si.SIService 对象所属的 org.ngb.broadcast.dvb.si.SITransportStream 对象。

B.7.4.1.14 getBouquets

原型：public org.ngb.broadcast.dvb.si.SIBouquet[] getBouquets()
描述：获取本 org.ngb.broadcast.dvb.si.SIService 对象所属的所有 org.ngb.broadcast.dvb.si.SIBouquet 对象。
参数：无。
返回：org.ngb.broadcast.dvb.si.SIBouquet 对象数组，表示本 org.ngb.broadcast.dvb.si.SIService 对象所属的所有 org.ngb.broadcast.dvb.si.SIBouquet 对象。

B.7.4.1.15 getFreeCAMode

原型：public boolean getFreeCAMode()
描述：从 SDT 表中获取 free_CA_mode 标志。
参数：无。
返回：boolean 型，表示 free_CA_mode 标志，取值 true 表示业务被加扰，接收由 CA 控制；false 表示业务未被加扰，自由接收。

B.7.4.1.16 getPcrPID

原型：public int getPcrPID()
描述：获取业务所参考的 PCR 的 TS_PID（即 PMT 携带的 PCR_PID 字段）。

参数：无。

返回：int 型，表示广播业务的 PCR_PID。

B.7.4.1.17 getEITPresentFollowingFlag

原型：public boolean getEITPresentFollowingFlag()

描述：从 SDT 表中获取 EIT_present_following_flag 标志。

参数：无。

返回：boolean 型，表示 EIT_present_follwing_flag 标志，取值 true 表示该业务有 EIT 当前/后续信息，false 表示该业务无 EIT 当前/后续信息。

B.7.4.1.18 getEITScheduleFlag

原型：public booelan getEITScheduleFlag()

描述：从 SDT 表中获取 EIT_schedule_flag 标志。

参数：无。

返回：boolean 型，表示 EIT_schedule_flag 标志，取值 true 表示该业务有 EIT 时间表信息，false 表示该业务无 EIT 时间表信息。

B.7.4.1.19 retrieveElementaryStreams

原型：public SIRequest retrieveElementaryStreams(java.lang.Object appData,
org.ngb.broadcast.dvb.si.SIRetrieveListener listener,
short[] someDescriptorTags)
throws java.lang.IllegalArgumentException

描述：异步方法，获取广播业务中包含的基本流信息。

——当接口成功完成获取动作时，将向监听者发送一个 org.ngb.broadcast.dvb.si.SISuccessRetrieveEvent 事件，通过该事件对象的 getResult()方法返回 org.ngb.broadcast.dvb.si.SICommonInformation 对象数组，从该数组可以获取所有检索到的基本流对象(SIElementaryStream)；

——若没有检索到符合条件的基本流对象，应向监听者发送 org.ngb.broadcast.dvb.si.SIFailureRetrieveEvent 事件，从该事件对象中可以获得失败原因 org.ngb.broadcast.dvb.si.SIRequestFailureType。

参数：appData - java.lang.Object 对象，表示应用提供的附加信息，当检索动作完成时，这个对象将由系统传递给监听接口。应用可以使用该对象进行内部通信，如果应用不需要任何附加信息，该参数可以置为 null；

listener - org.ngb.broadcast.dvb.si.SIRetrieveListener 对象，用以接收检索通知事件；

someDescriptorTags - short 型数组，表示一组应用关心的描述符的标签值。接口应检索出应用所关心的所有描述符信息，以 org.ngb.broadcast.dvb.si.SIDescriptor 对象表示。

——如果数组只包含一个元素且值为-1，表明应用关心 PMT 表中的所有描述符；

——如果数组对象为 null，表明应用不关心任何描述符。

返回：SIRequest 对象，表示一次信息检索请求会话。

异常：java.lang.IllegalArgumentException — 若输入参数无效，则抛出该异常。

B.7.5 接口org.ngb.broadcast.dvb.si.SITransportStream

原型：public interface org.ngb.broadcast.dvb.si.SITransportStream

extends org.ngb.broadcast.dvb.si.SICommonInformation

描述：传送流(transport_stream)信息接口。每个 org.ngb.broadcast.dvb.si.SITransportStream 对象由 network_id、original_network_id 和 transport_stream_id 三者共同唯一确定。在同一个网络中，传送流由 original_network_id 和 transport_stream_id 两者唯一确定；但一个传送流有可能会在多个网络传输，本部分规定应为这样的传送流创建多个 org.ngb.broadcast.dvb.si.SITransportStream 实例，通过 network_id 区分，即在接收多网络信号的应用场景下，通过 network_id、original_network_id 和 transport_stream_id 三者共同唯一确定一个 org.ngb.broadcast.dvb.si.SITransportStream 对象。

B.7.5.1 方法

B.7.5.1.1 getNetworkID

原型：public int getNetworkID()

描述：获取传送流对象所属的网络标识。

参数：无。

返回：int 型，表示传送流对象所属的网络标识。

B.7.5.1.2 getOriginalNetworkID

原型：public int getOriginalNetworkID()

描述：获取实现该接口的 SI 对象所属的原始网络标识。

参数：无。

返回：int 型，表示实现该接口的 SI 对象所属的原始网络标识。

B.7.5.1.3 getTransportStreamID

原型：public int getTransportStreamID()

描述：获取实现该接口的 SI 对象所属的传送流标识。

参数：无。

返回：int 型，表示实现该接口的 SI 对象所属的传送流标识。

B.7.5.1.4 getNetwork

原型：public org.ngb.broadcast.dvb.si.SINetwork getNetwork()

描述：获取本 org.ngb.broadcast.dvb.si.SITransportStream 对象所属的 SINetwork 对象。

参数：无。

返回：org.ngb.broadcast.dvb.si.SINetwork 对象，表示本 org.ngb.broadcast.dvb.si.SITransportStream 对象所属的 SINetwork 对象。

B.7.6 接口org.ngb.broadcast.dvb.si.SIElementaryStream

原型：public interface org.ngb.broadcast.dvb.si.SIElementaryStream
extends org.ngb.broadcast.dvb.si.SICommonInformation

描述：广播业务的基本流(elementary_stream)信息接口。
每个 org.ngb.broadcast.dvb.si.SIElementaryStream 对象是由 network_id、original_network_id、transport_stream_id、service_id 和 component_tag（或 elementary_PID）标识符共同唯一确定。一个基本流有可能属于多个业务，本部分规定应为这

样的基本流创建多个org.ngb.broadcast.dvb.si.SIElementaryStream对象,通过network_id、original_network_id、transport_stream_id、service_id和component_tag（或elementary_PID）标识符共同唯一确定。

B.7.6.1 方法

B.7.6.1.1 getComponentTag

原型：public byte getComponentTag()
描述：获取基本流组件标签。
参数：无。
返回：byte 型，表示基本流组件标签。若该基本流未携带 stream_identifier_descriptor，则其组件标签值缺省为-2。

B.7.6.1.2 getElementaryPID

原型：public short getElementaryPID()
描述：获取承载该基本流的传送流包标识(TS_PID)。
参数：无。
返回：short 型，表示承载该基本流的传送流包标识。

B.7.6.1.3 getNetworkID

原型：public int getNetworkID()
描述：获取基本流对象所属的网络标识。
参数：无。
返回：int 型，表示网络标识（network_id）。

B.7.6.1.4 getOriginalNetworkID

原型：public int getOriginalNetworkID()
描述：获取基本流对象所属的原始网络标识。
参数：无。
返回：int 型，表示实现该接口的基本流对象所属的原始网络标识。

B.7.6.1.5 getTransportStreamID

原型：public int getTransportStreamID()
描述：获取基本流对象所属的传送流标识。
参数：无。
返回：int 型，表示基本流对象所属的传送流标识。

B.7.6.1.6 getServiceID

原型：public int getServiceID()
描述：获取基本流对象所属的业务标识。
参数：无。
返回：int 型，表示基本流对象所属的业务标识。

B.7.6.1.7 getStreamType

原型：public byte getStreamType()
描述：获取基本流类型。
参数：无。
返回：byte 型，表示基本流类型,取值详见 org.ngb.broadcast.dvb.si.SIStreamType 接口的流类型常量域定义。

B.7.6.1.8 getService

原型：public SIService getService()
描述：获取本 org.ngb.broadcast.dvb.si.SIElementaryStream 对象所属的 org.ngb.broadcast.dvb.si.SIService 对象。
参数：无。
返回：org.ngb.broadcast.dvb.si.SIService 对象，
表示本 org.ngb.broadcast.dvb.si.SIElementaryStream 对象所属的 SIService 对象。

B.7.7 接口org.ngb.broadcast.dvb.si.SIEvent

原型：public interface org.ngb.broadcast.dvb.si.SIEvent
extends org.ngb.broadcast.dvb.si.SICommonInformation
描述：节目事件(event)信息接口，每个 org.ngb.broadcast.dvb.si.SIEvent 对象由 network_id、original_network_id、transport_stream_id、service_id 和 event_id 共同唯一标识。一个节目事件有可能属于多个业务，本部分规定应为这样的节目事件创建多个 org.ngb.broadcast.dvb.si.SIEvent 对象，通过 network_id、original_network_id、transport_stream_id、service_id 和 event_id 共同唯一标识。

B.7.7.1 方法

B.7.7.1.1 getNetworkID

原型：public int getNetworkID()
描述：获取基本流对象所属的网络标识。
参数：无。
返回：int 型，表示网络标识（network_id）。

B.7.7.1.2 getOriginalNetworkID

原型：public int getOriginalNetworkID()
描述：获取事件(event)对象所属的原始网络标识。
参数：无。
返回：int 型，表示事件(event)对象所属的原始网络标识。

B.7.7.1.3 getTransportStreamID

原型：public int getTransportStreamID()
描述：获取事件(event)对象所属的传送流标识。
参数：无。
返回：int 型，表示事件(event)对象所属的传送流标识。

B.7.7.1.4 getServiceID

原型：public int getServiceID()
描述：获取事件(event)对象所属的业务标识。
参数：无。
返回：int 型，表示事件(event)对象所属的业务标识。

B.7.7.1.5 getEventID

原型：public int getEventID()
描述：获取事件标识符。
参数：无。
返回：int 型，表示事件标识符。

B.7.7.1.6 getDvbLocator

原型：public org.davic.net.dvb.DvbNetworkBoundLocator getDvbLocator()
描述：获取事件(event)对象的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator 对象，表示事件(event)对象的定位符。

B.7.7.1.7 getService

原型：public org.ngb.broadcast.dvb.si.SIService getService()
描述：获取本 org.ngb.broadcast.dvb.si.SIEvent 对象所属的 org.ngb.broadcast.dvb.si.SIService 对象。
参数：无。
返回：org.ngb.broadcast.dvb.si.SIService 对象，表示本 org.ngb.broadcast.dvb.si.SIEvent 对象所属的 org.ngb.broadcast.dvb.si.SIService 对象。

B.7.7.1.8 getNibbles

原型：public byte[] getNibbles()
描述：获取节目内容分类信息，该信息来自于 EIT 中的内容描述符(content_descriptor)。
参数：无。
返回：byte 型数组，表示与本节目事件关联的内容分类信息。若内容描述符(content_descriptor)不存在，则返回 null。
——byte[0] - 高 4 位表示二级内容分类（content_nibble_level_2）；低 4 位表示一级内容分类（content_nibble_level_1）；
——byte[1] - 高 4 位表示二级自定义分类（user_nibble_level_2）；低 4 位表示一级自定义分类（user_nibble_level_1）。

B.7.7.1.9 getStartTime

原型：public java.util.Date getStartTime()
描述：获取事件的起始时间。
参数：无。
返回： java.util.Date 对象，表示广播节目事件的起始时间。

B.7.7.1.10 getDuration

原型：public long getDuration()
描述：获取事件的持续时间。
参数：无。
返回：long 型，表示事件的持续时间，单位为秒。

B.7.7.1.11 getEndTime

原型：public Date getEndTime()
描述：获取事件的终止时间。
参数：无。
返回： java.util.Date 对象，表示事件的终止时间。

B.7.7.1.12 getEventName

原型： String getEventName()
描述：获取事件名称全称(event_name)。
参数：无。
返回： java.lang.String 对象，表示事件名称全称。

B.7.7.1.13 getShortEventName

原型：public java.lang.String getShortEventName()
描述：获取事件名称简称。
参数：无。
返回： java.lang.String 对象，表示事件名称简称。若无简称信息，应返回 null。

B.7.7.1.14 getEventDescription

原型：public java.lang.String getEventDescription()
描述：获取事件的描述。
参数：无。
返回： java.lang.String 对象，表示事件的描述。

B.7.7.1.15 getFreeCAMode

原型：public boolean getFreeCAMode()
描述：获取广播节目事件加扰标志（free_CA_mode）。
参数：无。
返回：boolean 型，表示广播节目事件是否被加扰，取值 true 表示该事件被加扰，接收由 CA 控制；false 表示未加扰，自由接收。

B.7.7.1.16 getRunningStatus

原型：public byte getRunningStatus()
描述：获取事件的运行状态信息。
参数：无。
返回：byte 型，表示事件的运行状态，取值详见 org.ngb.broadcast.dvb.si.SIRunningStatus 接口的“运行状态”常量域定义。

B.7.8 接口org.ngb.broadcast.dvb.si.SITime

原型：public interface org.ngb.broadcast.dvb.si.SITime
extends org.ngb.broadcast.dvb.si.SICommonInformation
描述：时间信息接口，提供了获取时间信息的方法，时间信息从 TDT 或 TOT 获得。

B.7.8.1 方法

B.7.8.1.1 getUTCTime

原型： java.util.Date getUTCTime()
描述：获取 UTC 时间日期。
参数：无。
返回： java.util.Date 对象，表示 UTC 时间，从 TDT 或 TOT 中获取信息。

B.7.9 接口org.ngb.broadcast.dvb.si.SIDescriptor

原型：public interface org.ngb.broadcast.dvb.si.SIDescriptor
描述：描述符信息接口。描述符由三部分构成,见图 B.3。
——标签（descriptor_tag） - 唯一标识一个描述符；
——内容长度（descriptor_length） - 指示内容部分的字节个数；
——内容：字节数组，具体语法语义与标签类型相关。

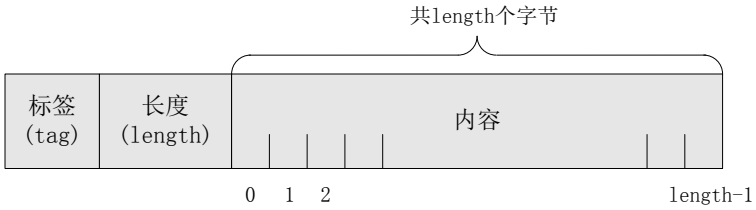


图 B.3 SIDescriptor 结构示意图

B.7.9.1 方法

B.7.9.1.1 getByteAt

原型：public byte getByteAt(int index) throws java.lang.IndexOutOfBoundsException
描述：获取内容中指定索引位置的字节。
参数：index — int 型，描述符内容位置索引值，位置 0 对应 descriptor_length 字段后的第一个字节。
返回：byte 型，表示指定位置的字节数据。
异常：java.lang.IndexOutOfBoundException - 若参数 index 小于 0 或 index 大于等于 descriptor_length，则抛出此异常。

B.7.9.1.2 getContent

原型：public byte[] getContent()
描述：获取描述符的内容。
参数：无。
返回：byte 数组。表示描述符的内容部分，即 descriptor_length 字段后的所有字节。

B.7.9.1.3 getContentLength

原型：public short getContentLength()
描述：获取描述符内容长度。
参数：无。
返回：short 型，表示描述符内容的长度，即描述符长度（descriptor_length）字段值，单位为字节。

B.7.9.1.4 getTag

原型：public short getTag()
描述：获取描述符标签（descriptor_tag）。
参数：无。
返回：short 型，表示描述符标签字段值，取值见 SIDescriptorTag 接口的“描述符标签”常量域定义。

B.7.10 接口org.ngb.broadcast.dvb.si.SIRequest

原型：public interface org.ngb.broadcast.dvb.si.SIRequest
描述：PSI/SI 信息请求接口，描述了应用产生的一次 PSI/SI 信息检索请求。应用可以通过该对象取消该次请求。

B.7.10.1 方法

B.7.10.1.1 cancelRequest

原型：public void cancelRequest()
描述：取消本次请求动作。
参数：无。
返回：无。

B.7.11 接口org.ngb.broadcast.dvb.si.SIRetrieveListener

原型：public interface org.ngb.broadcast.dvb.si.SIRetrieveListener extends java.util.EventListener
描述：SI 信息获取事件监听器，由应用程序实现。

B.7.11.1 方法

B.7.11.1.1 postEvent

原型：public void postEvent(org.ngb.broadcast.dvb.si.SIRetrieveEvent event)
描述：发送 SI 信息获取事件。
参数：event - org.ngb.broadcast.dvb.si.SIRetrieveEvent 对象，表示 SI 信息获取事件。应用应调用 instanceOf 方法进一步判断 event 的对象类型，可能为
org.ngb.broadcast.dvb.si.SISuccessRetrieveEvent 对象或
org.ngb.broadcast.dvb.si.SIFailureRetrieveEvent 对象。
返回：无。

B.7.12 接口org.ngb.broadcast.dvb.si.SIUpdateListener

原型：public interface org.ngb.broadcast.dvb.si.SIUpdateListener extends
java.util.EventListener
描述：PSI/SI 表更新事件监听器，由应用程序实现。

B.7.12.1 方法

B.7.12.1.1 postEvent

原型：public void postEvent(org.ngb.broadcast.dvb.si.SIUpdateEvent event)
描述：PSI/SI 更新事件回调方法，用来获取发生变化的 PSI/SI 对象。
参数：event - org.ngb.broadcast.dvb.si.SIUpdateEvent 对象，表示 PSI/SI 信息发生变化的事件。
返回：无。

B.7.13 接口org.ngb.broadcast.dvb.si.SIDescriptorTag

原型：public interface org.ngb.broadcast.dvb.si.SIDescriptorTag
描述：描述符标签常量接口，取值详见 GB/T 28161—2011。

B.7.13.1 常量域——描述附标签

B.7.13.1.1 TAG_NETWORK_NAME

原型：public static final short TAG_NETWORK_NAME = 64
描述：网络名称描述符标签。

B.7.13.1.2 TAG_SERVICE_LIST

原型：public static final short TAG_SERVICE_LIST = 65
描述：业务列表描述符标签。

B.7.13.1.3 TAG_STUFFING

原型：public static final short TAG_STUFFING = 66
描述：填充描述符标签。

B.7.13.1.4 TAG_SATELLITE_DELIVERY_SYSTEM

原型：public static final short TAG_SATELLITE_DELIVERY_SYSTEM = 67
描述：卫星传送系统描述符标签。

B.7.13.1.5 TAG_CABLE_DELIVERY_SYSTEM

原型：public static final short TAG_CABLE_DELIVERY_SYSTEM = 68
描述：有线传送系统描述符标签。

B.7.13.1.6 TAG_VBI_DATA

原型：public static final short TAG_VBI_DATA = 69
描述：VBI 数据描述符标签。

B.7.13.1.7 TAG_TELETEXT

原型：public static final short TAG_TELETEXT = 70

描述：图文电视描述符标签。

B.7.13.1.8 TAG_BOUQUET_NAME

原型：public static final short TAG_BOUQUET_NAME = 71
描述：业务群名称描述符标签。

B.7.13.1.9 TAG_SERVICE

原型：public static final short TAG_SERVICE = 72
描述：业务描述符标签。

B.7.13.1.10 TAG_COUNTRY_AVAILABILITY

原型：public static final short TAG_COUNTRY_AVAILABILITY = 73
描述：国家/地区业务准用描述符。

B.7.13.1.11 TAG_LINKAGE

原型：public static final short TAG_LINKAGE = 74
描述：链接描述符标签。

B.7.13.1.12 TAG_NVOD_REFERENCE

原型：public static final short TAG_NVOD_REFERENCE = 75
描述：准视频点播参考描述符标签。

B.7.13.1.13 TAG_TIME_SHIFTED_SERVICE

原型：public static final short TAG_TIME_SHIFTED_SERVICE = 76
描述：时移业务描述符标签。

B.7.13.1.14 TAG_SHORT_EVENT

原型：public static final short TAG_SHORT_EVENT = 77
描述：短事件描述符标签。

B.7.13.1.15 TAG_EXTENDED_EVENT

原型：public static final short TAG_EXTENDED_EVENT = 78
描述：扩展事件描述符标签。

B.7.13.1.16 TAG_TIME_SHIFTED_EVENT

原型：public static final short TAG_TIME_SHIFTED_EVENT = 79
描述：时移事件描述符标签。

B.7.13.1.17 TAG_COMPONENT

原型：public static final short TAG_COMPONENT = 80
描述：组件描述符标签。

B.7.13.1.18 TAG_MOSAIC

原型：public static final short TAG_MOSAIC = 81
描述：马赛克描述符标签。

B.7.13.1.19 TAG_STREAM_IDENTIFIER

原型：public static final short TAG_STREAM_IDENTIFIER = 82
描述：流标识描述符标签。

B.7.13.1.20 TAG_CA_IDENTIFIER

原型：public static final short TAG_CA_IDENTIFIER = 83
描述：条件接收标识描述符标签。

B.7.13.1.21 TAG_CONTENT

原型：public static final short TAG_CONTENT = 84
描述：内容描述符标签。

B.7.13.1.22 TAG_PARENTAL_RATING

原型：public static final short TAG_PARENTAL_RATING = 85
描述：家长分级描述符标签。

B.7.13.1.23 TAG_VBI_TELETEXT

原型：public static final short TAG_VBI_TELETEXT = 86
描述：VBI 图文电视描述符。

B.7.13.1.24 TAG_TELEPHONE

原型：public static final short TAG_TELEPHONE = 87
描述：电话描述符标签。

B.7.13.1.25 TAG_LOCAL_TIME_OFFSET

原型：public static final short TAG_LOCAL_TIME_OFFSET = 88
描述：本地时间偏移描述符标签。

B.7.13.1.26 TAG_SUBTITLING

原型：public static final short TAG_SUBTITLING = 89
描述：字幕描述符标签。

B.7.13.1.27 TAG_TERRESTRIAL_DELIVERY_SYSTEM

原型：public static final short TAG_TERRESTRIAL_DELIVERY_SYSTEM = 90
描述：地面传送系统描述符标签。

B.7.13.1.28 TAG_MULTILINGUAL_NETWORK_NAME

原型：public static final short TAG_MULTILINGUAL_NETWORK_NAME = 91
描述：多语种网络名称描述符标签。

B.7.13.1.29 TAG_MULTILINGUAL_BOUQUET_NAME

原型：public static final short TAG_MULTILINGUAL_BOUQUET_NAME = 92
描述：多语种业务群描述符标签。

B.7.13.1.30 TAG_MULTILINGUAL_SERVICE_NAME

原型：public static final short TAG_MULTILINGUAL_SERVICE_NAME = 93
描述：多语种业务名称描述符标签。

B.7.13.1.31 TAG_MULTILINGUAL_COMPONENT

原型：public static final short TAG_MULTILINGUAL_COMPONENT = 94
描述：多语种组件描述符标签。

B.7.13.1.32 TAG_PRIVATE_DATA_SPECIFIER

原型：public static final short TAG_PRIVATE_DATA_SPECIFIER = 95
描述：专用数据说明符描述符标签。

B.7.13.1.33 TAG_SERVICE_MOVE

原型：public static final short TAG_SERVICE_MOVE = 96
描述：业务转移描述符标签。

B.7.13.1.34 TAG_SHORT_SMOOTHING_BUFFER

原型：public static final short TAG_SHORT_SMOOTHING_BUFFER = 97
描述：短平滑缓冲区描述符标签。

B.7.13.1.35 TAG_FREQUENCY_LIST

原型：public static final short TAG_FREQUENCY_LIST = 98
描述：频率列表描述符标签。

B.7.13.1.36 TAG_PARTIAL_TRANSPORT_STREAM

原型：public static final short TAG_PARTIAL_TRANSPORT_STREAM = 99
描述：频率列表描述符标签。

B.7.13.1.37 TAG_DATA_BROADCAST

原型：public static final short TAG_DATA_BROADCAST = 100
描述：数据广播描述符标签。

B.7.13.1.38 TAG_CA_SYSTEM

原型：public static final short TAG_CA_SYSTEM = 101
描述：CA 系统描述符标签。

B.7.13.1.39 TAG_DATA_BROADCAST_ID

原型：public static final short TAG_DATA_BROADCAST_ID = 102
描述：数据广播标识描述符标签。

B.7.13.1.40 TAG_TRANSPORT_STREAM

原型：public static final short TAG_TRANSPORT_STREAM = 103
描述：传送流描述符标签。

B.7.13.1.41 TAG_DSNG

原型：public static final short TAG_DSNG = 104
描述：数字卫星新闻采集描述符标签。

B.7.13.1.42 TAG_PDC

原型：public static final short TAG_PDC = 105
描述：节目传送控制描述符标签。

B.7.13.1.43 TAG_AC_3

原型：public static final short TAG_AC_3 = 106
描述：AC3 描述符标签。

B.7.13.1.44 TAG_ANCILLARY_DATA

原型：public static final short TAG_ANCILLARY_DATA = 107
描述：附属数据描述符标签。

B.7.13.1.45 TAG_ANNOUNCEMENT_SUPPORT

原型：public static final short TAG_ANNOUNCEMENT_SUPPORT = 110
描述：公告支持描述符标签。

B.7.14 接口org.ngb.broadcast.dvb.si.SIRunningStatus

原型：public interface org.ngb.broadcast.dvb.si.SIRunningStatus
描述：广播业务(service)或节目(event)的运行状态常量定义接口，取值详见 GB/T 28161—2011。

B.7.14.1 常量域——运行状态

B.7.14.1.1 UNDEFINED

原型：public static final byte UNDEFINED = 0
描述：运行状态——未定义。

B.7.14.1.2 NOT_RUNNING

原型：public static final byte NOT_RUNNING = 1
描述：运行状态——未运行。

B.7.14.1.3 STARTS_IN_A_FEW_SECONDS

原型：public static final byte STARTS_IN_A_FEW_SECONDS = 2
描述：运行状态——即将运行。

B.7.14.1.4 PAUSING

原型：public static final byte PAUSING = 3
描述：运行状态——暂停状态。

B.7.14.1.5 RUNNING

原型：public static final byte RUNNING = 4
描述：运行状态——运行。

B.7.15 接口org.ngb.broadcast.dvb.si.SIServiceType

原型：public interface org.ngb.broadcast.dvb.si.SIServiceType
描述：业务类型(service_type)常量定义接口，取值详见GB/T 28161—2011。

B.7.15.1 常量域——业务类型

B.7.15.1.1 SERVICE_TYPE_RESERVED

原型：public static final short SERVICE_TYPE_RESERVED = 0
描述：业务类型——预留使用。

B.7.15.1.2 SERVICE_TYPE_DIGITAL_TELEVISION

原型：public static final short SERVICE_TYPE_DIGITAL_TELEVISION = 1
描述：业务类型——数字电视广播业务。

B.7.15.1.3 SERVICE_TYPE_DIGITAL_RADIO_SOUND

原型：public static final short SERVICE_TYPE_DIGITAL_RADIO_SOUND = 2
描述：业务类型——数字声音广播业务。

B.7.15.1.4 SERVICE_TYPE_TELETEXT

原型：public static final short SERVICE_TYPE_TELETEXT = 3
描述：业务类型——图文电视业务。

B.7.15.1.5 SERVICE_TYPE_NVOD_REFERENCE

原型：public static final short SERVICE_TYPE_NVOD_REFERENCE = 4
描述：业务类型——NVOD参考业务。

B.7.15.1.6 SERVICE_TYPE_NVOD_TIME_SHIFTED

原型：public static final short SERVICE_TYPE_NVOD_TIME_SHIFTED = 5
描述：业务类型——NVOD时移业务。

B.7.15.1.7 SERVICE_TYPE_MOSAIC

原型：public static final short SERVICE_TYPE_MOSAIC = 6
描述：业务类型——马赛克业务。

B.7.15.1.8 SERVICE_TYPE_DATA_BROADCAST

原型：static final short SERVICE_TYPE_DATA_BROADCAST = 12

描述：业务类型——数据广播业务。

B.7.16 接口org.ngb.broadcast.dvb.si.SIStreamType

原型：public interface org.ngb.broadcast.dvb.si.SIStreamType
描述：基本流流类型常量(stream_type)定义接口，流类型常量取值详见 GB/T 17975.1—2010 中的定义。

B.7.16.1 常量域——ES流类型

B.7.16.1.1 ES_PRIVATE_RESERVED

原型：public static final byte ES_PRIVATE_RESERVED = 0
描述：ES 流类型——保留。

B.7.16.1.2 ES_MPEG1_VIDEO

原型：public static final byte ES_MPEG1_VIDEO = 1
描述：ES 流类型——GB/T 17191.2 视频。

B.7.16.1.3 ES_MPEG2_VIDEO

原型：public static final byte ES_MPEG2_VIDEO = 2
描述：ES 流类型——GB/T 17975.2 视频。

B.7.16.1.4 ES_MPEG1_AUDIO

原型：public static final byte ES_MPEG1_AUDIO = 3
描述：ES 流类型——GB/T 17191.2 音频。

B.7.16.1.5 ES_MPEG2_AUDIO

原型：public static final byte ES_MPEG2_AUDIO = 4
描述：ES 流类型——GB/T 17975.2 音频。

B.7.16.1.6 ES_PRIVATE_SECTION

原型：public static final byte ES_PRIVATE_SECTION = 5
描述：ES 流类型——私有段。

B.7.16.1.7 ES_PRIVATE_DATA

原型：public static final byte ES_PRIVATE_DATA = 6
描述：ES 流类型——包含私有数据的 PES 包。

B.7.16.1.8 ES_MHEG

原型：public static final byte ES_MHEG = 7
描述：ES 流类型——ISO/IEC 13522-1 MHEG。

B.7.16.1.9 ES_DSMCC

原型：public static final byte ES_DSMCC = 8
描述：流类型——ISO/IEC 13818-1 附录 B DSMCC。

B.7.16.1.10 ES_DSMCC_A

原型：public static final byte ES_DSMCC_A = 10
描述：ES 流类型——ISO/IEC 13818-6 类型 A。

B.7.16.1.11 ES_DSMCC_B

原型：public static final byte ES_DSMCC_B = 11
描述：ES 流类型——ISO/IEC 13818-6 类型 B。

B.7.16.1.12 ES_DSMCC_C

原型：public static final byte ES_DSMCC_C = 12
描述：ES 流类型——ISO/IEC 13818-6 类型 C。

B.7.16.1.13 ES_DSMCC_D

原型：public static final byte ES_DSMCC_D = 13
描述：ES 流类型——ISO/IEC 13818-6 类型 D。

B.7.16.1.14 ES_AVS_VIDEO

原型：public static final byte ES_AVS_VIDEO = 66
描述：ES 流类型——GB/T 20090.2—2006 视频类型。

B.7.17 类org.ngb.broadcast.dvb.si.SIDatabase

原型：public class org.ngb.broadcast.dvb.si.SIDatabase
描述:PSI/SI 信息数据库,提供了 DVB 模式下 PSI/SI 信息数据库的管理和操作方法,是应用获取 PSI/SI 信息的入口类。每个物理广播网络接口对应一个 SIDatabase 对象，若接收终端只有一个物理广播网络接口，则只有一个 org.ngb.broadcast.dvb.si.SIDatabase 对象。系统实现时，需要考虑下列特殊应用场景：

——场景 1：同一个物理接口连接多个广播网络，例如地面无线 Tuner 可能接入到不同运营商的无线网络，此时该物理接口只对应一个 org.ngb.broadcast.dvb.si.SIDatabase 对象；

——场景 2：多个物理接口连接同一个广播网络，例如具备 PVR 功能的接收机有两个 Tuner 同时连接至有线网络，此时系统应实现多个 org.ngb.broadcast.dvb.si.+SIDatabase 对象，尽管每个 SIDatabase 所维护的信息都相同。

B.7.17.1 方法

B.7.17.1.1 getDatabase

原型：public static org.ngb.broadcast.dvb.si.SIDatabase[] getDatabase()
描述：获取 SI 数据库实例。接收终端可能有多个广播网络接口（例如同时连接有线网络和国标地面无线网络），每个网络接口对应一个 SI 数据库实例，若有多个网络接口，本方法则返回多个 SI 数据库实例。
返回：org.ngb.broadcast.dvb.si.SIDatabase 对象数组，表示 SI 数据库实例。
注：系统实现时，需要考虑下列特殊场景：

——场景 1：同一个物理网络接口连接多个广播网络，例如地面无线 Tuner 可能会接入到不同运营商的无线网络，在该场景下，只对应一个 org.ngb.broadcast.dvb.si.SIDatabase 实例；

——场景 2：多个物理网络接口连接同一个广播网络，例如具备 PVR 功能的接收机有两个以上 Tuner，同时接入到同一个网络，在此场景下，对应多个 org.ngb.broadcast.dvb.si.SIDatabase 实例。

B.7.17.1.2 getID

原型：public int getID()

描述：获取 org.ngb.broadcast.dvb.si.SIDatabase 对象的全局唯一标识符。

参数：无。

返回：int 型，表示 org.ngb.broadcast.dvb.si.SIDatabase 对象的全局唯一标识符。

B.7.17.1.3 addNITUpdateListener

原型：public boolean addNITUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId)
throws java.lang.IllegalArgumentException

描述：注册 NIT 更新事件监听器，当 NIT 发生更新时，将向注册的监听器对象发送一个 SIUpdateEvent 事件。全程监控，当与该 org.ngb.broadcast.dvb.si.SIDatabase 对象关联的网络接口调谐到另一个传送流时，监控过程不停止。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注册的接收 NIT 更新事件的监听器对象；
networkId - int 型，表示 NIT 的唯一标识。

返回：boolean 型，表示注册结果，取值 true 表示注册成功，false 表示注册失败。

异常：java.lang.IllegalArgumentException - 若由参数 network_id 指定的网络不存在，则抛出此异常。

B.7.17.1.4 removeNITUpdateListener

原型：public boolean removeNITUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int network_id)
throws java.lang.IllegalArgumentException

描述：注销 NIT 更新事件监听器。若与参数 network_id 关联的监听器不存在，则此方法失败，但不抛出异常。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注销的接收 NIT 更新事件的监听器对象；
network_id - int 型，表示 NIT 的唯一标识。

返回：boolean 型，表示注销结果，取值 true 表示注销成功，false 表示注销失败。

异常：java.lang.IllegalArgumentException - 若由参数 network_id 指定的网络不存在，则抛出此异常。

B.7.17.1.5 addBATUpdateListener

原型：public boolean addBATUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int bouquetId)
throws java.lang.IllegalArgumentException

描述：注册 BAT 更新事件监听器，当 BAT 发生更新时，将向注册的监听器对象发送一个 org.ngb.broadcast.dvb.si.SIUpdateEvent 事件。全程监控，当与该 org.ngb.broadcast.dvb.si.SIDatabase 对象关联的网络接口调谐到另一个传送流时，监控过

程不停止。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注册的接收 BAT 更新事件的监听器对象；
networkId - int 型，表示 BAT 所在网络的网络标识；
bouquetId - int 型，表示 BAT 的唯一标识。

返回：boolean 型，表示注册结果，取值 true 表示注册成功，false 表示注册失败。

异常：java.lang.IllegalArgumentException - 若由参数 network_id 和 bouquet_id 指定的业务群不存在，则抛出此异常。

B.7.17.1.6 removeBATUpdateListener

原型：public boolean removeBATUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int bouquetId)
throws java.lang.IllegalArgumentException

描述：注销 BAT 更新事件监听器。若与参数 bouquet_id 关联的监听器不存在，则此方法失败，但不抛出异常。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注销的接收 BAT 更新事件的监听器对象；
networkId - int 型，表示 BAT 所在网络的网络标识；
bouquetId - int 型，表示 BAT 的唯一标识。

返回：boolean 型，表示注销结果，取值 true 表示注销成功，false 表示注销失败。

异常：java.lang.IllegalArgumentException - 若由参数 networkId 和 bouquetId 指定的业务群不存在，则抛出此异常。

B.7.17.1.7 addPATUpdateListener

原型：public void addPATUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener)

描述：注册 PAT 更新事件监听器，当 PAT 发生更新时，将向注册的监听器对象发送一个 SIUpdateEvent 事件。仅对当前传送流中的 PAT 进行监控，当与该 org.ngb.broadcast.dvb.si.SIDatabase 对象关联的网络接口调谐到另一个传送流时，监控过程应在后台停止。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注册的接收 PAT 更新事件的监听器对象。

返回：无。

B.7.17.1.8 removePATUpdateListener

原型：public void removePATUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener)

描述：注销 PAT 更新事件监听器。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注销的接收 PAT 更新事件的监听器对象。

返回：无。

B.7.17.1.9 addPMTUpdateListener

原型：public boolean addPMTUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int originalNetworkId, int transportStreamId, int serviceId)

throws java.lang.IllegalArgumentException

描述：注册 PMT 更新事件监听器，若由 networkId、originalNetworkId、transportStreamId 和 serviceId 指定的业务在当前传送流中承载，当 PMT 发生更新时，将向注册的监听器对象发送一个 org.ngb.broadcast.dvb.si.SIUpdateEvent 事件。仅对当前传送流中的 PMT 进行监控，当与该 org.ngb.broadcast.dvb.si.SIDatabase 对象关联的网络接口调谐到另一个传送流时，监控过程应在后台停止。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注册的接收 PMT 更新事件的监听器对象；
networkId - int 型，表示网络标识；
originalNetworkId - int 型，表示原始网络标识；
transportStreamId - int 型，表示传送流标识；
serviceId - int 型，表示业务标识。

返回：boolean 型，表示注册结果，取值 true 表示注册成功，false 表示注册失败。

异常：java.lang.IllegalArgumentException - 若由参数 networkId、originalNetworkId、transportStreamId 和 serviceId 指定的业务不存在，则抛出此异常。

B.7.17.1.10 removePMTUpdateListener

原型：public boolean removePMTUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int originalNetworkId, int transportStreamId,int serviceId) throws java.lang.IllegalArgumentException

描述：注销 PMT 更新事件监听器。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注销的接收 PMT 更新事件的监听器对象；
networkId - int 型，表示网络标识；
originalNetworkId - int 型，表示原始网络标识；
transportStreamId - int 型，表示传送流标识；
serviceId - int 型，表示业务标识。

返回：boolean 型，表示注销结果，取值 true 表示注销成功，false 表示注销失败。

异常：java.lang.IllegalArgumentException - 若由参数 networkId、originalNetworkId、transportStreamId 和 serviceId 指定的业务不存在，则抛出此异常。

B.7.17.1.11 addSDTUpdateListener

原型：public boolean addSDTUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int originalNetworkId, int transportStreamId) throws java.lang.IllegalArgumentException

描述：注册 SDT 更新事件监听器，由参数 network_id、originalNetworkId 和 transportStreamId 指定的 SDT 发生更新时，将向注册的监听器对象发送一个 SIUpdateEvent 事件。仅对当前传送流中的 SDT 进行监控，当与该 SIDatabase 对象关联的网络接口开始调谐到另一个传送流时，监控过程应该在后台停止。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注册的接收 SDT 更新事件的监听器对象；
networkId - int 型，表示网络标识；
originalNetworkId - int 型，表示原始网络标识；

transportStreamId - int 型，表示传送流标识。

返回：boolean 型，表示注册结果，取值 true 表示注册成功，false 表示注册失败。

异常：java.lang.IllegalArgumentException - 若由参数 networkId、originalNetworkId 和 transportStreamId 指定的 SDT 不存在，则抛出此异常。

B.7.17.1.12 removeSDTUpdateListener

原型：public boolean removeSDTUpdateListener(org.ngb.broadcast.dvb.si.SIUpdateListener listener, int networkId, int originalNetworkId, int transportStreamId) throws java.lang.IllegalArgumentException

描述：注销 SDT 更新事件监听器。

参数：listener - org.ngb.broadcast.dvb.si.SIUpdateListener 对象，表示待注销的接收 SDT 更新事件的监听器对象；

networkId - int 型，表示网络标识；

originalNetworkId - int 型，表示原始网络标识；

transportStreamId - int 型，表示传送流标识。

返回：boolean 型，表示注销结果，取值 true 表示注销成功，false 表示注销失败。

异常：java.lang.IllegalArgumentException - 若由参数 networkId、originalNetworkId 和 transportStreamId 指定的 SDT 不存在，则抛出此异常。

B.7.17.1.13 getAllNetworks

原型：public org.ngb.broadcast.dvb.si.SINetwork[] getAllNetworks()

描述：获取现行网络信息。

返回：org.ngb.broadcast.dvb.si.SINetwork 数组

B.7.17.1.14 getAllServices

原型：public org.ngb.broadcast.dvb.si.SIService[] getAllServices()

描述：获取业务信息。

返回：org.ngb.broadcast.dvb.si.SIService 数组

B.7.17.1.15 getAllTransportStreams

原型：public org.ngb.broadcast.dvb.si.SITransportStream[] getAllTransportStreams()

描述：获取所有传输流信息。

返回：org.ngb.broadcast.dvb.si.SITransportStream 数组

B.7.17.1.16 getSIBouquets

原型：public org.ngb.broadcast.dvb.si.SIBouquet[] getSIBouquets(int network_id, int original_network_id, int transport_stream_id, int service_id)

描述：获取某个频点、节目所在的 org.ngb.broadcast.dvb.si.SIBouquet，若参数都为-1 则返回所有的 org.ngb.broadcast.dvb.si.SIBouquet。

参数：network_id- int 型，表示网络标识，若为-1，表示检索现行和其他网络 NIT 表描述的网络信息；

original_network_id- int 型，原始网络标识；

transport_stream_id- int 型，传输流标识；

service_id- 节目标识，即频道节目号，若为-1 表示不匹配节目标识；

返回：org.ngb.broadcast.dvb.si.SIBouquet 对象数组。

B.7.17.1.17 getSIElementStreams

原型：public org.ngb.broadcast.dvb.si.SIElementaryStream[] getSIElementStreams(int network_id, int original_network_id, int transport_stream_id, int service_id)

描述:获取某个节目中的SIElementaryStream,若参数都为-1则返回所有节目的SIElementaryStream。

参数：network_id- int 型，表示网络标识，若为-1，表示检索现行和其他网络 NIT 表描述的网络信息；

original_network_id- int 型，原始网络标识；

transport_stream_id- int 型，传输流标识；

service_id- 节目标识，即频道节目号，若为-1 表示不匹配节目标识；

返回：org.ngb.broadcast.dvb.si.SIElementaryStream 对象数组。

B.7.17.1.18 getSIServices

原型：public org.ngb.broadcast.dvb.si.SIService[] getSIServices(int network_id,int original_network_id,int transport_stream_id,int service_id)

描述：获取指定的业务信息 org.ngb.broadcast.dvb.si.SIService。

参数：network_id- int 型，表示网络标识，若为-1，表示检索现行和其他网络 NIT 表描述的网络信息；

original_network_id- int 型，原始网络标识；

transport_stream_id- int 型，传输流标识；

service_id- 节目标识，即频道节目号，若为-1 表示不匹配节目标识；

返回：org.ngb.broadcast.dvb.si.SIService 对象数组

B.7.17.1.19 getPreferredLanguage

原型：public java.lang.String getPreferredLanguage()

描述：获取应用设置的查询 SI 文本信息的默认语种类型。

参数：无。

返回： java.lang.String 对象，表示应用设置的查询 SI 文本信息的默认语种类型，语种三字母代码遵循 GB/T 4880.2—2000。

B.7.17.1.20 setPreferredLanguage

原型：public void setPreferredLanguage(java.lang.String iso639code)

描述：设置应用查询 SI 文本信息的默认语种类型。

参数：iso639code - java.lang.String 对象，表示 SI 文本信息语种类型，语种三字母代码遵循 GB/T 4880.2—2000。

返回：无。

B.7.18 类org.ngb.broadcast.dvb.si.SIRequestFailureType

原型：public class org.ngb.broadcast.dvb.si.SIRequestFailureType

描述：PSI/SI 信息检索失败原因。

B.7.18.1 常量域——失败原因

B.7.18.1.1 UNKNOWN

原型：public static final int UNKNOWN = 0
描述：检索失败原因——未知。

B.7.18.1.2 CANCELED

原型：public static final int CANCELED = 1
描述：检索失败原因——请求被应用取消。

B.7.18.1.3 DATA_UNAVAILABLE

原型：public static final int DATA_UNAVAILABLE = 2
描述：检索失败原因——数据不可用。

B.7.18.1.4 INSUFFICIENT_RESOURCES

原型：public static final int INSUFFICIENT_RESOURCES = 3
描述：检索失败原因——资源不足。

B.7.18.2 方法

B.7.18.2.1 getCode

原型：public int getCode()
描述：获取信息检索失败原因代码。
参数：无。
返回：int 型，表示信息检索失败原因代码，
取值详见 org.ngb.broadcast.dvb.si.SIRequestFailureType 接口的“失败原因”常量域定义。

B.7.18.2.2 toString

原型：public java.lang.String toString()
描述：获取信息检索失败原因的文本描述。
重写： java.lang.Object 类的 toString()方法。
参数：无。
返回： java.lang.String 对象，表示信息检索失败原因的文本描述。

B.7.19 事件org.ngb.broadcast.dvb.si.SIRetrieveEvent

原型：public class org.ngb.broadcast.dvb.si.SIRetrieveEvent extends java.util.EventObject
描述：PSI/SI 信息请求事件，是本包定义的一组与 PSI/SI 信息请求相关事件的基类。一次 PSI/SI 信息请求只会产生一个这样的事件。

B.7.19.1 方法

B.7.19.1.1 getSource

原型：public java.lang.Object getSource()
描述：获取产生该事件的 SIRequest 对象。
重写： java.util.EventObject 类的 getSource()方法。

参数：无。

返回： org.ngb.broadcast.dvb.si.SIRequest 对象，表示产生该事件的 SIRequest 对象。

B.7.19.1.2 getAppData

原型：public java.lang.Object getAppData()

描述：获取附加应用数据信息。

参数：无。

返回： java.lang.Object 对象，表示附加应用数据。

B.7.20 事件org.ngb.broadcast.dvb.si.SISuccessRetrieveEvent

原型：public class org.ngb.broadcast.dvb.si.SISuccessRetrieveEvent
extends org.ngb.broadcast.dvb.si.SIRetrieveEvent

描述：PSI/SI 信息或描述符请求成功事件。

B.7.20.1 方法

B.7.20.1.1 getResult

原型：public java.util.Enumeration getResult()

描述：获取成功检索结果。

参数：无。

返回： java.util.Enumeration 对象，表示检索结果。由于通过异步方法检索 PSI/SI 信息和描述符信息均通过该事件对象返回，因此该枚举对象元素可能为
org.ngb.broadcast.dvb.si.SICommonInformation 类型或
org.ngb.broadcast.dvb.si.SIDescriptor 类型，应用应进一步通过 instanceof 方法判断枚举对象元素的类型。

B.7.21 事件org.ngb.broadcast.dvb.si.SIFailureRetrieveEvent

原型：public class org.ngb.broadcast.dvb.si.SIFailureRetrieveEvent
extends org.ngb.broadcast.dvb.si.SIRetrieveEvent

描述：PSI/SI 信息请求失败事件。

B.7.21.1 方法

B.7.21.1.1 getReason

原型：public org.ngb.broadcast.dvb.si.SIRequestFailureType getReason()

描述：获取 PSI/SI 信息检索失败的原因。

参数：无。

返回：org.ngb.broadcast.dvb.si.SIRequestFailureType 对象，表示 PSI/SI 信息检索失败的原因。

B.7.22 事件org.ngb.broadcast.dvb.si.SIUpdateEvent

原型：public class org.ngb.broadcast.dvb.si.SIUpdateEvent extends java.util.EventObject

描述：PSI/SI 表更新事件。应用程序首先应调用 getTableID()方法，以判断是哪种 PSI/SI 表发生更新，然后再根据具体的表的类型获取其他一些参数。

B.7.22.1 方法

B.7.22.1.1 getTableID

原型：public int getTableID()
描述：获取发生更新的PSI/SI表标识(table_id)。
返回：int型，表示发生更新的PSI/SI表标识。

B.7.22.1.2 getBouquetID

原型：public int getBouquetID()
描述：获取业务群标识(bouquet_id)。
参数：无。
返回：int型，表示业务群标识。当BAT发生更新时返回值才有意义。

B.7.22.1.3 getNetworkID

原型：public int getNetworkID()
描述：获取网络的标识。
参数：无。
返回：int型，表示网络标识。当NIT发生更新时返回值才有意义。

B.7.22.1.4 getOriginalNetworkID

原型：public int getOriginalNetworkID()
描述：获取原始网络标识。
参数：无。
返回：int型，表示原始网络标识。当PMT/SDT/EIT发生更新时返回值才有意义.

B.7.22.1.5 getServiceID

原型：public int getServiceID()
描述：获取业务标识符。
返回：int型，表示业务标识符（service_id）。当PMT/EIT发生更新时返回值才有意义。

B.7.22.1.6 getTransportStreamID

原型：public int getTransportStreamID()
描述：获取传送流标识。
参数：无。
返回：int型，表示传送流标识。当PAT/PMT/SDT/EIT发生更新时返回值才有意义。

B.7.23 异常org.ngb.broadcast.dvb.si.InvalidPeriodException

原型：public class org.ngb.broadcast.dvb.si.InvalidPeriodException
extends java.lang.Exception
描述：无效周期异常。当指定的日期无效时，抛出该异常。

附 录 C
（规范性附录）
JAVA-双向宽带网络接入单元

C.1 概述

本附录定义了与双向宽带网络接入相关的JAVA接口，包括以太网管理模块和WiFi管理模块。

C.2 以太网管理模块

以太网管理模块提供 DHCP 信息以及对以太网的管理信息。
以太网管理模块概要见表 C.1。

表 C.1 以太网管理模块概要

接口	
Listener	以太网状态改变事件监听器，由应用层实现
类	
DhcpInfo	提供了获取 DHCP 配置信息的方法。
EthernetManager	提供以太网管理的方法。

C.2.1 接口org.tvos.net.Listener

原型：public interface org.tvos.net.Listener
描述：以太网状态改变事件监听器，由应用层实现。

C.2.1.1 方法

C.2.1.1.1 onAvailabilityChanged

原型：public void onAvailabilityChanged(boolean isAvailable)
描述：以太网是否可用状态通知接口。
参数：isAvailable – boolean型，true代表可用，false代表不可用。
返回：无。

C.2.2 类org.tvos.net.DhcpInfo

原型：public class org.tvos.net.DhcpInfo implements org.tvos.os.Parcelable
描述：DHCP配置信息类，提供了获取DHCP配置信息的方法。

C.2.2.1 属性

C.2.2.1.1 ipAddress

原型：public int ipAddress
描述：表示 IP 地址信息。

C.2.2.1.2 gateway

原型：public int gateway
描述：表示网关信息。

C.2.2.1.3 dns1

原型：public int dns1
描述：表示 dns1 信息。

C.2.2.1.4 dns2

原型：public int dns2
描述：表示 dns2 信息。

C.2.2.1.5 serverAddress

原型：public int serverAddress
描述：表示 serverAddress 信息。

C.2.3 类org.tvos.net.EthernetManager

原型：public org.tvos.net.EthernetManager(org.tvos.content.Context context,
org.tvos.net.IEthernetManager service)
描述：以太网连接管理类，提供了对以太网网连接的方法。

C.2.3.1 常量域

C.2.3.1.1 ETHERNET_STATE_DISABLED

原型：public static final int ETHERNET_STATE_DISABLED = 0
描述：以太网去使能常量定义。

C.2.3.1.2 ETHERNET_STATE_ ENABLED

原型：public static final int ETHERNET_STATE_ENABLED = 1
描述：以太网使能常量定义。

C.2.3.1.3 ETHERNET_STATE_ UNKNOWN

原型：public static final int ETHERNET_STATE_UNKNOWN = 2
描述：以太网 UNKNOWND 常量定义,通常表示在 initialed 阶段。

C.2.3.1.4 EVENT_DHCP_CONNECT_SUCCESSED

原型：public static final int EVENT_DHCP_CONNECT_SUCCESSED = 10
描述：DHCP 连接成功。

C.2.3.1.5 EVENT_DHCP_CONNECT_FAILED

原型：public static final int EVENT_DHCP_CONNECT_FAILED = 11
描述：DHCP 连接失败。

C.2.3.2 方法

C.2.3.2.1 isAvailable

原型：public boolean isAvailable()
描述：当前以太网接口是否可用。
参数：无 。
返回：boolean类型，true表示可用，false表示不可用。

C.2.3.2.2 addListener

原型：public void addListener(org.tvos.net.Listener listener)
描述：添加以太网状态改变事件侦听接口。
参数：listener –org.tvos.net.Listener型，侦听回调函数，由应用实现。
返回：无。

C.2.3.2.3 removeListener

原型：public void removeListener(org.tvos.net.Listener listener)
描述：删除以太网状态改变事件侦听接口。
参数：listener –org.tvos.net.Listener型，侦听回调函数，由应用实现。
返回：无。

C.2.3.2.4 getConfiguration

原型：public org.tvos.net.IpConfiguration getConfiguration()
描述：获取网络配置信息。
参数：无。
返回：org.tvos.net.IpConfiguration对象，表示网络配置信息。

C.2.3.2.5 setConfiguration

原型：public void setConfiguration(org.tvos.net.IpConfiguration config)
描述：设置网络配置信息。
参数：config –org.tvos.net.IpConfiguration对象，表示网络的配置信息。
返回：无。

C.2.3.2.6 setEthernetEnabled

原型：public void setEthernetEnabled(boolean enable)
描述：设置以太网使能。
参数：enable – boolean型，true表示使能；false表示去使能。
返回：无。

C.2.3.2.7 getEthernetState

原型：public int getEthernetState()
描述：获取以太网状态。
参数：无。
返回：int型，表示以太网状态。

——ETHERNET_STATE_DISABLED，以太网去使能状态；
——ETHERNET_STATE_ENABLED，以太网使能状态；
——ETHERNET_STATE_UNKNOWN，未知状态。

C.2.3.2.8 getDhcpInfo

原型：public org.tvos.net.DhcpInfo getDhcpInfo()
描述：获取以太网DHCP信息。
参数：无。
返回：org.tvos.net.DhcpInfo对象，表示以太网的DHCP信息。

C.2.3.2.9 getNetLinkStatus

原型：public boolean getNetLinkStatus()
描述：获取网络的物理链路状态。
参数：无。
返回： boolean类型，true表示已连接，false表示未连接。

C.2.3.2.10 getNetLinkStatus

原型：public int getNetLinkStatus()
描述：制定网口获取网络的物理链路状态。
参数：无。
返回：int型，1表示已连接；0表示未连接；-1表示未找到制定网口。

C.2.3.2.11 getInterfaceName

原型：public java.lang.String getInterfaceName()
描述：获取网络名称。
参数：无。
返回：java.lang.String型，表示网络名称。

C.2.3.2.12 setEthernetMode

原型：public void setEthernetMode(java.lang.String mode, org.tvos.net.DhcpInfo dhcpInfo)
描述：设置网络模式。
参数：mode -java.lang.String型，可有ETHERNET_CONNECT_MODE_DHCP、
ETHERNET_CONNECT_MODE_MANUAL、ETHERNET_CONNECT_MODE_PPPOE、
ETHERNET_CONNECT_MODE_NONE四种取值；
dhcpInfo -org.tvos.net.DhcpInfo对象，表示DHCP信息。
返回：无。

C.2.3.2.13 getEthernetMode

原型：public java.lang.String getEthernetMode()
描述：获取网络模式。
参数：无。

返回：java.lang.Sting型，表示网络模式，可有ETHERNET_CONNECT_MODE_DHCP、ETHERNET_CONNECT_MODE_MANUAL、ETHERNET_CONNECT_MODE_PPPOE、ETHERNET_CONNECT_MODE_NONE四种取值。

C.2.3.2.14 getDeviceNameList

原型：public java.lang.String[] getDeviceNameList()
描述：获取可用的物理网口名称。
参数：无。
返回：java.lang.String数组，表示网口名称列表。

C.2.3.2.15 getTotalInterface

原型：public int getTotalInterface()
描述：获取可用的物理网口数量。
参数：无。
返回：int型，表示网口数量 。

C.2.3.2.16 enableEthernet

原型：public void enableEthernet(boolean enable)
描述：使能网络。
参数：enable – boolean型，true代表使能；false代表去使能。
返回：无。

C.2.3.2.17 setInterfaceName

原型：public boolean setInterfaceName(java.lang.String iface)
描述：设置网口名称。
参数：iface –java.lang.String型，取值eth0，eth1等。
返回：boolean型，true代表成功；false代表失败。

C.2.3.2.18 getDhcpOption60State

原型：public int getDhcpOption60State()
描述：获取Dhcp Option60/Option61状态。
参数：无。
返回：int型，可取如下三种取值OPTION60_STATE_DISABLED，OPTION60_STATE_ENABLED，OPTION60_STATE_UNKNOWN。

C.2.3.2.19 getDhcpOption60Login

原型：public java.lang.String getDhcpOption60Login()
描述：获取Dhcp Option61用户名。
参数：无。
返回：java.lang.Sting型，表示Option61用户名。

C.2.3.2.20 getDhcpOption60Password

原型：public java.lang.String getDhcpOption60Password()

描述：获取Dhcp Option60密码。
参数：无。
返回：java.lang.String型，表示密码。

C.2.3.2.21 setDhcpOption60

原型：public void setDhcpOption60(boolean enable, java.lang.String login, java.lang.String password)
描述：设置Dhcp Option60的用户名和密码。
参数：enable – boolean型，表示使能或去使能；
login –java.lang.String型，表示用户名；
password – java.lang.String，表示密码。
返回：无。

C.2.3.2.22 getDhcpOption125State

原型：public int getDhcpOption125State()
描述：获取DHCP Option125状态。
参数：无。
返回：int型，取值OPTION125_STATE_DISABLED或OPTION125_STATE_ENABLED或OPTION125_STATE_UNKNOWN。

C.2.3.2.23 getDhcpOption125Info

原型：public java.lang.String getDhcpOption125Info()
描述：获取DHCP Option125信息。
参数：无。
返回：java.lang.String型，表示DHCP Option125信息。

C.2.3.2.24 getDhcpOption125Info

原型：public void setDhcpOption125(boolean enable, java.lang.String option125Info)
描述：设置DHCP Option125信息
参数：enable – boolean型，表示使能或去使能。
option125Info –java.lang.String型，表示Option125信息。
返回：无。

C.2.3.2.25 enableIpv6

原型：public void enableIpv6(boolean enable)
描述：使能IPv6。
参数：enable – boolean型，表示使能或去使能
返回：无。

C.2.3.2.26 getIpv6PersistedState

原型：public int getIpv6PersistedState()
描述：获取DHCPv6状态。
参数：无。

返回：int型，取值DHCPV6_STATE_ENABLED，DHCPV6_STATE_DISABLED，DHCPV6_STATE_UNKNOWN。

C.2.3.2.27 setEthernetMode6

原型：public void setEthernetMode6(java.lang.String mode)
描述：设置获取IPv6地址的模式。
参数：mode –java.lang.String型，取值ETHERNET_CONNECT_MODE_DHCP或
ETHERNET_CONNECT_MODE_MANUAL。
返回：无。

C.3 WiFi管理模块

WiFi 管理模块提供了与 WiFi 网络接口控制相关的类和方法。
WiFi 管理模块概要见表 C.2。

表 C.2 WiFi 管理模块概要

接口	
ActionListener	WiFi 状态改变事件监听器，由应用层实现。
类	
WifiInfo	WiFi 连接信息类，提供了获取 WiFi 连接信息的方法。
WifiManager	WiFi 管理器，提供 WiFi 无线网络的管理功能： 1) 发现、扫描和管理当前可用的无线网络访问点(AP)； 2) 管理当前活动的网络访问链接，如建立连接、断开连接、禁用连接、删除连接等。
ScanResult	WiFi 扫描结果，描述了 WiFi 扫描发现的一个连接访问点信息。

C.3.1 接口org.tvos.net.wifi.ActionListener

原型：public interface org.tvos.net.wifi.ActionListener
描述：WiFi状态改变事件监听器接口，由应用层实现。

C.3.1.1 方法

C.3.1.1.1 onSuccess

原型：public void onSuccess()
描述：WiFi连接成功。
参数：无 。
返回：无。

C.3.1.1.2 onFailure

原型：public void onFailure(int reason)
描述：WiFi连接失败。
参数：reason – int型，表示连接失败的原因，取值详见WifiManager的“WiFi状态”常量域定义。
返回：无。

C.3.2 类org.tvos.net.wifi.WifiInfo

原型：public class org.tvos.net.wifi.WifiInfo implements org.tvos.os.Parcelable
描述：WiFi连接信息类，描述WiFi连接状态。

C.3.2.1 方法

C.3.2.1.1 getMacAddress

原型：public java.lang.String getMacAddress()
描述：获取热点路由MAC地址。
参数：无。
返回：java.lang.String对象，表示热点路由MAC地址。

C.3.2.1.2 getSSID

原型：public java.lang.String getSSID()
描述：获取当前网络连接的SSID。
参数：无。
返回：java.lang.String对象，表示当前网络连接的SSID。

C.3.2.1.3 isHiddenSSID

原型：public boolean isHiddenSSID()
描述：判断当前连接是否隐藏SSID（即不广播SSID）。
参数：无。
返回：boolean型，表示连接是否隐藏SSID，取值true表示隐藏SSID，取值false表示广播SSID。

C.3.2.1.4 getLinkSpeed

原型：public int getLinkSpeed()
描述：获取网络连接的速度。
参数：无。
返回：int型，表示网络连接速度，单位为兆比特每秒（Mb/s）。

C.3.2.1.5 getNetworkId

原型：public int getNetworkId()
描述：获取当前网络连接的标识号。
参数：无。
返回：int型，表示当前网络连接的网络标识。

C.3.2.1.6 getBSSID

原型：public java.lang.String getBSSID()
描述：获取访问点BSSID。
参数：无。
返回：java.lang.String型，返回访问点BSSID。

C.3.2.1.7 getIpAddress

原型：public int getIpAddress()

描述：获取当前连接的ip地址。
参数：无。
返回：int型，表示当前连接的IP地址

C.3.2.1.8 getFrequency

原型：public int getFrequency()
描述：获取当前连接的发射频率。
参数：无。
返回：int型，表示当前连接的发射频率，单位MHz

C.3.3 类org.tvos.net.wifi.WifiManager

原型：public class org.tvos.net.wifi.WifiManager
描述：WiFi管理器，提供WiFi无线网络的管理功能：
1）发现、扫描和管理当前可用的无线网络访问点(AP)；
2）管理当前活动的网络访问链接，如建立连接，断开连接，禁用连接，删除连接。

C.3.3.1 常量域——WiFi状态

C.3.3.1.1 ERROR_AUTHENTICATING

原型：public static final int ERROR_AUTHENTICATING = 1
描述：当认证失败时返回的错误码。

C.3.3.1.2 WIFI_STATE_DISABLING

原型：public static final int WIFI_STATE_DISABLING = 0
描述：wifi去使能时的中间状态。

C.3.3.1.3 WIFI_STATE_DISABLED

原型：public static final int WIFI_STATE_DISABLED = 1
描述：wifi去使能状态。

C.3.3.1.4 WIFI_STATE_ENABLING

原型：public static final int WIFI_STATE_ENABLING = 2
描述：wifi使能的中间状态。

C.3.3.1.5 WIFI_STATE_ENABLED

原型：public static final int WIFI_STATE_ENABLED = 3
描述：wifi使能状态。

C.3.3.1.6 WIFI_STATE_UNKNOWN

原型：public static final int WIFI_STATE_UNKNOWN = 4
描述：wifi unknown状态，返回此状态时，表示通常有错误发生。

C.3.3.2 方法

C.3.3.2.1 WifiManager

原型：public WifiManager(org.tvos.content.Context context, org.tvos.net.wifi.IWifiManager service)

描述：获取系统实现的WiFi无线网络管理器实例。

参数：context －org.tvos.content.Context对象，表示应用的上下文

service － org.tvos.net.wifi.Iwifimanager对象，表示WiFi服务。

返回：无。

C.3.3.2.2 getConfiguredNetworks

原型：public List<org.tvos.net.wifi.WifiConfiguration> getConfiguredNetworks()

描述：获取当前可用的WiFi网络连接配置。

参数：无。

返回：org.tvos.net.wifi.WifiConfiguration对象数组，表示当前可用的WiFi网络连接配置。若无可用的网络连接配置，则返回的数组长度为0。

C.3.3.2.3 addNetwork

原型：public int addNetwork(org.tvos.net.wifi.WifiConfiguration config)

描述：添加一个网络配置。应用可以通过getConfiguredNetworks方法得到可用的网络配置，并选择一个配置设置连接密码，重新加入到WiFi管理器中。

参数：wifiConfiguration －org.tvos.net.wifi.WifiConfiguration对象，表示网络配置对象。

返回：int型，若成功则返回网络配置标识（ID），若失败则返回-1。

C.3.3.2.4 removeNetwork

原型：public boolean removeNetwork(int netId)

描述：移除某个指定的网络配置。

参数：networkId － int型，表示网络配置标识。

返回：无。

C.3.3.2.5 enableNetwork

原型：public boolean enableNetwork(int netId, boolean disableOthers)

描述：启用某个网络连接。将WiFi关联到某个指定的网络连接配置上，同时启动网络连接。调用该接口产生网络连接状态变化事件。

参数：networkId – int型，表示某个网络连接的标识，从WifiConfiguration对象中获得；

disableOthers – boolean型，表示是否禁用其他网络。取值true表示禁用其他网络，false表示不影响其他网络使用。

返回：无。

C.3.3.2.6 disableNetwork

原型：public boolean disableNetwork(int netId)

描述：禁用某个网络连接，该网络连接配置不再作为有效的候选连接。调用该接口产生网络连接状态变化事件。

参数：networkId － int型，表示某个网络连接的标识，从WifiConfiguration对象中获得。

返回：无。

C.3.3.2.7 getConnectionInfo

原型：public org.tvos.net.wifi.WifiInfo getConnectionInfo()
描述：获取当前连接信息。
参数：无。
返回：org.tvos.net.wifi.WifiInfo对象，表示当前连接状态信息。

C.3.3.2.8 isWifiEnabled

原型：public boolean isWifiEnabled()
描述：查询WiFi是否开启。
参数：无。
返回：boolean型，取值true表示WiFi处于开启状态，false表示WiFi处于关闭状态。

C.3.3.2.9 setWifiEnabled

原型：public boolean setWifiEnabled(boolean enabled)
描述：设置WiFi开启或关闭。
参数：enabled - boolean型，取值true表示开启WiFi，取值false表示关闭WiFi。
返回：boolean型，表示操作结果，取值true表示操作成功，false表示操作失败。

C.3.3.2.10 saveConfiguration

原型：public boolean saveConfiguration()
描述：保存网络配置。本接口通知底层保存当前配置好的网络热点信息，以便下次开机时自动连接。
参数：无。
返回：无。

C.3.3.2.11 getScanResult

原型：public List< org.tvos.net.wifi.ScanResult> getScanResults()
描述：获取上一次扫描结果。
参数：无。
返回：org.tvos.net.wifi.ScanResult对象数组，表示上一次扫描结果。若无扫描结果，则返回的数组长度为0。

C.3.3.2.12 startScan

原型：public boolean startScan()
描述：异步方法，启动扫描WiFi无线网络连接访问点。方法立即返回，扫描结果通过事件形式返回给调用者。
参数：无。
返回：boolean型，取值true表示扫描启动成功；false表示扫描启动失败。

C.3.3.2.13 disconnect

原型：public boolean disconnect()
描述：从当前网络连接点断开。调用该接口产生网络连接状态变化事件。
参数：无。
返回：无。

C.3.3.2.14 reassociate

原型：public boolean reassociate()
描述：重新关联当前的网络连接点，即使已经连接到当前网络。调用该接口产生网络连接状态变化事件。
参数：无。
返回：无。

C.3.3.2.15 reconnect

原型：public boolean reconnect()
描述：如果当前网络连接已经断开，重新连接当前网络。调用该接口产生网络连接状态变化事件。
参数：无。
返回：无。

C.3.3.2.16 getDhcpInfo

原型：public org.tvos.net.DhcpInfo getDhcpInfo()
描述：从获取上一次DHCP请求的结果信息。
参数：无。
返回：org.tvos.net.DhcpInfo对象。

C.3.3.2.17 getWifiState

原型：public int getWifiState()
描述：获取wifi的状态。
参数：无。
返回：见常量域的定义，可取以下数值：
WIFI_STATE_DISABLING、WIFI_STATE_ENABLED、WIFI_STATE_ENABLING、WIFI_STATE_UNKNOWN。

C.3.4 类org.tvos.net.wifi.ScanResult

原型：public class org.tvos.net.wifi.ScanResult implements org.tvos.os.Parcelable
描述：描述了WiFi扫描结果中一个连接访问点信息。

C.3.4.1 属性

C.3.4.1.1 SSID

原型：public java.lang.String SSID
描述：WiFi无线网络连接访问点名称。

C.3.4.1.2 BSSID

原型：public java.lang.String BSSID
描述：WiFi无线网络连接访问点地址。

C.3.4.1.3 capabilities

原型：public java.lang.String capabilities
描述：WiFi无线网络连接访问点的能力配置。

C.3.4.1.4 frequency

原型：public int frequency
描述：WiFi无线网络连接访问点的发射频率,单位为兆赫（MHz）。

C.3.4.1.5 level

原型：public int level
描述：WiFi无线网络连接访问点的信号强度, 单位为dBm。

附 录 D
（规范性附录）
JAVA-人机交互单元

D.1 概述

本附录定义了与人机交互相关的JAVA接口定义。

D.2 人机交互模块

人机交互模块提供了人机交互相关的类和方法，包括用户输入和输出。
人机交互模块概要见表 D.1。

表 D.1 人机交互模块概要

接口	
UserInput	用户输入消息定义。
NgbKeyListener	按键事件监听器接口，由需要监听 KeyEvent 的应用程序实现。
NgbMouseListener	鼠标事件监听器接口，由需要监听 MouseEvent 的应用程序实现。
NgbVoiceListener	语音事件监听器接口，由需要监听语音识别的应用程序实现。
类	
FrontPanel	前面板信息显示输出控制，包括 LED 指示灯和 LED 数码管显示控制。
NgbInputManager	输入控制管理器，用于按键、鼠标等事件的监听接收以及注入控制。
NgbVoiceManager	语音相关控制管理器，用于语音识别、语音播报等相关功能的控制和实现。
事件	
NgbInputEvent	NGB 输入事件类，是本包中其他 NGB 扩展输入事件的基类。
KeyEvent	按键事件类，继承 NgbInputEvent 类。
MouseEvent	鼠标事件类，继承 NgbInputEvent 类。

用户输入是指用户通过遥控器、鼠标、键盘、前面板按键等一些输入设备向接收终端发送用户指令，这些用户指令统一封装成按键消息进行处理，本部分约定按键的键值：

——鼠标、键盘的键值与 KeyEvent 和 MouseEvent 保持兼容；

——遥控器和前面板的键值，部分与键盘的键值统一，其余为扩展定义。

用户输入消息中含有消息来源指示，用以区分消息源。

用户输出是指接收终端通过前面板或显示屏幕向用户反馈信息。本附录主要约定前面板的信息输出。

D.2.1 接口org.ngb.interact.UserInput

原型：public interface org.ngb.interact.UserInput
描述：用户输入接口。

D.2.1.1 常量域——用户输入消息来源

D.2.1.1.1 INPUT_UNKNOWN

原型：public static final int INPUT_UNKNOWN = 0x0 (0)
描述：按键消息来源——表示未知。

D.2.1.1.2 INPUT_FRONTPANEL

原型：public static final int INPUT_FRONTPANEL = 0x1 (1)
描述：按键消息来源——表示接收机前面板按键输入。

D.2.1.1.3 INPUT_KEYBOARD

原型：public static final int INPUT_KEYBOARD = 0x2 (2)
描述：按键消息来源——表示 PC 键盘按键输入。

D.2.1.1.4 INPUT_MOUSE

原型：public static final int INPUT_MOUSE = 0x3 (3)
描述：按键消息来源——表示鼠标按键输入。

D.2.1.1.5 INPUT_REMOTECTRL

原型：public static final int INPUT_REMOTECTRL = 0x4 (4)
描述：按键消息来源——表示遥控器按键输入。

D.2.1.1.6 INPUT_USER

原型：public static final int INPUT_USER = 0x5 (5)
描述：按键消息来源——表示用户输入。

D.2.1.2 常量域——键盘按键状态

D.2.1.2.1 KEY_TYPED

原型：public static final int KEY_TYPED = 0x0190 (400)
描述：按键状态常量——表示击键状态，兼容 KeyEvent.KEY_TYPED。

D.2.1.2.2 KEY_PRESSED

原型：public static final int KEY_PRESSED = 0x0191 (401)
描述：按键状态常量——表示按下状态，兼容 KeyEvent.KEY_PRESSED。

D.2.1.2.3 KEY_RELEASED

原型：public static final int KEY_RELEASED = 0x0192 (402)
描述：按键状态常量——表示释放状态，兼容 KeyEvent.KEY_RELEASED。

D.2.1.3 常量域——鼠标按键状态

D.2.1.3.1 MOUSE_PRESSED

原型：public static final int MOUSE_PRESSED = 0x01F5 (501)
描述：鼠标键状态常量——表示按下状态，兼容 MouseEvent.MOUSE_PRESSED。

D.2.1.3.2 MOUSE_RELEASED

原型：public static final int MOUSE_RELEASED = 0x01F6（502）
描述：鼠标键状态常量——表示释放状态，兼容 MouseEvent.MOUSE_RELEASED。

D.2.1.3.3 MOUSE_MOVED

原型：public static final int MOUSE_MOVED = 0x01F7（503）
描述：鼠标键状态常量——表示移动状态，兼容 MouseEvent.MOUSE_MOVED。

D.2.1.4 常量域——鼠标按键消息码

D.2.1.4.1 MOUSE_NOBUTTON

原型：public static final int MOUSE_NOBUTTON = 0x00（0）
描述：鼠标按键消息码——表示一个无效的按钮，兼容 MouseEvent.NOBUTTON。

D.2.1.4.2 MOUSE_LBUTTON

原型：public static final int MOUSE_LBUTTON = 0x01（1）
描述：鼠标按键消息码——表示鼠标左键，兼容 MouseEvent.BUTTON1。

D.2.1.4.3 MOUSE_MBUTTON

原型：public static final int MOUSE_MBUTTON = 0x02（2）
描述：鼠标按键消息码——表示鼠标中间键，兼容 MouseEvent.BUTTON2。

D.2.1.4.4 MOUSE_RBUTTON

原型：public static final int MOUSE_RBUTTON = 0x03（3）
描述：鼠标按键消息码——表示鼠标右键，兼容 MouseEvent.BUTTON3。

D.2.1.5 常量域——按键消息码

表 D.2 按键消息码键值映射表

键值	消息名称	消息说明
0	KEYCODE_UNKNOWN	未知
1	KEYCODE_SOFT_LEFT	
2	KEYCODE_SOFT_RIGHT	
3	KEYCODE_HOME	Home
4	KEYCODE_BACK	返回
5	KEYCODE_CALL	拨号
6	KEYCODE_ENDCALL	挂机
7	KEYCODE_0	0
8	KEYCODE_1	1
9	KEYCODE_2	2
10	KEYCODE_3	3
11	KEYCODE_4	4
12	KEYCODE_5	5

表 D.2（续）

键值	消息名称	消息说明
13	KEYCODE_6	6
14	KEYCODE_7	7
15	KEYCODE_8	8
16	KEYCODE_9	9
17	KEYCODE_STAR	*
18	KEYCODE_POUND	#
19	KEYCODE_DPAD_UP	导航键上
20	KEYCODE_DPAD_DOWN	导航键下
21	KEYCODE_DPAD_LEFT	导航键左
22	KEYCODE_DPAD_RIGHT	导航键右
23	KEYCODE_DPAD_CENTER	导航键中
24	KEYCODE_VOLUME_UP	音量增加
25	KEYCODE_VOLUME_DOWN	音量减小
26	KEYCODE_POWER	电源
27	KEYCODE_CAMERA	拍照
28	KEYCODE_CLEAR	
29	KEYCODE_A	A
30	KEYCODE_B	B
31	KEYCODE_C	C
32	KEYCODE_D	D
33	KEYCODE_E	E
34	KEYCODE_F	F
35	KEYCODE_G	G
36	KEYCODE_H	H
37	KEYCODE_I	I
38	KEYCODE_J	J
39	KEYCODE_K	K
40	KEYCODE_L	L
41	KEYCODE_M	M
42	KEYCODE_N	N
43	KEYCODE_O	O
44	KEYCODE_P	P
45	KEYCODE_Q	Q
46	KEYCODE_R	R
47	KEYCODE_S	S
48	KEYCODE_T	T
49	KEYCODE_U	U
50	KEYCODE_V	V

表 D.2（续）

键值	消息名称	消息说明
51	KEYCODE_W	W
52	KEYCODE_X	X
53	KEYCODE_Y	Y
54	KEYCODE_Z	Z
55	KEYCODE_COMMA	,
56	KEYCODE_PERIOD	.
57	KEYCODE_ALT_LEFT	左 ALT
58	KEYCODE_ALT_RIGHT	右 ALT
59	KEYCODE_SHIFT_LEFT	左 SHIFT
60	KEYCODE_SHIFT_RIGHT	右 SHIFT
61	KEYCODE_TAB	制表符
62	KEYCODE_SPACE	空格
63	KEYCODE_SYM	Symbol modifier
64	KEYCODE_EXPLORER	浏览
65	KEYCODE_ENVELOPE	邮件
66	KEYCODE_ENTER	回车
67	KEYCODE_DEL	退格
68	KEYCODE_GRAVE	`
69	KEYCODE_MINUS	-
70	KEYCODE_EQUALS	=
71	KEYCODE_LEFT_BRACKET	[
72	KEYCODE_RIGHT_BRACKET	]
73	KEYCODE_BACKSLASH	\
74	KEYCODE_SEMICOLON	;
75	KEYCODE_APOSTROPHE	’
76	KEYCODE_SLASH	/
77	KEYCODE_AT	@
78	KEYCODE_NUM	Num
79	KEYCODE_HEADSETHOOK	耳机接听键
80	KEYCODE_FOCUS	拍照对焦
81	KEYCODE_PLUS	+
82	KEYCODE_MENU	菜单
83	KEYCODE_NOTIFICATION	通知
84	KEYCODE_SEARCH	搜索
85	KEYCODE_MEDIA_PLAY_PAUSE	多媒体键 播放/暂停
86	KEYCODE_MEDIA_STOP	多媒体键 停止
87	KEYCODE_MEDIA_NEXT	多媒体键 下一首
88	KEYCODE_MEDIA_PREVIOUS	多媒体键 上一首
89	KEYCODE_MEDIA_REWIND	多媒体键 快退

表 D.2（续）

键值	消息名称	消息说明
90	KEYCODE_MEDIA_FAST_FORWARD	多媒体键 快进
91	KEYCODE_MUTE	话筒静音
92	KEYCODE_PAGE_UP	上翻页
93	KEYCODE_PAGE_DOWN	下翻页
94	KEYCODE_PICTSYMBOLS	Picture Symbols modifier
95	KEYCODE_SWITCH_CHARSET	Switch Charset modifier
96	KEYCODE_BUTTON_A	游戏手柄按钮 A
97	KEYCODE_BUTTON_B	游戏手柄按钮 B
98	KEYCODE_BUTTON_C	游戏手柄按钮 C
99	KEYCODE_BUTTON_X	游戏手柄按钮 X
100	KEYCODE_BUTTON_Y	游戏手柄按钮 Y
101	KEYCODE_BUTTON_Z	游戏手柄按钮 Z
102	KEYCODE_BUTTON_L1	游戏手柄按钮 L1
103	KEYCODE_BUTTON_R1	游戏手柄按钮 L2
104	KEYCODE_BUTTON_L2	游戏手柄按钮 R1
105	KEYCODE_BUTTON_R2	游戏手柄按钮 R2
106	KEYCODE_BUTTON_THUMBL	Left Thumb Button
107	KEYCODE_BUTTON_THUMBR	Right Thumb Button
108	KEYCODE_BUTTON_START	游戏手柄按钮 Start
109	KEYCODE_BUTTON_SELECT	游戏手柄按钮 Select
110	KEYCODE_BUTTON_MODE	游戏手柄按钮 Mode
111	KEYCODE_ESCAPE	ESC 键
112	KEYCODE_FORWARD_DEL	删除
113	KEYCODE_CTRL_LEFT	左 CTRL
114	KEYCODE_CTRL_RIGHT	右 CTRL
115	KEYCODE_CAPS_LOCK	大写锁定
116	KEYCODE_SCROLL_LOCK	滚动锁定
117	KEYCODE_META_LEFT	左 meta
118	KEYCODE_META_RIGHT	右 meta
119	KEYCODE_FUNCTION	Fn
120	KEYCODE_SYSRQ	系统请求/屏幕截图
121	KEYCODE_BREAK	休息/暂停
122	KEYCODE_MOVE_HOME	光标移动到开始
123	KEYCODE_MOVE_END	光标移动到末尾
124	KEYCODE_INSERT	插入
125	KEYCODE_FORWARD	向前移动光标
126	KEYCODE_MEDIA_PLAY	多媒体键 播放
127	KEYCODE_MEDIA_PAUSE	多媒体键 暂停

表 D.2（续）

键值	消息名称	消息说明
128	KEYCODE_MEDIA_CLOSE	多媒体键 关闭
129	KEYCODE_MEDIA_EJECT	多媒体键 弹出
130	KEYCODE_MEDIA_RECORD	多媒体键 录制
131	KEYCODE_F1	F1
132	KEYCODE_F2	F2
133	KEYCODE_F3	F3
134	KEYCODE_F4	F4
135	KEYCODE_F5	F5
136	KEYCODE_F6	F6
137	KEYCODE_F7	F7
138	KEYCODE_F8	F8
139	KEYCODE_F9	F9
140	KEYCODE_F10	F10
141	KEYCODE_F11	F11
142	KEYCODE_F12	F12
143	KEYCODE_NUM_LOCK	小键盘锁
144	KEYCODE_NUMPAD_0	小键盘 0
145	KEYCODE_NUMPAD_1	小键盘 1
146	KEYCODE_NUMPAD_2	小键盘 2
147	KEYCODE_NUMPAD_3	小键盘 3
148	KEYCODE_NUMPAD_4	小键盘 4
149	KEYCODE_NUMPAD_5	小键盘 5
150	KEYCODE_NUMPAD_6	小键盘 6
151	KEYCODE_NUMPAD_7	小键盘 7
152	KEYCODE_NUMPAD_8	小键盘 8
153	KEYCODE_NUMPAD_9	小键盘 9
154	KEYCODE_NUMPAD_DIVIDE	小键盘/
155	KEYCODE_NUMPAD_MULTIPLY	小键盘*
156	KEYCODE_NUMPAD_SUBTRACT	小键盘-
157	KEYCODE_NUMPAD_ADD	小键盘+
158	KEYCODE_NUMPAD_DOT	小键盘.
159	KEYCODE_NUMPAD_COMMA	小键盘,
160	KEYCODE_NUMPAD_ENTER	小键盘 Enter
161	KEYCODE_NUMPAD_EQUALS	小键盘=
162	KEYCODE_NUMPAD_LEFT_PAREN	小键盘(
163	KEYCODE_NUMPAD_RIGHT_PAREN	小键盘)
164	KEYCODE_VOLUME_MUTE	扬声器静音
165	KEYCODE_INFO	Info
166	KEYCODE_CHANNEL_UP	频道加

表 D.2（续）

键值	消息名称	消息说明
167	KEYCODE_CHANNEL_DOWN	频道减
168	KEYCODE_ZOOM_IN	放大
169	KEYCODE_ZOOM_OUT	缩小
170	KEYCODE_TV	电视键 直播
171	KEYCODE_WINDOW	电视键 画中画
172	KEYCODE_GUIDE	电视键 节目指南
173	KEYCODE_DVR	电视键 录制
174	KEYCODE_BOOKMARK	电视键 书签
175	KEYCODE_CAPTIONS	文本字幕开关
176	KEYCODE_SETTINGS	系统设置
177	KEYCODE_TV_POWER	电视键 电源开关
178	KEYCODE_TV_INPUT	电视键 输入源
179	KEYCODE_STB_POWER	电视键 外部机顶盒电源开关
180	KEYCODE_STB_INPUT	电视键 外部机顶盒输入源
181	KEYCODE_AVR_POWER	电视键 家庭影院电源开关
182	KEYCODE_AVR_INPUT	电视键 家庭影院输入源
183	KEYCODE_PROG_RED	电视键 红
184	KEYCODE_PROG_GREEN	电视键 绿
185	KEYCODE_PROG_YELLOW	电视键 黄
186	KEYCODE_PROG_BLUE	电视键 蓝
187	KEYCODE_APP_SWITCH	应用切换键
188	KEYCODE_BUTTON_1	通用游戏手柄按钮 #1
189	KEYCODE_BUTTON_2	通用游戏手柄按钮 #2
190	KEYCODE_BUTTON_3	通用游戏手柄按钮 #3
191	KEYCODE_BUTTON_4	通用游戏手柄按钮 #4
192	KEYCODE_BUTTON_5	通用游戏手柄按钮 #5
193	KEYCODE_BUTTON_6	通用游戏手柄按钮 #6
194	KEYCODE_BUTTON_7	通用游戏手柄按钮 #7
195	KEYCODE_BUTTON_8	通用游戏手柄按钮 #8
196	KEYCODE_BUTTON_9	通用游戏手柄按钮 #9
197	KEYCODE_BUTTON_10	通用游戏手柄按钮 #10
198	KEYCODE_BUTTON_11	通用游戏手柄按钮 #11
199	KEYCODE_BUTTON_12	通用游戏手柄按钮 #12
200	KEYCODE_BUTTON_13	通用游戏手柄按钮 #13
201	KEYCODE_BUTTON_14	通用游戏手柄按钮 #14
202	KEYCODE_BUTTON_15	通用游戏手柄按钮 #15
203	KEYCODE_BUTTON_16	通用游戏手柄按钮 #16
204	KEYCODE_LANGUAGE_SWITCH	输入法语言切换键
205	KEYCODE_MANNER_MODE	静音/震动模式开关

表 D.2（续）

键值	消息名称	消息说明
206	KEYCODE_3D_MODE	2D/3D 开关
207	KEYCODE_CONTACTS	通讯录
208	KEYCODE_CALENDAR	日历
209	KEYCODE_MUSIC	音乐
210	KEYCODE_CALCULATOR	计算器
211	KEYCODE_ZENKAKU_HANKAKU	日语 全宽/半宽键
212	KEYCODE_EISU	日语 字母数字键
213	KEYCODE_MUHENKAN	日语 非转换键
214	KEYCODE_HENKAN	日语 转换键
215	KEYCODE_KATAKANA_HIRAGANA	日语 片假名/平假名键
216	KEYCODE_YEN	日元
217	KEYCODE_RO	日语 XX
218	KEYCODE_KANA	日语 XX
219	KEYCODE_ASSIST	助手
220	KEYCODE_BRIGHTNESS_DOWN	亮度减
221	KEYCODE_BRIGHTNESS_UP	亮度加
222	KEYCODE_MEDIA_AUDIO_TRACK	音轨
223	KEYCODE_SLEEP	休眠
224	KEYCODE_WAKEUP	唤醒
225	KEYCODE_PAIRING	外围设备配对
226	KEYCODE_MEDIA_TOP_MENU	顶部菜单
227	KEYCODE_11	11
228	KEYCODE_12	12
229	KEYCODE_LAST_CHANNEL	上一个频道
230	KEYCODE_TV_DATA_SERVICE	数据业务
231	KEYCODE_VOICE_ASSIST	语音助手
232	KEYCODE_TV_RADIO_SERVICE	视频音频切换
233	KEYCODE_TV_TELETEXT	图文电视开关
234	KEYCODE_TV_NUMBER_ENTRY	切台数字键
235	KEYCODE_TV_TERRESTRIAL_ANALOG	模拟地面电视
236	KEYCODE_TV_TERRESTRIAL_DIGITAL	数字地面电视
237	KEYCODE_TV_SATELLITE	数字卫星电视
238	KEYCODE_TV_SATELLITE_BS	日本 BS 数字卫星电视
239	KEYCODE_TV_SATELLITE_CS	日本 CS 数字卫星电视
240	KEYCODE_TV_SATELLITE_SERVICE	BS/CS 切换
241	KEYCODE_TV_NETWORK	双向/广播切换
242	KEYCODE_TV_ANTENNA_CABLE	天线/有线切换
243	KEYCODE_TV_INPUT_HDMI_1	HDMI 输入 1
244	KEYCODE_TV_INPUT_HDMI_2	HDMI 输入 2

表 D.2（续）

键值	消息名称	消息说明
245	KEYCODE_TV_INPUT_HDMI_3	HDMI 输入 3
246	KEYCODE_TV_INPUT_HDMI_4	HDMI 输入 4
247	KEYCODE_TV_INPUT_COMPOSITE_1	CVBS 输入 1
248	KEYCODE_TV_INPUT_COMPOSITE_2	CVBS 输入 2
249	KEYCODE_TV_INPUT_COMPONENT_1	YPbPr 输入 1
250	KEYCODE_TV_INPUT_COMPONENT_2	YPbPr 输入 2
251	KEYCODE_TV_INPUT_VGA_1	VGA 输入
252	KEYCODE_TV_AUDIO_DESCRIPTION	音频描述符开关
253	KEYCODE_TV_AUDIO_DESCRIPTION_MIX_UP	音频描述符 音量加
254	KEYCODE_TV_AUDIO_DESCRIPTION_MIX_DOWN	音频描述符 音量减
255	KEYCODE_TV_ZOOM_MODE	缩放模式
256	KEYCODE_TV_CONTENTS_MENU	内容菜单
257	KEYCODE_TV_MEDIA_CONTEXT_MENU	上下文菜单
258	KEYCODE_TV_TIMER_PROGRAMMING	时间调整
259	KEYCODE_HELP	帮助
260	KEYCODE_NAVIGATE_PREVIOUS	导航 上一个
261	KEYCODE_NAVIGATE_NEXT	导航 下一个
262	KEYCODE_NAVIGATE_IN	导航 进入
263	KEYCODE_NAVIGATE_OUT	导航 退出
264	KEYCODE_STEM_PRIMARY	穿戴 电源/重启
265	KEYCODE_STEM_1	穿戴 通用键 1
266	KEYCODE_STEM_2	穿戴 通用键 2
267	KEYCODE_STEM_3	穿戴 通用键 3
268	KEYCODE_DPAD_UP_LEFT	导航 左上
269	KEYCODE_DPAD_DOWN_LEFT	导航 左下
270	KEYCODE_DPAD_UP_RIGHT	导航 右上
271	KEYCODE_DPAD_DOWN_RIGHT	导航 右下
272	KEYCODE_MEDIA_SKIP_FORWARD	多媒体 快进
273	KEYCODE_MEDIA_SKIP_BACKWARD	多媒体 快退
274	KEYCODE_MEDIA_STEP_FORWARD	多媒体 步进
275	KEYCODE_MEDIA_STEP_BACKWARD	多媒体 步退
276	KEYCODE_SOFT_SLEEP	软休眠
277	KEYCODE_CUT	剪切
278	KEYCODE_COPY	复制
279	KEYCODE_PASTE	粘贴
280	KEYCODE_SYSTEM_NAVIGATION_UP	系统导航 上
281	KEYCODE_SYSTEM_NAVIGATION_DOWN	系统导航 下
282	KEYCODE_SYSTEM_NAVIGATION_LEFT	系统导航 左
283	KEYCODE_SYSTEM_NAVIGATION_RIGHT	系统导航 右

表 D.2（续）

键值	消息名称	消息说明
1000	KEYCODE_VK_CANCEL	取消键。
1001	KEYCODE_VK_PRINT	打印键。
1002	KEYCODE_VK_EXECUTE	执行键。
1003	KEYCODE_VK_SEPARATOR	分割符号键。
1004	KEYCODE_VK_AMPERSAND	“&”键
1005	KEYCODE_VK_QUOTEDBL	“ “ ”键。
1006	KEYCODE_VK_LESS	″<″键。
1007	KEYCODE_VK_GREATER	”>“键。
1008	KEYCODE_VK_BRACELEFT	″{″键。
1009	KEYCODE_VK_BRACERIGHT	″}″键。
1010	KEYCODE_PICTUREMODE	画面模式
1011	KEYCODE_SOURCE	信源
1012	KEYCODE_TVSETUP	电视设置
1013	KEYCODE_RECALL	
1014	KEYCODE_HEADSET_IN	耳机插入
1015	KEYCODE_HEADSET_OUT	耳机拔出
1016	KEYCODE_MIC_ON	麦克打开
1017	KEYCODE_ICLOUD	
1018	KEYCODE_VOIP	VOIP 快捷启动键
1019	KEYCODE_RESOLUTION_RATIO	分辨率
1020	KEYCODE_AUDIO	音频模式切换（立体声、左/右声道）
1021	KEYCODE_NPVR	准视频录制
1022	KEYCODE_PQ	
1023	KEYCODE_QUIT	退出
1024	KEYCODE_SERVICE	服务
1025	KEYCODE_CHANNEL_ALTERNATE	频道交换
1026	KEYCODE_FAST_REVERSE	
1027	KEYCODE_SD	标清
1028	KEYCODE_HD	高清
1029	KEYCODE_SOUND_EFFECT	音效
1030	KEYCODE_SMART	智能
1031	KEYCODE_REFRESH	刷新键，网页实时刷新
1032	KEYCODE_VK_COLON	“:”键。
1033	KEYCODE_VK_CIRCUMFLEX	“^”键。
1034	KEYCODE_VK_DOLLAR	“$”键。
1035	KEYCODE_VK_EURO_SIGN	欧元符号键。
1036	KEYCODE_VK_EXCLAMATION_MARK	“!”键。
1037	KEYCODE_VK_INVERTED_EXCLAMATION_MARK	反向“!”键。
1038	KEYCODE_VK_UNDERSCORE	“_”键。

表 D.2（续）

键值	消息名称	消息说明
1039	KEYCODE_RCK_SELECT	选择键。
1040	KEYCODE_RCK_FAVORITE	“喜爱/收藏”键。
1041	KEYCODE_RCK_LANGUAGE	语言选择键。
1042	KEYCODE_RCK_SOFT_KEYBOARD	软键盘键。
1043	KEYCODE_RCK_RESUME	恢复键。
1044	KEYCODE_RCK_REVIEW	重看键。
1045	KEYCODE_RCK_REWIND	回转键。
1046	KEYCODE_RCK_GOTO	跳转键。
1047	KEYCODE_RCK_TITLE	"主题/标题"键。
1048	KEYCODE_RCK_POSITION	定位键。
1049	KEYCODE_RCK_ANGLE	"角度/视角"选择键。
1050	KEYCODE_RCK_SLOW	慢放键。
1051	KEYCODE_RCK_PROGRAM_PARADE	节目预览键。
1052	KEYCODE_RCK_RECOMMEND	推荐键。
1053	KEYCODE_RCK_DESCRIPTION	"说明/描述"键。
1054	KEYCODE_RCK_HISTORY_FORWORD	标签下一个键。
1055	KEYCODE_RCK_HISTORY_BACKWORD	标签上一个键。
1056	KEYCODE_RCK_RADIO	广播键。
1057	KEYCODE_RCK_VOD	点播键。
1058	KEYCODE_RCK_NVOD	准视频点播键。
1059	KEYCODE_RCK_MOVIE	电影键。
1060	KEYCODE_RCK_ALBUM	专辑键。
1061	KEYCODE_RCK_WEB	WEB 键。
1062	KEYCODE_RCK_STOCK	股票键。
1063	KEYCODE_RCK_MESSAGING	消息键。
1064	KEYCODE_RCK_READER	阅读键。
1065	KEYCODE_RCK_GAME	游戏键。
1066	KEYCODE_RCK_MOSAIC	马赛克键。
1067	KEYCODE_RCK_LIST	列表键。
1068	KEYCODE_RCK_REFRESH	刷新键。
1069	KEYCODE_RCK_BUSINESS	商务键。
1070	KEYCODE_RCK_BUY	购买键。
1071	KEYCODE_RCK_FORETELL	预告键。
1072	KEYCODE_RCK_STATUS	状态键。
1073	KEYCODE_RCK_SECURITIES	证券键。
1074	KEYCODE_RCK_CLASS	分类键。
1075	KEYCODE_RCK_DISPLAY	"显示/展现"键。

D.2.2 接口org.ngb.interact.NgbKeyListener

原型：public interface org.ngb.interact.NgbKeyListener
描述：按键事件监听器。

D.2.2.1 方法

D.2.2.1.1 notifyKeyEvent

原型：public boolean notifyKeyEvent(org.ngb.interact.KeyEvent event)
描述：按键事件通知。
参数：event - org.ngb.interact.KeyEvent事件对象，表示一次按键事件。
返回：boolean型，按键事件处理结果，true 处理成功 false 未处理或处理失败。

D.2.3 接口org.ngb.interact.NgbMouseListener

原型：public interface org.ngb.interact.NgbMouseListener
描述：鼠标事件监听器。

D.2.3.1 方法

D.2.3.1.1 notifyMouseEvent

原型：public boolean notifyMouseEvent(org.ngb.interact.MouseEvent event)
描述：鼠标事件通知。
参数：event - org.ngb.interact.MouseEvent事件对象，标识一次鼠标事件。
返回：boolean型，表示鼠标事件处理结果，true 处理成功 false 未处理或处理失败。

D.2.4 接口org.ngb.interact.NgbVoiceListener

原型：public interface org.ngb.interact.NgbVoiceListener
描述：ngb语音事件监听器。

D.2.4.1 常量域——语音错误码

D.2.4.1.1 ERROR_NETWORK_TIMEOUT

原型：public static final int ERROR_NETWORK_TIMEOUT = 1
描述：错误码——网络超时。

D.2.4.1.2 ERROR_NETWORK

原型：public static final int ERROR_NETWORK = 2
描述：错误码——其他网络相关错误。

D.2.4.1.3 ERROR_AUDIO

原型：public static final int ERROR_AUDIO = 3
描述：错误码——音频录制错误。

D.2.4.1.4 ERROR_SERVER

原型：public static final int ERROR_SERVER = 4
描述：错误码——服务器返回错误状态。

D.2.4.1.5 ERROR_CLIENT

原型：public static final int ERROR_CLIENT = 5
描述：错误码——终端侧出现错误。

D.2.4.1.6 ERROR_SPEECH_TIMEOUT

原型：public static final int ERROR_SPEECH_TIMEOUT = 6
描述：错误码——无语音输入。

D.2.4.1.7 ERROR_NO_MATCH

原型：public static final int ERROR_NO_MATCH = 7
描述：错误码——未找到匹配的语音识别结果。

D.2.4.1.8 ERROR_RECOGNIZER_BUSY

原型：public static final int ERROR_RECOGNIZER_BUSY = 8
描述：错误码——语音识别服务正忙。

D.2.4.1.9 ERROR_INSUFFICIENT_PERMISSIONS

原型：public static final int ERROR_INSUFFICIENT_PERMISSIONS = 9
描述：错误码——没有足够的权限。

D.2.4.2 方法

D.2.4.2.1 onVoiceStart

原型：public void onVoiceStart()
描述：语音输入开始。
参数：无。
返回：无。

D.2.4.2.2 onVoiceEnd

原型：public void onVoiceStart()
描述：语音输入结束。
参数：无。
返回：无。

D.2.4.2.3 onVoiceResult

原型：public void onVoiceResult(java.lang.String msg)
描述：语义识别的结果。
参数：字符串型，表示此次语音识别的结果，结果格式为json类型，包含具体的语音原文和语义指令。
返回：无。

D.2.4.2.4 onError

原型：public void onError(int code)
描述：语音过程中的错误。

参数：整型，语音过程中的错误码值，详见“语音错误码”常量域定义。
返回：无。

D.2.5 类org.ngb.interact.FrontPanel

原型：public class org.ngb.interact.FrontPanel
描述：前面板信息显示输出控制，包括LED指示灯和LED数码管显示控制。

D.2.5.1 常量域——对齐方式

D.2.5.1.1 ALIGN_CENTER

原型：public static final int ALIGN_CENTER = 0
描述：前面板字符对齐方式——水平居中。

D.2.5.1.2 ALIGN_LEFT

原型：public static final int ALIGN_LEFT = 1
描述：前面板字符对齐方式——水平居左。

D.2.5.1.3 ALIGN_RIGHT

原型：public static final int ALIGN_RIGHT = 2
描述：前面板字符对齐方式——水平居右。

D.2.5.2 常量域——指示状态

D.2.5.2.1 STATUS_OFF

原型：public static final int STATUS_OFF = 0
描述：指示状态——关闭。

D.2.5.2.2 STATUS_ON

原型：public static final int STATUS_ON = 1
描述：指示状态——开启。

D.2.5.2.3 STATUS_UNKNOWN

原型：public static final int STATUS_UNKNOWN = 2
描述：指示状态——未知。

D.2.5.3 常量域——指示类型

D.2.5.3.1 TYPE_MAIL

原型：public static final int TYPE_MAIL = 0
描述：指示类型——邮件。

D.2.5.3.2 TYPE_SIGNAL

原型：public static final int TYPE_SIGNAL = 1
描述：指示类型——信号。

D.2.5.3.3 TYPE_POWER

原型：public static final int TYPE_POWER = 2
描述：指示类型——电源。

D.2.5.3.4 TYPE_RADIO

原型：public static final int TYPE_RADIO = 3
描述：指示类型——广播。

D.2.5.4 方法

D.2.5.4.1 getInstance

原型：public static org.ngb.interact.FrontPanel getInstance()
描述：获取系统实现的前面板类的唯一实例。
参数：无。
返回：org.ngb.interact.FrontPanel对象，表示系统实现的前面板类单例。

D.2.5.4.2 clear

原型：public boolean clear()
描述：清除前面板显示信息，包括前面板显示的字符串信息、时间日期信息等。
参数：无。
返回：boolean型，表示清除结果，取值true表示清除成功，false表示清除失败。

D.2.5.4.3 displayDate

原型：public boolean displayDate(java.util.Date date)
描述：显示当前的时间日期信息。
参数：date - java.util.Date型，表示日期和时间。
返回：boolean型，表示显示结果，取值true表示显示成功，false表示显示失败。若终端不支持时间日期显示，可不响应该方法的调用，并返回false。

D.2.5.4.4 displayText

原型：public boolean displayText(java.lang.String str)
描述：显示字符串，缺省采用水平居中对齐方式显示。
参数：str - java.lang.String对象，表示待显示的字符串。
返回：boolean型，取值true 表示显示成功，false表示显示失败。

D.2.5.4.5 displayText

原型：public boolean displayText(java.lang.String str, int align)
描述：采用指定对齐方式显示字符串。
参数：str - java.lang.String对象，表示待显示的字符串；
align－int型，表示水平对齐方式，取值详见org.ngb.interact.FrontPanel类的“对齐方式”常量域定义。
返回：boolean型，取值true表示显示成功，false表示显示失败。

D.2.5.4.6 getStatus

原型：public int getStatus(int type)
描述：根据指定的类型获取指示状态。
参数：type - int型，表示指示类型，取值详见org.ngb.interact.FrontPanel类的“指示类型”常量域定义。
返回：int型，表示指示状态。
——若type参数指定一个有效的指示类型，则返回其实际的状态（即取值STATUS_ON或STATUS_OFF）；
——若type参数指定一个无效的指示类型，则返回STATUS_UNKNOWN。

D.2.5.4.7 setStatus

原型：public boolean setStatus(int type, int value)
描述：设置指示状态。
参数：type - int型，表示指示类型，取值详见org.ngb.interact.FrontPanel类的“指示类型”常量域定义；
value - int型，表示指示状态，取值详见org.ngb.interact.FrontPanel类的“指示状态”常量域定义。
返回：boolean型，取值true表示设置成功，false表示设置失败。

D.2.5.4.8 getMaxChars

原型：public int getMaxChars()
描述：获取前面板支持的显示字符数量。
参数：无。
返回：int型，表示前面板支持的显示字符数量。

D.2.6 类org.ngb.interact.NgbInputManager

原型：public class org.ngb.interact.NgbInputManager
描述：输入/输出控制管理器，用于按键、鼠标等事件的监听接收以及注入控制。

D.2.6.1 方法

D.2.6.1.1 getInstance

原型：public static org.ngb.interact.NgbInputManager getInstance()
描述：获取系统实现的输入/输出控制管理类的唯一实例。
参数：无。
返回：org.ngb.interact.NgbInputManager对象，表示系统实现的输入/输出控制管理类单例。

D.2.6.1.2 addKeyEventListener

原型：public void addKeyEventListener(org.ngb.interact.NgbKeyListener listener)
描述：向系统注册指定的按键事件监听器。
参数：listener－org.ngb.interact.NgbKeyListener对象，按键事件监听器，用于按键事件的通知。
返回：无。

D.2.6.1.3 removeKeyEventListener

原型：public void removeKeyEventListener(org.ngb.interact.NgbKeyListener listener)
描述：从系统移除指定的按键事件监听器。
参数：listener－org.ngb.interact.NgbKeyListener对象，按键事件监听器，用于按键事件的通知。
返回：无。

D.2.6.1.4 injectKeyEvent

原型：public boolean injectKeyEvent(org.ngb.interact.KeyEvent event)
描述：向系统强制注入一个按键事件。
参数：event － org.ngb.interact.KeyEvent对象，按键事件。
返回：boolean型，表示注入按键事件结果，true 注入成功 false 注入失败。

D.2.6.1.5 addMouseEventListener

原型：public void addMouseEventListener(org.ngb.interact.NgbMouseListener listener)
描述：向系统注册指定的鼠标事件监听器。
参数：listener － org.ngb.interact.NgbMouseListener对象，鼠标事件监听器，用于鼠标事件的通知。
返回：无。

D.2.6.1.6 removeMouseEventListener

原型：public void removeMouseEventListener(org.ngb.interact.NgbMouseListener listener)
描述：从系统移除指定的鼠标事件监听器。
参数：listener － org.ngb.interact.NgbMouseListener对象，鼠标事件监听器，用于鼠标事件的通知。
返回：无。

D.2.6.1.7 injectMouseEvent

原型：public boolean injectMouseEvent(org.ngb.interact.MouseEvent event)
描述：向系统强制注入一个鼠标事件。
参数：event － org.ngb.interact.MouseEvent对象，鼠标事件。
返回：boolean型，注入鼠标事件结果，true 注入成功 false 注入失败。

D.2.7 类org.ngb.interact.NgbVoiceManager

原型：public class org.ngb.interact.NgbVoiceManager
描述：语音相关控制管理器，用于语音识别、语音播报等相关功能的控制和实现。

D.2.7.1 方法

D.2.7.1.1 getInstance

原型：public static org.ngb.interact.NgbVoiceManager getInstance(org.tvos.content.Context context,org.tvos.content.ComponentName cn)
描述：获取语音相关控制管理器唯一实例对象，用于语音识别、语音播报等相关功能的控制和实现。
参数：context － org.tvos.content.Context对象，表示当前使用语音的上下文场景；
cn － org.tvos.content.ComponentName对象，表示准备使用的语音服务引擎组件。
返回：org.ngb.interact.NgbVoiceManager对象，表示语音相关控制管理器类单例。

D.2.7.1.2 startListening

原型：public void startListening(org.tvos.content.Intent intent)
描述：启动语音监听。
参数：intent - Intent 对象，语音识别的相关参数。
返回：无。

D.2.7.1.3 stopListening

原型：public void stopListening()
描述：停止语音监听。
参数：无。
返回：无。

D.2.7.1.4 cancel

原型：public void cancel()
描述：取消此次语音识别。
参数：无。
返回：无。

D.2.7.1.5 setVoiceListener

原型：public void setVoiceListener(org.ngb.interact.NgbVoiceListener listener)
描述：设置语音监听器。
参数：listener - org.ngb.interact.NgbVoiceListener 对象，语音监听器。
返回：无。

D.2.7.1.6 release

原型：public void release()
描述：释放此语音管理器以及其占用资源。
参数：无。
返回：无。

D.2.8 事件org.ngb.interact.NgbInputEvent

原型：public abstract class org.ngb.interact.NgbInputEvent
描述：扩展的输入事件类，是按键、鼠标等输入事件的基类，包含输入事件产生事件、产生源类型等接口。

D.2.8.1 方法

D.2.8.1.1 getEventTime

原型：public abstract long getEventTime()
描述：获取输入事件产生的时间值。
参数：无。
返回：long型，表示输入事件产生的时间值，以机顶盒开机时间为基础，返回距离机顶盒开机时间的毫秒值。

D.2.8.1.2 getSource

原型：public abstract int getSource()
描述：获取输入事件的产生源。
参数：无。
返回：int型，表示输入事件产生的源，可取值为INPUT_UNKNOWN、INPUT_FRONTPANEL、INPUT_KEYBOARD、INPUT_MOUSE、INPUT_REMOTECTRL、INPUT_USER，详见UserInput接口的“用户输入消息来源”常量域定义。

D.2.9 事件org.ngb.interact.KeyEvent

原型：public class org.ngb.interact.KeyEvent extends org.ngb.interact.NgbInputEvent
描述：按键事件类，继承org.ngb.interact.NgbInputEvent类。

D.2.9.1 方法

D.2.9.1.1 getAction

原型：public int getAction()
描述：获取按键事件的状态。
参数：无。
返回：int型，表示按键事件的状态，可取值为KEY_TYPED、KEY_PRESSED或KEY_RELEASED，详见org.ngb.interact.UserInput接口的“键盘按键状态”常量域定义。

D.2.9.1.2 getCode

原型：public int getCode()
描述：获取按键事件的键值码。
参数：无。
返回：int型，表示按键事件键值码，取值详见org.ngb.interact.UserInput接口的按键消息码常量域定义。

D.2.10 事件org.ngb.interact.MouseEvent

原型：public class org.ngb.interact.MouseEvent extends org.ngb.interact.NgbInputEvent
描述：鼠标事件类，继承org.ngb.interact.NgbInputEvent类。

D.2.10.1 方法

D.2.10.1.1 getAction

原型：public int getAction()
描述：获取鼠标事件的状态。
参数：无。
返回：int型，表示鼠标事件的状态，可取值为MOUSE_PRESSED、MOUSE_RELEASED或MOUSE_MOVED，详见UserInput接口的“鼠标按键状态”常量域定义。

D.2.10.1.2 getCode

原型：public int getCode()
描述：获取鼠标事件的键值码。

参数：无。

返回：int型，表示鼠标事件键值码，取值见org.ngb.interact.UserInput接口的“鼠标按键消息码”常量域定义。

D.2.10.1.3 getButton

原型：public int getButton()

描述：当鼠标按下或者松开时，获取发出该消息的鼠标按钮。

参数：无。

返回：int 型，表示发出此消息的鼠标按键。

——当消息子类型为 MOUSE_RELEASED 或 MOUSE_PRESSED 时，则返回对应的鼠标按键码（MOUSE_LBUTTON/MOUSE_RBUTTON/MOUSE_MBUTTON）；

——当消息子类型为 MOUSE_MOVED 时，则返回 MOUSE_NOBUTTON。

D.2.10.1.4 getX

原型：public int getX()

描述：获取鼠标焦点在 X 轴的位置。

参数：无。

返回：int 型，表示鼠标焦点在 X 轴的位置，单位为像素。

D.2.10.1.5 getY

原型：public int getY()

描述：获取鼠标焦点在 Y 轴的位置。

参数：无。

返回：int型，表示鼠标焦点在Y轴的位置，单位为像素。

附 录 E
（规范性附录）
JAVA-AV 设置单元

E.1 概述

本附录定义了与AV设置相关的JAVA接口。

E.2 AV设置模块

AV 设置模块提供了音视频输出参数设置相关的类和方法。
AV 设置模块概要见表 E.1。

表 E.1 AV 设置模块概要

类	
AudioSetting	音频输入/输出单元的各种参数设置。
VideoSetting	视频输入/输出单元的各种参数设置。

E.2.1 类org.ngb.util.setting.AudioSetting

原型：public class org.ngb.util.setting.AudioSetting
描述：音频输入/输出单元的各种参数设置。系统应对应用程序的权限进行验证，只有特权应用才能调用本类提供的方法。
——音频输出参数设置包括音量等。
——音频输入参数设置保留将来扩展。
注：某广播节目的实际音量 = 全局音量 + 该广播节目相对于全局音量的增值。

E.2.1.1 常量域——声道类型

E.2.1.1.1 CHANNEL_STEREO

原型：public static final int CHANNEL_STEREO = 0
描述：声道类型——立体声。

E.2.1.1.2 CHANNEL_LEFT

原型：public static final int CHANNEL_LEFT = 1
描述：声道类型——左声道。

E.2.1.1.3 CHANNEL_RIGHT

原型：public static final int CHANNEL_RIGHT = 2
描述：声道类型——右声道。

E.2.1.1.4 CHANNEL_MIXED_MONO

原型：public static final int CHANNEL_MIXED_MONO = 3
描述：声道类型——混合声。

E.2.1.2 方法

E.2.1.2.1 getOutputInterfaceList

原型：public static java.lang.String[] getOutputInterfaceList()
描述：获取接收终端所有可用的音频输出端口列表。
参数：无。
返回：java.lang.String 对象数组，表示接收终端所有可用的音频输出端口名称，例如“RCA”、“S/PDIF”、“HDMI”等。若无音频输出端口，则返回的数组长度为0。

E.2.1.2.2 getOutputInterfaceStatus

原型：public static boolean getOutputInterfaceStatus(java.lang.String port)
描述：获取音频输出端口的使能状态。
参数：port - java.lang.String 对象，表示音频输出端口的名称。
注：音频输出端口名称由 getOutputInterfaceList()方法获得。
返回：boolean 型，表示音频输出端口的使能状态，取值 true 表示该音频输出端口允许输出，取值 false 表示该音频输出端口禁止输出。

E.2.1.2.3 disableOutputInterface

原型：public static boolean disableOutputInterface(java.lang.String port)
描述：禁止音频输出端口输出。
参数：port - java.lang.String 对象，表示音频输出接口的名称。
注：音频输出端口名称由 getOutputInterfaceList()方法获得。
返回：boolean 型，取值 true 表示成功，false 表示失败。

E.2.1.2.4 enableOutputInterface

原型：public static boolean enableOutputInterface(java.lang.String port)
描述：允许音频输出端口输出。
参数：port - java.lang.String 对象，表示音频输出端口的名称。
注：音频输出端口名称由 getOutputInterfaceList()方法获得。
返回：boolean 型，取值 true 表示成功，false 表示失败。

E.2.1.2.5 getOutputVolume

原型：public static int getOutputVolume()
描述：获取全局音频输出音量。
参数：无。
返回：int 型，表示输出音量的大小，取值范围为 0～100，0 表示静音，100 表示最大音量。

E.2.1.2.6 setOutputVolume

原型：public static boolean setOutputVolume(int volume)
描述：设置全局音频输出音量。

注：某广播节目的实际输出音量 = 全局输出音量 + 该广播节目相对于全局输出音量的增值。

参数：volume - int 型，表示输出音量的大小，取值范围 0～100，0 表示静音，100 表示最大音量。

返回：boolean 型，表示设置结果，取值 true 表示设置成功，false 表示设置失败。

E.2.1.2.7 getOutputChannelMode

原型：public static int getOutputChannelMode()

描述：获取输出声道类型。

参数：无。

返回：int 型，表示输出声道类型，取值详见 AudioSetting 类的“声道类型”常量域定义。

E.2.1.2.8 setOutputChannelMode

原型：public static boolean setOutputChannelMode(int type)

描述：设置输出声道。

参数：type－int型，表示待设置的声道类型，取值详见AudioSetting类的“声道类型”常量域定义。

返回：boolean型，表示设置结果，取值true表示设置成功，false表示设置失败。

E.2.1.2.9 getOutputSPDIFMode

原型：public static int getOutputSPDIFMode()

描述：获取 S/PDIF 输出接口数据格式（压缩或 PCM 格式）。

参数：无。

返回：int 型，表示 S/PDIF 输出接口数据格式，取值：

—— 0 - 表示 PCM 格式；

—— 1 - 表示压缩格式。

E.2.1.2.10 setOutputSPDIFMode

原型：public boolean static setOutputSPDIFMode(int mode)

描述：设置 S/PDIF 输出接口数据格式（压缩或 PCM 格式）。

参数：mode - int 型，表示 S/PDIF 输出接口数据格式，取值：

—— 0 - 表示 PCM 格式，即压缩音频由接收终端解码；

—— 1 - 表示压缩格式，即压缩音频由外接解码设备解码。

返回：boolean 型，表示设置结果，取值 true 表示设置成功，false 表示设置失败。

E.2.1.2.11 isMute

原型：public static boolean isMute()

描述：判断音频输出是否处于静音状态。

参数：无。

返回：boolean 型，取值 true 表示处于静音状态，false 表示处于有声状态。

E.2.1.2.12 mute

原型：public static boolean mute()

描述：静音。

参数：无。

返回：boolean 型，表示设置结果，取值 true 表示静音成功，false 表示静音失败。

E.2.1.2.13 unMute

原型：public static boolean unMute()
描述：取消静音。
参数：无。
返回：boolean 型，表示取消结果，取值 true 表示取消成功，false 表示取消失败。

E.2.1.2.14 getOutputHDMIFMode

原型：public static int getOutputHDMIFMode()
描述：获取HDMI输出接口数据格式。
参数：无。
返回：int型，表示HDMI输出接口数据格式.
—— 0 – 表示关闭；
—— 1 – 自动协商；
—— 2 – LPCM，输出解码后信号；
—— 3 – RAW，输出原始信号。

E.2.1.2.15 setOutputHDMIMode

原型：public static boolean setOutputHDMIMode(int mode)
描述：设置HDMI输出接口数据格式（压缩或PCM格式）。
参数：mode – int型，表示HDMI输出接口数据格式，取值：
—— 0 – 表示关闭；
—— 1 – 自动协商；
—— 2 – LPCM，输出解码后信号；
—— 3 – RAW，输出原始信号。
返回：boolean型，表示设置结果，取值true表示设置成功，false表示设置失败。

E.2.2 类org.ngb.util.setting.VideoSetting

原型：public class org.ngb.util.setting.VideoSetting
描述：视频输入/输出单元的各种参数设置。系统应对应用程序的权限进行验证，只有授权应用才能调用本类提供的方法。
——视频输出参数设置包括制式（分辨率、场频、幅型比）、亮度、对比度、透明度等；
——视频输入参数设置保留将来扩展。

E.2.2.1 常量域——匹配模式

E.2.2.1.1 MATCH_METHOD_LETTER_BOX

原型：public static final int MATCH_METHOD_LETTER_BOX = 1
描述：视频窗口匹配模式——信箱模式（letter_box）。

E.2.2.1.2 MATCH_METHOD_PAN_SCAN

原型：public static final int MATCH_METHOD_PAN_SCAN = 2
描述：视频窗口匹配模式——全画面模式（pan_scan）。

E.2.2.1.3 MATCH_METHOD_COMBINED

原型：public static final int MATCH_METHOD_COMBINED = 3
描述：视频窗口匹配模式——组合模式（combined）。

E.2.2.1.4 MATCH_METHOD_IGNORE

原型：public static final int MATCH_METHOD_IGNORE = 4
描述：视频窗口匹配模式——忽略模式（ignore）。

E.2.2.2 常量域——输出通道

E.2.2.2.1 VOUT_SD

原型：public static final int VOUT_SD = 1
描述：标清输出通道。

E.2.2.2.2 VOUT_HD

原型：public static final int VOUT_HD = 2
描述：高清输出通道。

E.2.2.3 常量域——输出制式

E.2.2.3.1 VOUT_STANDARD_UNKNOWN

原型：public static final int VOUT_STANDARD_UNKNOWN = 0
描述：视频输出制式：未知。

E.2.2.3.2 VOUT_STANDARD_NTSC_J

原型：public static final int VOUT_STANDARD_NTSC_J = 101
描述：视频输出制式：标清——NTSC-J/3.5795MHz 彩色副载波。

E.2.2.3.3 VOUT_STANDARD_NTSC_M

原型：public static final int VOUT_STANDARD_NTSC_M = 102
描述：视频输出制式：标清——NTSC-M/3.5795MHz 彩色副载波。

E.2.2.3.4 VOUT_STANDARD_NTSC_443

原型：public static final int VOUT_STANDARD_NTSC_443 = 103
描述：视频输出制式：标清——NTSC-443/4.4336MHz 彩色副载波。

E.2.2.3.5 VOUT_STANDARD_PAL_B

原型：public static final int VOUT_STANDARD_PAL_B = 211
描述：视频输出制式：标清——PAL-B（澳大利亚）。

E.2.2.3.6 VOUT_STANDARD_PAL_B1

原型：public static final int VOUT_STANDARD_PAL_B1 = 212
描述：视频输出制式：标清——PAL-B1（匈牙利）。

E.2.2.3.7 VOUT_STANDARD_PAL_D

原型：public static final int VOUT_STANDARD_PAL_D = 213
描述：视频输出制式：标清——PAL-D（中国大陆）。

E.2.2.3.8 VOUT_STANDARD_PAL_D1

原型：public static final int VOUT_STANDARD_PAL_D1 = 214
描述：视频输出制式：标清——PAL-D1（波兰）。

E.2.2.3.9 VOUT_STANDARD_PAL_G

原型：public static final int VOUT_STANDARD_PAL_G = 215
描述：视频输出制式：标清——PAL-G（欧洲）。

E.2.2.3.10 VOUT_STANDARD_PAL_H

原型：public static final int VOUT_STANDARD_PAL_H = 216
描述：视频输出制式：标清——PAL-H（欧洲）。

E.2.2.3.11 VOUT_STANDARD_PAL_I

原型：public static final int VOUT_STANDARD_PAL_I = 217
描述：视频输出制式：标清——PAL-I（英国、香港、澳门）。

E.2.2.3.12 VOUT_STANDARD_PAL_K

原型：public static final int VOUT_STANDARD_PAL_K = 218
描述：视频输出制式：标清——PAL-K（欧洲）。

E.2.2.3.13 VOUT_STANDARD_PAL_M

原型：public static final int VOUT_STANDARD_PAL_M = 220
描述：视频输出制式：标清——PAL-M（巴西）。

E.2.2.3.14 VOUT_STANDARD_PAL_N

原型：public static final int VOUT_STANDARD_PAL_N = 221
描述：视频输出制式：标清——PAL-N（牙买加、乌拉圭）。

E.2.2.3.15 VOUT_STANDARD_PAL_NC

原型：public static final int VOUT_STANDARD_PAL_NC = 222
描述：视频输出制式：标清——PAL-NC（阿根廷）。

E.2.2.3.16 VOUT_STANDARD_SECAM_B

原型：public static final int VOUT_STANDARD_SECAM_B = 311
描述：视频输出制式：标清——SECAM-B。

E.2.2.3.17 VOUT_STANDARD_SECAM_D

原型：public static final int VOUT_STANDARD_SECAM_D = 312

描述：视频输出制式：标清———SECAM-D。

E.2.2.3.18 VOUT_STANDARD_SECAM_G

原型：public static final int VOUT_STANDARD_SECAM_G = 313
描述：视频输出制式：标清———SECAM-G。

E.2.2.3.19 VOUT_STANDARD_SECAM_I

原型：public static final int VOUT_STANDARD_SECAM_I = 314
描述：视频输出制式：标清———SECAM-I。

E.2.2.3.20 VOUT_STANDARD_SECAM_K

原型：public static final int VOUT_STANDARD_SECAM_K = 315
描述：视频输出制式：标清———SECAM-K。

E.2.2.3.21 VOUT_STANDARD_SMPTE274_1080I_50

原型：public static final int VOUT_STANDARD_SMPTE274_1080I_50 = 27400
描述：视频输出制式：高清———SMPTE274/1920x1080I/50HZ/1125 行。

E.2.2.3.22 VOUT_STANDARD_SMPTE274_1080I_59_94

原型：public static final int VOUT_STANDARD_SMPTE274_1080I_59_94 = 27401
描述：视频输出制式：高清———SMPTE274/1920x1080I/59.94HZ/1125 行。

E.2.2.3.23 VOUT_STANDARD_SMPTE274_1080I_60

原型：public static final int VOUT_STANDARD_SMPTE274_1080I_60 = 27402
描述：视频输出制式：高清———SMPTE274/1920x1080I/60HZ/1125 行。

E.2.2.3.24 VOUT_STANDARD_SMPTE274_1080P_23_98

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_23_98 = 27410
描述：视频输出制式：高清———SMPTE274/1920x1080P/23.98HZ/1125 行。

E.2.2.3.25 VOUT_STANDARD_SMPTE274_1080P_24

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_24 = 27411
描述：视频输出制式：高清———SMPTE274/1920x1080P/24HZ/1125 行。

E.2.2.3.26 VOUT_STANDARD_SMPTE274_1080P_25

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_25 = 27412
描述：视频输出制式：高清———SMPTE274/1920x1080P/25HZ/1125 行。

E.2.2.3.27 VOUT_STANDARD_SMPTE274_1080P_29_97

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_29_97 = 27413
描述：视频输出制式：高清———SMPTE274/1920x1080P/29.97HZ/1125 行。

E.2.2.3.28 VOUT_STANDARD_SMPTE274_1080P_30

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_30 = 27414
描述：视频输出制式：高清——SMPTE274/1920x1080P/30HZ/1125 行。

E.2.2.3.29 VOUT_STANDARD_SMPTE274_1080P_50

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_50 = 27415
描述：视频输出制式：高清——SMPTE274/1920x1080P/50HZ/1125 行。

E.2.2.3.30 VOUT_STANDARD_SMPTE274_1080P_59_94

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_59_94 = 27416
描述：视频输出制式：高清——SMPTE274/1920x1080P/59.94HZ/1125 行。

E.2.2.3.31 VOUT_STANDARD_SMPTE274_1080P_60

原型：public static final int VOUT_STANDARD_SMPTE274_1080P_60 = 27417
描述：视频输出制式：高清——SMPTE274/1920x1080P/60HZ/1125 行。

E.2.2.3.32 VOUT_STANDARD_SMPTE295_1080I_50

原型：public static final int VOUT_STANDARD_SMPTE295_1080I_50 = 29500
描述：视频输出制式：高清——SMPTE295/1920x1080I/50HZ/1250 行。

E.2.2.3.33 VOUT_STANDARD_SMPTE295_1080P_50

原型：public static final int VOUT_STANDARD_SMPTE295_1080P_50 = 29510
描述：视频输出制式：高清——SMPTE295/1920x1080P/50HZ/1250 行。

E.2.2.3.34 VOUT_STANDARD_SMPTE296_720P_23_98

原型：public static final int VOUT_STANDARD_SMPTE296_720P_23_98 = 29610
描述：视频输出制式：高清——SMPTE296/1280x720P/23.98HZ/750 行。

E.2.2.3.35 VOUT_STANDARD_SMPTE296_720P_24

原型：public static final int VOUT_STANDARD_SMPTE296_720P_24 = 29611
描述：视频输出制式：高清——SMPTE296/1280x720P/24HZ/750 行。

E.2.2.3.36 VOUT_STANDARD_SMPTE296_720P_25

原型：public static final int VOUT_STANDARD_SMPTE296_720P_25 = 29612
描述：视频输出制式：高清——SMPTE296/1280x720P/25HZ/750 行。

E.2.2.3.37 VOUT_STANDARD_SMPTE296_720P_29_97

原型：public static final int VOUT_STANDARD_SMPTE296_720P_29_97 = 29613
描述：视频输出制式：高清——SMPTE296/1280x720P/29.97HZ/750 行。

E.2.2.3.38 VOUT_STANDARD_SMPTE296_720P_30

原型：public static final int VOUT_STANDARD_SMPTE296_720P_30 = 29614
描述：视频输出制式：高清——SMPTE296/1280x720P/30HZ/750 行。

E.2.2.3.39 VOUT_STANDARD_SMPTE296_720P_50

原型：public static final int VOUT_STANDARD_SMPTE296_720P_50 = 29615
描述：视频输出制式：高清——SMPTE296/1280x720P/50HZ/750 行。

E.2.2.3.40 VOUT_STANDARD_SMPTE296_720P_59_94

原型：public static final int VOUT_STANDARD_SMPTE296_720P_59_94 = 29616
描述：视频输出制式：高清——SMPTE296/1280x720P/59.94HZ/750 行。

E.2.2.3.41 VOUT_STANDARD_SMPTE296_720P_60

原型：public static final int VOUT_STANDARD_SMPTE296_720P_60 = 29617
描述：视频输出制式：高清——SMPTE296/1280x720P/60HZ/750 行。

E.2.2.3.42 VOUT_STANDARD_2160P_24

原型：public static final int VOUT_STANDARD_2160P_24 = 29618
描述：视频输出制式：4K 高清——/3840x2160P/24。

E.2.2.3.43 VOUT_STANDARD_2160P_25

原型：public static final int VOUT_STANDARD_2160P_25 = 29619
描述：视频输出制式：4K 高清——/3840x2160P/25。

E.2.2.3.44 VOUT_STANDARD_2160P_30

原型：public static final int VOUT_STANDARD_2160P_30 = 29620
描述：视频输出制式：4K 高清——/3840x2160P/30。

E.2.2.3.45 VOUT_STANDARD_2160P_50

原型：public static final int VOUT_STANDARD_2160P_24 = 29621
描述：视频输出制式：4K 高清——/3840x2160P/50。

E.2.2.3.46 VOUT_STANDARD_2160P_60

原型：public static final int VOUT_STANDARD_2160P_60 = 29622
描述：视频输出制式：4K 高清——/3840x2160P/60。

E.2.2.3.47 VOUT_STANDARD_4096P_24

原型：public static final int VOUT_STANDARD_4096P_24 = 29623
描述：视频输出制式：4K 高清——/3840x2160P/24。

E.2.2.4 方法

E.2.2.4.1 getOutputInterfaceList

原型：public static java.lang.String[] getOutputInterfaceList()
描述：获取接收终端所有可用的视频输出端口列表。
参数：无。
返回：java.lang.String 对象数组，表示接收终端所有可用的视频输出接口名称，例如“CVBS”、

“YUV”、“HDMI-0”、“HDMI-1”、“DVO”等。若无视频输出端口，则返回的数组长度为0。

E.2.2.4.2 getOutputInterfaceStatus

原型：public static boolean getOutputInterfaceStatus(java.lang.String port)

描述：获取视频输出端口的使能状态。

参数：port - java.lang.String 对象，表示视频输出端口的名称。

注：视频输出端口的名称由 getOutputInterfaceList()方法获得。

返回：boolean 型，表示视频输出端口的使能状态，取值 true 表示该视频输出接口允许输出，取值 false 表示该视频输出接口禁止输出。

E.2.2.4.3 disableOutputInterface

原型：public static boolean disableOutputInterface(java.lang.String port)

描述：禁止视频输出端口输出。

参数：port - java.lang.String 对象，表示视频输出端口的名称。

注：视频输出端口的名称由 getOutputInterfaceList()方法获得。

返回：boolean 型，取值 true 表示禁止成功，false 表示禁止失败。

E.2.2.4.4 enableOutputInterface

原型：public static void enableOutputInterface(java.lang.String port)

描述：允许视频输出端口输出。

参数：port - java.lang.String 对象，表示视频输出端口的名称。

注：视频输出端口的名称由 getOutputInterfaceList()方法获得。

返回：boolean 型，取值 true 表示允许成功，false 表示允许失败。

E.2.2.4.5 getOutputMatchMethod

原型：public static int getOutputMatchMethod()

描述：获取视频与显示窗口的匹配模式。具体效果需要根据所播放节目的宽高比、匹配模式以及显示设备是否有自适应功能来确定。

参数：无。

返回：int 型，表示视频输出端口的匹配模式，取值详见 VideoSetting 类的“匹配模式”常量域定义。

E.2.2.4.6 getOutputBrightness

原型：public static int getOutputBrightness()

描述：获取视频输出的亮度。

参数：无。

返回：int 型，表示视频输出的亮度，取值范围从小到大为 0～100，0 表示最黑，100 表示最亮。

E.2.2.4.7 getOutputContrast

原型：public static int getOutputContrast()

描述：获取视频输出的对比度。

参数：无。

返回：int 型，表示视频输出的对比度，取值范围从小到大为 0～100。

E.2.2.4.8 getOutputSaturation

原型：public static int getOutputSaturation()
描述：获取视频输出的饱和度（色度）。
参数：无。
返回：int 型，表示视频输出的饱和度（色度），取值范围从小到大为 0～100。

E.2.2.4.9 getOutputStandard

原型：public static int getOutputStandard(int device)
描述：获取视频输出的制式。
参数：device－int型，表示视频输出单元，可取值为VOUT_SD或VOUT_HD，详见VideoSetting类的“输出通道”常量域定义。
返回：int型，表示视频输出的制式，取值详见VideoSetting类的“输出制式”常量域定义。

E.2.2.4.10 getOutputStandards

原型：public static int[] getOutputStandards(int device)
描述：获取视频输出的所有支持制式。
参数：device－int型，表示视频输出单元，可取值为VOUT_SD或VOUT_HD，详见VideoSetting类的“输出通道”常量域定义。
返回：int型数组，表示视频输出的所有支持制式，取值详见org.ngb.util.setting.VideoSetting类的“输出制式”常量域定义。

E.2.2.4.11 getOutputTransparency

原型：public static int getOutputTransparency()
描述：获取视频输出的透明度。
参数：无。
返回：int 型，表示透明度，取值范围为 0～100，0 表示完全不透明，100 表示完全透明。

E.2.2.4.12 setOutputMatchMethod

原型：public static void setOutputMatchMethod(int matchMethod)
描述：设置屏幕的适应模式。具体效果需要根据所播放节目的宽高比、用户设置的屏幕适应模式，以及显示设备是否有自适应功能来确定。
参数：matchMethod － int 型，视频与显示窗口的匹配方式，取值详见org.ngb.util.setting.VideoSetting类的“匹配模式”常量域定义。
返回：无。

E.2.2.4.13 setOutputBrightness

原型：public static void setOutputBrightness(int value)
描述：设置视频输出的亮度。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value － int 型，表示视频输出的亮度，取值范围为 0～100， 0 表示最黑，100 表示最亮。
返回：无。

E.2.2.4.14 setOutputContrast

原型：public static void setOutContrast(int value)
描述：视频输出的对比度。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value - int 型，表示视频输出的对比度，取值范围为 0～100。
返回：无。

E.2.2.4.15 setOutputSaturation

原型：public static void setOutputSaturation(int value)
描述：设置视频输出的饱和度（色度）。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value - int 型，表示饱和度（色度），取值范围为 0～100。
返回：无。

E.2.2.4.16 setOutputStandard

原型：public static int setOutputStandard(int device, int standard)
throws unsupportedOperationException
描述：设置视频输出的制式。需要对标清输出单元和高清输出单元单独设置。
参数：device - int型，表示视频输出单元，取值VOUT_SD或VOUT_HD，详见org.ngb.util.setting.VideoSetting类的“输出通道”常量域定义；
standard - int型，表示视频输出制式，标清输出单元只能选标清的制式，高清输出单元只能选高清的制式，取值详见org.ngb.util.setting.VideoSetting类的“输出制式”常量域定义。
返回：int型，表示设置前的视频输出制式，取值详见org.ngb.util.setting.VideoSetting类的“输出制式”常量域定义。
异常：unsupportedOperationException – 若不支持此操作，则抛出此异常。

E.2.2.4.17 setOutputTransparency

原型：public static void setOutputTransparency(int value)
描述：设置视频输出的透明度。该方法无法针对单个视频输出单元进行设置，对所有视频输出单元都生效。
参数：value - int 型，表示透明度，取值范围为 0～100，0 表示完全不透明，100 表示完全透明。
返回：无。

E.2.2.4.18 getOutputAspectRatio

原型：public static int getOutputAspectRatio(int device)
描述：获取视频输出宽高比。
参数：device - int 型，表示视频输出单元，取值 VOUT_SD 或 VOUT_HD，详见org.ngb.util.setting.VideoSetting 类的“输出通道”常量域定义；
返回：int 型，0 表示 16:9；1 表示 4:3。

E.2.2.4.19 setOutputAspectRatio

原型：public static boolean setOutputAspectRatio(int device, int mode)
描述：设置视频输出的宽高比。
参数：device - int 型，表示视频输出单元，取值 VOUT_SD 或 VOUT_HD，详见org.ngb.util.setting.VideoSetting 类的“输出通道”常量域定义；

mode - int 型，0 表示 16:9；1 表示 4:3。

返回：boolean型，表示视频输出宽高比设置执行结果，true表示执行成功，取值false表示失败。

E.2.2.4.20 GetColorSpaceMode

原型：public static int GetColorSpaceMode()

描述：获取色彩空间模式。

参数：无。

返回： int 型，0-RGB444，1-YCBCR422， 2 -YCBCR444，3-YCBCR420。

E.2.2.4.21 GetDeepColorMode

原型：public static int GetDeepColorMode()

描述：获取色彩深度模式。

参数：无

返回： int 型，0-COLOR_24BIT，1- COLOR_30BIT， 2 -COLOR_36BIT，3-COLOR_DEEP_OFF。

E.2.2.4.22 SetColorSpaceAndDeepColor

原型：public static boolean SetColorSpaceAndDeepColor(int colorSpace, int deepColor)

描述：设置色彩空间，色深模式。

参数：colorSpace - int 型，0-RGB444，1-YCBCR422， 2 -YCBCR444，3-YCBCR420

deepColor - int 型，0-COLOR_24BIT，1-COLOR_30BIT，2-COLOR_36BIT，3-COLOR_DEEP_OFF

返回：boolean 型，true 表示执行成功，取值 false 表示失败。

E.2.2.4.23 GetHDRType

原型：public static int GetHDRType()

描述：获取 HDR 模式。

参数：无。

返回：int 型，0- HDRTYPE_SDR，1- HDRTYPE_DOLBY， 2 - HDRTYPE_HDR10，3-C HDRTYPE_AUTO。

E.2.2.4.24 SetHDRType

原型：public static boolean SetHDRType(int type)

描述：设置 HDR 模式。

参数：type - int 型，0- HDRTYPE_SDR，1- HDRTYPE_DOLBY， 2 - HDRTYPE_HDR10，3-C HDRTYPE_AUTO

返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

E.2.2.4.25 GetStereoOutMode

原型：public static int GetStereoOutMode()

描述：获取 3D 输出模式。

参数：无。

返回：int 型,0- 3D_NONE,1- 3D_FRAME_PACKING，2 - 3D_SIDE_BY_SIDE_HALF,3-3D_TOP_AND_BOTTOM，4-3D_FIELD_ALTERNATIVE,5-3D_LINE_ALTERNATIVE,6-3D_SIDE_BY_SIDE_FULL,7-3D_L_DEPTH，8-3D_L_DEPTH_GRAPHICS_GRAPHICS_DEPTH。

E.2.2.4.26 SetStereoOutMode

原型：public static boolean SetStereoOutMode(int mode,int fps)
描述：设置3D输出模式。
参数：mode－int 型，0- 3D_NONE，1- 3D_FRAME_PACKING，2 - 3D_SIDE_BY_SIDE_HALF，3-3D_TOP_AND_BOTTOM，4-3D_FIELD_ALTERNATIVE，5-3D_LINE_ALTERNATIVE，6-3D_SIDE_BY_SIDE_FULL，7-3D_L_DEPTH，8-3D_L_DEPTH_GRAPHICS_GRAPHICS_DEPTH；
fps- int 型，视频帧率，23，24，25，30，50，59，60；
返回：boolean型，取值true表示设置成功，false表示设置失败。

E.2.2.4.27 GetRightEyeFirst

原型：public static int GetRightEyeFirst()
描述：获取3D输出信号出哪眼画面先出。
参数：无。
返回：int型，0-LEFT_EYE_FIRST，1- RIGHT_EYE_FIRST。

E.2.2.4.28 SetRightEyeFirst

原型：public static boolean SetRightEyeFirst(int Outpriority)
描述：设置3D输出信号出哪眼画面先出。
参数：Outpriority－int型，0-LEFT_EYE_FIRST，1- RIGHT_EYE_FIRST。
返回：boolean型，取值true表示设置成功，false表示设置失败。

E.2.2.4.29 GetStereoDepth

原型：public static int GetStereoDepth()
描述：获取3D画面深度调节信息。
参数：无。
返回：int 型，0～10。

E.2.2.4.30 SetStereoDepth

原型：public static boolean SetStereoDepth(int depth)
描述：设置3D画面深度调节信息。
参数：depth－int型，0～10。
返回：boolean型，取值true表示设置成功，false表示设置失败。

E.2.2.4.31 getPictureMode

原型：public static int getPictureMode()
描述：获取画面模式。
参数：无。
返回：int型，0标准，1动态，2柔和，4用户，5艳丽，6自然，7运动。

E.2.2.4.32 setPictureMode

原型：public static boolean setPictureMode(int mode)
描述：设置画面模式。
参数：mode - int型，0标准，1动态，2柔和，4用户，5艳丽，6自然，7运动。
返回：boolean型，取值true表示设置成功，false表示设置失败.

E.2.2.4.33 getDisplayHue

原型：public static int getDisplayHue()
描述：获取画面模式。
参数：无。
返回：int 型，取值范围 0～100，表示色调调整值。

E.2.2.4.34 setDisplayHue

原型：public static boolean setDisplayHue(int hue)
描述：设置画面模式。
参数：hue - int 型，取值范围 0-100，表示色调调整值。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

E.2.2.4.35 SaveDisplayFmt

原型：public static boolean SaveDisplayFmt()
描述：保存视频输出格式永久生效。
参数：无。
返回：boolean型，取值true表示成功，false表示失败。

E.2.2.4.36 setOptimalFormatEnable

原型：public static boolean setOptimalFormatEnable(int enabled)
描述：设置自动优化视频输出格式使能。
参数：enabled --- int 型，0-表示不使能；1-表示使能。
返回：boolean型，取值true表示成功，false表示失败。

E.2.2.4.37 getOptimalFormatEnable

原型：public static int getOptimalFormatEnable()
描述：获取自动优化视频输出格式使能状态。
参数：无。
返回：int型，0-表示不使能；1-表示使能。

附 录 F
（规范性附录）
JAVA-媒体处理单元

F.1 概述

本附录定义了与媒体播放相关的JAVA接口。

F.2 媒体处理模块

媒体处理模块定义了提供所有媒体功能的接口类以及相关的音、视频信息类。媒体处理模块用于支持包括 DVB 直播、VOD 点播、IP 直播、IP 点播以及本地播放等媒体播放功能。

媒体处理模块定义的最基本的类有：

——媒体播放器类（MediaPlayer）；

——音频、视频、字幕等轨道类（TrackInfo）；

——音视频流信息类（MediaFormat）。

媒体处理模块概要见表 F.1。

表 F.1 媒体处理模块概要

类	
MediaPlayer	提供所有媒体功能的接口。支持包括 DVB 直播、VOD 点播、IP 直播、IP 点播以及本地播放场景。
TrackInfo	描述音频、视频、字幕等轨道信息。
MediaFormat	使用 HashMap 存放音视频流等信息。

F.2.1 类org.tvos.media.MediaPlayer

原型：public class org.tvos.media.MediaPlayer

描述：直接承担了所有功能，提供所有媒体功能的接口。支持包括DVB直播、VOD点播、IP直播、IP点播和本地播放功能。

F.2.1.1 常量域——错误原因

F.2.1.1.1 MEDIA_ERROR_UNKNOWN

原型：public static final int MEDIA_ERROR_UNKNOWN = 1

描述：错误原因——未知。

F.2.1.1.2 MEDIA_ERROR_SERVER_DIED

原型：public static final int MEDIA_ERROR_SERVER_DIED = 100

描述：错误原因——媒体服务死亡。

F.2.1.1.3 MEDIA_ERROR_NOT_VALID_FOR_PROGRESSIVE_PLAYBACK

原型：public static final int MEDIA_ERROR_NOT_VALID_FOR_PROGRESSIVE_PLAYBACK = 200
描述：错误原因——渐进回放不合法。

F.2.1.1.4 TVOS_MEDIA_ERROR_START_FAILD

原型：TVOS_MEDIA_ERROR_START_FAILD = 1000
描述：错误原因——播放失败。

F.2.1.1.5 TVOS_MEDIA_ERROR_SETPACE_FAILED

原型：public static final int TVOS_MEDIA_ERROR_SETPACE_FAILED = 1001
描述：错误原因——设置设置倍速播放失败。

F.2.1.1.6 TVOS_MEDIA_ERROR_SEEK_FAILD

原型：public static final int TVOS_MEDIA_ERROR_SEEK_FAILD = 1002
描述：错误原因——SEEK 失败。

F.2.1.1.7 TVOS_MEDIA_ERROR_PAUSE_FAILD

原型：public static final int TVOS_MEDIA_ERROR_PAUSE_FAILD = 1003
描述：错误原因——暂停失败。

F.2.1.1.8 TVOS_MEDIA_ERROR_RESUME_FAILD

原型：public static final int TVOS_MEDIA_ERROR_RESUME_FAILD = 1004
描述：错误原因——恢复失败。

F.2.1.1.9 TVOS_MEDIA_ERROR_STOP_FAILD

原型：public static final int TVOS_MEDIA_ERROR_STOP_FAILD = 1005
描述：错误原因——停止失败。
描述：错误原因——URL 有效。

F.2.1.1.10 TVOS_MEDIA_ERROR_URL_INVALID

原型：public static final int TVOS_MEDIA_ERROR_URL_INVALID = 1006
描述：错误原因——URL 无效。

F.2.1.1.11 TVOS_MEDIA_ERROR_RESOURCE_UNAVAILABLE

原型：public static final int TVOS_MEDIA_ERROR_RESOURCE_UNAVAILABLE = 1007
描述：错误原因——播放资源不可用。

F.2.1.1.12 TVOS_MEDIA_ERROR_AUDIO_DECODE_ERROR

原型：public static final int TVOS_MEDIA_ERROR_AUDIO_DECODE_ERROR = 1008
描述：错误原因——音频解码失败。

F.2.1.1.13 TVOS_MEDIA_ERROR_VIDEO_DECODE_ERROR

原型：public static final int TVOS_MEDIA_ERROR_VIDEO_DECODE_ERROR = 1009
描述：错误原因——视频解码失败。

F.2.1.1.14 TVOS_MEDIA_ERROR_UNSUPPORT_VIDEO_DEC

原型：public static final int TVOS_MEDIA_ERROR_UNSUPPORT_VIDEO_DEC = 1010
描述：错误原因——不支持视频格式解码

F.2.1.1.15 TVOS_MEDIA_ERROR_UNSUPPORT_AUDIO_DEC

原型：public static final int TVOS_MEDIA_ERROR_UNSUPPORT_AUDIO_DEC = 1011
描述：错误原因——不支持音频格式解码。

F.2.1.1.16 TVOS_MEDIA_ERROR_CONNECT_FAILED

原型：public static final int TVOS_MEDIA_ERROR_CONNECT_FAILED = 1012
描述：错误原因——连接服务器失败,建立会话失败或者服务器返回超时。

F.2.1.1.17 TVOS_MEDIA_ERROR_VOD_SEARCH_FAILED

原型：public static final int TVOS_MEDIA_ERROR_VOD_SEARCH_FAILED = 1300
描述：错误原因——IPQAM 播放时搜索数据失败。

F.2.1.1.18 TVOS_MEDIA_ERROR_VOD_OUT_OF_RANGE

原型：public static final int TVOS_MEDIA_ERROR_VOD_OUT_OF_RANGE = 1023
描述：错误原因——传入的时间超出有效范围。

F.2.1.2 常量域——媒体信息

F.2.1.2.1 MEDIA_INFO_UNKNOWN

原型：public static final int MEDIA_INFO_UNKNOWN = 1
描述：媒体信息——未知。

F.2.1.2.2 MEDIA_INFO_STARTED_AS_NEXT

原型：public static final int MEDIA_INFO_STARTED_AS_NEXT = 2
描述：媒体信息——下一个开始播放。

F.2.1.2.3 MEDIA_INFO_RENDERING_START

原型：public static final int MEDIA_INFO_RENDERING_START = 3
描述：媒体信息——首帧开始渲染。

F.2.1.2.4 MEDIA_INFO_VIDEO_TRACK_LAGGING

原型：public static final int MEDIA_INFO_VIDEO_TRACK_LAGGING = 700
描述：媒体信息——视频编码复杂，解码性能不足。

F.2.1.2.5 MEDIA_INFO_BUFFERING_START

原型：public static final int MEDIA_INFO_BUFFERING_START = 701
描述：媒体信息——媒体开始缓冲。

F.2.1.2.6 MEDIA_INFO_BUFFERING_END

原型：public static final int MEDIA_INFO_BUFFERING_END = 702
描述：媒体信息——媒体缓冲结束。

F.2.1.2.7 MEDIA_INFO_NETWORK_BANDWIDTH

原型：public static final int MEDIA_INFO_NETWORK_BANDWIDTH = 703
描述：媒体信息——网络带宽。

F.2.1.2.8 MEDIA_INFO_BAD_INTERLEAVING

原型：public static final int MEDIA_INFO_BAD_INTERLEAVING = 800
描述：媒体信息——媒体播放卡顿。

F.2.1.2.9 MEDIA_INFO_NOT_SEEKABLE

原型：public static final int MEDIA_INFO_NOT_SEEKABLE = 801
描述：媒体信息——媒体播放不支持 SEEK。

F.2.1.2.10 MEDIA_INFO_METADATA_UPDATE

原型：public static final int MEDIA_INFO_METADATA_UPDATE = 802
描述：媒体信息——媒体元信息更新。

F.2.1.2.11 MEDIA_INFO_EXTERNAL_METADATA_UPDATE

原型：public static final int MEDIA_INFO_EXTERNAL_METADATA_UPDATE = 803
描述：媒体信息——外部的元数据更新。

F.2.1.2.12 MEDIA_INFO_TIMED_TEXT_ERROR

原型：public static final int MEDIA_INFO_TIMED_TEXT_ERROR = 900
描述：媒体信息——同步字幕错误。

F.2.1.2.13 MEDIA_INFO_UNSUPPORTED_SUBTITLE

原型：public static final int MEDIA_INFO_UNSUPPORTED_SUBTITLE = 901
描述：媒体信息——不支持的字幕。

F.2.1.2.14 MEDIA_INFO_SUBTITLE_TIMED_OUT

原型：public static final int MEDIA_INFO_SUBTITLE_TIMED_OUT = 902
描述：媒体信息——字幕超时。

F.2.1.2.15 TVOS_MEDIA_INFO_START_SUCCESS

原型：public static final int TVOS_MEDIA_INFO_START_SUCCESS = 1000
描述：媒体信息——播放成功。

F.2.1.2.16 TVOS_MEDIA_INFO_TUNE_LOCK_SUCCES

原型：public static final int TVOS_MEDIA_INFO_TUNE_LOCK_SUCCES = 1001
描述：媒体信息——Tuner 锁频成功或者信号恢复。

F.2.1.2.17 TVOS_MEDIA_INFO_TUNE_LOCK_LOST

原型：public static final int TVOS_MEDIA_INFO_TUNE_LOCK_LOST = 1002
描述：媒体信息——Tuner 信号丢失。

F.2.1.2.18 TVOS_MEDIA_INFO_NO_STREAM

原型：public static final int TVOS_MEDIA_INFO_NO_STREAM = 1003
描述：媒体信息——没有流信息（或者突然断流）。

F.2.1.2.19 TVOS_MEDIA_INFO_STREAM_RECOVER

原型：public static final int TVOS_MEDIA_INFO_STREAM_RECOVER = 1004
描述：媒体信息——流恢复。

F.2.1.2.20 TVOS_MEDIA_INFO_AUDIO_DECODE_SUCCESS

原型：public static final int TVOS_MEDIA_INFO_AUDIO_DECODE_SUCCESS = 1005
描述：媒体信息——音频解码成功。

F.2.1.2.21 TVOS_MEDIA_INFO_VIDEO_DECODE_SUCCESS

原型：public static final int TVOS_MEDIA_INFO_VIDEO_DECODE_SUCCESS = 1006
描述：媒体信息——视频解码成功。

F.2.1.2.22 TVOS_MEDIA_INFO_PACE_CHANGE

原型：public static final int TVOS_MEDIA_INFO_PACE_CHANGE = 1007
描述：媒体信息——速度变化。

F.2.1.2.23 TVOS_MEDIA_INFO_DVB_CA_READY

原型：public static final int TVOS_MEDIA_INFO_DVB_CA_READY = 1100
描述：媒体信息——DVB CA 就绪。

F.2.1.2.24 TVOS_MEDIA_INFO_DVB_CA_NOT_READY

原型：public static final int TVOS_MEDIA_INFO_DVB_CA_NOT_READY = 1101
描述：媒体信息——DVB CA 未就绪。

F.2.1.2.25 TVOS_MEDIA_INFO_VOD_END_OF_STREAM

原型：public static final int TVOS_MEDIA_INFO_VOD_END_OF_STREAM = 1300
描述：媒体信息——时移频道或VOD影片播放到了点播结束的位置。

F.2.1.2.26 TVOS_MEDIA_INFO_VOD_BEGIN_OF_STREAM

原型：public static final int TVOS_MEDIA_INFO_VOD_BEGIN_OF_STREAM = 1301
描述：媒体信息——时移频道或VOD影片播放到了点播开始的位置。

F.2.1.3 方法

F.2.1.3.1 MediaPlayer

原型： public MediaPlayer()

描述：构造方法。
参数：无。

F.2.1.3.2 setDataSource

原型：void setDataSource(java.lang.String path)
描述：设置资源的路径。DTV 播放的 path 格式包含三种方式，匹配三种场景，约束如下。
场景一：App 搜索节目后，通过节目三元素播放，其格式为 dvb://onid.tsid.sid，针对多 Tuner 场景扩展为 dvb://onid.tsid.sid?tunerid=xxx。
场景二：App 不用搜索节目，通过指定频点等节目信息播放，以满足快速启播，其格式为 dvbelement://frequency.symbolrate.modulation.serviceid.pmtpid.pcrpid.videotype.videopid.audiotype.audiopid。其中 frequency 单位为千赫兹，所有参数为十进制，如 dvbelement://131000.6875.64.3.4.1.2.3.1.2 。针对多 Tuner 场景扩展为 dvbelement://frequency.symbolrate.modulation.serviceid.pmtpid.pcrpid.videotype.videopid.audiotype.audiopid?tunerid=xxx。
场景三：App 不用搜索节目，通过指定频点等节目信息播放，以满足不搜索节目时加扰流、多音轨播放，其格式为 dvbelement://frequency.symbolrate.modulation.serviceid.-1.-1.-1.-1.-1.-1。此场景媒体引擎根据 serviceid 从码流中解析 pmtpid，根据 pmtpid 解析 pcrpid、videotype、videopid、audiotype 和 audiopid ，若 pmtpid 存在且有效，则不搜索 pmtpid，根据 pmtpid 解析 pcrpid、videotype、videopid、audiotype 和 audiopid。其中 frequency 单位为千赫兹，所有数字为十进制，无效参数为-1，如 dvbelement://131000.6875.64.3.-1.-1.-1.-1.-1.-1 或 dvbelement://131000.6875.64.3.4.-1.-1.-1.-1.-1。针对多 Tuner 场景扩展为 dvbelement://frequency.symbolrate.modulation.serviceid.-1.-1.-1.-1.-1.-1?tunerid=xxx。
参数：path – java.lang.String对象，表示资源路径。
返回：无。

F.2.1.3.3 prepareAsync

原型：void prepareAsync()
描述：准备播放器，异步。
参数：无。
返回：无。

F.2.1.3.4 prepare

原型：void prepare()
描述：准备播放器进行播放，同步。
参数：无。
返回：无。

F.2.1.3.5 start

原型：void start()
描述：启动或恢复播放。
参数：无。

返回：无。

F.2.1.3.6 pause

原型：void pause()
描述：暂停播放。
参数：无。
返回：无。

F.2.1.3.7 getDuration

原型：int getDuration()
描述：获取文件的持续时间。
参数：无。
返回：int 型，表示文件的持续时间。

F.2.1.3.8 getCurrentPosition

原型：int getCurrentPosition()
描述：获取当前播放时间。
参数：无。
返回：int 型，表示当前播放位置。

F.2.1.3.9 setVolume

原型：void setVolume(float volume)
描述：设置此播放器的音量。
参数：volume – float 型，表示音量的数值。
返回：无。

F.2.1.3.10 seekTo

原型：void seekTo(int pos)
描述：Seek 到指定位置。
参数：pos – int 型，表示要 seek 到的位置。
返回：无。

F.2.1.3.11 reset

原型：void reset()
描述：重置到播放器的初始化状态。
参数：无。
返回：无。

F.2.1.3.12 release

原型：void release()
描述：释放播放器关联的资源。
参数：无。
返回：无。

F.2.1.3.13 setOnInfoListener

原型：void setOnInfoListener(org.tvos.media.MediaPlayer.OnInfoListener listener)
描述：注册一个回调，当信息/警告可用时调用。。
参数：listener – org.tvos.media.MediaPlayer.OnInfoListener ,表示信息监听器。
返回：无。

F.2.1.3.14 setOnErrorListener

原型：void setOnErrorListener(org.tvos.media.MediaPlayer.OnErrorListener listener)
描述：注册在异步操作期间发生错误时调用的回调函数。
参数：listener – org.tvos.media.MediaPlayer.OnErrorListener,表示错误监听器。
返回：无。

F.2.1.3.15 setOnCompletionListener

原型：void setOnCompletionListener(org.tvos.media.MediaPlayer.OnCompletionListener listener)
描述：注册一个回调函数，当媒体源播放结束时被调用。。
参数：listener – org.tvos.media.MediaPlayer.OnCompletionListener,表示播放结束监听器。
返回：无。

F.2.1.3.16 setOnPreparedListener

原型：void setOnPreparedListener(org.tvos.media.MediaPlayer.OnPreparedListener listener)
描述：注册媒体源准备播放时调用的回调函数。
参数：listener – org.tvos.media.MediaPlayer.OnPreparedListener,表示媒体源准备播放的监听器。
返回：无。

F.2.1.3.17 setOnBufferingUpdateListener

原型：public void setOnBufferingUpdateListener(org.tvos.media.MediaPlayer.OnBufferingUpdateListener listener)
描述：注册一个回调函数，当网络流缓冲区的状态发生变化时调用它。
参数：listener – org.tvos.media.MediaPlayer.OnBufferingUpdateListener,表示缓冲区的状态更新监听器。
返回：无。

F.2.1.3.18 setDisplay

原型：void setDisplay(org.tvos.view.SurfaceHolder sh)
描述：设置媒体视频播放的 SurfaceHolder。
参数：sh– org.tvos.view.SurfaceHolder,表示需要设置的 SurfaceHolder。
返回：无。

F.2.1.3.19 setAudioStreamType

原型：void setAudioStreamType(int streamtype)

描述：设置音频流类型。
参数：streamtype – int 型,表示音频流类型。
返回：无。

F.2.1.3.20 setStopMode

原型：boolean setStopMode(int mode)
描述：设置静帧模式。设置静帧模式:mode 为 0 代表黑屏，为 1 代表静帧。设置模式可以只调用 1 次，后续切台均保持该模式
参数：mode – int 型,表示播放器的静帧模式。
返回：boolean 型，表示是否设置成功，成功返回 true，失败返回 false。

F.2.1.3.21 getStopMode

原型：int getStopMode()
描述：获取播放器的静帧模式。:mode 为 0 代表黑屏，为 1 代表静帧。
参数：无。
返回：int 型，表示播放器的停止模式。

F.2.1.3.22 getStartTime

原型：long getStartTime()
描述：增加获取时移（或者回看）节目的起始时间。
参数：无。
返回：long 型，表示时移（或者回看）节目的起始时间。

F.2.1.3.23 getPace

原型：int getPace()
描述：获取快进或快退的速度。
参数：无。
返回：int 型，表示快进或快退的速度。

F.2.1.3.24 setPace

原型：void setPace(int pace)
描述：设置快进或快退的速度。
参数：pace – int 型,表示快进或快退的速度。
返回：无。

F.2.1.3.25 setClip

原型：void setClip(org.tvos.graphics.Rect rect)
描述：设置剪切区域。
参数：rect–Rect,表示剪切区域。
返回：无。

F.2.1.3.26 getClip

原型： org.tvos.graphics.Rect getClip()

描述：获取剪切区域。
参数：无。
返回： org.tvos.graphics.Rect 型,表示剪切区域。

F.2.1.3.27 clearVideo

原型：int clearVideo()
描述：断流清屏。使用场景介绍：头端流掐断后，引擎会检测到断流，会上报断流消息--TVOS_MEDIA_INFO_NO_STREAM，app 应用收到消息后，需要调该接口把屏幕清成黑屏。
参数：无。
返回：int 型，清屏成功返回 0，失败返回-1。

F.2.1.3.28 setVideoDisplay

原型：boolean setVideoDisplay(int mode)
描述：设置视频输出
参数：mode – int 型, 表示视频显示是关闭、还是打开，取值如下。
TVOS_VIDEO_DISPLAY_CLOSE = 0：关闭;
TVOS_VIDEO_DISPLAY_OPEN = 1：打开。
返回：boolean 型，表示是否设置成功，成功返回 true，失败返回 false。

F.2.1.3.29 getVideoDisplay

原型：int getVideoDisplay()
描述：获取视频输出模式。
参数：无。
返回：int 型，表示视频显示是关闭、还是打开，取值如下。
TVOS_VIDEO_DISPLAY_CLOSE = 0：关闭;
TVOS_VIDEO_DISPLAY_OPEN = 1：打开。

F.2.2 类org.tvos.media.TrackInfo

原型：public class TrackInfo
描述：描述音频、视频、字幕等轨道信息。

F.2.2.1 常量域——媒体类型

F.2.2.1.1 MEDIA_TRACK_TYPE_UNKNOWN

原型：public static final int MEDIA_TRACK_TYPE_UNKNOWN = 0
描述：媒体类型——未知类型。

F.2.2.1.2 MEDIA_TRACK_TYPE_VIDEO

原型：public static final int MEDIA_TRACK_TYPE_VIDEO = 1
描述：媒体类型——视频类型。

F.2.2.1.3 MEDIA_TRACK_TYPE_AUDIO

原型：public static final int MEDIA_TRACK_TYPE_AUDIO = 2
描述：媒体类型——音频类型。

F.2.2.1.4 MEDIA_TRACK_TYPE_TIMEDTEXT

原型：public static final int MEDIA_TRACK_TYPE_TIMEDTEXT = 3
描述：媒体类型——定时文本类型。

F.2.2.1.5 MEDIA_TRACK_TYPE_SUBTITLE

原型：public static final int MEDIA_TRACK_TYPE_SUBTITLE = 4
描述：媒体类型——字幕/标题类型。

F.2.2.2 方法

F.2.2.2.1 getTrackInfo

原型：org.tvos.media.TrackInfo[] getTrackInfo()
描述：获取当前正在播放的节目所有可操作的音频流信息
1，此处返回所有track信息，包括音频/视频/字幕/同步字幕等
2，设置音轨需从该track中获取音轨部分
参数：无。
返回：org.tvos.media.TrackInfo数组，表示track信息数组。

F.2.2.2.2 selectTrack

原型：void selectTrack(int index)
描述：选择一个track。
参数：index - int型,表示track的索引。
返回：无。

F.2.2.2.3 getSelectedTrack

原型：int getSelectedTrack(int trackType)
描述：获取当前选择用于回放的音频、视频或字幕磁道的索引。
参数：trackType - int型,表示track的类型，详情取值见类TrackInfo的常量域。
返回：int型，表示当前选择用于回放的音频、视频或字幕磁道的索引。

F.2.3 类org.tvos.media.MediaFormat

原型：public class MediaFormat
描述：使用HashMap存放音视频流等信息。

F.2.3.1 常量域——类型描述

F.2.3.1.1 MIMETYPE_VIDEO_VP8

原型：public static final java.lang.String MIMETYPE_VIDEO_VP8 = "video/x-vnd.on2.vp8"
描述：类型描述——表示MIME视频类型VP8。

F.2.3.1.2 MIMETYPE_VIDEO_VP9

原型：public static final java.lang.String MIMETYPE_VIDEO_VP9 = "video/x-vnd.on2.vp9"
描述：类型描述——表示MIME视频类型VP9。

F.2.3.1.3 MIMETYPE_VIDEO_AVC

原型：public static final java.lang.String MIMETYPE_VIDEO_AVC = "video/avc"
描述：类型描述——表示MIME视频类型AVC。

F.2.3.1.4 MIMETYPE_VIDEO_HEVC

原型：public static final java.lang.String MIMETYPE_VIDEO_HEVC = "video/hevc"
描述：类型描述——表示MIME视频类型HEVC。

F.2.3.1.5 MIMETYPE_VIDEO_MPEG4

原型：public static final java.lang.String MIMETYPE_VIDEO_MPEG4 = "video/mp4v-es"
描述：类型描述——表示MIME视频类型MPEG4。

F.2.3.1.6 MIMETYPE_VIDEO_H263

原型：public static final java.lang.String MIMETYPE_VIDEO_H263 = "video/3gpp"
描述：类型描述——表示MIME视频类型H263。

F.2.3.1.7 MIMETYPE_VIDEO_MPEG2

原型：public static final java.lang.String MIMETYPE_VIDEO_MPEG2 = "video/mpeg2"
描述：类型描述——表示MIME视频类型MPEG2。

F.2.3.1.8 MIMETYPE_VIDEO_RAW

原型：public static final java.lang.String MIMETYPE_VIDEO_RAW = "video/raw"
描述：类型描述——表示MIME视频类型RAW。

F.2.3.1.9 MIMETYPE_AUDIO_AMR_NB

原型：public static final java.lang.String MIMETYPE_AUDIO_AMR_NB = "audio/3gpp"
描述：类型描述——表示MIME音频类型AMR_NB。

F.2.3.1.10 MIMETYPE_AUDIO_AMR_WB

原型：public static final java.lang.String MIMETYPE_AUDIO_AMR_WB = "audio/amr-wb"
描述：类型描述——表示MIME音频类型AMR_WB。

F.2.3.1.11 MIMETYPE_AUDIO_MPEG

原型：public static final java.lang.String MIMETYPE_AUDIO_MPEG = "audio/mpeg"
描述：类型描述——表示MIME音频类型MPEG。

F.2.3.1.12 MIMETYPE_AUDIO_AAC

原型：public static final java.lang.String MIMETYPE_AUDIO_AAC = "audio/mp4a-latm"
描述：类型描述——表示MIME音频类型AAC。

F.2.3.1.13 MIMETYPE_AUDIO_QCELP

原型：public static final java.lang.String MIMETYPE_AUDIO_QCELP = "audio/qcelp"
描述：类型描述——表示MIME音频类型QCELP。

F.2.3.1.14 MIMETYPE_AUDIO_VORBIS

原型：public static final java.lang.String MIMETYPE_AUDIO_VORBIS = "audio/vorbis"
描述：类型描述——表示MIME音频类型VORBIS。

F.2.3.1.15 MIMETYPE_AUDIO_OPUS

原型：public static final java.lang.String MIMETYPE_AUDIO_OPUS = "audio/opus"
描述：类型描述——表示MIME音频类型OPUS。

F.2.3.1.16 MIMETYPE_AUDIO_G711_ALAW

原型：public static final java.lang.String MIMETYPE_AUDIO_G711_ALAW = "audio/g711-alaw"
描述：类型描述——表示MIME音频类型G711_ALAW。

F.2.3.1.17 MIMETYPE_AUDIO_G711_MLAW

原型：public static final java.lang.String MIMETYPE_AUDIO_G711_MLAW = "audio/g711-mlaw"
描述：类型描述——表示MIME音频类型G711_MLAW。

F.2.3.1.18 MIMETYPE_AUDIO_RAW

原型：public static final java.lang.String MIMETYPE_AUDIO_RAW = "audio/raw"
描述：类型描述——表示MIME音频类型。

F.2.3.1.19 MIMETYPE_AUDIO_FLAC

原型：public static final java.lang.String MIMETYPE_AUDIO_FLAC = "audio/flac"
描述：类型描述——表示MIME音频类型FLAC。

F.2.3.1.20 MIMETYPE_AUDIO_MSGSM

原型：public static final java.lang.String MIMETYPE_AUDIO_MSGSM = "audio/gsm"
描述：类型描述——表示MIME音频类型MSGSM。

F.2.3.1.21 MIMETYPE_AUDIO_AC3

原型：public static final java.lang.String MIMETYPE_AUDIO_AC3 = "audio/ac3"
描述：类型描述——表示MIME音频类型AC3。

F.2.3.1.22 MIMETYPE_TEXT_VTT

原型：public static final java.lang.String MIMETYPE_TEXT_VTT = "text/vtt"
描述：类型描述——表示MIME文本类型VTT。

F.2.3.1.23 MIMETYPE_TEXT_CEA_608

原型：public static final java.lang.String MIMETYPE_TEXT_CEA_608 = "text/cea-608"
描述：类型描述——表示MIME文本类型CEA_608。

F.2.3.2 常量域——属性描述

F.2.3.2.1 KEY_MIME

原型：public static final java.lang.String KEY_MIME = "mime"
描述：属性描述——表示格式的类型。

F.2.3.2.2 KEY_LANGUAGE

原型：public static final java.lang.String KEY_LANGUAGE = "language"
描述：属性描述——表示内容的语言。

F.2.3.2.3 KEY_SAMPLE_RATE

原型：public static final java.lang.String KEY_SAMPLE_RATE = "sample-rate"
描述：属性描述——表示音频格式的采样率。

F.2.3.2.4 KEY_CHANNEL_COUNT

原型：public static final java.lang.String KEY_CHANNEL_COUNT = "channel-count"
描述：属性描述——表示音频格式中的频道数。

F.2.3.2.5 KEY_WIDTH

原型：public static final java.lang.String KEY_WIDTH = "width"
描述：属性描述——表示视频格式中内容的宽度。

F.2.3.2.6 KEY_HEIGHT

原型：public static final java.lang.String KEY_HEIGHT = "height"
描述：属性描述——表示视频格式内容的高度。

F.2.3.2.7 KEY_MAX_WIDTH

原型：public static final java.lang.String KEY_MAX_WIDTH = "max-width"
描述：属性描述——表示视频解码器格式中的内容的最大期望宽度，以防视频内容中的分辨率发生变化。相关联的值是整数。

F.2.3.2.8 KEY_MAX_HEIGHT

原型：public static final java.lang.String KEY_MAX_HEIGHT = "max-height"
描述：属性描述——表示最大期望在视频内容高度解码格式，如有视频内容分辨率变化。关联的值是整数。

F.2.3.2.9 KEY_MAX_INPUT_SIZE

原型：public static final java.lang.String KEY_MAX_INPUT_SIZE = "max-input-size"
描述：属性描述——表示在一个缓冲区的数据描述的这mediaformat字节的最大大小。关联的值是整数。

F.2.3.2.10 KEY_BIT_RATE

原型：public static final java.lang.String KEY_BIT_RATE = "bitrate"
描述：属性描述——表示比特/秒的比特率。关联的值是整数。

F.2.3.2.11 KEY_STREAM_PID

原型：public static final java.lang.String KEY_STREAM_PID = "stream-pid"
描述：属性描述——表示音视频ES的PID。关联的值是整形。

F.2.3.2.12 KEY_COLOR_FORMAT

原型：public static final java.lang.String KEY_COLOR_FORMAT = "color-format"
描述：属性描述——表示在视频格式内容的颜色格式。

F.2.3.2.13 KEY_CAPTURE_RATE

原型：public static final java.lang.String KEY_CAPTURE_RATE = "capture-rate"
描述：属性描述——表示帧/秒中视频格式的捕获率。

F.2.3.2.14 KEY_I_FRAME_INTERVAL

原型：public static final java.lang.String KEY_I_FRAME_INTERVAL = "i-frame-interval"
描述：属性描述——表示数秒内I帧之间的I帧频率。

F.2.3.2.15 KEY_TEMPORAL_LAYERING

原型：public static final java.lang.String KEY_TEMPORAL_LAYERING = "ts-schema"
描述：属性描述——表示时态分层模式。

F.2.3.2.16 KEY_REPEAT_PREVIOUS_FRAME_AFTER

原型：public static final java.lang.String KEY_REPEAT_PREVIOUS_FRAME_AFTER = "repeat-previous-frame-after"
描述：属性描述——表示仅在“表面输入”模式下配置视频编码器时应用。关联的值是一个long类型的值，并给出了微秒的时间，在此之后，如果没有新帧可用，那么先前提交给编码器的帧将重复（一次）。

F.2.3.2.17 KEY_PUSH_BLANK_BUFFERS_ON_STOP

原型：public static final java.lang.String KEY_PUSH_BLANK_BUFFERS_ON_STOP = "push-blank-buffers-on-shutdown"
描述：属性描述——如果在将视频解码器配置到一个表面时指定，将使解码器输出“空白”，即当停止清除先前显示内容时的黑色帧到表面。相关的值是一个价值1的整数。

F.2.3.2.18 KEY_DURATION

原型：public static final java.lang.String KEY_DURATION = "durationUs"
描述：属性描述——表示持续时间（微秒）。

F.2.3.2.19 KEY_IS_ADTS

原型：public static final java.lang.String KEY_IS_ADTS = "is-adts"
描述：属性描述——可选，表示如果解码的是AAC音频内容，将此键设置为1表示每个音频帧都以ADTS头为前缀。

F.2.3.2.20 KEY_CHANNEL_MASK

原型：public static final java.lang.String KEY_CHANNEL_MASK = "channel-mask"

描述：属性描述——可选，表示音频通道分配的掩码。

F.2.3.2.21 KEY_AAC_PROFILE

原型：public static final java.lang.String KEY_AAC_PROFILE = "aac-profile"
描述：属性描述——可选，表示如果内容为AAC音频，则指定所需的配置文件。

F.2.3.2.22 KEY_AAC_SBR_MODE

原型：public static final java.lang.String KEY_AAC_SBR_MODE = "aac-sbr-mode"
描述：属性描述——可选，表示如果内容为AAC音频，则指定所需的SBR模式。

F.2.3.2.23 KEY_AAC_MAX_OUTPUT_CHANNEL_COUNT

原型：public static final java.lang.String KEY_AAC_MAX_OUTPUT_CHANNEL_COUNT =
"aac-max-output-channel_count"
描述：属性描述——可选，表示如果内容为AAC音频，则指定解码器输出的最大通道数。

F.2.3.2.24 KEY_AAC_DRC_TARGET_REFERENCE_LEVEL

原型：public static final java.lang.String KEY_AAC_DRC_TARGET_REFERENCE_LEVEL =
"aac-target-ref-level"
描述：属性描述——可选，表示如果内容为AAC音频，则指定目标参考电平。

F.2.3.2.25 KEY_AAC_ENCODED_TARGET_LEVEL

原型：public static final java.lang.String KEY_AAC_ENCODED_TARGET_LEVEL =
"aac-encoded-target-level"
描述：属性描述——可选，表示如果内容为AAC音频，则指定编码器使用的目标参考电平。

F.2.3.2.26 KEY_AAC_DRC_BOOST_FACTOR

原型：public static final java.lang.String KEY_AAC_DRC_BOOST_FACTOR = "aac-drc-boost-level"
描述：属性描述——可选，表示如果内容为AAC音频，则指定DRC增强因子。

F.2.3.2.27 KEY_AAC_DRC_ATTENUATION_FACTOR

原型：public static final java.lang.String KEY_AAC_DRC_ATTENUATION_FACTOR =
"aac-drc-cut-level"
描述：属性描述——可选，表示如果内容为AAC音频，则指定DRC衰减因子。

F.2.3.2.28 KEY_AAC_DRC_HEAVY_COMPRESSION

原型：public static final java.lang.String KEY_AAC_DRC_HEAVY_COMPRESSION =
"aac-drc-heavy-compression"
描述：属性描述——可选，表示如果内容为AAC音频，则指定是否使用较大的压缩。

F.2.3.2.29 KEY_FLAC_COMPRESSION_LEVEL

原型：public static final java.lang.String KEY_FLAC_COMPRESSION_LEVEL =
"flac-compression-level"
描述：属性描述——可选，表示如果内容为FLAC音频，则指定所需的压缩级别。

F.2.3.2.30 KEY_COMPLEXITY

原型：public static final java.lang.String KEY_COMPLEXITY = "complexity"
描述：属性描述——表示编码复杂度。

F.2.3.2.31 KEY_PROFILE

原型：public static final java.lang.String KEY_PROFILE = "profile"
描述：属性描述——表示编码器需要使用的概要文件。

F.2.3.2.32 KEY_BITRATE_MODE

原型：public static final java.lang.String KEY_BITRATE_MODE = "bitrate-mode"
描述：属性描述——表示编码器应用需要的比特率模式。

F.2.3.2.33 KEY_AUDIO_SESSION_ID

原型：public static final java.lang.String KEY_AUDIO_SESSION_ID = "audio-session-id"
描述：属性描述——表示音频的会话标识。

F.2.3.2.34 KEY_IS_AUTOSELECT

原型：public static final java.lang.String KEY_IS_AUTOSELECT = "is-autoselect"
描述：属性描述——表示track的自主选择，如果未指定，默认为TRUE。

F.2.3.2.35 KEY_IS_DEFAULT

原型：public static final java.lang.String KEY_IS_DEFAULT = "is-default"
描述：属性描述——表示 track 的自主选择，如果未指定，默认为 false。

F.2.3.2.36 KEY_IS_FORCED_SUBTITLE

原型：public static final java.lang.String KEY_IS_FORCED_SUBTITLE = "is-forced-subtitle"
描述：属性描述——表示字幕的强制字段。如果它是强迫字幕。如果未指定，则强制默认为false。

F.2.3.3 方法

F.2.3.3.1 MediaFormat

原型：public MediaFormat()
描述：构造函数。
参数：无。
返回：无。

F.2.3.3.2 containsKey

原型：public final boolean containsKey(java.lang.String name)
描述：判断是否存在 key 值为 name 的信息。
参数：name - java.lang.String 对象，表示 key 值；
返回：boolean型，存在key值为name的信息返回true，否则返回false。

F.2.3.3.3 getInteger

原型：public final int getInteger(java.lang.String name)
描述：获取 key 值为 name 的整型属性值。
参数：name - java.lang.String 对象，表示 key 值；
返回：int型，表示key值为name对应的整形属性值。如果key不存在、或者对应的属性值不是整形就会抛出异常。

F.2.3.3.4 getLong

原型：public final long getLong(java.lang.String name)
描述：获取 key 值为 name 的长整型属性值。
参数：name - java.lang.String 对象，表示 key 值；
返回：long 型，表示key值为name对应的长整形属性值。如果key不存在、或者对应的属性值不是整形就会抛出异常。

F.2.3.3.5 getFloat

原型：public final float getFloat(java.lang.String name)
描述：获取 key 值为 name 的浮点型属性值。
参数：name - java.lang.String 对象，表示 key 值；
返回：float型，表示key值为name对应的浮点型属性值。如果key不存在、或者对应的属性值不是整形就会抛出异常。

F.2.3.3.6 getString

原型：public final String getString(java.lang.String name)
描述：获取key值为name的字符串属性值。
参数：name - java.lang.String对象，表示key值；
返回：String对象，表示key值为name对应的字符串属性值。如果key不存在、或者对应的属性值不是String对象就会抛出异常。

F.2.3.3.7 getByteBuffer

原型：public final ByteBuffer getByteBuffer(java.lang.String name)
描述：获取key值为name的字符串属性值。
参数：name - java.lang.String对象，表示key值；
返回：java.nio.ByteBuffer对象，表示key值为name对应的属性值。如果key不存在、或者对应的属性值不是ByteBuffer对象就会抛出异常。

附 录 G
（规范性附录）
JAVA-系统管理单元

G.1 概述

本附录定义了与系统管理相关的JAVA接口。

G.2 系统管理模块

系统管理模块提供了配置参数访问、软硬件配置信息获取、外设管理和系统操作等的类和方法。
系统管理模块概要见表 G.1。

表 G.1 系统管理模块概要

接口	
PeripheralType	TVOS 所支持的外设类型常量定义接口。
Peripheral	外设描述接口，提供了获取外设名称、状态、类型、ID 等方法。
PeripheralListener	外设事件监听器，由应用程序实现。
类	
PeripheralManager	外设管理器，是外设管理模块的入口类。
DataConfig	配置数据访问类，提供了访问存储于接收终端 NVM 中的配置数据的方法。
HardwareInfo	硬件信息描述类，提供了获取接收终端硬件参数信息的方法。
SoftwareInfo	软件信息描述类，提供了获取接收终端软件参数信息的方法。
SysTool	系统工具类，提供了系统待机、休眠以及重启等操作的方法。
事件	
PeripheralEvent	外设通知事件。

G.2.1 接口org.ngb.system.PeripheralType

原型：public interface org.ngb.system.PeripheralType
描述：TVOS 所支持的外设类型常量定义接口。

G.2.1.1 常量域——外设类型

G.2.1.1.1 PERIPHERAL_ALL

原型：pubic static final java.lang.String PERIPHERAL_ALL = "all"
描述：外设类型——全部。

G.2.1.1.2 PERIPHERAL_MOUSE

原型：pubic static final java.lang.String PERIPHERAL_MOUSE = "mouse"
描述：外设类型——鼠标。

G.2.1.1.3 PERIPHERAL_PC_KEYBOARD

原型：pubic static final java.lang.String PERIPHERAL_PC_KEYBOARD = "keyboard"
描述：外设类型——PC 键盘。

G.2.2 接口org.ngb.system.Peripheral

原型：public interface org.ngb.system.Peripheral
描述：外设描述接口，提供了获取外设名称、状态、类型、ID 等方法。

G.2.2.1 方法

G.2.2.1.1 getType

原型：pubic java.lang.String getType()
描述：获取外设类型。
返回：java.lang.String 对象，表示外设类型，取值详见 PeripheralType 接口“外设类型”常量域定义。

G.2.2.1.2 getID

原型：pubic long getID()
描述：获取外设唯一标识。
参数：无。
返回：long 型，表示外设全局唯一标识。

G.2.2.1.3 getName

原型：pubic java.lang.String getName()
描述：获取外设名称。
参数：无。
返回：java.lang.String 对象，表示外设名称。

G.2.2.1.4 getStatus

原型：pubic int getStatus()
描述：获取外设状态。
参数：无。
返回：int 型，表示外设状态，取值与具体外设类型相关。

G.2.3 接口org.ngb.system.PeripheralListener

原型：public interface org.ngb.system.PeripheralListener
描述：外设事件监听器，由应用程序实现。

G.2.3.1 方法

G.2.3.1.1 processPeripheralEvent

原型：pubic void processPeripheralEvent(org.ngb.system.PeripheralEvent event)
描述：外设消息处理方法。
参数：event - org.ngb.system.PeripheralEvent 对象，表示外设事件消息。

G.2.4 类org.ngb.system.PeripheralManager

原型：public class org.ngb.system.PeripheralManager
描述：外设管理器，提供了外设管理的方法，是外设管理模块的入口类。

G.2.4.1 方法

G.2.4.1.1 getInstance

原型：public static org.ngb.system.PeripheralManager getInstance()
描述：获取系统实现的 org.ngb.system.PeripheralManager 类的唯一实例。
参数：无。
返回：PeripheralManager 对象，表示系统实现的 org.ngb.system.PeripheralManager 类的唯一实例。

G.2.4.1.2 addPeripheralEventListener

原型：public void addPeripheralEventListener(org.ngb.system.PeripheralListener listener)
描述：注册一个外设事件监听器。
参数：listener - org.ngb.system.PeripheralListener 对象，表示待注册的外设事件监听器。
返回：无。

G.2.4.1.3 removePeripheralListener

原型：public void removePeripheralListener(org.ngb.system.PeripheralListener listener)
描述：注销一个外设事件监听器。
参数：listener - org.ngb.system.PeripheralListener 对象，表示待注销的外设事件监听器。
返回：无。

G.2.4.1.4 getAllPeripheralsByType

原型：public org.ngb.system.Peripheral[] getAllPeripheralsByType(java.lang.String strType)
描述：获取指定类型的所有设备。
参数：strType - java.lang.String 对象，表示外设类型，取值详见 org.ngb.system.PeripheralType 接口“外设类型”常量域定义。
返回：org.ngb.system.Peripheral 对象数组，表示符合指定类型的所有设备。若无符合条件的外设，则返回的数组长度为 0。

G.2.4.1.5 getPeripheralsByID

原型：public org.ngb.system.Peripheral getPeripheralsByID(long id)
描述：根据全局唯一 ID 获取外设。
参数：id - long 型，表示外设全局唯一 ID。
返回：org.ngb.system.Peripheral 对象，表示符合条件的外设。若无符合条件的外设，则返回 null。

G.2.4.1.6 uninstallPeripheralByID

原型：public boolean uninstallPeripheralByID(long id)

描述：根据全局唯一 ID 卸载外设。

参数：id - long 型，表示外设全局唯一 ID。

返回：boolean 型，取值 true 表示卸载成功，false 表示卸载失败。

G.2.5 类org.ngb.system.DataConfig

原型：public class org.ngb.system.DataConfig

描述：配置数据访问类，提供了访问存储于接收终端 NVM 中的系统数据表的方法。

——系统数据表用以存储系统设置的、全局的配置信息，采用“键名+键值”的方式访问，键值在 RAM 和 NVM 中都以 JSON 格式存储，键名和键值 JSON 格式对外公开，授权应用可设置/读取系统参数，但不能删除数据项。应用在读取数据时，应根据键名所对应的应用场景进行解析。系统应对应用程序的权限进行验证，只有特权应用才能访问系统数据表。系统数据表的键名和键值 JSON 格式定义见表 G.2。

——用户数据表的访问采用标准的 Properties 接口所提供的方法。

表 G.2 系统数据表键名和键值表

<table>
<tr><th>键名</th><th colspan="2">键值</th></tr>
<tr><td rowspan="2">DVBMainFrequencyInfo</td><td>用途</td><td>描述DVB主频点信息。</td></tr>
<tr><td>JSON格式</td><td>[{
"deliveryType":1,
"deliveryParams":[
{"frequency":626000, "symbolRate":6875, "modulation":3},
{"frequency":634000, "symbolRate":6875, "modulation":3},
{"frequency":642000, "symbolRate":6875, "modulation":3}
]
},
...
{
"deliveryType":10,
"deliveryParams":[
{"frequency":12020, "symbolRate":28800, "polarization":3}
]
},
{
"deliveryType":12,
"deliveryParams":[
{"frequency":642000}
]
}]
注：DVBMainFrequencyInfo键的键值JSON格式可以描述多个传送系统，每个传送系统可以描述多个主频点信息。在本示例中，上述JSON字符串描述了DVB-C、ABS-SS和DTMB传送系统，其中DVB-C传送系统包含3个主频点信息，ABS-SS传送系统包含1个主频点信息，DTMB传送系统包含1个主频点信息。</td></tr>
</table>

表 G.2（续）

键名	键值		
DVBMainFrequencyInfo	说明	deliveryType	int型，表示传送系统类型，可取值详见DeliverySystemType接口和DvbDeliverySystemType接口“传送系统类型”常量域定义。
		frequency	int型，表示调谐频率，单位与deliveryType字段的取值有关： ——若deliveryType=1，表示DVB-C传送系统，单位为千赫（kHz）； ——若deliveryType=10，表示ABS-SS传送系统，单位为兆赫（MHz）； ——若deliveryType=12，表示DTMB传送系统，单位为千赫（kHz）； ——若deliveryType为其他值，则该键无意义。
		symbolRate	int型，表示符号率，单位为千符号每秒（ksymbol/s）。 ——若deliveryType=12，表示DTMB传送系统，则该键无意义。
		modulation	int型，表示调制方式，取值与deliveryType字段的取值有关： ——若deliveryType=1，表示DVB-C传送系统，取值详见DvbcTunningParameters类的“调制方式”常量域定义； ——若deliveryType为其他值，则该键无意义。
		polarization	int型，表示极化方式，取值与deliveryType字段的取值有关： ——若deliveryType=10，表示ABS-SS传送系统，取值详见AbsssTunningParameters类的“极化方式”常量域定义； ——若deliveryType为其他值，则该键无意义。
EPGSetting	用途	描述EPG搜索设置信息。	
	JSON格式	{"search_start_date":0, "search_days":7, "program_event_maxcount":255}	
	说明	search_start_date	int型，表示搜索节目表的起始日期。缺省为0，即将当天作为搜索的起始日期，1表示第二天，2表示第三天,……,依次类推。
		search_days	int型，表示将搜索连续多少天的节目表。
		program_event_max_count	int型，表示EPG搜索节目事件个数的最大值，若取值-1，则表示无限制。
AudioSetting	用途	描述音频设置信息。	

	JSON格式	{
		"enableGlobalVolume":0,
		"outputVolume":50,
		"spdifMode":"PCM",
		"soundMode":0
		}	

表 G.2（续）

键名	键值		
AudioSetting	说明	enableGlobalVolume	int型，表示是否启用统一音量控制，取值： ——0 - 表示应用允许用户单独设置每个频道的音量； ——1 - 表示所有直播电视、音频广播、NVOD、马赛克的音量统一为outputVolume值。
		outputVolume	int型，表示接收终端输出模拟音量，取值范围0～100，0表示静音，100表示最大音量，设置后立即生效。
		spdifMode	字符串，表示接收终端SPDIF输出接口信号格式，取值： ——"RIGINAL" - 表示原码输出（未解码）； ——"PCM" - 表示PCM格式输出（解码）； ——"default" - 表示根据PMT表的stream_type字段值决定原码或PCM输出； ——"OFF" - 表示关闭SPDIF输出。
		soundMode	int型，表示声道模式，取值： ——0 - 表示立体声； ——1 - 表示单左声道，即右声道没有声音； ——2 - 表示单右声道，即左声道没有声音； ——3 - 表示混音。
VideoSetting	用途	描述视频设置信息。	

	JSON格式	{
		"videoStandard":27400,
		"OSDAlpha":0
		}	

	说明	videoStandard	int型，表示视频输出制式，取值见VideoSetting类“输出制式”常量域定义。
		OSDAlpha	int型，表示OSD层的透明度，取值范围0～100，0表示完全不透明，100表示完全透明。
UserPreference	用途	描述用户偏好设置信息。	

	JSON格式	{
		"audioLang":"zho",
		"osdLang":"eng"
		}	

	说明	audioLang	字符串，表示用户偏好的音频语种，语种三字母代码遵循GB/T 4880.2—2000标准。
		osdLang	字符串，表示用户偏好的界面语种，语种三字母代码遵循GB/T 4880.2—2000标准。
Portal	用途	描述门户服务器地址信息。	

	JSON格式	{
		"address":"http://ngb.com",
		"port":8080
		}	

表 G.2（续）

<table>
<tr><td>键名</td><td colspan="3">键值</td></tr>
<tr><td rowspan="2">Portal</td><td rowspan="2">说明</td><td>address</td><td>字符串，表示门户服务器地址。</td></tr>
<tr><td>port</td><td>int型，表示门户服务器访问端口。</td></tr>
<tr><td rowspan="4">NTP</td><td>用途</td><td colspan="2">描述NTP服务器地址信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2">{
"address":"http://ntp.com",
"port":8080
}</td></tr>
<tr><td rowspan="2">说明</td><td>address</td><td>字符串，表示NTP服务器地址。</td></tr>
<tr><td>port</td><td>int型，表示NTP服务器访问端口。</td></tr>
<tr><td rowspan="4">VODChannel</td><td>用途</td><td colspan="2">描述VOD服务器地址信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2">{
"MD5":"5A8B6493",
"VODParams":[
{
"frequency":634000,
"symbolRate":6875,
"modulation":3,
"QAMName":1234
},
{
"frequency":642000,
"symbolRate":6875,
"modulation":3,
"QAMName":4321
},
{
"frequency":650000,
"symbolRate":6875,
"modulation":3,
"QAMName":8888
}
]
}
注：仅适用于有线数字电视。</td></tr>
<tr><td rowspan="5">说明</td><td>MD5</td><td>字符串，表示配置文件MD5码。</td></tr>
<tr><td>frequency</td><td>int型，表示IPQAM频点频率，单位为千赫（kHz）。</td></tr>
<tr><td>symbolRate</td><td>int型，表示IPQAM频点符号率，单位为千符号每秒(ksymbol/s)。</td></tr>
<tr><td>modulation</td><td>int型，表示IPQAM频点调制方式，取值详见DvbcTunningParameters类的“调制方式”常量域定义。</td></tr>
<tr><td>QAMName</td><td>int型，表示VOD区域码。</td></tr>
</table>

表 G.2（续）

<table>
<tr><th>键名</th><th colspan="3">键值</th></tr>
<tr><td rowspan="5">UserInfo</td><td>用途</td><td colspan="2">描述终端用户信息。</td></tr>
<tr><td>JSON
格式</td><td colspan="2">{
"account":"882012459",
"customerGroup":"group1",
"adminPassword":"12345"
}</td></tr>
<tr><td rowspan="3">说明</td><td>account</td><td>字符串，表示设备账户（UserID,由BOSS/SMS同步的用户账号信息）。</td></tr>
<tr><td>customerGroup</td><td>字符串，表示终端所处的用户组。</td></tr>
<tr><td>adminPassword</td><td>字符串，表示管理员密码，用于进入系统设置界面和观看加锁频道。</td></tr>
<tr><td rowspan="3">Autodeployer</td><td>用途</td><td colspan="2">描述应用自动部署信息。</td></tr>
<tr><td>JSON
格式</td><td colspan="2">{
"mode":"auto-ip",
"ocPath":[
{
"deliveryType":1,
"deliveryParams":[
{"frequency":626000, "symbolRate":6875, "modulation":3},
{"frequency":634000, "symbolRate":6875, "modulation":3},
{"frequency":642000, "symbolRate":6875, "modulation":3}
]
},
...
],
"ipPath":[
{
"udpPath":"192.168.1.12",
"udpPort":8080,
"downloadTimeOut":60
},
...
]
}
注：单向广播通道支持多个传送系统、多个频点下发XML信令文件；双向宽带通道支持多个UDP服务器下发XML信令文件。</td></tr>
</table>

表 G.2（续）

键名	键值		
Autodeployer	说明	mode	字符串，表示自动部署XML信令文件的获取方式，取值： ——"ip" - 表示从双向宽带通道获取XML信令文件； ——"oc" - 表示从单向广播通道获取XML信令文件； ——"auto-ip" - 表示自适应，优先从双向宽带通道获取XML信令文件； ——"auto-oc" - 表示自适应，优先从单向广播通道获取XML信令文件。
		ocPath	JSON对象类型，表示自动部署xml信令文件的OC下载路径。 ——deliveryType：int型，表示传送系统类型，同DVBMainFrequencyInfo键的解释； ——deliveryParams：JSON对象，同DVBMainFrequencyInfo键的解释。
Autodeployer	说明	ipPath	JSON对象类型，表示自动部署xml信令文件的ip下载路径。 ——udpPath：字符串，表示UDP服务器地址； ——udpPort：int型，表示UDP服务端口； ——downloadTimeOut：int型，表示应用下载超时时间，单位为秒。

G.2.5.1 方法

G.2.5.1.1 getInstance

原型：public static DataConfig getInstance()
描述：获取系统实现的 DataConfig 类的唯一实例。
参数：无。
返回：DataConfig 类单例。

G.2.5.1.2 getProperty

原型：public java.lang.String getProperty(java.lang.String strItem)
描述：获取系统数据表某个数据项的值（从内存中读）。
参数：strItem - java.lang.String 对象，表示数据项键名。
返回：java.lang.String 对象，表示数据项键值，采用 JSON 字符串描述；若该数据项不存在，则返回 null。

G.2.5.1.3 setProperty

原型：public int setProperty(java.lang.String strItem, java.lang.String strValue)
描述：设置系统数据表某个数据项的值（写入到内存）。
参数：strItem - java.lang.String 对象，表示数据项键名；
strValue - java.lang.String 对象，表示数据项键值，采用 JSON 字符串描述。
返回：int 型，表示修改结果，取值如下：
——若修改成功，则返回值大于 0；

——若未知原因导致修改内容失败，则返回 0；
——若该数据项不存在，则返回-1。

G.2.5.1.4 restoreDefault

原型：public boolean restoreDefault()
描述：将 NVM 中的系统数据表恢复到出厂设置状态，并同步更新内存中的系统数据表。
返回：boolean 型，取值 true 表示恢复出厂设置成功，false 表示恢复出厂设置失败。

G.2.5.1.5 restoreFromNvm

原型：public boolean restoreFromNvm()
描述：将 NVM 中的系统数据表读取到内存中，覆盖内存中的当前数据。
参数：无。
返回：boolean 型，表示读取结果，取值 true 表示读取成功，false 表示读取失败。

G.2.5.1.6 saveToNvm

原型：public boolean saveToNvm()
描述：将内存中的系统数据表写入到 NVM 中，覆盖 NVM 中的系统数据表。
参数：无。
返回：boolean 型，表示写入结果，取值 true 表示写入成功，false 表示写入失败。

G.2.6 类org.ngb.system.HardwareInfo

原型：public class org.ngb.system.HardwareInfo
描述：硬件配置信息类，提供了获取接收终端硬件配置参数信息的方法。

G.2.6.1 常量域——硬件配置项

G.2.6.1.1 FLASH_SIZE

原型：public static final java.lang.String FLASH_SIZE = "flash_size"
描述：接收终端闪存的大小，单位为兆字节（MB）。

G.2.6.1.2 RAM_SIZE

原型：public static final java.lang.String RAM_SIZE = "ram_size"
描述：接收终端内存的大小，单位为兆字节（MB）。

G.2.6.1.3 RAM_TYPE

原型：public static final java.lang.String RAM_TYPE = "ram_type"
描述：接收终端内存的类型，取值“SDRAM”、“DDR”等。

G.2.6.1.4 SOC_MODEL

原型：public static final java.lang.String SOC_MODEL = "soc_model"
描述：接收终端主芯片的型号。

G.2.6.1.5 SOC_FREQUENCY

原型：public static final java.lang.String SOC_FREQUENCY = "soc_frequency"

描述：接收终端主芯片的工作频率，单位为兆赫（MHz）。

G.2.6.1.6 SOC_PROVIDER

原型：public static final java.lang.String SOC_PROVIDER = "soc_provider"
描述：接收终端主芯片的提供商名称。

G.2.6.1.7 DEFINITION_TYPE

原型：public static final java.lang.String DEFINITION_TYPE = "definition_type"
描述：接收终端的清晰度类型，可取值"HD"、"SD"。

G.2.6.1.8 HW_VERSION

原型：public static final java.lang.String HW_VERSION = "hw_version"
描述：接收终端硬件版本号。

G.2.6.1.9 STB_BRAND

原型：public static final java.lang.String STB_BRAND = "stb_brand"
描述：接收终端的品牌名称。

G.2.6.1.10 STB_MODEL

原型：public static final java.lang.String STB_MODEL = "stb_model"
描述：接收终端的型号。

G.2.6.1.11 STB_PROVIDER

原型：public static final java.lang.String STB_PROVIDER = "stb_provider"
描述：接收终端的提供商名称。

G.2.6.1.12 STB_SERIAL_NUMBER

原型：public static final java.lang.String STB_SERIAL_NUMBER = "stb_serial_number"
描述：接收终端的序列号。

G.2.6.1.13 TRANSPORT_TYPE

原型：public static final java.lang.String TRANSPORT_TYPE = "transport_type"
描述：接收终端的传输模式类型，可取值"DVB-C"、"DVB-S"、"DVB-T"等组合。

G.2.6.2 方法

G.2.6.2.1 getProperty

原型：public static java.lang.String getProperty(java.lang.String key)
throws IllegalArgumentException
描述：获取接收终端硬件配置信息。
参数：key - java.lang.String 对象，表示硬件配置属性关键字，取值详见 HardwareInfo 类的“硬件配置项”常量域定义。
返回：java.lang.String 对象，表示硬件配置参数。

异常：IllegalArgumentException - 若硬件配置项关键字不合法，则抛出此异常。

G.2.7 类org.ngb.system.SoftwareInfo

原型：public class org.ngb.system.SoftwareInfo
描述：软件配置信息类，提供了获取接收终端软件配置参数信息的方法。

G.2.7.1 常量域——软件配置项

G.2.7.1.1 CA_NAME

原型：public static final java.lang.String CA_NAME = "ca_name"
描述：CA 模块的名称。

G.2.7.1.2 CA_PROVIDER

原型：public static final java.lang.String CA_PROVIDER = "ca_provider"
描述：CA 模块的提供商名称。

G.2.7.1.3 CA_VERSION

原型：public static final java.lang.String CA_VERSION = "ca_version"
描述：CA 模块的版本号。

G.2.7.1.4 DRVIER_VERSION

原型：public static final java.lang.String DRIVER_VERSION = "driver_version"
描述：接收终端驱动版本号。

G.2.7.1.5 LOADER_NAME

原型：public static final java.lang.String LOADER_NAME = "loader_name"
描述：软件更新模块（loader）的名称。

G.2.7.1.6 LOADER_PROVIDER

原型：public static final java.lang.String LOADER_PROVIDER = "loader_provider"
描述：软件更新模块（loader）的提供商名称。

G.2.7.1.7 LOADER_SIZE

原型：public static final java.lang.String LOADER_SIZE = "loader_size"
描述：软件更新模块（loader）的大小，单位为千字节（KB）。

G.2.7.1.8 LOADER_VERSION

原型：public static final java.lang.String LOADER_VERSION = "loader_version"
描述：软件更新模块（loader）的版本。

G.2.7.1.9 MW_COPYRIGHT

原型：public static final java.lang.String MW_COPYRIGHT = "mw_copyright"
描述：系统软件的版权信息。

G.2.7.1.10 MW_NAME

原型：public static final java.lang.String MW_NAME = "mw_name"
描述：系统软件的名称。

G.2.7.1.11 MW_NVM_SIZE

原型：public static final java.lang.String MW_NVM_SIZE = "mw_nvm_size"
描述：系统软件所占用的闪存空间，单位为千字节（KB）。

G.2.7.1.12 MW_PLATFORM_LEVEL

原型：public static final java.lang.String MW_PLATFORM_LEVEL = "mw_platform_level"
描述：系统软件所支持的平台级别。

G.2.7.1.13 MW_PLATFORM_PROFILE

原型：public static final java.lang.String MW_PLATFORM_PROFILE = "mw_platform_profile"
描述：系统软件所支持的平台档次。

G.2.7.1.14 MW_PROVIDER

原型：public static final java.lang.String MW_PROVIDER = "mw_provider"
描述：系统软件的提供商名称。

G.2.7.1.15 MW_RAM_SIZE

原型：public static final java.lang.String MW_RAM_SIZE = "mw_ram_size"
描述：系统软件所占用的内存空间，单位为千字节（KB）。

G.2.7.1.16 MW_RELEASE_DATE

原型：public static final java.lang.String MW_RELEASE_DATE = "mw_release_date"
描述：系统软件的发布日期。

G.2.7.1.17 MW_VERSION

原型：public static final java.lang.String MW_VERSION = "mw_version"
描述：系统软件的版本号。

G.2.7.1.18 OS_NAME

原型：public static final java.lang.String OS_NAME = "os_name"
描述：操作系统软件的名称。

G.2.7.1.19 OS_PROVIDER

原型：public static final java.lang.String OS_PROVIDER = "os_provider"
描述：操作系统软件的提供商名称。

G.2.7.1.20 OS_VERSION

原型：public static final java.lang.String OS_VERSION = "os_version"

描述：操作系统软件的版本号。

G.2.7.2 方法

G.2.7.2.1 getProperty

原型：public static java.lang.String getProperty(java.lang.String key)
throws IllegalArgumentException
描述：获取接收终端软件配置信息。
参数：key - java.lang.String 对象，表示软件配置项关键字，取值详见 SoftwareInfo 类的“软件配置项”常量域定义。
返回：java.lang.String 对象，表示软件配置参数。
异常：IllegalArgumentException - 若软件配置项关键字不合法，则抛出此异常。

G.2.8 类org.ngb.system.SysTool

原型：public class org.ngb.system.SysTool
描述：系统工具类，提供了系统待机、休眠以及重启等操作的方法。系统应对应用程序的权限进行验证，只有授权应用才能调用本类提供的方法。
——待机：接收终端的 CPU 仍在工作，用以运行一些后台程序，例如推送下载、软件升级监测等，关闭所有音视频输出，从表象上看接收终端停止了工作；
——休眠：接收终端的 CPU 停止工作，彻底切断 CPU 和主板电源，靠外部单片机或其他方式监控遥控器发出的激活命令，然后启动开关电源给 CPU 和主板供电，实现遥控开机。
应根据接收机的实际能力实现本类的相关方法。

G.2.8.1 方法

G.2.8.1.1 getInstance

原型：public static org.ngb.system.SysTool getInstance()
描述：获取系统实现的 org.ngb.system.SysTool 类的唯一实例。
参数：无。
返回：org.ngb.system.SysTool 对象，表示系统实现的 org.ngb.system.SysTool 类的唯一实例。

G.2.8.1.2 getStandByStatus

原型：public boolean getStandByStatus()
描述：获取待机状态。
参数：无。
返回：boolean 型，取值 true 表示进入待机状态，false 表示退出待机状态（即工作状态）。

G.2.8.1.3 reboot

原型：public void reboot()
描述：重启接收机。
参数：无。
返回：无。

G.2.8.1.4 sleep

原型：public void sleep()
描述：控制接收机进入休眠状态，CPU 将断电停止工作。
参数：无。
返回：无。

G.2.8.1.5 standBy

原型：public void standBy()
描述：控制接收机转入待机状态。CPU 仍在运行。
参数：无。
返回：无。

G.2.8.1.6 wakeUp

原型：public void wakeUp()
描述：唤醒接收机进入工作状态。
参数：无。
返回：无。

G.2.9 事件org.ngb.system.PeripheralEvent

原型：public abstract class org.ngb.system.PeripheralEvent
描述：外设通知事件。

G.2.9.1 常量域——外设事件类型

G.2.9.1.1 TYPE_FOUND

原型：public static final int TYPE_FOUND = 0
描述：发现外设，可获得外设信息，但不可访问外设。

G.2.9.1.2 TYPE_READY

原型：public static final int TYPE_READY = 1
描述：外设就绪，可进行访问。

G.2.9.1.3 TYPE_ERROR

原型：public static final int TYPE_ERROR = 2
描述：外设存在不可恢复错误，无法访问。

G.2.9.1.4 TYPE_PLUGOUT

原型：public static final int TYPE_PLUGOUT = 3
描述：外设已被移除。

G.2.9.2 方法

G.2.9.2.1 getPeripheral

原型：public org.ngb.system.Peripheral getPeripheral()
描述：获取发出此事件的外设。

参数：无。

返回：org.ngb.system.Peripheral 对象，表示发出此事件消息的外设。

G.2.9.2.2 getType

原型：public int getType()

描述：获取外设事件类型。

参数：无。

返回：int 型，表示外设事件类型，取值详见 org.ngb.system.PeripheralEvent 类的“外设事件类型”常量域定义。

G.3 OTA升级模块

OTA 升级模块提供了 OTA 软件升级侦测与处理操作的类和方法。

OTA 升级模块概要见表 G.3。

表 G.3 OTA 升级模块概要

接口	
OTAEventListener	OTA 事件监听器，由应用程序实现。
类	
OTAManager	OTA 管理器，是 OTA 功能模块的入口类。
事件	
OTAEvent	OTA 事件。

G.3.1 接口org.ngb.system.OTAEventListener

原型：public interface org.ngb.system.OTAEventListener

描述：OTA 事件监听器，由应用程序实现。

G.3.1.1 方法

G.3.1.1.1 processEvent

原型：void processEvent(org.ngb.system.OTAEvent event)

描述：OTA 升级消息处理方法。

参数：event - org.ngb.system.OTAEvent 对象，表示 OTA 升级提示消息。

返回：无。

G.3.2 类org.ngb.system.OTAManager

原型：public class org.ngb.system.OTAManager

描述：OTA 管理器。系统应对应用程序的权限进行验证，只有特权应用才能调用本类提供的方法。

G.3.2.1 方法

G.3.2.1.1 getInstance

原型：public static org.ngb.system.OTAManager getInstance()

描述：获取系统实现的 org.ngb.system.OTAManager 类的唯一实例。
参数：无。
返回：OTAManager 对象，表示系统实现的 org.ngb.system.OTAManager 类的唯一实例。

G.3.2.1.2 checkOTA

原型：public boolean checkOTA()
描述：判断前端是否布署新的 OTA 升级。该方法主要用于手动检测 OTA 升级信息。
返回：boolean 型，表示判断结果，取值 true 表示有 OTA 升级，false 表示无 OTA 升级。

G.3.2.1.3 getOTAName

原型：public java.lang.String getOTAName()
描述：获取 OTA 升级事件名称，不同于 OTA 提供者名称，指本次 OTA 升级事件的文本描述。
参数：无。
返回：java.lang.String 对象，表示 OTA 升级事件名称。若前端未提供升级事件名称，则返回 null。

G.3.2.1.4 startOTA

原型：public boolean startOTA()
描述：异步方法，开始升级，调用后立即返回。
参数：无。
返回：boolean型，取值true表示启动OTA升级成功，false表示启动OTA升级失败。

G.3.2.1.5 addOTAEventListener

原型：public void addOTAEventListener(org.ngb.system.OTAEventListener listener)
描述：注册一个 OTA 事件监听器。
参数：listener - org.ngb.system.OTAEventListener 对象，表示待注册的 OTA 事件监听器。
返回：无。

G.3.2.1.6 removeOTAEventListener

原型：public void removeOTAEventListener(org.ngb.system.OTAEventListener listener)
描述：注销一个 OTA 事件监听器。
参数：listener - org.ngb.system.OTAEventListener 对象，表示待注销的 OTA 事件监听器。
返回：无。

G.3.3 事件org.ngb.system.OTAEvent

原型：public class org.ngb.system.OTAEvent extends EventObject
描述：OTA 事件。

G.3.3.1 常量域——升级类型

G.3.3.1.1 OTA_FORCE

原型：public static final int OTA_FORCE = 0
描述：强制升级。运营前端布署了新的 OTA 升级包，通知接收终端进行 OTA 升级。应用接收到该消息

后不提示用户，直接调用 OTAManager 类的 startOTA()方法直接强制进行 OTA 升级。

G.3.3.1.2 OTA_NORMAL

原型：public static final int OTA_NORMAL = 1

描述：正常升级。运营前端布署了新的 OTA 升级包，通知接收终端进行 OTA 升级。应用接收到该消息后提示用户，待用户确认后才调用 OTAManager 类的 startOTA()方法进行升级。

G.3.3.2 方法

G.3.3.2.1 getType

原型：public int getType()

描述：获取 OTA 升级事件类型。

参数：无。

返回：int 型，表示 OTA 升级事件类型，可取值为 OTA_FORCE 或 OTA_NORMAL，详见 OTAEvent 类的“升级类型”常量域定义。

G.4 存储管理模块

存储管理模块提供了存储设备以及存储设备分区访问的类和方法。

存储管理模块概要见表 G.4。

表 G.4 存储管理模块概要

接口	
Storage	描述了存储设备信息，例如名称、大小、空闲状态、分区等。
StorageEventListener	存储事件监听器，由应用程序实现。
StoragePartition	描述了存储设备的分区信息，例如名称、大小、空闲状态、访问路径、分区类型等。
类	
StorageManager	提供了管理存储设备和存储设备分区的方法。
事件	
StorageEvent	与存储设备相关的存储事件。

G.4.1 接口org.ngb.system.Storage

原型：public interface org.ngb.system.Storage

描述：存储设备描述接口，提供了获取序列号、分区等方法。

G.4.1.1 方法

G.4.1.1.1 getAllPartitions

原型：org.ngb.system.StoragePartition[] getAllPartitions()

描述：获取存储设备的所有分区对象。

参数：无。

返回：org.ngb.system.StoragePartition 对象数组，表示存储设备的所有分区。若无，则返回的数组长度为 0。

G.4.1.1.2 getSerialNumber

原型： java.lang.String getSerialNumber()
描述：获取存储设备序列号。
参数：无。
返回： java.lan.String 对象，表示存储设备的序列号。

G.4.2 接口org.ngb.system.StorageEventListener

原型：public interface org.ngb.system.StorageEventListener
描述：存储事件监听器，由应用程序实现。

G.4.2.1 方法

G.4.2.1.1 processStorageEvent

原型：void processStorageEvent(org.ngb.system.StorageEvent event)
描述：存储事件处理方法。
参数：event - org.ngb.system.StorageEvent 对象，表示存储事件。
返回：无。

G.4.3 接口org.ngb.system.StoragePartition

原型：public interface org.ngb.system.StoragePartition
描述：存储分区描述接口，提供了获取存储设备分区的名称、大小、空闲状态、访问路径等方法。

G.4.3.1 方法

G.4.3.1.1 getID

原型：long getID()
描述：获取存储设备分区全局唯一 ID。
参数：无。
返回：long 型，表示存储设备分区全局唯一 ID。

G.4.3.1.2 getName

原型：java.lang.String getName()
描述：获取存储设备分区名称。
参数：无。
返回：java.lang.String 对象，表示存储设备分区名称。

G.4.3.1.3 getPath

原型：java.lang.String getPath()
描述：获取存储设备分区的访问路径。
参数：无。
返回：java.lang.String 对象，表示存储设备分区的访问路径。

G.4.3.1.4 getStatus

原型：java.lang.String getStatus()
描述：获取存储设备分区状态。
参数：无。
返回：java.lang.String 对象，表示存储设备分区状态，例如“状态良好”、“未格式化”等。

G.4.3.1.5 getFreeSize

原型：long getFreeSize()
描述：获取存储设备分区空闲的存储空间大小。
参数：无。
返回：long 型，表示存储设备分区空闲的存储空间大小，单位为千字节（KB）。

G.4.3.1.6 getTotalSize

原型：long getTotalSize()
描述：获取存储设备分区总的存储空间大小。
参数：无。
返回：long 型，表示存储设备分区总的存储空间大小，单位为千字节（KB）。

G.4.4 类org.ngb.system.StorageManager

原型：public class org.ngb.system.StorageManager
描述：存储设备管理器，提供了管理存储设备和存储设备分区的方法，是存储设备管理功能模块的入口类。系统应对应用程序的权限进行验证，只有授权应用才能调用本类提供的方法。

G.4.4.1 方法

G.4.4.1.1 getInstance

原型：public static org.ngb.system.StorageManager getInstance()
描述：获取系统实现的存储设备管理器的唯一实例。
参数：无。
返回：org.ngb.system.StorageManager 对象单例。

G.4.4.1.2 addStorageEventListener

原型：public void addStorageEventListener(org.ngb.system.StorageEventListener listener)
描述：注册一个存储事件监听器。
参数：listener - org.ngb.system.StorageEventListener 对象，表示待注册的存储事件监听器。
返回：无。

G.4.4.1.3 removeStorageEventListener

原型：public void removeStorageEventListener(org.ngb.system.StorageEventListener listener)
描述：注销一个存储事件监听器。
参数：listener - org.ngb.system.StorageEventListener 对象，表示待注销的存储事件监听器。

G.4.4.1.4 getAllStorages

原型：public org.ngb.system.Storage[] getAllStorages()

描述：获取所有存储设备对象。
返回：org.ngb.system.Storage 对象数组，表示所有存储设备对象。

G.4.4.1.5 uninstallStorage

原型：public boolean uninstallStorage(org.ngb.system.Storage storage)
描述：卸载存储设备。
参数：storage - org.ngb.system.Storage 对象，表示存储设备信息。
返回：boolean 型，表示卸载结果，取值 true 表示卸载成功，false 表示卸载失败。

G.4.5 事件org.ngb.system.StorageEvent

原型：public class org.ngb.system.StorageEvent
描述：与存储设备相关的存储事件。

G.4.5.1 常量域——消息类型

G.4.5.1.1 TYPE_PARTITION_FOUND

原型：public static final int TYPE_PARTITION_FOUND = 22
描述：设备消息类型——分区发现。

G.4.5.1.2 TYPE_PARTITION_MOUNTED

原型：public static final int TYPE_PARTITION_MOUNTED = 23
描述：设备消息类型——分区挂载成功。

G.4.5.1.3 TYPE_PARTITION_MOUNT_FAILED

原型：public static final int TYPE_PARTITION_MOUNT_FAILED = 24
描述：设备消息类型——分区挂载失败。

G.4.5.1.4 TYPE_PARTITION_UNINSTALL

原型：public static final int TYPE_PARTITION_UNINSTALL = 25
描述：设备消息类型——分区卸载消息。

G.4.5.1.5 TYPE_INSUFFICIENT_SPACE

原型：public static final int TYPE_INSUFFICIENT_SPACE = 32
描述：存储事件——无足够空间。

G.4.5.2 方法

G.4.5.2.1 getType

原型：public int getType()
描述：获取存储事件消息类型。
参数：无。
返回：int 型，表示存储事件消息类型，取值详见 StorageEvent 类的“消息类型”常量域定义。

G.4.5.2.2 getStorage

原型：public org.ngb.system.Storage getStorage()
描述：获取发送此事件的存储设备对象。
参数：无。
返回：org.ngb.system.Storage 对象，表示发送此事件的存储设备对象。

G.4.5.2.3 getStoragePartition

原型：public org.ngb.system.StoragePartition getStoragePartition()
描述：获取发送此事件的存储设备分区对象。
参数：无。
返回：org.ngb.system.StoragePartition对象，表示发送此事件的存储设备分区对象。

附 录 H
（规范性附录）
JAVA-应用引擎单元

H.1 概述

本附录定义了应用引擎 JAVA 接口，包括频道搜索模块、电子节目指南模块和信息搜索模块。

H.2 频道搜索模块

频道搜索模块提供了与频道搜索相关的类和方法。
频道搜索模块概要见表 H.1。

表 H.1 频道搜索模块概要

接口	
ChannelScanListener	搜台过程事件的监听器，由应用程序实现。
类	
ChannelScanEngine	搜索引擎。频道搜索功能单元的入口类。
事件	
ChannelScanEvent	频道搜索事件，基类。
ChannelScanFailureEvent	频道搜索失败事件，继承 ChannelScanEvent 类。
ChannelScanFinishEvent	频道搜索结束事件，继承 ChannelScanEvent 类。
ChannelScanNITSuccessEvent	频道搜索成功解析 NIT 事件，继承 ChannelScanEvent 类。
ChannelScanSuccessEvent	频道搜索成功事件，继承 ChannelScanEvent 类。

频道搜索方式定义：
——手动搜索—根据设定的调谐解调参数，在单个频点内搜索广播电视节目频道；
——自动搜索—根据运营商指定的起始频道调谐解调参数，搜索 NIT，然后根据 NIT 的指示自动进行调谐解调，搜索全网络的广播电视节目频道；运营商有可能会指定多个起始频点，只要在其中任意一个频点搜索到 NIT，即可完成自动搜索；
——区间搜索—按照我国数字电视频道分配表，在指定的频率范围内逐频点搜索广播电视节目频道。

H.2.1 接口org.ngb.toolkit.channelscan.ChannelScanListener

原型：public interface org.ngb.toolkit.channelscan.ChannelScanListener
extends java.util.EventListener
描述：搜台过程事件的监听器，由应用程序实现。

H.2.1.1 方法

H.2.1.1.1 processEvent

原型：void processEvent(org.ngb.toolkit.channelscan.ChannelScanEvent event)

描述：频道扫描事件处置方法。

参数：event - org.ngb.toolkit.channelscan.ChannelScanEvent对象，表示搜台事件。应用通过instanceof方法进一步判断事件对象的原型。

返回：无。

H.2.2 类org.ngb.toolkit.channelscan.ChannelScanEngine

原型：public class org.ngb.toolkit.channelscan.ChannelScanEngine

描述：搜索引擎，频道搜索功能单元的入口类。

H.2.2.1 常量域——频道搜索方式

H.2.2.1.1 CHANNELSCAN_TYPE_MANUAL

原型：public static final int CHANNELSCAN_TYPE_MANUAL = 0

描述：频道搜索方式——手动搜索。

H.2.2.1.2 CHANNELSCAN_TYPE_NIT

原型：public static final int CHANNELSCAN_TYPE_NIT = 1

描述：频道搜索方式——自动搜索。

H.2.2.1.3 CHANNELSCAN_TYPE_ZONE

原型：public static final int CHANNELSCAN_TYPE_ZONE = 2

描述：频道搜索方式——区间搜索。

H.2.2.1.4 CHANNELSCAN_TYPE_JSON

原型：public static final int CHANNELSCAN_TYPE_JSON = 3

描述：频道搜索方式——JSON搜索。该类型有单独的搜索入口函数startScanExt。

H.2.2.2 方法

H.2.2.2.1 createInstance

原型：public static org.ngb.toolkit.channelscan.ChannelScanEngine createInstance(int tunerId)throws org.davic.mpeg.ResourceException

描述：创建搜台引擎对象。

参数：tunerId – int型，表示要创建的搜索引擎对应的tunerid，不填写该参数时默认tunerid为0。

返回：org.ngb.toolkit.channelscan.ChannelScanEngine对象，表示搜台引擎。

异常：org.davic.mpeg.ResourceException – 若底层资源不足，则抛出此异常。

H.2.2.2.2 addChannelScanListener

原型：public void addChannelScanListener(org.ngb.toolkit.channelscan.ChannelScanListener listener)

描述：注册搜索过程状态监听器。

参数：listener - org.ngb.toolkit.channelscan.ChannelScanListener对象，表示待注册的搜索过程监听器对象。

返回：无。

H.2.2.2.3 removeChannelScanListener

原型：public void removeChannelScanListener(org.ngb.toolkit.channelscan.
org.ngb.toolkit.channelscan.ChannelScanListener listener)

描述：注销搜索过程状态监听器。

参数：listener - org.ngb.toolkit.channelscan.ChannelScanListener对象，表示待注销的搜索过程监听器对象。

返回：无。

H.2.2.2.4 startScan

原型：public void startScan(int type, org.ngb.broadcast.dvb.tuner.TuningParameters[] params)
throws java.lang.IllegalArgumentException,org.davic.mpeg.ResourceException

描述：异步方法，启动频道搜索。搜索结果通过搜索事件
org.ngb.toolkit.channelscan.ChannelScanEvent发送给应用程序。
——当搜索完NIT表时，发送org.ngb.toolkit.channelscan.ChannelScanNITSuccessEvent；
——当搜索完一个频点时，发送org.ngb.toolkit.channelscan.ChannelScanSuccessEvent；
——当搜索结束时，发送org.ngb.toolkit.channelscan.ChannelScanFinishEvent；
——当搜索过程由于各种原因失败时，发送
org.ngb.toolkit.channelscan.ChannelScanFailureEvent。应用可以通过
ChannelScanFailureEvent.getReason()得到失败的具体原因。

参数：type - int型，表示搜索类型，可取值为CHANNELSCAN_TYPE_MANUAL、CHANNELSCAN_TYPE_NIT或CHANNELSCAN_TYPE_ZONE，详见频道搜索方式常量域定义；
params - org.ngb.broadcast.dvb.tuner.TuningParameters对象数组，表示搜索调谐解调参数。数组的长度与搜索类型相关：
——若type = CHANNELSCAN_TYPE_MANUAL，数组的长度为1；
——若type = CHANNELSCAN_TYPE_NIT，数组的长度等于运营商部署的起始频点数量；
——若type = CHANNELSCAN_TYPE_ZONE，数组的长度为2，params[0]表示开始搜索频点的调谐参数，params[1]表示终止搜索频点的调谐参数。

返回：无。

异常：java.lang.IllegalArgumentException - 若参数无效，则抛出此异常；
org.davic.mpeg.ResourceException – 若底层资源不足，则抛出此异常。

H.2.2.2.5 startScanExt

原型：public void startScanExt(org.ngb.broadcast.dvb.tuner.TuningParameters params,int pid, int tableid)
throws java.lang.IllegalArgumentException, org.davic.mpeg.ResourceException

描述：异步方法，启动JSON格式节目信息搜索。对应搜索类型：CHANNELSCAN_TYPE_JSON，结果通过事件org.ngb.toolkit.channelscan.ChannelScanEvent发送给应用程序。
——当搜索结束时，发送org.ngb.toolkit.channelscan.ChannelScanFinishEvent；
——当搜索过程由于各种原因失败时，发送
org.ngb.toolkit.channelscan.ChannelScanFailureEvent。应用可以通过

org.ngb.toolkit.channelscan.ChannelScanFailureEvent.getReason()得到失败的具体原因。

——当该搜索启动后，需要强制取消时，跟其他搜索类型一致，调用cancel()函数。

参数：params - org.ngb.broadcast.dvb.tuner.TuningParameters对象，表示节目信息数据所在频点的调谐解调参数。

pid - 节目信息数据所在TS包的PID

tableid- 节目信息数据分配的tableid

返回：无。

异常：java.lang.IllegalArgumentException - 若参数无效，则抛出此异常；

org.davic.mpeg.ResourceException – 若底层资源不足，则抛出此异常。

H.2.2.2.6 cancel

原型：public void cancel()

描述：异步方法，取消频道搜索。应用在捕获到搜索失败(org.ngb.toolkit.channelscan.ChannelScanFailureEvent)或搜索结束(org.ngb.toolkit.channelscan.ChannelScanFinishEvent)事件之前，可以通过调用该方法取消本次频道搜索。成功取消搜索后，将向应用发送org.ngb.toolkit.channelscan.ChannelScanFinishEvent事件。若频道搜索未启动，调用本方法不执行任何动作。

参数：无。

返回：无。

H.2.2.2.7 release

原型：public void release()

描述：指示系统释放搜索引擎所使用的资源。若搜索引擎正在运行，必须先取消搜索，然后才能调用本方法。

参数：无。

返回：无。

H.2.2.2.8 saveScanResult

原型：public boolean saveScanResult()

描述：将本次搜索结果保存到NVM中。

返回：boolean型，取值true表示保存成功，false表示保存失败。

H.2.2.2.9 saveServicesInfo

原型：public boolean saveServicesInfo(java.lang.String jsonSIInfo)

描述：将JSON格式的节目信息数据传入DTV组件进行解析保存。

由于仅仅是数据解析，不消耗太多时间，所以采用同步模式。

参数：jsonSIInfo- 应用从运营商前端ip方式获取的JSON格式的PSI/SI信息。

返回：boolean型，取值true表示解析和保存成功，false表示数据格式解析失败

H.2.3 事件org.ngb.toolkit.channelscan.ChannelScanEvent

原型：public abstract class org.ngb.toolkit.channelscan.ChannelScanEvent

extends java.util.EventObject

描述：搜台事件的基类。

H.2.4 事件org.ngb.toolkit.channelscan.ChannelScanFailureEvent

原型：public class org.ngb.toolkit.channelscan.ChannelScanFailureEvent
extends org.ngb.toolkit.channelscan.ChannelScanEvent

描述：搜台过程失败事件，应用可以通过getReason()方法得到搜索失败的具体原因。

H.2.4.1 常量域——搜台失败原因

H.2.4.1.1 REASON_UNKOWN

原型：public static int REASON_UNKOWN = 0

描述：搜台失败原因——未知原因。

H.2.4.1.2 REASON_TUNE_LOCK_FAILED

原型：public static int REASON_TUNE_LOCK_FAILED = 1

描述：搜台失败原因——锁频失败。

H.2.4.1.3 REASON_NIT_SEARCH_FAILED

原型：public static int REASON_NIT_SEARCH_FAILED = 2

描述：搜台失败原因——NIT搜索失败。

H.2.4.1.4 REASON_BAT_SEARCH_FAILED

原型：public static int REASON_BAT_SEARCH_FAILED = 3

描述：搜台失败原因——BAT搜索失败。

H.2.4.1.5 REASON_PAT_SEARCH_FAILED

原型：public static int REASON_PAT_SEARCH_FAILED = 4

描述：搜台失败原因——PAT搜索失败。

H.2.4.1.6 REASON_PMT_SEARCH_FAILED

原型：public static int REASON_PMT_SEARCH_FAILED = 5

描述：搜台失败原因——PMT搜索失败。

H.2.4.2 方法

H.2.4.2.1 getReason

原型：public int getReason()

描述：获取搜索失败的原因。

返回：int型，表示失败的具体原因，取值详见
org.ngb.toolkit.channelscan.ChannelScanFailureEvent类搜台失败原因常量域定义。

H.2.5 事件org.ngb.toolkit.channelscan.ChannelScanFinishEvent

原型：public class org.ngb.toolkit.channelscan.ChannelScanFinishEvent

extends org.ngb.toolkit.channelscan.ChannelScanEvent
描述：搜索过程结束事件。

H.2.5.1 方法

H.2.5.1.1 getServiceCount

原型：public int getServiceCount()
描述：获取搜索得到的业务个数。
参数：无。
返回：int型，表示业务个数。

H.2.5.1.2 getTransportStreamCount

原型：public int getTransportStreamCount()
描述：获取搜索得到的传输流个数。
参数：无。
返回：int型，表示传输流个数。

H.2.6 事件org.ngb.toolkit.channelscan.ChannelScanNITSuccessEvent

原型：public class org.ngb.toolkit.channelscan.ChannelScanNITSuccessEvent
extends org.ngb.toolkit.channelscan.ChannelScanEvent
描述：搜台成功解析NIT事件。

H.2.6.1 方法

H.2.6.1.1 getTransportStream

原型：public org.ngb.broadcast.dvb.si.SITransportStream[] getTransportStream()
描述：获取NIT表中描述的传输流对象。
参数：无。
返回：org.ngb.broadcast.dvb.si.TransportStream对象数组，表示NIT表中描述的传输流对象。

H.2.7 事件org.ngb.toolkit.channelscan.ChannelScanSuccessEvent

原型：public class org.ngb.toolkit.channelscan.ChannelScanSuccessEvent
extends org.ngb.toolkit.channelscan.ChannelScanEvent
描述：频道搜索成功事件。应用可以通过getResult()方法获得成功搜索到的业务对象。

H.2.7.1 方法

H.2.7.1.1 getResult

原型：public org.ngb.broadcast.dvb.si.SIService[] getResult()
描述：获取搜索结果。
参数：无。
返回：org.ngb.broadcast.dvb.si.SIService对象数组，表示搜索结果。

H.3 电子节目指南模块

电子节目指南（EPG）为终端用户提供了浏览广播业务相关信息的方法，例如业务名称、节目起止时间、内容梗概等，便于终端用户对业务进行快速检索和访问。

电子节目指南模块提供了获取 EPG 信息的类和方法。EPG 信息获取可采用缓存（Cache）机制，也可以在需要时临时装载。若采用缓存机制，应对 EPG 信息进行实时监控，以保证应用能提取到最新 EPG 信息。

电子节目指南模块概要见表 H.2。

表 H.2　电子节目指南模块概要

接口	
ProgramEvent	描述了某个节目事件信息。
ProgramEventFilter	定义了应用从 EPG 模块查询节目事件信息所用的过滤器接口，由应用层实现。
ProgramService	描述了某个节目业务信息，是一组属于同一个业务的节目事件信息的封装。
ProgramServiceFilter	定义了应用从 EPG 模块查询节目业务信息所用的过滤器接口，由应用层实现。
EPGUpdateListener	EPG 信息更新事件监听器，由应用程序实现。
类	
EPGManager	EPG 管理器，EPG 信息获取的入口类。
事件	
EPGUpdateEvent	EPG 信息更新事件。

H.3.1　接口org.ngb.toolkit.epg.ProgramEvent

原型：public interface org.ngb.toolkit.epg.ProgramEvent

描述：节目事件信息描述类。

H.3.1.1　方法

H.3.1.1.1　getTitle

原型：public java.lang.String getTitle()

描述：获取节目事件名称。

参数：无。

返回：java.lang.String 对象，表示节目事件名称。

H.3.1.1.2　getShortDescription

原型：public java.lang.String getShortDescription()

描述：获取节目事件简介。

参数：无。

返回：java.lang.String 对象，表示节目事件简介。

H.3.1.1.3　getBeginDate

原型：public java.util.Date getBeginDate()

描述：获取节目事件的起始时间。

参数：无。

返回：java.util.Date 对象，表示节目事件起始时间。

H.3.1.1.4　getDuration

原型：public long getDuration()
描述：获取节目事件的时间长度。
参数：无。
返回：long 型，表示节目事件时长，单位为秒。

H.3.1.1.5 getEndDate

原型：public java.util.Date getEndDate()
描述：获取节目事件的结束时间。
参数：无。
返回：java.util.Date 对象，表示节目事件结束时间。

H.3.1.1.6 getLanguageCode

原型：public java.lang.String getLanguageCode()
描述：获取节目事件描述信息的编码语种。
参数：无。
返回：java.lang.String 对象，表示节目事件描述信息语种，语种三字母代码遵循 GB/T 4880.2—2000。

H.3.1.1.7 getNibbles

原型：public byte[] getNibbles()
描述：获取节目事件内容分类信息。
参数：无。
返回：byte型数组，表示与本节目事件关联的内容分类信息。
——byte[0] - 高4位表示二级内容分类（content_nibble_level_2）；低4位表示一级内容分类（content_nibble_level_1）；
——byte[1] - 高4位表示二级自定义分类（user_nibble_level_2）；低4位表示一级自定义分类（user_nibble_level_1）。

H.3.1.1.8 getCALockMode

原型：public boolean getCALockMode()
描述：判断该节目事件是否需要 CA 授权。
注：如何识别节目事件是否需要 CA 授权，由系统内部调用 CA 相关接口自行实现，对实现方式不作强制规定。
参数：无。
返回：boolean 类型，取值 true 表示 CA 未授权节目，false 表示 CA 授权节目。

H.3.1.1.9 getDvbLocator

原型：public org.davic.net.dvb.DvbNetworkBoundLocator getDvbLocator()
描述：获取节目事件的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator 对象，表示节目事件的定位符。

H.3.1.1.10 getProgramService

原型：public org.ngb.toolkit.epg.ProgramService getProgramService()
描述：获取本节目事件所属的节目业务对象。

参数：无。

返回：org.ngb.toolkit.epg.ProgramService 对象，表示本节目事件所属的节目业务对象。

H.3.2 接口org.ngb.toolkit.epg.ProgramEventFilter

原型：public interface org.ngb.toolkit.epg.ProgramEventFilter

描述：EPG 节目事件查询过滤器接口，由应用层实现。

H.3.2.1 方法

H.3.2.1.1 accept

原型：public boolean accept(org.ngb.toolkit.epg.ProgramEvent programEvent)

描述：判断节目事件是否满足过滤条件，过滤行为由应用自行实现。

参数：org.ngb.toolkit.epg.programEvent – org.ngb.toolkit.epg.ProgramEvent对象，表示待过滤的节目事件。

返回：boolean型，取值true表示由参数org.ngb.toolkit.epg.programEvent指定的节目事件满足过滤条件，取值false表示不满足过滤条件。

H.3.3 接口org.ngb.toolkit.epg.ProgramService

原型：public interface org.ngb.toolkit.epg.ProgramService

描述：节目业务信息，表示一组属于同一个业务的节目事件（org.ngb.toolkit.epg.ProgramEvent）的封装。节目业务所包含的节目事件属于同一个业务，且因不同的查询条件而包含不同的节目事件。通过 org.ngb.toolkit.epg.ProgramService 接口提供的方法可以获取节目事件信息。

H.3.3.1 常量域——业务类型

H.3.3.1.1 SERVICE_TYPE_RESERVED

原型：public static final short SERVICE_TYPE_RESERVED = 0

描述：业务类型——预留使用。

H.3.3.1.2 SERVICE_TYPE_DIGITAL_TELEVISION

原型：public static final short SERVICE_TYPE_DIGITAL_TELEVISION = 1

描述：业务类型——数字电视广播业务。

H.3.3.1.3 SERVICE_TYPE_DIGITAL_RADIO_SOUND

原型：public static final short SERVICE_TYPE_DIGITAL_RADIO_SOUND = 2

描述：业务类型——数字声音广播业务。

H.3.3.1.4 SERVICE_TYPE_TELETEXT

原型：public static final short SERVICE_TYPE_TELETEXT = 3

描述：业务类型——图文电视业务。

H.3.3.1.5 SERVICE_TYPE_NVOD_REFERENCE

原型：public static final short SERVICE_TYPE_NVOD_REFERENCE = 4

描述：业务类型——NVOD 参考业务。

H.3.3.1.6 SERVICE_TYPE_NVOD_TIME_SHIFTED

原型：public static final short SERVICE_TYPE_NVOD_TIME_SHIFTED = 5
描述：业务类型——NVOD 时移业务。

H.3.3.1.7 SERVICE_TYPE_MOSAIC

原型：public static final short SERVICE_TYPE_MOSAIC = 6
描述：业务类型——马赛克业务。

H.3.3.1.8 SERVICE_TYPE_PAL

原型：public static final short SERVICE_TYPE_PAL = 7
描述：业务类型——PAL 编码信号。

H.3.3.1.9 SERVICE_TYPE_SECAM

原型：public static final short SERVICE_TYPE_SECAM = 8
描述：业务类型——SECAM 编码信号。

H.3.3.1.10 SERVICE_TYPE_D_D2_MAC

原型：public static final short SERVICE_TYPE_D_D2_MAC = 9
描述：业务类型——D/D2-MAC。

H.3.3.1.11 SERVICE_TYPE__FM_RADIO

原型：public static final short SERVICE_TYPE_FM_RADIO = 10
描述：业务类型——FM 无线。

H.3.3.1.12 SERVICE_TYPE__NTSC

原型：public static final short SERVICE_TYPE_NTSC = 11
描述：业务类型——NTSC 编码信号。

H.3.3.1.13 SERVICE_TYPE_DATA_BROADCAST

原型：public static final short SERVICE_TYPE_DATA_BROADCAST = 12
描述：业务类型——数据广播业务。

H.3.3.1.14 SERVICE_TYPE_RESERVED_FOR_CI

原型：public static final short SERVICE_TYPE_RESERVED_FOR_CI = 13
描述：业务类型——reserved for Common Interface Usage。

H.3.3.1.15 SERVICE_TYPE_RCS_MAP

原型：public static final short SERVICE_TYPE_RCS_MAP = 14
描述：业务类型——RCS Map。

H.3.3.1.16 SERVICE_TYPE_RCS_FLS

原型：public static final short SERVICE_TYPE_RCS_FLS = 15
描述：业务类型——RCS FLS。

H.3.3.1.17 SERVICE_TYPE_DVB_MHP

原型：public static final short SERVICE_TYPE_DVB_MHP = 16
描述：业务类型——DVB MHP service。

H.3.3.2 方法

H.3.3.2.1 getDvbLocator

原型：public org.davic.net.dvb.DvbNetworkBoundLocator getDvbLocator()
描述：获取该节目业务对象所对应的业务的定位符。
参数：无。
返回： org.davic.net.dvb.DvbNetworkBoundLocator 对象，表示业务定位符。

H.3.3.2.2 getNetworkID

原型：public int getNetworkID()
描述：获取该节目业务对象所属网络的网络 ID。
参数：无。
返回：int 型，表示节目业务对象所属网络的网络 ID。

H.3.3.2.3 getServiceName

原型：public java.lang.String getServiceName()
描述：获取该节目业务对象所对应的业务的名称。
参数：无。
返回：java.lang.String 对象，表示业务名称。

H.3.3.2.4 getServiceLogicNumber

原型：public int getServiceLogicNumber()
描述：获取该节目业务对象所对应的业务的逻辑频道号。
注：逻辑频道号的获取方式由系统自行决定。
参数：无。
返回：int 型，表示业务逻辑频道号。

H.3.3.2.5 getServiceType

原型：public int getServiceType()
描述：获取该节目业务对象所对应的业务的类型。
参数：无。
返回：int 型，表示业务类型，取值详见 org.ngb.toolkit.epg.ProgramService 接口的“业务类型”常量域定义。

H.3.3.2.6 getCAFreeMode

原型：public boolean getCAFreeMode()
描述：获取该节目业务对象所对应的业务是否加扰。
参数：无。
返回：boolean 型，表示业务是否加扰，取值 true 表示加扰，false 表示未加扰。

H.3.3.2.7 getPresentProgramEvent

原型：public org.ngb.toolkit.epg.ProgramEvent getPresentProgramEvent()
描述：获取该节目业务对象所对应的业务的当前节目事件。
返回：org.ngb.toolkit.epg.ProgramEvent 对象，表示当前节目事件。

H.3.3.2.8 getFollowingProgramEvent

原型：public ProgramEvent getFollowingProgramEvent()
描述：获取该节目业务对象所对应的业务的后续节目事件。
返回：org.ngb.toolkit.epg.ProgramEvent 对象，表示后续节目事件。

H.3.3.2.9 getProgramEvents

原型：public org.ngb.toolkit.epg.ProgramEvent[] getProgramEvents
(ProgramEventFilter filter)
描述：根据指定的过滤器获取该节目业务对象所包含的节目事件
（org.ngb.toolkit.epg.ProgramEvent）对象。
参数：filter – org.ngb.toolkit.epg.ProgramEventFilter对象，表示节目事件过滤器。若参数为null，则表示获取所有节目事件。
返回：org.ngb.toolkit.epg.ProgramEvent对象数组，表示该节目业务对象所包含的节目事件。若无符合过滤条件的节目事件对象，则返回的数组长度为0。

H.3.3.2.10 getEPGManager

原型：public org.ngb.toolkit.epg.EPGManager getEPGManger()
描述：获取本节目业务对象所属的 org.ngb.toolkit.epg.EPGManager 实例。
参数：无。
返回：org.ngb.toolkit.epg.EPGManager 对象，表示本节目业务对象
所属的 org.ngb.toolkit.epg.EPGManager 实例。

H.3.4 接口org.ngb.toolkit.epg.ProgramServiceFilter

原型：public interface org.ngb.toolkit.epg.ProgramServiceFilter
描述：EPG 节目业务查询过滤器接口，由应用层实现。

H.3.4.1 方法

H.3.4.1.1 accept

原型：public boolean accept(org.ngb.toolkit.epg.ProgramService programService)
描述：判断节目业务是否满足过滤条件，过滤行为由应用实现。
参数：org.ngb.toolkit.epg.programService – org.ngb.toolkit.epg.ProgramService对象，表示待过滤的节目业务对象。
返回：boolean型，表示判断结果，取值true表示满足过滤条件，false表示不满足过 .....
滤条件。

H.3.5 接口org.ngb.toolkit.epg.EPGUpdateListener

原型：public interface org.ngb.toolkit.epg.EPGUpdateListener

extends java.util.EventListener

描述：EPG信息更新事件监听器，由应用程序实现。

H.3.5.1 方法

H.3.5.1.1 onUpdate

原型：public void onUpdate(org.ngb.toolkit.epg.EPGUpdateEvent event)

描述：EPG信息更新处理方法。

参数：event - org.ngb.toolkit.epg.EPGUpdateEvent对象，表示EPG信息更新事件。

返回：无。

H.3.6 类org.ngb.toolkit.epg.EPGManager

原型：public class org.ngb.toolkit.epg.EPGManager

描述：EPG 信息管理器，提供了获取 EPG 信息的方法，是 epg 单元的入口类。

H.3.6.1 方法

H.3.6.1.1 getEPGManager

原型：public static org.ngb.toolkit.epg.EPGManager[] getEPGManager()

描述：获取 EPG 管理器实例。接收终端可能有多个广播网络接口（例如同时连接有线网络和国标地面无线网络），每个网络接口对应一个 EPG 管理器实例，若有多个网络，本方法则返回多个 EPG 管理器实例。

返回：EPGManager 对象数组，表示 EPG 管理器实例，在只有一个网络接口的系统中，数组的长度为 1。

注：系统实现时，需要考虑下列特殊场景：

——场景 1 - 同一个物理网络接口连接多个广播网络，例如地面无线 Tuner 可能会接入到不同运营商的无线网络，在该场景下，只对应一个 org.ngb.toolkit.epg.EPGManager 实例；

——场景 2 - 多个物理网络接口连接同一个广播网络，例如具备 PVR 功能的接收机有两个以上 Tuner，同时接入到同一个网络，在此场景下，对应多个 EPGManager 实例。

H.3.6.1.2 getNetworkIDs

原型：public java.lang.Integer[] getNetworkIDs()

描述：获取本 org.ngb.toolkit.epg.EPGManager 实例所对应网络的网络 ID。

参数：无。

返回：Integer 对象数组，表示本 org.ngb.toolkit.epg.EPGManager 实例所对应网络的网络 ID。

注：在同一个物理网络接口连接多个广播网络的场景下会返回数组。

H.3.6.1.3 getNetworkName

原型：public java.lang.String getNetworkName(int networkId)

描述：获取由参数 networkId 指定的本 EPGManager 实例所对应网络的名称。

参数：networkId - int 型，表示网络标识。

返回：java.lang.String 对象，表示由参数 networkId 指定的本 org.ngb.toolkit.epg.EPGManager 实例所对应网络的名称。

H.3.6.1.4 addUpdateListener

原型：public void addUpdateListener(org.ngb.toolkit.epg.EPGUpdateListener listener)
描述：注册一个EPG更新事件监听器。
参数：listener - org.ngb.toolkit.epg.EPGUpdateListener对象，表示待注册的EPG更新事件监听器。
返回：无。

H.3.6.1.5 removeUpdateListener

原型：public void removeUpdateListener(org.ngb.toolkit.epg.EPGUpdateListener listener)
描述：注销一个EPG更新事件监听器。
参数：listener - org.ngb.toolkit.epg.EPGUpdateListener对象，表示待注销的EPG更新事件监听器。
返回：无。

H.3.6.1.6 getService

原型：public org.ngb.toolkit.epg.ProgramService getService(int networkId, int logicNumber)
throw Java.security.InvalidParameterException
描述：根据业务逻辑频道号获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。
参数：networkId - int 型，表示广播业务所属网络的网络标识；
logicNumber - int 型，表示广播业务的逻辑频道号。
返回：org.ngb.toolkit.epg.ProgramService 对象，表示一个节目业务。
异常：java.security.InvalidParameterException - 若由参数 networkId 和 logicNumber 指定的 org.ngb.toolkit.epg.ProgramService 对象不存在,则抛出此异常。

H.3.6.1.7 getService

原型：public org.ngb.toolkit.epg.ProgramService getService
(org.davic.net.dvb.DvbNetworkBoundLocator serviceLocator)
throw java.security.InvalidParameterException
描述：根据 DVB 定位符获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。
参数：serviceLocator - org.davic.net.dvb.DvbNetworkBoundLocator 对象，表示业务定位符。
返回：org.ngb.toolkit.epg.ProgramService 对象，表示一个节目业务。
异常：java.security.InvalidParameterException - 若参数 serviceLocator 无效，则抛出此异常。

H.3.6.1.8 getServices

原型：public org.ngb.toolkit.epg.ProgramService[] getServices(ProgramServiceFilter filter)
throw java.security.InvalidParameterException
描述：根据指定的过滤器获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。
参数：filter - org.ngb.toolkit.epg.ProgramServiceFilter 对象，表示节目业务过滤器。
返回：org.ngb.toolkit.epg.ProgramService 对象数组，表示符合过滤条件的节目业务；若无符合过滤条件的 org.ngb.toolkit.epg.ProgramService 对象，则返回的数组长度为 0。
异常：java.security.InvalidParameterException - 若参数 filter 为 null，则抛出此异常。

H.3.6.1.9 getPresentFollowingEvent

原型：public ProgramService getPresentFollowingEvent(DvbNetworkBoundLocator locator)

throws java.security.InvalidParameterException

描述：同步方法，获取指定业务的PF信息，不搜索Schedule。返回值为org.ngb.toolkit.epg.ProgramServic对象，NVOD参考业务可能有多个PF，，该对象org.ngb.toolkit.epg.getProgramEvents方法应根据时间顺序返回参考业务的PF信息，即：P[0]F[0]P[1]F[1] … P[n]F[n]。

——若成功检索到符合条件的EPG PF信息，将返回org.ngb.toolkit.epg.ProgramService对象，通过org.ngb.toolkit.epg.ProgramService对象的方法，进一步可以获取节目事件（org.ngb.toolkit.epg.ProgramEvent）对象；

——若由参数locator指定的业务在本EPGManager实例所关联的当前传送流上时，节目事件从流中获取还是从缓存中获取由系统自行决定，但应保证本接口获取到的节目事件是最新的；

——若由参数locator指定的业务不在本EPGManager实例所关联的当前传送流上时，节目事件应该从缓存中获取。

参数：locator – org.davic.net.dvb.DvbNetworkBoundLocator对象，表示业务的定位符；

返回：org.ngb.toolkit.epg.ProgramService对象。

异常：java.security.InvalidParameterException– 若locator参数无法定位一个业务，则抛出此异常。

H.3.6.1.10 getProgramService

原型：public org.ngb.toolkit.epg.ProgramService getProgramService
(org.davic.net.dvb.DvbNetworkBoundLocator locator,
java.util.Date beginDate, java.util.Date endDate)
throws java.security.InvalidParameterException, InvalidPeriodException

描述：同步方法，获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。该节目业务对象所包含的节目事件起止时间由参数beginDate和endDate确定。

——若成功检索到符合条件的EPG信息，将返回org.ngb.toolkit.epg.ProgramService对象，通过org.ngb.toolkit.epg.ProgramService对象的方法，进一步可以获取节目事件（org.ngb.toolkit.epg.ProgramEvent）对象；

——若由参数locator指定的业务在本org.ngb.toolkit.epg.EPGManager实例所关联的当前传送流上时，节目事件从流中获取还是从缓存中获取由系统自行决定，但应保证本接口获取到的节目事件是最新的；

——若由参数locator指定的业务不在本org.ngb.toolkit.epg.EPGManager实例所关联的当前传送流上时，节目事件应该从缓存中获取。

参数：locator – org.davic.net.dvb.DvbNetworkBoundLocator对象，表示业务的定位符；
beginDate – java.util.Date对象，表示该节目业务对象所包含的节目事件的起始时间；
endDate – java.util.Date对象，表示该节目业务对象所包含的节目事件的结束时间。

返回：org.ngb.toolkit.epg.ProgramService对象。

异常：java.security.InvalidParameterException– 若locator参数无法定位一个业务，则抛出此异常。
org.ngb.broadcast.dvb.si.InvalidPeriodException – 若指定的起止时间无效，则抛出此异常。

H.3.6.1.11 getProgramService

原型：public org.ngb.toolkit.epg.ProgramService getProgramService(
org.davic.net.dvb.DvbNetworkBoundLocator locator,java.util.Date beginDate,
int count, boolean isForward)
throws java.security.InvalidParameterException

描述：同步方法，获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。该节目业务对象所包含的节目事件起止时间由参数beginDate、count和isForward确定。

——若成功检索到符合条件的EPG信息，将返回org.ngb.toolkit.epg.ProgramService对象，通过org.ngb.toolkit.epg.ProgramService对象的方法，进一步可以获取节目事件（org.ngb.toolkit.epg.ProgramEvent）对象；

——若指定的业务在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件从流中获取还是从缓存中获取由实现者自行决定，但应保证本接口获取到的节目事件是最新的；

——若指定的业务不在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件应该从缓存中获取。

参数：locator - org.davic.net.dvb.DvbNetworkBoundLocator对象，表示指定业务的定位符；
beginDate – java.util.Date对象，表示节目事件的起始时间；
count - int型，表示要搜索的节目事件数量；
isForward - boolean型，取值true表示从指定时刻开始往后获取节目事件；false表示从指定时刻开始往前获取节目事件。

返回：org.ngb.toolkit.epg.ProgramService对象。

异常：java.security.InvalidParameterException– 若locator参数无法定位一个业务，则抛出此异常。

H.3.6.1.12 getProgramServices

原型：public org.ngb.toolkit.epg.ProgramService getProgramServices
(org.ngb.toolkit.epg.ProgramServiceFilter filter)

描述：同步方法，根据过滤器获取节目业务对象。该节目业务对象将包含所有可用的节目事件对象。

——若成功检索到符合条件的EPG信息，将返回org.ngb.toolkit.epg.ProgramService对象数组，从该对象数组可以获取检索到的节目事件（org.ngb.toolkit.epg.ProgramEvent）对象；

——若指定的业务在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件从流中获取还是从缓存中获取由实现者自行决定，但应保证本接口获取到的节目事件是最新的；

——若指定的业务不在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件应该从缓存中获取。

参数：filter - org.ngb.toolkit.epg.ProgramServiceFilter对象，应用实现的过滤器对象。

返回：org.ngb.toolkit.epg.ProgramService对象。

H.3.6.1.13 getProgramServices

原型：public org.ngb.toolkit.epg.ProgramService getProgramServices
(org.ngb.toolkit.epg.ProgramServiceFilter filter,
java.util.Date beginDate, java.util.Date endDate)
throws org.ngb.broadcast.dvb.si.InvalidPeriodException

描述：异步方法，根据过滤器获取节目业务（org.ngb.toolkit.epg.ProgramService）对象。节目业务对象所包含的节目事件起止时间由参数beginDate和endDate确定。

——若成功检索的符合条件的EPG信息，将返回org.ngb.toolkit.epg.ProgramService对象数组，从该对象数组可以获取检索到的节目事件（org.ngb.toolkit.epg.ProgramEvent）对象；

——若指定的业务在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件从流中获取还是从缓存中获取由实现者自行决定，但应保证本接口获取到的节目事件是最新的；

——若指定的业务不在本org.ngb.toolkit.epg.EPGManager实例关联的当前传送流上时，节目事件应该从缓存中获取。

参数：filter - org.ngb.toolkit.epg.ProgramServiceFilter对象，应用实现的过滤器对象；
beginDate – java.util.Date对象，表示节目事件的起始时间；
endDate – java.util.Date对象，表示节目事件的结束时间。

返回：org.ngb.toolkit.epg.ProgramService对象。

异常：org.ngb.broadcast.dvb.si.InvalidPeriodException – 若指定的时间范围无效，则抛出此异常。

H.3.7 事件org.ngb.toolkit.epg.EPGUpdateEvent

原型：public class org.ngb.toolkit.epg.EPGUpdateEvent extends java.util.EventObject

描述：EPG信息变化事件。

H.3.7.1 方法

H.3.7.1.1 getResult

原型：public org.ngb.toolkit.epg.ProgramService[] getResult()

描述：获取发生更新的节目业务对象。

参数：无。

返回：org.ngb.toolkit.epg.ProgramService对象数组，表示发生更新的节目业务。

H.3.7.1.2 EPGUpdateEvent

原型：protected EPGUpdateEvent(Object object, org.ngb.toolkit.epg.ProgramService[] result)

描述：构造方法，创建一个EPGUpdateEvent对象。不对应用层显露。

参数：无。

返回： org.ngb.toolkit.epg.EPGUpdateEvent对象，表示本节目业务
对象所属的org.ngb.toolkit.epg.EPGUpdateEvent实例。

H.4 信息搜索模块

信息搜索模块提供了与全局搜索和匹配搜索相关的类和方法。“全局搜索”和“匹配搜索”的术语定义如下：

全局搜索 - 根据用户设置的搜索条件查找SI、PVR等内容，并返回有意义的搜索结果，用以提升用户体验；

匹配搜索 - 根据用户输入给出目前数据源中可以匹配的字符串，以缩短用户输入查询关键字的时间和降低用户输入查询关键字的难度。

信息搜索模块分为以下组件：

——搜索管理器(org.ngb.toolkit.search.SearchManager) - 信息搜索模块的入口类；

——全局搜索会话(org.ngb.toolkit.search.GlobalSearchSession) - 与全局搜索过程相关联的会话；

——匹配搜索会话(org.ngb.toolkit.search.AutoCompleteSearchSession) - 与正在进行的匹配搜索过程相关联的会话。

信息搜索模块概要见表 H.3。

表 H.3 信息搜索模块概要

接口	
AutoCompleteSearchListener	匹配搜索事件监听器，由应用程序实现。
AutoCompleteSearchResultItem	描述了匹配搜索结果，提供了获取匹配搜索结果各种信息的方法。
AutoCompleteSearchResultList	描述了匹配搜索结果列表对象，提供了存取匹配搜索结果条目的方法。
AutoCompleteSearchSession	描述了一个匹配搜索会话，提供了匹配搜索会话控制方法。
GlobalSearchListener	全局搜索事件监听器，由应用程序实现。
GlobalSearchResultItem	描述了一个全局搜索结果对象，提供了访问搜索结果相关信息的方法。
GlobalSearchResultList	描述了全局搜索结果列表对象，提供了获取全局搜索结果条目的方法。
GlobalSearchSession	描述了一个全局搜索会话，提供了全局搜索会话控制方法。
RetrieveDirection	定义了搜索结果查找方向常量。
SearchContentType	内容类型常量定义。
SearchCriteriaFlags	过滤条件标记定义。
SearchFields	提供了设置搜索字段的接口。
SearchHistoryItem	描述了一条搜索历史记录，提供了获取搜索历史记录各种信息的方法。
SearchHistoryList	描述了搜索历史记录列表，提供了遍历搜索历史记录列表的功能。
SearchStatus	定义了搜索状态常量。
SourceType	搜索数据源常量定义。
类	
AutoCompleteSearchFilter	描述了一个自动匹配搜索过滤器，提供了匹配搜索过滤条件设置和获取方法。
GlobalSearchFilter	描述了一个全局搜索过滤器，提供了全局搜索过滤条件设置和获取方法。
SearchManager	全局搜索和匹配搜索的搜索管理器，是搜索功能模块的入口类。
SortCriteria	定义了排序常量和排序方法。
异常	
SearchException	描述了全局搜索出现错误时的异常。

H.4.1 接口org.ngb.toolkit.search.AutoCompleteSearchListener

原型：public interface org.ngb.toolkit.search.AutoCompleteSearchListener

描述：匹配搜索事件监听器，由应用程序实现。搜索引擎向应用程序通知的匹配搜索事件包括：

——搜索会话开始onAutoCompleteSearchStart(AutoCompleteSearchSession, int)；

——搜索会话结束onAutoCompleteSearchStop(AutoCompleteSearchSession, int)；

——搜索会话销毁onAutoCompleteSearchDestroy(AutoCompleteSearchSession, int)；
——搜索会话错误onAutoCompleteSearchError(AutoCompleteSearchSession, int)。

H.4.1.1 方法

H.4.1.1.1 onAutoCompleteSearchStart

原型：void onAutoCompleteSearchStart
(org.ngb.toolkit.search.AutoCompleteSearchSession searchSession, int status)
描述：调用AutoCompleteSearchSession对象的startSearch()方法完成匹配搜索后的回调方法。
参数：searchSession - org.ngb.toolkit.search.AutoCompleteSearchSession对象，表示发出此事件的匹配搜索会话实例；
status - int型，表示当前匹配搜索状态，可取值为SearchStatus.COMPLETED。
返回：无。

H.4.1.1.2 onAutoCompleteSearchStop

原型：void onAutoCompleteSearchStop(org.ngb.toolkit.search.AutoCompleteSearchSession searchSession, int status)
描述：调用org.ngb.toolkit.search.AutoCompleteSearchSession对象的stopSearch()方法停止匹配搜索会话后的回调方法。
参数：searchSession - org.ngb.toolkit.search.AutoCompleteSearchSession对象，表示发出此事件的匹配搜索会话实例；
status - int型，表示当前匹配搜索状态，可取值为SearchStatus.STOP_SUCCESS。
返回：无。

H.4.1.1.3 onAutoCompleteSearchDestroy

原型：void onAutoCompleteSearchDestroy
(org.ngb.toolkit.search.AutoCompleteSearchSession searchSession, int status)
描述：调用AutoCompleteSearchSession对象的dispose()方法销毁匹配搜索会话后的回调方法。
参数：searchSession - org.ngb.toolkit.search.AutoCompleteSearchSession对象，表示发出此事件的匹配搜索会话实例；
status - int型，表示当前匹配搜索状态，可取值为SearchStatus.DISPOSE_SUCCESS。
返回：无。

H.4.1.1.4 onAutoCompleteSearchError

原型：void onAutoCompleteSearchError
(org.ngb.toolkit.search.AutoCompleteSearchSession searchSession, int status)
描述：匹配搜索会话出错时的回调方法。
参数：searchSession - org.ngb.toolkit.search.AutoCompleteSearchSession对象，表示发出此事件的匹配搜索会话实例；
status - int型，表示当前匹配搜索状态，可取值为：
——SearchStatus.FAILED；
——SearchStatus.TIMEOUT；
——SearchStatus.STOP_FAILED；

——SearchStatus.DISPOSE_FAILED。

返回：无。

H.4.2 接口org.ngb.toolkit.search.AutoCompleteSearchResultItem

原型：public interface org.ngb.toolkit.search.AutoCompleteSearchResultItem

描述：描述了一条匹配搜索结果，提供了获取匹配搜索结果各种信息的方法。

H.4.2.1 方法

H.4.2.1.1 getSource

原型：int getSource()

描述：获取匹配搜索结果的数据源。

返回：int型，表示匹配搜索结果的数据源，取值详见org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域定义。

H.4.2.1.2 getString

原型：java.lang.String getString()

描述：获取与用户输入的关键字相匹配的字符串。

参数：无。

返回：java.lang.String对象，表示与用户输入的关键字相匹配的字符串。

H.4.3 接口org.ngb.toolkit.search.AutoCompleteSearchResultList

原型：public interface org.ngb.toolkit.search.AutoCompleteSearchResultList extends java.util.ListIterator

描述：描述了匹配搜索结果列表对象，提供了存取匹配搜索结果条目的方法。

注：AutoCompleteSearchResultList是AutoCompleteSearchResultItem对象的集合。

H.4.4 接口org.ngb.toolkit.search.AutoCompleteSearchSession

原型：public interface org.ngb.toolkit.search.AutoCompleteSearchSession

描述：匹配搜索会话接口，提供了匹配搜索会话控制方法。

——应用程序调用 startSearch()方法开始匹配搜索会话；

——应用程序调用 stopSearch()方法停止匹配搜索会话；

——应用程序调用 dispose()方法销毁匹配搜索会话；

——应用程序调用 getSearchResultList()方法获取匹配搜索结果；

——搜索引擎调用 org.ngb.toolkit.search.AutoCompleteSearchListener 对象提供的回调方法来通知应用程序匹配搜索的状态。

示例：

```
// Application gets the AutoCompleteSearchFilter object and fill it with default parameters.
AutoCompleteSearchFilter acFilter = AutoCompleteSearchFilter.getAutoCompleteSearchFilter();
acFilter.setMaxResults(10);
acFilter.setSource(SourceType.ALL);
acFilter.setTimeLimit(250);
acFilter.setSearchField(SearchFields.TITLE);
acFilter.setSearchLanguage("zho");      //搜索中文
```

```
SearchManager searchManager = SearchManager.getInstance();
// Get the auto complete search object
AutoCompleteSearchSession acSearch = searchManager.getAutoCompleteSearchSession(acFilter, this);

// User entered some characters and waits for the Auto complete List to be displayed to them.
// EPG start the auto complete search
acSearch.startSearch(searchString);

// Engine start the search with middleware and then wait for notification when search is success.
// Once available, Engine notifies EPG with the search status.
// listener.onSearchStart(acSearch, SearchStatus.COMPLETED);

// EPG can now fetch the auto complete result list.
AutoCompleteSearchResultList acList = acSearch.getSearchResultList();

// The list is an iteration and hence EPG can iterate through the list fetching the elements.
// If at any instant of time, if EPG want to stop the on going search
// may be because user has entered another character before the previous search
// is completed or may be middleware is taking more time for search
acSearch.stopSearch();

// When the auto complete search object is no more in use, then
// Application should dispose() the search object to make sure that the search
// session is destroyed and all resources are freed properly.
acSearch.dispose();
```

H.4.4.1 方法

H.4.4.1.1 startSearch

原型：void startSearch(java.lang.String searchStr)
throws org.ngb.toolkit.search.SearchException, java.lang.IllegalArgumentException

描述：启动一次新的匹配搜索请求，一旦搜索完成，搜索引擎会通知应用程序获取搜索结果。启动搜索后，应用程序需要等待来自引擎的SearchStatus.COMPLETED通知，仅当收到通知后，应用程序才能成功地获取结果。

注：

——若AutoCompleteSearchFilter超时时间设置为0，只有当显式调用stopSearch()方法后，搜索才会停止。

——通过调用getSearchResultList()方法来获取AutoCompleteSearchResultList对象，通过调用stopSearch()方法终止前面一个还在进行中的搜索请求，若不再使用该匹配搜索会话，应调用dispose()方法关闭搜索会话并释放资源。

——在同一个时间内，只能有一个匹配搜索会话处于活动状态，在这个期间可以重复调用startSearch()和stopSearch()方法，若不再需要使用该匹配搜索会话，应调用dispose()方法关闭搜索会话并释放资源。

——搜索结果存放在同一个缓存中，一旦启动一个新的搜索，就不能再使用旧的AutoCompleteSearchResultList

对象来获取以前的搜索结果。

参数：searchStr - java.lang.String对象，表示用户输入的字符串。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若启动会话请求操作失败，抛出此异常；
java.lang.IllegalArgumentException - 若输入参数非法，则抛出此异常。

H.4.4.1.2 stopSearch

原型：void stopSearch() throws org.ngb.toolkit.search.SearchException

描述：终止之前已经启动的匹配搜索请求。

注1：该方法只是终止匹配搜索，但是匹配搜索会话仍然有效，相应的资源也没有被释放。在下面的场景中该方法应该被强制调用：

——若已经开始匹配搜索会话且在等待返回结果，在搜索引擎返回结果之前，用户输入了另外的字符，必须调用stopSearch()方法终止本次搜索；

——匹配搜索过滤器超时时间设置为0，这样会一直等待搜索结果，若没有相应的匹配结果，在启动另外一个搜索之前，必须调用stopSearch()方法终止本次搜索。

注2：若匹配搜索已经完成，可不用调用该方法。

参数：无。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若未启动匹配搜索或停止匹配搜索失败，则抛出此异常。

H.4.4.1.3 dispose

原型：void dispose() throws org.ngb.toolkit.search.SearchException

描述：销毁不再需要的匹配搜索会话对象。

注：在匹配搜索完成后，一旦不再需要该会话，必须调用该方法以释放相应的资源。

参数：无。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若未启动匹配搜索或销毁匹配搜索会话失败，则抛出此异常。

H.4.4.1.4 getSearchResultList

原型：AutoCompleteSearchResultList getSearchResultList()
throws org.ngb.toolkit.search.SearchException

描述：获取匹配搜索的结果。

注：AutoCompleteSearchListener对象在收到SearchStatus.COMPLETED的通知后才能调用该方法。搜索结果存放在同一个缓存中，一旦初始化一个新的匹配搜索，就不能再使用旧的AutoCompleteSearchResultList对象来获取以前的搜索结果。

返回：AutoCompleteSearchResultList对象，表示匹配搜索结果列表。

异常：org.ngb.toolkit.search.SearchException -
若org.ngb.toolkit.search.AutoCompleteSearchListener对象未
收到SearchStatus.COMPLETED的通知（即在匹配搜索未完成时），调用该方法，则抛出此异常。

H.4.5 接口org.ngb.toolkit.search.GlobalSearchListener

原型：public interface org.ngb.toolkit.search.GlobalSearchListener

描述：全局搜索事件监听器，由应用程序实现。搜索引擎向应用程序通知的匹配搜索事件包括：

——搜索会话开始 onGlobalSearchStart(GlobalSearchSession, int)；

——搜索会话结束 onGlobalSearchStop(GlobalSearchSession, int)；

——搜索会话关闭 onGlobalSearchDestroy(GlobalSearchSession, int)；

——搜索会话错误 onGlobalSearchError(GlobalSearchSession, int)；

——获取到局部数据 onGlobalSearchRetrieval(GlobalSearchSession, int)。

H.4.5.1 方法

H.4.5.1.1 onGlobalSearchStart

原型：void onGlobalSearchStart(org.ngb.toolkit.search.GlobalSearchSession searchSession, int status)

描述：调用org.ngb.toolkit.search.GlobalSearchSession对象的startSearch()方法完成全局搜索后的回调方法。

参数：searchSession - org.ngb.toolkit.search.GlobalSearchSession对象，表示发出此事件的全局搜索实例；

status - int型，表示当前全局搜索状态，可取值为：

——SearchStatus.COMPLETED；

——SearchStatus.IN_PROGRESS；

——SearchStatus.INITIATED。

返回：无。

H.4.5.1.2 onGlobalSearchStop

原型：void onGlobalSearchStop(org.ngb.toolkit.search.GlobalSearchSession searchSession, int status)

描述：调用org.ngb.toolkit.search.GlobalSearchSession对象的stopSearch()方法停止全局搜索后的回调方法。

参数：searchSession - org.ngb.toolkit.search.GlobalSearchSession对象，表示发出此事件的全局搜索会话实例；

status - int型，表示当前全局搜索状态，可取值为SearchStatus.STOP_SUCCESS。

返回：无。

H.4.5.1.3 onGlobalSearchDestroy

原型：void onGlobalSearchDestroy(org.ngb.toolkit.search.GlobalSearchSession searchSession, int status)

描述：调用GlobalSearchSession对象的dispose()方法销毁全局搜索会话后的回调方法。

参数：searchSession - org.ngb.toolkit.search.GlobalSearchSession对象，表示发出此事件的全局搜索会话实例；

status - int型，表示当前全局搜索状态，可取值为SearchStatus.DISPOSE_SUCCESS。

返回：无。

H.4.5.1.4 onGlobalSearchError

原型：void onGlobalSearchError(org.ngb.toolkit.search.GlobalSearchSession searchSession, int status)

描述：全局搜索会话出错的回调方法。

参数：searchSession - org.ngb.toolkit.search.GlobalSearchSession对象，表示发出此事件的全局搜索会话实例；

status - int型，表示全局搜索状态，可取值为：

——SearchStatus.FAILED；

——SearchStatus.INTERRUPTED；

——SearchStatus.TIMEOUT_STOP_FAILED；

——SearchStatus.TIMEOUT；

——SearchStatus.DISPOSE_FAILED；

——SearchStatus.RETRIEVAL_FAILED；

——SearchStatus.RETRIEVAL_INSUFFICIENT；

——SearchStatus.STOP_FAILED。

H.4.5.1.5 onGlobalSearchRetrieval

原型：void onGlobalSearchRetrieval(org.ngb.toolkit.search.GlobalSearchSession searchSession, int status)

描述：全局搜索获得部分搜索结果的回调方法。

参数：searchSession - org.ngb.toolkit.search.GlobalSearchSession对象，表示发出此事件的全局搜索会话实例；

status - int型，表示全局搜索状态，可取值为SearchStatus.RETRIEVAL_SUCCESS。

返回：无。

H.4.6 接口org.ngb.toolkit.search.GlobalSearchResultItem

原型：public interface org.ngb.toolkit.search.GlobalSearchResultItem

描述：描述了一条全局搜索结果，提供了访问搜索结果相关信息的方法。

H.4.6.1 方法

H.4.6.1.1 getContent

原型：java.lang.Object getContent()

描述：获取与此搜索结果相关联的对象。

参数：无。

返回：java.lang.Object对象，该对象可能为SIEvent类型。

注：应用应通过instanceof方法进一步判断返回对象类型。

H.4.7 接口org.ngb.toolkit.search.GlobalSearchResultList

原型：public interface org.ngb.toolkit.search.GlobalSearchResultList extends java.util.ListIterator

描述：描述了全局搜索结果列表对象，提供了获取全局搜索结果条目的方法。

注：org.ngb.toolkit.search.GlobalSearchResultList

是org.ngb.toolkit.search.GlobalSearchResultItem的集合。

H.4.8 接口org.ngb.toolkit.search.GlobalSearchSession

原型：public interface org.ngb.toolkit.search.GlobalSearchSession

描述：全局搜索会话接口，提供了全局搜索会话控制方法。

——应用程序调用 startSearch()方法开始全局搜索会话；

——应用程序调用 stopSearch()方法停止全局搜索会话；

——应用程序调用 dispose()方法来销毁全局搜索会话并释放资源；

——应用程序调用 retrievePage()方法在第一页、下一页和前一页之间导航；

——应用程序调用 getSearchResultList()方法获取搜索结果；

——搜索引擎调用 org.ngb.toolkit.search.GlobalSearchListener 对象提供的方法通知应用程序全局搜索状态。

注：应用程序通过调用 setPageSize()来设置页面大小，若未设置页面大小，则使用默认值 DEFAULT_PAGE_SIZE。

示例：

```
// If application want to filter the search results by providing advanced criteria,
// like if they want to filter on broadcast events, which will be broadcasted
// between 10 AM and 10 PM,
// with genre ALL, with parental rating as 13 and star rating between 2 - 5
// Application gets the GlobalSearchFilter object and fill it with required
// parameters.

GlobalSearchFilter gsFilter = GlobalSearchFilter.getGlobalSearchFilter();
gsFilter.setSource(BROADCAST);
gsFilter.setMaxResults(5);

// Fills the advanced criteria
gsFilter.setCategory(MAIN_CATEGORY_ALL, SUB_CATEGORY_ALL);

gsFilter.setTimeLimit(1500); // 1.5 secs
gsFilter.setSearchField(TITLE); // search only in title.

// Sort criteria
SortCriteria sortCriteria = new SortCriteria();
sortCriteria.setOrder(SORT_ORDER_ASCENDING);
sortCriteria.setType(SORT_TYPE_TITLE);

SearchManager searchManager = SearchManager.getInstance();
// Get the global search object.
GlobalSearchSession gsSearch = searchManager.getGlobalSearchSession(gsFilter, sortCriteria, this);

// start the global search
gsSearch.startSearch(searchString);

// Engine start the search with middleware and then wait for notification when
```

```
// search is success.
// Once available, Engine notifies EPG with the search status.
// listener.onGlobalSearchStart(gsSearch, SearchStatus.INITIATED);
// Application can update the UI indicating the initiated status.

// Engine notifies the in progress notification.
// listener.onGlobalSearchStart(gsSearch, SearchStatus.IN_PROGRESS);

// EPG can now fetch the call the retrieve API to fetch the results.
gsSearch.setPageSize(6);
gsSearch.retrievePage(FIRST_PAGE);

// Engine notifies the retrieve status.
//listener.onGlobalSearchRetrieve(gsSearch, SearchStatus.RETRIEVAL_SUCCESS);

// EPG can now display the result list.
GlobalSearchResultList gsList = gsSearch.getSearchResultList();

// The list is an iteration and hence EPG can iterate through the list fetching
// the elements.

// If at any instant of time, if EPG want to stop the on going search
// may be because user has pressed back key or explicitly stopped.
gsSearch.stopSearch();

// When the global search object is no more in use, then
// Application should dispose() the search object to make sure that the search
// session is destroyed and all resources are freed properly.
gsSearch.dispose();
```

H.4.8.1 常量域——默认业务大小

H.4.8.1.1 DEFAULT_PAGE_SIZE

原型：public static final int DEFAULT_PAGE_SIZE = 6
描述：全局搜索结果默认的页面大小，即每页所包含的搜索结果条目数量。

H.4.8.2 方法

H.4.8.2.1 startSearch

原型：void startSearch(java.lang.String searchStr)
throws org.ngb.toolkit.search.SearchException, java.lang.IllegalArgumentException
描述：启动一次新的全局搜索请求，一旦搜索完成，搜索引擎会通知应用程序获取搜索结果。启动搜索后，应用程序需要等待来自搜索引擎的通知，仅当收到通知后，应用程序才能成功获取结果。
注：

——若GlobalSearchFilter超时时间设置为0，只有当应用显式调用stopSearch()方法后，搜索才会停止。

——通过调用getSearchResultList()方法来获取GlobalSearchResultList对象，通过调用stopSearch()方法终止前面一个还在进行中的搜索请求，如果不再使用全局搜索会话，应调用dispose()方法来释放它。

——在同一个时间内只有一个搜索会话处于活动状态。搜索结果存放在同一个缓存中，一旦启动一个新的搜索，就不能再使用旧的GlobalSearchResultList对象来获取以前的搜索结果。

参数：searchStr - java.lang.String对象，表示用户输入的字符串。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若启动会话请求操作失败，则抛出此异常；
java.lang.IllegalArgumentException - 若输入参数非法，则抛出此异常。

H.4.8.2.2 stopSearch

原型：void stopSearch() throws org.ngb.toolkit.search.SearchException

描述：终止之前已经启动的全局搜索请求。

注：该操作只是终止全局搜索，但是全局搜索会话仍然有效，相应的资源也未被释放。

参数：无。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若未启动搜索或停止搜索失败，则抛出此异常。

H.4.8.2.3 dispose

原型：void dispose() throws org.ngb.toolkit.search.SearchException

描述：销毁不再需要的全局搜索会话对象。

注：在全局搜索完成后，一旦不再需要改会话，必须调用该方法以释放相应的资源。

参数：无。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若未启动搜索或销毁操作失败，则抛出此异常。

H.4.8.2.4 getPageSize

原型：int getPageSize()

描述：获取搜索返回结果每页条目数量大小。

参数：无。

返回：int型，表示当前设置的页面大小。

H.4.8.2.5 getResultCount

原型：int getResultCount() throws org.ngb.toolkit.search.SearchException

描述：获取搜索结果的元素数目。如果搜索还没有完成并且部分更新，结果数也将会根据搜索元素的改变而更新。对于每个阈值限制，应用程序将会收到SearchStatus.IN_PROGRESS的通知。

参数：无。

返回：int型，表示搜索结果的元素数目。

异常：org.ngb.toolkit.search.SearchException – 若搜索没有开始或者已经被终止，抛出此异常。

H.4.8.2.6 getSearchResultList

原型：org.ngb.toolkit.search.GlobalSearchResultList getSearchResultList()
throws org.ngb.toolkit.search.SearchException

描述：获取全局搜索结果。

注：在调用该方法之前，应用程序需要等候通知。该方法返回的列表只包含主内容列表。结果存放在同一个缓存中，一旦一个新的搜索被初始化，就不能再使用旧的GlobalSearchResultList对象来获取以前的搜索结果。

返回：org.ngb.toolkit.search.GlobalSearchResultList对象，表示全局搜索结果列表。

异常：org.ngb.toolkit.search.SearchException - 若搜索没有开始或者已经被终止，抛出此异常。

H.4.8.2.7 retrievePage

原型：void retrievePage(int retrieveDirection)
throws org.ngb.toolkit.search.SearchException, java.lang.IllegalArgumentException

描述：从搜索结果列表获取第一页、下一页或者前一页的数据。

参数：retrieveDirection - int型，用来指定获取第一页、前一页还是下一页的结果，可取值为NEXT_PAGE、PREVIOUS_PAGE或FIRST_PAGE，详见RetrieveDirection接口的“查找方向”常量域定义。

异常：org.ngb.toolkit.search.SearchException – 若发生任何失败操作，则抛出此异常；
java.lang.IllegalArgumentException - 若搜索方向不正确，则抛出此异常。

H.4.8.2.8 saveRecentSearchQuery

原型：void saveRecentSearchQuery() throws org.ngb.toolkit.search.SearchException

描述：保存最新的搜索结果。该方法只能在初始化一个搜索或者销毁对象之前调用。

参数：无。

返回：无。

异常：org.ngb.toolkit.search.SearchException - 若操作失败，抛出此异常。

H.4.8.2.9 setPageSize

原型：void setPageSize(int pageSize)
throws java.lang.IllegalArgumentException

描述：设置搜索返回结果每页条目数量大小。

若没有调用该方法设置页面大小，则默认的页面大小为DEFAULT_PAGE_SIZE。

参数：pageSize - int型，表示每页返回的结果数。

异常：java.lang.IllegalArgumentException - 若参数小于等于0或者超出系统支持的最大值，抛出此异常。

H.4.9 接口org.ngb.toolkit.search.RetrieveDirection

原型：public interface org.ngb.toolkit.search.RetrieveDirection

描述：定义了搜索结果查找方向常量。

H.4.9.1 常量域——查找方向

H.4.9.1.1 FIRST_PAGE

原型：public static final int FIRST_PAGE = 0

描述：查找方向——第一页。

H.4.9.1.2 NEXT_PAGE

原型：public static final int NEXT_PAGE = 1

描述：查找方向——下一页。

H.4.9.1.3 PREVIOUS_PAGE

原型：public static final int PREVIOUS_PAGE = 2
描述：查找方向——前一页。

H.4.10 接口org.ngb.toolkit.search.SearchContentType

原型：public interface org.ngb.toolkit.search.SearchContentType
描述：内容类型常量定义。

H.4.10.1 常量域——内容类型

H.4.10.1.1 ALL

原型：public static final int ALL = 0
描述：内容类型——音视频。

H.4.10.1.2 AUDIO_ONLY

原型：public static final int AUDIO_ONLY = 1
描述：内容类型——音频。

H.4.10.1.3 VIDEO_ONLY

原型：public static final int VIDEO_ONLY = 2
描述：内容类型——视频。

H.4.11 接口org.ngb.toolkit.search.SearchCriteriaFlags

原型：public interface org.ngb.toolkit.search.SearchCriteriaFlags
描述：过滤条件标记定义。

H.4.11.1 常量域——过滤条件

H.4.11.1.1 FLAG_NONE

原型：public static final int FLAG_NONE = 0
描述：过滤条件标记——空。

H.4.11.1.2 FLAG_SD_EVENT

原型：public static final int FLAG_SD_EVENT = 1
描述：过滤条件标记——过滤标清（2D）内容。

H.4.11.1.3 FLAG_HD_EVENT

原型：public static final int FLAG_HD_EVENT = 2
描述：过滤条件标记——过滤高清（2D）内容。

H.4.11.1.4 FLAG_3D_CONTENT

原型：public static final int FLAG_3D_CONTENT = 4

描述：过滤条件标记——过滤3D内容。

H.4.11.1.5 FLAG_CLEAR

原型：public static final int FLAG_CLEAR = 32
描述：过滤条件标记——过滤未加扰内容。

H.4.11.1.6 FLAG_SCRAMBLED

原型：public static final int FLAG_SCRAMBLED = 64
描述：过滤条件标记——过滤加扰内容。

H.4.12 接口org.ngb.toolkit.search.SearchFields

原型：public interface org.ngb.toolkit.search.SearchFields
描述：提供了设置搜索域的接口。

H.4.12.1 常量域——搜索域

H.4.12.1.1 ALL_STRING_FIELDS

原型：public static final int ALL_STRING_FIELDS = 0
描述：查找所有的信息。

H.4.12.1.2 SYNOPSIS

原型：public static final int SYNOPSIS = 1
描述：仅在简介中查找。

H.4.12.1.3 TITLE

原型：public static final int TITLE = 2
描述：仅在标题中查找。

H.4.13 接口org.ngb.toolkit.search.SearchHistoryItem

原型：public interface org.ngb.toolkit.search.SearchHistoryItem
描述：描述了一条搜索历史记录，提供了获取搜索历史记录各种信息的方法。

H.4.13.1 方法

H.4.13.1.1 getContentType

原型：public int getContentType()
描述：获取该搜索的内容类型。
参数：无。
返回：int型，表示搜索的内容类型，取值详见org.ngb.toolkit.search.SearchContentType接口的“内容类型”常量域定义。

H.4.13.1.2 getCriteriaFlags

原型：public int getCriteriaFlags()
描述：获取该搜索的指示符。

参数：无。

返回：int型，表示搜索指示符，取值详见org.ngb.toolkit.search.SearchCriterialFlags接口的“过滤条件”常量域定义。

H.4.13.1.3 getSearchField

原型：public int getSearchField()

描述：获取该搜索的搜索字段。

参数：无。

返回：int型，表示搜索字段，取值详见org.ngb.toolkit.search.SearchFields接口的“搜索源”常量域定义。

H.4.13.1.4 getSearchString

原型：public java.lang.String getSearchString()

描述：获取该搜索的匹配字符串。

参数：无。

返回：java.lang.String对象，表示搜索匹配字符串。

H.4.13.1.5 getSortCriteria

原型：public SortCriteria getSortCriteria()

描述：获取该搜索的搜索结果排序信息，比如排序方式和类型。

参数：无。

返回：org.ngb.toolkit.search.SortCriteria对象，表示搜索结果排序信息。

H.4.13.1.6 getSources

原型：public int getSources()

描述：获取该搜索的搜索数据源。

参数：无。

返回：int型，表示搜索数据源，取值详见SourceType接口的“搜索数据源”常量域定义。

H.4.14 接口org.ngb.toolkit.search.SearchHistoryList

原型：public interface SearchHistoryList extends java.util.ListIterator

描述：描述了搜索历史记录列表，提供了遍历搜索历史记录列表的功能。

注：SearchHistoryList是一个SearchHistoryItem的集合。

H.4.15 接口org.ngb.toolkit.search.SearchStatus

原型：public interface org.ngb.toolkit.search.SearchStatus

描述：搜索状态接口，定义了搜索状态常量。

H.4.15.1 常量域——搜索状态

H.4.15.1.1 INITIATED

原型：public static final byte INITIATED = 0

描述：搜索状态——初始化完成。

注：仅对全局搜索有效。

H.4.15.1.2 IN_PROGRESS

原型：public static final byte IN_PROGRESS = 1
描述：搜索状态——正在进行。此状态表示从现在开始，可以向应用提供搜索结果，但不是全部。
注：仅对全局搜索有效。

H.4.15.1.3 COMPLETED

原型：public static final byte COMPLETED = 2
描述：搜索状态——结束。此状态表明搜索完成，可以检索所有结果。

H.4.15.1.4 INTERRUPTED

原型：public static final byte INTERRUPTED = 3
描述：搜索状态——中断。搜索正在进行时，应用程序调用GlobalSearchSession对象的stopSearch()方法停止搜索，应用程序将会得到该通知。
注：仅对全局搜索有效。

H.4.15.1.5 TIMEOUT

原型：public static final byte TIMEOUT = 4
描述：搜索状态——超时后搜索自动停止。若在超时时间内未检索到结果，应用程序会得到此状态通知。此状态也表明，搜索成功停止，因此应用程序并不需要调用GlobalSearchSession对象的stopSearch()方法或AutoCompleteSearchSession对象的stopSearch()方法明确停止搜索。

H.4.15.1.6 TIMEOUT_STOP_FAILED

原型：public static final byte TIMEOUT_STOP_FAILED = 5
描述：搜索状态——超时后搜索停止失败。若在超时时间内未检索到结果，应用程序会得到此状态通知。此状态也表明，搜索没有成功停止，因此应用程序应调用GlobalSearchSession对象的stopSearch()方法明确停止搜索。
注：仅对全局搜索有效。

H.4.15.1.7 FAILED

原型：public static final byte FAILED = 6
描述：搜索状态——搜索失败。此状态表示开始搜索失败，因此没有结果可以返回给应用。

H.4.15.1.8 STOP_SUCCESS

原型：public static final byte STOP_SUCCESS = 7
描述：搜索状态——停止搜索成功。

H.4.15.1.9 STOP_FAILED

原型：public static final byte STOP_FAILED = 8
描述：搜索状态——停止搜索失败。

H.4.15.1.10 DISPOSE_SUCCESS

原型：public static final byte DISPOSE_SUCCESS = 9
描述：搜索状态——关闭搜索成功。

H.4.15.1.11 DISPOSE_FAILED

原型：public static final byte DISPOSE_FAILED = 10
描述：搜索状态——关闭搜索失败。

H.4.15.1.12 RETRIEVAL_SUCCESS

原型：public static final byte RETRIEVAL_SUCCESS = 11
描述：搜索状态——成功获取搜索结果。该状态表示存取成功，应用可以通过org.ngb.toolkit.search.GlobalSearchSession对象的getSearchResultList()方法获取搜索结果列表。
注：仅对全局搜索有效。

H.4.15.1.13 RETRIEVAL_FAILED

原型：public static final byte RETRIEVAL_FAILED = 12
描述：搜索状态——获取搜索结果失败。此状态表示由于某些失败原因，搜索结果不能被检索，因此不能通过org.ngb.toolkit.search.GlobalSearchSession对象的getSearchResultList()方法获取搜索结果列表。
注：仅对全局搜索有效。

H.4.15.1.14 RETRIEVAL_INSUFFICIENT

原型：public static final byte RETRIEVAL_INSUFFICIENT = 13
描述：搜索状态——无足够的搜索结果可用。此状态表示由于没有足够的搜索结果导致不能被检索，因此不能通过调用GlobalSearchSession.getSearchResultList()得到搜索结果列表。
注：仅对全局搜索有效。

H.4.16 接口org.ngb.toolkit.search.SourceType

原型：public interface org.ngb.toolkit.search.SourceType
描述：搜索数据源常量定义。

H.4.16.1 常量域——搜索数据源

H.4.16.1.1 ALL

原型：public static final int ALL = 0
描述：搜索数据源——从SI信息库、本地录制节目等有效数据源中进行搜索。

H.4.16.1.2 BROADCAST

原型：public static final int BROADCAST = 1
描述：搜索数据源——仅搜索SI信息库。

H.4.16.1.3 RECORDED

原型：public static final int RECORDED = 2

描述：搜索数据源——仅搜索本地录制/下载的内容。

H.4.17 类org.ngb.toolkit.search.AutoCompleteSearchFilter

原型：public abstract class org.ngb.toolkit.search.AutoCompleteSearchFilter

描述：匹配搜索过滤器，提供了匹配搜索过滤条件设置和获取方法。过滤条件可以是：

——搜索数据源；

——搜索字段；

——文本信息语种；

——返回结果的最大数目；

——搜索超时时间限制。

注：应用程序使用AutoCompleteSearchFilter可以获得以下效果：按照指定的搜索数据源和搜索字段，在超时时间限制范围内列出与用户输入的字符相匹配的提示字符串列表，字符串数量小于等于应用程序设定的最大值。

H.4.17.1 常量域——搜索结果数量默认最大值

H.4.17.1.1 DEFAULT_MAX_AUTO_COMPLETE_SEARCH_RESULTS

原型：public static final int DEFAULT_MAX_AUTO_COMPLETE_SEARCH_RESULTS = 10

描述：常量——定义了搜索结果数量默认最大值。

该常量是默认的搜索结果的最大值，如果应用程序需要设置成其他值，可以通过调用setMaxResults()方法实现。

H.4.17.2 方法

H.4.17.2.1 getAutoCompleteSearchFilter

原型：public static AutoCompleteSearchFilter getAutoCompleteSearchFilter()

描述：获取系统实现的AutoCompleteSearchFilter类的实例。

参数：无。

返回：AutoCompleteSearchFilter对象，

表示系统实现的org.ngb.toolkit.search.AutoCompleteSearchFilter类实例。

H.4.17.2.2 getMaxResults

原型：public abstract int getMaxResults()

描述：获取匹配搜索返回结果数目的最大值。

参数：无。

注：若应用程序没有设置，则默认返回DEFAULT_MAX_AUTO_COMPLETE_SEARCH_RESULTS。

返回：int型，表示匹配搜索返回结果数目的最大值。

H.4.17.2.3 getSearchField

原型：public abstract int getSearchField()

描述：获取匹配搜索需要查找的字段。

参数：无。

返回：int型，表示匹配搜索需要查找的字段，取值详见org.ngb.toolkit.search.SearchFields接口的“搜索域”常量域定义。

H.4.17.2.4 getSearchLanguage

原型：public abstract String getSearchLanguage()

描述：获取匹配搜索的文本信息语种类型。

参数：无。

返回：java.lang.String对象，表示匹配搜索的文本信息语种类型，语种三字母代码遵循GB/T 4880.2—2000标准。

H.4.17.2.5 getSource

原型：public abstract int getSource()

描述：获取匹配搜索的搜索数据源。

参数：无。

返回：int型，表示匹配搜索的搜索数据源，取值详见SourceType接口的“搜索数据源”常量域定义。若应用程序未设置匹配搜索数据源，则默认返回SourceType.ALL。

H.4.17.2.6 getTimeLimit

原型：public abstract int getTimeLimit()

描述：获取匹配搜索的超时时间限制。

注：若应用程序没有设置超时时间限制，则默认返回0。

返回：int型，表示匹配搜索的超时时间限制，单位为毫秒。

H.4.17.2.7 setMaxResults

原型：public abstract void setMaxResults(int maxResults)
throws java.lang.IllegalArgumentException

描述：设置匹配搜索返回结果数目的最大值。

参数：maxResults - int型，表示匹配搜索返回结果数目的最大值。

返回：无。

异常：java.lang.IllegalArgumentException - 若maxResults小于等于0或超出系统支持的最大值，则抛出此异常。

H.4.17.2.8 setSearchField

原型：public abstract void setSearchField(int searchField)
throws java.lang.IllegalArgumentException

描述：设置匹配搜索需要查找的字段。搜索引擎将会搜索数据库中的下列字段：

——标题；

——关键字；

——简介。

参数：searchField - int型，表示搜索字段范围，取值详见SearchFields接口的“搜索域”常量域定义。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数searchField不在SearchFields接口的“搜索域”常量域中定义，则抛出此异常。

H.4.17.2.9 setSearchLanguage

原型：public abstract void setSearchLanguage(java.lang.String iso639code)

throws java.lang.IllegalArgumentException

描述：设置匹配搜索的文本信息语种类型。

注：在一次匹配搜索中只能设置一种搜索语种，若没有设置，则默认语种为“zho”。

参数：iso639code - java.lang.String对象，表示匹配搜索的文本信息语种类型，语种三字母代码遵循GB/T 4880.2—2000标准。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数iso639code不符合GB/T 4880.2—2000标准，则抛出此异常。

H.4.17.2.10 setSource

原型：public abstract void setSource(int sourceType)
throws java.lang.IllegalArgumentException

描述：设置匹配搜索的搜索数据源。

注：若应用程序没有设置匹配搜索数据源，则默认使用SourceType.ALL。

参数：sourceType - int型，表示匹配搜索的数据源，取值详见org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域定义。

返回：无。

异常：java.lang.IllegalArgumentException - 若参数sourceType指定的搜索源未在org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域中定义，则抛出此异常。

H.4.17.2.11 setTimeLimit

原型：public abstract void setTimeLimit(int timeLimit)
throws java.lang.IllegalArgumentException

描述：设置匹配搜索的超时时间限制。

注：若设置了超时时间，则到达超时时间时无论有没有匹配结果，匹配搜索都将返回。

参数：timeLimit - int型，表示超时时间，单位为毫秒。若输入0表示一直等待。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数timeLimit小于0，则抛出此异常。

H.4.18 类org.ngb.toolkit.search.GlobalSearchFilter

原型：public abstract class org.ngb.toolkit.search.GlobalSearchFilter

描述：描述了一个全局搜索过滤器，提供了全局搜索过滤条件设置和获取方法。过滤条件可以是：

——搜索数据源；

——搜索字段；

——文本信息语种；

——返回结果的最大数目；

——搜索超时时间限制。

注：应用程序使用org.ngb.toolkit.search.GlobalSearchFilter可以获得以下效果：按照指定的搜索数据源和搜索字段，在超时时间限制范围内列出与用户输入的关键字相匹配的搜索结果列表，搜索结果数量小于等于应用程序设定的最大值。

H.4.18.1 常量域

H.4.18.1.1 DEFAULT_MAX_GLOBAL_SEARCH_RESULTS

原型：public static final int DEFAULT_MAX_GLOBAL_SEARCH_RESULTS = 50

描述：常量——定义了搜索结果数量默认最大值。该常量是默认的搜索结果的最大值，如果应用程序需要设置其他值，可以通过调用setMaxResults(int)方法来实现。

H.4.18.2 方法

H.4.18.2.1 getGlobalSearchFilter

原型：public static GlobalSearchFilter getGlobalSearchFilter()

描述：获取系统实现的GlobalSearchFilter类的实例。

参数：无。

返回：org.ngb.toolkit.search.GlobalSearchFilter对象，表示系统实现的GlobalSearchFilter类实例。

H.4.18.2.2 getContentNibble

原型：public abstract int getContentNibble()

描述：获取待过滤节目的类别。

参数：无。

返回：int型，表示待过滤节目的类别。

H.4.18.2.3 getContentType

原型：public abstract int getContentType()

描述：获取待过滤器节目的内容类型。

参数：无。

返回：int型，表示待过滤节目的内容类型，取值详见SearchContentType接口的“内容类型”常量域定义。若应用程序没有调用setContentType()方法设置该值，则默认返回SearchContentType.ALL。

H.4.18.2.4 getCriteriaFlags

原型：public abstract long getCriteriaFlags()

描述：获取搜索条件标记。

参数：无。

返回：int型，表示搜索条件标记，取值详见SearchCriteriaFlags接口的“过滤条件”常量域定义。若应用程序没有使用setCriteriaFlags()方法设置该值，则默认返回SearchCriteriaFlags.FLAG_NONE。

H.4.18.2.5 getMaxResults

原型：public abstract int getMaxResults()

描述：获取全局搜索返回结果数目的最大值。

参数：无。

返回：int型，表示全局搜索返回结果数目的最大值。若应用程序没有设置，则默认返回DEFAULT_MAX_GLOBAL_SEARCH_RESULTS。

H.4.18.2.6 getSearchFieId

原型：public abstract int getSearchField()
描述：获取全局搜索需要查找的字段。
参数：无。
返回：int型，表示全局搜索需要查找的字段，取值详见org.ngb.toolkit.search.SearchFields接口的“搜索域”常量域定义。

H.4.18.2.7 getSearchLanguage

原型：public abstract String getSearchLanguage()
描述：获取全局搜索的文本信息语种类型。
参数：无。
返回：java.lang.String对象，表示全局搜索的文本信息语种类型，语种三字母代码遵循GB/T 4880.2—2000标准。

H.4.18.2.8 getSource

原型：public abstract int getSource()
描述：获取全局搜索的搜索数据源。
参数：无。
返回：int型，表示全局搜索的搜索数据源，取值详见org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域定义。若应用程序没有设置全局搜索数据源，则默认返回SourceType.ALL。

H.4.18.2.9 getThreshold

原型：public abstract int getThreshold()
描述：获取搜索查询结果的阈值。
参数：无。
返回：int型，表示搜索查询结果阈值。若应用程序没有设置阈值，则默认返回0。

H.4.18.2.10 getTimeLimit

原型：public abstract int getTimeLimit()
描述：获取全局搜索的超时时间限制。
参数：无。
返回：int型，表示匹配全局搜索的超时时间限制，单位为毫秒。若应用程序没有设置超时时间限制，则默认返回0。

H.4.18.2.11 setContentNibble

原型：public abstract void setContentNibble(int contentNibble)
throws java.lang.IllegalArgumentException
描述：设置待过滤节目的类别。
参数：contentNibble - int型，表示待过滤节目的类别
返回：无。
异常：java.lang.IllegalArgumentException - 若输入参数非法，则抛出此异常。

H.4.18.2.12 setContentType

原型：public abstract void setContentType(int contentType)

throws java.lang.IllegalArgumentException

描述：设置待过滤节目的内容类型。

参数：contentType - int型，表示待过滤节目的内容类型，可取值为SearchContentType.ALL、SearchContentType.AUDIO_ONLY、SearchContentType.VIDEO_ONLY，详见SearchContentType接口的“内容类型”常量域定义。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数contentType未在org.ngb.toolkit.search.SearchContentType接口的“内容类型”的常量域中定义，则抛出此异常。

H.4.18.2.13 setCriteriaFlags

原型：public abstract void setCriteriaFlags(long criteriaFlags)
throws java.lang.IllegalArgumentException

描述：设置搜索条件标记。

参数：criteriaFlags - long型，表示搜索条件，取值详见org.ngb.toolkit.search.SearchCriteriaFlags接口的“过滤条件”常量域定义。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数criteriaFlags未在org.ngb.toolkit.search.SearchCriteriaFlags接口的“过滤条件”常量域中定义，则抛出此异常。

H.4.18.2.14 setMaxResults

原型：public abstract void setMaxResults(int maxResults)
throws java.lang.IllegalArgumentException

描述：设置全局搜索返回结果数目的最大值。

参数：maxResults - int型，表示全局搜索返回结果数目的最大值。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数maxResults小于等于0或超出系统支持的最大值，则抛出此异常。

H.4.18.2.15 setSearchField

原型：public abstract void setSearchField(int searchField)
throws java.lang.IllegalArgumentException

描述：设置全局搜索需要查找的字段。搜索引擎将会搜索数据库中的下列字段：

——标题；

——关键字；

——简介。

参数：searchField - int型，表示搜索字段范围，取值详见org.ngb.toolkit.search.SearchFields接口的“搜索域”常量域定义。

返回：无。

异常：java.lang.IllegalArgumentException - 若输入参数searchField取值不在org.ngb.toolkit.search.SearchFields接口的“搜索域”常量域中定义，则抛出此异常。

H.4.18.2.16 setSearchLanguage

原型：public abstract void setSearchLanguage(java.lang.String iso639code)
throws java.lang.IllegalArgumentException
描述：设置全局搜索的文本信息语种类型。
注：在一次全局搜索中只能设置一种语种，若未设置，则默认语种为中文“zho”。
参数：iso639code - java.lang.String对象，表示全局搜索的文本信息语种类型，语种三字母代码遵循GB/T 4880.2—2000规定。
返回：无。
异常：java.lang.IllegalArgumentException - 若输入参数iso639code不符合GB/T 4880.2—2000规定，则抛出该异常。

H.4.18.2.17 setSource

原型：public abstract void setSource(int sourceType)
throws java.lang.IllegalArgumentException
描述：设置执行搜索的数据源。
注：若应用程序没有设置全局搜索数据源，则默认使用SourceType.ALL。
参数：sourceType - int型，表示全局搜索数据源，取值详见org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域定义。
返回：无。
异常：java.lang.IllegalArgumentException - 若参数sourceType指定的搜索源未在org.ngb.toolkit.search.SourceType接口的“搜索数据源”常量域中定义，则抛出此异常。

H.4.18.2.18 setThreshold

原型：public abstract void setThreshold(int thresholdLimit)
throws java.lang.IllegalArgumentException
描述：设置搜索查询结果的阈值。每次一旦达到阈值，应用程序将会收到SearchStatus.IN_PROGRESS状态通知。默认的阈值若设为0，应用程序将会收到SearchStatus.COMPLETED的状态通知。
参数：thresholdLimit - int型，表示搜索查询结果的阈值。
返回：无。
异常：java.lang.IllegalArgumentException - 若输入参数thresholdLimit小于0，则抛出此异常。

H.4.18.2.19 setTimeLimit

原型：public abstract void setTimeLimit(int timeLimit)
throws java.lang.IllegalArgumentException
描述：设置全局搜索的超时时间限制。
注：若设置了超时时间，则到达超时时间时无论有没有结果，全局搜索都将返回。
参数：timeLimit - int型，表示超时时间，单位为毫秒。若输入0表示一直等待。
返回：无。
异常：java.lang.IllegalArgumentException - 若输入参数timeLimit小于0，则抛出此异常。

H.4.19 类org.ngb.toolkit.search.SearchManager

原型：public class org.ngb.toolkit.search.SearchManager
描述：全局搜索和匹配搜索的搜索管理器，是搜索功能模块的入口类。

应用程序必须遵循以下几个步骤来实现匹配搜索功能：

——创建 AutoCompleteSearchFilter 对象；

——调用 getAutoCompleteSearchSession()方法获取 AutoCompleteSearchSession 对象；

——等待用户输入查询字符串；

——用户输入完成后，调用 org.ngb.toolkit.search.AutoCompleteSearchSession 对象的 startSearch()方法启动匹配搜索；

——在超时时间内，一旦有了结果，搜索引擎将把搜索状态告知应用程序，即通过 org.ngb.toolkit.search.AutoCompleteSearchListener 对象的回调方法通知应用程序；

——应用程序得到来自搜索引擎的搜索状态后调用 org.ngb.toolkit.search.AutoCompleteSearchSession 对象的 getSearchResultList()方法获得匹配搜索结果列表。

应用程序必须遵循以下几个步骤来实现全局搜索功能：

——创建 org.ngb.toolkit.search.GlobalSearchFilter 和 org.ngb.toolkit.search.SortCriteria 并设置好各字段值；

——调用 getGlobalSearchSession()方法获取 GlobalSearchSession 对象；

——调用org.ngb.toolkit.search.GlobalSearchSession对象的setPageSize()方法设置每页保存的搜索结果条目数量；

——等待用户输入查询字符串；

——用户输入完成后，调用 org.ngb.toolkit.search.GlobalSearchSession 对象的 startSearch()方法启动全局搜索功能；

——在超时时间内，一旦有了结果，搜索引擎将把搜索状态告知应用程序，即通过 org.ngb.toolkit.search.GlobalSearchListener 对象的回调方法通知应用程序；

——应用程序得到来自搜索引擎的搜索状态后调用 org.ngb.toolkit.search.GlobalSearchSession 对象的 retrievePage()方法获取分好页的搜索结果；

——检索成功后，应用程序通过调用 org.ngb.toolkit.search.GlobalSearchSession 对象的 getSearchResultList()方法获得全局搜索结果列表。

H.4.19.1 方法

H.4.19.1.1 getInstance

原型：public static org.ngb.toolkit.search.SearchManager getInstance()

描述：获取系统实现的org.ngb.toolkit.search.SearchManager类的唯一实例。

参数：无。

返回：org.ngb.toolkit.search.SearchManager类单例。

H.4.19.1.2 getAutoCompleteSearchSession

原型：public org.ngb.toolkit.search.AutoCompleteSearchSession getAutoCompleteSearchSession (org.ngb.toolkit.search.AutoCompleteSearchFilter filter, org.ngb.toolkit.search.AutoCompleteSearchListener listener) throws java.lang.IllegalArgumentException

描述：获取匹配搜索会话对象。每个匹配搜索会话对象都是一个独立的搜索会话，为了在同一时间能有多个匹配搜索会话，应用程序不仅需要获取不同对象，还要处理相应的回调功能。

参数：filter - org.ngb.toolkit.search.AutoCompleteSearchFilter对象，表示匹配搜索过滤器；
listener - org.ngb.toolkit.search.AutoCompleteSearchListener对象，表示匹配搜索过程监听器。
返回：org.ngb.toolkit.search.AutoCompleteSearchSession对象，表示一个匹配搜索会话实例。
异常：java.lang.IllegalArgumentException – 若输入参数非法，则抛出此异常。

H.4.19.1.3 getGlobalSearchSession

原型：public org.ngb.toolkit.search.GlobalSearchSession getGlobalSearchSession
(org.ngb.toolkit.search.GlobalSearchFilter filter,
org.ngb.toolkit.search.SortCriteria sortCriteria,
org.ngb.toolkit.search.GlobalSearchListener listener)
throws java.lang.IllegalArgumentException
描述：获取全局搜索会话对象。每个全局搜索会话对象都是一个独立的搜索会话，为了在同一时间能有多个全局搜索会话，应用程序不仅需要获取不同对象，还要处理相应的回调功能。根据终端能力，由系统自行决定是否同时支持多个全局搜索会话。
参数：filter - org.ngb.toolkit.search.GlobalSearchFilter对象，表示全局搜索过滤器；
sortCriteria - org.ngb.toolkit.search.SortCriteria对象，表示搜索结果排序准则；
listener - org.ngb.toolkit.search.GlobalSearchListener对象，表示全局搜索过程监听器。
返回：org.ngb.toolkit.search.GlobalSearchSession对象，表示一个全局搜索会话实例。
异常：java.lang.IllegalArgumentException – 若输入参数非法，则抛出此异常。

H.4.19.1.4 getSearchHistory

原型：public SearchHistoryList getSearchHistory()
throws org.ngb.toolkit.search.SearchException
描述：获取历史搜索信息列表。
——若应用程序在本次搜索之前完成过某些搜索，则此列表会包含该搜索记录，该列表的每一项都包含诸如搜索关键字、搜索结果来源等信息；
——若应用程序在本次搜索之前未进行过搜索，则无对象列表返回，并向应用程序抛出org.ngb.toolkit.search.SearchException异常。
参数：无。
返回：org.ngb.toolkit.search.SearchHistoryList对象，表示历史搜索记录，若无历史搜索记录，则返回null。
异常：org.ngb.toolkit.search.SearchException – 若获取历史搜索记录失败，则抛出此异常。

H.4.19.1.5 clearHistory

原型：public void clearHistory() throws Org.ngb.toolkit.search.SearchException
描述：清除历史搜索记录。在任何时刻，应用程序都可以调用本方法清除历史搜索记录，以保护个人隐私。
参数：无。
返回：无。
异常：org.ngb.toolkit.search.SearchException – 若历史记录清除失败，则抛出此异常。

H.4.20 类org.ngb.toolkit.search.SortCriteria

原型：public class org.ngb.toolkit.search.SortCriteria
描述：定义了排序常量和排序方法。

H.4.20.1 常量域——排序方式

H.4.20.1.1 SORT_ORDER_NONE

原型：public static final byte SORT_ORDER_NONE = 0
描述：排序方式——无。

H.4.20.1.2 SORT_ORDER_ASCENDING

原型：public static final byte SORT_ORDER_ASCENDING = 1
描述：排序方式——升序。

H.4.20.1.3 SORT_ORDER_DESCENDING

原型：public static final byte SORT_ORDER_DESCENDING = 2
描述：排序方式——降序。

H.4.20.2 常量域——排序要素

H.4.20.2.1 SORT_TYPE_NONE

原型：public static final int SORT_TYPE_NONE = 0
描述：排序要素——不指定排序字段。

H.4.20.2.2 SORT_TYPE_TITLE

原型：public static final int SORT_TYPE_TITLE = 1
描述：排序要素——基于标题排序。

H.4.20.2.3 SORT_TYPE_START_TIME

原型：public static final int SORT_TYPE_START_TIME = 2
描述：排序要素——基于开始时间排序。

H.4.20.2.4 SORT_TYPE_CONTENT_NIBBLE

原型：public static final int SORT_TYPE_CONTENT_NIBBLE = 15
描述：排序要素——基于事件类型排序。

H.4.20.3 方法

H.4.20.3.1 SortCriteria

原型：public SortCriteria()
描述：构造方法，创建一个搜索排序准则对象。
参数：无。

H.4.20.3.2 getOrder

原型：public int getOrder()

描述：获取排序方式。
参数：无。
返回：int型，表示排序方式，可取值为SORT_ORDER_NONE、SORT_ORDER_ASCENDING或SORT_ORDER_DESCENDING，详见SortCriteria接口的“排序方式”常量域定义。

H.4.20.3.3 setOrder

原型：public void setOrder(int order) throws java.lang.IllegalArgumentException
描述：设置搜索结果返回的排序方式。
参数：order - int型，表示排序方式，可取值为SORT_ORDER_ASCENDING、SORT_ORDER_DESCENDING或SORT_ORDER_NONE，详见org.ngb.toolkit.search.SortCriteria接口的“排序方式”常量域定义。
返回：无。
异常：java.lang.IllegalArgumentException – 若输入参数非法，则抛出此异常。

H.4.20.3.4 getType

原型：public int getType()
描述：获取排序要素。
参数：无。
返回：int型，表示排序要素，可取值为SORT_TYPE_NONE、SORT_TYPE_TITLE、SORT_TYPE_SOURCE、SORT_TYPE_START_TIME，详见org.ngb.toolkit.search.SortCriteria接口的“排序要素”常量域定义。

H.4.20.3.5 setType

原型：public void setType(int sortType) throws java.lang.IllegalArgumentException
描述：设置搜索结果返回的排序要素。
参数：sortType - int型，表示排序要素，可取值为SORT_TYPE_NONE、SORT_TYPE_TITLE、SORT_TYPE_SOURCE、SORT_TYPE_START_TIME，详见SortCriteria接口的“排序要素”常量域定义。
返回：无。
异常：java.lang.IllegalArgumentException – 若输入参数非法，则抛出此异常。

H.4.21 异常org.ngb.toolkit.search.SearchException

原型：public class org.ngb.toolkit.search.SearchException extends java.lang.Exception
描述：描述了全局搜索出现错误时的异常。可能的失败原因有：
——开始搜索失败；
——停止搜索失败；
——终止搜索会话失败；
——搜索查询失败；
——搜索展开失败；
——创造结果集失败。

H.4.21.1 常量域——搜索异常

H.4.21.1.1 SEARCH_NOT_STARTED_PREVIOUSLY

原型：public static final int SEARCH_NOT_STARTED_PREVIOUSLY = 100
描述：搜索异常——表示搜索未启动。当应用程序试图停止或者销毁一个未启动的搜索时抛出此异常。

H.4.21.1.2 SEARCH_START_FAILED

原型：public static final int SEARCH_START_FAILED = 101
描述：搜索异常——表示搜索启动失败。

H.4.21.1.3 SEARCH_STOP_FAILED

原型：public static final int SEARCH_STOP_FAILED = 102
描述：搜索异常——表示搜索停止失败。

H.4.21.1.4 SEARCH_DISPOSE_FAILED

原型：public static final int SEARCH_DISPOSE_FAILED = 103
描述：搜索异常——表示搜索注销失败。

H.4.21.1.5 SEARCH_RETRIEVAL_FAILED

原型：public static final int SEARCH_RETRIEVAL_FAILED = 104
描述：搜索异常——表示搜索结果获取失败。

H.4.21.1.6 SEARCH_EXPAND_FAILED

原型：public static final int SEARCH_EXPAND_FAILED = 105
描述：搜索异常——表示搜索扩展失败。

H.4.21.1.7 PREVIOUS_SEARCH_NOT_COMPLETED

原型：public static final int PREVIOUS_SEARCH_NOT_COMPLETED = 106
描述：搜索异常——表示上次搜索未完成。

H.4.21.1.8 RESULT_LIST_CREATION_FAILED

原型：public static final int RESULT_LIST_CREATION_FAILED = 107
描述：搜索异常——表示生成搜索结果失败。

H.4.21.1.9 SAVING_HISTORY_FAILED

原型：public static final int SAVING_HISTORY_FAILED = 108
描述：搜索异常——表示保存搜索历史记录失败。

H.4.21.1.10 CLEARING_HISTORY_FAILED

原型：public static final int CLEARING_HISTORY_FAILED = 109
描述：搜索异常——表示清除搜索历史记录失败。

H.4.21.1.11 RETRIEVAL_IN_PROGRESS

原型：public static final int RETRIEVAL_IN_PROGRESS = 110
描述：搜索异常——表示当前一个搜索进程未完成时又启动了另外一个搜索进程。应用程序应该在前一个搜索完成后才能启动新的搜索请求。

H.4.21.2 方法

H.4.21.2.1 getMessageId

原型：public int getMessageId()
描述：获取异常的标识符。
参数：无。
返回：int型，表示异常的标识符，取值详见org.ngb.toolkit.search.SearchException类的“搜索异常”常量域定义。

附 录 I
（规范性附录）
JAVA-多屏互动单元

I.1 概述

本附录支持多屏互动局域网接口，支持在局域网中客户端和服务端进行发现和连接。本附录定义了与多屏互动相关的JAVA接口。

I.1.1 场景描述

多屏互动单元的使用场景描述如下：

a) 假设当前具有两台TVOS系统的播放设备A和B，A、B系统均连接至同一个局域网。

b) A系统的应用程序通过A系统提供的IMultiScreenService多端联动的通信接口，使用startMultiScreenServer方法启动多端联动服务，将B系统的多端联动组件信息传递给A系统。

c) 接着应用程序使用findSPs方法搜索到同一个局域网下的B系统设备，通过connect方法经由A系统实现与B系统的连接。

d) 连接成功后，A系统程序通过B系统提供的IMultiScreenCallBack接口，接收连接状态信息，并由onConnected方法通知A系统连接状态。至此A、B系统实现多端联动通讯，应用程序可由IMultiScreenService与IMultiScreenCallBack接口提供的inputKeyCode等方法将A系统的操作数据信息，或是更多视频数据信息传递给B系统，在B系统上实现相关操作与播放。

I.2 多屏互动模块

多屏互动模块可实现应用客户端在局域网内发现、连接、控制服务端设备功能。

多屏互动模块概要见表I.1。

表 I.1 多屏互动模块概要

接口	
IMultiScreenService	局域网环境中用于支撑多屏互动组件服务的功能接口。
IMultiScreenCallBack	多屏互动组件对外提供的远程访问接口。

I.2.1 接口org.tvos.multiscreen.IMultiScreenService

原型：public interface org.tvos.multiscreen.IMultiScreenService
描述：局域网环境中用于支撑多屏互动组件服务的功能接口。

I.2.1.1 常量域——命令类型

I.2.1.1.1 START_MULTISCREENSERVER

原型: public static final int START_MULTISCREENSERVER= (IBinder.FIRST_CALL_TRANSACTION + 0)

描述：启动服务端。

I.2.1.1.2 STOP_MULTISCREENSERVER

原型：public static final int STOP_MULTISCREENSERVER = (IBinder.FIRST_CALL_TRANSACTION + 1)

描述：停止服务端。

I.2.1.1.3 START_MULTISCREENCLIENT

原型：public static final int START_MULTISCREENCLIENT= (IBinder.FIRST_CALL_TRANSACTION + 2)

描述：启动客户端。

I.2.1.1.4 STOP_MULTISCREENCLIENT

原型：public static final int STOP_MULTISCREENCLIENT = (IBinder.FIRST_CALL_TRANSACTION + 3)

描述：断开客户端。

I.2.1.1.5 FIND_SPS

原型：public static final int FIND_SPS = (IBinder.FIRST_CALL_TRANSACTION + 4)

描述：发现服务端。

I.2.1.1.6 CONNECT

原型：public static final int CONNECT = (IBinder.FIRST_CALL_TRANSACTION + 5)

描述：连接服务端。

I.2.1.1.7 SET_CALLBACK

原型：public static final int SET_CALLBACK = (IBinder.FIRST_CALL_TRANSACTION + 6)

描述：设置回调。

I.2.1.1.8 QUERY_INFO

原型：public static final int QUERY_INFO = (IBinder.FIRST_CALL_TRANSACTION + 7)

描述：查询信息。

I.2.1.1.9 EXEC_CMD

原型：public static final int EXEC_CMD = (IBinder.FIRST_CALL_TRANSACTION + 8)

描述：执行命令。

I.2.1.1.10 INPUT_KEYCODE

原型：public static final int INPUT_KEYCODE = (IBinder.FIRST_CALL_TRANSACTION + 9)

描述：调用按键。

I.2.1.1.11 NOTIFY_ALL_REMOTE

原型：public static final int NOTIFY_ALL_REMOTE = (IBinder.FIRST_CALL_TRANSACTION + 10)

描述：通知所有局域网设备。

I.2.1.2 方法

I.2.1.2.1 startMultiScreenServer

原型：public int startMultiScreenServer(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion,String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：远程接口，启动多屏互动组件服务端。
参数：spName –java.lang.String型，多屏互动组件服务端名称；
spVersion- java.lang.String型，多屏互动组件服务端版本；
deviceType - java.lang.String型，多屏互动组件服务端设备类型；
port - int型，搜索到的多屏互动组件设备端口号。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.2 stopMultiScreenServer

原型：public int stopMultiScreenServer() throws RemoteException
描述：远远程接口，停止多屏互动组件服务端。
参数：无。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.3 startMultiScreenClient

原型：public int startMultiScreenClient(java.lang.String clientName) throws RemoteException
描述：远程接口，启动多屏互动组件客户端。
参数：clientName –java.lang.String型，输入参数，多屏互动组件客户端名称。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.4 stopMultiScreenClient

原型：public int stopMultiScreenClient() throws RemoteException
描述：远程接口，关闭多屏互动组件客户端。
参数：无。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.5 findSPs

原型：public int findSPs() throws RemoteException
描述：远程接口，搜索局域网下的多屏互动服务组件设备。
参数：无。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.6 connect

原型：public int connect(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion, java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException

描述：远程接口，连接局域网下的多屏互动服务组件服务端设备。
参数：spName –java.lang.String型，输入参数，多屏互动组件服务端名称；
spDeviceType – java.lang.String型，输入参数，多屏互动组件服务端设备类型；
spServiceInfo – java.lang.String型，输入参数，多屏互动组件服务端服务信息；
spVersion – java.lang.String型，输入参数，多屏互动组件服务端版本；
ipaddress – java.lang.String型，输入参数，搜索到的多屏互动组件设备ip地址；
port – int型，输入参数，搜索到的多屏互动组件设备端口号；
hostname – java.lang.String型，输入参数，搜索到的多屏互动组件设备主机名。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.7 setCallBack

原型：public int setCallBack(org.tvos.os.IBinder ibinder) throws RemoteException
描述：远程接口，设置远程调用回调接口。
参数：ibinder –org.tvos.os.Ibinder对象，输入参数，远程接口回调实例。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.8 queryInfo

原型：public int queryInfo(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String cmdid, java.lang.String atrribute, java.lang.String params)throws RemoteException
描述：远程接口，多屏互动组件客户端请求获取信。
参数：ipaddress– java.lang.String型，输入参数，搜索到的多屏互动组件设备ip地址；
port– int型，输入参数，搜索到的多屏互动组件设备端口号；
hostname – java.lang.String型，输入参数，搜索到的多屏互动组件设备主机名；
cmdid– java.lang.String型，输入参数，请求信息的指令id；
atrribute – java.lang.String型，输入参数，请求信息的指令名称；
params– java.lang.String型，输入参数，请求信息的指令参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.9 execCmd

原型：public int execCmd(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String cmd, java.lang.String param)throws RemoteException
描述：远程接口，多屏互动组件客户端请求执行指令。
参数：ipaddress – java.lang.Sting型，输入参数，搜索到的多屏互动组件设备ip地址；
port – int型，输入参数，搜索到的多屏互动组件设备端口号；
hostname – java.lang.Sting型，输入参数，搜索到的多屏互动组件设备主机名；
action – java.lang.Sting型，输入参数，按键指令；
param – java.lang.Sting型，输入参数，按键指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.10 inputKeyCode

原型：public int inputKeyCode(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String action, java.lang.String param)throws RemoteException

描述：远程接口，多屏互动组件客户端发送虚拟按键输入的指令。
参数：ipaddress- java.lang.String型，输入参数，搜索到的多屏互动组件设备ip地址；
port – int型，输入参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输入参数，搜索到的多屏互动组件设备主机名；
action - java.lang.String型，输入参数，按键指令；
param - java.lang.String型，输入参数，按键指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.1.2.11 boardCastAllDevice

原型：public int boardCastAllDevice(java.lang.String cmd, java.lang.String param)throws RemoteException
描述：远程接口，多屏互动组件服务端往所有连上的设备发送广播。
参数：cmd –java.lang.String型，输入参数，广播指令；
param –java.lang.String型，输入参数，广播指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2 接口org.tvos.multiscreen.IMultiScreenCallBack

原型：public interface org.tvos.multiscreen.IMultiScreenCallBack
描述：局域网环境中用于支撑多屏互动组件服务的功能接口。

I.2.2.1 常量域——状态回调

I.2.2.1.1 ON_SP_FOUNDED

原型:static final int ON_SP_FOUNDED = (IBinder.FIRST_CALL_TRANSACTION + 0)
描述：当客户端发现。

I.2.2.1.2 ON_CONNECTED

原型：static final int ON_CONNECTED = (IBinder.FIRST_CALL_TRANSACTION + 1)
描述：当连接上服务端。

I.2.2.1.3 ON_CONNECTEDREFUSED

原型：static final int ON_CONNECTEDREFUSED= (IBinder.FIRST_CALL_TRANSACTION + 2)
描述：当连接拒绝。

I.2.2.1.4 ON_DISCONNECTED

原型：static final int ON_DISCONNECTED = (IBinder.FIRST_CALL_TRANSACTION + 3)
描述：断开服务端时。

I.2.2.1.5 ON_SERVICE_ACTIVITED

原型：static final int ON_SERVICE_ACTIVITED = (IBinder.FIRST_CALL_TRANSACTION + 4)
描述：当服务端活跃响应。

I.2.2.1.6 ON_SERVICE_DEACTIVITED

原型：static final int ON_SERVICE_DEACTIVITED = (IBinder.FIRST_CALL_TRANSACTION + 5)
描述：当服务端无响应。

I.2.2.1.7 ON_QUERY_INFO

原型：static final int ON_QUERY_INFO = (IBinder.FIRST_CALL_TRANSACTION + 6)
描述：当服务端响应查询信息。

I.2.2.1.8 ON_QUERY_RESPONSE

原型：static final int ON_QUERY_RESPONSE= (IBinder.FIRST_CALL_TRANSACTION + 7)
描述：当查询响应信息。

I.2.2.1.9 ON_EXECUTE

原型：static final int ON_EXECUTE = (IBinder.FIRST_CALL_TRANSACTION + 8)
描述：当执行命令后的返回结果。

I.2.2.1.10 ON_INPUT

原型：static final int ON_INPUT = (IBinder.FIRST_CALL_TRANSACTION + 9)
描述：当输入方式的回调返回。

I.2.2.1.11 ON_NOTIFY

原型：static final int ON_NOTIFY= (IBinder.FIRST_CALL_TRANSACTION + 10)
描述：当接收到通知返回。

I.2.2.2 方法

I.2.2.2.1 onSpFounded

原型：public int onSpFounded(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion, java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：通知JAVA适配层多屏互动组件服务搜索完成。
参数：spName –java.lang.String型，输出参数，搜索到的多屏互动组件服务名；
spDeviceType – java.lang.String型，输出参数，搜索到的多屏互动组件设备类型；
spServiceInfo– java.lang.String型，输出参数，搜索到的多屏互动组件服务端信息；
spVersion– java.lang.String型，输出参数，搜索到的多屏互动组件服务端服务版本；
ipaddress– java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port– int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname – java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.2 onConnected

原型：public int onConnected(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion, java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：通知JAVA适配层多屏互动组件服务连接完成。

参数： spName - java.lang.String型，输出参数，搜索到的多屏互动组件服务名；
spDeviceType - java.lang.String型，输出参数，搜索到的多屏互动组件设备类型；
spServiceInfo- java.lang.String型，输出参数，搜索到的多屏互动组件服务端信息；
spVersion- java.lang.String型，输出参数，搜索到的多屏互动组件服务端服务版本；
ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port- int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.3 onConnectRefused

原型：public int onConnectRefused(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion, java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：通知JAVA适配层多屏互动组件服务连接被拒。
参数： spName - java.lang.String型，输出参数，搜索到的多屏互动组件服务名；
spDeviceType - java.lang.String型，输出参数，搜索到的多屏互动组件设备类型；
spServiceInfo- java.lang.String型，输出参数，搜索到的多屏互动组件服务端信息；
spVersion- java.lang.String型，输出参数，搜索到的多屏互动组件服务端服务版本；
ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port- int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。
返回： int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.4 onDisconnected

原型：public int onDisconnected(java.lang.String spName, java.lang.String spDeviceType, java.lang.String spServiceInfo, java.lang.String spVersion, java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：通知JAVA适配层多屏互动组件服务连接断开。
参数： spName - java.lang.String型，输出参数，搜索到的多屏互动组件服务名；
spDeviceType - java.lang.String型，输出参数，搜索到的多屏互动组件设备类型；
spServiceInfo- java.lang.String型，输出参数，搜索到的多屏互动组件服务端信息；
spVersion- java.lang.String型，输出参数，搜索到的多屏互动组件服务端服务版本；
ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port- int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。
返回： int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.5 onServiceActivited

原型：public int onServiceActivited(java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException
描述：通通知JAVA适配层多屏互动组件服务被激活。
参数： ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port- int型，输出参数，搜索到的多屏互动组件设备端口号；

hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。

返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.6 onServiceDeactived

原型：public int onServiceDeactived(java.lang.String ipaddress, int port, java.lang.String hostname) throws RemoteException

描述：通知JAVA适配层多屏互动组件服务被注销。

参数： ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；

port- int型，输出参数，搜索到的多屏互动组件设备端口号；

hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名。

返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.7 onQueryInfo

原型：public int onQueryInfo(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String id, java.lang.String attribute, java.lang.String param) throws RemoteException

描述：通知JAVA适配层接收到多屏互动组件客户端发过来的数据请求。

参数：ipaddress - java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；

port - int型，输出参数，搜索到的多屏互动组件设备端口号；

hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名；

id - java.lang.String型，输出参数，接收到的请求的命令id；

attribute - java.lang.String型，输出参数，接收到的请求的命令属性；

param - java.lang.String型，输出参数，接收到的请求的附带的参数。

返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.8 onQueryResponse

原型：public int onQueryResponse(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String id, java.lang.String attribute, java.lang.String param) throws RemoteException

描述：通知JAVA适配层接收到多屏互动组件服务端回复了数据请求。

参数：ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；

port - int型，输出参数，搜索到的多屏互动组件设备端口号；

hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名；

id - java.lang.String型，输出参数，接收到的请求的命令id；

attribute - java.lang.String型，输出参数，接收到的请求的命令属性；

param - java.lang.String型，输出参数，接收到的请求的附带的参数。

返回：int型， 远程接口调用成功返回0，否则返回错误码。

I.2.2.2.9 onExecute

原型：public int onExecute(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String cmd, java.lang.String param) throws RemoteException

描述：通知JAVA适配层接收到多屏互动组件客户端发送了执行指令请求。

参数：ipaddress- java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；

port - int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名；
cmd - java.lang.String型，输出参数，执行的指令；
param - java.lang.String型，输出参数，执行指令附带的参数。
返回：int型， 远程接口调用成功返回0，否则返回错误码。

I.2.2.2.10 onInputKeyCode

原型：public int onInputKeyCode(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String action, java.lang.String param) throws RemoteException
描述：通知JAVA适配层接收到多屏互动组件客户端发送了执行按键注入的请求。
参数：ipaddress - java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port - int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名；
cmd - java.lang.String型，输出参数，执行的指令；
param - java.lang.String型，输出参数，执行指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.11 onNotify

原型：public int onNotify(java.lang.String ipaddress, int port, java.lang.String hostname, java.lang.String cmd, java.lang.String param) throws RemoteException
描述：通知JAVA适配层多屏互动组件接收到通知。
参数：ipaddress - java.lang.String型，输出参数，搜索到的多屏互动组件设备ip地址；
port - int型，输出参数，搜索到的多屏互动组件设备端口号；
hostname - java.lang.String型，输出参数，搜索到的多屏互动组件设备主机名；
cmd - java.lang.String型，输出参数，接收到的指令；
param - java.lang.String型，输出参数，接收到的指令参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

I.2.2.2.12 onTransact

原型：public boolean onTransact(int code, Parcel data, Parcel reply, int flags) throws RemoteException
描述：AIDL远程调用回调接口。
参数：code - int型，输出参数，对应的接口ID。
data – Parcel对象，输出参数,对应的接口数据；
reply – Parcel对象，输入参数，对应的接口回复数据；
flags – int型，输出参数，远程接口标记位。
返回：int型，远程接口调用成功返回0，否则返回错误码。

附 录 J
（规范性附录）
JAVA-DRM 管理单元

J.1 概述

本附录定义了DRM管理相关的JAVA接口。

J.2 DRM管理模块

DRM管理模块概要见表J.1。

表 J.1 DRM 管理模块概要

类	
ChinaDrmManager	DRM模块管理类。
ChinaDrmTeeRetVal	ChinaDrmManager类的ChinaDrm_SendCommandToTEE方法的返回类型类。

J.2.1 类org.ngb.drm.services.ChinaDrmManager

原型：public class ChinaDrmManager
描述：DRM模块管理类。

J.2.1.1 接口

J.2.1.1.1 OnMessageListener

原型：public interface OnMessageListener
描述：DRM消息通知接口，DRM Manager发送通知消息时调用，DRM APP实现。
参数：无。
返回：无。

J.2.1.1.2 OnLicenseListener

原型：public interface OnLicenseListener
描述：DRM消息通知接口，DRM Manager发送获取许可证请求消息时调用，DRM APP实现。
参数：无。
返回：无。

J.2.1.1.3 OnDecryptListener

原型：public interface OnDecryptListener
描述：DRM消息通知接口，DRM Manager发送解密请求消息时调用，DRM APP实现。
参数：无。
返回：无。

J.2.1.2 方法

J.2.1.2.1 setOnMessageListener

原型：public synchronized void setOnMessageListener(org.ngb.drm.services.ChinaDrmManager.OnMessageListener messageListener)
描述：设置消息通知的回调函数，DRM Manager发送通知消息时调用。
参数：messageListener - OnMessageListener 型，表示接口定义的回调函数。
返回：无。

J.2.1.2.2 setOnLicenseListener

原型：public synchronized void setOnLicenseListener(org.ngb.drm.services.ChinaDrmManager.OnLicenseListener licenseListener)
描述：设置获取许可证的回调函数，DRM Manager发送获取许可证请求消息时调用。
参数：licenseListener - OnLicenseListener型，表示接口定义的回调函数。
返回：无。

J.2.1.2.3 setOnDecryptListener

原型：public synchronized void setOnDecryptListener(org.ngb.drm.services.ChinaDrmManager.OnDecryptListener decryptListener)
描述：设置数据解密的回调函数，DRM Manager发送解密请求消息时调用。
参数：decryptListener - OnDecryptListener 型，表示接口定义的回调函数。
返回：无。

J.2.1.2.4 registerApp

原型：public int registerApp(java.lang.String drmId,byte[] uuid,int register_commandId,
byte[] register_pridata,int enflag,int licensereq_commmandId,
int decrypt_commandId)
描述：DRM APP注册，可以自己私有定义licensereq和decrypt 的命令ID。
参数：drm_Id - java.lang.String对象，表示DRM APP唯一标识；
uuid - byte[] 型，表示DRM APP对应TApp的唯一标识；
register_commandId - int型，表示与TApp通信的注册commandid；
register_pridata - int型，表示与TApp通信的注册携带的私有注册数据，用于TApp验证DRM APP合法性；
enflag - int型，表示解密调用方式；
licensereq_commandId - int型，表示查询许可证对应的commandid；
decrypt_commandId - int型，表示解密数据对应的commandid。
返回：int型，0表示成功，非0表示失败。

J.2.1.2.5 unRegisterApp

原型：public int unRegisterApp()
描述：DRM APP注销。
参数：无。
返回：int型，0表示成功，非0表示失败。

J.2.1.2.6 sendCommandToTEE

原型：public ChinaDrmTeeRetVal sendCommandToTEE(int commandId, byte[] inputData)
描述：发送命令到TEE，并获取数据。
参数：commandId -int型，表示命令ID；
inputData -byte[]型，表示传入TEE的数据。
返回：ChinaDrmTeeRetVal类的对象。

J.2.1.2.7 sendMessageToPlayer

原型：public int sendMessageToPlayer(int type, byte[] message)
描述：发送消息到播放器。
参数：type -int型，表示消息类型；
message -byte[]型，表示消息。
返回：int型，0表示成功，非0表示失败。

J.2.2 类org.ngb.drm.services.ChinaDrmTeeRetVal

原型：public class ChinaDrmTeeRetVal
描述：ChinaDrmManager类的ChinaDrm_SendCommandToTEE方法的返回类型类。

J.2.2.1 方法

J.2.2.1.1 getData

原型：public byte[] getData()
描述：得到DRM Tapp返回的数据。
参数：无。
返回：byte[]型数据。

J.2.2.1.2 getDatalen

原型：public long getDatalen()
描述：得到DRM Tapp返回的数据的长度。
参数：无。
返回：long型，数据的长度。

J.2.2.1.3 getOriginCode

原型：public int getOriginCode()
描述：得到返回值的来源。
参数：无。
返回：int型，1为API，2为通讯（COMMS）,3为TEE，4为DRM Tapp。

J.2.2.1.4 getReturnCode

原型：public int getReturnCode()
描述：得到返回值。
参数：无。
返回：int型，0表示成功，非0表示失败。

附 录 K
（规范性附录）
JAVA-DCAS 管理单元

K.1 概述

本附录定义了与DCAS管理相关的JAVA接口。

K.2 CAS解扰模块

CAS解扰模块提供了DCAS终端软件平台上层API。
CAS解扰模块概要见表K.1。

表 K.1 CAS 解扰模块概要

接口	
CASModule	用于表示请求解扰一组基本流的CASModule对象。
CASDataUtils	用来获取和设置CA信息，读写DCAS数据。
CADescriptor	提供了CA描述符的信息，给定业务的PMT中的可能提供CA描述符。此外CAT中也会有CA描述符的出现。
CAServiceComponentInfo	用于提取特定CA业务成分的信息，如ECM PID和用于装载控制字的DescramblerContext。
CASPacketListener	DCAS应用通过本接口来接收带外CAS Packets（如 EMMs）。
CASSession	提供CAS会话相关信息。
CAStatus	DCAS应用在每次DescramblerContext中的解扰状态发生变化时发送CAStatus，本状态用来指示解扰成功与否。如果所解扰成分中任何一个发生解扰失败，本状态必须汇报整个业务的解扰请求失败。当终端软件平台收到一个新的CAStatus,它应通过本段描述的CAS事件通知其他应用。
CATListener	DCAS应用需实现该接口，使用CAT中的CA描述符来过滤带内EMM。
CATNotifier	DCAS应用使用本方法注册用于获取CAT更新通知的监听器。
类	
CASModuleManager	用来注册所有被DCAS应用实现的CASModule。
CASPermission	任一DCAS应用必须获取CASPersmission方可访问CASModuleManager，本机制用于确保只有被网络运营商授权的DCAS应用方可使用DCAS API。

K.2.1 接口org.ngb.net.cas.module.CASModule

原型：public interface org.ngb.net.cas.module.CASModule
描述：本接口描述用于表示请求解扰一组基本流的CASModule对象。

K.2.1.1 方法

K.2.1.1.1 startDescrambling

原型：public void startDescrambling(org.ngb.net.cas.module.CASSession cassession,
org.ngb.net.cas.module.CAServiceComponentInfo[] casci)
描述：本方法由终端软件平台调用，请求CASModule解扰给定会话中的一组基本流。
DCAS应用从CAS会话中可以得到相关NetworkInterface对象。
从NetworkInterface中，可以获取当前TransportStream对象，用于sectionsAPI进行ECM section过滤。
参数：cassession – org.ngb.net.cas.module.CASSession对象，请求解扰操作的会话；
casci – org.ngb.net.cas.module.CAServiceComponentInfo数组，CA业务组件信息数组。该数组可以用于获取相关ECM PID,以及用以DCAS装载控制字的DescramblerContext对象。
返回：无。

K.2.1.1.2 updateDescrambling

原型：public void updateDescrambling(org.ngb.net.cas.module.CASSession casSession,
org.ngb.net.cas.module.CAServiceComponentInfo[] casci)
描述：终端软件平台调用本方法来更新CASModule中的解扰组件列表。
根据请求，CASModule将开始解扰添加到数组的组件，并停止解扰移除的组件。
对于更新后不变的成分，不发生任何变化。
注：本方法很少被调用，通常在一个会话中由于PMT变化而发生。
终端软件平台还可在CAS会话如会话类型发生变化时通过调用本方法通知CASModule。
参数：casSession – org.ngb.net.cas.module.CASSession对象，请求解扰操作的会话；
casci – org.ngb.net.cas.module.CAServiceComponentInfo数组，CA业务组件信息数组，该数组可以用于获取相关ECM PID,以及用以DCAS装载控制字的DescramblerContext对象。
返回：无。

K.2.1.1.3 stopDescrambling

原型：public void stopDescrambling(org.ngb.net.cas.module.CASSession casSession)
描述：终端软件平台调用本接口请求 CASModule 停止解扰给定会话中的所有组件。
参数：casSession –org.ngb.net.cas.module.CASSession 对象，请求解扰操作的会话。
返回：无。

K.2.1.1.4 getCAInfo

原型：public java.lang.String getCAInfo(int cmdId, java.lang.String data)
描述：终端软件平台调用本接口获取 CA 信息。
参数：cmdId – int 型，命令的唯一标识，可根据实际项目扩展；
data – java.lang.String 型，查询参数。
返回：java.lang.String 型，CA 信息数据。

K.2.1.1.5 setCAInfo

原型：public int setCAInfo(int cmdId, java.lang.String data)
描述：终端软件平台调用本接口设置 CA 信息。
参数：cmdId – int 型，命令的唯一标识，可根据实际项目扩展；
data – java.lang.String 型，CA 信息数据。
返回：int 型，设置成功返回 0。

K.2.2 接口org.ngb.net.cas.module.CASDataUtils

原型：public interface org.ngb.net.cas.module.CASDataUtils
描述：本接口描述用于表示获取CA相关信息的通用接口。

K.2.2.1 方法

K.2.2.1.1 getCAInfo

原型：public java.lang.String getCAInfo(int casId, int cmdId, java.lang.String data)
描述：终端软件平台响应用户操作向 DCAS 应用获取 CA 信息。
参数：casId - int 型，指定的 CAS 厂商；
cmdId - int 型，命令的唯一标识，可根据实际项目扩展；
data - java.lang.String 型，查询参数。
返回：java.lang.String 型，CA 信息数据。

K.2.2.1.2 setCAInfo

原型：public int setCAInfo(int casId, int cmdId, java.lang.String data)
描述：终端软件平台响应用户操作设置 CA 信息给 DCAS 应用。
参数：casId - int 型，指定的 CAS 厂商；
cmdId - int 型，命令的唯一标识，可根据实际项目扩展；
data - java.lang.String 型，查询参数。
返回：int 型，设置成功返回 0。

K.2.2.1.3 getData

原型：public java.lang.String getData(int casId, int cmdId, int[] type)
描述：终端软件平台响应用户操作向 DCAS 管理器获取数据。
参数：casId - int 型，指定的 CAS 厂商；
cmdId - int 型，命令的唯一标识，可根据实际项目扩展；
type - int 数组，数据类型引用；
返回：java.lang.String 型，获取的数据。

K.2.2.1.4 setData

原型：public int setData(int casId, int cmdId, int type, java.lang.String data)
描述：终端软件平台响应用户操作设置 CA 信息给 DCAS 应用。
参数：casId - int 型，指定的 CAS 厂商；
cmdId - int 型，命令的唯一标识，可根据实际项目扩展；
data - java.lang.String 型，DCAS 相关数据；
type - int 型，数据类型。
返回：int 型，设置成功返回 0。

K.2.3 接口org.ngb.net.cas.module.CADescriptor

原型：public interface org.ngb.net.cas.module.CADescripter
描述：本接口提供了CA描述符的信息，给定业务的PMT中的可能提供CA描述符。此外CAT中也会有CA描述符的出现。

K.2.3.1 方法

K.2.3.1.1 getCASystemId

原型：public int getCASystemId()
描述：本方法返回CA描述符的CASystemId。
参数：无。
返回：int 型,表示 CASystemId。

K.2.3.1.2 getPid

原型：public int getPid()
描述：本方法返回CA描述符中的PID (ECM PID或EMM PID)。
参数：无。
返回：int 型,表示 PID值。

K.2.3.1.3 getPrivateData

原型：public byte[] getPrivateData()
描述：本方法返回CA描述符终端的私有数据数组。
参数：无。
返回：byte 数组，表示 privateData。

K.2.4 接口org.ngb.net.cas.module.CAServiceComponentInfo

原型：public interface org.ngb.net.cas.module.CAServiceComponentInfo
描述：本接口用于提取特定CA业务成分的信息，如ECM PID和用于装载控制字的DescramblerContext。

K.2.4.1 方法

K.2.4.1.1 getDescramblerContext

原型：public org.ngb.net.cas.controller.DescramblerContext getDescramblerContext()
描述：本方法返回用于DCAS应用装载控制字的DescramberContext对象，在PMT的组件循环中出现多次相同CA描述符（相同的ECMPID和私有数据）的情形下，应该只存在唯一一个DescramblerContext。
参数：无。
返回：org.ngb.net.cas.controller.DescramblerContext对象。

K.2.4.1.2 getCADescriptor

原型：public org.ngb.net.cas.module.CADescriptor getCADescriptor()
描述：本方法返回业务组件相关的CA描述符，CADescriptor实例由PMT中的CA信息产生。
参数：无。
返回：org.ngb.net.cas.module.CADescriptor 对象。

K.2.4.1.3 getComponentStreamPIDs

原型：public int[] getComponentStreamPIDs()

描述：本方法返回一个由PMT中描述的基本流数组，数组元素的顺序应同getComponentStreamType返回的数组元素顺序一致。

参数：无。

返回：ES（基本流）PID数组。

K.2.4.1.4 getComponentStreamTypes

原型：public int[] getComponentStreamTypes()

描述：本方法返回PMT中的流类型数组，流类型值应遵循MPEG标准ISO/IEC 13818-1。数组元素的顺序应同getComponentStreamPID返回的数组元素顺序一致。

参数：无。

返回：int[],流类型数组。

K.2.4.1.5 getServiceIdentifiers

原型：public int[] getServiceIdentifiers()

描述：本方法返回对象所关联的业务标识数组，业务标识的表现形式由具体的使用环境而定。

参数：无。

返回：int[],ServiceID数组。

K.2.5 接口org.ngb.net.cas.module.CASPacketListener

原型：public interface org.ngb.net.cas.module.CASPacketListener

描述：DCAS应用通过本接口来接收带外CAS Packets (如 EMMs)。DCAS应用根据给定的CA系统标识通过CASModuleManager 类提供的registerCasPacketListener方法注册本监听器。CA系统标识由参数casId表示。CAS包的接收依赖终端软件平台来实现。

K.2.5.1 方法

K.2.5.1.1 casPacketArrived

原型：public void casPacketArrived(int casId, byte [] casPacketData, byte [] casPacketHeader)

描述：DCAS应用通过已注册的监听器来获得CAS包。

参数：casId – int型，CA系统标识；

casPacketData – byte数组，CAS包数据；

casPacketHeader – byte数组，依赖终端软件平台的CAS包头。

返回：无。

K.2.6 接口org.ngb.net.cas.module.CASSession

原型：public interface org.ngb.net.cas.module.CASSession

描述：本接口提供CAS会话相关信息。

K.2.6.1 常量域——会话类型

K.2.6.1.1 TYPE_PRESENTATION

原型：public static final int TYPE_PRESENTATION = 0x00000001

描述：表示会话是TYPE_PRESENTATION类型。

K.2.6.1.2 TYPE_RECORDING

原型：public static final int TYPE_RECORDING = 0x00000002
描述：表示会话是TYPE_RECORDING类型。

K.2.6.1.3 TYPE_BUFFERING

原型：public static final int TYPE_BUFFERING = 0x00000004
描述：表示会话是TYPE_BUFFERING类型。

K.2.6.2 方法

K.2.6.2.1 getType

原型：public int getType()
描述：本方法返回本会话的操作类型。
参数：无。
返回：int型，操作类型，可以是本接口中定义的值之一或组合。

K.2.6.2.2 getNetworkInterface

原型：public org.davic.net.tuning.NetworkInterface getNetworkInterface()
描述：本方法返回同CAS会话相关联的NetworkInterface，DCAS应用可以从CAS会话中获取相关NetworkInterface对象。使用NetworkInterface, DCAS应用可得到先TransportStream对象，用于调用sections应用接口进行ECM Section过滤。
参数：无。
返回：一个org.davic.net.tuning.NetworkInterface对象。

K.2.6.2.3 getAssociatedService

原型：public java.lang.Object getAssociatedService()
描述：本方法返回CAS会话相关的业务。
参数：无。
返回：一个Service对象。

K.2.6.2.4 getServiceContext

原型：public java.lang.Object getServiceContext()
描述：本方法返回CAS会话相关的ServiceContext。
注意：在某些实现中ServiceContext没有实际意义，本方法将返回null。
参数：无。
返回：一个ServiceContext对象。

K.2.7 接口org.ngb.net.cas.module.CAStatus

原型：public interface org.ngb.net.cas.module.CAStatus
描述：DCAS应用当调用CASModuleManager中的sendDescramblingEvent 方法时使用本接口。DCAS应用在每次DescramblerContext中的解扰状态发生变化时发送CAStatus，本状态用来指示解扰成功与否，如果所解扰成分中任何一个发生解扰失败，本状态必须汇报整个业务的解扰请求

失败，当终端软件平台收到一个新的CAStatus,它应通过本段描述的CAS事件通知其他应用。具体应用接口在扩展应用接口部分说明。

K.2.7.1 方法

K.2.7.1.1 isSuccess

原型：public boolean isSuccess()
描述：本方法返回解扰请求的状态。
参数：无。
返回：boolean型，如果解扰成功返回true,解扰失败返回false。

K.2.7.1.2 getCAToken

原型：public int getCAToken()
描述：本方法返回用于其他应用通过IXC向DCAS应用查询网络相关信息的参数。
参数：无。
返回： int型，表示CA令牌。

K.2.8 接口org.ngb.net.cas.module.CATListener

原型：public interface org.ngb.net.cas.module.CATListener
描述：DCAS应用需实现该接口，使用CAT中的CA描述符来过滤带内EMM。DCAS应用需要通过CATNotifier接口中定义的registerCATListener注册本监听器。

K.2.8.1 方法

K.2.8.1.1 catUpdate

原型：public void catUpdate(org.ngb.net.cas.module.CADescriptor desc,
org.davic.net.tuning.NetworkInterface ni)
描述：本接口用于通知DCAS应用特定网络接口上的CAT更新。DCAS应用可以通过NetworkInterface对象，获取现TransportStream对象。TransportStream对象可用sections应用程序接口来实现EMM section 过滤。终端软件平台将CAT更新通知到跟casId相匹配并且注册过的CAT监听器。
参数：desc – org.ngb.net.cas.module.CADescriptor对象，DCAS应用通过CASDescriptor对象获取EMM PID;
ni – org.davic.net.tuning.NetworkInterface对象，CAT更新所在的NetworkInterface。
返回：无。

K.2.9 接口org.ngb.net.cas.module.CATNotifier

原型：public interface org.ngb.net.cas.module.CATNotifier
描述：DCAS应用使用本方法注册用于获取CAT更新通知的监听器，DCAS应用使用CAT信息过滤带内EMM。

K.2.9.1 方法

K.2.9.1.1 registerCATListener

原型：public void registerCATListener(int casId, org.ngb.net.cas.module.CATListener catListener)

描述：DCAS应用调用本方法注册一个CATListener。

参数：casId – int型，CA系统标识；

catListener – org.ngb.net.cas.module.CATListener对象，由于注册的CATListener。

返回：无。

K.2.9.1.2 unregisterCATListener

原型：public void unregisterCATListener(org.ngb.net.cas.module.CATListener catListener)

描述：DCAS应用调用本方法取消注册某个CATListener。

参数：catListener – org.ngb.net.cas.module.CATListener对象，需要取消注册的CATListener。

返回：无。

K.2.10 类org.ngb.net.cas.module.CASModuleManager

原型：public class org.ngb.net.cas.module.CASModuleManager

描述：用来注册所有被DCAS应用实现的CASModule。

K.2.10.1 方法

K.2.10.1.1 getInstance

原型：public static org.ngb.net.cas.module.CASModuleManager getInstance() throws java.lang.SecurityException

描述：本方法用于获取一个CASModuleManager单例。

参数：无。

返回：org.ngb.net.cas.module.CASModuleManager实例。

异常：java.lang.SecurityException-当安全策略被强制启用，但调用方没有被赋予一个CASPermission，抛出此异常。

K.2.10.1.2 registerCASmodule

原型：public void registerCASModule(org.ngb.net.cas.module.CASModule aModule,
int caSystemId,
int networkCAPriority,
java.lang.Object context)
throws java.lang.IllegalArgumentException

描述：本方法用于DCAS应用在终端软件平台上注册一个CASModule。

参数：aModule –org.ngb.net.cas.module.CASModule对象，需要注册的CASModule；

caSystemId – int型，CASModule管理的caSystemId；

networkCAPriority – int型，用于在超过一个CASModule在CASModuleManager中注册时使用，运营商可根据每个CASModule决定是否该参数可选，当优先级策略启用时，为运营商需要为每个CASModule指定优先级。终端软件平台应选择已注册，具有最高优先级，并且所管理的caSystemId – int型，在PMT中有相应CA描述符的CASModule发出解扰请求。当优先级策略禁用时，DCAS应用应给该参数置零，CASModule的决定方法由终端软件平台自行实现；

context – java.lang.Object对象，希望进行注册CASModule的DCAS应用的Context，用于终端软件平台决定DCAS应用的身份。

返回：无。

异常：java.lang.IllegalArgumentException-如果指定的CASModule实例已被注册，抛出此异常。

K.2.10.1.3 updateCASystemId

原型：public void updateCASystemId(org.ngb.net.cas.module.CASModule aModule, int caSystemId)
throws java.lang.IllegalArgumentException

描述：本方法用于DCAS应用向应用软件平台更新某CASModule中的CASystemId。

参数：aModule – org.ngb.net.cas.module.CASModule对象，指定CASModule；
caSystemId – int型，module管理的新caSystemId。

返回：无。

异常：java.lang.IllegalArgumentException 如果给定的CASModule实例尚未注册。

K.2.10.1.4 sendDescramblingEvent

原型：public void sendDescramblingEvent(org.ngb.net.cas.module.CASModule aModule,
org.ngb.net.cas.module.CASSession casSession,
org.ngb.net.cas.module.CAStatus aCAStatus)
throws java.lang.IllegalArgumentException

描述：本方法被用于DCAS应用向终端软件平台返回一个CAStatus当某个DescramblerContext每次由于相应服务中的加扰成分的加扰组件改变而变化时，DCAS应用必须发送CAStatus用于指示解扰是否成功。如果任何一个组件解扰失败，CAStatus必须通知此服务的解扰失败。当终端软件平台收到一个新的CAStatus时，应继续通过在扩展API中定义的CAS Event继续将该信息传至相应应用。

参数：aModule – org.ngb.net.cas.module.CASModule对象，指定CASModule；
casSession – org.ngb.net.cas.module.CASSession对象，解扰请求操作的会话；
aCAStatus – org.ngb.net.cas.module.CAStatus对象，需要发送的CAStatus。

返回：无。

异常：java.lang.IllegalArgumentException-如果所给的CASModule实例没有被注册，抛出此异常。

K.2.10.1.5 unregisterCASModule

原型：public void unregisterCASModule(org.ngb.net.cas.module.CASModule aModule)
throws java.lang.IllegalArgumentException

描述：本方法用于DCAS应用从终端软件平台上取消某CASModule的注册。

参数：aModule – org.ngb.net.cas.module.CASModule对象，需要取消注册的CASModule。

返回：无。

异常：java.lang.IllegalArgumentException 如果给定的CASModule尚未注册，抛出此异常。

K.2.10.1.6 getChipControllers

原型：public org.ngb.net.cas.controller.ChipController[] getChipControllers()

描述：本方法用于DCAS应用从终端软件平台请求可被使用的芯片控制器列表，本方法为每个终端安全芯片返回一个芯片控制器。很多终端只支持单个芯片控制器，这种情况下，返回的数组中只含有一个元素。

参数：无。

返回：org.ngb.net.cas.controller.ChipController数组，一个芯片控制器数组。

K.2.10.1.7 setcurrentController

原型：public void setCurrentController(org.ngb.net.cas.module.CASModule aModule,
org.ngb.net.cas.controller.ChipController aChipController)throws
java.lang.IllegalArgumentException

描述：本方法用于设置根据所给CAModule进行解扰操作所缺省使用的芯片控制器。如果本方法没有被调用，在CASModuleManager的选择无需指定。

参数：aModule —org.ngb.net.cas.module.CASModule对象，指定CASModule；
aChipController – org.ngb.net.cas.controller.ChipController对象，CASModule-所使用的缺省芯片控制器。

返回：无。

异常：java.lang.IllegalArgumentException 如果给定的CASModule尚未注册，抛出此异常。

K.2.10.1.8 setCCIBits

原型：public void setCCIBits(org.ngb.net.cas.module.CASModule aModule,
org.ngb.net.cas.module.CASSession casSession, int cciBits)

描述：本方法用于设定终端对于本业务进行拷贝保护所需拷贝控制信息数据（CCI bits），CCI比特信息的定义由终端软件平台指定并解释执行。

参数：aModule – org.ngb.net.cas.module.CASModule对象，指定CASModule；
casSession – org.ngb.net.cas.module.CASSession对象，解扰请求会话；
cciBits – int型，当前服务使用的CCI比特值；

返回：无。

K.2.10.1.9 setServiceListFilter

原型：public void setServiceListFilter(int filterData)

描述：本方法用于向终端软件平台提供用于业务列表过滤的参数，业务列表参数的具体含义由终端软件平台指定并执行。

参数：filterData – int型，业务列表过滤参数。

返回：无。

K.2.10.1.10 registerCASPacketListener

原型：public void registerCASPacketListener(int casId,
org.ngb.net.cas.module.CASPacketListener casPacketListener)
throws java.lang.IllegalArgumentException

描述：本方法用于DCAS应用注册一个CAPacketListener，CAPacketListener由终端软件平台调用，并向DCAS应用传送CAS数据包（如 EMM）。CA系统标识由参数casID表示， CAS数据包的接收由终端软件平台各自实现。

参数：casId - int型，CasystemID；
casPacketListener – org.ngb.net.cas.module.CASPacketListener对象，需要注册的CASPacketListener。

返回：无。

异常：java.lang.IllegalArgumentException 如果给定casId已经注册过listener。

K.2.10.1.11 unregisterCASPacketListener

原型：public void unregisterCASPacketListener(
org.ngb.net.cas.module.CASPacketListener casPacketListener)
throws java.lang.IllegalArgumentException
描述：本方法用于DCAS应用取消CASPacketListener的注册。
参数：casPacketListener – org.ngb.net.cas.module.CASPacketListener对象，需要取消注册的listener。
返回：无。
异常：java.lang.IllegalArgumentException 如果给定CASPacketListener尚未注册。

K.2.10.1.12 getDetachableSecurityDevices

原型：public org.ngb.net.cas.detachable.DetachableSecurityDevice[]
getDetachableSecurityDevices()
描述：本方法用于DCAS应用获得可分离设备（智能卡等）的对象句柄。
参数：无。
返回：org.ngb.net.cas.detachable.DetachableSecurityDevice对象数组。

K.2.10.1.13 receiveOsdMsg

原型：public void receiveOsdMsg(byte[] msg, int[] flags)
描述：显示OSD信息，其参数的具体含义和具体项目相关。
参数：msg - byte数组，OSD信息内容，可包括文本外的描述信息；
flags - int数组，OSD类型指示。
返回：无。

K.2.10.1.14 showFingerMsg

原型：public void showFingerMsg(org.ngb.net.cas.module.CASModule aModule,
org.ngb.net.cas.module.CASSession casSession, byte[] msg)
描述：显示OSD信息，其参数的具体含义和具体项目相关。
参数：aModule – org.ngb.net.cas.module.CASModule对象，指定CASModule；
casSession – org.ngb.net.cas.module.CASSession对象，请求解扰操作的会话；
msg – byte数组，指纹信息为空时显示信息。
返回：无。

K.2.10.1.15 receiveTuningAlert

原型：public void receiveTuningAlert(int[] serviceIdentifiers, int[] flags)
描述：应急广播。在某些项目中应急广播的参数不是通过CA系统来下发的，在此类情况下不必实现此函数。
参数：serviceIdentifiers - int数组，一组用于标示应急广播频道参数的数值。数值的含义由具体项目定义；
flags - int数组，可用于表示应急广播类型的参数。
返回：无。

K.2.10.1.16 getCATNotifier

原型：public org.ngb.net.cas.module.CATNotifier getCATNotifier()

描述：本方法被DCAS应用调用来获取CATnotifer对象，DCAS应用可以通过CAT notifier注册获取CAT更新通知的监听器。DCAS应用需要CAT信息来过滤带内EMM。

参数：无。

返回：org.ngb.net.cas.module.CATNotifier对象。

K.2.11 类org.ngb.net.cas.module.CASPermission

原型：public class org.ngb.net.cas.module.CASPermission extends java.security.BasicPermission

描述：任一DCAS应用必须获取CASPersmission方可访问CASModuleManager。本机制用于确保只有被网络运营商授权的DCAS应用方可使用DCAS API。

K.2.11.1 方法

K.2.11.1.1 CASPermission

原型：public CASPermission(java.lang.String name)

描述：创建一个新CASPermission. Name字符串现不使用，应设为空字符串。

参数：name – java.lang.String型，本CASPermission的名称。

返回：无。

K.2.11.1.2 CASPermission

原型：public CASPermission(java.lang.String name, java.lang.String actions)

描述：创建一个新CASPermission。 Name字符串现不使用，应设为空字符串，actions字符串现不使用，应设为空（null）。本构造方法用于Policy对象实例化一个新Permission objects。

参数：name – java.lang.String型，本CASPermission的名称；

actions – java.lang.String型，action列表。

返回：无。

K.3 CAS控制模块

CAS 控制模块提供了DCAS终端软件平台底层API。

CAS 控制模块概要见表K.2。

表 K.2 CAS 控制模块概要

接口	
DescramblerContext	用来控制终端安全芯片解扰功能的组件。可以实例化多个DescramblerContext来使用不同密钥解扰多个码流的情形。
ChipController	用来控制终端安全芯片执行的组件。
类	
Key	一个基本密码密钥。用来决定K-LAD使用的密码算法及密码函数的输出参数。
CWKey	解扰密钥或控制字。
CASTEEManager	与TEE中TA通信的功能接口。

K.3.1 接口org.ngb.net.cas.controller.DescramblerContext

原型：public interface org.ngb.net.cas.controller.DescramblerContext

描述：表示用来控制终端安全芯片解扰功能的组件。可以实例化多个DescramblerContext来使用不同密钥解扰多个码流的情形。

K.3.1.1 方法

K.3.1.1.1 loadCW

原型：public void loadCW(int Vendor_SysID, org.ngb.net.cas.controller.CWKey cwKey,
org.ngb.net.cas.controller.Key[] levelKeys, int schemeId)
throws org.ngb.net.cas.controller.CADriverException

描述：本方法用于通知终端软件平台向解扰器装入控制字，并向终端安全芯片装入所需密钥。一个解扰器通道是一个被单个控制字解扰的所有流的逻辑集合。解扰器通道的使用依赖于DescrablerContext。此外，DCAS应用应该通知终端软件平台当前控制字已失效（例如，由于授权原因），终端软件平台应当停止相应的解扰行为。在这种情况下，DCAS应用会提供一个null CWKey。

参数：Vendor_SysID – int型，该值用于标识CA厂商在控制器中用于支持根密钥派生。终端安全芯片的根密钥由该值派生，否则Vendor_SysID被忽略；
cwKey – org.ngb.net.cas.controller.CWKey控制字对象，如果控制字是明文，levelKeys参数被忽略。如果cwKey为null,即DCAS应用没有提供有效的控制字；
levelKeys – org.ngb.net.cas.controller.Key数组，用于置入终端安全芯片的多级密钥。密钥数组的索引等于终端安全芯片中的绝对位置。在数组中特定元素值为Null表明终端安全芯片中相应位置不应装入密钥，
即：levelKey[0]是Key1（被Key2加密）；levelKey[1]是Key2（被Key3加密）；levelKey[2]不必使用；
schemeId – int型，本schemeId用于指定终端安全芯片的加密算法(例如，AES，TDES)，ChipController接口中定义了方式（scheme）值的列表。如果控制器只支持一种方式，则该值被忽略。

返回：无。

异常：org.ngb.net.cas.controller.CADriverException 如果装入失败，抛出此异常。

K.3.1.1.2 overrideChipController

原型：public void overrideChipController(org.ngb.net.cas.controller.ChipController aChipController)
throws org.ngb.net.cas.controller.CADriverException

描述：本方法用于DCAS应用请求终端软件平台覆盖缺省终端安全芯片层级密钥的实现。可通过调用CASModuleManager的setCurrentController方法设置）。如果本方法没有被调用，终端安全芯片将使用缺省控制器。本方法只用于实现了多个终端安全芯片的终端安全芯片系统中。

参数：aChipController – org.ngb.net.cas.controller.ChipController对象，所要覆盖的控制器。

返回：无。

异常：org.ngb.net.cas.controller.CADriverException 如果操作失败，抛出此异常。

K.3.2 接口org.ngb.net.cas.controller.Chipcontroller

原型：public interface org.ngb.net.cas.controller.ChipController
描述：表示控制终端安全芯片执行的组件。

K.3.2.1 常量域

K.3.2.1.1 SCHEME_TDES

原型：public static final int SCHEME_TDES=0
描述：用于指示终端安全芯片应使用TDES的值。

K.3.2.1.2 SCHEME_AES

原型：public static final int SCHEME_AES=1
描述：用于指示终端安全芯片应使用AES的值。

K.3.2.1.3 PROCESSING_MODE_REGULAR

原型：public static final int PROCESSING_MODE_REGULAR=0
描述：用于指示终端安全芯片认证应答算法中不需进行额外处理的值。

K.3.2.1.4 PROCESSING_MODE_POST_PROCESSING

原型：public static final int PROCESSING_MODE_POST_PROCESSING=1
描述：用于指示终端安全芯片认证应答算法中需要实施后处理阶段的值。

K.3.2.2 方法

K.3.2.2.1 getPublicId

原型：public byte[] getPublicId() throws org.ngb.net.cas.controller.CADriverException
描述：本方法返回终端安全芯片的公共标识。
参数：无。
返回：byte数组，终端安全芯片的公共标识publicId。
异常：org.ngb.net.cas.controller.CADriverException 如果访问终端安全芯片驱动过程中有通信错误，抛出此异常。

K.3.2.2.2 getChipType

原型：public byte[] getChipType() throws org.ngb.net.cas.controller.CADriverException
描述：本方法返回终端安全芯片的类型标识。
参数：无。
返回：byte数组，终端安全芯片类型。
异常：org.net.net.cas.controller.CADriverException如果访问终端安全芯片驱动过程中有通信错误，抛出此异常。

K.3.2.2.3 getChipControllerProperty

原型：public java.lang.String getChipControllerProperty(java.lang.String propertyName) throws org.ngb.net.cas.controller.CADriverException

描述：本方法根据提供的终端安全芯片属性名称，返回该属性对应的值。本功能在本接口中被预留，可用于读出控制器中将来扩增的属性。现阶段没有定义任何属性名称。

参数：propertyName – java.lang.String型，属性名称。

返回：java.lang.String型，表示属性值。

异常：org.ngb.net.cas.controller.CADriverException如果访问终端安全芯片驱动过程中有通信错误，抛出此异常。

K.3.2.2.4 authenticate

原型：public byte[] authenticate(int Vendor_SysID, byte[] challenge,
org.ngb.net.cas.controller.Key[] levelKeys, int schemeId, int processingMode) throws org.ngb.net.cas.controller.CADriverExceptio

描述：本方法用于认证终端安全芯片中的层级密钥机制，终端安全芯片应根据送入的随机握手信息计算认证信息。

参数：Vendor_SysID – int型，该值用于标识CA厂商. 在控制器中用于支持根密钥派生。终端安全芯片的根密钥由该值派生. 否则Vendor_SysID被忽略；

Challenge – byte数组，握手信息，随机数；

levelKeys –org.ngb.net.cas.controller.Key数组，层级密钥所需各级密钥.密钥数组的索引等于终端安全芯片中的绝对位置。在数组中特定元素值为Null表明终端安全芯片中相应位置不应装入密钥。即：levelKey[0]是null；levelKey[1]是Key 2（被Key 3加密）；levelKey[2]不必使用。

schemeId – int型，本schemeId用于指定终端安全芯片的加密算法(例如，AES，TDES)如果控制器只支持一种方式，则该值被忽略；

processingMode – int型，用于指定计算应答结果计算过程中是否实施额外后处理过程的值，如果控制器只支持没有后处理模式，则该参数被忽略。

返回：byte数组，终端安全芯片所计算的应答响应。

异常：org.ngb.net.cas.controller.CADriverException 如果访问终端安全芯片驱动过程中有通信错误，抛出此异常。

K.3.2.2.5 encryptData

原型：public void encryptData(int Vendor_SysID,
org.ngb.net.cas.controller.CWKey cwKey,
org.ngb.net.cas.controller.Key[] levelKeys,
int schemeId,
int encryptionId,
byte[] src,
int srcPos,
byte[] dest,
int destPos,
int length)
throws org.ngb.net.cas.controller.CADriverException

描述：本方法调用芯片功能来加密内存中的数据。

参数：Vendor_SysID – int型，本参数用于标示CA厂商。安全芯片用此数值来派生根密钥；

cwKey – org.ngb.net.cas.controller.CWKey对象，加密用的控制字。如果控制字没有加密，之后的levelKeys将被忽略；
levelKeys – org.ngb.net.cas.controller.Key数组，层级密钥，数组中密钥的索引值等于其在层级密钥中的绝对位置。数组中的Null元素表明在此层级位置没有key需要被设置；
schemeId – int型，层级密钥所使用的加密算法。如果芯片只支持一种算法，则此参数将被忽略；
encryptionId – int型，数据加密/解密的算法（如AES，TDES）。如果芯片只支持一种算法，则此参数将被忽略；
src – byte数组，源数据数组；
srcPos – int型，源数据数组的起始位置；
dest – byte数组，目的数据数组；
destPos – int型，目的数据数组的起始位置；
length – int型，需要处理的数据字节数。

异常：层级密钥通讯错误时，抛出org.ngb.net.cas.controller.CADriverException异常。

K.3.2.2.6 decryptData

原型：public void decryptData(int Vendor_SysID,
org.ngb.net.cas.controller.CWKey cwKey,
org.ngb.net.cas.controller.Key[] levelKeys,
int schemeId,
int encryptionId,
byte[] src,
int srcPos,
byte[] dest,
int destPos,
int length)
throws org.ngb.net.cas.controller.CADriverException

描述：本方法调用芯片功能来解密内存中的数据。

参数：Vendor_SysID – int型，本参数用于标示CA厂商。安全芯片用此数值来派生根密钥；
cwKey – org.ngb.net.cas.controller.CWKey对象，解密用的控制字。如果控制字没有加密，之后的levelKeys将被忽略；
levelKeys – org.ngb.net.cas.controller.Key数组，层级密钥。数组中密钥的索引值等于其在层级密钥中的绝对位置。数组中的Null元素表明在此层级位置没有key需要被设置；
schemeId – int型，层级密钥所使用的加密算法。如果芯片只支持一种算法，则此参数将被忽略；
encryptionId – int型，数据加密/解密的算法（如AES，TDES）。如果芯片只支持一种算法，则此参数将被忽略；
src – byte数组，源数据数组；
srcPos – int型，源数据数组的起始位置；
dest – byte数组，目的数据数组；
destPos – int型，目的数据数组的起始位置；
length – int型，需要处理的数据字节数。

返回：无。

异常：层级密钥通讯错误时，抛出org.ngb.net.cas.controller.CADriverException异常。

K.3.3 类org.ngb.net.cas.controller.Key

原型：public class org.ngb.net.cas.controller.Key
描述：表示一个基本密码密钥。用来决定K-LAD使用的密码算法及密码函数的输出参数。

K.3.3.1 方法

K.3.3.1.1 Key

原型：public Key(byte[] value, boolean encrypted)
参数：value – byte数组，密钥的值；
encrypted – boolean型，密钥是否被加密的标志，true表示密钥已被加密，false表示密钥是明文。

K.3.3.1.2 getKeyValue

原型：public byte[] getKeyValue()
描述：本方法返回密钥的值。
参数：无。
返回：byte数组，密钥的值。

K.3.3.1.3 isEncrypted

原型：public boolean isEncrypted()
描述：本方法返回true时，表示密钥是加密的，false表示密钥未加密。
参数：无。
返回：boolean型，true密钥是加密的，false表示密钥未加密。

K.3.4 类org.ngb.net.cas.controller.CWKey

原型：public class org.ngb.net.cas.controller.CWKey extends
org.ngb.net.cas.controller.Key
描述：表示解扰密钥或控制字。

K.3.4.1 常量域

K.3.4.1.1 PARITY_EVEN

原型：public static final int PARITY_EVEN = 0
描述：表示PARITY_EVEN类型。

K.3.4.1.2 PARITY_ODD

原型：public static final int PARITY_ODD = 1
描述：表示PARITY_ODD类型。

K.3.4.2 方法

K.3.4.2.1 CWKey

原型：public CWKey(byte[] value, boolean encrypted, int parity)
描述：构造函数。
参数：value – byte数组，密钥的值；
encrypted – boolean型，真值表示密钥是加密的，假值表示密钥未加密；
parity – int型，奇偶值，表明控制字的奇偶性。

K.3.4.2.2 getParity

原型：public int getParity()
描述：本方法返回控制字的奇偶性。
参数：无。
返回：int型，控制字的奇偶性。

K.3.5 类org.ngb.net.cas.controller.CASTEEManager

原型：public class org.ngb.net.cas.controller.CASTEEManager
描述：与TEE中TA通信的接口。

K.3.5.1 方法

K.3.5.1.1 sendCommandToTEE

原型：public byte[] sendCommandToTEE(byte[] teeAppUUID, int commandId, byte[] inputData)
throws org.ngb.net.cas.controller.CADriverException
描述：DCAS应用选择对应的安全应用，并发送数据给安全应用。
参数：teeAppUUID – byte数组，TAPP的UUID标识。
commandId – int型，命令类型。
inputData – byte数组，输入的数据。
返回：byte数组，返回的数据。
异常：org.ngb.net.cas.controller.CADriverException如果跟TEE交互驱动过程中有通信错误。

K.4 CAS消息模块

CAS 消息模块提供了DCAS用于扩展API包，TVOS的DCAS应用需实现此包。
CAS 消息模块概要见表K.3。

表 K.3 CAS 消息模块概要

接口	
CASEventListener	应被需要接收CAS事件的应用实现。
CASAppInfo	提供DCAS应用的信息。
CASEventInfo	提供CASEvent的信息。
类	
CASEventManager	应用使用CASEventManager注册监听器，来获取CAS事件。

K.4.1 接口org.ngb.net.cas.event.CASEventListener

原型：public interface org.ngb.net.cas.event.CASEventListener

描述：本接口应被需要接收CAS事件的应用实现。CASevents提供了当前ServiceContext的CA Status和基本信息。

K.4.1.1 方法

K.4.1.1.1 receiveCASEvent

原型：public void receiveCASEvent(java.lang.Object serviceContext,
int appId,
int orgId,
boolean isSuccess,
int caToken)

描述：本方法用于向注册了CAS事件监听器的应用传递CAS事件。

参数：serviceContext – java.lang.Object对象，CAS事件所属的句柄
appId – int型，用于标识发送事件的DCAS应用。这些标识可被应用通过IXC和DCAS应用通信。在没有DCAS应用可以解扰给定码流时，终端软件平台应当使用值null作为appId的值调用本方法，获取此种CAS事件通知的应用应根据自己的设计和实现来处理该情况；
orgId – int型，用于标识发送事件的DCAS应用。标识App所属组织；
isSuccess – boolean型，用于指示解扰是否成功的布尔值；
caToken – int型，通过IXC传回DCAS应用的令牌，应用可使用该令牌通过IXC向DCAS应用查询特定的网络信息。

返回：无。

K.4.1.1.2 receiveCASOSDEvent

原型：public void receiveCASOSDEvent(java.lang.Object serviceContext,
int appId,
int orgId,
byte[] msg,
int[] flag)

描述：本方法用于向注册了CAS事件监听器的应用传递CAS的OSD事件。

参数：serviceContextCAS – java.lang.Object对象，事件所属的句柄；
appId – int型，用于标识发送事件的DCAS应用。这些标识可被应用通过IXC和DCAS应用通信。在没有DCAS应用可以解扰给定码流时，终端软件平台应当使用值null作为casAppId的值调用本方法，获取此种CAS事件通知的应用应根据自己的设计和实现来处理该情况；
orgId – int型，用于标识发送事件的DCAS应用。标识App所属组织。
msg – byte数组，用于传递OSD的内容；
flag – int数组，用于标识OSD的类型。

返回：无。

K.4.1.1.3 receiveCASFingerEvent

原型：public void receiveCASFingerEvent(java.lang.Object serviceContext,
int appId,
int orgId,
byte[] msg)

描述：本方法用于向注册了CAS事件监听器的应用传递CAS的指纹事件。
参数：serviceContext – java.lang.Object对象，CAS事件所属的句柄；
appId – int型，用于标识发送事件的DCAS应用。这些标识可被应用通过IXC和DCAS应用通信。在没有DCAS应用可以解扰给定码流时，终端软件平台应当使用值null作为casAppId的值调用本方法，获取此种CAS事件通知的应用应根据自己的设计和实现来处理该情况；
orgId – int型，用于标识发送事件的DCAS应用。标识App所属组织；
msg – byte数组，用于传递指纹的内容。
返回：无。

K.4.2 接口org.ngb.net.cas.event.CASAppInfo

原型：public interface org.ngb.net.cas.event.CASAppInfo
描述：本接口提供DCAS应用的信息。

K.4.2.1 方法

K.4.2.1.1 getAID

原型：public int getAID()
描述：本方法返回DCAS应用的application ID。
参数：无。
返回：int型，DCAS应用的application ID。

K.4.2.1.2 getOID

原型：public int getOID()
描述：本方法返回DCAS应用的organization ID。
参数：无。
返回：int型，DCAS应用的organization ID。

K.4.3 接口org.ngb.net.cas.event.CASEventInfo

原型：public interface org.ngb.net.cas.event.CASEventInfo
描述：本接口提供CASEvent的信息。

K.4.3.1 常量域

K.4.3.1.1 TYPE_PRESENTATION

原型：public static final int TYPE_PRESENTATION = 0x00000001
描述：表示TYPE_PRESENTATION类型。

K.4.3.1.2 TYPE_RECORDING

原型：public static final int TYPE_RECORDING = 0x00000002
描述：表示TYPE_RECORDING类型。

K.4.3.1.3 TYPE_BUFFERING

原型：public static final int TYPE_BUFFERING = 0x00000004
描述：表示TYPE_BUFFERING类型。

K.4.3.2 方法

K.4.3.2.1 getType

原型：public int getType()
描述：本方法返回产生CAS Event的操作类型。
参数：无。
返回：int型，操作类型，可以是本接口中定义的值其中之一或组合。

K.4.3.2.2 getNetworkInterface

原型：public org.davic.net.tuning.NetworkInterface getNetworkInterface()
描述：本方法返回和CAS Event相关的NetworkInterface。
参数：无。
返回：一个org.davic.net.tuning.NetworkInterface对象。

K.4.3.2.3 getAssociatedService

原型：public java.lang.Object getAssociatedService()
描述：本方法返回CAS Event相关联的业务。
参数：无。
返回：一个Service对象。

K.4.3.2.4 getServiceContext

原型：public java.lang.Object getServiceContext()
描述：本方法返回同CAS event相关联的ServiceContext。
注意，在某些操作中ServiceContext无实际意义。本方法返回null。
参数：无。
返回：一个ServiceContext对象。

K.4.4 类org.ngb.net.cas.event.CASEventManager

原型：public class org.ngb.net.cas.event.CASEventManager
描述：应用使用CASEventManager注册监听器，来获取CAS事件。
CAS events提供了当前的CA Status和基本信息。

K.4.4.1 方法

K.4.4.1.1 getInstance

原型：public static org.ngb.net.cas.event.CASEventManager getInstance()
描述：本方法用于取得一个CASEventManager实例单体。
参数：无。
返回：org.ngb.net.cas.event.CASEventManager实例。

K.4.4.1.2 addListener

原型：public void addListener(org.ngb.net.cas.event.CASEventListener aCASEventListener)
描述：本方法用于应用注册一个CASEventListener。该监听器用于传递所有的的CAS事件。

参数：aCASEventListener – org.ngb.net.cas.event.CASEventListener对象，需要注册的CASEventListener。
返回：无。

K.4.4.1.3 removeListener

原型：public void removeListener(org.ngb.net.cas.event.CASEventListener aCASEventListener)
描述：本方法用于应用取消注册一个CASEventListener。
参数：aCASEventListener – org.ngb.net.cas.event.CASEventListener对象，已经注册的CASEventListener。
返回：无。

K.5 CAS监听器模块

CAS 监听器模块提供了DCAS可分离安全设备API。
CAS 监听器模块概要见表K.4。

表 K.4 CAS 监听器模块概要

接口	
DetachableSecurityDevice	用于应用注册可分离安全设备的监听器，来获取设备插拔状态。
DetachableSecurityDeviceListener	可分离安全设备的监听器，应被需要监听设备插拔状态的应用实现。

K.5.1 接口org.ngb.net.cas.detachable.DetachableSecurityDevice

原型：public interface org.ngb.net.cas.detachable.DetachableSecurityDevice
描述：本接口表示用于控制同可分离安全设备通信的组件（智能卡等）。

K.5.1.1 方法

K.5.1.1.1 open

原型：public void open() throws org.ngb.net.cas.controller.CADriverException
描述：本方法用于DCAS应用初始化同可分离安全设备间的会话。
参数：无。
返回：无。
异常：org.ngb.net.cas.controller.CADriverException 如果发生驱动错误，抛出此异常。

K.5.1.1.2 close

原型：public void close() throws org.ngb.net.cas.controller.CADriverException
描述：本方法用于DCAS应用关闭用可分离安全设备间的会话。
参数：无。
返回：无。
异常：org.ngb.net.cas.controller.CADriverException 如果发生驱动错误，抛出此异常。

K.5.1.1.3 reset

原型：public byte[] reset() throws org.ngb.net.cas.controller.CADriverException
描述：本方法用于重置可分离安全设备，并返回数据（智能卡时数据为ATR）。
参数：无。
返回：byte数组，存储设备重置返回的数据。
异常：org.ngb.net.cas.controller.CADriverException 如果发生驱动错误，抛出此异常。

K.5.1.1.4 sendData

原型：public void sendData(byte [] data) throws
org.ngb.net.cas.controller.CADriverException
描述：本方法用于DCAS应用向可分离安全设备发送数据。
参数：data - byte数组，需要发送的数据。
返回：无。
异常：org.ngb.net.cas.controller.CADriverException 如果发生驱动错误，抛出此异常。

K.5.1.1.5 registerListener

原型：public void
registerListener(org.ngb.net.cas.detachable.DetachableSecurityDeviceListener
aListener)
描述：本方法用于DCAS应用注册接收可分离安全设备发送数据的监听器
参数：aListener －要注册的org.ngb.net.cas.detachable.DetachableSecurityDeviceListener。
返回：无。

K.5.1.1.6 removeListener

原型：public void removeListener()
描述：本方法用于DCAS应用来删除已注册的监听器。
参数：无。
返回：无。

K.5.2 接口org.ngb.net.cas.detachable.DetachableSecurityDeviceListener

原型：public interface org.ngb.net.cas.detachable.DetachalbeSecurityDeviceListener
描述：本接口应被DCAS应用实现，用于接收可分离安全设备状态和发送的数据。

K.5.2.1 字段

K.5.2.1.1 DEVICE_IN

原型：public static final int DEVICE_IN = 1
描述：用于描述可分离安全设备状态：插入（智能卡时表示智能卡插入）。

K.5.2.1.2 DEVICE_OUT

原型：public static final int DEVICE_OUT = 2
描述：用于描述可分离安全设备状态：拔出（智能卡时表示智能卡拔出）。

K.5.2.1.3 DEVICE_ERROR

原型：public static final int DEVICE_ERROR = 3
描述：用于描述可分离安全设备状态：出错（智能卡时表示智能卡出错）。

K.5.2.2 方法

K.5.2.2.1 receiveDeviceStatus

原型：public void receiveDeviceStatus(int status)
描述：本方法应被DCAS应用实现，用于接收可分离安全设备状态。
可分离安全设备状态变化时通知DCAS应用。
参数：status – int型，可分离安全设备状态。
返回：无。

K.5.2.2.2 receiveData

原型：public void receiveData(byte[] data)
描述：本方法在可分离安全设备向DCAS应用发送数据时被调用。
参数：data – byte数组，可分离安全设备发送的数据。
返回：无。

附 录 L
（规范性附录）
JavaScript-单向广播网络接入单元

L.1 概述

本附录定义了与单向广播网络接入相关的JavaScript接口，主要是调谐解调模块。

L.2 调谐解调模块

本模块定义了与调谐解调相关的JS对象：DvbcTuningParameters、AbsssTuningParameters、DtmbTuningParameters、DvbTune、DvbTunerInfo、DvbScan。其结构关系见图L.1。

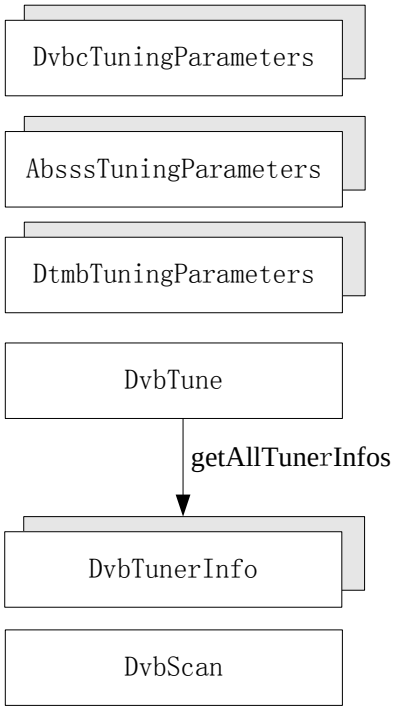


图 L.1 调谐解调模块对象结构图

L.2.1 消息

调谐解调模块可能发给应用层的消息定义见表L.1。

表 L.1　调谐解调模块消息

消息名称	event.which	event.modifiers	消息说明
MSG_DVB_TUNE_SUCCESS	10001	number	锁定成功，消息字符串 JSON 格式为： {“deliveryType”:param1[注1]，“freq”:param2[注2]}
MSG_DVB_TUNE_FAILED	10002	number	锁定失败，消息字符串 JSON 格式为： {“deliveryType”:param1，“freq”:param2}
MSG_DVB_TUNE_SIGNAL_WEAK	10003	number	当前频点信号弱，消息字符串 JSON 格式为： {“deliveryType”:param1，“freq”:param2}
MSG_DVB_TUNE_SIGNAL_LOST	10004	number	当前频点信号丢失，消息字符串 JSON 格式为： {“deliveryType”:param1，“freq”:param2}
MSG_DVB_TUNE_SIGNAL_RECOVER	10005	number	当前频点信号恢复，消息字符串 JSON 格式为： {“deliveryType”:param1，“freq”:param2}
保留	10006～10024	-	
MSG_DVB_SCAN_START	10025	number	频道搜索开始，消息字符串 JSON 格式为： {“deliveryType”:param1}
MSG_DVB_SCAN_FINISHED	10026	number	频道搜索完成，消息字符串 JSON 格式为： {“deliveryType”:param1}
MSG_DVB_SCAN_FAILED	10027	number	频道搜索失败，消息字符串 JSON 格式为： {“deliveryType”:param1} 频道搜索结束后没有搜到任何业务，发送此消息。

MSG_DVB_SCAN_FIND_SERVICES	10028	number	频道搜索发现业务，消息字符串 JSON 格式为：
			{
			“deliveryType”: param1,
			“freq”: param2,
			“serviceCount”: param3
			}
			在结束当前频点搜索时发出该消息。

MSG_DVB_SCAN_STOP_SUCCESS	10029	number	终止搜索成功，消息字符串 JSON 格式为： {“deliveryType”: param1}
MSG_DVB_SCAN_STOP_FAILED	10030	number	终止搜索失败，消息字符串 JSON 格式为： {“deliveryType”: param1}（最好返回失败原因）
保留	10031～10100	-	
MSG_DVB_SCAN_SAVE_SUCCESS	10101	-	保存频道数据到 NVM 成功。
MSG_DVB_SCAN_SAVE_FAILED	10102	-	保存频道数据到 NVM 失败。
MSG_DVB_SCAN_REVERT_SUCCESS	10103	-	从 NVM 导入频道数据成功。
MSG_DVB_SCAN_REVERT_FAILED	10104	-	从 NVM 导入频道数据失败。
MSG_DVB_SCAN_DELETE_SUCCESS	10105	-	清除 NVM 和 RAM 中的频道数据成功。
MSG_DVB_SCAN_DELETE_FAILED	10106	-	清除 NVM 和 RAM 中的频道数据失败。

表 L.1（续）

消息名称	event.which	event. modifiers	消息说明
保留	10107～10200		
event.modifiers 值由系统内部自动给出，其数据类型： ——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。 ——“-”，表示 event.modifiers 为 undefined。 注 1：param1：number 型，取值见“DVB 传送系统类型”常量定义。 注 2：param2：number 型，表示频点频率，若 deliveryType 取值为 10（表示 ABS-SS 传送系统），则单位为兆赫（MHz）；否则单位为千赫（kHz）。 注 3：param3：number 型，表示在由 param2 指定的频点中搜到的业务数量。			

L.2.2 常量

调谐解调模块常量定义见表L.2。

表 L.2 调谐解调模块常量

常量	说明
DVB 传送系统类型	
const DVB_DELIVERY_TYPE_DVB_C = 1;	DVB-C 传送系统。
const DVB_DELIVERY_TYPE_ABS_SS = 10;	ABS-SS 传送系统。
const DVB_DELIVERY_TYPE_DTMB = 12;	DTMB 传送系统。
DVB-C 调制方式	
const DVB_C_MOD_UNDEFINED = 0;	未定义。
const DVB_C_MOD_QAM16 = 1;	16QAM。
const DVB_C_MOD_QAM32 = 2;	32QAM。
const DVB_C_MOD_QAM64 = 3;	64QAM。
const DVB_C_MOD_QAM128 = 4;	128QAM。
const DVB_C_MOD_QAM256 = 5;	256QAM。
ABS-SS 极化方式	
const ABS_SS_POLAR_LINEAR_H = 0;	线极化-水平极化。
const ABS_SS_POLAR_LINEAR_V = 1;	线极化-垂直极化。
const ABS_SS_POLAR_CIRCULAR_L = 2;	圆极化-左旋圆极化。
const ABS_SS_POLAR_CIRCULAR_R = 3;	圆极化-右旋圆极化。
DTMB 调制方式	
const DTMB_MOD_UNDEFINED = 0;	未定义。
const DTMB_MOD_QAM4 = 1;	4-QAM。
const DTMB_MOD_QAM4_NR = 2;	4-QAM-NR。
const DTMB_MOD_QAM16 = 3;	16-QAM。
const DTMB_MOD_QAM32 = 4;	32-QAM。
const DTMB_MOD_QAM64 = 5;	64-QAM。

L.2.3 DvbcTuningParameters对象

DvbcTuningParameters 对象为本地对象，表示适用于 DVB-C 传送系统的调谐解调参数。

示例 1：

```
//创建 DvbcTuningParameters 对象
var dvbcParams = new DvbcTuningParameters();
dvbcParams.frequency = 312000;                  //312.000MHz
dvbcParams.symbol_rate = 27450;                 //27.450Msymbol/s
dvbcParams.modulation = DVB_C_MOD_QAM64;        //64 QAM
```

示例 2：

```
//创建 DvbcTuningParameters 对象
var dvbcParams = new DvbcTuningParameters(312000, 27450, DVB_C_MOD_QAM64);
```

L.2.3.1 属性

DvbcTuningParameters 对象的属性定义见表 L.3。

注： DvbcTuningParameters 属性详见 GB/T 28161—2011 中 cable_delivery_system_descriptor 定义。

表 L.3 DvbcTuningParameters 对象属性

属性名称	类型	读写属性	说明
frequency	number	读/写	表示 DVB-C 信号调谐频率，单位为千赫（kHz）。
symbol_rate	number	读/写	表示 DVB-C 信号符号率，单位为千符号每秒（ksymbol/s）。
modulation	number	读/写	表示 DVB-C 信号调制方式。

L.2.3.2 方法

L.2.3.2.1 DvbcTuningParameters

原型：DvbcTuningParameters()
描述：构造方法，创建一个默认的DVB-C调谐解调参数对象。
参数：无。

L.2.3.2.2 DvbcTuningParameters

原型：DvbcTuningParameters(frequency, symbolRate, modulation)
描述：构造方法，按照指定的参数创建一个DVB-C调谐解调参数对象。
参数：frequency - number型，表示DVB-C信号调谐频率，单位为千赫（kHz）；
symbolRate - number型，表示DVB-C信号符号率，单位为千符号每秒（ksymbol/s）；
modulation - number型，表示DVB-C信号调制方式。

L.2.4 AbsssTuningParameters对象

AbsssTuningParameters 对象为本地对象，表示 ABS-SS 传送系统调谐解调参数。

示例 1：

```
//创建 AbsssTuningParameters 对象:
var absssParams = new AbsssTuningParameters();
absssParams.frequency = 12020;                  //12.020GHz
absssParams.symbol_rate = 28800;                //28.8Msymbol/s
```

```
absssParams.polarization = ABS_SS_POLAR_CIRCULAR_R;            //右旋圆极化
```

示例 2：

```
//创建 AbsssTuningParameters 对象
var absssParams = new AbsssTuningParameters(12020, 28800, ABS_SS_POLAR_CIRCULAR_R);
```

L.2.4.1 属性

AbsssTuningParameters 对象的属性定义见表 L.4。

表 L.4 AbsssTuningParameters 属性表

属性名称	类型	属性	说明
frequency	number	读/写	表示 ABS-SS 信号频率，单位为兆赫（MHz）。
symbol_rate	number	读/写	表示 ABS-SS 信号符号率，单位为千符号每秒（ksymbol/s）。
polarization	number	读/写	表示 ABS-SS 信号极化方式。

L.2.4.2 方法

L.2.4.2.1 AbsssTuningParameters

原型：AbsssTuningParameters()
描述：构造方法。
参数：无。

L.2.4.2.2 AbsssTuningParameters

原型：AbsssTuningParameters(frequency, symbol_rate, polarization)
描述：构造方法。
参数：frequency - number型，ABS-SS信号频率，单位MHz；
symbol_rate - number型，ABS-SS信号符号率，单位ksymbol/s；
polarization - number型，表示ABS-SS信号极化方式。

L.2.5 DtmbTuningParameters对象

DtmbTuningParameters 对象为本地对象，表示 DTMB 传送系统调谐解调参数。

示例 1：

```
//创建 DtmbTuningParameters 对象:
var dtmbParams = new DtmbTuningParameters();
dtmbParams.frequency = 714000;      //714.000MHz
```

示例 2：

```
var dtmbParams = new DtmbTuningParameters(714000);
```

L.2.5.1 属性

DtmbTuningParameters 对象的属性定义见表 L.5。

注： DtmbTuningParameters 属性详见 GB/T 28161—2011 中 terrestrial_delivery_system_descriptor 定义。

表 L.5 DtmbTuningParameters 属性表

属性名称	类型	属性	说明
frequency	number	读/写	表示 DTMB 信号中心频率，单位为千赫（kHz）。
modulation	number	只读	表示 DTMB 信号调制方式。
codingRatio	string	只读	表示 DTMB 信号编码效率，可取值为“0.4”、“0.6”或“0.8”。
PNMode	string	只读	表示 DTMB 信号帧头模式，可取值例如“PN945”。只有成功调谐解调后此属性才有意义。

L.2.5.2 方法

L.2.5.2.1 DtmbTuningParameters

原型：DtmbTuningParameters()
描述：构造方法。
参数：无。

L.2.5.2.2 DtmbTuningParameters

原型：DtmbTuningParameters(frequency)
描述：构造方法。
参数：frequency - number型，表示DTMB信号中心频率，单位为千赫（kHz）；

L.2.6 DvbTune对象

DvbTune对象为内置对象，实现频道调谐和信号解调。定义两种对象，NGBDvbTune()默认tunerId为0，NGBDvbTune(tunerId)由创建对象时指定tunerId。

L.2.6.1 方法

L.2.6.1.1 tune

原型：tune(deliveryType, paramsObj)
描述：异步方法，调谐到指定频点。
——若锁频成功，则向页面发送消息 MSG_DVB_TUNE_SUCCESS；
——若锁频失败，则向页面发送消息 MSG_DVB_TUNE_FAILED；
——若信号弱，则向页面发送消息 MSG_DVB_TUNE_SIGNAL_WEAK；
——若信号丢失，则向页面发送消息 MSG_DVB_TUNE_SIGNAL_LOST；
——若信号恢复，则向页面发送消息 MSG_DVB_TUNE_SIGNAL_RECOVER。
参数：deliveryType - number型，指示paramsObj对象的类型；
paramsObj - 表示调谐解调参数，类型由deliveryType参数指定：
——DVB_DELIVERY_TYPE_DVB_S - paramsObj的类型为DvbsTuningParameters；
——DVB_DELIVERY_TYPE_DVB_C - paramsObj的类型为DvbcTuningParameters；
——DVB_DELIVERY_TYPE_DVB_T - paramsObj的类型为DvbtTuningParameters；
——DVB_DELIVERY_TYPE_ABS_SS - paramsObj的类型为AbsssTuningParameters；
——DVB_DELIVERY_TYPE_DTMB - paramsObj的类型为DtmbTuningParameters。
返回：无。

L.2.6.1.2 getTunerSignalInfo

原型：string getTunerSignalInfo()
描述：同步方法，获取 Tuner 信息，包括信号质量、信号强度、误码率、信号电平、信噪比。
参数：无。
返回：——若获取成功，则会以
“{signalQuality:XXX,signalStrength:XXX,errorRate:XXX,signalLevel:XXX,signalNoiseRatio:XXX}”形式返回描述字符串。
——若获取失败，返回空。

L.2.6.1.3 getAllTunerInfos

原型：DvbTunerInfo[] getAllTunerInfos()
描述：获取当前系统支持的所有Tuner信息。
参数：无。
返回：DvbTunerInfo 对象数组。

L.2.7 DvbTunerInfo对象

DvbTunerInfo对象为本地对象，用于描述TunerId与Tuner类型的匹配关系。

L.2.7.1 属性

DvbTunerInfo 对象的属性定义见表 L.6。

表 L.6 DvbTunerInfo 对象属性

属性名称	类型	读写属性	说明
tunerId	number	读/写	表示 Tuner ID 值。
tunerType	number	读/写	表示对应该 Tuner ID 的 Tuner 类型。

L.2.8 DvbScan对象

DvbScan对象为内置对象，实现频道搜索。
频道搜索方式定义：
——手动搜索：根据设定的调谐解调参数，在单个频点内搜索广播电视节目频道；
——自动搜索：根据运营商指定的起始频道搜索NIT，然后根据NIT的指示自动搜索全网络的广播电视节目频道；运营商有可能会指定多个起始频道，只要在其中任意一个频点成功搜索到NIT，即可进行自动搜索；
——区间搜索：按照我国数字电视频道分配表，在指定的起止频率范围内逐频点搜索广播电视节目。

本附录强制规定频道搜索结果在接收终端中的存储机制，为帮助理解DvbScan对象所提供的频道数据存取方法，本部分给出示例以说明接收终端可能采用的存储机制。

示例：

频道搜索完成后，PSI/SI 数据（除 EIT 表外）将保存在三个存储区域，分别为 A 区、B 区和 C 区，存储区域的划分见图 L.2。

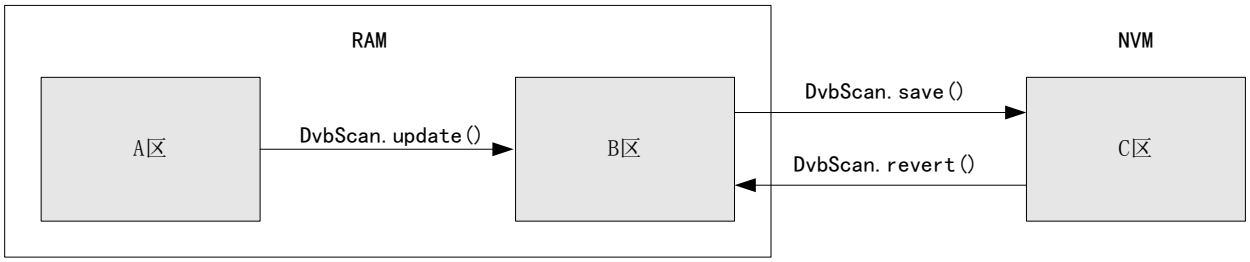


图 L.2　频道搜索结果存储区域示意图

NVM 区域中的存储信息掉电不丢失，该类型的存储器有 Flash、E2PROM、磁盘等。在 NVM 中分配存储区域 C，用以保存接收终端成功搜索到频道数据。

接收终端处于掉电状态时，PSI/SI 数据只保存在 NVM 中。当接收终端上电后，系统将自动从 C 区导入 PSI/SI 数据到 RAM 中的 B 区，供应用调用。

在搜索频道过程中，新搜索到的数据将临时保存在 A 区，搜索完毕待用户确认后，应用调用 DvbScan.update()方法将新搜索到的数据更新到 B 区。应用调用 DvbScan.save()方法将 B 区中的数据保存到 C 区，否则掉电后本次搜索结果将丢失。若应用想放弃新搜索到的结果，可调用 DvbScan.revert()方法将 C 区中旧的数据还原到 B 区。

DvbScan 有两种构造方式，DvbScan（）默认使用 tunerId 为 0，DvbScan(tunerId)可以指定搜索时 tunerId 值。

L.2.8.1　方法

L.2.8.1.1　startScan

原型：startScan(scanType, deliveryType, objArray[])

描述：异步方法，启动频道搜索，搜索方式由scanType参数确定，系统将自动进行调谐和解调。

——搜索开始，将向页面发送消息MSG_DVB_SCAN_START；

——当前频点搜索完毕，若发现业务则向页面发送消息MSG_DVB_SCAN_FIND_SERVICES；

——所有频道全部搜索完毕，则向页面发送消息MSG_DVB_SCAN_FINISHED；

——若没有搜到任何频道，则向页面发送消息MSG_DVB_SCAN_FAILED。

参数：scanType - number型，表示频道搜索方式，取值：

——0 - 表示手动搜索，参数objArray数组的长度为1，表示待搜索频点的调谐解调参数；

——1 - 表示自动搜索，参数objArray数组表示起始频道调谐解调参数，数组的长度大于等于1；只要在其中一个频点成功解析NIT，即可根据NIT的指示自动完成搜索，数组的其他频点参数可忽略；

——2 - 表示区间搜索，参数objArray数组的长度为2，objArray[0]表示区间起点的调谐解调参数，objArray[1]表示区间终点的调谐解调参数；区间设置应符合国家相关规定。

deliveryType- number型，指示objArray对象数组的类型。

objArray[] - 表示调谐解调参数，类型由deliveryType参数指定；

——DVB_DELIVERY_TYPE_DVB_S- objArray的类型为4er；

——DVB_DELIVERY_TYPE_DVB_C- objArray的类型为DvbcTuningParameters；

——DVB_DELIVERY_TYPE_DVB_T- objArray的类型为DvbtTuningParameters；

——DVB_DELIVERY_TYPE_ABS_SS - objArray的类型为AbsssTuningParameters；

——DVB_DELIVERY_TYPE_DTMB - objArray的类型为DtmbTuningParameters。

返回：无。

L.2.8.1.2　startScan

原型：startScan(pid, tableid, deliveryType, objArray[])

描述：异步方法，启动频道搜索，搜索方式由scanType参数确定，系统将自动进行调谐和解调。

——搜索开始，将向页面发送消息MSG_DVB_SCAN_START；

——搜索完毕，则向页面发送消息MSG_DVB_SCAN_FINISHED；

——若没有搜到任何频道，则向页面发送消息MSG_DVB_SCAN_FAILED。

参数：pid - number型，表示需要搜索的指定节目信息数据所在TS包的PID

tableid - number型，表示需要搜索指定的节目信息数据分配的tableid

deliveryType – number型，指示objArray对象数组的类型。

objArray[] - 表示调谐解调参数，类型由deliveryType参数指定；

——DVB_DELIVERY_TYPE_DVB_S– objArray的类型为4er；

——DVB_DELIVERY_TYPE_DVB_C– objArray的类型为DvbcTuningParameters；

——DVB_DELIVERY_TYPE_DVB_T– objArray的类型为DvbtTuningParameters；

——DVB_DELIVERY_TYPE_ABS_SS – objArray的类型为AbsssTuningParameters；

——DVB_DELIVERY_TYPE_DTMB – objArray的类型为DtmbTuningParameters。

返回：无。

L.2.8.1.3 startScan

原型：startScan(string jsonSIInfo)

描述： 将JSON格式的节目信息数据传入DTV组件进行解析保存。

由于仅仅是数据解析，不消耗太多时间，所以采用同步模式。

参数：jsonSIInfo- string型，应用从运营商前端获取的JSON格式的PSI/SI信息，JSON信息详细字段定义见表L.7、表L.8、表L.9、表L.10、表L.11、表L.12、表L.13、表L.14、表L.15和表L.16。

返回：0 - 成功；

-1 - 数据格式错误。

表 L.7 配置表节目信息数据结构定义

字段	类型	说明
Version	Number	节目数据的版本，用于保证终端和前端节目一致。前端初始值为 1，节目数据有任何变动该值加 1，并于 NIT 表中的 version 一致。
OperatorID	Number	运营商编号。
NetWorkID	Number	网络 ID。
OperatorNames	Array	运营商名称。
DeliveryInfos	Object	传输流信息，网络中的频点信息。
Services	Array	节目信息。
Groups	Array	节目分组信息。

表 L.8 运营商名称信息数据结构定义

字段	类型	说明
lang	String	语言类型。
Name	String	运营商名称。

表 L.9　传输信息数据结构定义

字段	类型	说明
DeliveryType	Number	传输类型，1:DVB-C、2:DTMB-T、3:ABS-S。
TSNumb	Number	传输流个数。
Translates	Object	频点信息。

表 L.10　频点信息数据结构定义

字段	类型	说明
TsIndex	Number	传输流序号。
Name	String	传输流名称。
OriginalNetworkID	Number	原始网络 ID
TSId	Number	传输流 ID。
frequency	Number	频率，KHz 为单位。
symboRate	Number	符号率，Kpbs 为单位。
Modulation	Number	调制模式，见 SI 定义：0，保留 1:16QAM，2:32QAM，3:64QAM 4：128QAM，5:256QAM。

表 L.11　节目信息数据结构定义

字段	类型	说明
OriginalNetworkID	Number	原始网络 ID。
TSId	Number	传输流 ID。
ServerID	Number	服务 ID。
TSId	Number	传输流 ID。
RegionCode	Array	区域码。
PcrPid	Number	PCR PID。
ServiceNames	Array	节目名称（多编码格式名称）。
PMTPid	Number	Pmt Pid，保持跟 SDT 表中描述一致。
ServiceType	Number	节目类型，见 SI 定义。
logicNo	Number	逻辑频道号。
SoundChannel	Number	0：立体声、1：左声道、2：右声道、3：单声道。
VolumeOffset	Number	音量补偿值。取值范围为负 5 到正 5。
Video	Object	视频流信息。
Audios	Array	音频流数组。
OtherES	Array	其他数据流信息。

表 L.12　视频流信息数据结构定义

字段	类型	说明
VideoPid	Number	视频流 PID。
VideoType	Number	视频流编码类型。
CAInfo	Array	条件接收信息。

表 L.13　音频流信息数据结构定义

字段	类型	说明
AudioLang	String	语言类型。
AudioPid	Number	音频 PID。
AudioType	Number	音频流编码类型。
CAInfo	Array	条件接收信息。

表 L.14　其他基本流信息数据结构定义

字段	类型	说明
ESType	Number	基本流类型。
Pid	Number	PID。
CAInfo	Array	条件接收信息。

表 L.15　条件接收信息数据结构定义

字段	类型	说明
CASystemID	Number	CA 系统商 ID。
EcmPid	Number	控制字 PID。

表 L.16　节目分组信息数据结构定义

字段	类型	说明
Names	Array	分组名称信息。
GroupID	Number	分组编号。
GroupServices	Array	组内节目编号列表。

示例：

```
TVOS-SI
{
    "Version":0001,
    “OperatorID”:3356,
    “NetWorkID”:1234,
    “OperatorNames ”:[
        {  "Lang":"chi",
```

```
                "Name":"吉林广视",
            },
        ],
        "DeliveryInfo":{
            "DeliveryType":1,
            "TSNumb": 25,
            "Translates":[
                "TsIndex":1,
                "Name":"ss"
                "OriginalNetworkID":223,
                "TSId":123,
                "frequency":256000,
                "symboRate":6875,
                "Modulation":"64QAM",
            ],
        },
    /*  "DeliveryInfo":{
            "DeliveryType":"ABS-S",
            "TSNumb": xxx,
            "Translates":[
                "TsIndex":1,
                "Name":"ss"
                "OriginalNetworkID":223,
                "TSId":123,
                "Polarisation":00
                "frequency":300000,
                "symboRate":6875,
            ],
    */
        "Services":[
             “OrgNetWorkID” :223,
             “TSID” :123
            "ServerID":XXX,
            "RegionCode":[xxx,xxx],
            "PcrPid":2011,
            "ServiceNames":[
                { "lang":"chi",
                    "ServiceName":"吉林卫视",
                    "ProviderName":"吉林电视台",
                    },
                { "lang":"eng",
                    "ServiceName":"jilinweishi",
                    "ProviderName":"jilin",
```

```
            },
        ],
        "PMTPid":xxx,
        "ServiceType":1,
        "logicNo":xxx,
        "SoundChannel":xxx,
        "VolumeOffset":xxx,
        "Video":{
            "VideoPid":xxx,
            "VideoType":1,
            "CAInfo":[
                "CASystemID":XXX
                "EcmPid":xxx,
            ],
        },
        "Audios":[
            {"AudioLang":"ch",
                "AudioPid":xxx,
                "AudioType":1,
                "CAInfo":[
                    "CASystemID":XXX
                    "EcmPid":xxx,
                ],
            },
            {"AudioLang":"eng",
                "AudioPid":xxx,
                "AudioType":1,
                "CAInfo":[
                    "CASystemID":XXX
                    "EcmPid":xxx,
                ],
            },
        ],
        "OtherES":[
            "ESType":xx,
            "Pid":xxx,
            "CAInfo":[
                "CASystemID":XXX
                "EcmPid":xxx,
            ],
        ],
    ],
    "Groups":[
```

```
        {"Names":[
               { "lang":"chi",
                    "name":"高清",
                    },
               ]
          "GroupID":xxx
          "GroupServices":[
                    "ServiceID":123,
                    "ServiceID":125,
               ]
          },
     ],
}
```

L.2.8.1.4 stopScan

原型：stopScan()

描述：异步方法，终止频道搜索。

——若终止成功，则向页面发送 MSG_DVB_SCAN_STOP_SUCCESS 消息；

——若终止失败，则向页面发送 MSG_DVB_SCAN_STOP_FAILED 消息。

参数：无。

返回：无。

L.2.8.1.5 update

原型：number update()

描述：更新 PSI/SI 数据。

示例：

用 A 区中已经成功搜索到的 PSI/SI 数据，更新 B 区中的对应数据，B 区中的其他数据保持不变。

参数：无。

返回：number型，1表示更新成功，0表示更新失败。

L.2.8.1.6 save

原型：save()

描述：异步方法，保存 PSI/SI 数据到 NVM。在搜索完毕后，调用此方法更新 NVM 中的数据，否则重启接收终端后还将保持原有的频道数据。

——若保存成功则向页面发送 MSG_DVB_SCAN_SAVE_SUCCESS 消息；

——若保存失败则向页面发送 MSG_DVB_SCAN_SAVE_FAILED 消息。

示例：

将 RAM 中 B 区的 PSI/SI 数据保存到 NVM 中的 C 区。

参数：无。

返回：无。

L.2.8.1.7 revert

原型：revert()

描述：异步方法，从 NVM 导入 PSI/SI 数据到 RAM 中。

——若导入成功则向页面发送 MSG_DVB_SCAN_REVERT_SUCCESS 消息；

——若导入失败则向页面发送 MSG_DVB_SCAN_REVERT_FAILED。

示例：

删除 B 区中的全部数据后，从 NVM 中的 C 区重新导入 PSI/SI 数据。

参数：无。

返回：无。

L.2.8.1.8 deleteAll

原型：deleteAll()

描述：异步方法，清除 RAM 和 NVM 中的 PSI/SI 数据。

——若清除成功则向页面发送 MSG_DVB_SCAN_DELETE_SUCCESS 消息；

——若清除失败则像页面发送 MSG_DVB_SCAN_DELETE_FAILED 消息。

示例：

清除 B 和 C 区中的 PSI/SI 数据。

参数：无。

返回：无。

附 录 M
（规范性附录）
JavaScript-广播协议处理单元

M.1 概述

本附录定义了与广播协议处理相关的JavaScript接口，主要包含DVB协议模块。

M.2 DVB协议处理模块

本模块定义了与 DVB 广播协议处理相关的 JS 对象：DvbBroadcast、DvbNetwork、DvbBouquet、DvbTS、DvbService、DvbVideoES、DvbAudioES、DvbOtherES，其结构关系见图 M.1。

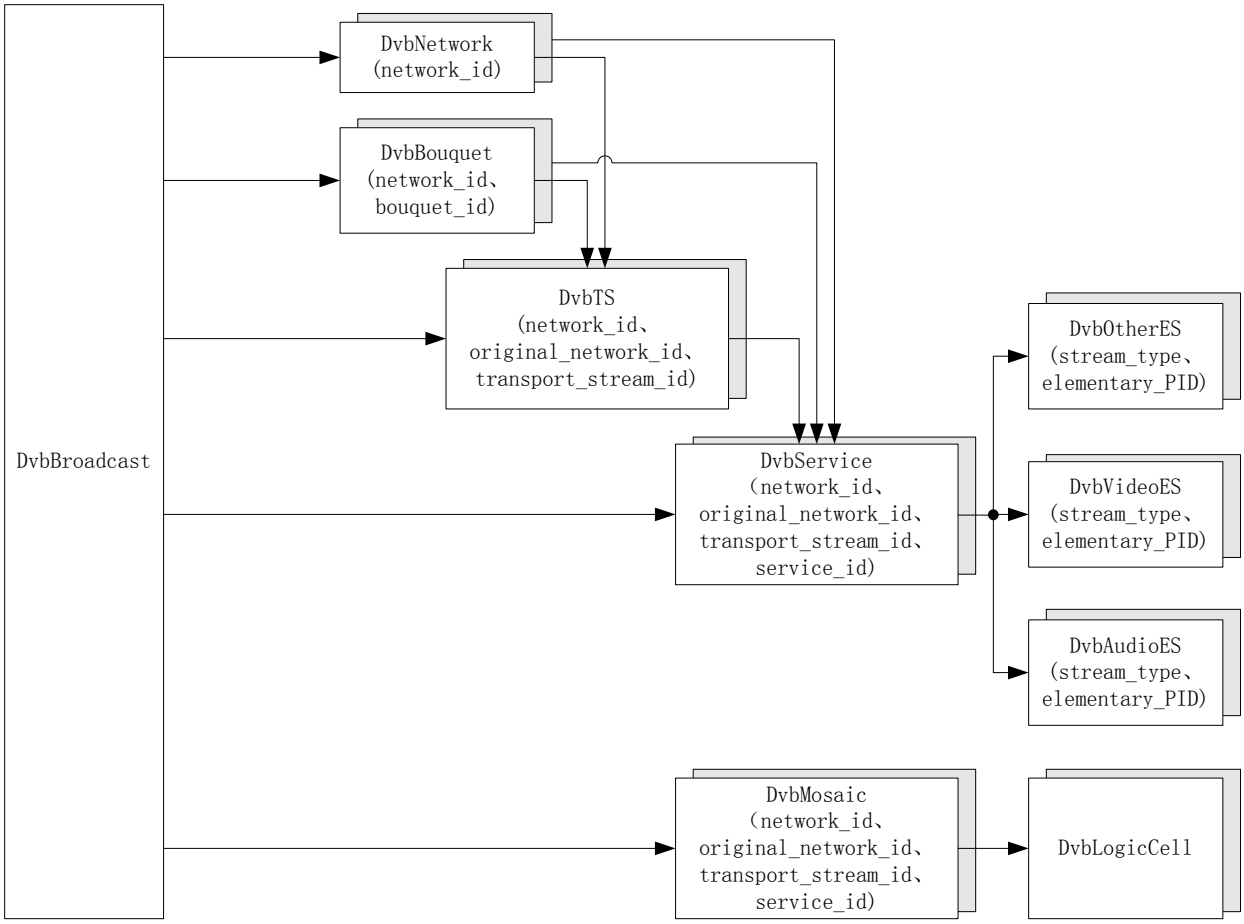


图 M.1 DVB 协议处理模块对象结构图

M.2.1 消息

DVB协议处理模块可能发给应用层的消息定义见表M.1。

表 M.1 DVB 协议处理模块消息定义

消息名称	event.which	event.modifiers	消息说明
MSG_DVB_NIT_CHANGE	11001	number	NIT 版本变更，消息字符串 JSON 格式为：{“oldVersion”:param1，“newVersion”:param2}
MSG_DVB_PAT_CHANGE	11002	number	PAT 版本变更，消息字符串 JSON 格式为：{“oldVersion”:param1，“newVersion”:param2}
保留	11003～11200	-	
event.modifiers 值由系统内部自动给出，其数据类型： ——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。 ——“-”，表示 event.modifiers 为 undefined。			

M.2.2 常量

DVB 协议处理模块的常量定义见表 M.2。

表 M.2 DVB 协议处理模块常量

常量	说明
业务类型	
const DVB_SERVICE_TYPE_DTV = 0x01;	数字电视广播业务
const DVB_SERVICE_TYPE_DAB = 0x02;	数字声音广播业务
const DVB_SERVICE_TYPE_TELETEXT = 0x03;	图文电视业务
const DVB_SERVICE_TYPE_NVOD_REF = 0x04;	NVOD 参考业务
const DVB_SERVICE_TYPE_NVOD_SHIFT = 0x05;	NVOD 时移业务
const DVB_SERVICE_TYPE_MOSAIC = 0x06;	马赛克业务
const DVB_SERVICE_TYPE_DATA = 0x0C;	数据广播业务
const DVB_SERVICE_TYPE_AVC_SD = 0x16;	高级编码标清数字电视（H.264）
const DVB_SERVICE_TYPE_AVC_SD_NVOD_SHIFT = 0x17;	高级编码标清 NVOD 时移业务（H.264）
const DVB_SERVICE_TYPE_AVC_SD_NVOD_REF = 0x18;	高级编码标清 NVOD 参考业务（H.264）
const DVB_SERVICE_TYPE_AVC_HD = 0x19;	高级编码高清数字电视（H.264）
const DVB_SERVICE_TYPE_AVC_HD_NVOD_SHIFT = 0x1A;	高级编码高清 NVOD 时移业务（H.264）
const DVB_SERVICE_TYPE_AVC_HD_NVOD_REF = 0x1B;	高级编码高清 NVOD 参考业务（H.264）
const DVB_SERVICE_TYPE_AVC_3DHD = 0x1C;	高级编码高清帧兼容 3D 数字电视（H.264）
const DVB_SERVICE_TYPE_AVC_3DHD_SHIFT = 0x1D;	高级编码高清帧兼容 3D NVOD 时移业务（H.264）
const DVB_SERVICE_TYPE_AVC_3DHD_REF = 0x1E;	高级编码高清帧兼容 3D NVOD 参考业务（H.264）
排序依据	
const DVB_SORT_TYPE_NETWORK_ID = 0x01;	依据 network_id 排序。
const DVB_SORT_TYPE_BOUQUET_ID = 0x02;	依据 bouquet_id 排序。
const DVB_SORT_TYPE_ONET_ID = 0x03;	依据 original_network_id 排序。

表 M.2（续）

常量	说明
const DVB_SORT_TYPE_TS_ID = 0x04;	依据 transport_stream_id 排序。
const DVB_SORT_TYPE_SERVICE_ID = 0x05;	依据 service_id 排序。
const DVB_SORT_TYPE_SERVICE_TYPE = 0x06;	依据 service_type 排序。
const DVB_SORT_TYPE_SERVICE_NAME = 0x07;	依据 service_name 排序。
const DVB_SORT_TYPE_CHANNEL_ID = 0x10;	依据用户频道号排序。
const DVB_SORT_TYPE_LOGICAL_ID = 0x11;	依据逻辑频道号排序。
const DVB_SORT_TYPE_FTA_SCR = 0x20;	依据是否加密排序。
排列方式	
const DVB_SORT_ORDER_NONE = 0x00;	不进行排列。
const DVB_SORT_ORDER_ASC = 0x01;	——若排序依据为 DVB_SORT_TYPE_FTA_SCR，则表示非加密到加密排列； ——若排序依据为其他，则表示升序排列。
const DVB_SORT_ORDER_DESC = 0x02;	——若排序依据为 DVB_SORT_TYPE_FTA_SCR，则表示加密到非加密排列； ——若排序依据为其他，则表示降序排列。
注：业务类型常量定义引用自 GB/T 28161—2011 表 70“业务类型编码”和 ETSI EN 300 468 v1.12.1(2011-10)表 87“Service type coding”。	

M.2.3 DvbBroadcast对象

DvbBroadcast 对象为内置对象，提供了 DVB 技术体系下 PSI/SI 信息获取的方法。

M.2.3.1 属性

DvbBroadcast 对象的属性定义见表 M.3。

表 M.3 DvbBroadcast 对象属性

属性名称	类型	读写属性	说明
currentDvbNetwork	DvbNetwork	只读	表示当前 DvbNetwork 对象。
currentDvbBouquets	DvbBouquet 数组	只读	表示当前 DvbBouquet 对象数组。
currentDvbTS	DvbTS	只读	表示当前 DvbTS 对象。
currentDvbService	DvbService	只读	表示当前 DvbService 对象。
currentDvbMosaic	DvbMosaic	只读	表示当前 DvbMosaic 对象。

M.2.3.2 方法

M.2.3.2.1 getAllNetworks

原型：DvbNetwork[] getAllNetworks()
描述：获取当前接收终端能接入的所有单向广播网络对象，不进行排序。
参数：无。

返回：DvbNetwork 对象数组。

M.2.3.2.2 getAllNetworks

原型：DvbNetwork[] getAllNetworks(sortOrder)
描述：获取当前接收终端能接入的所有单向广播网络对象，按指定方式排序。
参数：sortOrder - number 型，表示排序方式，取值见表 M.2 中“排序方式”常量定义。
注：只能根据 network_id 进行排序，即排序依据为 DVB_SORT_TYPE_NETWORK_ID，见表 M.2 “排序依据”常量。
返回：DvbNetwork 对象数组。

M.2.3.2.3 getNetwork

原型：DvbNetwork getNetwork(network_id)
描述：获取指定的单向广播网络对象。
参数：network_id - number 型，表示网络 ID。
返回：DvbNetwork 对象。若指定的对象不存在，则返回 null。

M.2.3.2.4 getAllBouquets

原型：DvbBouquet[] getAllBouquets()
描述：获取所有单向广播业务群对象,不进行排序。
参数：无。
返回：DvbBouquet 对象数组。若不存在业务群对象（例如流中未发现 BAT 表），则返回的数组长度为 0。

M.2.3.2.5 getAllBouquets

原型：DvbBouquet[] getAllBouquets(sortTypeArray[], sortOrderArray[])
描述：获取所有单向广播业务群对象，按指定方式排序。
参数：sortTypeArray[] – number型数组，表示DvbBouquet对象排序依据，最多有两个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高；数组的每个成员可取值为DVB_SORT_TYPE_NETWORK_ID和DVB_SORT_TYPE_BOUQUET_ID,见表M.2中“排序依据”常量。
sortOrderArray[] - number型数组，表示排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 “排序方式”常量定义。sortOrderArray与sortTypeArray必须一一对应，即sortOrderArray[i]仅适用于sortTypeArray[i]。
返回：DvbBouquet对象数组。若不存在业务群对象（例如流中未发现BAT表），则返回的数组长度为0。

M.2.3.2.6 getBouquet

原型：DvbBouquet getBouquet(network_id, bouquet_id)
描述：获取指定的单向广播业务群对象。
参数：network_id - number 型，表示网络 ID;
bouquet_id - number 型，表示业务群 ID。
返回：DvbBouquet 对象。若指定的对象不存在，则返回 null。

M.2.3.2.7 getAllTSs

原型：DvbTS[] getAllTSs()
描述：获取所有单向广播传送流对象，不进行排序。
参数：无。
返回：DvbTS 对象数组。

M.2.3.2.8 getAllTSs

原型：DvbTS[] getAllTSs(sortTypeArray[], sortOrderArray[])
描述：获取所有单向广播传送流对象，按指定方式排序。
参数：sortTypeArray[] – number型数组，表示DvbTS对象排序依据，最多有三个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值DVB_SORT_TYPE_NETWORK_ID 、 DVB_SORT_TYPE_ONET_ID 和DVB_SORT_TYPE_TS_ID，见表M.2 中“排序依据”常量定义。
sortOrderArray[] – number型数组，表示排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。sortOrderArray与sortTypeArray必须一一对应，即sortOrderArray[i]仅适用于sortTypeArray[i]。
返回：DvbTS对象数组。
示例：

```
//将DvbTS对象先按照original_network_id升序排序,再按照transport_stream_id降序排序
getAllTSs([DVB_SORT_TYPE_ONET_ID, DVB_SORT_TYPE_TS_ID],
          [DVB_SORT_ORDER_ASC,DVB_SORT_ORDER_DESC]);
```

M.2.3.2.9 getTS

原型：DvbTS getTS(network_id, original_network_id, transport_stream_id)
描述：获取指定的单向广播传送流对象。
参数：network_id – number 型，表示网络 ID。
original_network_id – number 型，表示原始网络 ID;
transport_stream_id – number 型，表示传送流 ID。
返回：DvbTS 对象。若指定的传送流对象不存在，则返回 null。

M.2.3.2.10 getAllServices

原型：DvbService[] getAllServices()
描述：获取所有单向广播业务对象，不进行排序。
参数：无。
返回：DvbService 对象数组。

M.2.3.2.11 getAllServices

原型：DvbService[] getAllServices(sortTypeArray[], sortOrderArray[])
描述：获取所有单向广播业务对象，按指定方式排序。
参数：sortTypeArray[] – number型数组，表示DvbService对象排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量定义。
sortOrderArray[] – number型数组，表示DvbService对象排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。

sortOrderArray 与 sortTypeArray 必须一一对应，即 sortOrderArray[i] 仅适用于 sortTypeArray[i]。

返回：DvbService对象数组。

示例：

```
//将DvbService对象先按照original_network_id升序排序,再按照transport_stream_id降序排序，再按照
//service_id升序排序
getAllServices([DVB_SORT_TYPE_ONET_ID, DVB_SORT_TYPE_TS_ID, DVB_SORT_TYPE_SERVICE_ID],
               [DVB_SORT_ORDER_ASC,DVB_SORT_ORDER_DESC,DVB_SORT_ORDER_ASC]);
```

M.2.3.2.12 getService

原型：DvbService getService (network_id, original_network_id, transport_stream_id, service_id)

描述：获取指定的单向广播业务对象。

参数：network_id – number型，表示网络ID；
original_network_id – number型，表示原始网络ID；
transport_stream_id– number型，表示传送流ID；
service_id – number型，表示业务ID。

返回：DvbService对象。若指定的对象不存在，则返回null。

M.2.3.2.13 getAllMosaics

原型：DvbMosaic[] getAllMosaics()

描述：获取所有马赛克对象，不进行排序。

参数：无。

返回：DvbMosaic 对象数组。若无马赛克对象，则返回的数组长度为 0。

M.2.3.2.14 getEntryMosaic

原型：DvbMosaic getEntryMosaic(network_id)

描述：获取指定网络的入口马赛克对象。

参数：network_id - number 型，表示网络 ID。

返回：DvbMosaic 对象。如无马赛克对象，则返回 null。

M.2.4 DvbNetwork对象

DvbNetwork对象为本地对象，该对象用以描述单向广播网络的相关信息,通过network_id唯一标识。

示例 1：

```
//通过 DvbBroadcast 对象的属性获取 DvbNetwork 对象
var dvbNetwork = DvbBroadcast.currentDvbNetwork;
```

示例 2：

```
//通过 DvbBroadcast 对象的方法，获取 DvbNetwork 对象数组，进而获取单个 DvbNetwork 对象
var dvbNetworkArray[] = DvbBroadcast.getAllNetworks(sortOrder);
var dvbNetwork = dvbNetworkArray[i];
```

示例 3：

```
//通过 DvbBroadcast 对象的方法，获取指定的 DvbNetwork 对象
var dvbNetwork = DvbBroadcast.getNetwork(network_id);
```

M.2.4.1 属性

DvbNetwork 对象的属性定义见表 M.4。

表 M.4 DvbNetwork 对象属性

属性名称	类型	读写属性	说明
network_id	number	只读	表示单向广播网络的网络标识，用以区别其他的单向广播网络。
network_name	string	只读	表示单向广播网络的网络名称。
network_type	number	只读	表示单向广播网络的传送类型。

M.2.4.2 方法

M.2.4.2.1 getNetworkName

原型：string getNetworkName()

描述：获取网络名称全称。网络名称全称从 network_name_descriptor 描述符或者从 multilingual_network_name_descriptor 描述符中获取。本接口应返回与用户设置的偏好语种相匹配的网络名称全称，若从 network_name_descriptor 或 multilingual_network_name_descriptor 描述符中无法获得该语种编码的网络名称全称，则默认返回 network_name_descriptor 中携带的网络名称全称。

参数：无。

返回：string 型，表示网络名称全称。

M.2.4.2.2 getShortNetworkName

原型：string getShortNetworkName()

描述：获取网络名称简称。

参数：无。

返回：string 型，表示网络名称简称，若无网络名称简称则返回 undefined。

示例：“[0x86]Asterix[0x87] Digital Satellite TV Network”

网络名称全称：“Asterix Digital Satellite TV Network”。

网络名称简称：“Asterix”。

M.2.4.2.3 getAllTSs

原型：DvbTS[] getAllTSs()

描述：获取当前网络对象中的所有 DvbTS 对象，不进行排序。

参数：无。

返回：DvbTS 对象数组。

M.2.4.2.4 getAllTSs

原型：DvbTS[] getAllTSs(sortTypeArray[], sortOrderArray[])

描述：获取当前网络对象中的所有DvbTS对象，并按照指定方式进行排序。

参数：sortTypeArray – number型数组，表示排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量定义。

sortOrderArray - number型数组，表示排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。sortOrderArray与sortTypeArray必须一一对应，即sortOrderArray[i]仅适用于sortTypeArray[i]。

返回：DvbTS对象数组。

示例：

```
//将DvbTS对象先按照original_network_id升序排序,再按照transport_stream_id降序排序
getAllTSs([DVB_SORT_TYPE_ONET_ID, DVB_SORT_TYPE_TS_ID],
          [DVB_SORT_ORDER_ASC,DVB_SORT_ORDER_DESC]);
```

M.2.4.2.5 getTS

原型：DvbTS getTS(original_network_id, transport_stream_id)

描述：获取当前网络对象中指定的 DvbTS 对象。

参数：original_network_id - number 型，表示原始网络 ID；
transport_stream_id - number 型，表示传送流 ID。

返回：DvbTS 对象，若指定的对象不存在，则返回 null。

M.2.4.2.6 getAllServices

原型：DvbService[] getAllServices()

描述：获取当前网络对象中的所有单向广播业务对象，不进行排序。

参数：无。

返回：DvbService 对象数组。

M.2.4.2.7 getAllServices

原型：DvbService[] getAllServices(sortTypeArray[], sortOrderArray[])

描述：获取当前网络对象中的所有单向广播业务对象，按指定方式排序。

参数：sortTypeArray[] – number型数组，表示DvbService对象排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量。
sortOrderArray[] - number型数组，表示DvbService对象排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量。sortOrderArray 与 sortTypeArray 必须一一对应，即 sortOrderArray[i] 仅适用于 sortTypeArray[i]。

返回：DvbService对象数组。

示例：

```
//将DvbService对象先按照original_network_id升序排序,再按照transport_stream_id降序排序,
//再按照service_id升序排序
getAllServices([DVB_SORT_TYPE_ONET_ID, DVB_SORT_TYPE_TS_ID, DVB_SORT_TYPE_SERVICE_ID],
               [DVB_SORT_ORDER_ASC,DVB_SORT_ORDER_DESC,DVB_SORT_ORDER_ASC]);
```

M.2.4.2.8 getService

原型：DvbService getService(original_network_id, transport_stream_id, service_id)

描述：获取当前网络对象中指定的单向广播业务对象。

参数：original_network_id - number 型，表示原始网络 ID；

transport_stream_id - number 型，表示传送流 ID；
service_id - number 型，表示业务 ID。

返回：DvbService 对象，若指定的对象不存在，则返回 null。

M.2.5 DvbBouquet对象

DvbBouquet对象为本地对象，用以描述单向广播业务群信息,通过network_id和bouquet_id唯一标识。

示例 1：

```
//通过 DvbBroadcast 对象的方法，获取 DvbBouquet 对象数组，进而获取单个 DvbBouquet 对象
var dvbBouquetArray[] = DvbBroadcast.getAllBouquets(sortTypeArray[], sortOrderArray[]);
var dvbBouquet = dvbBouquetArray[i];
```

示例 2：

```
//通过 Broadcast 对象的方法，获取指定的 DvbBouquet 对象
var dvbBouquet = DvbBroadcast.getBouquet(network_id, bouquet_id);
```

示例 3：

```
//通过 Broadcast 对象的属性，获取当前 DvbBouquet 对象数组
var dvbBouquets[] = DvbBroadcast.currentDvbBouquets;
```

M.2.5.1 属性

DvbBouquet 对象的属性定义见表 M.5。

表 M.5 DvbBouquet 属性表

属性名称	类型	读写属性	说明
network_id	number	只读	表示单向广播网络标识。
bouquet_id	number	只读	表示单向广播业务群标识。
bouquet_name	string	只读	表示单向广播业务群名称。

M.2.5.2 方法

M.2.5.2.1 getBouquetName

原型：string getBouquetName()

描述：获取业务群名称全称。业务群名称全称从 bouquet_name_descriptor 描述符或从 multilingual_bouquet_name_descriptor 描述符中获取。本接口应返回与用户设置的偏好语种相匹配的业务群名称全称，若从 bouquet_name_descriptor 或 multilingual_bouquet_name_descriptor 描述符中无法获得该语种编码的业务群名称全称，则默认返回 bouquet_name_descriptor 中携带的业务群名称全称。

参数：无。

返回：string 型，表示业务群名称全称。

M.2.5.2.2 getShortBouquetName

原型：string getShortBouquetName()

描述：获取业务群名称简称。

参数：无。

返回：string 型，表示业务群名称简称，若业务群名称简称不存在则返回 undefined。

M.2.5.2.3 getAllTSs

原型：DvbTS[] getAllTSs()
描述：获取当前业务群对象中的所有 DvbTS 对象，不进行排序。
参数：无。
返回：DvbTS 对象数组。

M.2.5.2.4 getAllTSs

原型：DvbTS[] getAllTSs(sortTypeArray[], sortOrderArray[])
描述：获取当前业务群对象中的所有DvbTS对象，并按照指定方式进行排序。
参数：sortTypeArray – number型数组，表示排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量定义。
sortOrderArray – number型数组，表示排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。sortOrderArray与sortTypeArray必须一一对应，即sortOrderArray[i]仅适用于sortTypeArray[i]。
返回：DvbTS对象数组。

M.2.5.2.5 getTS

原型：DvbTS getTS(original_network_id, transport_stream_id)
描述：获取当前业务群对象中指定的 DvbTS 对象。
参数：original_network_id – number 型，表示原始网络 ID;
transport_stream_id – number 型，表示传送流 ID。
返回：DvbTS 对象。若指定的对象不存在，则返回 null。

M.2.5.2.6 getAllServices

原型：DvbService[] getAllServices()
描述：获取当前业务群中所有 DvbService 对象，不进行排序。
参数：无。
返回：DvbService 对象数组。

M.2.5.2.7 getAllServices

原型：DvbService[] getAllServices(sortTypeArray[], sortOrderArray[])
描述：获取当前业务群中所有DvbService对象，并按指定方式排序。
参数：sortTypeArray[] – number型数组，表示DvbService对象排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量定义。
sortOrderArray[] – number型数组，表示DvbService对象排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。sortOrderArray 与 sortTypeArray 必须一一对应，即 sortOrderArray[i] 仅适用于 sortTypeArray[i]。
返回：DvbService对象数组。

M.2.5.2.8 getService

原型：DvbService getService(original_network_id, transport_stream_id, service_id)
描述：获取当前业务群中指定的 DvbService 对象。
参数：original_network_id - number 型，表示原始网络 ID；
transport_stream_id - number 型，表示传送流 ID；
service_id - number 型，表示业务 ID。
返回：DvbService 对象。若指定的对象不存在，则返回 null。

M.2.6 DvbTS对象

DvbTS对象为本地对象，用以描述单向广播传送流相关信息，通过network_id、original_network_id和transport_stream_id唯一标识。

示例 1：

```
//通过 DvbBroadcast 的属性，获取当前 DvbTS 对象
 var dvbTS = DvbBroadcast.currentTS;
```

示例 2：

```
 //通过 DvbBroadcast 对象的方法，获取 DvbTS 对象数组，进而获得单个 DvbTS 对象
 var dvbTSArray[] = DvbBroadcast.getAllTSs(sortTypeArray[], sortOrderArray[]);
 var dvbTS = dvbTSArray[i];
```

示例 3：

```
 //通过 DvbBroadcast 对象的方法获取指定的 DvbTS 对象
 var dvbTS = DvbBroadcast.getTS(network_id, original_network_id, transport_stream_id);
```

示例 4：

```
 //通过 DvbNetwork 对象的方法，获取 DvbTS 对象数组，进而获得单个 DvbTS 对象
 var dvbTSArray[] = dvbNetwork.getAllTSs(sortTypeArray[], sortOrderArray[]);
 var dvbTS = dvbTSArray[i];
```

示例 5：

```
 //通过 DvbNetwork 对象的方法，获取指定的 DvbTS 对象
 var dvbTS = dvbNetwork.getTS(original_network_id, transport_stream_id);
```

示例 6：

```
 //通过 DvbBouquet 对象的方法，获取 DvbTS 对象数组，进而获得单个 DvbTS 对象
 var dvbTSArray[] = dvbBouquet.getAllTSs(sortTypeArray[], sortOrderArray[]);
 var dvbTS = dvbTSArray[i];
```

示例 7：

```
 //通过 DvbBouquet 对象的方法，获取指定的 DvbTS 对象
 var dvbTS = dvbBouquet.getTS(original_network_id, transport_stream_id);
```

M.2.6.1 属性

DvbTS 对象的属性定义见表 M.6。

表 M.6 DvbTS 对象属性

属性名称	类型	读写属性	说明
network_id	number	只读	表示传送流所属的网络 ID。
original_network_id	number	只读	表示传送流的原始网络 ID。
transport_stream_id	number	只读	表示传送流 ID。

表 M.6（续）

属性名称	类型	读写属性	说明
deliveryType	number	只读	表示传送系统类型，取值见 “DVB 传送系统类型”常量定义。
dvbsTuningParams	DvbsTuningParameters	只读	表示 DVB-S 调谐解调参数，当 deliveryType=DVB_DELIVERY_TYPE_DVB_S 时，此属性有效。
dvbcTuningParams	DvbcTuningParameters	只读	表示 DVB-C 调谐解调参数，当 deliveryType=DVB_DELIVERY_TYPE_DVB_C 时，此属性有效。
dvbtTuningParams	DvbtTuningParameters	只读	表示 DVB-T 调谐解调参数，当 deliveryType=DVB_DELIVERY_TYPE_DVB_T 时，此属性有效。
abssTuningParams	AbsssTuningParameters	只读	表示 ABS-SS 调谐解调参数，当 deliveryType=DVB_DELIVERY_TYPE_ABS_SS 时，此属性有效。
dtmbTuningParams	DtmbTuningParameters	只读	表示 DTMB 调谐解调参数，当 deliveryType=DVB_DELIVERY_TYPE_DTMB 时，此属性有效。
signalQuality	number	只读	表示传送流所在频点的信号质量，取值范围为 0～100，0 表示信号最差，100 表示信号最好。
signalStrength	number	只读	表示传送流所在频点的信号强度，取值范围为 0～100，0 表示信号最弱，100 表示信号最强。
errorRate	string	只读	表示传送流所在频点的误码率。
signalLevel	string	只读	表示传送流所在频点的信号电平。
signalNoiseRatio	string	只读	表示传送流所在频点的信噪比。

M.2.6.2 方法

M.2.6.2.1 getAllServices

原型：DvbService[] getAllServices()
描述：获取当前传送流中的所有 DvbService 对象，不进行排序。
参数：无。
返回：DvbService 对象数组。

M.2.6.2.2 getAllServices

原型：DvbService[] getAllServices(sortTypeArray[], sortOrderArray[])
描述：获取当前传送流中的所有DvbService对象，并按照指定方式进行排序。
参数：sortTypeArray[] – number型数组，表示DvbService对象排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。数组的每个成员可取值见表M.2 中“排序依据”常量定义。

sortOrderArray[] - number型数组，表示DvbService对象排序方式，数组的长度与sortTypeArray参数一致，数组的每个成员可取值见表M.2 中“排序方式”常量定义。sortOrderArray 与 sortTypeArray 必须一一对应，即 sortOrderArray[i] 仅适用于sortTypeArray[i]。

返回：DvbService对象数组。

M.2.6.2.3 getServiceByID

原型：DvbService getServiceByID(service_id)

描述：获取当前传送流中指定的 DvbService 对象。

参数：service_id - number 型，表示业务 ID。

返回：DvbService 对象。若指定的对象不存在，则返回 null。

M.2.6.2.4 getServicesByType

原型：DvbService[] getServicesByType(service_type)

描述：获取当前传送流中指定业务类型的所有 DvbService 对象。

参数：service_type - number 型，表示业务类型，取值见表 M.2 中“业务类型”常量定义。

返回：DvbService 对象数组。若指定的业务对象不存在，则返回的数组长度为 0。

M.2.7 DvbService对象

DvbService 对象为本地对象，用以描述单向广播业务相关信息，通过 network_id、original_network_id、transport_stream_id和service_id唯一标识。

示例 1：

```
//通过 DvbBroadcast 对象的属性，获取当前 DvbService 对象。
 var dvbService = DvbBroadcast.currentDvbService;
```

示例 2：

```
//通过 DvbBroadcast 对象的方法，获取 DvbService 对象数组，进而获取单个 DvbService 对象。
var dvbServiceArray = DvbBroadcast.getAllServices(sortTypeArray[], sortOrderArray[]);
var dvbService = dvbServiceArray[i];
```

示例 3：

```
//通过 DvbNetwork 对象的方法，获取 DvbService 对象数组，进而获取单个 DvbService 对象。
var dvbServiceArray = dvbNetwork.getAllServices(sortTypeArray[], sortOrderArray[]);
var dvbService = dvbServiceArray[i];
```

示例 4：

```
//通过 DvbNetwork 对象的方法，获取指定的 DvbService 对象。
var dvbService = dvbNetwork.getService(original_network_id, transport_stream_id, service_id);
```

示例 5：

```
//通过 DvbBouquet 对象的方法，获取 DvbService 对象数组，进而获取单个 DvbService 对象。
var dvbServiceArray = dvbBouquet.getAllServices(sortTypeArray[], sortOrderArray[]);
var dvbService = dvbServiceArray[i];
```

示例 6：

```
//通过 DvbBouquet 对象的方法，获取指定的 DvbService 对象数组。
var dvbService = dvbBouquet.getService(original_network_id, transport_stream_id, service_id);
```

示例 7：

```
//通过 DvbTS 对象的方法，获取 DvbService 对象数组，进而获取单个 DvbService 对象。
var dvbServiceArray = dvbTS.getAllServices(sortTypeArray[], sortOrderArray[]);
var dvbService = dvbServiceArray[i];
```

示例 8：

```
//通过 DvbTS 对象的方法，获取指定的 DvbService 对象。
var dvbService = dvbTS.getServiceByID(service_id);
```

示例 9：

```
//通过 DvbTS 对象的方法，获取指定的 DvbService 对象数组，进而获取单个 DvbService 对象。
var dvbServiceArray = dvbTS.getServicesByType(service_type);
var dvbService = dvbServiceArray[i];
```

M.2.7.1 属性

DvbService 对象的属性定义见表 M.7。

表 M.7 DvbService 对象属性

属性名称	类型	读写属性	说明
network_id	number	只读	表示业务所在的网络 ID。
original_network_id	number	只读	表示业务所在的原始网络 ID。
transport_stream_id	number	只读	表示业务所在传送流 ID。
service_id	number	只读	表示业务 ID。
service_name	string	只读	表示业务名称。
service_type	number	只读	表示业务类型，取值见 “业务类型”常量定义。
service_provider_name	string	只读	表示业务提供者名称。
running_status	number	只读	表示业务运行状态，取值： ——0 - 未定义； ——1 - 未运行； ——2 - 几秒后开始（例如录像）； ——3 - 暂停； ——4 - 运行； ——5～7 - 预留使用。 **注**：对于 NVOD 业务，running_status=0。
EIT_present_following_flag	boolean	只读	描述了业务的当前/后续信息是否存在于当前传送流中，取值： ——true - 表示业务的 EIT 当前/后续信息存在于当前传送流中； ——false- 表示业务的 EIT 当前/后续信息不在当前传送流中。
EIT_schedule_flag	boolean	只读	描述了业务的 EIT 时间表信息是否存在于当前传送流中，取值： ——true – 表示业务的 EIT 时间表信息存在于当前传送中。 ——false - 表示业务的 EIT 时间表信息不在当前传送流中。

表 M.7 （续）

属性名称	类型	读写属性	说明
free_CA_mode	boolean	只读	描述了业务是否被加扰，取值： ——true - 一路或多路码流的接收由 CA 系统控制； ——false - 业务的所有组件都未被加扰。
bouquetIDs	number Array	只读	获取业务所属的所有业务群 ID 数组，并按照 bouquet_id 升序排列。
referServiceID	number	只读	若 service_type=SERVICE_TYPE_NVOD_SHIFT,可通过该属性获取对应的参考业务的 service_id，否则该属性返回 undefined。
timeShiftServiceIDs	number Array	只读	若 service_type= SERVICE_TYPE_NVOD_REF，可通过该属性获取对应的所有时移业务的service_id,否则该属性返回undefined。
curVideoObj	DvbVideo ES	只读	表示业务当前正在播放的视频 ES。
curAudioObj	DvbAudio ES	只读	表示业务当前正在播放的音频 ES。
PCRPID	number	只读	表示业务参考的 PCR PID。

M.2.7.2 方法

M.2.7.2.1 getServiceName

原型：string getServiceName()

描述：获取业务名称全称。业务名称全称从 service_descriptor 描述符或从 multilingual_service_name_descriptor 描述符中获取。本接口应返回与用户设置的偏好语种相匹配的业务名称全称,若从service_descriptor或multilingual_service_name_descriptor描述符中无法获得该语种编码的业务名称全称，则默认返回 service_descriptor 中携带的业务名称全称。

参数：无。

返回：string 型，表示业务名称全称。

M.2.7.2.2 getShortServiceName

原型：string getShortServiceName()

描述：获取业务名称简称。

参数：无。

返回：string 型，表示业务名称简称，若业务名称简称不存在，则返回 undefined。

示例：

The [0x86]**P**[0x87]ay [0x86]**M**[0x87]ovie [0x86]**C**[0x87]hannel

业务名称全称——“The Pay Movie Channel”。

业务名称简称——“PMC”。

M.2.7.2.3 getLocation

原型：string getLocation()

描述：获取以 original_network_id、transport_stream_id 和 service_id 三要素描述的业务路径。

参数：无。
返回：string 型。

M.2.7.2.4 getEvents

原型：DvbEvent[] getEvents(start, end)
描述：获取该业务指定时间范围内的 DvbEvent 对象。
参数：start - string 型，表示起始时间，格式为“YYYY-MM-DD hh:mm:ss”；
end - string 型，表示终止时间，格式为“YYYY-MM-DD hh:mm:ss”。
返回：DvbEvent 对象数组。若指定的对象不存在，则返回的数组长度为 0。

M.2.7.2.5 getVideoESs

原型：DvbVideoES[] getVideoESs()
描述：获取该业务包含的所有视频流（包含任意的视频格式类型）。
参数：无。
返回：DvbVideoES 对象数组。若无此对象，则返回的数组长度为 0。

M.2.7.2.6 getAudioESs

原型：DvbAudioES[] getAudioESs()
描述：获取该业务包含的所有音频流（包含任意的音频格式类型）。
参数：无。
返回：DvbAudioES 对象数组。若无此对象，则返回的数组长度为 0。

M.2.7.2.7 getOtherESs

原型：DvbOtherES[] getOtherESs()
描述：获取该业务包含的所有非视音频流。
参数：无。
返回：DvbOtherES 对象数组。若无此对象，则返回的数组长度为 0。

M.2.8 DvbVideoES对象

DvbVideoES 对象为本地对象，用以描述某个业务中的视频基本流信息。

M.2.8.1 属性

DvbVideoES对象的属性定义见表M.8。

表 M.8 DvbVideoES 属性表

属性名称	类型	读写属性	说明
stream_type	number	只读	表示视频基本流的类型，取值： ——0x01 – GB/T 17191.2 视频； ——0x02 – ISO/IEC 13818-2 视频或 GB/T 17191.2 受限参数视频流。
elementary_PID	number	只读	获取承载该视频基本流的传送流 PID。
component_tag	number	只读	表示组件标识。 系统应从与该基本流相关的 stream_identifier_descriptor 获取信息，若无可用信息，则默认置为-1。

M.2.9 DvbAudioES对象

DvbAudioES 对象为本地对象，用以描述某个业务中的音频基本流信息。

M.2.9.1 属性

DvbAudioES对象的属性定义见表M.9。

表 M.9 DvbAudioES 属性表

属性名称	类型	读写属性	说明
stream_type	number	只读	表示音频基本流的类型，取值： ——0x03 – GB/T 17191.3 音频； ——0x04 – ISO/IEC 13818-3 音频。
elementary_PID	number	只读	表示承载该音频基本流的传送流 PID。
component_tag	number	只读	表示组件标识。 系统应从与该基本流相关的 stream_identifier_descriptor 获取信息，若无可用信息，则默认置为-1。
lingual	string	只读	表示音频语种，语种三字母代码遵循 GB/T 4880.2—2000 标准。

M.2.10 DvbOtherES对象

DvbOtherES 对象为本地对象，用以保存某个业务中除视音频之外的基本流信息。

M.2.10.1 属性

DvbOtherES对象的属性定义见表M.10。

表 M.10 DvbOtherES 属性表

属性名称	类型	读写属性	说明
stream_type	number	只读	表示基本流的类型，取值除 0x01、0x02、0x03 和 0x04 以外其他允许的值。
elementary_PID	number	只读	表示承载该基本流的传送流 PID。
component_tag	number	只读	表示组件标识。 系统应从与该基本流相关的 stream_identifier_descriptor 获取信息，若无可用信息，则默认置为-1。

M.2.11 DvbEvent对象

DvbEvent对象为本地对象，用以保存数字电视广播和声音广播频道中的节目事件信息，通过network_id、original_network_id、transport_stream_id、service和event_id唯一标识。

示例：

```
//可通过以下方式获取 DvbEvent 对象。
var dvbEventArray[] = dvbService.getEvents(startDate, endDate);
```

M.2.11.1 属性

DvbEvent 对象的属性定义见表 M.11。

表 M.11 DvbEvent 属性表

属性名称	类型	读写属性	说明
network_id	number	只读	表示节目事件所属的网络 ID。
original_network_id	number	只读	表示节目事件所属的原始网络 ID。
transport_stream_id	number	只读	表示节目事件所属的传送流 ID。
service_id	number	只读	表示节目事件所属的业务 ID。
event_id	number	只读	表示节目事件 ID。
event_name	string	只读	表示节目事件名称。
event_description	string	只读	表示节目事件描述。
running_status	number	只读	表示节目事件运行状态，取值： ——0 - 未定义； ——1 - 未运行； ——2 - 几秒后开始（例如录像）； ——3 - 暂停； ——4 - 运行； ——5～7 预留使用。 注：对于 NVOD 参考节目事件，running_status=0。
startDate	string	只读	表示该事件播放的起始日期，格式为"YYYY-MM-DD"。
startTime	string	只读	表示该事件播放的起始时间，格式为"hh:mm:ss"。
duration	number	只读	表示该事件播放的持续时间，单位为秒。
endDate	string	只读	表示该事件播放的结束日期，格式为"YYYY-MM-DD"。
endTime	string	只读	表示该事件播放的结束时间，格式为"hh:mm:ss"。
content_nibble	number	只读	表示该事件分类值（8 比特），其中高 4 位为一级节目内容分类值(content_nibble_level_1)，低 4 位为二级节目内容分类值(content_nibble_level_2)。
user_nibble	number	只读	表示该事件运营商定义的分类值（8 比特），其中高 4 位为一级节目内容分类值(content_nibble_level_1)，低 4 位为二级节目内容分类值(content_nibble_level_2)。
minAge	number	只读	表示该事件可以收看的最小年龄。
free_CA_mode	boolean	只读	表示该事件是否被加扰，取值： ——true - 已加扰； ——false - 未加扰。

M.2.11.2 方法

M.2.11.2.1 getEventName

原型：string getEventName()
描述：获取节目事件名称全称。
参数：无。
返回：string 型，表示节目事件名称全称。

M.2.11.2.2 getShortEventName

原型：string getShortEventName()
描述：获取节目事件名称简称。
参数：无。
返回：string 型，表示节目事件名称简称。若事件名称简称不存在则返回 undefined。

M.2.11.2.3 getEventDescription

原型：string getEventDescription()
描述：获取节目事件描述信息。节目事件描述信息从 short_event_descriptor 描述中获取，本方法应返回与用户设置的偏好语种相匹配的节目事件描述信息。
参数：无。
返回：string 型，表示节目事件描述。

M.2.11.2.4 getLocation

原型：string getLocation()
描述：获取以 original_network_id、transport_stream_id、service_id 和 event_id 四要素描述的事件定位符。
参数：无。
返回：string 型，表示节目事件定位符。

附 录 N
（规范性附录）
JavaScript-双向宽带网络接入单元

N.1 概述

本附录定义了与双向宽带网络接入控制相关的功能模块：宽带网络设置。

N.2 宽带网络设置模块

本模块定义了与宽带网络设置相关的 JS 对象：Broadband、Ethernet、IP、Proxy，其关系示意见图N.1。

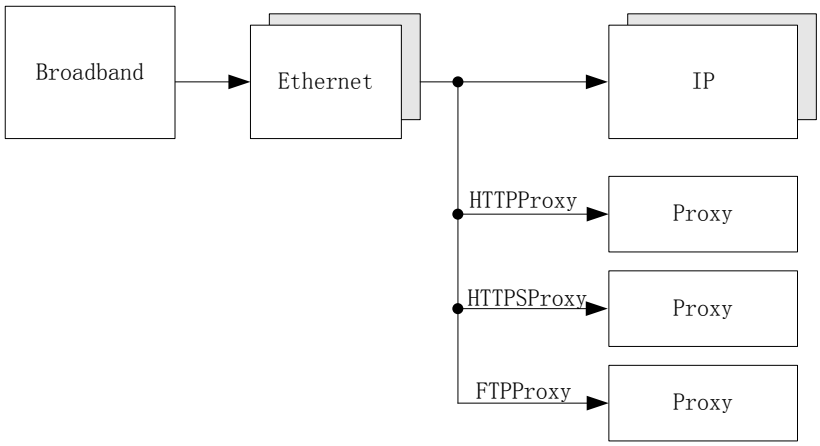


图 N.1 宽带网络设置模块的对象关系示意图

N.2.1 消息

宽带网络设置模块可能发给应用层的消息定义见表N.1。

表 N.1 宽带网络设置模块的消息

消息名称	event.which	event.modifiers	消息说明
MSG_BROADBAND_SAVE_CFG_SUCCESS	12001	-	网络配置信息写入 NVM 成功。
MSG_BROADBAND_SAVE_CFG_FAILED	12002	-	网络配置信息写入 NVM 失败。
MSG_BROADBAND_SUBMIT_SUCCESS	12003	-	提交网络配置参数成功。
MSG_BROADBAND_SUBMIT_FAILED	12004	-	提交网络配置参数失败。
MSG_BROADBAND_NTP_READY_SUCCESS	12005	-	网络 NTP 时间同步成功。
MSG_BROADBAND_NTP_SYNC_TIMEOUT	12006	-	网络 NTP 时间同步超时。
保留	12007～12014		
MSG_BROADBAND_DHCP_READY_SUCCESS	12015	-	网络 DHCP 功能启用成功。
MSG_BROADBAND_DHCP_TIMEOUT	12016	-	网络 DHCP 功能启用超时。

表 N.1(续)

消息名称	event.which	event.modifiers	消息说明
保留	12017～12027		
MSG_BROADBAND_PING_RESPONSE	12028	number	PING 命令响应。

MSG_BROADBAND_GET_NETWORK_STATE	12029	number	网络状况，消息描述字符串 JSON 格式为：
			{
			“targetAddress”:param1 注1,
			“lostPacketRate”:param2 注2,
			“bandwidth”:param3 注3,
			“delay”:param4 注4
			}

MSG_BROADBAND_LINK_AP_SUCCESS	12030	-	无线 AP 连接成功。
MSG_BROADBAND_LINK_AP_FAILED	12031	-	无线 AP 连接失败。
MSG_BROADBAND_SCAN_AP_FIND	12032	number	每成功搜索到一个无线热点，发送该消息。
MSG_BROADBAND_SCAN_AP_SUCCESS	12033	number	完成指定个数无线热点搜索时，发送该消息。
MSG_BROADBAND_SCAN_AP_FAILED	12034	-	在指定的超时时间内没有搜到任何无线热点,发送该消息。
保留	12035～12055		
MSG_BROADBAND_DISCONNECTED	12056	-	网线断开。
MSG_BROADBAND_CONNECTED	12057	-	网线插上。
MSG_BROADBAND_LAN_DISCONNECTED	12058	-	当网络断开时发送该消息。
MSG_BROADBAND_LAN_CONNECTED	12059	-	当网络连接时发送该消息。
MSG_BROADBAND_IP_RENEW	12060	number	在 DHCP 模式下，IP 地址自动更新。
保留	12061～12500		

event.modifiers 值由系统内部自动给出，其数据类型：

——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。

——“-”，表示 event.modifiers 为 undefined。

注1：param1：string型，表示目标IP地址。

注2：param2：number型，表示丢包率。

注3：param3：number型，表示带宽，单位为兆比特每秒（Mbps）。

注4：param2：number型，表示延时，单位为毫秒。

N.2.2 Broadband对象

Broadband 对象为内置对象，描述了双向宽带的属性及网络配置信息。

N.2.2.1 属性

Broadband 对象的属性定义见表 N.2。

表 N.2 Broadband 对象属性

属性名称	类型	属性	说明
portalIP	string	只读	表示门户服务器的 IP 地址，格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。
portalPort	number	只读	表示门户服务器的端口。
appServerIP	string	只读	表示应用下载服务器的 IP 地址,格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。
appServerPort	number	只读	表示应用下载服务器的端口。
host	string	读/写	表示网络对终端设备的一种识别字符串，即主机名，不超过 255 字节。在局域网中也可通过此名称识别此终端设备。
workGroup	string	读/写	表示该网络所属工作组。

N.2.2.2 方法

N.2.2.2.1 getAllEthernets

原型：Ethernet[] getAllEthernets()
描述：获取接收终端所有的宽带网络设备对象。
参数：空。
返回：Ethernet 对象数组。如无此对象，则返回的数组长度为 0。

N.2.2.2.2 ping

原型：ping(address, parameter)
描述：异步方法，执行操作系统的 ping 命令。系统每收到一个 ping 包响应，则向页面发送消息 MSG_BROADBAND_PING_RESPONSE，event.modifiers 属性返回消息描述字符串的 ID，通过 Utility.getEventInfo()方法获取该消息描述字符串。
参数：address - string 型，表示 ping 命令的目标地址，可为 IP 或者域名。
parameter - string 型，表示 ping 命令的参数，取值：
——"-t"- 表示连续地 ping 目标地址，直到手动停止。
返回：无。

N.2.2.2.3 cancelPing

原型：number cancelPing()
描述：取消正在进行的 ping 操作。
参数：无。
返回：number 型，取值：
——0 - 表示当前 ping 操作已经结束或当前无 ping 操作；
——1 - 表示成功停止正在执行的 ping 操作。

N.2.2.2.4 queryNetworkState

原型：queryNetworkState(targetAddress)

描述：异步方法，获取当前网络到指定目标地址的网络状况。操作完成时，系统应向页面发送消息 MSG_BROADBAND_GET_NETWORK_STATE，event.modifiers 属性返回消息描述字符串的 ID,通过 Utility.getEventInfo(ID)方法获取消息描述字符串，JSON 格式为：

```
{
 “targetAddress”:“192.168.1.12”,
 “lostPacketRate”:0,
 “bandwidth”:8,
 “delay”:100
}
```

其中：

——targetAddress - string 型，表示目标地址，格式：

- IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791；
- IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。

——lostPacketRate - number 型，表示丢包率，百分比，取值范围 0～100；

——bandwidth - number 型，表示带宽，单位为兆比特每秒（Mbps）；

——delay - number 型，表示延时，单位为毫秒。

参数：targetAddress - string 类型，表示目标 IP 地址，格式：

- IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791；
- IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。

返回：无。

N.2.2.2.5 getDeviceState

原型：number getDeviceState(device)

描述：获取设备状态。

参数:device - string 型,表示设备名称,设备类型可以为网卡或有线线缆调制解调器(CableModem)。

——若是网卡，取值“eth0”、“eth1”等；

——若是有线线缆调制解调器（CableModem），取值“cm”。

返回：number 型，设备状态值见表 N.3。

表 N.3 设备状态表

设备	返回值	说明
所有设备均适用	0	设备不存在，不可访问。
	1	设备存在，但无法获取状态。
eth0, eth1 等网卡设备	101	网线连接。
	102	网线脱落。
cm	201	网络连接正常。
	202	网络连接中断。
	203	网络状态不改变，但是不良。
	204	有线线缆调制解调器（CableModem）与接收终端通信中断。
	205	有线线缆调制解调器（CableModem）与接收终端通信中断，重新上线。

N.2.2.2.6 NTPUpdate

原型：boolean NTPUpdate()

描述：异步方法，同步 NTP 时间。

——若页面获取到消息 MSG_BROADBAND_NTP_READY_SUCCESS，表示同步网络 NTP 时间成功；

——若页面获取到消息 MSG_BROADBAND_NTP_SYNC_TIMEOUT，表示同步网络 NTP 时间超时。

参数：无。

返回：boolean 型，true 表示开始执行 NTP 时间同步，false 表示未执行。

注：返回值只是表示程序是否开始执行NTP时间同步，并不表示实际的同步结果，实际的同步结果由消息通知。如返回false，可能是因为没有设置NTP服务器等原因引起。

N.2.2.2.7 save

原型：save()

描述：异步方法，将网络信息写入 NVM 中。

——若保存成功，则发送 MSG_BROADBAND_SAVE_CFG_SUCCESS 消息；

——若保存失败，则发送 MSG_BROADBAND_SAVE_CFG_FAILED 消息。

参数：无。

返回：无。

N.2.3 Ethernet对象

Ethernet 对象为本地对象,用以描述本地网卡详细信息，与网卡一一对应,即有几块网卡就有几个 Ethernet 对象。

示例：

```
var ethernetArray = Broadband.getAllEthernets();//获取接收终端所有网卡对象数组
var ethernet = ethernetArray[i];
```

N.2.3.1 属性

Ethernet 对象的属性定义见表 N.4。

表 N.4 Ethernet 对象属性

属性名称	类型	读写属性	说明
enableFlag	number	读/写	表示网卡是否可用，等同于 PC 上的启用或禁用功能，取值： ——1 - 表示网卡可用； ——0 - 表示网卡不可用。 注：只有调用 submitParameters()方法后才生效。
DHCPflag	boolean	读/写	表示是否启用 DHCP，取值： ——true - 表示启用 DHCP； ——false - 表示关闭 DHCP。 注：只有调用 submitParameters()方法后才生效。
DHCPAutoGetDNS	number	读/写	表示此 Ethernet 对象是否使用自动获取的 DNS，取值： ——0 - 表示不使用动态获取的 DNS，使用手动设置的 DNS； ——1 - 表示使用动态获取的 DNS，不使用手动设置的 DNS。 若 Ethernet.DHCPflag=false；设置此属性无效。

表 N.4（续）

属性名称	类型	读写属性	说明
			注：只有调用 submitParameters()方法后才生效。
description	string	只读	表示网卡的描述信息。
MACAddress	string	只读	表示当前网卡 6 个字节的物理地址，每两个十六进制数之间以“-”分隔开，例如“00-15-F2-63-7A-83”。
IPs	IP Array	只读	表示 IP 对象数组。
DNSs	string Array	只读	表示 DNS 服务器 IP 地址数组，格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。
DHCPLeaseObtained	string	只读	表示在 DHCP 模式下租用 IP 地址开始时间,格式为:yyyy-mm-dd hh:mm:ss。
DHCPLeaseExpires	string	只读	表示在 DHCP 模式下 IP 地址的过期时间，格式为：yyyy-mm-dd hh:mm:ss。
DHCPIP	IP	只读	表示 DHCP 动态分配的 IP 对象。
DHCPServer	string	只读	表示当前网络上提供 DHCP 服务器 IP 地址，格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。
DHCPPort	number	只读	表示 DHCP 的端口号。
communicateWay	number	读/写	表示网络带宽及通信方式，取值： ——0 - 表示自适应； ——1 - 表示 10M 全双工； ——2 - 表示 10M 半双工； ——3 - 表示 100M 全双工； ——4 - 表示 100M 半双工。
LANStatus	number	只读	表示 Ethernet 模式的连接状态，取值： ——0 - 表示没有连接上； ——1 - 表示已连接上。
sentPackages	number	只读	表示网卡发送的数据包数量。
receivedPackages	number	只读	表示网卡接收到的数据包数量。
currentConnectionType	string	只读	表示当前网卡的连接类型，取值“ethernet”、“ppp”或“pppoe”等。
HTTPProxy	Proxy	只读	表示 HTTP 代理对象。
HTTPSProxy	Proxy	只读	表示 HTTPS 代理对象。
FTPProxy	Proxy	只读	表示 FTP 代理对象。

N.2.3.2 方法

N.2.3.2.1 setDNS

原型：setDNS(index, dns)
描述：修改DNS服务器。
参数：index - 要修改的DNS服务器的索引；
dns - string型，DNS的IP地址，格式：

——IPv4 - “xxx.xxx.xxx.xxx”，详见IETF RFC 791；
——IPv6 - “x:x:x:x:x:x:x:x”，详见IETF RFC 2373。
返回：无。

N.2.3.2.2 addIP

原型：boolean addIP(ip)
描述：添加一个 IP 对象到该网卡。
参数：ip - IP 对象。
返回：boolean 型，true 表示成功，false 表示失败。

N.2.3.2.3 deleteIPByIndex

原型：boolean deleteIPByIndex(index)
描述：通过 IP 数组下标来删除该网卡中所设置的 IP 对象。
参数：index - number 型，表示数组下标。
返回：boolean 型，true 表示成功，false 表示失败。

N.2.3.2.4 deleteIPByAddress

原型：boolean deleteIPByAddress(address)
描述：通过IP地址来删除该网卡中所设置的IP对象。
参数：address – string型，表示IP地址,格式：
——IPv4 - “xxx.xxx.xxx.xxx”，详见IETF RFC 791；
——IPv6 - “x:x:x:x:x:x:x:x”，详见IETF RFC 2373。
返回：boolean型，true表示成功；false表示失败。

N.2.3.2.5 getAPs

原型：AP[] getAPs()
描述：获取所有无线热点对象。仅用于无线网卡。
参数：无。
返回：AP 对象数组。

N.2.3.2.6 scanAP

原型：scanAP(maxCount, timeOut)
描述：异步方法，搜索无线AP。
——若在参数timeOut指定的超时时间范围内没有搜到任何无线热点，则向页面发送MSG_BROADBAND_SCAN_AP_FAILED消息；
——每搜索到一个无线热点，则向页面发送MSG_BROADBAND_SCAN_AP_FIND消息，event.modifiers属性携带AP对象JSON字符串ID，通过Utility.getEventInfo(ID)方法获取JSON字符串。
——若已经完成maxCount个无线热点搜索，则向页面发送MSG_BROADBAND_SCAN_AP_SUCCESS消息。
参数：maxCount – number型，表示待搜索AP的最大数量。
timeOut – number型，表示搜索超时时间，单位为秒。
返回：无。

N.2.3.2.7 connectAP

原型：connectAP(essId, keyType, key)
描述：异步方法，连接无线网卡接入点。
——若连接成功则发送 MSG_BROADBAND_LINK_AP_SUCCESS 消息；
——若连接失败则发送 MSG_BROADBAND_LINK_AP_FAILED 消息。
参数：essId - string 型，表示无线网卡 ID。
keyType - number 型，表示无线网卡密钥数据类型。
key - string 型，表示认证密钥。
返回：无。

N.2.3.2.8 disconnectAP

原型：number disconnectAP()
描述：断开与无线热点的连接。
参数：无。
返回：number 型，成功返回 0，失败则返回错误码。

N.2.3.2.9 getConnectedAP

原型：AP getConnectedAP()
描述：获取被连接的无线热点对象。
参数：无。
返回：AP 对象。

N.2.3.2.10 submitParameters

原型：submitParameters()
描述：异步方法，将网卡 Ethernet 对象的所有属性一次性提交给底层。
——若成功提交，则向页面发送 MSG_BROADBAND_SUBMIT_SUCCESS 消息；
——若成功失败，则向页面发送 MSG_BROADBAND_SUBMIT_FAILED 消息。
参数：无。
返回：无。

N.2.4 AP对象

AP 对象为本地对象，用以描述无线热点信息。

N.2.4.1 属性

AP 对象的属性定义见表 N.5。

表 N.5 AP 对象属性表

属性名称	类型	读写属性	说明
essId	string	只读	表示无线热点的 ID。
signalStrength	string	只读	表示无线热点的信号强度。
linkQuality	string	只读	表示无线热点的链接质量。
encType	number	只读	表示无线热点认证加密类型。

无线网卡认证加密类型定义见表 N.6。

表 N.6　无线网卡认证加密类型

类型值	说明
0	无加密。
1	WEPOpen 加密。
2	shared 共享模式来加密 WEP 信息。
3	Wi-Fi 保护接入协议。
4	Wi-Fi 保护接入协议 2。

无线网卡错误码对照表见表 N.7。

表 N.7　无线网卡错误码对照表

错误码	含义
0	成功。
1	参数错误。
2	无线网卡启动失败。
3	无线网卡关闭失败。
4	连接错误。
5	设置密钥错误。
6	清除认证信息错误。
7	未获取到无线网卡接入点名称。

无线网卡密钥数据类型定义见表 N.8。

表 N.8 无线网卡密钥数据类型

keyType 值	说明
0	16 进制方式加密。
1	ansi 编码方式加密。
2	string 方式加密。

N.2.5　IP对象

IP 对象为本地对象，用以描述 IP 地址信息。

示例：

```
//通过网卡创建设置 IP 对象
var ethernetArray = Broadband.getAllEthernets();//获取接收终端所有网卡对象数组
var ethernet = ethernetArray[0];
var ip = ethernet.IPs[0];//获取第 0 个网卡对象下的 IP 对象数组中的第 0 个 IP 对象
//通过构造方法，创建 IP 对象
ip = new IP（address, mask, gateway）;
```

N.2.5.1 属性

IP 对象的属性定义见表 N.9。

表 N.9 IP 对象属性表

属性名称	类型	属性	说明
address	string	读/写	表示 IP 地址，格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。
mask	string	读/写	表示 IP 地址的子网掩码，格式与 IP 地址相对应。
gateway	string	读/写	表示 IP 地址的网关，格式： ——IPv4 - “xxx.xxx.xxx.xxx”，详见 IETF RFC 791； ——IPv6 - “x:x:x:x:x:x:x:x”，详见 IETF RFC 2373。

N.2.5.2 方法

N.2.5.2.1 IP

原型：IP(address, mask, gateway)

描述：IP 对象构造方法。

参数：address - string 型，表示 IP 地址，格式同表 T.9“address”属性相关说明。

mask - string 型，表示 IP 地址的子网掩码，格式同表 T.9“mask”属性相关说明。

gateway - string 型，表示 IP 地址的网关，格式同表 T.9“gateway”属性相关说明。

N.2.6 Proxy对象

Proxy 对象为本地对象，描述网络中的代理服务器信息。

N.2.6.1 属性

Proxy 对象的属性定义见表 N.10。

表 N.10 Proxy 对象属性表

属性名称	类型	属性	说明
userName	string	读/写	表示当前代理服务器接入方式的用户名，取值为 0～30 个字符，不可以是空格。 代理服务器在有些代理需要用户名和密码才能连接。
password	string	读/写	表示当前代理服务器接入方式的密码，取值为 0～30 个字符，可以是空格。 代理服务器在有些代理需要用户名和密码才能连接。
enable	number	读/写	表示是否启用代理，取值： ——0 - 表示当前代理无效； ——1 - 表示当前代理立即生效。
unusedProxyURLs	string Array	读/写	表示不使用代理的 URL，每个字符串最多不能超过 255 个字符。
server	string	读/写	表示代理服务器的IP地址或者域名地址，不能超过255个字符。
port	number	读/写	表示提供代理服务的服务器端口号，取值为0～5个数字字符。

附 录 0
（规范性附录）
JavaScript-人机交互单元

0.1 概述

本附录定义了与人机交互相关的功能模块：用户输入模块、前面板输出模块。

0.2 用户输入模块

0.2.1 用户输入模块概述

用户输入是指用户通过遥控器、鼠标、键盘、前面板按键等一些输入设备向接收终端发送用户指令，这些用户指令统一封装成按键消息进行处理。

应用层捕获到的消息来源：

遥控器、键盘、鼠标、前面板等按键触发消息；

应用层通过以下方式捕获消息：

——键盘：应用通过 document.onkeydown、document.onkeyup 和 document.onkeypress 等方式捕获键盘消息，消息码与 PC 方式保持一致；

——鼠标：应用通过 document.onmousedown、document.onmouseup、document.onmousemove 等方式捕获鼠标消息，消息码与 PC 方式保持一致；

——遥控器：应用通过 document.onkeydown、document.onkeyup 捕获遥控器消息，消息码与 PC 方式保持兼容；

——前面板：应用通过 document.onkeydown、document.onkeyup 捕获前面板消息，消息码与遥控器方式保持一致。

0.2.2 event对象

0.2.2.1 属性

event 对象为内置对象，event 对象的属性见表 0.1。

表 0.1 event 对象的属性

属性名称	类型	读写属性	说明
event.type	number	只读	表示发生的事件的类型，即当前 event 对象表示的事件的名称，它与注册的事件句柄同名,例如“onclick”;或者是事件句柄属性删除前缀“on”,例如“click”。
event.source	number	只读	表示消息的来源。
event.which	number	只读	表示消息的代码值。
event.modifiers	number	只读	表示消息的扩展属性。若该消息的扩展属性为空，则 modifiers 返回 0；若该消息的扩展属性为 number 型，则 modifiers 返回该数值；若该消息的扩展属性为字符串，则 modifiers 返回一个 ID 值，该值由系统内部生成，作为具体字符串内容的指针，应用可调用 Utility.getEventInfo(ID)方法取出字符串内容。

0.2.2.2 消息来源

消息来源（event.source）的定义见表0.2。

表0.2 event.source定义

event.source	描述
1002	表示遥控器按键消息。
1003	表示前面板按键消息

0.2.2.3 按键消息

按键消息定义见表D.2。

0.3 前面板输出模块

本模块定义了与前面板输出相关的JS对象：FrontPanel。

0.3.1 FrontPanel对象

FrontPanel对象为内置对象，提供字符串显示、状态指示、时间日期显示、信息清除等前面板操作接口。

0.3.1.1 常量

FrontPanel对象的常量定义见表0.3。

表0.3 FrontPanel对象常量

常量	说明
指示类型	
const TYPE_MAIL = 0;	邮件
const TYPE_SIGNAL = 1;	信号
const TYPE_POWER = 2;	电源
const TYPE_RADIO = 3;	广播
指示状态	
const STATUS_OFF = 0;	关闭
const STATUS_ON = 1;	开启
const STATUS_UNKNOWN = 2;	未知
字符串显示对齐方式	
const ALIGN_CENTER = 0;	水平居中对齐
const ALIGN_LEFT = 1;	水平左对齐
const ALIGN_RIGHT = 2;	水平右对齐

0.3.1.2 方法

0.3.1.2.1 clear

原型：boolean clear()

描述：清除前面板显示信息，包括前面板显示的字符串信息、时间日期信息等。
参数：空。
返回：boolean型，true表示清除成功，false表示清除失败。

0.3.1.2.2 displayDate

原型：boolean displayDate(date)
描述：显示当前的时间日期信息。
参数：date - Date 型，表示时间和日期。
返回：boolean 型，true 表示显示成功，false 表示显示失败。若终端不支持，可不响应该方法的调用，并返回 false。

0.3.1.2.3 displayText

原型：boolean displayText(str)
描述：显示字符串，缺省采用水平居中对齐方式显示。
参数：str - string 型，表示待显示的字符串。
返回：boolean 型，取值 true 表示显示成功，false 表示显示失败。

0.3.1.2.4 displayText

原型：boolean displayText(str, align)
描述：采用指定对齐方式显示字符串，若终端不支持指定的对齐方式，可忽略该参数。
参数：str - string 型，表示待显示的字符串。
align - number 型，表示水平对齐方式。
返回：boolean 型，取值 true 表示显示成功，false 表示显示失败。

0.3.1.2.5 getMaxChars

原型：number getMaxChars()
描述：获取前面板支持的显示字符数量。
参数：无。
返回：number 型，表示前面板支持的显示字符数量。

0.3.1.2.6 getStatus

原型：number getStatus(type)
描述：根据类型获取前面板指示状态。
参数：type - number 型，指定指示类型。
返回：number 型，表示指示状态。若 type 参数指定一个有效的指示类型，则返回其实际的状态（取值 STATUS_ON 或 STATUS_OFF);若 type 参数指定一个无效的指示类型,则返回 STATUS_UNKNOWN。

0.3.1.2.7 setStatus

原型：boolean setStatus(type, status)
描述：设置前面板指示状态。
参数：type - number 型，指定指示类型;
status - number 型，指示状态。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

附 录 P
（规范性附录）
JavaScript-AV 设置单元

P.1 概述

本附录定义了与AV设置相关的功能模块：音频参数设置和视频参数设置。所有设置方法不仅将设置参数写入NVM中，而且同时立即生效。

P.2 音视频参数设置模块

本模块定义了与音视频参数设置相关的JS对象：AudioSetting和VideoSetting。系统应对应用程序的权限进行验证，只有特权应用才能调用本类提供的方法。

P.2.1 AudioSetting对象

AudioSetting为内置对象，提供了对音频参数的设置方法。

P.2.1.1 常量

AudioSetting 对象的常量定义见表 P.1。

表 P.1 AudioSetting 对象常量

常量	说明
声道类型	
const CHANNEL_STEREO = 0;	立体声
const CHANNEL_LEFT = 1;	左声道
const CHANNEL_RIGHT = 2;	右声道
const CHANNEL_MIXED_MONO = 3;	混合声
音频端口类型	
const PORT_HDMI = 0;	HDMI 端口
const PORT_SPDIF = 1;	SPDIF 端口

P.2.1.2 方法

P.2.1.2.1 getOutputInterfaceList

原型：string[] getOutputInterfaceList()
描述：获取接收终端所有可用的音频输出端口列表。
参数：无。
返回：string 数组，表示接收终端所有可用的音频输出接口名称，例如“RCA”、“S/PDIF”、“HDMI”等。若无可用的音频输出端口，则返回的数组长度为 0。

P.2.1.2.2 getOutputInterfaceStatus

原型：boolean getOutputInterfaceStatus(port)
描述：获取音频输出端口的使能状态。
参数：port – string型，表示音频输出接口的名称，由getOutputInterfaceList()方法获得。
返回：boolean型，取值true表示该音频输出端口允许输出，false表示该音频输出端口禁止输出。

P.2.1.2.3 disableOutputInterface

原型：boolean disableOutputInterface(port)
描述：禁止音频输出端口输出。
参数：port – string 型，表示音频输出接口的名称，由 getOutputInterfaceList()方法获得。
返回：boolean 型，取值 true 表示禁止成功，false 表示禁止失败。

P.2.1.2.4 enableOutputInterface

原型：boolean enableOutputInterface(port)
描述：允许音频端口输出。
参数：port – string 型，表示音频输出接口的名称，由 getOutputInterfaceList()方法获得。
返回：boolean 型，取值 true 表示成功，false 表示失败。

P.2.1.2.5 getOutputVolume

原型：number getOutputVolume()
描述：获取全局输出音量。
参数：无。
返回：number型，表示全局输出音量大小，取值范围为0～100，0表示静音，100表示最大音量。

P.2.1.2.6 setOutputVolume

原型：boolean setOutputVolume(volume)
描述：设置全局输出音量。
注：某广播节目的实际输出音量 = 全局输出音量 + 该广播节目相对于全局输出音量的增值。
参数：volume– number型，表示全局输出音量大小，取值范围为0～100，0表示静音，100表示最大音量。
返回：boolean型，true表示设置成功，false表示设置失败。

P.2.1.2.7 getOutputChannelMode

原型：number getOutputChannelMode()
描述：获取当前输出声道类型。
参数：无。
返回：number 型。

P.2.1.2.8 setOutputChannelMode

原型：boolean setOutputChannelMode(audioChannel)
描述：设置当前输出声道类型。
参数：audioChannel –number 型。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.1.2.9 getOutputSPDIFMode

原型：number getOutputSPDIFMode()
描述：获取 S/PDIF 输出接口数据格式（压缩或 PCM 格式）。
参数：无。
返回：number 型，表示 S/PDIF 输出接口数据格式，取值 0 表示 PCM 格式，1 表示压缩格式。

P.2.1.2.10 setOutputSPDIFMode

原型：setOutputSPDIFMode(mode)
描述：设置S/PDIF输出接口数据格式（压缩或PCM格式）。
参数：mode-number型，表示SPDIF输出接口音频模式，取值：
—— 0 – 表示PCM格式，即压缩音频由接收终端解码；
—— 1 – 表示压缩格式，即压缩音频由外接解码设备解码。
返回：无。

P.2.1.2.11 isMute

原型：boolean isMute()
描述：获取静音状态标志。
参数：无。
返回：boolean 型，取值 true 表示静音，false 表示有声。

P.2.1.2.12 mute

原型：boolean mute()
描述：设置静音。
参数：无。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.1.2.13 unMute

原型：boolean unMute()
描述：取消静音。
参数：无。
返回：boolean 型，取值 true 表示取消成功，false 表示取消失败。

P.2.1.2.14 getOutputHDMIFMode

原型：number getOutputHDMIFMode()
描述：获取HDMI输出接口数据格式。
参数：无。
返回：number型，表示HDMI输出接口数据格式.
—— 0 – 表示关闭；
—— 1 – 自动协商；
—— 2 – LPCM，输出解码后信号；
—— 3 – RAW，输出原始信号。

P.2.1.2.15 setOutputHDMIFMode

原型：setOutputHDMIMode(mode)

描述：设置HDMI输出接口数据格式（压缩或PCM格式）。

参数：mode-number型，表示HDMI输出接口音频模式，取值：

—— 0 – 表示关闭；

—— 1 – 自动协商；

—— 2 – LPCM，输出解码后信号；

—— 3 – RAW，输出原始信号。

返回：无。

P.2.2 VideoSetting对象

VideoSetting为内置对象，提供了对视频参数的设置方法。

P.2.2.1 常量

VideoSetting对象常量定义见表P.2。

表P.2 VideoSetting对象常量

常量	说明
视频输出通道	
const VOUT_SD= 1;	标清输出通道
const VOUT_HD = 2;	高清输出通道
视频窗口匹配模式	
const MATCH_METHOD_LETTER_BOX = 1;	信箱模式（letter_box）
const MATCH_METHOD_PAN_SCAN = 2;	全画面模式（pan_scan）
const MATCH_METHOD_COMBINED = 3;	组合模式（combined）
const MATCH_METHOD_IGNORE = 4;	忽略模式（ignore）
视频输出制式	
const VOUT_STANDARD_UNKNOWN = 0;	未知
const VOUT_STANDARD_NTSC_J= 101;	标清——NTSC-J/3.5795MHz彩色副载波
const VOUT_STANDARD_NTSC_M= 102;	标清——NTSC-M/3.5795MHz彩色副载波
const VOUT_STANDARD_NTSC_443 = 103;	标清——NTSC-443/4.4336MHz彩色副载波
const VOUT_STANDARD_PAL_B= 211;	标清——PAL-B（澳大利亚）
constVOUT_STANDARD_PAL_B1 = 212;	标清——PAL-B1（匈牙利）
const VOUT_STANDARD_PAL_D= 213;	标清——PAL-D（中国大陆）
const VOUT_STANDARD_PAL_D1= 214;	标清——PAL-D1（波兰）
const VOUT_STANDARD_PAL_G= 215;	标清——PAL-G（欧洲）
const VOUT_STANDARD_PAL_H= 216;	标清——PAL-H（欧洲）
constVOUT_STANDARD_PAL_I= 217;	标清——PAL-I（英国、香港、澳门）
const VOUT_STANDARD_PAL_K= 218;	标清——PAL-K（欧洲）
const VOUT_STANDARD_PAL_M= 220;	标清——PAL-M（巴西）

表 P.2（续）

常量	说明
const VOUT_STANDARD_PAL_N= 221;	标清——PAL-N（牙买加、乌拉圭）
const VOUT_STANDARD_PAL_NC= 222;	标清——PAL-NC（阿根廷）
constVOUT_STANDARD_SECAM_B= 311;	标清——SECAM-B
const VOUT_STANDARD_SECAM_D= 312;	标清——SECAM-D
const VOUT_STANDARD_SECAM_G= 313;	标清——SECAM-G
const VOUT_STANDARD_SECAM_I= 314;	标清——SECAM-I
const VOUT_STANDARD_SECAM_K= 315;	标清——SECAM-K
constVOUT_STANDARD_SMPTE274_1080I_50 = 27400;	高清——SMPTE274/1920x1080I/50HZ/1125行
const VOUT_STANDARD_SMPTE274_1080I_59_94= 27401;	高清——SMPTE274/1920x1080I/59.94HZ/1125行
const VOUT_STANDARD_SMPTE274_1080I_60= 27402;	高清——SMPTE274/1920x1080I/60HZ/1125行
const VOUT_STANDARD_SMPTE274_1080P_23_98= 27410;	高清——SMPTE274/1920x1080P/23.98HZ/1125行
const VOUT_STANDARD_SMPTE274_1080P_24= 27411;	高清——SMPTE274/1920x1080P/24HZ/1125行
constVOUT_STANDARD_SMPTE274_1080P_25 = 27412;	高清——SMPTE274/1920x1080P/25HZ/1125行
const VOUT_STANDARD_SMPTE274_1080P_29_97= 27413;	高清——SMPTE274/1920x1080P/29.97HZ/1125行
const OUT_STANDARD_SMPTE274_1080P_30= 27414;	高清——SMPTE274/1920x1080P/30HZ/1125行
const VOUT_STANDARD_SMPTE274_1080P_50= 27415;	高清——SMPTE274/1920x1080P/50HZ/1125行
const VOUT_STANDARD_SMPTE274_1080P_59_94= 27416;	高清——SMPTE274/1920x1080P/59.94HZ/1125行
constVOUT_STANDARD_SMPTE274_1080P_60 = 27417;	高清——SMPTE274/1920x1080P/60HZ/1125行
const VOUT_STANDARD_SMPTE295_1080I_50= 29500;	高清——SMPTE295/1920x1080I/50HZ/1250行
const VOUT_STANDARD_SMPTE295_1080P_50= 29510;	高清——SMPTE295/1920x1080P/50HZ/1250行
const VOUT_STANDARD_SMPTE296_720P_23_98= 29610;	高清——SMPTE296/1280x720P/23.98HZ/750行
const VOUT_STANDARD_SMPTE296_720P_24= 29611;	高清——SMPTE296/1280x720P/24HZ/750行
constVOUT_STANDARD_SMPTE296_720P_25 = 29612;	高清——SMPTE296/1280x720P/25HZ/750行
const VOUT_STANDARD_SMPTE296_720P_29_97= 29613;	高清——SMPTE296/1280x720P/29.97HZ/750行
const VOUT_STANDARD_SMPTE296_720P_30= 29614;	高清——SMPTE296/1280x720P/30HZ/750行
const VOUT_STANDARD_SMPTE296_720P_50= 29615;	高清——SMPTE296/1280x720P/50HZ/750行
const VOUT_STANDARD_SMPTE296_720P_59_94= 29616;	高清——SMPTE296/1280x720P/59.94HZ/750
constVOUT_STANDARD_SMPTE296_720P_60 = 29617;	高清——SMPTE296/1280x720P/60HZ/750行
const VOUT_STANDARD_2160P_24 = 29618;	高清——3840x2160P/24HZ
const VOUT_STANDARD_2160P_25 = 29619;	高清——3840x2160P/25HZ
const VOUT_STANDARD_2160P_30 = 29620;	高清——3840x2160P/30HZ
const VOUT_STANDARD_2160P_50 = 29621;	高清——3840x2160P/50HZ
constVOUT_STANDARD_2160P_60 = 29622;	高清——3840x2160P/60HZ
constVOUT_STANDARD_4096P_24 = 29623;	高清——4096x2160P/24HZ

P.2.2.2 方法

P.2.2.2.1 getOutputInterfaceList

原型：string[] getOutputInterfaceList()

描述：获取接收终端所有可用的视频输出端口列表。
参数：无。
返回：string数组，表示接收终端所有可用的视频输出接口名称，例如“CVBS”、“YUV”、“HDMI-0”、“HDMI-1”、“DVO”等。若无可用的视频输出端口，则返回的数组长度为0。

P.2.2.2.2 getOutputInterfaceStatus

原型：boolean getOutputInterfaceStatus(port)
描述：获取视频输出端口的使能状态。
参数：port – string型，表示视频输出接口的名称，由getOutputInterfaceList()方法获得。
返回：boolean型，取值true表示该视频输出接口允许输出，false表示该视频输出接口禁止输出。

P.2.2.2.3 disableOutputInterface

原型：boolean disableOutputInterface(port)
描述：禁止视频输出端口输出。
参数：port –string型，表示视频输出接口的名称，由getOutputInterfaceList()方法获得。
返回：boolean型，取值true表示禁止成功，false表示禁止失败。

P.2.2.2.4 enableOutputInterface

原型：boolean enableOutputInterface(port)
描述：允许视频输出端口输出。
参数：port – string型，表示视频输出接口的名称，由getOutputInterfaceList()方法获得。
返回：boolean型，取值true表示允许成功，false表示允许失败。

P.2.2.2.5 getOutputBrightness

原型：number getOutputBrightness()
描述：获取视频输出的亮度。
参数：无。
返回：number型,表示视频输出的亮度，取值范围从小到大为0～100，0表示最黑，100表示最亮。

P.2.2.2.6 getOutputContrast

原型：number getOutputContrast()
描述：获取视频输出的对比度。
参数：无。
返回：number型，表示视频输出的对比度，取值范围从小到大为0～100。

P.2.2.2.7 getOutputSaturation

原型：number getOutputSaturation()
描述：获取视频输出的饱和度（色度）。
参数：无。
返回：number型，表示视频输出的饱和度（色度），取值范围从小到大为0～100。

P.2.2.2.8 getOutputStandard

原型：number getOutputStandard(device)

描述：获取视频输出的制式。
参数：device – number型，表示视频输出通道，取值VOUT_SD或VOUT_HD。
返回：number型，表示视频输出的制式。

P.2.2.2.9 getOutputTransparency

原型：number getOutputTransparency()
描述：获取视频输出的透明度。
参数：无。
返回：number型，表示透明度，取值范围为0～100，0表示完全不透明，100表示完全透明。

P.2.2.2.10 setOutputBrightness

原型：boolean setOutputBrightness(value)
描述：设置视频输出的亮度。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value – number型，表示视频输出的亮度，取值范围从小到大为0～100，0表示最黑，100表示最亮。
返回：boolean型，取值true表示设置成功，false表示设置失败。

P.2.2.2.11 setOutputContrast

原型：boolean setOutputContrast(value)
描述：视频输出的对比度。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value – number，表示视频输出的对比度，取值范围从小到大为0～100。
返回：boolean型，取值true表示设置成功，false表示设置失败。

P.2.2.2.12 setOutputSaturation

原型：boolean setOutputSaturation(value)
描述：设置视频输出的饱和度（色度）。该设置无法针对单个视频输出单元，对所有视频输出单元同时生效。
参数：value – number，表示饱和度（色度），取值范围从小到大为0～100。
返回：boolean型，取值true表示设置成功，false表示设置失败。

P.2.2.2.13 setOutputStandard

原型：number setOutputStandard(device, standard)
描述：设置视频输出的制式。需要对标清输出单元和高清输出单元单独设置。
参数：device - number型，表示视频输出单元；
standard - number型，表示视频输出制式。标清输出单元只能选标清制式，高清输出单元只能选高清制式。
返回：boolean型，取值true表示设置成功，false表示设置失败。

P.2.2.2.14 setOutputTransparency

原型：boolean setOutputTransparency(value)
描述：设置视频输出的透明度。该方法无法针对单个视频输出单元进行设置，对所有视频输出单元都生效。
参数：value – number，表示透明度，取值范围为0～100，0表示完全不透明，100表示完全透明。

返回：boolean型，取值true表示设置成功，false表示设置失败。

P.2.2.2.15 setOutputMatchMethod

原型：setOutputMatchMethod(mode)
描述：设置视频输出窗口匹配模式。
参数：mode －number 型。
返回：无。

P.2.2.2.16 getOutputMatchMethod

原型：number getOutputMatchMethod()
描述：获取视频输出窗口匹配模式。
参数：无。
返回：number 型。

P.2.2.2.17 getOutputAspectRatio

原型：getOutputAspectRatio()
描述：获取视频输出宽高比。
参数：无。
返回：number 型，0 表示 16:9；1 表示 4:3。

P.2.2.2.18 setOutputAspectRatio

原型：setOutputAspectRatio(mode)
描述：设置视频输出宽高比。
参数：mode － number 型，0 表示 16:9；1 表示 4:3。
返回：无。

P.2.2.2.19 GetColorSpaceMode

原型：number GetColorSpaceMode()
描述：获取色彩空间模式。
参数：无。
返回：number 型，0-RGB444，1-YCBCR422， 2 -YCBCR444，3-YCBCR420。

P.2.2.2.20 GetDeepColorMode

原型：number GetDeepColorMode()
描述：获取色彩空间模式。
参数：无。
返回： number 型，0-COLOR_24BIT，1- COLOR_30BIT，2-COLOR_36BIT，3-COLOR_DEEP_OFF。

P.2.2.2.21 SetColorSpaceAndDeepColor

原型：boolean SetColorSpaceAndDeepColor(colorSpace, deepColor)
描述：设置色彩空间，深色深模式。
参数：colorSpace－ number 型，0-RGB444，1-YCBCR422，2-YCBCR444，3-YCBCR420
deepColor－ number 型，0-COLOR_24BIT，1-COLOR_30BIT，2-COLOR_36BIT，3-COLOR_DEEP_OFF

返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.22 GetHDRType

原型：number GetDeepColorMode()
描述：获取 HDR 模式。
参数：无。
返回： number 型，0- HDRTYPE_SDR，1- HDRTYPE_DOLBY，2- HDRTYPE_HDR10，3-C HDRTYPE_AUTO。

P.2.2.2.23 SetHDRType

原型：boolean SetHDRType(type)
描述：设置 HDR 模式。
参数：type－number 型，0- HDRTYPE_SDR，1- HDRTYPE_DOLBY，2- HDRTYPE_HDR10，3-C HDRTYPE_AUTO
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.24 GetStereoOutMode

原型：number GetStereoOutMode()
描述：获取 3D 输出模式。
参数：无。
返回： number 型，0- 3D_NONE，1- 3D_FRAME_PACKING，2- 3D_SIDE_BY_SIDE_HALF，
3-3D_TOP_AND_BOTTOM，4-3D_FIELD_ALTERNATIVE，5-3D_LINE_ALTERNATIVE，
6-3D_SIDE_BY_SIDE_FULL，7-3D_L_DEPTH，8-3D_L_DEPTH_GRAPHICS_GRAPHICS_DEPTH。

P.2.2.2.25 SetStereoOutMode

原型：boolean SetStereoOutMode(mode, fps)
描述：设置 3D 输出模式。
参数：mode－number 型，0- 3D_NONE，1- 3D_FRAME_PACKING，2- 3D_SIDE_BY_SIDE_HALF，
3-3D_TOP_AND_BOTTOM，4-3D_FIELD_ALTERNATIVE，5-3D_LINE_ALTERNATIVE，
6-3D_SIDE_BY_SIDE_FULL，7-3D_L_DEPTH，8-3D_L_DEPTH_GRAPHICS_GRAPHICS_DEPTH
Fps- number 型，视频帧率， 取值 23、24、25、30、50、59、60。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.26 GetRightEyeFirst

原型：number GetRightEyeFirst()
描述：获取 3D 输出信号出哪眼画面先出。
参数：无。
返回： number 型，取值 0-LEFT_EYE_FIRST，1- RIGHT_EYE_FIRST。

P.2.2.2.27 SetRightEyeFirst

原型：boolean SetRightEyeFirst(Outpriority)
描述：设置 3D 输出信号出哪眼画面先出。
参数：Outpriority－number 型，0-LEFT_EYE_FIRST，1- RIGHT_EYE_FIRST
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.28 GetStereoDepth

原型：number GetStereoDepth()
描述：获取 3D 画面深度调节信息。
参数：无。
返回： number 型，取值 0-10。

P.2.2.2.29 SetStereoDepth

原型：boolean SetStereoDepth(depth)
描述：设置 3D 画面深度调节信息。
参数：depth－number 型，0-10
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.30 getPictureMode()

原型：number getPictureMode()
描述：获取画面模式。
参数：无。
返回：number 型，0 标准，1 动态，2 柔和，4 用户，5 艳丽，6 自然，7 运动。

P.2.2.2.31 setPictureMode

原型：boolean setPictureMode(mode)
描述：设置画面模式。
参数：mode － number 型，0 标准，1 动态，2 柔和，4 用户，5 艳丽，6 自然，7 运动。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.32 getDisplayHue

原型：number getDisplayHue()
描述：获取画面模式。
参数：无。
返回：number 型，取值范围 0-100，表示色调调整值。

P.2.2.2.33 setDisplayHue

原型：boolean setDisplayHue(hue)
描述：设置画面模式。
参数：hue － number 型，取值范围 0-100，表示色调调整值。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.34 SaveDisplayFmt

原型：boolean SaveDisplayFmt()
描述：保存视频输出格式永久生效。
参数：无。
返回：boolean型，取值true表示成功，false表示失败。

P.2.2.2.35 setOptimalFormatEnable

原型：boolean setOptimalFormatEnable(enabled)

描述：设置自动优化视频输出格式使能。
参数：enabled - number 型，0 表示不使能，1 表示使能。
返回：boolean 型，取值 true 表示设置成功，false 表示设置失败。

P.2.2.2.36 getOptimalFormatEnable

原型：number getOptimalFormatEnable()
描述：获取自动优化视频输出格式使能。
参数：无 - number 型，0 表示不使能，1 表示使能。
返回：number 型，0 表示不使能，1 表示使能。

附 录 Q
（规范性附录）
JavaScript-媒体处理单元

Q.1 概述

本附录定义了与媒体处理相关的功能模块：媒体播放模块。

Q.2 媒体播放模块

本模块定义了与媒体播放相关的 JS 对象：MediaPlayer。

Q.2.1 消息

媒体播放模块发给应用层的消息定义见表Q.1，应用上接收消息采用document.onsystemevent。

表 Q.1 媒体播放模块消息

消息名称	event.which	event.modifiers	消息说明
MSG_MEDIA_URL_VALID	13001	-	媒体源路径有效。
MSG_MEDIA_URL_INVALID	13002	-	媒体源路径无效。
MSG_MEDIA_PLAY_SUCCESS	13003	-	开始播放成功。
MSG_MEDIA_PLAY_FAILED	13004	-	开始播放失败。
MSG_MEDIA_SETPACE_SUCCESS	13005	-	步长设置成功。
MSG_MEDIA_SETPACE_FAILED	13006	-	步长设置失败。
MSG_MEDIA_SEEK_SUCCESS	13007	-	设置播放时间点成功。
MSG_MEDIA_SEEK_FAILED	13008	-	设置播放时间点失败。
MSG_MEDIA_PAUSE_SUCCESS	13009	-	暂停播放成功。
MSG_MEDIA_PAUSE_FAILED	13010	-	暂停播放失败。
MSG_MEDIA_RESUME_SUCCESS	13011	-	恢复播放成功。
MSG_MEDIA_RESUME_FAILED	13012	-	恢复播放失败。
MSG_MEDIA_STOP_SUCCESS	13013	-	停止播放成功。
MSG_MEDIA_STOP_FAILED	13014	-	停止播放失败。
保留	13015～13200		

event.modifiers 值由系统内部自动给出，其数据类型：

——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。

——“-”，表示event.modifiers为undefined。

Q.2.2 MediaPlayer对象

MediaPlayer 对象为本地对象，需要先 new 创建再使用。

该对象定义了在 Web 页面中进行媒体播放的属性和方法。媒体源可以是电视广播、声音广播或 NVOD，也可以是一个 UDP 单播或组播流，还可以是存储在接收终端本地的媒体文件。本对象只需知道媒体类型和媒体所在位置（URL）就可播放此媒体。

视频层的显示，必须通过 video element 来呈现透明，在 page layout 时，由 video element 指定了视频在页面上位置、大小、层次，页面上 必须采用 <video src=" "></video>，src 为空，例如：

<video id="video" src="" style="position: absolute; z-index:18; top:200px; left:180px; width:400px; height:400px;"></video>

MediaPlayer 对象构造后只能在单个 Web 页面中使用，通过对象名来唯一标识；播放器实例对应于接收终端的媒体解码资源，不和 Web 页面绑定，可以跨页面使用，通过播放器实例 ID 来唯一标识，播放器实例 ID 由系统内部自动产生。浏览器必须通过 MediaPlayer 对象提供的属性和方法，对播放器实例进行控制实现媒体播放，且一个 MediaPlayer 对象只能绑定一个播放器实例，一个播放器实例同时也只能被一个 MediaPlayer 对象绑定。MediaPlayer 对象和播放器实例之间的关系示意见图 Q.1。

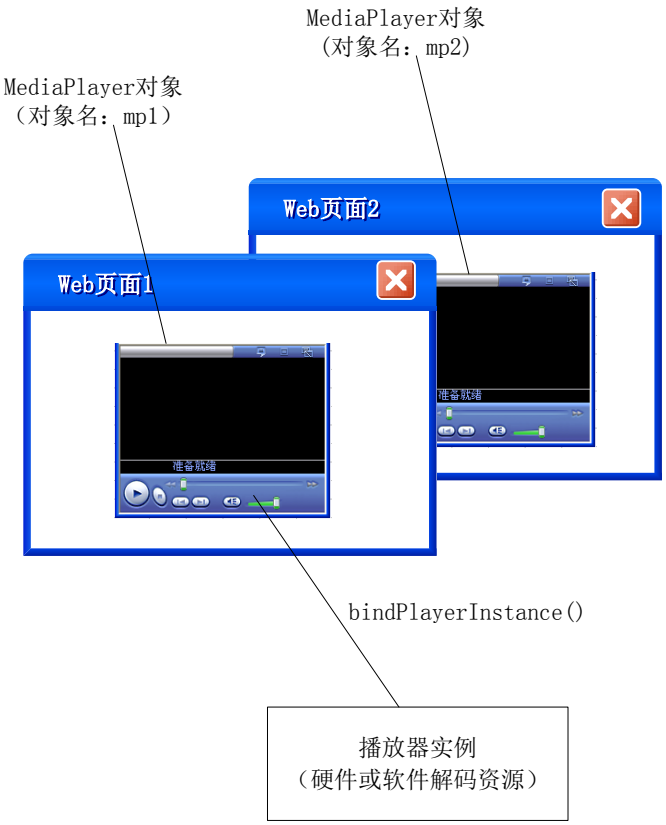


图 Q.1　MediaPlayer 对象和媒体播放器实例之间的关系示意图

示例：

Web 页面中创建的 MediaPlayer 对象，可控制接收终端上的播放器实例，播放器实例的生命周期是跨页面的。通过 MediaPlayer 对象的 playerInstanceID 属性来标识当前所绑定的播放器实例。

```
//在第一个 Web 页面中，创建一个 MediaPlayer 对象 mp1
var mp1 = new MediaPlayer();
//读取本地的媒体播放器实例 ID
var nativePlayerInstanceID = mp1.getPlayerInstanceID();
//通过全局变量保留这个播放器实例标识以便跨页面时使用
```

```
GlobalVarManager.setItemValue(“PLAYER_INSTANCE_ID”, nativePlayerInstanceID);
//MediaPlayer 对象与播放器实例绑定
mp1.bindPlayerInstance(nativePlayerInstanceID);
mp1.setMediaSource(mediaURL); //设置媒体源
mp1.play();//开始播放
...
mp1.setPace(2);   //快进，2 倍速播放
...
mp1.setPace(1);   //恢复正常速度播放
...
mp1.pause(); //暂停播放
mp1.resume();//恢复播放
mp1.stop();  //停止播放
mp1.unbindPlayerInstance(nativePlayerInstanceID);
//在下一个 Web 页面中，创建一个 MediaPlayer 对象 mp2
var mp2 = new MediaPlayer();
//通过全局变量获得上个页面的播放器实例 ID
var nativePlayerInstanceID = GlobalVarManager.getItemValue(“PLAYER_INSTANCE_ID”);
//根据上一个页面传递过来的播放器实例 ID ，绑定 MediaPlayer 对象和媒体播放实例
mp2.bindPlayerInstance(nativePlayerInstanceID);
mp2.setMediaSource(mediaURL); //设置媒体源
mp2.play(); //开始播放
mp2.pause(); //暂停播放
mp2.resume();//恢复播放
mp2.stop();  //停止播放
mp2.unbindPlayerInstance(nativePlayerInstanceID);
```

Q.2.2.1　媒体播放状态

媒体播放器存在以下播放状态：初始化、绑定、停止、播放（前进/后退/快进/慢进/快退/慢退）、暂停和解除绑定，状态转移示意图见图 Q.2。

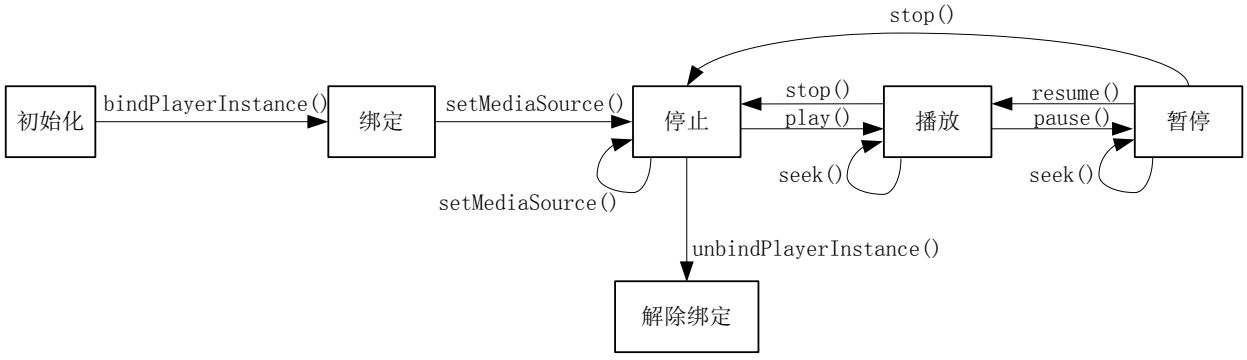


图 Q.2　媒体播放状态转移示意图

图Q.2中媒体播放的各种状态说明如下：

——初始化状态：当创建 MediaPlayer 对象，即进入初始化状态。

——绑定状态：当 MediaPlayer 对象调用 bindPlayerInstance()方法后，即进入绑定状态。

——停止状态：当 MediaPlayer 对象调用 setMediaSource()或 stop()方法后，即进入停止状态，此时视音频停止播放，视频画面处于隐藏状态。

——播放状态：当 MediaPlayer 对象调用 play()方法后，即进入播放状态，包括前进、后退、快进、慢进、快退和慢退，播放步长与播放状态关系见表 Q.2。

表 Q.2　播放步长与播放状态关系

播放状态	步长（pace）	说明
前进	1	当用play()方法启动播放后，pace=1，即进入前进状态，缺省步长为1。
后退	-1	当用play()方法启动播放后，pace=-1，若媒体并非从头播放，即进入后退状态。
快进	2、4、8、16、32	当用play()方法启动播放后，pace=2、4、8、16、32时，即进入快进状态。
慢进	1/2、1/4、1/8	当用play()方法启动播放后，pace=1/2、1/4、1/8时，即进入慢进状态。
快退	-2、-4、-8、-16、-32	当用play()方法启动播放后，pace=-2、-4、-8、-16、-32时，若媒体并非从头播放，即进入快退状态。
慢退	-1/2、-1/4、-1/8	当用play()方法启动播放后，pace=-1/2、-1/4、-1/8时，若媒体并非从头播放，即进入慢退状态。

——暂停状态：当MediaPlayer对象调用pause()方法后，即可进入暂停状态；在暂停状态下，视频画面继续占据屏幕空间，且可将其设置为静帧（默认）或黑场。

——解除绑定状态：当MediaPlayer对象调用unbindPlayerInstance()方法后，即可进入解除绑定状态。释放资源。

播放的媒体源有以下几种形式：

——广播业务，包括电视业务、音频广播业务、NVOD 业务等；

——互动电视业务，包括视频点播、时移电视和频道回看等；

——通过 IP-UDP 方式下发到终端的码流；

——储存在终端本地的媒体文件。

不同来源的媒体所支持的播放状态和播放步长见表 Q.3。

表 Q.3　媒体播放状态和步长表

媒体来源	初始化	绑定	就绪	播放						暂停	停止	解除绑定
				前进	后退	快进	快退	慢进	慢退			
广播业务	√	√	√	√	N/A	N/A	N/A	N/A	N/A	√	√	√
互动电视业务	√	√	√	√	√	√	√	N/A	N/A	√	√	√
IP-UDP 码流	√	√	√	√	N/A	N/A	N/A	N/A	N/A	√	√	√
本地媒体文件	√	√	√	√	√	√	√	√	√	√	√	√

Q.2.2.2 属性

媒体播放对象的属性定义见表Q.4。

表Q.4 MediaPlayer对象的属性

属性名称	类型	读写属性	说明
location	string	只读	表示媒体文件的定位器。
playerInstanceID	number	只读	表示当前MediaPlayer对象所绑定的播放器实例ID。若取值-1，则表示当前MediaPlayer对象尚未绑定任何播放器实例。 例如在调用 bindPlayerInstance()方法之前，或在调用unbindPlayerInstance()方法之后，读取该属性，应返回-1。

Q.2.2.3 方法

Q.2.2.3.1 MediaPlayer

原型：MediaPlayer()
描述：构造方法，创建一个默认的MediaPlayer对象。
参数：无。

Q.2.2.3.2 getPlayerInstanceID

原型：number getPlayerInstanceID()
描述：获取接收终端本地可用的播放器实例ID，该ID由系统内部自动分配。
参数：无。
返回：number型，若成功则返回0～255，若失败则返回-1。

Q.2.2.3.3 bindPlayerInstance

原型：number bindPlayerInstance(playerInstanceID)
描述：MediaPlayer对象与播放器实例绑定。
参数：playerInstanceID - number型，指示本地播放器实例ID，取值范围0～255。
返回：number型，若成功则返回0，若失败则返回-1。

Q.2.2.3.4 unbindPlayerInstance

原型：number unbindPlayerInstance(playerInstanceID)
描述：MediaPlayer对象和当前播放器实例解除绑定，并释放播放器的相关资源。
参数：playerInstanceID - number型，指示本地播放器实例ID，取值范围0～255。
返回：number型，若成功则返回0，若失败则返回-1。

Q.2.2.3.5 setMediaSource

原型：number setMediaSource(mediaURL)
描述：异步方法，设置待播放媒体的URL地址。设置此参数后，系统自动检测设置的mediaURL的合法性。
——若URL合法，则向页面发送MSG_MEDIA_URL_VALID消息；
——若URL不合法，则向页面发送MSG_MEDIA_URL_INVALID消息。

页面只有接收到MSG_MEDIA_URL_VALID消息后，才可以调用play()方法进行播放。

参数：mediaURL - string型，以URL格式表示的媒体路径，URL格式说明见表Q.5。

表Q.5 媒体来源格式

媒体源	mediaURL	说明
广播业务	dvb://<original_network_id>.<transport_stream_id>.<service_id>[.<component_tag>{&<component_tag>}][;event_id>]{/<path-element>}	可用于访问： ——广播业务； ——业务组件，例如视频、音频或字幕等基本流； ——广播业务的事件； ——OC/DC轮播中的文件。
广播业务	dvbelement://frequency.symbolrate.modulation.serviceid.pmtpid.pcrpid.videotype.videopid.audiotype.audiopid	通过指定频点等节目信息播放，以满足快速启播。其中frequency单位为千赫兹，所有参数为十进制。 针对多Tuner场景扩展为 dvbelement://frequency.symbolrate.modulation.serviceid.pmtpid.pcrpid.videotype.videopid.audiotype.audiopid?tunerid=xxx。
广播业务	dvbelement://frequency.symbolrate.modulation.serviceid.-1.-1.-1.-1.-1.-1	通过指定频点等节目信息播放，以满足不搜索节目时加扰流、多音轨播放，其格式为 dvbelement://frequency.symbolrate.modulation.serviceid.-1.-1.-1.-1.-1.-1。此场景媒体引擎根据serviceid从码流中解析pmtpid，根据pmtpid解析pcrpid、videotype、videopid、audiotype和audiopid，若pmtpid存在且有效，则不搜索pmtpid，根据pmtpid解析pcrpid、videotype、videopid、audiotype和audiopid。其中frequency单位为千赫兹，所有数字为十进制，无效参数为-1，如 dvbelement://131000.6875.64.3.-1.-1.-1.-1.-1.-1 或 dvbelement://131000.6875.64.3.4.-1.-1.-1.-1.-1。 针对多Tuner场景扩展为 dvbelement://frequency.symbolrate.modulation.serviceid.-1.-1.-1.-1.-1.-1?tunerid=xxx。
互动电视业务	rtsp://<会话管理器IP>:<会话管理器端口>/;EntitlementCode=<EntitlementCode>	可用于访问： ——视频点播； ——时移电视； ——频道回看。

表 Q.5（续）

媒体源	mediaURL	说明
IP-UDP 流	udp://MulticastAddress:UDPPort:Programnumber:VideoPID:AudioPID	--当前端使用 UDP 单播时，MulticastAddress 为“0.0.0.0”;当前端使用 UDP 组播时,MulticastAddress 为发送组播地址； -- UDPPort 为 UDP 端口号； -- Programnumber 为业务 ID 号； -- VideoPID 为视频 PID； -- AudioPID 为音频 PID； VideoPID 可填 0，系统默认取该业务下 PID 值最小的视频流，若该业务为纯音频业务，该值为 0；AudioPID 可填 0，系统默认取该业务下 PID 值最小的音频流。
媒体文件	file://<host>/<path>	访问本地媒体文件。
HTTP	http://<远程主机 IP>:<远程主机端口>/<path>	访问远程资源文件。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.6 play

原型：number play()

描述：异步方法，从媒体起始点开始播放，对电视广播以实时方式开始播放。

——若播放成功，则向页面发送 MSG_MEDIA_PLAY_SUCCESS 消息；

——若播放失败，则向页面发送 MSG_MEDIA_PLAY_FAILED 消息。

返回：number 型，1 表示成功，0 表示失败。

Q.2.2.3.7 seek

原型：number seek(type，timestamp)

描述：异步方法，从当前媒体的某个时间点开始播放，对实时播放的电视广播该调用无效，但对处于时移状态的电视广播有效。

——若调用成功，则发送MSG_MEDIA_SEEK_SUCCESS消息；

——若调用失败，则发送MSG_MEDIA_SEEK_FAILED消息。

注：仅在播放和暂停状态下可以调用seek()方法。

参数：type – number型，取值：

—— 1 - 表示Normal Play Time；

—— 2 - 表示Absolute Time。

timestamp–Normal Play Time(NPT)和Absolute Time(ClockTime)两种时间类型格式。

——对VOD而言，是从媒体起始点开始计算的相对时间；

——对时移等有时间基的媒体而言就是绝对时间。

它可以表示起始时间（例如：“123-”），也可以表示时间段（例如：“123-333”）。若是时间段的方式，播放到达结束时间时应为暂停状态，而不是停止状态。

返回：number型，若成功则返回0，否则返回其他。

Q.2.2.3.8 setPace

原型：number setPace(pace)

描述：异步方法，设置播放步长。

——若设置成功，则向页面发送 MSG_MEDIA_SET_PACE_SUCCESS 消息；

——若设置失败，则向页面发送 MSG_MEDIA_SET_PACE_FAILED 消息。

注：仅在播放状态下可以调用setPace()方法。

参数：pace - number 型，播放步长。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.9 pause

原型：number pause()

描述：异步方法，暂停播放。

——若暂停成功，则向页面发送 MSG_MEDIA_PAUSE_SUCCESS 消息；

——若暂停失败，则向页面发送 MSG_MEDIA_PAUSE_FAILED 消息。

参数：无。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.10 resume

原型：number resume()

描述：异步方法，恢复播放。

——若恢复播放成功，则向页面发送 MSG_MEDIA_RESUME_SUCCESS 消息；

——若恢复播放失败，则向页面发送 MSG_MEDIA_RESUME_FAILED 消息。

参数：无。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.11 stop

原型：number stop()

描述：异步方法，停止播放。

——若停止成功，则向页面发送 MSG_MEDIA_STOP_SUCCESS 消息；

——若停止失败，则向页面发送 MSG_MEDIA_STOP_FAILED 消息。

参数：无。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.12 refresh

原型：number refresh()

描述：调用 setVideoDisplayMode()、setVideoDisplayArea()等属性调整视频参数后并不是立即生效，而是要在调用该方法后才会生效。

参数：无。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.13 enableTrickMode

原型：number enableTrickMode(flag)

描述：设置 MediaPlayer 对象当前所绑定的播放器实例在其生命周期内是否允许特技操作（包括快进/快退/暂停等），与媒体本身的特技模式属性是逻辑与的关系。

例如：某个流前面 30 秒的广告不能进行特技操作，则可以在该时间段内调用enableTrickMode(0),此时特技被禁止;过了 30 秒后,Web 页面中再调用 enableTrickMode (1),以启动特技功能。若所播放的媒体源本身不支持特技模式，则此方法无效。

参数：flag - boolean 型，取值 false 表示禁止特技操作(缺省值)；取值 true 表示允许特技操作。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.14 getTrickModeFlag

原型：boolean getTrickModeFlag()

描述：获取特技模式操作标志。

参数：无。

返回：boolean 型，取值 true 表示允许特技操作，false 表示不允许特技操作(默认值)。

Q.2.2.3.15 setVideoDisplayMode

原型：number setVideoDisplayMode(mode)

描述：设置MediaPlayer对象对应的视频窗口的显示模式。每次调用该方法后，视频显示窗口并不会立即显示设置的模式，只有调用refresh()方法后才会刷新窗口来显示设置的模式。

参数：mode – number型，视频显示模式，取值如下：

——0-按setVideoDisplayArea()方法中设定的height、width、left和top属性所指定的位置和大小来显示；

——1-全屏显示，按全屏高度和宽度显示(默认值)；

——2-宽度显示，指在不改变原有图像纵横比的情况下按全屏宽度显示；

——3-高度显示，指在不改变原有图像纵横比的情况下按全屏高度显示；

——255-视频显示窗口将被关闭，它将在保持媒体流连接的前提下，隐藏视频窗口，若媒体播放没有停止，将继续播放。

返回：number型，若成功则返回0，否则返回其他值。

Q.2.2.3.16 getVideoDisplayMode

原型：number getVideoDisplayMode()

描述：获取MediaPlayer对象对应的视频窗口的显示模式。

参数：无。

返回：number型，取值如下：

——0-按setVideoDisplayArea()方法中设定的height、width、left和top属性所指定的位置和大小显示；

——1-全屏显示，按全屏高度和宽度显示(默认值)；

——2-宽度显示，指在不改变原有图像纵横比的情况下按全屏宽度显示；

——3-高度显示，指在不改变原有图像纵横比的情况下按全屏高度显示；

——255-视频显示窗口将被关闭，它将在保持媒体流连接的前提下，隐藏视频窗口。若媒体播放没有暂停，将继续播放。

——其他：显示模式非法。

Q.2.2.3.17 setVideoDisplayArea

原型：number setVideoDisplayArea(rect)

描述：设置窗口显示区域。每次调用该方法后，视频显示窗口并不立即按设置的区域显示，只有调用

refresh() 方法后才按照设置的区域来显示视频。当需要将视频在 setVideoDisplayArea()方法设置的区域显示时，需要调用 setVideoDisplayMode(0)；但二者没有调用的先后顺序，只需要在调用 refresh()方法之前即可。

参数：rect - Rectangle 对象，描述显示区域的位置信息。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.18 getVideoDisplayArea

原型：Rectangle getVideoDisplayArea()

描述：获取视频窗口显示区域的位置信息。

参数：无。

返回：Rectangle 对象，表示视频窗口显示区域的位置信息。

Q.2.2.3.19 setVolume

原型：number setVolume(volume)

描述：设置当前音量。若当前播放的是广播业务，应记忆该频道相对于全局音量的增量。

注：音量增值 = 当前音量 - 全局音量

参数：volume - number 型，取值范围 0～100，0 表示最小音量（静音），100 表示最大音量。

返回：number 型，若成功则返回 0，否则返回其他。

Q.2.2.3.20 getVolume

原型：number getVolume()

描述：获取当前音量。

参数：无。

返回：number 型，取值范围 0～100，0 表示最小音量（静音），100 表示最大音量。

Q.2.2.3.21 getCurrentLanguage

原型：string getCurrentLanguage()

描述：获取当前所用的语种。

参数：无。

返回：string 型，表示音频语种，语种三字母代码遵循 GB/T 4880.2—2000 标准。若无法获得，则返回 undefined。

Q.2.2.3.22 listAvailableLanguages

原型：string[] listAvailableLanguages()

描述：获取当前所有可用的音频语种。

参数：无。

返回：string 型数组，表示音频语种列表，语种三字母代码遵循 GB/T 4880.2—2000 标准。若无法获得，则返回长度为 0 的数组。

Q.2.2.3.23 selectDefaultLanguage

原型：string selectDefaultLanguage()

描述：将音频语种设置为缺省语种。

参数：无。

返回：string 型，表示缺省的音频语种，语种三字母代码遵循 GB/T 4880.2—2000 标准。

Q.2.2.3.24 selectLanguage

原型：selectLanguage(language)
描述：设置当前音频所用的语种。
参数：language - string 型，表示音频语种，语种三字母代码遵循 GB/T 4880.2—2000 标准。
返回：无。

Q.2.2.3.25 getMediaDuration

原型：string getMediaDuration()
描述：获取当前播放媒体的总时长。
参数：无。
返回：string 型，格式为“hh:mm:ss”。
——当媒体来源属性为本地媒体文件时，该方法返回正播放媒体的总时长；
——当媒体来源为广播业务、互动电视业务、IP-UDP 码流时，该方法返回 undefined。

Q.2.2.3.26 getCurrentPlayTime

原型：string getCurrentPlayTime()
描述：获取媒体播放的当前时间点。
参数：无。
返回：string 型，表示媒体播放的当前时间点，Normal Play Time 和 Absolute Time 两种时间类型。
——点播用 Normal Play Time 格式；
——时移用 Absolute Time 格式。

Q.2.2.3.27 getPlaybackMode

原型：string getPlaybackMode()
描述：获取播放器当前的播放模式。
参数：无。
返回：string 型，表示 JSON 字符串，其中至少包括“PlayMode”和“Speed”两类信息。
PlayMode 可取值为 Normal、Pause、Trickmode，当为 Trickmode 时必须带 2x/-2x、4x/-4x、8x/-8x、16x/-16x、32x/-32x 参数来表示快进/快退的速度参数。返回值举例：
{“PlayMode”:“Trickmode”，“Speed”:“2x”}。

Q.2.2.3.28 getServiceLocation

原型：string getServiceLocation(flag)
描述：根据节目标志对象获取指定节目的定位器。
参数：flag - number 型，取值范围见表 Q.6。

表 Q.6 flag 取值表

flag	说明
0	表示正在播放的节目。
1	表示前一个播放的节目。
2	表示前一个播放的电视节目。
3	表示前一个播放的广播节目。

返回：string 型，表示业务的定位器（URL）。

Q.2.2.3.29 setPauseMode

原型：setPauseMode(mode)
描述：设置暂停状态时视频的输出模式。
参数：mode - number 型，取值 0 表示黑屏，1 表示静帧。
返回：无。

Q.2.2.3.30 getPauseMode

原型：number getPauseMode()
描述：获取暂停状态视频输出模式。
参数：无。
返回：number 型，取值 0 表示黑屏，1 表示静帧。

附 录 R
（规范性附录）
JavaScript-应用管理单元

R.1 概述

本附录定义了与应用管理相关的功能模块：应用管理模块，应用管理模块功能详细定义见表 R.1。

表 R.1 用管理模块功能定义

功能	描述
应用下载	1、可以从网站或指定 URL、APPSTORE，通过 http、https、DVB OC 数据推送等多种数据通道方式将应用下载到本地存储器； 2、直接从其他存储设备拷贝应用到本地存储器。
应用安装	1、安装包需要通过签名校验后才能安装； 2、安装包解压到本地存储器的指定位置。
应用卸载	1、删除安装包； 2、删除应用在本地存储的缓冲数据文件； 3、删除解压后的目录。
应用升级	1、到服务器上获取应用列表； 2、到服务器上查询更新版本，如果有新版本，就升级。
应用权限管理	1、应用对系统资源的访问权限； 2、应用权限需要划分等级、优先级； 3、应用权限的设置可以通过安装包的配置文件来进行修改配置； 4、应用启动、运行需要在沙箱中进行，也需要权限来控制。
应用信息查询	1、应用运行状态信息查询； 2、应用所拥有的系统资源访问权限信息； 3、应用的版本信息。
应用安全	1、通过数字签名，保证应用来源合法性； 2、生成的安装包数据库，需要 root 权限控制，防止非授权用户修改安装包数据库，保证该文件的安全性； 3、解压后的应用文件与目录树，只有应用管理器有读写权限，防止应用间互相读，保证应用间的隔离； 4、启动与运行、访问资源需要通过权限控制，保证应用运行安全。
应用启动与切换	1、通过配置文件解析出各个应用的大小、图标、授权、多语言、本地化、启动 HTML 文件、特征、文字编码、资源位置； 2、应用与 launcher 的切换； 3、应用与应用间的切换； 4、焦点的管理； 5、独立应用的处理。

R.2 应用管理模块

本模块定义了与应用管理相关的 JS 对象：widget。

R.2.1 window对象扩展属性widget

```
interface partial Window : EventTarget {
      readonly attribute Widget widget;
};
```

R.2.2 widget对象

widget 对象为内置对象，直接使用。

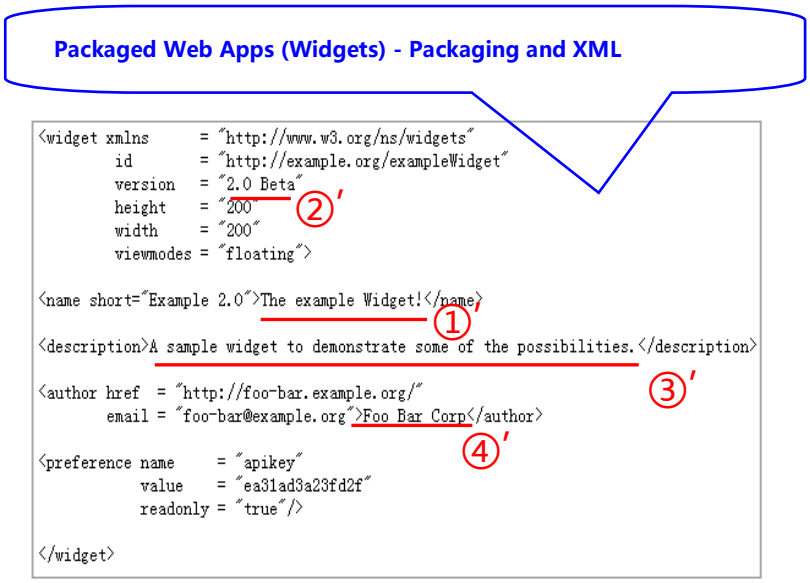
R.2.2.1 概述

widget 对象包含了只读属性、方法、消息反馈，详细定义如下：

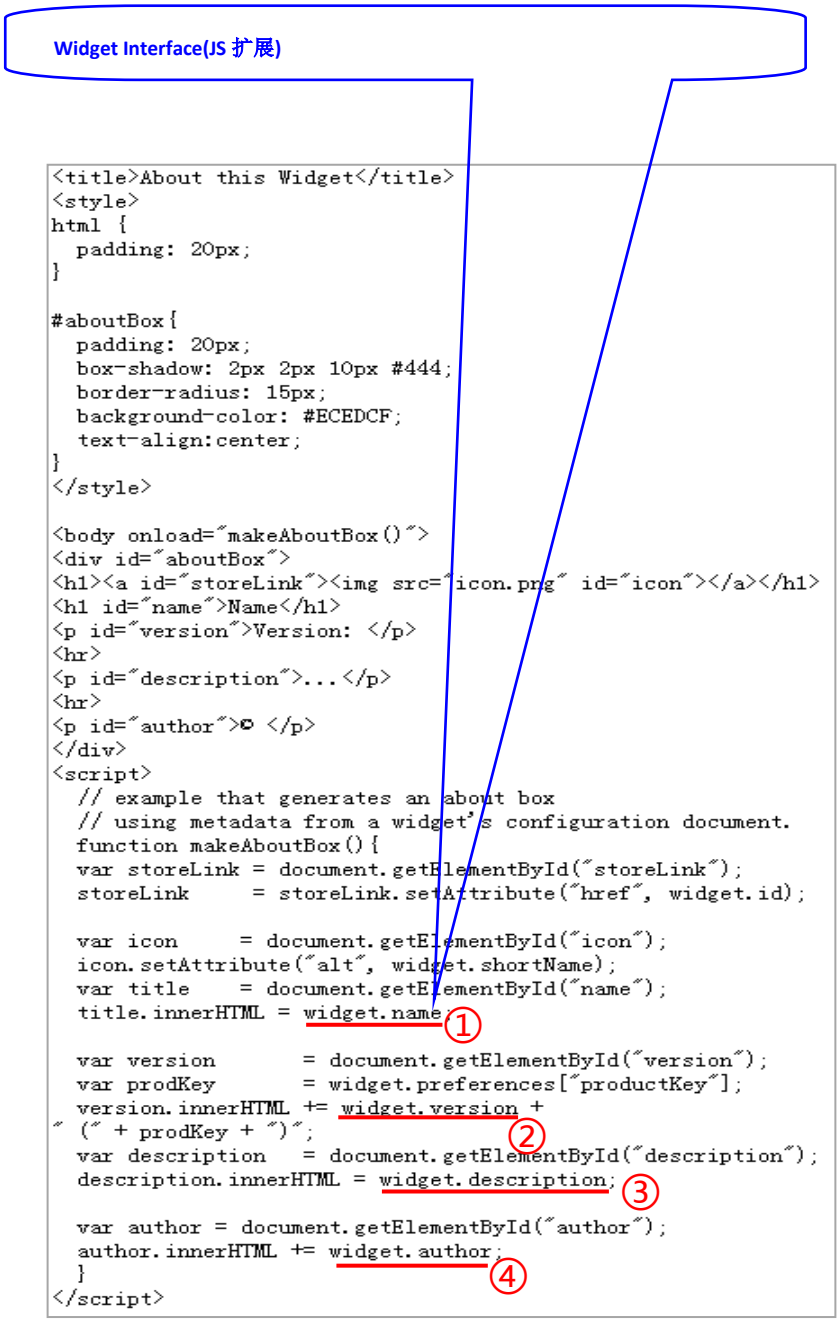
```
interface Widget {
    readonly attribute DOMString      author;      //config.xml 的 author 元素对应
    readonly attribute DOMString      description; //config.xml 的 description 元素对应
    readonly attribute DOMString      name;        //config.xml 的 name 元素对应
    readonly attribute DOMString      shortName;   //config.xml 的 name short 属性对应
    readonly attribute DOMString      version;     //config.xml 的 widget version 属性对应
    readonly attribute DOMString      id;          //config.xml 的 widget id 属性对应
    readonly attribute DOMString      authorEmail; //config.xml 的 author email 属性对应
    readonly attribute DOMString      authorHref;  //config.xml 的 author href 属性对应
    readonly attribute Storage        preferences; //config.xml 的 preference 元素对应
    readonly attribute unsigned long  height;      //config.xml 的 widget height 属性对应
    readonly attribute unsigned long  width;       //config.xml 的 widget width 属性对应
    void launchWidget(DOMString widgetname, DOMString src, DOMString type);
    void installWidget(DOMString widgetname, DOMString id, DOMString url);     //安装 widget
    void updateWidget(DOMString widgetname, DOMString id, DOMString url);      //更新 widget
    DOMString checkWidget(DOMString widgetname, DOMString id, DOMString url);  //检测 widget, 返回版本
号
   void uninstallWidget(DOMString widgetname, DOMString id);                   //卸载 widget
   void deletePackage(DOMString widgetname, DOMString id);                     //删除应用包
   // 下面几个是应用管理器反馈给应用的几个消息
   attribute EventHandler onstart;
   attribute EventHandler onresume;
   attribute EventHandler onpause;
   attribute EventHandler onstop;
   attribute EventHandler onterminate;
   attribute EventHandler ondownloadsuccess;
   attribute EventHandler ondownloadfail;
   attribute EventHandler ondownloadsize;
};
```

示例：

widget 的 config.xml 配置文件如下：



通过 JS widget API 访问 config.xml 的页面代码如下：



R.2.2.2 属性

widget对象的属性定义见表R.2。

表 R.2 widget 对象的属性

属性名称	类型	读写属性	说明
author	string	只读	config.xml 的 author 元素对应
description	string	只读	config.xml 的 description 元素对应
name	string	只读	config.xml 的 name 元素对应
shortName	string	只读	config.xml 的 name short 属性对应
version	string	只读	config.xml 的 widget version 属性对应
id	string	只读	config.xml 的 widget id 属性对应
authorEmail	string	只读	config.xml 的 author email 属性对应
authorHref	string	只读	config.xml 的 author href 属性对应
preferences	Storage	只读	config.xml 的 preference 元素对应
height	Number unsigned long	只读	config.xml 的 widget height 属性对应
width	Number unsigned long	只读	config.xml 的 widget width 属性对应

R.2.2.3 方法

R.2.2.3.1 launchWidget

原型：void launchWidget(DOMString widgetname, DOMString src, DOMString type)
描述：Launcher上启动widget 或widget回到launcher上。
参数： widgetname – DOMString类型，为启动widget，为字符串，如果widgetname为widget.launcher.home，表示要回到launcher上；
src - DOMString类型,是通过其widget的config.xml文件的content元素的src来获取的；
type - DOMString类型,是通过其widget的config.xml文件的content元素的type来获取的。
返回：无。

R.2.2.3.2 installWidget

原型：void installWidget(DOMString widgetname, DOMString id, DOMString url)
描述：下载、安装 widget，如果 widget 不存在，就先从 url 指定的位置下载。
参数： widgetname - DOMString 类型，为 widget 名字，为字符串；
id - DOMString 类型，为 widget 识别号，为字符串；
url - DOMString 类型，为指定 widget 下载的位置，为字符串。
返回：无。

R.2.2.3.3 updateWidget

原型：void updateWidget(DOMString widgetname, DOMString id, DOMString url)
描述：更新 widget。
参数： widgetname - DOMString 类型，为 widget 名字，为字符串；
id - DOMString 类型，为 widget 识别号，为字符串；

url - DOMString 类型，为指定 widget 下载的位置，为字符串。

返回：无。

R.2.2.3.4 checkWidget

原型：DOMString checkWidget(DOMString widgetname, DOMString id, DOMString url)

描述：检测 widget。

参数： widgetname - DOMString 类型，为 widget 名字，为字符串；

id - DOMString 类型，为 widget 识别号，为字符串；

url - DOMString 类型，为指定 widget 下载的位置，为字符串。

返回：版本号。

R.2.2.3.5 uninstallWidget

原型：void uninstallWidget(DOMString widgetname, DOMString id)

描述：卸载 widget。

参数： widgetname - DOMString 类型，为 widget 名字，为字符串；

id - DOMString 类型，为 widget 识别号，为字符串；

返回：无。

R.2.2.3.6 deletePackage

原型：void deletePackage(DOMString widgetname, DOMString id)

描述：删除应用包。

参数： widgetname - DOMString 类型，为 widget 名字，为字符串；

id - DOMString 类型，为 widget 识别号，为字符串；

返回：无。

R.2.2.4 消息回调

消息回调，用于反馈当前应用进程状态和下载状态，回调详细信息见表R.3。

表 R.3 widget 对象的消息回调

回调名称	说明
onstart	应用进程启动时反馈状态
onresume	应用进程运行时反馈状态
onpause	应用进程暂停时反馈状态
onstop	应用进程停止时反馈状态
onterminate	应用进程中断时反馈状态
ondownloadsuccess	下载应用成功时反馈状态
ondownloadfail	下载应用失败时反馈状态
ondownloadsize	下载应用当前时反馈状态，包括总大小、当前字节数，详细见： interface WidgetEvent : Event { readonly attribute long widgetEvenTtype; readonly attribute long download_totoalsize; readonly attribute long download_currSize;};

附 录 S
（规范性附录）
JavaScript-系统管理单元

S.1 概述

本附录定义了与系统管理相关的功能模块：数据管理、存储设备管理、文件管理、多媒体文件、OTA软件升级、系统工具和软硬件信息查询。

S.2 数据管理模块

本模块定义了与数据管理相关的JS对象：DataConfig。

S.2.1 消息

数据管理模块可能发给应用层的消息定义见表S.1。

表S.1 数据管理模块消息定义表

消息名称	event.which	event.modifiers	消息说明
MSG_SAVE_DATA_SUCCESS	14001	-	数据写入完成。
MSG_SAVE_DATA_FAILED	14002	-	数据写入未完成。
MSG_REMOVE_DATA_SUCCESS	14003	-	数据删除完成。
MSG_REMOVE_DATA_FAILED	14004	-	数据删除未完成。
MSG_REVERT_DATA_SUCCESS	14005	-	数据恢复完成。
MSG_REVERT_DATA_FAILED	14006	-	数据恢复未完成。
MSG_RESTORE_TO_DEFAULT_SUCCESS	14007	-	数据恢复出厂设置完成。
MSG_RESTORE_TO_DEFAULT_FAILED	14008	-	数据恢复出厂设置未完成。
保留	14009～14100		

注：event.modifiers值由系统内部自动给出，其数据类型：

——“number”，表示该值为消息描述字符串的ID，可通过Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串JSON格式，则按格式取出消息内容。

——“-”，表示event.modifiers为undefined。

S.2.2 DataConfig对象

DataConfig对象为内置对象。

系统参数和用户参数采用数据表的形式存于接收终端NVM中，数据表采用“键名+键值”的方式访问。键值在RAM和NVM中都以JSON格式存储，应用在读取数据时，应根据键名所对应的应用场景进行解析。

系统参数由系统提供，存放于系统数据表中；用户参数由应用程序设定，存放于用户数据表中。系统数据表用以存储系统设置的、全局的配置信息，键名和键值JSON格式对外公开，授权应用可设置/读取参数，但不能删除数据项；用户数据表由应用自己维护，键名和键值JSON格式不对外公开。

接收终端开机后，自动将数据表从NVM读取到RAM中；用户进行参数设置后，可将RAM中的数据保存到NVM中。

注：系统保存下列功能的设置值，下次开机使用保存值，不在系统信息表中显示。

音量设置、设置S/PDIF输出接口数据格式、声道设置，

视频分辨率、图形层透明度、显示宽高比。

系统数据表的键名和键值JSON格式见表S.2。

表S.2　数据管理模块消息定义表

<table>
<tr><th>键名</th><th colspan="3">键值</th></tr>
<tr><td rowspan="6">DVBMainFrequencyInfo</td><td>用途</td><td colspan="2">描述DVB主频点信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2">[
{
"deliveryType":1,
"deliveryParams":[
{"frequency":626000, "symbolRate":6875, "modulation":3},
{"frequency":634000, "symbolRate":6875, "modulation":3},
{"frequency":642000, "symbolRate":6875, "modulation":3}
]
},
...
]
注：DVBMainFrequencyInfo键的键值JSON格式可以描述多个传送系统，每个传送系统可以描述多个主频点信息。在本示例中，上述JSON字符串描述了DVB-C传送系统，该传送系统包含3个主频点信息。</td></tr>
<tr><td rowspan="4">说明</td><td>deliveryType</td><td>int型，表示传送系统类型，可取值详见DeliverySystemType接口“传送系统类型”常量域定义。</td></tr>
<tr><td>frequency</td><td>int型，表示调谐频率，单位与deliveryType字段的取值有关：
——若deliveryType=1，表示DVB-C传送系统，单位为千赫（kHz）；
——若deliveryType为其他值，则该键无意义。</td></tr>
<tr><td>symbolRate</td><td>int型，表示符号率，单位为千符号每秒（ksymbol/s）。</td></tr>
<tr><td>modulation</td><td>int型，表示调制方式，取值与deliveryType字段的取值有关：
——若deliveryType=1，表示DVB-C传送系统，取值详见DvbcTunningParameters类的“调制方式”常量域定义；
——若deliveryType为其他值，则该键无意义。</td></tr>
</table>

表 S.2（续）

键名	键值		
EPGSetting	用途	描述EPG搜索设置信息。	
	JSON格式	{"search_start_date":0, "search_days":7, "program_event_maxcount":255}	
	说明	search_start_date	int型，表示系统搜索节目表的起始日期。缺省为0，即将当天作为搜索的起始日期，1表示第二天，2表示第三天,……,依次类推。
		search_days	int型，表示系统将搜索连续多少天的节目表。
		program_event_max_count	int型，表示EPG搜索节目事件个数的最大值，若取值-1，则表示无限制。
AudioSetting	用途	描述音频设置信息。	

	JSON格式	{
		"enableGlobalVolume":0
		}	

	说明	enableGlobalVolume	int型，表示是否启用统一音量控制，取值： ——表示是否表示应用允许用户单独设置每个频道的音量； ——应用允许表示所有直播电视、音频广播、NVOD、马赛克的音量统一为outputVolume值。
UserPreference	用途	描述用户偏好设置信息。	

	JSON格式	{
		"audioLang":"zho",
		"osdLang":"eng"
		}	

	说明	audioLang	字符串，表示用户偏好的音频语种，语种三字母代码遵循GB/T 4880.2—2000。
		osdLang	字符串，表示用户偏好的界面语种，语种三字母代码遵循GB/T 4880.2—2000。
Portal	用途	描述门户服务器地址信息。	

	JSON格式	{
		"address":"http://ngb.com",
		"port":8080,
		"returnChannel":“modulator"
		}	

	说明	address	字符串，表示门户服务器地址。
		port	int型，表示门户服务器访问端口。
		returnChannel	字符串，表示回传通道，取值modulator与ethernet分别表示网络调制器与以太网

表 S.2 （续）

<table>
<tr><th>键名</th><th colspan="3">键值</th></tr>
<tr><td rowspan="4">NTP</td><td>用途</td><td colspan="2">描述NTP服务器地址信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2">{
"address":"http://ntp.com",
"port":8080
}</td></tr>
<tr><td rowspan="2">说明</td><td>address</td><td>字符串，表示NTP服务器地址。</td></tr>
<tr><td>port</td><td>int型，表示NTP服务器访问端口。</td></tr>
<tr><td rowspan="6">VODChannel</td><td>用途</td><td colspan="2">描述VOD服务器地址信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2">{
"MD5":"5A8B6493",
"VODParams":[
{
"frequency":634000,
"symbolRate":6875,
"modulation":3,
"QAMName":1234
},
{
"frequency":642000,
"symbolRate":6875,
"modulation":3,
"QAMName":4321
},
{
"frequency":650000,
"symbolRate":6875,
"modulation":3,
"QAMName":8888
}
]
}
注：仅适用于有线数字电视。</td></tr>
<tr><td rowspan="5">说明</td><td>MD5</td><td>字符串，表示配置文件MD5码。</td></tr>
<tr><td>frequency</td><td>int型，表示IPQAM频点频率，单位为千赫（kHz）。</td></tr>
<tr><td>symbolRate</td><td>int型，表示IPQAM频点符号率，单位为千符号每秒（ksymbol/s）。</td></tr>
<tr><td>modulation</td><td>int型，表示IPQAM频点调制方式，取值详见DvbcTunningParameters类的“调制方式”常量域定义。</td></tr>
<tr><td>QAMName</td><td>int型，表示VOD区域码。</td></tr>
</table>

表 S.2（续）

<table>
<tr><th>键名</th><th colspan="3">键值</th></tr>
<tr><td rowspan="3">UserInfo</td><td>用途</td><td colspan="2">描述终端用户信息。</td></tr>
<tr><td>JSON
格式</td><td colspan="2">{
"adminPassword":"12345"
}</td></tr>
<tr><td>说明</td><td>adminPassword</td><td>字符串，表示管理员密码，用于进入系统设置界面和观看加锁频道。</td></tr>
<tr><td rowspan="2">Autodeployer</td><td>用途</td><td colspan="2">描述应用自动部署信息。</td></tr>
<tr><td>JSON
格式</td><td colspan="2">{
"mode":"auto-ip",
"ocPath":[
{
"deliveryType":1,
"deliveryParams":[
{"frequency":626000, "symbolRate":6875, "modulation":3},
{"frequency":634000, "symbolRate":6875, "modulation":3},
{"frequency":642000, "symbolRate":6875, "modulation":3}
]
},
...
],
"ipPath":[
{
"udpPath":"192.168.1.12",
"udpPort":8080,
"downloadTimeOut":60
},
...
]
}
注：单向广播通道支持多个传送系统、多个频点下发XML信令文件；双向宽带通道支持多个UDP服务器下发XML信令文件。</td></tr>
</table>

表 S.2（续）

<table>
<tr><th>键名</th><th colspan="3">键值</th></tr>
<tr><td rowspan="3">Autodeployer</td><td rowspan="3">说明</td><td>mode</td><td>字符串，表示自动部署XML信令文件的获取方式，取值：
——文件的获取方式表示从双向宽带通道获取XML信令文件；
——文件；宽带通道表示从单向广播通道获取XML信令文件；
——文件；广播通道获取值：多表示自适应，优先从双向宽带通道获取XML信令文件；
——文件；，优先从双向宽带通表示自适应，优先从单向广播通道获取XML信令文件。</td></tr>
<tr><td>ocPath</td><td>JSON对象类型，表示自动部署xml信令文件的OC下载路径。
——路径。表示自动部署广播通：int型，表示传送系统类型，同DVBMainFrequencyInfo键的解释；
——解释；inFrequency：JSON对象，同DVBMainFrequencyInfo键的解释。</td></tr>
<tr><td>ipPath</td><td>JSON对象类型，表示自动部署xml信令文件的ip下载路径。
——路径。表示自动：字符串，表示UDP服务器地址；
——器地址；示自动：int型，表示UDP服务端口；
——端口；；示自动部署uencyI：int型，表示应用下载超时时间，单位为秒。</td></tr>
<tr><td rowspan="6">SeaChangeVOD</td><td>用途</td><td colspan="2">描述SeaChange VOD 相关信息。</td></tr>
<tr><td>JSON格式</td><td colspan="2"><pre>{
"bakerFreq": {
"frequency": 626000,
"symbolRate": 6875,
"modulation": 3
},
"serviceGroupId": 1
}</pre></td></tr>
<tr><td rowspan="4"></td><td>frequency</td><td>int型，表示调谐频率，
——单位为千赫（kHz）；</td></tr>
<tr><td>symbolRate</td><td>int型，表示符号率，单位为千符号每秒（ksymbol/s）。</td></tr>
<tr><td>modulation</td><td>int型，表示调制方式，，取值详见DvbcTunningParameters类的“调制方式”常量域定义；</td></tr>
<tr><td>serviceGroupId</td><td>Service Group Id 值</td></tr>
</table>

S.2.2.1 方法

S.2.2.1.1 createUserPropertyTable

原型：number createUserPropertyTable(name)

描述：创建一个新的用户数据表。

参数：name -string 型，表示用户数据表的名称。

返回：number 型，取值：

——若创建用户数据表成功，则返回值大于 0，用以表示全局唯一的数据表标识；

——若未知原因导致创建用户数据表失败，则返回数值 0；

——若要创建的用户数据表已经存在，则返回数值-1；

——若要创建的用户数据表名称无效（例如为 null 或空字符串），则返回数值-2。

S.2.2.1.2 getUserPropertyTable

原型：number getUserPropertyTable(name)

描述：获取用户数据表。

参数：name- string 型，表示用户数据表的名称。

返回：number 型，取值如下：

——若获取用户数据表成功，则返回值大于 0，用以表示全局唯一的数据表标识；

——若未知原因导致获取用户数据表失败，则返回数值 0；

——若要获取的用户数据表不存在，则返回数值-1；

——若要获取的用户数据表名称无效（例如为 null 或空字符串），返回数值-2。

S.2.2.1.3 deleteUserPropertyTable

原型：number deleteUserPropertyTable(tableID)

描述：删除用户数据表。

参数：tableID- number 型，用户数据表标识。

返回：number 型，取值如下：

——若删除用户数据表成功，则返回值大于 0，用以表示全局唯一的数据表标识；

——若未知原因导致删除用户数据表失败，则返回数值 0；

——若要删除的用户数据表不存在，则返回数值-1；

——若要删除的用户数据表名称无效（例如为 null 或空字符串），则返回数值-2。

S.2.2.1.4 getSystemPropertyTable

原型：number getSystemPropertyTable()

描述：获取系统数据表。

参数：无。

返回：number 型，取值如下：

——若获取系统数据表成功，则返回值大于 0，用以表示全局唯一的数据表标识；

——若未知原因导致获取系统数据表失败，则返回数值 0；

——若要获取的系统数据表不存在，则返回数值-1。

S.2.2.1.5 createItem

原型：createItem(tableID,strItem,strValue)

描述：在用户数据表中创建一个数据项，并给这个数据项的内容赋予一个初始值。对系统数据表操作无效。

参数：tableID - number 型，用户数据表标识。

strItem - string 型，新创建的数据项的键名。

strValue - string 型，JSON 格式字符串，表示新创建的数据项的键值。

返回：number 型，取值如下：

——若创建数据项成功，则返回值大于 0；

——若未知原因导致创建数据项失败，则返回数值 0；

——若输入的数据表 ID 不存在，则返回数值-1；

——若输入的数据项的名称在该数据表下已存在，则返回数值-2；

——若输入的数据项的名称无效（例如为 null 或空字符串），则返回数值-3。

S.2.2.1.6 deleteItem

原型：deleteItem(tableID,strItem)

描述：删除用户数据表中的一个数据项。对系统数据表操作无效。

参数：tableID - number 型，数据表标识。

strItem - string 型，数据项名称。

返回：number 型，取值如下：

——若删除成功，则返回值大于 0；

——若未知原因导致删除数据项失败，则返回数值 0；

——若该数据项不存在，则返回数值-1。

S.2.2.1.7 getProperty

原型：string getProperty(tableID, strItem)

描述：获取某个数据项的值（从内存中读）。对系统数据表和用户数据表均可操作。

参数：tableID - number 型，数据表标识。

strItem -string 型，数据项名称。

返回：string 型，数据项值；若该数据项不存在，则返回 null。

S.2.2.1.8 setProperty

原型：number setProperty(tableID,strItem,strValue)

描述：设置某个数据项的值（写入到内存）。对系统数据表和用户数据表均可操作。

参数：tableID - number 型，表示数据表标识；

strItem - string 型，表示数据项名称；

strValue - string 型，表示数据项值。

返回：number型，取值：

——若修改成功，则返回值大于0；

——若未知原因导致修改内容失败，则返回数值0；

——若该数据项不存在，则返回数值-1。

S.2.2.1.9 saveToNvm

原型：boolean saveToNvm(tableID)

描述：异步方法，将 RAM 中的数据表写入到 NVM 中，覆盖 NVM 中的数据表。对系统数据表和用户数据表均可操作。

——若数据写入成功，则向页面发送消息 MSG_SAVE_DATA_SUCCESS；

——若数据写入失败，则向页面发送消息 MSG_SAVE_DATA_FAILED。

参数：tableID - number 型，数据表标识。

返回：boolean 型，true 表示开始执行，false 表示未执行，执行结果是否成功通过捕获消息得知。

S.2.2.1.10 removeFromNvm

原型：boolean removeFromNvm(tableID)

描述：异步方法，将NVM中由tableID参数指定的用户表删除。

——若数据删除成功，则向页面发送消息MSG_REMOVE_DATA_SUCCESS；

——若数据删除失败，则向页面发送消息MSG_REMOVE_DATA_FAILED。

参数：tableID - number型，表示数据表标识。

返回：boolean型，true表示开始执行，false表示未执行，执行结果是否成功通过捕获消息得知。

S.2.2.1.11 revertFromNvm

原型：boolean revertFromNvm(tableID)

描述：异步方法，将NVM中的数据表导入到RAM中，覆盖内存中的当前数据表。对系统数据表和用户数据表均可操作。

——若数据覆盖成功，则向页面发送消息MSG_REVERT_DATA_SUCCESS；

——若数据覆盖失败，则向页面发送消息MSG_REVERT_DATA_FAILED。

参数：tableID - number型，表示数据表标识。

返回：boolean型，true表示开始执行，false表示未执行，执行结果是否成功通过捕获消息得知。

S.2.2.1.12 restoreDefault

原型：boolean restoreDefault()

描述：异步方法，将NVM中的系统数据表恢复到出厂设置状态，并同步更新RAM中的系统数据表。

——若数据恢复成功，则向页面发送消息MSG_RESTORE_TO_DEFAULT_SUCCESS；

——若数据恢复失败，则向页面发送消息MSG_RESTORE_TO_DEFAULT_FAILED。

参数：无。

返回：boolean型，true表示开始执行，false表示未执行，执行结果是否成功通过捕获消息得知。

S.3 外部存储设备管理模块

本模块定义了与外部存储设备管理相关的JS对象：StorageDeviceManager、StorageDevice、StoragePartition。

S.3.1 消息

外部存储设备管理模块可能发给应用层的消息定义见表S.3。

表S.3 外部存储设备管理模块消息定义表

消息名称	event.which	event.modifiers	消息格式
MSG_DEVICE_UNINSTALL_SUCCESS	14101	-	设备卸载成功。
MSG_DEVICE_UNINSTALL_FAILED	14102	-	设备卸载失败。
保留	14103～14200		

注：event.modifiers值由系统内部自动给出，其数据类型：

——“number”，表示该值为消息描述字符串的ID，可通过Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串JSON格式，则按格式取出消息内容。

——“-”，表示event.modifiers为undefined。

S.3.2 StorageDeviceManager对象

StorageDeviceManager 对象为内置对象，该对象完成外部存储设备的管理、设置操作。

S.3.2.1 方法

S.3.2.1.1 uninstallDeviceByID

原型：public boolean uninstallDeviceByID(id)
描述：异步方法，根据设备 ID 卸载设备。
——若卸载成功则返回“MSG_UNINATALL_DEVICE_SUCCESS”消息；
——若卸载失败则返回“MSG_UNINATALL_DEVICE_FAILED”消息。
参数：id - number 型，设备 ID。
返回：boolean 型，取值 true 表示开始执行，取值 false 表示未执行，执行结果是否成功通过捕获消息得知。

S.3.2.1.2 getAllStorageDevices

原型：public StorageDevice[] getAllStorageDevices()
描述：获取所有存储设备信息。
参数：无。
返回：StorageDevice 对象数组。

S.3.3 StorageDevice对象

StorageDevice 对象为本地对象，该对象描述了存储设备信息，包括名称、大小、空闲空间等。
示例：

```
//通过 StorageDeviceManager 方法，创建 StorageDevice 对象。
var storageDeviceArray = StorageDeviceManager.getAllStorageDevices();
var storageDevice = storageDeviceArray[0];
```

S.3.3.1 属性

StorageDevice 对象的属性定义见表 S.4。

表 S.4 StorageDevice 属性表

属性名称	类型	读写属性	说明
id	number	只读	获取存储设备唯一标识。
name	string	只读	获取存储设备名称。
status	number	只读	获取存储设备的状态信息，取值： ——0 - 表示存储设备已卸载； ——1 - 表示发现存储设备并已确定类型； ——2 - 表示存储设备已就绪； ——3 - 表示存储设备不可用。
serialNumber	string	只读	返回存储设备序列号。

S.3.3.2 方法

S.3.3.2.1 getAllPartitions

原型：public StoragePartition[] getAllPartitions()
描述：获取设备所有分区信息。
参数：空。
返回：StoragePartition 对象数组。

S.3.3.2.2 getPartitionByID

原型：public StoragePartition getPartitionByID(id)
描述：获取指定 ID 的分区对象。
参数：id －number 型，指定分区 ID。
返回：StoragePartition 对象。

S.3.4 StoragePartition对象

StoragePartition 对象为本地对象。
该对象描述了存储设备的分区信息，包括名称、大小、空闲状态、访问路径、分区类型等。
示例：

```
//通过 StorageDevice 方法，创建 StoragePartition 对象。
var storagePartitionArray= storageDevice.getAllParitions();
var storagePartition = storagePartitionArray[0];
```

S.3.4.1 属性

StoragePartition 对象的属性定义见表 S.5。

表 S.5 StoragePartition 属性表

属性名称	类型	读写属性	说明
id	number	只读	分区 ID，即分区的唯一标识。
name	string	只读	获取存储设备分区名称。
totalSize	number	只读	获取存储设备分区空间大小，单位为千字节（KB）。
freeSize	number	只读	获取存储设备分区空闲空间大小，单位为千字节（KB）。
path	string	只读	获取存储设备分区访问路径。
fsType	string	只读	获取分区文件系统类型，或称分区格式，取值主要有 E2FS、FAT32、NTFS 等。
fsStatus	string	只读	表示存储设备分区状态，例如“状态良好”、“未格式化”等。

S.4 文件管理模块

本模块定义了与文件管理相关的 JS 对象：FileManager、FileObj、Directory，各对象之间的关系见图 S.1。

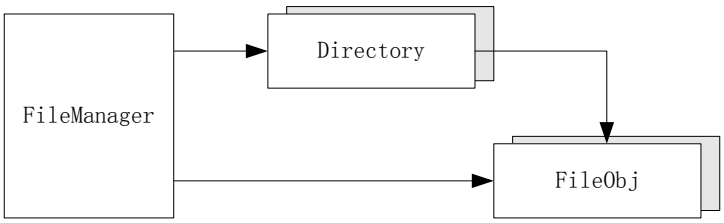


图 S.1 文件资源管理对象关系

1）本地存储文件系统

接收终端在本地存储中分配一块区域给系统（目前接收终端的本地存储通常用闪存实现，未来可能支持其他本地存储介质），供系统存放下载的 web 主应用、其他应用、本地配置信息、PSI/SI 信息、用户文件等数据。对于系统及应用而言，在本地存储中只有该区域是可写的，由系统和应用共同维护使用。该区域在应用中统一以/storage/storage0 标识其根路径，未来可能基于业务需要，增加第二、第三个可写区域，分别用/storage/storage1、/storage/storage2 标识，目前终端只需支持/storage/storage0 即可。

示例：

——以/storage/storage0/startapp/路径，访问开机应用存放的数据；

——以/storage/storage0/adv/路径，访问广告业务存放的数据

接收终端的/storage/storage0 路径与 linux 文件系统路径（例如/mnt/hd/HDD0）之间的对应关系，由系统自行实现。

2）USB 文件系统

在接收终端 USB 接口接入的存储设备，其路径使用本附录中的外部存储设备管理模块接口相关对象来获取。

S.4.1 消息

文件管理模块消息定义见表S.6。

表 S.6 文件管理模块消息

消息名称	event.which	event.modifiers	消息格式
MSG_COPYFILE_SUCCESS	14201	-	文件复制成功。
MSG_COPYFILE_FAILED	14202	-	文件复制失败。
MSG_FILE_NOT_EXIST	14203	-	源文件不存在。
MSG_SPACE_SHORTAGE	14204	-	目标存储空间不足。
MSG_MOVEFILE_SUCCESS	14205	-	文件移动成功。
MSG_MOVEFILE_FAILED	14206	-	文件移动失败。
MSG_CANNOT_DELETE_FILE	14207	-	无法删除源文件。
MSG_DELETEFILE_SUCCESS	14208	-	文件删除成功。
MSG_DELETEFILE_FAILED	14209	-	文件删除失败。
保留	14210～14300		
MSG_COPYDIRECTORY_SUCCESS	14301	-	目录复制成功。
MSG_COPYDIRECTORY_FAILED	14302	-	目录复制失败。
MSG_DIRECTORY_NOT_EXIST	14303	-	源目录不存在。

表 S.6（续）

消息名称	event.which	event.modifiers	消息格式
MSG_MOVEDIRECTORY_SUCCESS	14305	-	目录移动成功。
MSG_MOVEDIRECTORY_FAILED	14306	-	目录移动失败。
MSG_CANNOT_DELETE_DIRECTORY	14307	-	无法删除源目录。
MSG_DELETEDIRECTORY_SUCCESS	14308	-	目录删除成功。
MSG_DELETEDIRECTORY_FAILED	14309	-	目录删除失败。
MSG_DOWNLOAD_REMOTE_FILE_SUCCESS	14310	number	已下载前端文件到内存中。
MSG_DOWNLOAD_REMOTE_FILE_NOT_EXIST	14311	number	前端不存在要下载的文件。
MSG_DOWNLOAD_REMOTE_FILE_FAILED	14312	number	前端文件下载失败。
MSG_DOWNLOAD_REMOTE_FILE_TIMEOUT	14313	number	前端文件下载超时。
MSG_REMOTE_FILE_EXIST	14314	number	前端存在指定的文件。
MSG_REMOTE_FILE_NOT_EXIST	14315	number	前端不存在指定的文件。
MSG_REMOTE_FILE_SEARCH_TIMEOUT	14316	number	查找前端文件超时。
保留	14317～14400		

S.4.2 FileManager对象

FileManager 对象为内置对象，用于管理本地目录和文件。

S.4.2.1 方法

S.4.2.1.1 copyFile

原型：boolean copyFile(path1, path2)

描述：异步方法，将path1指定的文件复制到path2路径，不删除源文件。

——若复制成功，则向页面发送消息MSG_COPYFILE_SUCCESS；

——若源文件不存在，则向页面发送消息MSG_FILE_NOT_EXIST；

——若目标路径所在的存储区空间不足，则向页面发送消息MSG_SPACE_SHORTAGE；

——若复制失败，则向页面发送消息MSG_COPYFILE_FAILED。

参数：path1- string型，源文件的路径（含文件名），有以下两种方式：

——类似/storage/storage0/xx/xx.txt的本地存储文件路径；

——通过设备管理接口相关对象获取的USB外置存储文件路径。

path2- string型，目标文件的路径（含文件名），有以下两种方式：

——类似/storage/storage0/xx/xx.txt的本地存储文件路径；

——通过设备管理接口相关对象获取的USB外置存储文件路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.2 moveFile

原型：boolean moveFile(path1, path2)

描述：异步方法，将path1指定的文件复制到path2路径，删除源文件。

——若移动成功，则向页面发送消息MSG_MOVEFILE_SUCCESS；

——若源文件不存在，则向页面发送消息MSG_FILE_NOT_EXIST；

——若目标路径所在的存储区空间不足，则向页面发送消息MSG_SPACE_SHORTAGE；
——若无法删除源文件，则向页面发送消息MSG_CANNOT_DELETE_FILE；
——若移动失败，则向页面发送消息MSG_MOVEFILE_FAILED。

参数：path1- string型，源文件的路径（含文件名），有以下两种方式：
——类似/storage/storage0/xx/xx.txt的本地存储文件路径；
——通过设备管理接口相关对象获取的USB外置存储文件路径。
path2- string型，目标文件的路径（含文件名），有以下两种方式：
——类似/storage/storage0/xx/xx.txt的本地存储文件路径；
——通过设备管理接口相关对象获取的USB外置存储文件路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.3 deleteFile

原型：boolean deleteFile(path)

描述：异步方法，删除一个本地文件。
——若删除成功，则向页面发送消息MSG_DELETEFILE_SUCCESS；
——若删除失败，则向页面发送消息MSG_DELETEFILE_FAILED。

参数：path- string型，要删除的文件的路径（含文件名），有以下两种方式：
——类似/storage/storage0/xx/xx.txt的本地存储文件路径；
——通过设备管理接口相关对象获取的USB外置存储文件路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.4 existLocalFile

原型：boolean existLocalFile(path)

描述：判断一个本地文件是否存在。

参数：path- string型，本地文件的路径（含文件名），有以下两种方式：
——类似/storage/storage0/xx/xx.txt的本地存储文件路径；
——通过设备管理接口相关对象获取的USB外置存储文件路径。

返回：boolean型，取值true表示文件存在，false表示文件不存在。

S.4.2.1.5 copyDirectory

原型：boolean copyDirectory(path1, path2)

描述：将path1指定的目录及目录下的所有内容复制到path2路径，不删除源目录及其内容
——若复制成功，则向页面发送消息MSG_COPYDIRECTORY_SUCCESS；
——若源文件不存在，则向页面发送消息MSG_DIRECTORY_NOT_EXIST；
——若目标路径所在的存储区空间不足，则向页面发送消息MSG_SPACE_SHORTAGE；
——若复制失败，则向页面发送消息MSG_COPYDIRECTORY_FAILED。

参数：path1- string型，源目录的路径，可以有以下两种方式：
——类似/storage/storage0/xx的本地存储目录路径；
——通过设备管理接口相关对象获取的USB外置存储目录路径。
path2- string型，目标目录的路径，可以有以下两种方式：
——类似/storage/storage0/xx的本地存储目录路径；
——通过设备管理接口相关对象获取的USB外置存储目录路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.6 moveDirectory

原型：boolean moveDirectory(path1, path2)

描述：将path1指定的目录及目录下的所有内容复制到path2路径，且删除源目录及其内容

——当移动成功，则向页面发送消息MSG_MOVEDIRECTORY_SUCCESS；

——若源文件不存在，则向页面发送消息MSG_DIRECTORY_NOT_EXIST；

——若目标路径所在的存储区空间不足，则向页面发送消息MSG_SPACE_SHORTAGE；

——若无法删除源文件，则向页面发送消息MSG_CANNOT_DELETE_DIRECTORY；

——若移动失败，则向页面发送消息MSG_MOVEDIRECTORY_FAILED。

参数：path1- string型，源目录的路径，可以有以下两种方式：

——类似/storage/storage0/xx的本地存储目录路径；

——通过设备管理接口相关对象获取的USB外置存储目录路径。

path2- string型，目标目录的路径，可以有以下两种方式：

——类似/storage/storage0/xx的本地存储目录路径；

——通过设备管理接口相关对象获取的USB外置存储目录路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.7 deleteDirectory

原型：boolean deleteDirectory(path)

描述：删除一个目录及其所有内容。

——若删除成功，则向页面发送消息MSG_DELETEDIRECTORY_SUCCESS；

——若删除失败，则向页面发送消息MSG_DELETEDIRECTORY_FAILED。

参数：path- string型，表示待删除的目录的路径，可以有以下两种方式：

——类似/storage/storage0/xx的本地存储目录路径；

——通过设备管理接口相关对象获取的USB外置存储目录路径。

返回：boolean型，取值true表示开始执行，false表示未执行，执行结果通过捕获消息得知。

S.4.2.1.8 existDirectory

原型：boolean existDirectory(path)

描述：判断一个本地目录是否存在。

参数：path- string型，表示待检测的本地目录的路径，可以有以下两种方式：

——类似/storage/storage0/xx的本地存储目录路径；

——通过设备管理接口相关对象获取的USB外置存储目录路径。

返回：boolean型，取值true表示目录存在，false表示目录不存在。

S.4.2.1.9 downloadRemoteFile

原型：downloadRemoteFile(path,endureTime)

描述：从前端下载一个文件，在本地内存中生成相应的FileObj对象。

参数：path - string型，表示要下载的前端文件路径，可以有以下两种方式：

——类似dvb://onid.tsid.serviceid.componenttag/a/b.txt的OC地址；

——类似http://a.b.c/d/e.txt的http地址。

endureTime - number型，下载的超时时间，单位为秒；如果该值为0，表示本次异步下载过程没有时间限制。

返回：number型，系统为该次异步过程分配唯一的maskId。

备注：该方法为异步操作：如果下载成功，系统向应用发送消息MSG_DOWNLOAD_REMOTE_FILE_SUCCESS“已下载前端文件到内存中”，此时应用可通过FileManager.getRemoteFile(maskId)方法获取FileObj对象；如果前端不存在要下载的文件，系统向应用发送消息MSG_DOWNLOAD_REMOTE_FILE_NOT_EXIST“前端不存在要下载的文件”；如果文件下载失败，系统向应用发送消息MSG_DOWNLOAD_REMOTE_FILE_FAILED“文件下载失败”；如果endureTime规定的时间已到，文件下载仍未完成，系统向应用发送消息MSG_DOWNLOAD_FILE_TIMEOUT“超时时间到，文件下载未完成”，并终止下载操作。

S.4.2.1.10 getRemoteFile

原型：getRemoteFile(maskId)

描述：获取前端文件。应用收到消息 MSG_DOWNLOAD_REMOTE_FILE_SUCCESS“已下载前端文件到内存中”后，可通过此接口取出内存中的 FileObj 对象。获取的 FileObj 对象处于关闭状态，必须通过 FileObj.open(mode)方法打开文件，才能进行文件读取操作。

参数：maskId－number 型，在调用 FileManager.downloadRemoteFile()方法时，系统为该次异步过程自动分配的唯一标识。

返回：FileObj 类型，文件对象。

S.4.2.1.11 killObject

原型：killObject(obj)

描述：清除内存中的 File/Directory 对象，回收内存空间。

参数：obj －FileObj 或 Directory 类型，表示要清除的内存文件/目录对象。

返回：number 型，1 表示清除成功；0 表示清除失败。

S.4.3 Directory对象

Directory 对象为本地对象，该对象用于管理本地磁盘的目录。

示例：

```
//通过构造方法创建 Directory 对象
var dir = new Directory(path);
//通过 Directory 对象的方法获取
var directoryArray = dir.getDirList();
var directory = directoryArray[0];
```

S.4.3.1 属性

Directory对象的属性定义见表S.7。

表 S.7 Directory 对象属性表

属性名称	类型	读写属性	说明
name	string	只读	获取目录的名称。
path	string	只读	获取目录的绝对路径。 1)类似/storage/storage0/xx 的本地存储目录路径； 2)通过设备管理接口相关对象获取的 USB 外置存储目录路径。

S.4.3.2 方法

S.4.3.2.1 Directory

原型：Directory(path)
描述：构造方法，根据指定的路径名创建一个新的 Directory 对象实例。
参数：path - string 型，表示目录路径字符串。

S.4.3.2.2 getFileList

原型：FileObj[] getFileList()
描述：获取当前目录下包含的文件。
参数：无。
返回：File对象数组，若该目录下不含任何文件，则返回的数组长度为0。

S.4.3.2.3 getDirList

原型：Directory[] getDirList()
描述：获取当前目录下包含的子目录。
参数：无。
返回：Directory对象数组，若该目录下不含任何子目录，则返回的数组长度为0。

S.4.4 FileObj对象

FileObj 为本地对象，用于管理文件。该对象相当于本地存储区文件、USB 外置存储文件或前端文件在内存中的一个镜像，除非调用 saveAs(path)方法保存文件，否则实际的文件不会被改变。

S.4.4.1 属性

FileObj对象的属性定义见表S.8。

表 S.8 FileObj 对象属性表

属性名称	类型	读写属性	说明
size	number	只读	获取文件的大小，单位为字节。
name	string	只读	获取文件的名称(含扩展名)。
path	string	只读	获取文件所在的绝对路径(含文件名)，可以有以下四种方式： a) 类似/storage/storage0/xx/xx.txt 的本地存储文件路径； b) 通过设备管理接口相关对象获取的 USB 外置存储文件路径； c) dvb://onid.tsid.serviceid.component_tag/xx/xx.txt 的 oc 地址路径； d) 类似 http://x.x.x.x/xx/xx.txt 的 http 地址路径。

S.4.4.2 方法

S.4.4.2.1 FileObj

原型：FileObj(locator)
描述：构造方法，根据指定的文件路径名创建文件对象。
参数：locator - string 型，表示文件的定位符,格式见 FileObj 对象属性表 path 属性描述。

S.4.4.2.2 open

原型：open(mode)
描述：打开文件，以便进行各种编辑操作。
参数：mode - number 型，1 表示以文本方式打开；0 表示以二进制方式打开。
返回：number 型，1 表示打开成功；0 表示打开失败。

S.4.4.2.3 close

原型：close()
描述：关闭文件，以便进行保存操作。
参数：无。
返回：number 型，1 表示关闭成功；0 表示关闭失败。

S.4.4.2.4 readFile

原型：readFile(len)
描述：从文件中读取 len 字节的数据，并以字符串形式返回。
参数：len - number 型，表示读取 len 字节长度的内容。
返回：string 型，以字符串形式返回读取内容。

S.4.4.2.5 readAllFile

原型：readAllFile()
描述：读取文件全部内容，并以字符串形式返回。
参数：无。
返回：string 型，以字符串形式返回读取内容。

S.5 多媒体文件模块

本模块定义了与多媒体文件相关的 JS 对象：AudioFile、VideoFile 和 ImageFile。

S.5.1 AudioFile对象

AudioFile 对象为本地对象。该对象描述一个音频文件详细信息。

示例：

```
//创建一个 AudioFile 对象。
var audioFile = new AudioFile(path);
```

S.5.1.1 属性

AudioFile 对象的属性定义见表 S.9。

表 S.9 AudioFile 属性表

属性名称	类型	读写属性	说明
type	string	只读	获取音频文件类型，参照 MIME。
location	string	只读	获取音频文件的定位器（资源定位符）。
fileName	string	只读	获取音频文件名称。

表 S.9（续）

属性名称	类型	读写属性	说明
fileSize	number	只读	获取音频文件的大小，单位为字节。
sampleRate	number	只读	采样率，例如 32000、44100、48000，单位为赫兹（Hz）。
numberOfChannels	number	只读	声道数。
duration	number	只读	音频文件的播放时间，单位为秒。
date	Date	只读	音频的发行日期。
title	string	只读	音频的标题。
artist	string	只读	艺术家/表演者。
album	string	只读	专辑名称。

S.5.1.2 方法

S.5.1.2.1 AudioFile

原型：AudioFile(path)

描述：构造方法，根据指定路径来创建一个新的音频文件对象。

参数：path - string 型，表示音频文件路径。

S.5.2 VideoFile对象

VideoFile 对象为本地对象。该对象描述一个视频文件详细信息。

示例：

```
//创建一个 VideoFile 对象。
var videoFile = new VideoFile(path);
```

S.5.2.1 属性

VideoFile 对象的属性定义见表 S.10。

表 S.10 VideoFile 属性表

属性名称	类型	读写属性	说明
typeContainer	string	只读	封装格式，如“AVI”、“MP4”、“MKV”等。
typeVideo	string	只读	视频类型，参照 MIME。
typeAudio	string	只读	伴音音频类型，参照 MIME。
location	string	只读	视频文件的定位器（资源定位符）。
fileName	string	只读	视频文件名称。
fileSize	number	只读	视频频文件的大小，单位为字节。
bitRate	number	只读	播放码率，单位为千比特每秒（kb/s）
duration	number	只读	文件的播放时间，单位为秒。
width	number	只读	视频宽度，单位为像素。
height	number	只读	视频高度，单位为像素。
aspectRatio	string	只读	视频宽高比，如“16:9”。
fps	number	只读	视频帧率。
sampleRate	number	只读	伴音音频采样率，例如 32000、44100、48000，单位为赫兹（Hz）。
numberOfChannels	number	只读	声道数。

S.5.2.2 方法

S.5.2.2.1 VideoFile

原型：VideoFile(path)
描述：构造方法，根据指定的路径来创建一个新的视频文件对象。
参数：path - string 型，表示视频文件路径。

S.5.3 ImageFile对象

ImageFile 对象为本地对象。该对象描述一个图片文件详细信息。
示例：

```
//创建一个 ImageFile 对象。
var imageFile = new ImageFile(path);
```

S.5.3.1 属性

ImageFile 对象的属性见表 S.11。

表 S.11 ImageFile 属性表

属性名称	类型	属性	说明
type	string	只读	图片文件的类型，参照 MIME。
location	string	只读	图片文件的定位器（资源定位符）。
fileName	string	只读	图片文件的名字。
width	number	只读	图片文件的宽度，单位为像素。
height	number	只读	图片文件的高度，单位为像素。

S.5.3.2 方法

S.5.3.2.1 ImageFile

原型：ImageFile(path)
描述：构造方法，根据指定的路径来创建一个新的图片文件对象。
参数：path - string 型，图片文件路径。

S.6 OTA软件升级模块

本模块定义了与OTA软件升级相关的JS对象：Upgrade。

S.6.1 消息

OTA软件升级模块可能发给应用层的消息定义见表S.12。

表 S.12　OTA 软件升级模块消息定义表

消息名称	event.which	event.modifiers	消息说明
MSG_FORCE_OTA	22101	-	表示强制 OTA 升级，系统自动下载机顶盒软件映像并升级机顶盒。
MSG_OPTIONAL_OTA	22102	-	表示非强制 OTA 升级，用户可以选择是否本次升级。
保留	22103～22200		

S.6.2　Upgrade对象

Upgrade 对象为内置对象，该对象提供 OTA 或本地升级操作的 JS 方法。

S.6.2.1　方法

S.6.2.1.1　checkOTA

原型：boolean checkOTA()
描述：判断前端是否布署新的 OTA 升级。该方法主要用于手动检测 OTA 升级信息。
参数：无。
返回：boolean 型，true 表示有 OTA 升级，false 表示无 OTA 升级。

S.6.2.1.2　startOTA

原型：startOTA()
描述：启动 OTA 升级。
参数：无。
返回：无。

S.6.2.1.3　getOTAName

原型：string getOTAName()
描述：获取 OTA 升级事件名称，不同于 OTA 提供者名称，指本次 OTA 升级事件的文本描述。
参数：无。
返回：string 型，表示 OTA 升级事件名称。

S.7　系统工具模块

本模块定义了工具类 JS 对象：Utility、GlobalVarManager、Rectangle 和 SysTool。

S.7.1　Utility对象

Utility 对象为内置对象，该对象可以完成事件消息的获取，字符串的打印等。

S.7.1.1　方法

S.7.1.1.1　getEventInfo

原型：string getEventInfo(id)

描述：获得消息的详细描述信息。
参数：id －number 型，表示消息描述字符串 ID。
返回：string 型，表示消息的详细描述信息。

S.7.1.1.2 println

原型：println(str)
描述：打印字符串，参数支持字符串的加运算，打印会自动换行。该方法将打印信息输出至串口，方便开发者查阅调试信息。
参数：str － string 型，表示待打印的字符串。
返回：无。

S.7.2 GlobalVarManager对象

该对象为内置对象，用于全局变量管理。

S.7.2.1 方法

S.7.2.1.1 setItemValue

原型：setItemValue(key, value)
描述：设置全局变量参数。
参数：key －string 型，表示全局变量关键字，由应用程序自行设置；
value －number 型，表示全局变量值。
返回：无。

S.7.2.1.2 getItemValue

原型：number getItemValue(key)
描述：获取全局变量参数。
参数：key －string型，表示全局变量关键字，应与GlobalVarManager.setItemValue()方法中的key对应。
返回：number型，返回由关键字key指定的全局变量值；若关键字key指定的全局变量不存在，则返回null。

S.7.2.1.3 setItemStr

原型：setItemStr(key, str)
描述：设置全局变量参数。
参数：key －string 型，表示全局变量关键字，由应用程序自行设置；
str－string 型，表示全局变量值。
返回：无。

S.7.2.1.4 getItemStr

原型：string getItemStr(key)
描述：获取全局变量参数。
参数：key －string 型，表示全局变量关键字，应与 GlobalVarManager.setItemStr()方法中的 key 对应。

返回：string 型，返回由关键字 key 指定的全局变量值；若关键字 key 指定的全局变量不存在，则返回 null。

S.7.2.1.5 removeItem

原型：removeItem(key)
描述：删除一个全局变量参数。
参数：key -string 型，全局变量关键字；若关键字 key 指定的全局变量不存在，则无任何操作。
返回：无。

S.7.2.1.6 clearAll

原型：clearAll()
描述：删除所有全局变量参数。
参数：无。
返回：无。

S.7.3 Rectangle对象

Rectangle 对象为本地对象。

S.7.3.1 属性

Rectangle 对象的属性定义见表 S.13。

表 S.13 Rectangle 对象属性表

属性名称	类型	读写属性	说明
left	number	读/写	Rectangle 对象左上角 X 轴坐标，单位为像素。
top	number	读/写	Rectangle 对象左上角 Y 轴坐标，单位为像素。
width	number	读/写	Rectangle 对象宽度，单位为像素。
height	number	读/写	Rectangle 对象高度，单位为像素。

S.7.3.2 方法

S.7.3.2.1 Rectangle

原型：Rectangel(left, top, width, height)
描述：构造方法，根据指定的坐标和宽高创建一个新的矩形对象。
参数：left - number 类型，矩形左上角 X 轴坐标，单位为像素；
right - number 类型，矩形左上角 Y 轴坐标，单位为像素；
width - number 类型，矩形宽度，单位为像素；
height -number 类型，矩形高度，单位为像素。

S.7.4 SysTool对象

本对象为内置对象，提供了系统待机、休眠以及重启等操作的方法。系统应对应用程序的权限进行验证，只有特权应用才能调用本类提供的方法。待机——接收终端的 CPU 仍在工作，用以运行一些后台程序，例如推送下载、软件升级等，关闭所有音视频输出，从表象上看接收终端停止了工作。休眠——接收终端

的CPU停止工作，彻底切断CPU和主板电源。靠外部单片机或其他方式监控遥控器发出的激活命令，然后启动开关电源给 CPU 和主板供电，实现遥控开机。 这种方式能耗更低。应根据接收机的实际能力实现本对象的相关方法。例如若不支持休眠，sleep()方法的实现为空或同standBy()的方法。

S.7.4.1 方法

S.7.4.1.1 standBy

原型：standBy()
描述：控制接收机转入待机状态。CPU仍在运行。
参数：无。
返回：无。

S.7.4.1.2 sleep

原型：sleep()
描述：控制接收机进入休眠状态，CPU将断电停止工作。
参数：无。
返回：无。

S.7.4.1.3 reboot

原型：reboot()
描述：重启接收机。
参数：无。
返回：无。

S.7.4.1.4 wakeUp

原型：wakeUp()
描述：唤醒接收机。
参数：无。
返回：无。

S.7.4.1.5 getStandByState

原型：number getStandByState()
描述：获取待机状态。
参数：无。
返回：number 1-真待机/sleep；2-假待机/standBy；3-正常工作。

S.8 软硬件信息查询模块

本模块定义了与接收终端软硬件信息查询相关的JS对象：HardwareInfo、SoftwareInfo。

S.8.1 HardwareInfo对象

HardwareInfo为内置对象，该对象用以描述接收终端硬件参数信息。

S.8.1.1 属性

HardwareInfo 对象的属性定义见表 S.14。

表 S.14 HardwareInfo 对象属性表

属性名称	类型	读写属性	说明
Flash.size	string	只读	获取接收终端闪存的大小，单位为兆字节（MB）。
RAM.size	string	只读	获取接收终端内存的大小，单位为兆字节（MB）。
RAM.type	string	只读	获取接收终端内存的类型，可取值“SDRAM”、“DDR”、“DDR2”等。
SOC.model	string	只读	获取接收终端主芯片的型号。
SOC.frequency	string	只读	获取接收终端主芯片的工作频率，单位为兆赫（MHz）。
SOC.provider	string	只读	获取接收终端主芯片的提供商名称。
HW.version	string	只读	获取接收终端硬件版本号。
STB.TPtype	string	只读	获取接收终端的传输模式类型，可取值“DVB-C”、“DVB-S”、“DVB-T”“ABS-SS”、“DTMB”等组合。
STB.definition	string	只读	获取接收终端的清晰度类型，可取值“HD”、“SD”。
STB.provider	string	只读	获取接收终端的提供商名称。
STB.brand	string	只读	获取接收终端的品牌名称。
STB.model	string	只读	获取接收终端的型号。
STB.serialnumber	string	只读	获取接收终端的序列号。
STB.returnPath	string	只读	获取接收终端的回传通道属性。取值“modulator”表示 cableModem；取值“ethernet”表示以太网。

S.8.2 SoftwareInfo对象

SoftwareInfo 为内置对象，该对象用以描述接收终端软件参数信息。

S.8.2.1 属性

SoftwareInfo 对象的属性定义见表 S.15。

表 S.15 SoftwareInfo 属性表

属性名称	类型	读写属性	说明
OS.name	string	只读	获取操作系统软件的名称。
OS.version	string	只读	获取操作系统软件的版本号。
OS.provider	string	只读	获取操作系统软件的提供商名称。
middleware.name	string	只读	获取系统软件的名称。
middleware.version	string	只读	获取系统软件的版本号。
middleware.provider	string	只读	获取系统软件的提供商名称。
middleware.releaseDate	string	只读	获取系统软件的发布日期。
middleware.copyright	string	只读	获取系统软件的版权信息。
middleware.RAMSize	string	只读	获取系统软件所占用的内存空间，单位为千字节（KB）。
middleware.NVMSize	string	只读	获取系统软件所占用的 NVM 空间，单位为千字节（KB）。
middleware.platform_profile	string	只读	获取系统软件所支持的平台档次。

表 S.15（续）

属性名称	类型	读写属性	说明
loader.name	string	只读	获取软件更新模块（loader）的名称。
loader.version	string	只读	获取软件更新模块（loader）的版本号。
loader.provider	string	只读	获取软件更新模块（loader）的提供商。
loader.size	string	只读	获取软件更新模块（loader）的大小，单位为千字节（KB）。
CA.name	string	只读	获取 CA 软件的名称。
CA.version	string	只读	获取 CA 软件的版本号。
CA.provider	string	只读	获取 CA 软件的提供商。
Driver.vision	string	只读	接收终端驱动版本号。

附 录 T
（规范性附录）
JavaScript-消息管理单元

T.1 概述

本附录定义了与消息管理相关的功能模块。

T.2 消息管理模块

本模块定义了与消息处理相关的 JS 对象：event。
应用层捕获到的消息有以下两种来源：
——系统消息，例如锁频成功、搜台完毕、网络信号中断等；
——按键消息，例如遥控器、键盘、鼠标、前面板等按键触发消息；
应用层通过以下方式捕获消息：
——系统消息：应用通过 document.onsystemevent 捕获系统消息；
——按键消息：

- 键盘：应用通过 document.onkeydown、document.onkeyup 和 document.onkeypress 等方式捕获键盘消息，消息码与 PC 方式保持一致；
- 鼠标：应用通过 document.onmousedown、document.onmouseup、document.onmousemove 等方式捕获鼠标消息，消息码与 PC 方式保持一致；
- 遥控器：应用通过 document.onkeydown、document.onkeyup 捕获遥控器消息，消息码与 PC 方式保持兼容，扩展按键消息码定义见表 D.2；
- 前面板：应用通过 document.onkeydown、document.onkeyup 捕获前面板消息，消息码与遥控器方式保持一致。

T.2.1 event对象

event 对象为内置对象。

T.2.1.1 属性

event 对象的属性见表 T.1。

表 T.1 event 对象的属性

属性名称	类型	读写属性	说明
event.type	number	只读	表示发生的事件的类型，即当前 event 对象表示的事件的名称，它与注册的事件句柄同名，例如“onclick”；或者是事件句柄属性删除前缀“on”，例如“click”。同 W3C 定义。
event.source	number	只读	表示消息的来源。
event.which	number	只读	表示消息的代码值。

表 T.1（续）

属性名称	类型	读写属性	说明
event.modifiers	number	只读	表示消息的扩展属性。若该消息的扩展属性为空，则 modifiers 返回 0；若该消息的扩展属性为 number 型，则 modifiers 返回该数值；若该消息的扩展属性为字符串，则 modifiers 返回一个 ID 值，该值由系统内部生成，作为具体字符串内容的指针，应用可调用 Utility.getEventInfo(ID)方法取出字符串内容。

T.2.1.2 消息来源

消息来源（event.source）的定义见表 T.2。

表 T.2 event.source 定义

event.source	描述
1001	表示系统消息。
1002	表示遥控器按键消息。
1003	表示前面板按键消息

T.2.1.3 系统消息

系统消息见附录 L、附录 M、附录 N、附录 Q、附录 S、附录 U、附录 V 等附录中消息的定义。

T.2.1.4 按键消息

按键消息定义见 D.2.1.5。

附 录 U
（规范性附录）
JavaScript-应用引擎单元

U.1 概述

本附录定义了一些常用的应用引擎模块：频道管理、电子节目指南、预定提醒管理、信息搜索等。

U.2 频道管理模块

本模块定义了与频道管理相关的JS对象：ChannelManager、Channel。

U.2.1 消息

频道管理模块可能发给应用层的消息定义见表U.1。

表 U.1 频道管理模块消息

消息名称	event.which	event.modifiers	消息说明
MSG_CHANNEL_RAM_TO_NVM_SUCCESS	19001	-	表示从 RAM 写频道数据到 NVM 成功。
MSG_CHANNEL_RAM_TO_NVM_FAILED	19002	-	表示从 RAM 写频道数据到 NVM 失败。
MSG_CHANNEL_NVM_TO_RAM_SUCCESS	19003	-	表示从 NVM 恢复频道数据到 RAM 成功。
MSG_CHANNEL_NVM_TO_RAM_FAILED	19004	-	表示从 NVM 恢复频道数据到 RAM 失败。
保留	19005～19500		
注：event.modifiers 值由系统内部自动给出，其数据类型： ——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。 ——“-”，表示 event.modifiers 为 undefined。			

U.2.2 ChannelManager对象

本对象为内置对象，描述用户对频道的管理方式。

U.2.2.1 方法

U.2.2.1.1 getChannelByChannelID

原型：Channel getChannelByChannelID(channelId)
描述：根据频道 ID 来获取频道对象。
参数：channelId - number 型，表示频道 ID，必须是大于 0 的十进制整数。
返回：Channel 对象。

U.2.2.1.2 getChannelByLogicalID

原型：Channel getChannelByLogicalID(logicalId)
描述：根据逻辑频道号来获取频道对象。
参数：logicalId - number 型，表示逻辑频道号，必须是大于 0 的十进制整数。
返回：Channel 对象。

U.2.2.1.3 getChannelByServiceID

原型：Channel getChannelByServiceID(serviceId)
描述：根据业务 ID 来获取频道对象。
参数：serviceId - number 型，表示频道所对应的业务 ID，取值范围 0～65535。
返回：Channel 对象。

U.2.2.1.4 getLastChannel

原型：Channel getLastChannel()
描述：获取前一个打开的频道。
参数：无。
返回：Channel 对象。

U.2.2.1.5 getLastChannel

原型：Channel getLastChannel(serviceType)
描述：获取指定类型的前一个打开的频道。
参数：serviceType - number 型， 表示业务类型。
返回：Channel 对象。

U.2.2.1.6 getShutDownChannel

原型：Channel getShutDownChannel()
描述：获取关机频道。
参数：无。
返回：Channel 对象。
注：用户在打开某一频道，或切换频道后系统会立即将刚才打开的频道设置为关机频道。任何情况下都可以获取到关机频道，若系统曾设置关机频道，则获取的是设置的值；否则由系统自行决定关机频道。

U.2.2.1.7 getShutDownChannel

原型：Channel getShutDownChannel(serviceType)
描述：获取指定类型的关机频道。
参数：seviceType - number 型，表示业务类型。
返回：Channel 对象。
注：用户在打开某一频道，或切换频道后系统会立即将刚才打开的频道设置为关机频道。任何情况下都可以获取到关机频道，若系统曾设置关机频道，则获取的是设置的值；否则在serviceType指定的业务类型中，由系统自行决定关机频道。

U.2.2.1.8 delChannel

原型：number delChannel(obj)
描述：从频道列表中删除指定频道。

注：针对 RAM 中的频道列表操作。

参数：obj - Channel 对象。

返回：number 型，1 表示删除成功，0 表示删除失败。

U.2.2.1.9 deleteAll

原型：number deleteAll()

描述：删除频道列表中所有的频道。

注：针对 RAM 中的频道列表操作。

参数：无。

返回：number 型，1 表示成功，0 表示失败。

U.2.2.1.10 deleteAllDelMarked

原型：number deleteAllDelMarked()

描述：删除频道列表中所有带有删除标记的频道对象。

注：针对 RAM 中的频道列表操作。

返回：number 型，1 表示成功，0 表示失败。

U.2.2.1.11 deleteAllFavorites

原型：number deleteAllFavorites()

描述：删除所有喜爱频道。

注：针对 RAM 中的频道列表操作。

参数：无。

返回：number 型，1 表示成功，0 表示失败。

U.2.2.1.12 resetProperties

原型：resetProperties()

描述：将用户所有设置为喜爱、锁定、隐藏等标记的频道全部重置，所有的频道全部更改为非喜爱、非锁定、非隐藏。

注：针对 RAM 中的频道列表操作。

参数：无。

返回：number 型，1 表示成功，0 表示失败。

U.2.2.1.13 swap

原型：swap(obj1,obj2)

描述：交换频道对象 obj1 和频道对象 obj2 在频道列表中的位置。

注：针对 RAM 中的频道列表操作。

参数：obj1 - 表示 Channel 对象；

obj2 - 表示 Channel 对象。

返回：无。

U.2.2.1.14 sort

原型：sort(sortTypeArray[], sortOrderArray[])

描述：按指定方式进行频道排序。

注：针对RAM中的频道列表进行操作。

参数：sortTypeArray – number型数组，表示排序依据，可以有一个或多个排序依据，但不能重复。排序依据的优先级和数组成员的顺序相关，数组成员的下标越小，优先级越高。
sortOrderArray - number型数组，表示排序方式。

返回：无。

注：隐藏和加锁的频道也一样要被排序，如果两个或更多的频道排序依据完全相同，则这些频道之间维持此前的排列顺序。

示例：

```
//将频道按照service_id升序排序，其中service_id相同的频道按照本次排序之前的顺序排序
sort([DVB_SORT_TYPE_SERVICE_ID], [DVB_SORT_ORDER_ASC]);
//将频道先按照是否加密排序,非加密频道在前,加密频道在后；同是非加密或者同是加密的频道再按照
//service_name升序排序，service_name也相同的频道按照本次排序前的顺序排序
sort([DVB_SORT_TYPE_FTA_SCR, DVB_SORT_TYPE_SERVICE_NAME],
     [DVB_SORT_ORDER_ASC,DVB_SORT_ORDER_DESC]);
```

U.2.2.1.15 filter

原型：Channel[] filter(filterTypeArray[], valueArray[])

描述：在当前频道列表中过滤出指定条件的新的频道列表。

参数：filterTypeArray - number 型数组，表示过滤依据，数组的每个成员取值见表 U.2，可以有一个或多个排序依据，成员不可以重复，过滤依据没有优先级。
valueArray - number 型数组，长度与 filterTypeArray 一致，数组成员应和 filterTypeArray 数组成员一一对应，分别表示每个 filterTypeArray 成员的取值。

表 U.2 filter 过滤条件

filterTypeArray 取值	说明	对应 valueArray 取值
const FILTER_TYPE_SERVICETYPE = 1000;	过滤出指定业务类型的频道。	取值见 “业务类型常量”定义。
const FILTER_TYPE_FAV = 1001;	过滤出指定喜爱级别的频道。	取值： ——0 表示过滤非喜爱的频道； ——1 表示过滤喜爱的频道。
const FILTER_TYPE_BAT = 1002;	过滤出指定业务群下的频道。	表示 bouquet_id。
const FILTER_TYPE_FTA = 1003;	过滤出加密或者非加密频道。	取值： ——0 表示过滤出非加密频道； ——1 表示过滤出加密频道。
const FILTER_TYPE_HIDE = 1004;	过滤出隐藏或者非隐藏的频道。	取值： ——0 表示只过滤出非隐藏频道； ——1 表示只过滤出隐藏频道。

返回：Channel 对象数组。

示例：

```
//表示过滤出所有电视广播频道中的喜爱频道
filter([FILTER_TYPE_SERVICETYPE, FILTER_TYPE_FAV], [DVB_SERVICE_TYPE_DTV, 1]);
```

```
//表示过滤出 bouquet_id=10 的业务群里的所有加密频道
filter([FILTER_TYPE_BAT, FILTER_TYPE_FTA],[10,1]);
```

U.2.2.1.16 save

原型：save()
描述：异步方法，将 RAM 中的频道列表数据保存到 NVM 中。
——若保存成功则发送消息 MSG_CHANNEL_RAM_TO_NVM_SUCCESS；
——若保存失败则发送消息 MSG_CHANNEL_RAM_TO_NVM_FAILED。
参数：无。
返回：无。

U.2.2.1.17 restore

原型：restore()
描述：异步方法，将 NVM 中的频道列表数据恢复到 RAM 中。
——若恢复成功则发送消息 MSG_CHANNEL_NVM_TO_RAM_SUCCESS；
——若恢复失败则发送消息 MSG_CHANNEL_NVM_TO_RAM_FAILED。
参数：无。
返回：无。

U.2.3 Channel对象

Channel 对象为本地对象，用以描述与用户和运营商行为相关的属性和方法。

U.2.3.1 属性

Channel对象的属性定义见表U.3。

表 U.3 Channel 对象属性表

属性名称	类型	属性	说明
channelId	number	读/写	表示接收终端分配的频道编号。 **注**：接收终端分配的频道编号是指终端系统软件自行为频道分配的编号，分配策略由终端软件自行决定。channelId 生成之后，在 Channel 对象的生命周期内都保持不变，对频道排序和交换操作可能会修改 channelId 的值。
logicalId	number	只读	表示频道的逻辑频道号。 若该频道没有逻辑频道号，该属性返回数值-1。逻辑频道号的获取与运营商有关，由接收终端自行实现。
isDeleted	number	读/写	表示频道的删除标记。取值： ——0 - 表示没有设置删除标记； ——1 - 表示设置了删除标记。
isFavorite	number	读/写	表示频道在喜爱类中的喜爱级别，取值： ——0 - 表示不喜爱； ——1 - 表示喜爱。

表 U.3（续）

属性名称	类型	属性	说明
isLocked	number	读/写	表示频道的锁定状态，取值： ——0 - 表示该频道未被锁定； ——1 - 表示该频道被锁定。 当某频道被锁定后，用户需要先输入密码，才能观看该频道的节目。
isHided	number	读/写	表示频道的隐藏状态，取值： —— 0 表示不隐藏； —— 1 表示隐藏。 一旦某频道被“隐藏”，它将不会出现在频道列表中，也不能通过遥控器上/下键切换观看，只有进入“系统设置”取消其隐藏状态后，才能被观看。
deltVolume	number	读/写	表示相对于全局音量的增值（+/-），即： 频道实际音频 = 全局音量 + deltVolume 备注：修改立即生效。
supportPlayback	boolean	只读	表示该频道是否支持时移回看业务，取值： ——true - 表示支持； ——false - 表示不支持。

U.2.3.2 方法

U.2.3.2.1 getService

原型：DvbService getService()
描述：获取当前频道对象对应的 DvbService 对象。
参数：无。
返回：DvbService 对象。

U.3 电子节目指南模块

本模块定义了与电子节目指南相关的 JS 对象：EPGManager、ProgramEvent、ReferenceEvent 和 TimeShiftEvent。

U.3.1 消息

电子节目指南模块可能发给应用层的消息定义见表U.4。

表 U.4　电子节目指南模块消息

消息名称	event.which	event.modifiers	消息说明
MSG_EPG_SEARCH_SUCCESS	18001	number	成功完成 EPG 搜索。
MSG_EPG_SEARCH_EXCEED_MAX_COUNT	18002	number	搜索结果达到最大值，搜索自动停止。 ProgramEvent 最大个数由数据管理模块中 EPGSetting.program_event_maxcount 指定。
MSG_EPG_SEARCH_REFRESH	18003	number	EPG 数据更新。 当 EPG 搜索到部分数据时，发送该消息。
MSG_EPG_SEARCH_TIMEOUT	18004	number	搜索 EPG 超时。在接口指定的时间段内没有搜索到任何节目信息时，发送该消息。
MSG_EPG_RECEIVE_NVODREFERENCE_SUCCESS	18005	number	接收 NVOD 参考事件成功。
MSG_EPG_RECEIVE_ALL_NVODREFERENCE_SUCCESS	18006	number	全部的 NVOD 频点的数据接收完毕。
MSG_EPG_RECEIVE_NVODREFERENCE_TIMEOUT	18007	number	NVOD 参考事件搜索超时。
MSG_EPG_RECEIVE_NVODTIMESHIFT_SUCCESS	18008	number	接收某参考事件下的时移事件成功。
MSG_EPG_RECEIVE_NVODTIMESHIFT_TIMEOUT	18009	number	NVOD 时移事件接收超时。
保留	18010～18500		

注：event.modifiers 值由系统内部自动给出，其数据类型：

——“number”，表示搜索会话 maskId。

——“-”，表示event.modifiers为undefined。

U.3.2　EPGManager对象

EPGManager 对象为内置对象。该对象提供了获取指定分类下的事件数组方法。

U.3.2.1　常量

节目类型常量定义见表 U.5。

表 U.5　节目类型常量

常量	说明
电影/戏剧	
const CONTENT_TYPE_MOVIE=0x10;	影视/戏剧（一般）
const CONTENT_TYPE_MOVIE_DETECTIVE=0x11;	侦探/惊悚
const CONTENT_TYPE_MOVIE_ADVENTURE=0x12;	探险/西部/战争
const CONTENT_TYPE_MOVIE_FANTASY=0x13;	科幻/魔幻/恐怖

表 U.5（续）

常量	说明
const CONTENT_TYPE_MOVIE_COMEDY=0x14;	喜剧
const CONTENT_TYPE_MOVIE_SOAP=0x15;	连续剧/情景剧/民俗
const CONTENT_TYPE_MOVIE_ROMANCE=0x16;	浪漫
const CONTENT_TYPE_MOVIE_CLASSIC=0x17;	严肃/经典/宗教/历史电影/戏剧
const CONTENT_TYPE_MOVIE_ADULT=0x18;	成人电影/戏剧
新闻/实时	
const CONTENT_TYPE_NEWS=0x20;	新闻/时事（一般）
const CONTENT_TYPE_NEWS_WEATHER=0x21;	新闻/天气
const CONTENT_TYPE_NEWS_MAGAZINE=0x22;	新闻杂志
const CONTENT_TYPE_NEWS_DOCUMENTARY=0x23;	文献电视片
const CONTENT_TYPE_NEWS_DISCUSSION=0x24;	讨论/访谈/辩论
演出/游艺	
const CONTENT_TYPE_SHOW=0x30;	演出/游艺（一般）
const CONTENT_TYPE_SHOW_GAME=0x31;	游艺/竞猜/竞技
const CONTENT_TYPE_SHOW_VARIETY=0x32;	各类演出
const CONTENT_TYPE_SHOW_TALK=0x33;	脱口秀
体育	
const CONTENT_TYPE_SPORTS=0x40;	体育（一般）
const CONTENT_TYPE_SPORTS_SPECIAL=0x41;	特定赛事（奥运会、世界杯等）
const CONTENT_TYPE_SPORTS_MAGAZINE=0x42;	体育杂志
const CONTENT_TYPE_SPORTS_SOCCER=0x43;	足球/橄榄球
const CONTENT_TYPE_SPORTS_TENNIS=0x44;	网球/壁球
const CONTENT_TYPE_SPORTS_TEAM=0x45;	团体赛事（包含足球）
const CONTENT_TYPE_SPORTS_ATHLETIC=0x46;	田径
const CONTENT_TYPE_SPORTS_MOTOR=0x47;	赛车
const CONTENT_TYPE_SPORTS_WATER=0x48;	水上运动
const CONTENT_TYPE_SPORTS_WINTER=0x49;	冬季运动
const CONTENT_TYPE_SPORTS_EQUESTRAIN=0x4A;	赛马
const CONTENT_TYPE_SPORTS_MARTIAL=0x4B;	武术
儿童/青少节目	
const CONTENT_TYPE_CHILDREN=0x50;	儿童/青少节目（一般）
const CONTENT_TYPE_CHILDREN_PRESCHOOL=0x51;	学龄前儿童节目
const CONTENT_TYPE_CHILDREN_6TO14=0x52;	适于 6~14 岁的娱乐节目
const CONTENT_TYPE_CHILDREN_10TO16=0x53;	适于 10~16 岁的娱乐节目
const CONTENT_TYPE_CHILDREN_SCHOOL=0x54;	信息/教育/教学节目
const CONTENT_TYPE_CHILDREN_CARTOON=0x55;	卡通/木偶
音乐/芭蕾/舞蹈	
const CONTENT_TYPE_MUSIC=0x60;	音乐/芭蕾/舞蹈（一般）
const CONTENT_TYPE_MUSIC_POP=0x61;	摇滚/流行

表 U.5（续）

常量	说明
const CONTENT_TYPE_MUSIC_CLASSICAL=0x62;	严肃音乐/古典音乐
const CONTENT_TYPE_MUSIC_FOLK=0x63;	民谣/传统音乐
const CONTENT_TYPE_MUSIC_JAZZ=0x64;	爵士
const CONTENT_TYPE_MUSIC_OPERA=0x65;	音乐剧/歌剧
const CONTENT_TYPE_MUSIC_BALLET=0x66;	芭蕾
艺术/文化（不含音乐）	
const CONTENT_TYPE_ARTS=0x70;	艺术/文化（不含音乐，一般）
const CONTENT_TYPE_ARTS_PERFORMANCE=0x71;	表演艺术
const CONTENT_TYPE_ARTS_ARTS=0x72;	美术
const CONTENT_TYPE_ARTS_RELIGION=0x73;	宗教
const CONTENT_TYPE_ARTS_CULTURE=0x74;	流行文化/传统艺术
const CONTENT_TYPE_ARTS_LITERATURE=0x75;	文学
const CONTENT_TYPE_ARTS_FILM=0x76;	胶片/电影
const CONTENT_TYPE_ARTS_VIDEO=0x77;	实验电影/视频
const CONTENT_TYPE_ARTS_PRESS=0x78;	广播/新闻
const CONTENT_TYPE_ARTS_NEW=0x79;	新媒体
const CONTENT_TYPE_ARTS_MAGAZINE=0x7A;	艺术/文化杂志
const CONTENT_TYPE_ARTS_FASHION=0x7B;	时尚
社会/政治热点/经济	
const CONTENT_TYPE_SOCIAL=0x80;	社会/政治热点/经济（一般）
const CONTENT_TYPE_SOCIAL_MAGAZINE=0x81;	杂志/报道/纪实
const CONTENT_TYPE_SOCIAL_ECONOMICS=0x82;	经济/社会咨询
const CONTENT_TYPE_SOCIAL_PEOPLE=0x83;	特别人物
科学/教育/事实话题	
const CONTENT_TYPE_EDUCATION=0x90;	科学/教育/事实话题（一般）
const CONTENT_TYPE_EDUCATION_NATUAL=0x91;	自然/动物/环境
const CONTENT_TYPE_EDUCATION_TECNOLOGY=0x92;	科技/自然科学
const CONTENT_TYPE_EDUCATION_MEDICAL=0x93;	医药/生理/心理
const CONTENT_TYPE_EDUCATION_FOREIGN=0x94;	国外/探险
const CONTENT_TYPE_EDUCATION_SOCIAL=0x95;	社会/精神科学
const CONTENT_TYPE_EDUCATION_EDUCATION=0x96;	再教育
const CONTENT_TYPE_EDUCATION_LANGUAGE=0x97;	语言
休闲娱乐	
const CONTENT_TYPE_LEISURE=0xA0;	休闲娱乐（一般）
const CONTENT_TYPE_LEISURE_TRAVLE=0xA1;	观光/旅行
const CONTENT_TYPE_LEISURE_HANDICRAFT=0xA2;	手工艺
const CONTENT_TYPE_LEISURE_MOTOR=0xA3;	自驾
const CONTENT_TYPE_LEISURE_HEALTH=0xA4;	健身/健康
const CONTENT_TYPE_LEISURE_COOKING=0xA5;	烹饪

表 U.5（续）

常量	说明
const CONTENT_TYPE_LEISURE_SHOPPING=0xA6;	广告/购物
const CONTENT_TYPE_LEISURE_GARDENING=0xA7;	园艺
描述特殊特征	
const CONTENT_TYPE_DESCRIPTION_LANGUAGE=0xB0;	原始语言
const CONTENT_TYPE_DESCRIPTION_COLOR=0xB1;	黑/白
const CONTENT_TYPE_DESCRIPTION_UNPUBLISHED=0xB2;	未发行
const CONTENT_TYPE_DESCRIPTION_LIVE=0xB3;	现场直播
注：常量值高4位为一级节目内容分类，低4位为二级节目内容分类，如0xB1，其中0xB表示一级节目内容分类，0x1表示二级节目内容分类。	

U.3.2.2 方法

U.3.2.2.1 searchProgramEvent

原型：number searchProgramEvent(tsList, mask, endureTime)

描述：异步方法,在由全局参数EPGSetting.search_start_date和EPGSetting.search_days限定的日期范围内，按参数tsList指定的频点列表搜索由参数mask指定的节目事件信息。

——搜索完成后发出消息MSG_EPG_SEARCH_SUCCESS;

——搜索过程中接收到部分数据，可发出消息MSG_EPG_SEARCH_REFRESH，通知页面已经搜到部分数据，通过EPGManager.getProgramsByService()方法获取相应的数据;

——当搜索时间达到参数endureTime指定的超时值时，系统自动停止搜索，并发出消息MSG_EPG_SEARCH_TIMEOUT。

消息对象event.modifiers属性应携带此次搜索过程标识(maskId)。

参数：tsList – DvbTS对象数组，指定搜索节目信息的频点列表。

mask – number类型，表示搜索掩码，基本掩码有：

——0x01 – 表示搜索actual PF;

——0x02 – 表示搜索actual schedule;

——0x04 – 表示搜索other PF;

——0x08 – 表示搜索other schedule。

mask值可由一个或多个基本掩码相或组成，如0x03=(0x01|0x02)，表示搜索actual PF和actual schedule的数据。

endureTime – number型，表示搜索EPG信息的超时时间，单位为秒。

返回：number型，表示系统为该次异步过程分配的唯一标识(maskId)。

U.3.2.2.2 searchProgramEventByService

原型：number searchProgramEventByService(serviceLocator, mask, endureTime)

描述：异步方法，在由全局参数EPGSetting.search_start_date和EPGSetting.search_days限定的日期范围内，按参数serviceLocator指定的业务搜索由参数mask指定的节目事件信息。

——搜索完成后发出消息MSG_EPG_SEARCH_FINISHED;

——搜索过程中接收到部分数据，可发出消息MSG_EPG_SEARCH_REFRESH，通知页面已经搜到部分数据，通过EPGManager.getProgramsByService()等方法获取相应的数据;

——当搜索时间达到参数endureTime指定的超时值时，系统自动停止搜索，并发出消息MSG_EPG_SEARCH_TIMEOUT。

消息对象event.modifiers属性应携带此次搜索过程标识(maskId)。

参数：serviceLocator – string型，表示广播业务定位符。

mask – number类型，表示搜索掩码，基本掩码有：

——0x01 – 表示搜索actual PF；

——0x02 – 表示搜索actual schedule；

——0x04 – 表示搜索other PF；

——0x08 – 表示搜索other schedule。

mask值可由一个或多个基本掩码相或组成，如0x03=(0x01|0x02)，表示搜索actual PF和actual schedule的数据。

endureTime – number型，表示搜索EPG信息的超时时间，单位为秒。

返回：number型，表示系统为该次异步过程分配的唯一标识(maskId)。

U.3.2.2.3 searchNVODRefEvents

原型：number searchNVODRefEvents(endureTime)

描述：异步方法，通知系统开始接收NVOD参考事件数据。

——若搜索某个频点上的NVOD数据成功，则向页面发送消息MSG_EPG_RECEIVE_NVODREFERENCE_SUCCESS，通过getRefereneEvents()方法可以获取到收过的所有数据；

——若全部的NVOD频点的数据接收完毕，则向页面发送消息MSG_EPG_RECEIVE_ALL_NVODREFERENCE_SUCCESS,并将搜索结果保存在内存中，通过getReferenceEvents()可以获取到所有数据；

——若搜索时间达到endureTime时，则向页面发送消息MSG_EPG_RECEIVE_NVODREFERENCE_TIMEOUT，此时也可调用getReferenceEvents()方法获取已搜到的信息。

使用exitNVODMode()方法退出对参考事件的接收。

参数：endureTime：number型，表示搜索NVOD信息的超时时间，单位为秒。

返回：number型，表示系统为该次异步过程分配的唯一标识(maskId)。

U.3.2.2.4 searchNVODRefEvents

原型：number searchNVODRefEvents(tsArray, endureTime)

描述：异步方法，通知系统开始接收指定的TS频点的NVOD参考事件数据,

——若搜索某个频点上的NVOD数据成功，则向页面发送消息MSG_EPG_RECEIVE_NVODREFERENCE_SUCCESS，通过getRefereneEvents()方法可以获取到收过的所有数据；

——若全部的NVOD频点的数据接收完毕，则向页面发送消息MSG_EPG_RECEIVE_ALL_NVODREFERENCE_SUCCESS,并将搜索结果保存在内存中，通过getReferenceEvents()可以获取到所有数据；

——若搜索时间达到endureTime时，则向页面发送消息MSG_EPG_RECEIVE_NVODREFERENCE_TIMEOUT,此时也可调用getReferenceEvents()方法获取已搜到的信息。

使用exitNVODMode()方法退出对参考事件的接收。

参数：tsArray - DvbTS 对象数组，搜索指定的 TS 频点的 NVOD 数据。
endureTime - number 型，表示搜索 NVOD 信息的超时时间，单位为秒。
返回：number 型，表示系统为该次异步过程分配的唯一标识(maskId)。

U.3.2.2.5 getPresentProgram

原型：ProgramEvent getPresentProgram(serviceLocator)
描述：获取指定业务的当前节目。
参数：serviceLocator -string 型，表示广播业务定位符。
返回：ProgramEvent 对象。

U.3.2.2.6 getPresentProgramsByContentType

原型：ProgramEvent[] getPresentProgramsByContentType(contentType)
描述：根据参数中指定的节目内容分类值，在当前 EPG 数据库中查找符合条件的当前节目信息。
参数：contentType -number 型，表示节目内容分类类型。
返回：ProgramEvent 对象数组。

U.3.2.2.7 getPresentProgramsByName

原型：ProgramEvent[] getPresentProgramsByName(str)
描述：根据参数中指定的节目名称，在当前 EPG 数据库中查找符合条件的当前节目信息。
参数：str -string 型，表示搜索关键字。
返回：ProgramEvent 对象数组。

U.3.2.2.8 getFollowingProgram

原型：ProgramEvent getFollowingProgram(serviceLocator)
描述：获取指定业务的后续节目。
参数：serviceLocator -string 型，表示广播业务定位符。
返回：ProgramEvent 对象。

U.3.2.2.9 getFollowingProgramsByContentType

原型：ProgramEvent[] getFollowingProgramsByContentType(contentType)
描述：同步方法，根据参数中指定的节目内容分类值，在当前 EPG 数据库中查找符合条件的后续节目信息。
参数：contentType -number 型，表示节目内容分类类型。
返回：ProgramEvent 对象数组。

U.3.2.2.10 getFollowingProgramsByName

原型：ProgramEvent[] getFollowingProgramsByName(str)
描述：同步方法，根据参数中指定的节目名称，在当前 EPG 数据库中查找符合条件的当前节目信息。
参数：str -string 型，表示搜索关键字。
返回：ProgramEvent 对象数组。

U.3.2.2.11 getProgramsByService

原型：ProgramEvent[] getProgramsByService(serviceLocator)

描述：获取指定业务的所有节目信息。
参数：serviceLocator - string 型，表示广播业务定位符。
返回：ProgramEvent 对象数组。

U.3.2.2.12 getProgramsByDate

原型：ProgramEvent[] getProgramsByDate(serviceLocator, beginDate, endDate)
描述：同步方法，根据参数中指定的起始日期和结束日期，获取指定业务中符合条件的节目信息。
参数：serviceLocator - string 型，表示广播业务定位符。
beginDate - Date 类型对象，表示起始日期。
endDate - Date 类型对象，表示结束日期。
返回：ProgramEvent 对象数组。

U.3.2.2.13 getProgramsByDirection

原型：ProgramEvent[] getProgramsByDirection(serviceLocator, beginDate, count, isForward)
描述：同步方法，根据参数中指定的起始日期和检索方向，获取指定业务中指定个数的节目信息。
参数：serviceLocator - string 型，表示广播业务定位符。
beginDate - Date 类型对象，起始日期。
count - number 型，表示待获取的节目信息个数。
isForword - number 型，0 表示向后搜索；1 表示向前搜索。
返回：ProgramEvent 对象数组。

U.3.2.2.14 getProgramsByContentType

原型：ProgramEvent[] getProgramsByContentType(contentType)
描述：同步方法，根据参数中指定的节目内容分类值，在当前 EPG 数据库中查找符合条件的节目信息。
参数：contentType -number 型，节目内容分类类型。
返回：ProgramEvent 对象数组。

U.3.2.2.15 getProgramsByName

原型：ProgramEvent[] getProgramsByName(str)
描述：同步方法，根据参数中指定的节目名称，在当前 EPG 数据库中查找符合条件的节目信息。
参数：str -string 型，表示搜索关键字。
返回：ProgramEvent 对象数组。

U.3.2.2.16 getReferencePrograms

原型：ReferenceEvent[] getReferencePrograms(serviceLocator)
描述：同步方法，获取指定参考业务上的参考节目。
参数：serviceLocator - string 型，表示参考业务定位符。
返回：ReferenceEvent 对象数组。

U.3.2.2.17 getReferenceEvents

原型：ReferenceEvent[] getReferenceEvents(sortType, sortOrder)
描述：获取到搜索的ReferenceEvent对象数组。
参数：sortType – number型，表示排序依据，取值：

——1 - 代表根据参考事件ID排序；
——2 - 代表根据参考事件名称排序。
sortOrder – number型，表示排序方式，取值：
——0 - 表示降序排序；
——1 - 表示升序排序。
返回：ReferenceEvent对象数组。

U.3.2.2.18 exitNVODMode

原型：exitNVODMode()
描述：退出 NVOD 数据的接收。
参数：无。
返回：无。

U.3.3 ProgramEvent对象

ProgramEvent 对象为本地对象，用以保存与用户行为相关的节目事件信息。

U.3.3.1 属性

ProgramEvent 对象的属性定义见表 U.6。

表 U.6 ProgramEvent 属性表

属性名称	类型	属性	说明
channelObj	Channel 对象	只读	节目事件所属的 Channel 对象。
eventObj	DvbEvent 对象	只读	节目事件所对应的 DvbEvent 对象。
isBooked	number	读/写	节目是否被预定标记，取值： —— 0 - 表示未被预定； —— 1 - 表示被预定。

U.3.4 ReferenceEvent对象

ReferenceEvent 为本地对象，用以保存与用户行为相关的参考事件信息。

U.3.4.1 属性

ReferenceEvent 对象的属性定义见表 U.7。

表 U.7 ReferenceEvent 属性表

属性名称	类型	属性	说明
channelObj	Channel 对象	只读	参考事件所属的参考业务 Channel 对象。
eventObj	DvbEvent 对象	只读	参考事件所对应的 DvbEvent 对象。

U.3.4.2 方法

U.3.4.2.1 searchSchedules

原型：number searchSchedules(endureTime)

描述：异步方法，搜索参考事件对应的所有的时移事件信息。

参数：endureTime – number型，允许的接收超时时间，单位为秒。

返回：number型，

——1 - 表示cache中无数据，底层会自动去锁定频点，如果接收数据成功，则发出消息MSG_EPG_RECEIVE_NVODTIMESHIFT_SUCCESS；当搜索时间达到endureTime时，系统自动停止搜索，并发出消息MSG_EPG_RECEIVE_NVODTIMESHIFT_TIMEOUT，此时也可调用getSchedules()函数获取已搜到的信息。

——2 - 表示cache中已经存在想要收取的NVOD时移事件信息并且是完整的，可以直接获取，这时不会再有消息返回给页面，页面在判断返回值为2时就可以直接获取时移事件信息了。

U.3.4.2.2 getSchedules

原型：TimeShiftEvent[] getSchedules()

描述：获取该参考事件从此时此刻起，若干天之内的所有时移事件列表，当天已经播放完毕的时移事件将被丢弃，时移事件按照起始时间进行排序。

参数：无。

返回：TimeShiftEvent 对象数组。

U.3.4.2.3 getPresentSchedules

原型：TimeShiftEvent[] getPresentSchedules()

描述：获取该参考事件所带的当前正播放的所有时移事件对象，数组中的元素按照播放起始时间进行排序。

参数：无。

返回：TimeShiftEvent 对象数组。

U.3.4.2.4 getFollowingSchedules

原型：TimeShiftEvent[] getFollowingSchedules()

描述：获取该参考事件所带的下一个播放的所有时移事件对象，数组中的元素按照播放起始时间进行排序。

参数：无。

返回：TimeShiftEvent 对象数组。

U.3.5 TimeShiftEvent对象

TimeShiftEvent 对象为本地对象，用以保存时移事件信息。

U.3.5.1 属性

TimeShiftEvent对象的属性见表U.8。

表 U.8 TimeShiftEvent 对象属性表

属性名称	类型	属性	说明
channelObj	Channel 对象	只读	时移事件所属的 Channel 对象。
refChannelObj	Channel 对象	只读	时移事件所对应的参考业务 Channel 对象。
eventObj	DvbEvent 对象	只读	时移事件所对应的 DvbEvent 对象。
refEventObj	DvbEvent 对象	只读	时移事件所对应的参考 DvbEvent 对象。
status	number	只读	获取时移事件状态，取值： —— -1 - 表示已经播放完毕； —— 0 - 表示正在播； —— 1 - 表示还没播。 该属性为实时值。
orderIndex	number	只读	表示该时移事件在预订列表中的位置，取值： —— -1 - 表示未被预订； —— ≥0 - 表示该时移事件在预定列表中的位置。
preEvent	TimeShiftEvent 对象	只读	获取该时移事件的前一个时移事件，若返回 null，则表示该对象为第一个时移事件。
nextEvent	TimeShiftEvent 对象	只读	获取该时移事件的后一个时移事件，若返回 null，则表示该对象为最后一个时移事件。

U.4 预定提醒模块

U.4.1 消息

预定提醒模块可能发给应用层的消息定义见表U.9。

表 U.9 预定提醒模块消息

消息名称	event.which	event.modifiers	消息说明
MSG_REMIND_RAM_TO_NVM_SUCCESS	20001	-	表示从 RAM 写频道数据到 NVM 成功。
MSG_REMIND_RAM_TO_NVM_FAILED	20002	-	表示从 RAM 写频道数据到 NVM 失败。
MSG_REMIND_NVM_TO_RAM_SUCCESS	20003	-	表示从 NVM 恢复频道数据到 RAM 成功。
MSG_REMIND_NVM_TO_RAM_FAILED	20004	-	表示从 NVM 恢复频道数据到 RAM 失败。
保留	20005～20500		
注：event.modifiers 值由系统内部自动给出，其数据类型： ——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。 ——“-”，表示 event.modifiers 为 undefined。			

U.4.2 OrderManager对象

OrderManager 对象为内置对象，用以描述对用户预定的管理方式。

U.4.2.1 属性

OrderManager对象的属性定义见表U.10。

表U.10 OrderManager对象属性

属性名称	类型	属性	说明
advanceRemind	number	读/写	此接口用于设置已经预定的节目弹出提示的时间，也就是已经预定的节目将会在多长时间后进行播放,便于页面提醒用户多长时间后将要开始播放的已经预定的节目 默认值为60，单位为秒
conflictInterval	number	读/写	此接口用于设置预订节目的冲突门限值。用户要预订节目A，则在节目A开始时间正负各设置的分钟的范围内，不能有之前预订的其他节目开始播放，否则系统将不会添加节目预订，且在内存中生成预订冲突节目列表（哪些已经预订的节目和本次预订在开始时间上有冲突），供getConflictEvents()方法调用默认值为5，单位为分钟。
remindType	number	读/写	0 - 表示以设置的conflictInterval值为冲突判断条件; 1 - 表示以节目播放整个时长为冲突判断条件;

U.4.2.2 方法

U.4.2.2.1 getPlayingList

原型：Order[] getPlayingList()
描述：获取预定列表中当前正在播放的所有Order对象。
参数：无。
返回：Order对象数组。

U.4.2.2.2 getRemindList

原型：Order[] getRemindList()
描述：获取预定列表中将要播放的所有Remind对象。
参数：无。
返回：Order对象数组。

U.4.2.2.3 getOrders

原型：Order[] getOrders(type)
描述：按指定类型获取所有预定节目。
参数：type – number型，表示节目预定类型，取值：
——0或无参数表示获取所有类型的节目预定;
——1表示获取视频类型的节目预定;
——2表示获取音频类型的节目预定;
——3表示获取NVOD类型的节目预定。
返回：Order对象数组。

U.4.2.2.4 getOrderByID

原型：Order getOrderByID(orderID)

描述：根据预订的标识符获取预订对象。
参数：orderID - number 型，表示预订的标识符，取值范围 0～65535。
返回：Order 对象。

U.4.2.2.5 getOrderByEvent

原型：getOrderByEvent(type, eventObj)
描述：根据预订对应的事件对象获取预订对象。
参数：type – 表示eventObj参数对象类型，取值：
—— 0 – 表示eventObj为ProgramEvent对象；
—— 1 – 表示eventObj为TimeShiftEvent对象。
eventObj - ProgramEvent对象或者TimeshiftEvent对象，对象类型由type参数决定。
返回：Order对象，若无对应的Order对象，则返回null。

U.4.2.2.6 getConflictOrders

原型：Order[] getConflictOrders()
描述：获取冲突预定列表，当用户要预订新节目（应用调用 OrderManager.addEvent(obj)方法）时，如果新节目与已预订节目的开始时间按照 OrderManager.conflictInterval 有冲突，系统将不会添加节目预订，且在内存中生成预订冲突节目列表（哪些已经预订的节目和本次预订在开始时间上有冲突），供本方法访问取出。
参数：无。
返回：Order 对象数组。

U.4.2.2.7 getDelMarkedList

原型：Order[] getDelMarkedList()
描述：获取当前有删除标记的预定节目列表。
参数：无。
返回：Order 对象数组。

U.4.2.2.8 addEvent

原型：number addEvent(type, eventObj)
描述：增加一预订节目信息到用户预订列表中。
注：仅对RAM中的预定列表进行操作。
参数：type – 表示obj参数对象类型，取值：
—— 0 – 表示eventObj参数为ProgramEvent类型；
—— 1 – 表示eventObj参数为TimeShiftEvent类型。
eventObj– 待预定的ProgramEvent或TimeShiftEvent对象，类型由type参数指定。
返回：number型，取值：
—— 1 - 表示添加成功；
—— 0 - 表示节目已播；
—— -1 - 表示添加冲突；
—— -2 - 表示没有对应的业务；
—— -3 - 表示预定空间已满。

U.4.2.2.9 deleteOrder

原型：deleteOrder(orderObj)
描述：从预定节目列表中删除指定预定节目。
注：仅对 RAM 中的预定列表进行操作。
参数：orderObj - Order 对象。
返回：number 型，1 表示删除预定节目成功，0 表示删除预订节目失败。

U.4.2.2.10 deleteAll

原型：deleteAll()
描述：删除节目预定列表中的所有预定节目。
注：仅对 RAM 中的预定列表进行操作。
参数：无。
返回：无。

U.4.2.2.11 deleteAllDelMarked

原型：deleteAllDelMarked()
注：仅对 RAM 中的预定列表进行操作。
描述：从用户预定节目列表中删除指定类型所有预定节目，并且从 NVM 中清除相关信息。
参数：无。
返回：无。

U.4.2.2.12 delConflictOrders

原型：number delConflictOrders()
注：仅对 RAM 中的预定列表进行操作。
描述：删除所有有冲突的预定节目。
参数：无。
返回：number 型，取值 0 表示删除失败，1 表示删除成功。

U.4.2.2.13 restore

原型：restore()
描述：把当前 NVM 保存的节目预定列表导入到 RAM 中。
参数：无。
返回：number 型，取值 1 表示恢复 NVM 中预定数据到内存成功，0 表示失败。

U.4.2.2.14 save

原型：save()
描述：将 RAM 中的节目预定列表保存到接收终端的 NVM 中。
参数：无。
返回：无。

U.4.3 Order对象

Order 对象为本地对象，用以描述节目预订信息。

U.4.3.1 属性

Order 对象的属性见表 U.11。

表 U.11 Order 对象属性表

属性名称	类型	属性	说明
orderID	number	只读	获取当前预订的标识符，当用户在终端界面上预订一个节目时，系统即生成一个 Order 对象，并分配 orderID，而且必须保证在当前已经存在的所有用户的 Order 对象内，该 orderID 唯一。
channelObj	Channel	只读	获取当前预订所属的频道对象。
deleteFlag	number	读/写	将预订的节目打上删除的标记，取值： ——0 - 表示没有删除标记； ——1 - 表示有删除标记。
type	number	只读	表示 eventObj 对象类型，取值： —— 0 - 表示 eventObj 属性为 ProgramEvent 对象； —— 1 - 表示 eventObj 属性为 TimeShiftEvent 对象。
eventObj	ProgramEvent/ TimeShiftEvent	只读	获取当前预订对应的事件对象，类型由 type 属性指定。

U.5 信息搜索模块

信息搜索模块定义了与全局搜索和匹配搜索相关的JS对象：SearchManager、GlobalSearchSession、AutoCompleteSearchSession、GloalSearchFilter、AutoCompleteSearchFilter、GlobalSearchResultItem、AutoCompleteSearchItem和SearchHistoryItem。

U.5.1 常量

本模块的常量定义见表U.12。

表 U.12 信息搜索模块常量定义

常量	取值	描述
数据源类型		
SourceType.ALL	0	表示搜索的数据源为广播数据和本地录制数据两部分。
SourceType.BROADCAST	1	表示搜索的数据源为广播数据。
SourceType.RECORDED	2	表示搜索的数据源为本地录制数据。
搜索字段		
SearchFields.ALL_STRING_FIELDS	0	表示搜索所有字符串字段。
SearchFields.SYNOPSIS	4	表示搜索简介字段。
SearchFields.TITLE	5	表示搜索标题字段。
数据内容类型		
SearchContentType.ALL	0	表示搜索音频和视频两种格式的数据。
SearchContentType.AUDIO_ONLY	1	表示只搜索音频格式的数据。
SearchContentType.VIDEO_ONLY	2	表示只搜索视频格式的数据。
获取分页方向		
RetrieveDirection.FIRST_PAGE	0	表示按分页获取结果的情况，获取第一页数据。

表 U.12 （续）

常量	取值	描述
RetrieveDirection.NEXT_PAGE	1	表示按分页获取结果的情况，获取下一页数据。
RetrieveDirection.PREVIOUS_PAGE	2	表示按分页获取结果的情况，获取前一页数据。
过滤条件		
SearchCriteriaFlags.FLAG_NONE	0	过滤条件标记——空。
SearchCriteriaFlags.FLAG_SD_EVENT	1	过滤条件标记——过滤标清内容。
SearchCriteriaFlags.FLAG_HD_EVENT	2	过滤条件标记——过滤高清内容。
SearchCriteriaFlags.FLAG_3D_EVENT	4	过滤条件标记——过滤3D内容。
SearchCriteriaFlags.FLAG_CLEAR	32	过滤条件标记——过滤未加扰内容。
SearchCriteriaFlags.FLAG_SCRAMBLED	64	过滤条件标记——过滤加扰内容。
搜索状态		
SearchStatus.INITIATED	0	搜索状态——初始化完成。
SearchStatus.IN_PROGRESS	1	搜索状态——正在进行。
SearchStatus.COMPLETED	2	搜索状态——结束。
SearchStatus.INTERRUPTED	3	搜索状态——中断。
SearchStatus.TIMEOUT	4	搜索状态——超时后搜索停止。
SearchStatus.TIMEOUT_STOP_FAILED	5	搜索状态——超时后搜索停止失败。
SearchStatus.FAILED	6	搜索状态——搜索失败。
SearchStatus.STOP_SUCCESS	7	搜索状态——成功停止搜索。
SearchStatus.STOP_FAILED	8	搜索状态——停止搜索失败。
SearchStatus.DISPOSE_SUCCESS	9	搜索状态——成功关闭搜索。
SearchStatus.DISPOSE_FAILED	10	搜索状态——关闭搜索失败。
SearchStatus.RETRIEVAL_SUCCESS	11	搜索状态——成功获取搜索结果。
SearchStatus.RETRIEVAL_FAILED	12	搜索状态——获取搜索结果失败。
SearchStatus.RETRIEVAL_INSUFFICIENT	13	搜索状态——无足够的搜索结果可用。

U.5.2 SearchManager对象

SearchManager是一个内置对象，全局搜索和匹配搜索的管理器，是信息搜索模块的入口类。同时可以通过它来获得历史搜索记录。

U.5.2.1 方法

U.5.2.1.1 getAutoCompleteSearchSession

原型：AutoCompleteSearchSession getAutoCompleteSearchSession(autoCompleteFiter)

描述：获取匹配搜索会话对象。每个匹配搜索对象都是一个独立的搜索会话。系统在同一时间只能有一个匹配搜索会话。

参数：autoCompleteFilter - AutoCompleteSearchFilter 对象，表示匹配搜索过滤器。

返回：AutoCompleteSearchSession 对象，表示一个匹配搜索会话实例。

U.5.2.1.2 getGlobalSearchSession

原型：GlobalSearchSession getGlobalSearchSession(globalFilter,sortCriteria)

描述：获取全局搜索会话对象。每个全局搜索会话对象都是一个独立的搜索会话，为了在同一时间能有多个全局搜索会话，应用程序不仅需要获取不同对象，还要处理相应的回调功能。根据终端能力，由系统自行决定是否同时支持多个全局搜索会话。

参数：globalFilter - GlobalSearchFilter 对象，表示全局搜索过滤器。
sortCriteria - SortCriteria 对象，表示搜索结果排序准则。

返回：GlobalSearchSession 对象，表示一个全局搜索会话实例。

U.5.2.1.3 getSearchHistory

原型：SearchHistoryItem[] getSearchHistory()

描述：获取历史搜索记录列表。

参数：无。

返回：SearchHistoryItem 数组，表示历史搜索记录。若之前有成功的搜索记录，并且应用程序显式地调用了保存搜索结果方法，则此方法返回成功搜索的历史记录；若历史记录为空，则返回的数组长度为 0。

U.5.2.1.4 clearSearchHistory

原型：void clearSearchHistory()

描述：清除历史搜索记录。在任何时刻，应用程序都可以调用此方法清除历史搜索记录，以保护个人隐私。

参数：无。

返回：无。

U.5.3 GlobalSearchSession对象

GlobalSearchSession描述了一个全局搜索会话，提供了全局搜索会话以及相关控制接口。

U.5.3.1 常量

GlobalSearchSession对象的常量定义见表U.13。

表 U.13 GlobalSearchSession 对象常量定义表

常量	取值	说明
DEFAULT_PAGE_SIZE	6	全局搜索结果默认的页面大小，即每页所包含的搜索结果条目数量。

U.5.3.2 属性

GlobalSearchSession对象的属性定义见表U.14。

表 U.14 GlobalSearchSession 对象属性

属性名称	类型	读写属性	说明
pageSize	number	读写	按页返回搜索结果时，指定每页包含的结果个数。 默认值为 GlobalSearchSession.DEFAULT_PAGE_SIZE。
onGlobalSearchStart	函数指针	读写	全局搜索会话开始事件处置方法。
onGlobalSearchStop	函数指针	读写	全局搜索会话关闭事件处置方法。

表 U.14 （续）

属性名称	类型	读写属性	说明
onGlobalSearchDestroy	函数指针	读写	全局搜索会话销毁事件处置方法。
onGlobalSearchRetrieval	函数指针	读写	全局搜索会话获取到搜索结果事件处置方法。
onGlobalSearchError	函数指针	读写	全局搜索会话出错事件处置方法。

示例：

```
var globalSearchSession = SearchManager.getGlobalSearchSession(globalFilter, sortCriteria);
globalSearchSession.onGlobalSearchStart = eventHandle;
globalSearchSession.onGlobalSearchStop = eventHandle;
globalSearchSession.onGlobalSearchDestroy = eventHandle;
globalSearchSession.onGlobalSearchRetrieval = eventHandle;
globalSearchSession.onGlobalSearchError = eventHandle;
globalSearchSession.pageSize = 10;
......
globalSearchSession.startSearch("建党伟业");
......
//回调方法定义
function eventHandle(aEvent) {
   ......
   switch (aEvent){
   case COMPLETED:
       break;
   }
   ......
}
......
```

U.5.3.3 方法

U.5.3.3.1 startSearch

原型：void startSearch(searchStr)
描述：启动一个新的全局搜索请求，一旦搜索完成，系统会通过事件通知应用程序获取搜索结果。
参数：searchStr - string 型,表示用户输入的待搜索的字符串。
返回：无。

U.5.3.3.2 stopSearch

原型：void stopSearch()
描述：终止之前已经启动的全局搜索请求。
参数：无。
返回：无。

U.5.3.3.3 getSearchResultList

原型：GlobalSearchResultItem[] getSearchResultList()

描述：获取全局搜索结果，在调用该方法之前，应用程序需要等待事件触发。结果存放在同一个缓存中，一旦一个新的搜索被初始化，就不能再使用此接口来获得上次搜索的结果。

参数：无。

返回：GlobalSearchResultItem 数组，表示全局搜索结果。若无搜索结果，则返回的数组长度为 0。

U.5.3.3.4 getResultCount

原型：number getResultCount()

描述：获取搜索结果条目数量。如果搜索还没有完成并且部分更新，结果数也将会根据搜索元素的改变而更新。对于每个阈值限制，应用程序将会收到 SearchStatus.IN_PROGRESS 的通知。

参数：无。

返回：number 型，表示搜索结果条目数量。

U.5.3.3.5 dispose

原型：void dispose()

描述：销毁不再需要的全局搜索会话对象。

注：在全局搜索完成后，一旦不再需要该会话，必须调用该方法释放相应的资源。

参数：无。

返回：无。

U.5.3.3.6 retrievePage

原型：void retrievePage(retrieveDirection)

描述：从搜索结果列表中获取第一页、下一页或前一页的数据，异步方法，等待事件处理机制。

参数：retrieveDirection - number 型，用来指定获取结果的方向。

返回：无。

U.5.3.3.7 saveRecentSearchQuery

原型：void saveRecentSearchQuery()

描述：保存最新一次的搜索结果到历史记录。

参数：无。

返回：无。

U.5.3.4 回调方法

GlobalSearchSession对象的事件描述见表U.15，事件代码取值见表U.12中“搜索状态”常量定义。

表 U.15 GlobalSearchSession 对象相关事件

触发事件	回调处置方法
COMPLETED IN_PROGRESS INITIATED	onGlobalSearchStart
STOP_SUCCESS	onGlobalSearchStop

表 U.15 （续）

触发事件	回调处置方法
DISPOSE_SUCCESS	onGlobalSearchDestroy
RETRIEVAL_SUCCESS	onGlobalSearchRetrieval
FAILED INTERRUPTED TIMEOUT_STOP_FAILED TIMEOUT DISPOSE_FAILED RETRIEVAL_FAILED RETRIEVAL_INSUFFICIENT STOP_FAILED EXPAND_RESULT_FAILED	onGlobalSearchError

U.5.3.4.1 onGlobalSearchStart

描述：GlobalSearchSession对象的属性，全局搜索会话开始事件回调方法。用于监听全局搜索开始事件，应用程序应该赋给它一个事件处置函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示全局搜索事件。

返回：无。

U.5.3.4.2 onGlobalSearchStop

描述：GlobalSearchSession对象的属性，全局搜索会话停止事件回调方法。用于监听全局搜索停止事件，应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示全局搜索事件。

返回：无。

U.5.3.4.3 onGlobalSearchDestroy

描述：GlobalSearchSession对象的属性，全局搜索会话销毁事件回调方法。用于监听全局搜索销毁事件，应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示全局搜索事件。

返回：无。

U.5.3.4.4 onGlobalSearchRetrieval

描述：GlobalSearchSession对象的属性，全局搜索会话获取到搜索结果事件回调方法。用于监听全局搜索获取部分搜索结果事件，应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示全局搜索事件。

返回：无。

U.5.3.4.5 onGlobalSearchError

描述：GlobalSearchSession对象的属性，全局搜索会话出错事件回调方法。用于监听全局搜索出错事件，应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示全局搜索事件。

返回：无。

U.5.4 AutoCompleteSearchSession对象

匹配搜索会话对象，提供了匹配搜索会话控制方法。

U.5.4.1 属性

AutoCompleteSearchSession对象的属性定义见表U.16。

表U.16 AutoCompleteSearchSession对象属性

属性名称	类型	读写属性	说明
onAutoCompleteSearchStart	函数指针	读写	匹配搜索会话开始事件处置方法。
onAutoCompleteSearchStop	函数指针	读写	匹配搜索会话关闭事件处置方法。
onAutoCompleteSearchDestroy	函数指针	读写	匹配搜索会话销毁事件处置方法。
onAutoCompleteSearchRetrieval	函数指针	读写	匹配搜索会话获取到搜索结果事件处置方法。
onAutoCompleteSearchError	函数指针	读写	匹配搜索会话出错事件处置方法。

示例：

```
var autoCompleteSearchSession = SearchManager.getAutoCompleteSearchSession(autoCompleteFilter);
autoCompleteSearchSession.onAutoCompleteSearchStart = eventHandle;
autoCompleteSearchSession.onAutoCompleteSearchStop = eventHandle;
autoCompleteSearchSession.onAutoCompleteSearchDestroy = eventHandle;
autoCompleteSearchSession.onAutoCompleteSearchRetrieval = eventHandle;
autoCompleteSearchSession.onAutoCompleteSearchError = eventHandle;
......
autoCompleteSearchSession.startSearch("建党伟业");
......
//回调方法定义
function eventHandle(aEvent) {
   ......
   switch (aEvent){
   case COMPLETED:
       break;
   }
   ......
}
......
```

U.5.4.2 方法

U.5.4.2.1 startSearch

原型：void startSearch(searchStr)

描述：启动一次新的匹配搜索请求，一旦搜索完成，搜索引擎会通知应用程序获取搜索结果。启动搜索后，应用程序需要等待 onAutoCompleteSearchStart()事件通知然后去获取结果。

注 1：若 AutoCompleteSearchFilter 超时时间设置为 0，只有当显式调用 stopSearch()方法后，搜索才会停止。

注 2：

——通过调用 getSearchResultList()方法来获取匹配搜索结果；

——通过调用 stopSearch()方法终止前面一个还在进行中的搜索请求；

——如果不再使用匹配搜索会话，通过调用 dispose()方法来释放它；

——在同一个时间内，只能有一个匹配搜索会话处于活动状态，在这个期间可以重复调用 startSearch()和 stopSearch()方法，但是当不再需要使用匹配搜索会话时，应调用 dispose()方法关闭搜索会话并释放资源；搜索结果存放在同一个缓存中，一旦启动一个新的搜索,旧的搜索结果将不再有效。

参数：searchStr - string 型，表示用户输入的待搜索的关键字字符串。

返回：无。

U.5.4.2.2 stopSearch

原型：void stopSearch()

描述：终止之前已经启动的匹配搜索请求。

注 1：该操作只是终止匹配搜索，但是匹配搜索会话仍然有效，相应的资源也没有被释放。在下面的场景中该方法应该被强制调用:

——若已经开始匹配搜索且在等待返回结果，在搜索引擎返回结果之前，用户输入了另外的字符，必须调用 stopSearch()方法终止搜索；

——匹配搜索过滤器超时时间限制设为 0，这样会一直等待搜索结果，若没有相应的匹配结果，在启动另外一个搜索之前，必须调用 stopSearch()方法终止搜索。

注 2：若匹配搜索已经完成，可不用调用该方法。

参数：无。

返回：无。

U.5.4.2.3 dispose

原型：void dispose()

描述：销毁不再需要的匹配搜索会话对象。

注：在匹配搜索完成后，一旦不再需要该会话，必须调用该方法以释放相应的资源。

参数：无。

返回：无。

U.5.4.2.4 getSearchResultList

原型：string[] getSearchResultList()

描述：获取匹配搜索的结果。

参数：无。

返回：string 数组，表示所有搜索到的与用户输入的查询关键字相匹配的字符串数组。

U.5.4.3 回调方法

AutoCompleteSearchSession对象的事件描述见表U.17。

表 U.17 AutoCompleteSearchSession 对象相关事件

搜索事件	回调处置方法
COMPLETED	onAutoCompleteSearchStart
STOP_SUCCESS	onAutoCompleteSearchStop
DISPOSE_SUCCESS	onAutoCompleteSearchDestroy
FAILED TIMEOUT STOP_FAILED DISPOSE_FAILED	onAutoCompleteSearchError

U.5.4.3.1 onAutoCompleteSearchStart

描述：AutoCompleteSearchSession对象的属性，匹配搜索会话开始事件回调方法，用于监听匹配搜索开始事件。应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示匹配搜索事件。

返回：无。

U.5.4.3.2 onAutoCompleteSearchStop

描述：AutoCompleteSearchSession对象的属性，匹配搜索会话停止事件回调方法，用于监听匹配搜索停止事件。应用程序应该赋给它一个事件处理函数句柄来处理相应的事件

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示匹配搜索事件。

返回：无。

U.5.4.3.3 onAutoCompleteSearchDestroy

描述：AutoCompleteSearchSession对象的属性，匹配搜索会话销毁事件回调方法，用于监听匹配搜索销毁事件。应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示匹配搜索事件。

返回：无。

U.5.4.3.4 onAutoCompleteSearchError

描述：AutoCompleteSearchSession对象的属性，匹配搜索会话出错事件回调方法，用于监听匹配搜索出错事件。应用程序应该赋给它一个事件处理函数句柄来处理相应的事件。

原型：function eventHandle(aEvent)

参数：aEvent – number型，表示匹配搜索事件。

返回：无。

U.5.5 GlobalSearchFilter对象

全局搜索过滤器对象，提供了全局搜索过滤条件设置和获取方法。全局搜索过滤条件可以是：

——搜索数据源；
——搜索字段；
——文本信息语种；
——返回结果的最大数目；
——搜索超时时间限制。

注：应用程序使用GlobalSearchFilter可以获得以下效果：按照指定的搜索数据源和搜索字段，在超时时间限制范围内列出与用户输入的关键字相匹配的搜索结果列表，搜索结果数量小于等于应用程序设定的最大值。

U.5.5.1 常量

GlobalSearchFilter对象的常量定义见表U.18。

表U.18 GlobalSearchFilter对象常量定义表

常量	取值
DEFAULT_MAX_GLOBAL_SEARCH_RESULTS	50

U.5.5.2 属性

GlobalSearchFilter对象的属性定义见下表。

表U.19 GlobalSearchFilter对象属性定义

属性名称	类型	读写属性	说明
source	number	读写	指定搜索源，默认为SourceType.ALL。
contentType	number	读写	表示内容类型，默认为SearchContentType.ALL。
contentNibble	number	读写	表示待过滤节目的分类。
searchField	number	读写	搜索字段，默认为SearchFields.ALL_STRING_FIELDS。
searchLanauage	string	读写	接收终端预置系统语种，默认为“zho”。
criteriaFlags	number	读写	表示过滤条件。
maxResults	number	读写	表示最大返回结果数目，默认值为GlobalSearchFilter.DEFAULT_MAX_GLOBAL_SEARCH_RESULTS。
threshold	number	读写	设置搜索查询结果的阈值，默认为0。
timeLimit	number	读写	超时设置，默认为0，单位为毫秒。

U.5.5.3 方法

U.5.5.3.1 GlobalSearchFilter

原型：GlobalSearchFilter()
描述：构造方法，创建一个GlobalSearchFilter对象。
参数：无。

U.5.6 AutoCompleteSearchFilter对象

匹配搜索过滤器对象，提供了匹配搜索过滤条件设置和获取方法。匹配搜索过滤条件可以是：
——搜索数据源；

——搜索字段；
——文本信息语种；
——返回结果的最大数目；
——搜索超时时间限制。

注：应用程序使用AutoCompleteSearchFilter可以获得以下效果：按照指定的搜索数据源和搜索字段，在超时时间限制范围内列出与用户输入的字符相匹配的提示字符串列表，字符串数量小于等于应用程序设定的最大值。

U.5.6.1 常量

AutoCompleteSearchFilter对象的常量定义见下表。

表U.20 AutoCompleteSearchFilter对象常量

常量	取值
DEFAULT_MAX_AUTO_COMPLETE_SEARCH_RESULTS	10

U.5.6.2 属性

AutoCompleteSearchFilter对象的属性定义见下表。

表U.21 AutoCompleteSearchFilter对象属性

属性名称	类型	属性	说明
source	number	读写	指定搜索源，默认为SourceType.ALL。
searchField	number	读写	搜索字段，默认值为SearchFields.ALL_STRING_FIELDS。
searchLanauage	string	读写	表示搜索文字语种，遵循GB/T 4880.2—2000标准约定的三字母代码，默认值为“zho”。
maxResults	number	读写	表示最大返回结果数目，默认值为AutoCompleteSearchFilter.DEFAULT_MAX_AUTO_COMPLETE_SEARCH_RESULTS。
timeLimit	number	读写	超时设置，默认值为0，单位为毫秒。

U.5.6.3 方法

U.5.6.3.1 AutoCompleteSearchFilter

原型：AutoCompleteSearchFilter()
描述：构造方法，创建一个AutoCompleteSearchFilter对象。
参数：无。

U.5.7 SortCriteria对象

SortCriteria对象描述了搜索结果的排序机制，包括两种规则，一是按照指定的字段排序，二是按照升序还是降序使用。两种规则需要结合使用，即对某个字段进行升序或者降序排序。

U.5.7.1 常量

SortCriteria对象的常量定义见表U.22。

表U.22 SortCriteria对象常量

常量	取值
排序方式（升序/降序）	
SORT_ORDER_NONE	0
SORT_ORDER_ASCENDING	1
SORT_ORDER_DESCENDING	2
排序依据	
SORT_TYPE_NONE	0
SORT_TYPE_TITLE	1
SORT_TYPE_START_TIME	2
SORT_TYPE_CONTENT_NIBBLE	15

U.5.7.2 属性

SortCriteria对象的属性定义见表U.23。

表U.23 SortCriteria对象属性定义表

属性名称	类型	读写属性	说明
sortOrder	number	读写	可取值为表W.22中定义的“排序方式”常量。
sortType	number	读写	可取值为表W.22中定义的“排序依据”常量。

U.5.7.3 方法

U.5.7.3.1 SortCriteria

原型：SortCriteria()
描述：构造方法，创建一个默认的SortCriteria对象，用于搜索结果的排序。
参数：无。

U.5.7.3.2 SortCriteria

原型：SortCriteria(field, order)
描述：构造方法，根据指定的参数创建一个SortCriteria对象，用于搜索结果的排序。
参数：field- number型，指定针对哪个字段进行排序，可取值为表W.22中定义的“排序依据”常量。
order - number型，指定排序的升降类型，可取值为表W.22中定义的“排序方式”常量。

U.5.8 GlobalSearchResultItem对象

U.5.8.1 常量

GlobalSearchResultItem对象的常量定义见表U.24。

表 U.24 GlobalSearchResultItem 对象常量

常量	取值	描述
CONTENT_DVBEVENT	0	表示内容类型为DvbEvent类型。
CONTENT_PVREVENT	1	表示内容类型为PVREvent类型。

U.5.8.2 属性

GlobalSearchResultItem对象的属性定义见表U.25。

表 U.25 GlobalSearchResultItem 对象属性

属性名称	类型	读写属性	说明
contentType	number	只读	可取值为 CONTENT_DVBEVENT 或 CONTENT_PVREVENT。

U.5.8.3 方法

U.5.8.3.1 getContent

原型：object getContent()
描述：获取与此搜索结果相关联的对象，该对象可能为 DvbEvent 或 PVREvent 类型，由 contentType 属性指示。
参数：无。
返回：object 对象类型，具体类型根据 contentType 属性的值判定：
——若 contentType=GlobalSearchResultItem.CONTENT_DVBEVENT，则返回 DvbEvent 对象；
——若 contentType=GlobalSearchResultItem.CONTENT_PVREVENT，则返回 PVREvent 对象；

U.5.9 SearchHistoryItem对象

SearchHistoryItem对象描述了一条搜索历史记录，提供了获取搜索历史记录各种信息的方法。

U.5.9.1 方法

U.5.9.1.1 getContentType

原型：number getContentType()
描述：获取该搜索历史记录的内容类型。
参数：无。
返回：number 型，表示搜索历史记录的内容类型。

U.5.9.1.2 getCriteriaFlags

原型：number getCriteriaFlags()
描述：获取该搜索历史记录的过滤符。
参数：无。
返回：number 型，表示搜索历史记录的过滤条件。

U.5.9.1.3 getSearchField

原型：number getSearchField()

描述：获取该搜索历史记录的搜索字段。
参数：无。
返回：number 类型，表示搜索历史记录的搜索字段。

U.5.9.1.4 getSearchString

原型：string getSearchString()
描述：获取该搜索历史记录的搜索关键字字符串。
参数：无。
返回：string 型，表示用户输入的搜索关键字字符串。

U.5.9.1.5 getSortCriteria

原型：SortCriteria getSortCriteria()
描述：获取该搜索记录的排序信息。
参数：无。
返回：SortCriteria 对象，表示搜索结果排序方式和排序依据。

U.5.9.1.6 getSource

原型：number getSource()
描述：获取该搜索历史记录的搜索源。
参数：无。
返回：number 型，表示搜索源。

附 录 V
（规范性附录）
JavaScript-广播信息服务管理单元

V.1 概述

本附录定义了广播信息服务管理单元的相关JavaScript接口。

V.2 广播信息服务管理模块

本模块定义了广播信息服务管理相关的JS对象：DthManager、Ad、AdService。

V.2.1 消息

广播信息服务管理模块的消息定义见表 V.1。

表 V.1 广播信息服务管理模块消息定义

消息名称	event.which	event.modifiers	消息说明
DTH_EVENT_EMBD_TRIG	15001	number	应急广播事件触发事件，消息字符串 JSON 格式为： {“service_id”:param1 注1， “ts_id”:param2 注2， “orig_net_id”:param3 注3}
DTH_EVENT_EMBD_CANCEL	15002	-	应急广播事件取消事件。
DTH_EVENT_OSD_UPDATE	15003	number	提示应用，有新的 OSD 的提示信息
DTH_EVENT_SERVICE_UPDATE	15004	number，参数为 0 表示不需要立即更新节目，参数为 1 表示强制更新节目信息	NIT 业务更新
DTH_EVENT_RESET_DATA	15005	参数为 01 时，要求立即擦除机顶盒数据（dth 处理，只是上报一个消息给应用），参数为 0 时，表示从擦除机顶盒的状态恢复为正常状态。	擦除数据上报的消息
DTH_EVENT_GPRS_SEND_STATUS	15006	消息为 01 表示发送成功，消息为 00 表示发送失败	发送数据是否成功的判定
DTH_EVENT_BOUQUET_ID_UPDATE	15007	number	Bouquet id 更新消息

表 V.1（续）

消息名称	event.which	event.modifiers	消息说明
DTH_EVENT_UPGRADE_TRIG	15008	消息为 01 表示强制升级，消息为 00 表示非强制升级	软件升级消息
DTH_EVENT_FINGERPRINT_TRIG	15009	number	指纹触发事件。注 4
DTH_EVENT_GPRS_STATUS	15010	number	当前可用状态。注 5
DTH_EVENT_GPRS_BASE_STATION	15011	number	GPRS 当前基站信息。注 6
保留	15012-15100	-	

event.modifiers 值由系统内部自动给出，其数据类型：

——“number”，表示该值为消息描述字符串的 ID，可通过 Utility.getEventInfo()方法获取该消息描述字符串。若“消息说明”定义了消息字符串 JSON 格式，则按格式取出消息内容。

——“-”，表示 event.modifiers 为 undefined。

注 1：param1：number 型，节目服务 ID；

注 2：param2：number 型，传输流 ID。

注 3：param3：number 型，网络 ID。

注 4：指纹触发事件，消息字符串 JSON 格式为：{

char_color：number //文本颜色

reg_hight：number //背景区域高度

reg_widgth：number //背景区域宽度

reg_color：number //背景区域颜色

X_Reg_Offset：number //背景区域 x 坐标

Y_Reg_Offset：number //背景区域 y 坐标

X_Text_Offset：number //文本 x 坐标

Y_Text_Offset：number //文本 y 坐标

Duration：number //持续时间

fs_text：String //文本内容

};

注 5：当前可用状态：{

0, //模块正常当前可用

-1, //模块正在初始化

-2, //模块 SIM 异常

-3, //模块网络异常

-4, //模块忙状态，需要稍后获取。

-5, //模块无应答

-6, //其他错误

}

表 V.1（续）

注 6：基站信息回报事件，数组内包含 JSON 格式为：[{

```
LAC:number          /*location area code 位置区号*/
Cell_ID:number        /*小区(基站)编号*/
Bsic_ID:number       /*基站识别码(C 网)*/
ucMNC:number         /*网络号*/
strength:number      /*信号强度*/
},
…
]
```

V.2.2 DthManager对象

DthManager对象为内置对象，提供了DTH组件的管理方法。

V.2.2.1 方法

V.2.2.1.1 startServer

原型：number startServer()

描述：DTH 组件在直播星应用里面被调用。DTH 组件 server 端启动，需要等直播星应用启动之后，调用本接口才能开始 DTH 组件的各个功能。

参数：无。

返回：number 型，取值：0 ： 表示成功。 非 0 ： 表示失败。

V.2.2.1.2 stopServer

原型：number stopServer()

描述：直播星应用退出之后，调用本接口停止 DTH 组件的各个功能。

参数：无。

返回：number 型，取值：0 ： 表示成功。 非 0 ： 表示失败。

V.2.2.1.3 getADFinishStatus

原型：number getADFinishStatus()

描述：获取开机广告处理状态，即获取广告是否准备好。

参数：无。

返回：number 型，取值：1 表示处理完成, 应用可以正常工作, 0 表示正在处理, 应用需继续等待。

V.2.2.1.4 getAllAds

原型：Ad[] getAllAds(epgInfoType,serviceid)

描述： 获取广告图片信息对象。

参数： epgInfoType - number 型，表示各种广告位类型，定义如下

```
enum {
        EPGINFO_TYPE_EPG_CONFIG_DATA = 0xF0F0,
        EPGINFO_TYPE_BOOTLOGO = 1,
```

```
        EPGINFO_TYPE_MAINMENU,
        EPGINFO_TYPE_CHANNEL_LIST,
        EPGINFO_TYPE_FAV_CHANNEL_LIST,
        EPGINFO_TYPE_EPG_LIST,
        EPGINFO_TYPE_PFBAR,
        EPGINFO_TYPE_VOLUMEBAR,
        EPGINFO_TYPE_AUDIOLOGO,
        EPGINFO_TYPE_ERROR,
    };
```

serviced - number 类型 当为 EPGINFO_TYPE_PFBAR 或 EPGINFO_TYPE_VOLUMEBAR 时，serviceid 为对应的广告对应的频道的 serviceid，否则其他为 0。

返回：Ad 对象数组。

V.2.2.1.5 getOsdXmlFile

原型：String getOsdXmlFile()

描述：获取 OSD 文本更新的 XML 文件的保存地址。

参数：无。

返回：String 型，OSD 文本更新的 XML 文件在盒子中的保存地址。

V.2.2.1.6 getOsdInfo

原型：String getOsdInfo(tag)

描述：获取需要显示的 osd 事件信息。

参数：tag - String 型。为 osd 对应的的提示信息代码。

返回：String 型，返回对应的 osd 事件信息。

V.2.2.1.7 startAdPCT

原型：number startAdPCT(transport_stream_id)

描述： 获取开机广告第一张图片。应改为开启某个 tp 的实时广告过滤

参数：transport_stream_id - number 型，传输流 id。

返回：number 型，取值：0 表示获取成功 非 0 表示获取失败，应用显示默认图片。

V.2.2.1.8 stopEMBDAction

原型：number stopEMBDAction()

描述：直播星接收到取消应急广播的事件信息，调用本接口，使 DTH 组件停止发送当前应急广播。

参数：无。

返回：number 型，取值：0：操作成功， 非 0： 操作失败。

V.2.2.1.9 recordAVBEvent

原型：recordAVBEvent(event_id, event_param)

描述：检测用户事件并保存。

参数：event_id - number 型，事件编号；

event_param - number 型，根据事件编号设置事件参数。

返回：无。

V.2.2.1.10 dataBDStart

原型：number dataBDStart(file_path)
描述： 提供信息服务文件保存路径，启动信息服务接收。
参数：file_path - String 型，信息服务文件的保存路径，若参数为空，则默认路径为/data/db/dth/datadb。
返回：number 型，取值：0：操作成功，非 0：操作失败。

V.2.2.1.11 dataBDStop

原型：number dataBDStop()
描述：中止信息服务接收。
参数：无
返回：number 型，取值：0：操作成功， 非 0： 操作失败。

V.2.2.1.12 getDATABDFinishPercent

原型：number getDATABDFinishPercent()
描述：得到下载完成的百分比。
参数：无
返回：number 型，取值：0-100：下载完成的百分比，其他值：操作失败。

V.2.2.1.13 getDataBDXmlFileData

原型：String getDataBDXmlFileData(file_name)
描述：获取到对应的信息服务数据。
参数：file_name - String 型，信息服务文件的文件名，若参数为空，则默认路径为/data/db/dth/datadb。
返回：String型，返回对应的信息服务数据。

V.2.2.1.14 GPRSTransmit

原型：String GPRSTransmit(conn_type, addr, port, data, timeout,retry_count)
描述： 通过GPRS模块发送数据并接受返回数据。(异步接口)
参数：conn_type - number型，GPRS连接服务器的类型取值为0或者1，0：TCP；1：UDP；
addr - String型，服务器的域名或IP地址；
port - number型，服务器的端口号；
data - String型，传输的数据；
timeout - number型，传输数据等待返回的超时时间；
retry_count - number型，传输失败的重试次数。
返回：String型：为空则表示失败，非空则为接收模块返回的数据。

V.2.2.1.15 getGPRSStatus

原型：number getGPRSStatus()
描述：获取 GPRS 当前是否处于正常可用状态。
参数：无。
返回：number 型，取值：0：GPRS 状态正常， 非 0： GPRS 状态异常。

V.2.2.1.16 getGprsBaseStationInfo

原型：number getGprsBaseStationInfo()
描述：异步接口，获取当前基站信息。
参数：无
返回：0 表示成功，其他表示失败。

V.2.2.1.17 SaveNITServiceUpdateVersion

原型：number SaveNITServiceUpdateVersion()
描述：等应用更新完节目信息之后，通知 dth 对当前业务更新描述符版本进行保存。
参数：无。
返回：number 型，取值：0：保存成功，非 0：保存失败。

V.2.2.1.18 GetBouquetId

原型：number GetBouquetId()
描述：获取当前 bouquet id。
参数：无。
返回：number 型，当前的 bouquetid。

V.2.2.1.19 SaveNITServiceUpdateVersion

原型：number SaveNITServiceUpdateVersion()
描述：等应用更新完节目信息之后，通知dth对当前业务更新描述符版本进行保存。
参数：无。
返回：number型,0表示成功，其他表示失败。

V.2.2.1.20 dataBDdeleteFiles

原型：number dataBDdeleteFiles()
描述：删除下载的信息服务所有文件（退出信息服务时调用）。
参数：无。
返回值：number型,0表示成功，其他表示失败。

V.2.3 Ad对象

V.2.3.1 属性

Ad 对象的属性定义见表 V.2。

表 V.2 Ad 对象属性

属性名称	类型	读写属性	说明
adPath	string	只读	表示广告的保存地址。
startDate	string	只读	表示广告的起始播放日期，格式为"YYYY-MM-DD"。
startTime	string	只读	表示广告的起始播放时间，格式为"hh:mm:ss"。
endDate	string	只读	表示广告的播放结束日期，格式为"YYYY-MM-DD"。
endTime	string	只读	表示广告的终止播放时间，格式为"hh:mm:ss"。

表 V.2 （续）

属性名称	类型	读写属性	说明
duration	number	只读	表示广告的播放时间，单位为秒。
tableExtId	number	只读	表示广告的节目关联广告类型。

V.2.3.2 方法

V.2.3.2.1 getAllAdServices

原型：AdService[] getAllAdServices()
描述： 获取频道相关广告的所有相关频道参数，AdService对象。
参数：无。
返回值：AdService对象数组。

V.2.4 AdService对象

V.2.4.1 属性

AdService 对象的属性定义见表 V.3。

表 V.3 AdService 对象属性

属性名称	类型	读写属性	说明
onId	number	只读	表示频道相关广告的 onid。
tdId	number	只读	表示频道相关广告的 tdId。
serviceId	number	只读	表示频道相关广告的 serviceId。
associateType	number	只读	表示频道相关广告的 associateType。

附 录 W
（规范性附录）
JavaScript-多屏互动单元

W.1 概述

本附录定义了多屏互动单元相关的JavaScript接口。

W.1.1 场景描述

多屏互动单元的使用场景描述如下：

a) 当前具有两台TVOS系统的播放设备，分别为A与B，A、B系统均连接至同一个局域网。

b) A系统的应用程序通过A系统提供的MultiScreen多端联动的通信对象，使用startMultiScreenServer方法启动多端联动服务，将B系统的多端联动组件信息传递给A系统。

c) 接着应用程序使用findSPs方法搜索到同一个局域网下的B系统设备，通过connect方法经由A系统实现与B系统的连接。

d) 连接成功后，A系统程序通过B系统提供的MultiScreenEvent接口，接收连接状态信息，并由onConnected方法通知A系统连接状态。至此A、B系统实现多端联动通讯，应用程序可由MultiScreen对象提供的inputKeyCode等方法将A系统的操作数据信息，或是更多视频数据信息传递给B系统，在B系统上实现相关操作与播放。

W.2 多屏互动模块

本模块支持在局域网中客户端和服务端进行发现和连接，多屏互动组件对Web APP上层应用提供的JavaScript接口，可实现WEB应用在局域网内发现、连接、控制服务端设备功能。

本模块定义了多屏互动相关的JS对象：MultiScreen。

W.2.1 MultiScreen对象

W.2.1.1 方法

W.2.1.1.1 startMultiScreenServer

原型：int startMultiScreenServer (String spName, String spDeviceType, String spServiceInfo, String spVersion,String ipaddress, int port, String hostname)

描述：远程接口，启动多屏互动组件服务端。

参数：

spName – String类型，多屏互动组件服务端名称；

spDeviceType – String类型，多屏互动组件服务端设备类型；

spServiceInfo – String类型，多屏互动组件服务端服务信息；

spVersion – String类型，多屏互动组件服务端版本；

ipaddress – String类型，搜索到的多屏互动组件设备ip地址；

port – int型，搜索到的多屏互动组件设备端口号；

hostname – String类型，搜索到的多屏互动组件设备主机名。

返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.1.1.2 stopMultiScreenServer

原型：int stopMultiScreenServer()
描述：远远程接口，停止多屏互动组件服务端。
参数：无。
返回：int 型，远程接口调用成功返回 0，否则返回错误码。

W.2.1.1.3 startMultiScreenClient

原型：int startMultiScreenClient(String clientName)
描述：远程接口，启动多屏互动组件客户端。
参数：clientName - String 类型，多屏互动组件客户端名称。
返回：int 型，远程接口调用成功返回 0，否则返回错误码。

W.2.1.1.4 stopMultiScreenClient

原型：int stopMultiScreenClient()
描述：远程接口，关闭多屏互动组件客户端。
参数：无。
返回：int 型，远程接口调用成功返回 0，否则返回错误码。

W.2.1.1.5 findSPs

原型：int findSPs()
描述：远程接口，搜索局域网下的多屏互动服务组件设备。
参数：无。
返回：int 型，远程接口调用成功返回 0，否则返回错误码。

W.2.1.1.6 connect

原型：int connect(String spName, String spDeviceType, String spServiceInfo, String spVersion,String ipaddress, int port, String hostname)
描述：远程接口，连接局域网下的多屏互动服务组件服务端设备。
参数：spName – String类型，多屏互动组件服务端名称；
spDeviceType – String类型，多屏互动组件服务端设备类型；
spServiceInfo – String类型，多屏互动组件服务端服务信息；
spVersion – String类型，多屏互动组件服务端版本；
ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。
返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.1.1.7 queryInfo

原型：int queryInfo(String ipaddress, int port, String hostname,String cmdid,String atrribute,String params)
描述：远程接口，多屏互动组件客户端请求获取。

参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
cmdid – String类型，请求信息的指令id；
atrribute – String类型，请求信息的指令名称；
params – String类型，请求信息的指令参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.1.1.8 execCmd

原型：int execCmd(String ipaddress, int port, String hostname,String cmd,String param)
描述：远程接口，多屏互动组件客户端请求执行指令。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
action – String类型，按键指令；
param – String类型，按键指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.1.1.9 inputKeyCode

原型：int inputKeyCode(String ipaddress, int port, String hostname,String action,String param)
描述：远程接口，多屏互动组件客户端发送虚拟按键输入的指令。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
action – String类型，按键指令；
param – String类型，按键指令附带的参数。
返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.1.1.10 boardCastAllDevice

原型：int boardCastAllDevice(String cmd,String param)
描述：远程接口，多屏互动组件服务端往所有连上的设备发送广播。
参数：cmd – String类型，广播指令；
param – String类型，广播指令附带的参数；
返回：int型，远程接口调用成功返回0，否则返回错误码。

W.2.2 消息回调 EventHandler

```
interface MultiScreenEvent : Event {
    readonly attribute String spName;
    readonly attribute String spDeviceType;
    readonly attribute String spServiceInfo;
    readonly attribute String spVersion;
    readonly attribute String ipaddress;
```

```
        readonly attribute int port;
         readonly attribute String hostname;
         readonly attribute String  id;
         readonly attribute String  attribute;
         readonly attribute String  param;
         readonly attribute String  action;
         readonly attribute String  cmd;
    };
```

W.2.2.1 方法

W.2.2.1.1 onSpFounded

原型：[RuntimeEnabled= MultiScreenEvent]EventHandler onSpFounded
描述：通知Web APP多屏互动组件服务搜索完成。
参数：spName – String类型，搜索到的多屏互动组件服务名；
spDeviceType – String类型，搜索到的多屏互动组件设备类型；
spServiceInfo – String类型，搜索到的多屏互动组件服务端信息；
spVersion – String类型，搜索到的多屏互动组件服务端服务版本；
ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.2 onConnected

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onConnected
描述：通知Web APP多屏互动组件服务连接完成。
参数：spName – String类型，搜索到的多屏互动组件服务名；
spDeviceType – String类型，搜索到的多屏互动组件设备类型；
spServiceInfo – String类型，搜索到的多屏互动组件服务端信息；
spVersion – String类型，搜索到的多屏互动组件服务端服务版本；
ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.3 onConnectRefused

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onConnectRefused
描述：通知Web APP多屏互动组件服务连接被拒。
参数：spName – String类型，搜索到的多屏互动组件服务名；
spDeviceType – String类型，搜索到的多屏互动组件设备类型；
spServiceInfo – String类型，搜索到的多屏互动组件服务端信息；
spVersion – String类型，搜索到的多屏互动组件服务端服务版本；
ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.4 onDisconnected

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onConnectRefused
描述：通知Web APP多屏互动组件服务连接被拒。
参数：spName – String类型，搜索到的多屏互动组件服务名；
spDeviceType – String类型，搜索到的多屏互动组件设备类型；
spServiceInfo – String类型，搜索到的多屏互动组件服务端信息；
spVersion – String类型，搜索到的多屏互动组件服务端服务版本；
ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.5 onServiceActivited

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onServiceActivited
描述：通知Web APP多屏互动组件服务被激活。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.6 onServiceDeactived

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onServiceDeactived
描述：通知Web APP多屏互动组件服务被注销。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名。

W.2.2.1.7 onQueryInfo

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onQueryInfo
描述：通知Web APP接收到多屏互动组件客户端发过来的数据请求。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
id – String类型，接收到的请求的命令id；
attribute – String类型，接收到的请求的命令属性；
param – String类型，接收到的请求的附带的参数。

W.2.2.1.8 onQueryResponse

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onQueryResponse
描述：通知Web APP接收到多屏互动组件服务端回复了数据请求。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
id – String类型，接收到的请求的命令id；

attribute – String类型，接收到的请求的命令属性；
param – String类型，接收到的请求的附带的参数。

W.2.2.1.9 onExecute

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onExecute
描述：通知Web APP接收到多屏互动组件客户端发送了执行指令请求。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
cmd – String类型，执行的指令；
param – String类型，执行指令附带的参数。

W.2.2.1.10 onInputKeyCode

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onInputKeyCode
描述：通知Web APP接收到多屏互动组件客户端发送了执行按键注入的请求。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
cmd – String类型，执行的指令；
param – String类型，执行指令附带的参数。

W.2.2.1.11 onNotify

原型：[RuntimeEnabled= MultiScreenEvent] EventHandler onNotify
描述：通知Web APP多屏互动组件接收到通知。
参数：ipaddress – String类型，搜索到的多屏互动组件设备ip地址；
port – int型，搜索到的多屏互动组件设备端口号；
hostname – String类型，搜索到的多屏互动组件设备主机名；
cmd – String类型，接收到的指令；
param – String类型，接收到的指令参数。

附 录 X
（规范性附录）
JavaScript-DRM 管理单元

X.1 概述

本附录定义了 DRM 管理模块的 JavaScript 接口。

X.2 DRM管理模块

X.2.1 ChinaDrmManager对象

ChinaDrmManager 对象负责 DRM APP 的注册、注销和命令的发送。

示例：

```
var drmManager=new ChinaDrmManager();//创建 ChinaDrmManager 对象
var uuid=new String("2d7d041436f2a048a0c4c1cccbb64546");
drmManager.ChinaDrm_RegisterApp(“unitend”,uuid, 0, register_pridata,“unitend”,1,0,0);//DRM APP 注册
var resultdata = drmManager.ChinaDrm_SendCommandToTA(1,“unitend”);//给 TEE 发送命令
drmManager.ChinaDrm_UnRegisterApp();//注销 DRM APP
```

X.2.1.1 消息

DRM 模块给应用层发送的消息定义见表 X.1。

表 X.1 DRM 模块消息定义

消息名称	event.which	event.modifiers	消息说明
MSG_DRM_LICENSEREQ	11000	number	DRM 许可证获取消息，消息字符串 JSON 格式： ｛“DrmLicensereqData”:param1， “DrmLicenseDataLen”:param2｝。 其中，param1 是获取许可证的相关数据；param2 是获取许可证的相关数据的长度。
MSG_DRM_DECRYPTREQ	11001	number	DRM 解密消息，消息字符串 JSON 格 ｛“DrmDecryptreqData”:param1， “DrmDecryptreqDataLen”:param2｝。 其中，param1 是用于解密流的相关数据；param2 是用于解密流的相关数据的长度。

表 X.1（续）

消息名称	event.which	event.modifiers	消息说明
MSG_DRM_MESSAGE	11002	number	DRM 通知消息，消息字符串 JSON 格式： {“DrmMessageType”:param1， “DrmMessage”:param2， “DrmMessageLen”：param3}。 其中，param1 是通知消息的类型；param2 是通知消息传来的数据；param3：是通知消息传来的数据的长度。

X.2.1.2 方法

X.2.1.2.1 ChinaDrm_RegisterApp

原型：number ChinaDrm_RegisterApp (DrmSystemID, TAUUID register_commandid,register_pridata,enflag,licensereq_commandId,decrypt_commandId)

描述：DRM APP注册扩展，可以自己私有定义licensereq和decrypt 命令ID。

参数：DrmSystemID – string型，表示DRM APP唯一标识；

TAUUID – string型，表示DRM APP对应TApp的唯一标识；

register_commandid – number型，表示与TApp通信的注册commandid；

register_pridata – string型，表示与TApp通信的注册携带的私有注册数据，用于TApp验证DRM APP合法性；

enflag – number型，表示解密调用方式；

licensereq_commandId – number型，表示查询许可证对应的commandid；

decrypt_commandId – number型，表示解密数据对应的commandid。

返回：number型，0成功，非0 失败。

X.2.1.2.2 ChinaDrm_UnRegisterApp

原型：number ChinaDrm_UnRegisterApp()

描述：DRM APP注销。

参数：无。

返回：number型，0成功，非0 失败。

X.2.1.2.3 ChinaDrm_SendCommandToTA

原型：string ChinaDrm_SendCommandToTA(commandId, sendData)

描述：发送命令到TEE。

参数：commandId – string型，表示命令ID;;

sendData – string型，表示发送的数据。

返回：string型，返回JSON格式的TEE处理结果。

X.2.1.2.4 ChinaDrm_SendMessageToPlayer

原型：number ChinaDrm_SendMessageToPlayer(type,message)

描述：发送消息到播放器。

参数：type – number型，表示消息类型；
Message – string型，表示消息数据。
返回：number型，0表示成功，非0表示失败。

附 录 Y
（规范性附录）
JavaScript-DCAS 管理单元

Y.1 概述

本附录定义了与 DCAS 管理相关的功能模块,可以通过 DCAS JavaScript 应用程序编程接口开发 DCAS 用户端软件及应用软件。

Y.2 EPG DCAS模块

本模块定义了 DCAS 模块与 EPG 的相关 JavaScript 对象：EPG_DCAS 对象。

Y.2.1 EPG_DCAS对象

EPG_DCAS 对象为内置对象，提供了 EPG_DCAS 模块的 JavaScript 方法。

Y.2.1.1 方法

Y.2.1.1.1 getActivationStatus

原型：number getActivationStatus()
描述：获取激活状态。
参数：无。
返回：number 型，取值：0-表示已经激活。 非 0-表示状态未知。

Y.2.1.1.2 getBeidouInfo

原型：string getBeidouInfo()
描述：获取当前北斗信息。
参数：无。
返回：String 型，当前的北斗信息，字符串 JSON 格式为：

{
“islocked”:param1,
“latitude”:param2,
“longitude”:param3
“sat num”:param4
“SNR”:param5
}。

Y.2.1.1.3 getCASVersion

原型：string getCASVersion()
描述：获取 CAS 版本信息。
参数：无。

返回：String 型，取值：为空则表示未获取到 CAS 版本信息，非空则为收到的 CAS 版本信息。

Y.2.1.1.4 getCASVenderId

原型：number getCASVenderId()
描述：获取当前 CAS 厂商 ID。
参数：无。
返回：number 型，取值：0 则表示未获取到当前 CAS 厂商 ID，非 0 则为获取到当前 CAS 厂商 ID。

Y.2.1.1.5 getChipID

原型：string getChipID()
描述：获取芯片 ID。
参数：无。
返回：String 型，取值：为空则表示未获取到芯片 ID，非空则为获取到芯片 ID。

Y.2.1.1.6 getHSMID

原型：string getHSMID()
描述：获取 HSM 芯片 ID。
参数：无。
返回：String 型，取值：为空则表示未获取到 HSM 芯片 ID，非空则为获取到 HSM 芯片 ID。

Y.2.1.1.7 getValidPosition

原型：string getValidPosition()
描述：获取当前机顶盒允许使用的有效位置信息。
参数：无。
返回：String 型，取值：为空则表示未获取到有效位置信息，非空则为获取到有效位置信息。返回值为 JSON 格式：

```
{
“latitude”:param1,
“longitude”:param2
}。
```

Y.2.1.1.8 getPersonalBits

原型：number getPersonalBits()
描述：获取区域个人 Bits。
参数：无。
返回：number 型，取值：0 则表示未获取到区域个人 Bits，非 0 则为获取到区域个人 Bits。

Y.2.1.1.9 getZipCode

原型：string getZipCode()
描述：获取区域编码。
参数：无。
返回：String 型，取值：为空则表示未获取到区域编码，非空则为获取区域编码。

Y.3 DCAS_APP模块

DCAS_APP 模块对象说明见表 Y.1。

表 Y.1 DCAS_APP 模块的对象说明

对象名称	说明
JSDCAS.CASDescriptor	CA 描述对象，这个对象用于表达 PMT 或者 CAT 中的 CA 描述符。
JSDCAS.CASEcmEvent	ECM 事件对象，这个对象包含了 ECM 事件的信息。
JSDCAS.CASEmmEvent	ECM 事件对象，这个对象包含了 EMM 事件的信息。
JSDCAS.CASFilter	过滤器对象，此对象表达了过滤带内或带外 EMM 时，所需要的过滤条件。
JSDCAS.CASM	CASM 的全局对象，可以从它访问到 CAS manager 和 controller 对象。
JSDCAS.CASModule	CAS 模块对象接口，由 JS DCAS 应用实现，向平台提供的接口。
JSDCAS.CASModuleManager	CASModuleManager 对象，此对象提供 JSDCAS 应用需要的接口。
JSDCAS.CASPacketEvent	CAS 数据包事件对象，通知 JS DCAS 应用带外 CAS 数据包的事件。
JSDCAS.CASSession	CAS Sesion 对象，平台为每个解扰请求都生成一个 CAS Session 对象。
JSDCAS.CASStatus	CAS 状态对象，通过这个对象，JSDCAS 向平台传递解扰状态。
JSDCAS.TeeController	TEE 控制权对象，JS DCAS 和 TEE 通信的控制器。
JSDCAS.TeeRetVal	TEE 返回对象，对象中包含从 TEE 返回的数据，错误等信息。

Y.3.1 应用接口调用时序

DCAS应用接口基本调用时序见图Y.1。

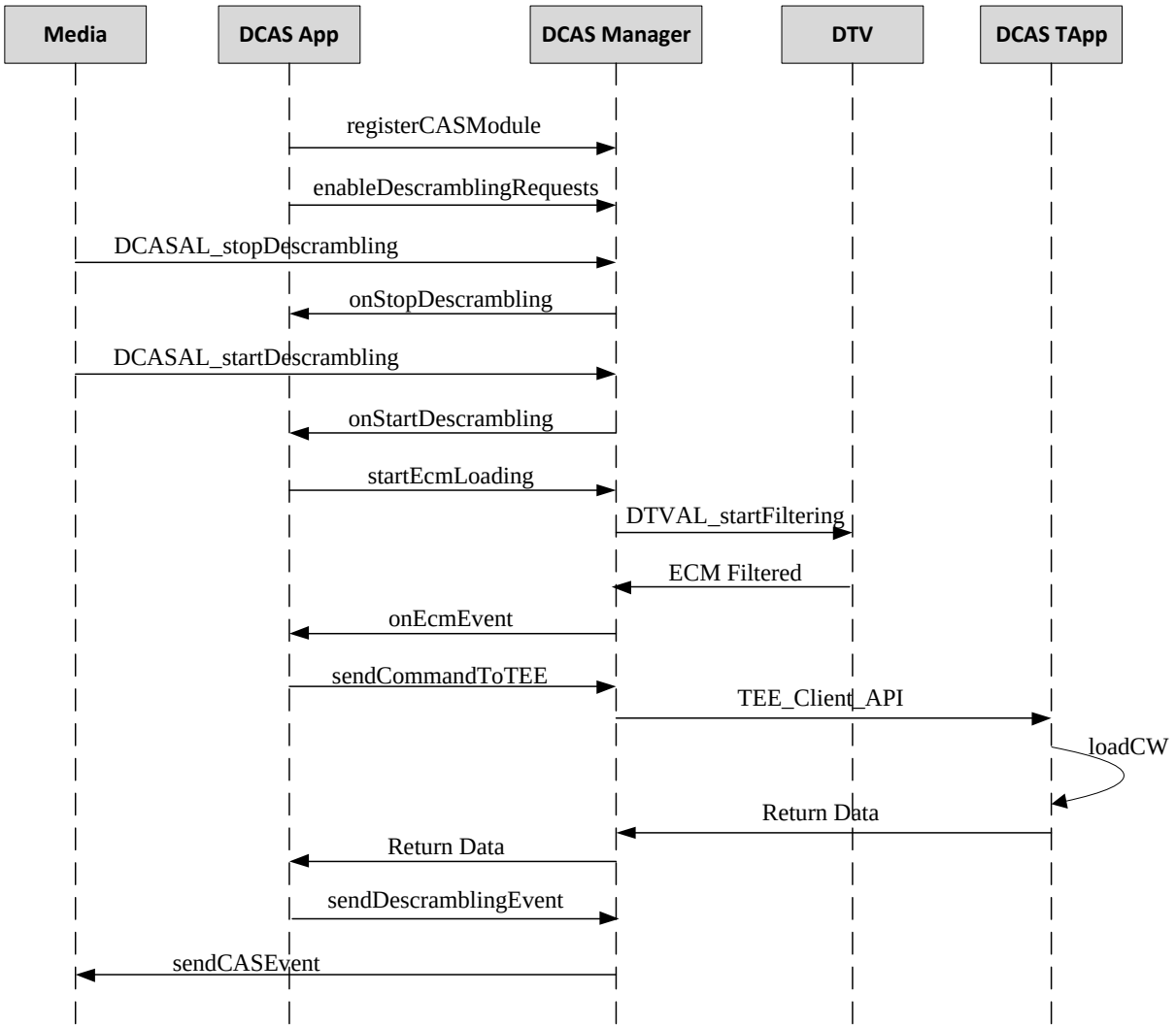


图 Y.1　DCAS 应用接口基本调用时序

Y.3.2　JSDCAS.CASDescriptor对象

这个对象用于表达PMT或者CAT中的CA描述符。

Y.3.2.1　方法

Y.3.2.1.1　getCasId

原型：number getCasId()
描述：本方法由终端软件平台提供，返回CA描述符中的CASID。
参数：无。
返回：number 类型，表示 CASID。

Y.3.2.1.2　getPid

原型：number getPid()
描述：返回 CA 描述符中的 PID。如果描述符来自 CAT，则表示 EMM PID。如果描述符来自 PMT，则表示 ECM PID。

参数：无。
返回：number 类型，返回 ECMPID。

Y.3.2.1.3 getPrivateData

原型：Uint8Array getPrivateData()
描述：返回 CA 描述符中的私有数据,私有数据以 Uint8Array 形式返回。
参数：无。
返回：Uint8Array 类型，表示私有数据。

Y.3.3 JSDCAS.CASEcmEvent对象

这个对象包含了 ECM 事件的信息,ECM 事件是通过 CASModule.onEcmEvent 或者 CASModuleManager.onStartDescrambling 传递给 JS DCAS 应用。它可以表达收到了 ECM 包，或者超时，或者过滤器内部错误。

Y.3.3.1 方法

Y.3.3.1.1 getEcmData

原型：Uint8Array getEcmData()
描述：返回完整的 ECM 数据。如果事件是超时或者内部错误，则返回 null。
参数：无。
返回：Uint8Array 类型，表示 ECM 数据。

Y.3.3.1.2 getError

原型：number getError()
描述：返回内部 Section Filter 过滤器的错误。只用于调试。这个方法只能在事件没有提供任何其他信息的时候调用。
参数：无。
返回：number 类型，错误信息。

Y.3.3.1.3 getTableId

原型：number getTableId()
描述：返回 ECM 包的 Table ID,这个方法只能在事件提供 ECM 数据的情况下调用。
参数：无。
返回：number 类型，表示 TableID。

Y.3.3.1.4 isTimeout

原型：boolean isTimeout()
描述：返回在获取第一个 ECM 包时是否超时。超时的时长可以在 CASModuleManager.enableDescramblingRequests 当中设定, 这个方法只能在事件没有提供 ECM 数据的情况下调用。
参数：无。
返回：boolean 类型，ture-超时；false-没有超时。

Y.3.4 JSDCAS.CASEmmEvent对象

这个对象包含了 EMM 事件的信息,EMM 事件是通过 CASModule.onInBandEmmEvent 传递给 JS DCAS 应用的。收到这个事件可能是因为收到了 CAT，或者任意 EMM 包，或者内部过滤器错误。

Y.3.4.1 方法

Y.3.4.1.1 getEmmData

原型：Uint8Array getEmmData()
描述：返回完整的EMM数据。如果事件是由CAT更新引起，或者由于内部错误引起，则返回null。
参数：无。
返回：Uint8Array 类型，EMM 数据。

Y.3.4.1.2 getError

原型：number getError()
描述：返回内部 Section Filter 过滤器的错误。只用于调试。这个方法只能在事件没有提供任何其他信息的时候调用。
参数：无。
返回：number 类型，错误信息。

Y.3.4.1.3 getTableId

原型：number getTableId()
描述：返回EMM包的Table ID。
参数：无。
返回：number 类型，表示 TableID。

Y.3.4.1.4 isCatUpdateNotification

原型：boolean isCatUpdateNotification()
描述：返回事件是否是由CAT更新引起的,如果是由CAT更新引起的，则EMM数据应该是空。
参数：无。
返回：boolean 类型。
——ture - 事件由CAT更新引起;
——false - 事件由任意EMM包引起，或者由内部错误引起。

Y.3.5 JSDCAS.CASFilter对象

这个对象用于表达过滤带内或带外 EMM 时，所需要的 Section Filter 过滤条件。平台应该只在数据包匹配过滤条件的时候才调用 CAS Module。对于那些不符合条件的数据包，应该由平台丢弃，JS DCAS 应用也不应该被调用。CASFilter 对象(或者对象数组)可以通过 CASModuleManager.startCasPacketLoading 和 CASModuleManager.startInbandEmmLoading 设定到平台。过滤条件包括：

a) 以字节为单位的偏移量。偏移量之前的数据将被忽略，不参与比较。
b) 用于和平台收到的数据进行比较的值，以 Bitmap 表示。
c) 用于表达哪些位需要参与比较，对于 Bitmap 中设定为 0 的位，不需要进行比较。

Y.3.5.1 方法

Y.3.5.1.1 getBitmapMask

原型：Uint8Array getBitmapMask()
描述：返回过滤掩码。
参数：无。
返回：Uint8Array 类型，返回过滤掩码。

Y.3.5.1.2 getBitmapValue

原型：Uint8Array getBitmapValue()
描述：返回用于比较的bitmap值。
参数：无。
返回：Uint8Array 类型，返回 bitmap 值。

Y.3.5.1.3 getOffset

原型：number getOffset()
描述：返回偏移量（单位为字节）。
参数：无。
返回：number 类型，表示偏移量。

Y.3.6 JSDCAS.CASM对象

CASM 是一个全局对象，可以从它访问到所有的 CAS Manager 和 Controller 对象。

Y.3.6.1 方法

Y.3.6.1.1 getCASModuleManager

原型：JSDCAS.CASModuleManager getCASModuleManager()
描述：返回CASModuleManager对象实例。
参数：无。
返回：CASModuleManager对象。

Y.3.6.1.2 getTeeController

原型：JSDCAS.TeeController getTeeController()
描述：返回TEEController对象实例。
参数：无。
返回：TeeController对象。

Y.3.7 JSDCAS.CASModule对象

CASModule 是一个 CAS 模块对象的接口，应该由 JS DCAS 应用来实现，并注册到平台的 CAS Module Manager 中以便接收解扰请求，ECM，EMM 或者其他 JS DCAS 应用关心的任意元数据。EMM 可以通过带内或者带外方式获得。由端到端的设计以及平台和设备的能力决定。

Y.3.7.1 方法

Y.3.7.1.1 getCasId

原型：number getCasId()

描述：返回CAS模块的唯一CASID。这个方法必须在调用CASModuleManager.registerCASModule之前实现。返回的值是预期要在CAT或者PMT中CA描述符中出现的值。

参数：无。

返回：number类型，表示CASID。

Y.3.7.1.2 onCasPacketEvent

原型：onCasPacketEvent(casPacketEvent)

描述：平台在收到带外EMM（或其他带外CAS数据包）的时候调用这个方法，参考CASModuleManager.startCasPacketLoading。

参数：casPacketEvent – CASPacketEvent对象，包含带外EMM或其他带外CAS数据包的CASPacketEveng对象实例。

Y.3.7.1.3 onEcmEvent

原型：onEcmEvent(casSession,ecmEvent)

描述：在平台过滤到了新的ECM包后，调用JS DCAS应用的这个方法。在fast模式下，平台在将ECM中的CW设定到K-LAD之后，调用这个方法。

参数：casSession – CASSession对象，从CASModule.onStartDescrambling获得的CAS Session对象。
CASEcmEvent ecmEvent-ECM事件对象。

Y.3.7.1.4 onInbandEmmEvent

原型：onInbandEmmEvent(casSessionForEMM,emmEvent)

描述：平台在收到了CAT更新或者带内EMM的时候调用这个方法。参考CASModuleManager.startInbandEmmLoading。

参数：casSessionForEMM – CASSession对象，特殊的CAS Session，这个session是关联到CAT所在的TS，同时也包含了CAT中的CA描述符(不是PMT中的CA描述符)。平台应该专门为此创建一个CAS Session，注意，这个CAS Session对象也许只有部分字段有效，注意：如果CAT中没有指定CAS的CA描述符，或者终端离开了当前频点，CAT更新事件还是会送到JS DCAS应用，但是CAS Session为null。

emmEvent – CASEmmEvent对象，包含了CAT更新或者EMM，或者其他错误的CASEmmEvent对象实例。

Y.3.7.1.5 onStartDescrambling

原型：onStartDescrambling(casSession,firstEcmEvent)

描述：这个方法由平台在有新的解扰请求时调用。 通常发生在平台开始播放一个新的加扰频道的时候。 JS DCAS应用只会在调用了CASModuleManager.enableDescramblingRequests之后才收到这个解扰请求。在auto-load的模式中，平台会自动开始过滤第一个ECM，并在收到ECM后调用这个方法。 这种情况下，JS DCAS应用不必再调用CASModuleManager.startEcmLoading。如果设备有多个Tuner，这个方法可能会被同时调用多次，每次对应一个播放解扰请求。每个请求都会有不同的CAS Session。另外，如果需要解扰的service的基础流加扰方式不同，这个方法也可能被调用多次。同样，每次调用也都有不同的CAS Session。

参数：casSession – CASSession对象，平台为特定解扰请求生成的CAS Session对象。每个对象有唯一的session ID，以及关于service以及基础流的所有信息。还会包含PMT中对应这个CAS模块的CA描述符。

firstEcmEvent – CASEcmEvent对象，在auto-load模式下，平台收到的第一个ECM包（或者超时，错误）。如果不是auto-load模式，则为空。

Y.3.7.1.6 onStopDescrambling

原型：onStopDescrambling(casSession)

描述：平台在停止播放当前频道的时候调用这个方法，通知JS DCAS应用停止解扰。

参数：casSession – CASSession对象，在CASModule.onStartDescrambling中获得的CAS Session。

Y.3.8 JSDCAS.CASModuleManager对象

JS DCAS 应用通过这个 CAS Module Manager 对象来实现接收解扰请求，ECM，EMM 以及报告 CAS 解扰状态。 JS DCAS 应用应该实现 CAS Module 对象，然后注册到 CAS Module Manager。

Y.3.8.1 方法

Y.3.8.1.1 disableDescramblingRequests

原型：number disableDescrambingRequests(casModule)

描述：JSDCAS应用调用这个方法停止接收解扰请求。 这个方法很少有机会被调用。比如在应用需要重新配置解扰请求参数的时候，需要先调用这个方法暂停接收，然后再重新启动接收。或者应用希望关闭自己。重新调用CASManager.enableDescramblingRequests可以重新开启接收。

参数：casModule – CASModule对象，CAS模块实例。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数;

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误。

Y.3.8.1.2 enableDescramblingRequests

原型：number enableDescramblingRequests(casModule,firstEcmTimeout,
autoLoadFirstEcm,isFastMode,ecmTableIds)

描述：JS DCAS应用通过调用这个方法启动接收解扰请求。通过不同的参数，可以配置平台CAS Manager和CAS模块之间不同的工作方式。这个方法通常只在CAS模块注册后调用，因为通常JS DCAS应用不会改变这个工作方式。如果JS DCAS应用要改变这个工作方式，需要先调用CASModuleManager.disableDescramblingRequests，然后重新调用这个方法。

参数：casModule - CASModule对象，CAS模块实例;

firstEcmTimeout – number类型，单位毫秒，平台等待第一个ECM的最长时间。如果超时，CAS模块会通过onEcmEvent调用或者onSTartDescrambling调用收到CASEcmEvent;

autoLoadFirstEcm – boolean类型,指定是否auto-load模式。Auto-load模式是指平台在JS DCAS应用调用这个方法后,自动开始过滤第一个ECM,而不必等待JS DCAS应用调用startEcmLoading。

isFastMode –boolean类型，快速模式（暂时仅是占位，并没有实际意义）;

ecmTableIds – Array类型，如果JS DCAS应用希望指定ECM的tableID。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数 ;

CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 平台未实现特定模式；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误 。

Y.3.8.1.3 fetchDataFromCasHeadend

原型：number fetchDataFromCasHeadend(casModule,inputData,casHeURI)
描述：JS DCAS应用通过调用这个方法，经过平台，通过GPRS，或者未来可能的其他手段，从头端获取数据。
参数：casModule –CASModule对象，CAS模块实例。
inputData – Uint8Array类型，将要发送给头端的数据。
casHeURI – String类型，头端服务的URI。
返回：number类型。
成功 – 头端返回的数据。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持；
CASModuleManager.ACTION_ERROR_NETWORK – 网络错误。

Y.3.8.1.4 registerCASModule

原型：number registerCASModule(vendorId,casModule,networkPriority,applicationContext)
描述：JS DCAS应用通过调用这个方法来将自己注册到平台的CAS Module Manager中。
参数：vendorId – number类型，CAS的Vendor Id。每个CAS厂家都有唯一的特殊ID。
casModule – CASModule对象，要注册的CAS模块实例。
networkPriority – number类型，如果CASModuleManager允许注册超过一个CAS模块， 这个可选参数表达了多个模块之间的优先级，具体的值由运营商决定。 绝对值越大，表示优先级越高，例如3的优先级高于2。 在知道了优先级后，平台在遇到PMT中有多个CA描述符的情况，应该按照优先级，发送解扰请求给拥有最高优先级的CAS模块。如果运营商并没有设定优先级，每个JS DCAS应用必须以0作为参数。这种情况下，哪个CAS模块可以收到解扰请求，是由平台的实现决定的。
applicationContext – 平台相关的应用参数。通常这个参数是由平台在初始化的时候告诉应用的。这个参数的使用情况是项目相关的。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_SECURITY – 调用者没有权限访问CASModuleManager；
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效的参数。

Y.3.8.1.5 removeCASModule

原型：number removeCASModule(vendorId,casModule,applicationContext)
描述：JS DCAS应用调用这个方法将一个CAS模块从CAS Module Manager中删除。 这个方法很少会被调用到。比如在应用希望换一个CAS ID或者在应用关闭自己之前。
参数：vendorId – number类型，CAS的VendorId。每个CAS厂家都有唯一的特殊ID。
casModule – CASModule对象，要注册的CAS模块实例。

applicationContext – 平台相关的应用参数。通常这个参数是由平台在初始化的时候告诉应用的。这个参数的使用情况是项目相关的。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；

CASModuleManager.ACTION_ERROR_SECURITY – 调用者没有权限。

Y.3.8.1.6 sendCommandToSTB

原型：number sendCommandToSTB(casModule,inputData)

描述：数据通道函数，JS DCAS应用通过调用这个方法，把数据发送给DCAS Manager，DCASManager把命令转发给相应模块处理，这些命令是包括OSD、升级触发、指纹、应急广播、收视调查等，这些命令由BOSS发送，DCAS仅负责转发，作为数据通道使用。

参数：casModule – CASModule对象，要注册的CAS模块实例。

inputData – Uint8Array类型，将要发送给DCAS管理器的数据。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；

CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持；

CASModuleManager.ACTION_ERROR_NETWORK – 网络错误 。

Y.3.8.1.7 sendDataToHeadend

原型：number sendDataToHeadend(casModule,inputData)

描述：JS DCAS应用通过调用这个方法，经过平台，通过GPRS，或者未来可能的其他手段，发送数据给头端。

参数：casModule – CASModule对象，要注册的CAS模块实例。

inputData – Uint8Array类型，将要发送给头端的数据。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；

CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持；

CASModuleManager.ACTION_ERROR_NETWORK – 网络错误。

Y.3.8.1.8 sendDescramblingEvent

原型：number sendDescrambingEvent(casModule,casSession,casStatus)

描述：JS DCAS应用调用这个方法报告CAS状态。CAS状态包括解扰成功与否。每次状态变化，JS DCAS都应该报告状态。

参数：casModule – CASModule对象，CAS模块实例。

casSession – CASSession对象，从CASModule.onSTartDescrambling获得的CASSession。
casStatus – CASStatus对象，JSDCAS应用生成的CASStatus对象。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.9 sendFreeTextOSD

原型：number sendFreeTextOSD(casModule,inputData,flags)
描述：JS DCAS应用通过调用这个方法转发接收到的文本信息给平台 平台可能将这个文本信息转发给UI应用或者自己处理。
参数：casModule – CASModule对象，CAS模块实例。
inputData – Uint8Array类型，文本信息。
flags – ArrayBuffer类型，用于显示方式格式等的额外信息，项目相关。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.10 setCCIBits

原型：number setCCIBits(casModule,casSession,cciBits)
描述：设定CCI(Copy Control Information)数据位。
参数：casModule – CASModule对象，CAS模块实例。
casSession – CASSession对象，从CASModule.onSTartDescrambling获得的CAS Session。
cciBits – number类型，CCI数据位。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.11 setData

原型：number setData(casModule, propertyId, propertyType, propertyValue)
描述：DCAS APP设置属性值给平台，包含BouquetID、激活状态、CAS信息、北斗信息、ChipID、HSMID、CASVenderID、区域码、CA版本等。
参数：casModule – CASModule对象，CAS模块实例；
propertyId – number类型，属性ID，见JSDCAS.CASModuleManager.PROP_ID_xxx；
propertyType – number类型，属性类型，见JSDCAS.CASModuleManager.PROP_TYPE_xxx；
propertyValue – number类型，属性值。
返回：number类型。

成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.12 setPinCode

原型：number setPinCode(casModule, pinCode)
描述：通知平台重置PIN。
参数： casModule – CASModule对象，CAS模块实例。
pinCode – number类型，PIN码。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.13 setServiceListFilter

原型：number setServiceListFilter(casModule,filterData)
描述：设置服务列表过滤条件 过滤条件的定义是平台相关的。
参数：casModule – CASModule对象，CAS模块实例。
filterData – number类型，过滤条件。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.14 startCasPacketLoading

原型：number startCasPacketLoading(casModule,cableModemFilter,sourceURL,casFilter)
描述：JS DCAS应用调用这个方法启动接收带外CAS数据包。 CAS数据包可以是EMM或者其他带外元数据。 接收方法由设备硬件，平台，网络等决定。对于带有Cable Modem的设备，可以通过ADSG或者BDSG协议接收，也可以通过IP over Cable来加入多播地址来接收。 IPTV设备，可以通过以太网或者Wifi来加入多播地址来接收。也可以通过一个本地的UDPsocket来接收。不论哪种情况，JS DCAS应用都可以在收到CAS数据包的时候通过CASModule.onCasPacketEvent来处理。注意：平台可能实现为同时从多个不同源接收数据包。这种情况下，这个方法可能会从不同URL调用多次。
参数： casModule – CASModule对象，CAS模块实例。
cableModemFilter – number类型，在Cable Modem 和DSG tunnelD的情况下，必须提供一个filter。在非DSG的情况下，这个参数应该为null。
sourceURL – String类型，如果从UDP接收数据包，则需要提供这个参数. 从本地UDP端口接收：“udp://@127.0.0.1:4444” 或 “udp://@localhost:4444”。
casFilter – CASFilter对象，过滤条件,可以是一个CASFilter数组。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；

CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.15 startEcmLoading

原型：number startEcmLoading(casModule,casSession)

描述：JS DCAS应用通过调用这个方法启动接收特定加扰节目的ECM 在收到了解扰请求后调用。注意：在auto-load模式中，并不需要调用这个方法。

参数：casModule – CASModule对象，CAS模块实例。

casSession – CASSession对象，从CASModule.onSTartDescrambling获得的CASSession。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误。

Y.3.8.1.16 startInbandEmmLoading

原型：number startInbandEmmLoading(casModule,emmTableIds,casFilter,includeCatNotifications)

描述：JS DCAS应用调用这个方法来启动接收带内EMM。 如果JS DCAS应用需要，也可以用来接收CAT。当有EMM或者CAT更新的时候，JS DCAS应用通过CASModule.onInbandEmmEvent来接收数据。

参数：casModule – CASModule对象，CAS模块实例。

emmTableIds – Array类型，EMM Table Id数组。

casFilter – CASFilter|Array 类型，CAS过滤器。只有符合条件的数据，平台才会通知到应用，传递一个Filter数组也是可以的。

includeCatNotifications – boolean类型，指定是否希望接收CAT更新通知。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。

失败 – 返回如下错误值：

CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；

CASModuleManager.ACTION_ERROR_DRIVER – 底层错误；

CASModuleManager.ACTION_ERROR_ACTION_NOT_SUPPORTED – 方法不支持。

Y.3.8.1.17 stopCasPacketLoading

原型：number stopCasPacketLoading(casModule,cableModemFilter,sourceURL)

描述：JS DCAS应用调用这个方法停止接收带外CAS数据包。

参数： casModule – CASModule对象，CAS模块实例。

cableModemFilter – number|string类型，Cable Modem需要。

sourceURL – String类型，通过UDP接收时需要。

返回：number类型。

成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误。

Y.3.8.1.18 stopEcmLoading

原型： number stopEcmLoading(casModule,casSession)
描述：JS DCAS应用通过调用这个方法停止接收ECM。 JS DCAS应用极少有机会调用这个方法。如果希望重新启动ECM接收，则需要重新调用CASManager.startEcmLoading。
参数：casModule – CASModule对象，CAS模块实例。
casSession – CASSession对象，从CASModule.onSTartDescrambling获得的CASSession。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误。

Y.3.8.1.19 stopInbandEmmLoading

原型：number stopInbandEmmLoading(casModule)
描述： JS DCAS应用通过调用这个方法停止接收带内EMM。 这个方法很少需要调用。重新调用CASManager.startInbandEmmLoading可以重新开始接收。
参数：casModule – CASModule对象，CAS模块实例。
返回：number类型。
成功 – CASModuleManager.ACTION_OK。
失败 – 返回如下错误值：
CASModuleManager.ACTION_ERROR_INVALID_PARAMETERS – 无效参数；
CASModuleManager.ACTION_ERROR_DRIVER – 底层错误。

Y.3.9 JSDCAS.CASPacketEvent对象

用于通知JS DCAS应用带外CAS数据包的事件。

Y.3.9.1 方法

Y.3.9.1.1 getCableModemFilter

原型：number|string getCableModemFilter()
描述：返回过滤这个包所用的cableModemFilter。 如果没有使用Cable Modem DSG，则返回空。
参数：无。
返回：number|string类型。
ADSG模式 – 返回CAS Tunner ID，数字类型。
BDSG模式 – 返回虚拟MAC地址。

Y.3.9.1.2 getPacketData

原型：Uint8Array getPacketData()
描述：返回数据包中的数据。

参数：无。
返回：Uint8Array 类型。

Y.3.9.1.3 getPacketHeader

原型：Uint8Array getPacketHeader()
描述：返回数据包的包头 包头包含了 IP 地址和 UDP 头。
参数：无。
返回：Uint8Array 类型。

Y.3.9.1.4 getSourceURL

原型：string getSourceURL()
描述：返回通过 UDP 接收 CAS 数据包的源地址。
参数：无。
返回：string 类型，源地址字符串。

Y.3.10 JSDCAS.CASSession对象

平台为每个解扰请求都生成一个CAS Session对象。它包含了唯一的session ID，以及所有关于播放节目的信息，还包含了PMT中与本次解扰相关的CA描述符。对每个解扰请求，JS DCAS应用通过CASModule.onStartDescrambling获得CAS Session。CASSession对象还可以用于接收CAT更新消息的场景，在此场景中，CASSession对象只会有部分字段有效。

Y.3.10.1 方法

Y.3.10.1.1 GetCasDescriptor

原型：CASDescriptor getCasDescriptor()
描述：返回 CA 描述符，可能是来自 PMT，也可能来自 CAT。
参数：无。
返回：CASDescriptor 对象，CA 描述符对象实例。

Y.3.10.1.2 getChannelNumber

原型：number getChannelNumber()
描述：返回频道号。这个方法是可选的，尤其是对于那些不能确定频道号的平台，可以返回 0。
参数：无。
返回：number 类型，频道号。

Y.3.10.1.3 getNetworkId

原型：number getNetworkId()
描述：返回原始网络 ID。这个方法是可选的，如果平台无法获得，可以返回 0。
参数：无。
返回：number 类型，原始网络 ID 。

Y.3.10.1.4 getOperationType

原型：number GetOperationType()

描述：返回操作类型。
参数：无。，
返回：number类型，操作类型值：
CASSession.OPERATION_TYPE_PRESENTATION；
CASSession.OPERATION_TYPE_RECORDING；
CASSession.OPERATION_TYPE_BUFFERING；
CASSession.OPERATION_TYPE_SECOND_DEVICE 。

Y.3.10.1.5 getProgramNumber

原型：number getProgramNumber()
描述：返回节目号。
参数：无。
返回：number 类型，节目号。

Y.3.10.1.6 getServiceIdentifier

原型：number getServiceIdentifier()
描述：返回正在解扰的服务标识符，此标识符是一个值或者一个对象；
参数：无。
返回：number 类型，服务标识符。

Y.3.10.1.7 getSessionId

原型：number getSessionId()
描述：返回 SessionID 。
参数：无。
返回：number 类型，表示 SessionID。

Y.3.10.1.8 getStreamPath

原型：Uint8Array getStreamPath()
描述：返回StreamPath 数据。
参数：无。
返回：Uint8Array类型，表示StreamPath数据。

Y.3.10.1.9 getStreamPIDs

原型：Array getStreamPIDs()
描述：返回Stream PIDs列表 。
参数：无。
返回：Array类型，PID列表。

Y.3.10.1.10 getStreamTypes

原型：Array getStreamTypes()
描述：返回StreamTypes列表。
参数：无。
返回：Array类型，Streamtypes列表。

Y.3.10.1.11 getTransmitterScrambingMode

原型：number getTransmitterScrambingMode()
描述：返回加扰模式值。
参数：无。
返回：number类型，加扰模式。

Y.3.10.1.12 getTransportStreamId

原型：number getTransportStreamId()
描述：返回正在解扰的TSID 。
参数：无。
返回：number类型，表示TSID。

Y.3.10.1.13 getTunerId

原型：number getTunerId()
描述：返回正在解扰节目使用的TunerID 。
参数：无。
返回：number类型，表示TunerID。

Y.3.11 JSDCAS.CASStatus对象

Y.3.11.1 JSDCAS.CASStatus对象概述

JS DCAS应用通过这个对象，向平台传递解扰状态。每次解扰状态有变化，JS DCAS应用应该调用CASModuleManager.sendDescramblingEvent通知平台。平台接收到这个状态变化后，可以自己处理，也可以转发给UI应用。UI应用可以简单的弹出OSD通知用户解扰成功或者失败，也可以通过解析CASStatus对象中的附加信息显示具体因为什么原因失败。这些额外的信息格式是项目相关的。如果JS DCAS应用没有附加额外信息，UI应用获得CASStatus对象中的Token也可以使用这个Token通过IPC或者其他平台提供的手段与JS DCAS应用通信获得更多信息。

Y.3.11.2 方法

Y.3.11.2.1 getCasToken

原型：number getCasToken()
描述：返回CAS Token。 如果平台将CASStatus的信息转发给UI应用，UI应用可以使用这个Token向JS DCAS应用发起请求，以获得更详细的状态信息。 请求的方式，是平台决定的，比如IPC。
参数：无。
返回：number类型，返回Token。

Y.3.11.2.2 getMajorContentProblem

原型：number getMajorContentProblem()
描述：返回不能观看节目的主要错误值。
参数：无。
返回：number类型，错误信息。

Y.3.11.2.3 getStatusData

原型：ArrayBuffer getStatusData()
描述：返回解扰状态扩展数据。通过扩展数据，UI应用可以显示更详细的关于解扰状态的信息。
参数：无。
返回：ArrayBuffer类型，表示扩展数据，如果没有扩展数据可以提供，应该返回null。

Y.3.11.2.4 isSuccess

原型：boolean isSuccess()
描述：返回解扰成功与否。
参数：无。
返回：boolean类型，ture-成功，false-失败 。

Y.3.12 JSDCAS.TeeController对象

JS DCAS和TEE通信的控制器。

Y.3.12.1 方法

Y.3.12.1.1 sendCommandToTEE

原型：TeeRetVal sendCommandToTEE(teeAppUUID,commandId,inputData,applicationContext)
描述：JS DCAS应用通过这个方法向TEE中运行的TA发送命令。
参数：

teeAppUUID - Uint8Array类型，TA的UUID，16字节。每个CA厂家都有不同的ID。
commandId - number类型，TEE通信中的Command ID。由各个CA厂家自己定义。
inputData - Uint8Array类型，发送给TA的数据。
applicationContext - 应用上下文，平台相关。通常在初始化的时候由平台提供给应用。

返回：TeeRetVal对象，这个对象包含从TA返回的数据、错误等信息。

Y.3.13 JSDCAS.TeeRetVal类

这个对象由TeeController.sendCommandToTEE返回。 对象中包含从TEE返回的数据、错误等信息。

Y.3.13.1 方法

Y.3.13.1.1 getOriginCode

原型：number getOriginCode()
描述：返回origin code。
参数：无。
返回：number类型，表示origincode。

Y.3.13.1.2 getResponseData

原型：Uint8Array getResponseData()
描述：得到返回自TA的数据。
参数：无。
返回：Uint8Array类型，TA返回的数据。对于某些命令可以为null。如果调用或者通信发生错误，也返回null。

Y.3.13.1.3 getReturnCode

原型：number getReturnCode()
描述：返回码。
参数：无。
返回：number类型，表示返回码。

中 华 人 民 共 和 国

广播电视行业标准

智能电视操作系统

第 3 部分：应用程序编程接口

GY/T 303.3－2018

*

国家新闻出版广电总局广播电视规划院出版发行

责任编辑：王佳梅

查询网址：www.abp2003.cn

北京复兴门外大街二号

联系电话：（010）86093424　　86092923

邮政编码：100866