

I n t e r n a t i o n a l T e l e c o m m u n i c a t i o n U n i o n

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

H.764

(11/2019)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

IPTV multimedia services and applications for IPTV –
IPTV multimedia application frameworks

IPTV services enhanced script language

Recommendation ITU-T H.764

ITU-T

ITU-T H-SERIES RECOMMENDATIONS
AUDIOVISUAL AND MULTIMEDIA SYSTEMS

CHARACTERISTICS OF VISUAL TELEPHONE SYSTEMS	H.100–H.199
INFRASTRUCTURE OF AUDIOVISUAL SERVICES	
General	H.200–H.219
Transmission multiplexing and synchronization	H.220–H.229
Systems aspects	H.230–H.239
Communication procedures	H.240–H.259
Coding of moving video	H.260–H.279
Related systems aspects	H.280–H.299
Systems and terminal equipment for audiovisual services	H.300–H.349
Directory services architecture for audiovisual and multimedia services	H.350–H.359
Quality of service architecture for audiovisual and multimedia services	H.360–H.369
Telepresence, immersive environments, virtual and extended reality	H.420–H.439
Supplementary services for multimedia	H.450–H.499
MOBILITY AND COLLABORATION PROCEDURES	
Overview of Mobility and Collaboration, definitions, protocols and procedures	H.500–H.509
Mobility for H-Series multimedia systems and services	H.510–H.519
Mobile multimedia collaboration applications and services	H.520–H.529
Security for mobile multimedia systems and services	H.530–H.539
Security for mobile multimedia collaboration applications and services	H.540–H.549
VEHICULAR GATEWAYS AND INTELLIGENT TRANSPORTATION SYSTEMS (ITS)	
Architecture for vehicular gateways	H.550–H.559
Vehicular gateway interfaces	H.560–H.569
BROADBAND, TRIPLE-PLAY AND ADVANCED MULTIMEDIA SERVICES	
Broadband multimedia services over VDSL	H.610–H.619
Advanced multimedia services and applications	H.620–H.629
Ubiquitous sensor network applications and Internet of Things	H.640–H.649
IPTV MULTIMEDIA SERVICES AND APPLICATIONS FOR IPTV	
General aspects	H.700–H.719
IPTV terminal devices	H.720–H.729
IPTV middleware	H.730–H.739
IPTV application event handling	H.740–H.749
IPTV metadata	H.750–H.759
IPTV multimedia application frameworks	H.760–H.769
IPTV service discovery up to consumption	H.770–H.779
Digital Signage	H.780–H.789
E-HEALTH MULTIMEDIA SYSTEMS, SERVICES AND APPLICATIONS	
Personal health systems	H.810–H.819
Interoperability compliance testing of personal health systems (HRN, PAN, LAN, TAN and WAN)	H.820–H.859
Multimedia e-health data exchange services	H.860–H.869
Safe listening	H.870–H.879

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T H.764

IPTV services enhanced script language

Summary

Recommendation ITU-T H.764 describes an object-oriented programming language called "Internet protocol television services enhanced script language (IPTV SESL)", which is based on LIME-Script of Recommendation ITU-T H.762, and includes enhanced functionalities. The language is used in IPTV services for performing computations and manipulating computational objects within an IPTV terminal device environment. There are two primary implementations of this Recommendation, the "Core script profile" and the "Extended script profile". The core script profile is conformant with the LIME-Script of Recommendation ITU-T H.762 and is used for the objects for client-side computation. The extended script profile is necessary for the functional extensions required for IPTV services.

This revision corrects or clarifies the definitions of some properties and methods in extended script profile and makes some editorial modifications.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T H.764	2012-06-29	16	11.1002/1000/11667
2.0	ITU-T H.764 V2	2019-11-29	16	11.1002/1000/14124

Keywords

IPTV SESL, multimedia application framework (MAFR), script language, web-based.

* To access the Recommendation, type the URL <http://handle.itu.int/> in the address field of your web browser, followed by the Recommendation's unique ID. For example, <http://handle.itu.int/11.1002/1000/11830-en>.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at <http://www.itu.int/ITU-T/ipr/>.

© ITU 2020

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

Page

1	Scope	1
2	References.....	1
3	Definitions	2
3.1	Terms defined elsewhere	2
3.2	Terms defined in this Recommendation.....	3
4	Abbreviations and acronyms	3
5	Conventions	4
6	Overview	4
7	Core script profile	5
7.1	Global	5
7.2	Object	5
7.3	Function.....	6
7.4	Array.....	6
7.5	String	7
7.6	Boolean.....	8
7.7	Number	8
7.8	Date.....	9
8	Extended script profile.....	11
8.1	MediaController object.....	12
8.2	Event object	33
8.3	Service object	36
8.4	XMLHttpRequest object	43
	Annex A – Data types	47
	Annex B – Content Metadata used by MediaController object	48
	Appendix I – Remote control key code	50
	Appendix II – Informative text on extended script profile	52
	II.1 MediaController object.....	52
	II.2 Event object	52
	II.3 Service object	62
	Bibliography.....	63

Recommendation ITU-T H.764

IPTV services enhanced script language

1 Scope

This Recommendation describes a script language for IPTV services, Internet protocol television services enhanced script language (IPTV SESL). This language is used to provide interoperability and harmonization among IPTV multimedia application frameworks. It is foreseeable that a combination of different standard multimedia application frameworks will be used to provide standard IPTV services globally. This Recommendation describes IPTV SESL, as one of these standard multimedia application frameworks for providing the interoperable use of IPTV services. The core script profile, as well as enhanced functionalities for IPTV services, are given.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

- | | |
|-----------------|--|
| [ITU-T H.721] | Recommendation ITU-T H.721 (2015), <i>IPTV terminal devices: Basic model</i> . |
| [ITU-T H.722] | Recommendation ITU-T H.722 (2014), <i>IPTV terminal device: Full-fledged model</i> . |
| [ITU-T H.730] | Recommendation ITU-T H.730 (2012), <i>Web-based terminal middleware for IPTV services</i> . |
| [ITU-T H.750] | Recommendation ITU-T H.750 (2008), <i>High-level specification of metadata for IPTV services</i> . |
| [ITU-T H.762] | Recommendation ITU-T H.762 (2011), <i>Lightweight interactive multimedia environment (LIME) for IPTV services</i> . |
| [ITU-T Y.1901] | Recommendation ITU-T Y.1901 (2009), <i>Requirements for the support of IPTV services</i> . |
| [IETF RFC 2326] | IETF RFC 2326 (1998), <i>Real time Streaming protocol (RTSP)</i> . |
| [ISO/IEC 8601] | ISO/IEC 8601:2004, <i>Data Elements and Interchange Formats – Information Interchange – Representation of dates and times</i> . |
| [ISO/IEC 16262] | ISO/IEC 16262:2011, <i>Information technology – Programming languages, their environments and system software interfaces – ECMAScript language specification</i> . |

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 ECMAScript [ISO/IEC 16262]: The programming language defined by ISO/IEC 16262:2011.

NOTE – Edition 5.1 of ECMAScript [b-ECMA-262] is fully aligned with [ISO/IEC 16262] (third edition).

3.1.2 electronic programme guide (EPG) [ITU-T Y.1901]: A structured set of data, intended to provide information on available content that may be accessed by end users.

NOTE – In some traditional broadcast services, EPG is defined as an on-screen guide used to display information on scheduled live broadcast television programmes, allowing a viewer to navigate, select and discover programmes by time, title, channel, genre. This traditional definition does not cover "catalogues" for on-demand and download services (sometimes called an electronic content guide (ECG) and broadband content guide (BCG) and bidirectional interactive services (sometimes called an interactive programme guide (IPG)) for end-user interaction with a server or head-end.

3.1.3 end system [ITU-T Y.1901]: A single or set of consumer devices that support IPTV services (e.g., delivery network gateway, display).

3.1.4 end user [ITU-T Y.1910]: The actual user of the products or services.

NOTE – The end user consumes the product or service. An end user can optionally be a subscriber.

3.1.5 Internet protocol television (IPTV) [ITU-T Y.1901]: Multimedia services such as television/video/audio/text/graphics/data delivered over IP-based networks that are managed to support the required level of QoS/QoE, security, interactivity and reliability.

3.1.6 IPTV terminal device [ITU-T Y.1901]: A terminal device which has ITF functionality, e.g., a STB.

3.1.7 IPTV terminal function (ITF) [ITU-T Y.1901]: The end-user function(s) associated with a) receiving and responding to network control channel messages regarding session set-up, maintenance, and teardown, and b) receiving the content of IP transport from the network and rendering.

3.1.8 linear TV [ITU-T Y.1901]: A television service in which a continuous stream flows in real time from the service provider to the terminal device and where the user cannot control the temporal order in which contents are viewed.

3.1.9 metadata [ITU-T Y.1901]: Structured, encoded data that describe characteristics of information-bearing entities to aid in the identification, discovery, assessment and management of the described entities.

NOTE – EPG metadata has many applications and may vary in depth from merely identifying the content package title or information to populate an EPG to providing a complete index of different scenes in a movie or providing business rules detailing how the content package may be displayed, copied or sold.

3.1.10 middleware [ITU-T Y.1901]: A layer of software between applications and resources, which consists of a set of service enablers that allow multiple functionalities running on one or more devices in an IPTV system to interact across a network.

3.1.11 network personal video recorder (nPVR) [ITU-T Y.1901]: This is the same as PVR except that the recording device is located at the service provider's premises.

3.1.12 personal video record (PVR) [ITU-T Y.1901]: An end-user controlled device that records, stores and plays back multimedia content. PVR is also known as personal digital recorder (PDR).

3.1.13 SCP [ITU-T Y.1901]: A combination of service protection and content protection.

3.1.14 terminal device (TD) [ITU-T Y.1901]: An end-user device which typically presents and/or processes the content, such as a personal computer, a computer peripheral, a mobile device, a TV set, a monitor, a VoIP terminal or an audiovisual media player.

3.1.15 time shifting [ITU-T Y.1901]: A function which allows playback of content after its initial transmission.

3.1.16 trick mode functionality [ITU-T Y.1901]: The ability to pause, rewind or forward stored content.

3.1.17 TV with trick mode [ITU-T Y.1901]: TV service with trick mode functionality.

3.1.18 user agent [b-W3C WebArch]: One type of Web agent; a piece of software acting on behalf of a person.

3.1.19 video-on-demand (VoD) [ITU-T Y.1901]: A service in which the end user can, on demand, select and view video content and where the end user can control the temporal order in which the video content is viewed (e.g., the ability to start the viewing, pause, fast forward, rewind, etc.).

NOTE – The viewing may occur sometime after the selection of the video content.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 application: A functional implementation realized as software running in one or spread over several interplaying hardware entities.

3.2.2 resource-constrained TD: An IPTV terminal device with a basic set of capabilities, such as those given in [ITU-T H.721].

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

DOM	Document Object Module
DRM	Digital Rights Management
FFW	Fast Forward
FRW	Fast Rewind
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
HTTPS	Hypertext Transport Protocol over Secure socket layer
IGMP	Internet Group Management Protocol
JSON	JavaScript Object Notation
LIME	Lightweight Interactive Multimedia Environment
NPT	Normal Play Time
PAL	Phase Alternating Line
PE	Presentation Engine
RTSP	Real Time Session Protocol
SCP	Service and Content Protection
SP	Service Provider
STB	Set-Top Box

TD	Terminal Device
TV	Television
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VoD	Video on Demand
XML	extensible Markup Language

5 Conventions

The following conventions apply to operational restrictions on attributes and properties:

- R/W The corresponding attribute or property can be read and written. An IPTV terminal device designed for basic services should support R/W of the corresponding attribute or property in the content.
- R The corresponding attribute can be read but cannot be written. An IPTV terminal device designed for basic services should support R/W of the corresponding attribute or property in the content. Write operations to this item may be ignored.
- M M indicates "Mandatory". The corresponding attribute or method must be implemented by an IPTV terminal device supporting this Recommendation.
- O O indicates "Optional". The corresponding attribute or method can either be implemented or ignored by an IPTV terminal device supporting this Recommendation.

6 Overview

The IPTV services enhanced script language (IPTV SESL) is based on LIME-Script [ITU-T H.762] with enhanced functionalities. IPTV SESL is an object-oriented programming language for performing computations and manipulating computational objects within a host environment. It was originally designed to be a web scripting language, providing a mechanism to enliven webpages in browsers and to perform server computation as part of web-based, client-server architecture. Together with the server-side scripting, it is possible to distribute computation between the client and the server, while providing a customised user interface for web-based applications.

NOTE – LIME-Script [ITU-T H.762] is a subset of ECMAScript [ISO/IEC 16262]. Only objects, properties and methods that are suitable for IPTV services are referred to in ITU-T H.762 LIME-Script.

In this Recommendation, two primary parts are defined to implement web-based IPTV services:

- The "core script profile", which is designed for resource-constrained IPTV TDs, such as the basic TD model defined in [ITU-T H.721]. This profile is fully conformant with LIME-Script of [ITU-T H.762] and comprises a subset of ECMAScript.
- The profile hereinafter called "extended script profile" is designed for enhanced IPTV TDs, such as the fully-fledged model defined in [ITU-T H.722]. This profile defines the functional extensions required for IPTV services. This profile is optional, but, if implemented, there are some mandatory objects, methods and properties.

IPTV SESL can be supported in many applications and can also be included as a component in many presentation engines (PE), e.g., LIME [ITU-T H.762]. Some implementations may have a completely different set of libraries; what makes applications written in one IPTV SESL dialect work does not necessarily work in another dialect.

In the IPTV TD, the user agent (e.g., a web browser) provides an IPTV SESL host environment for client-side computation, which includes objects that represent windows, menus, etc. Further, the user agent provides a means to attach scripting code to events such as page and image loading, unloading,

etc. Scripting code appears within the LIME HTML of [ITU-T H.762] and the displayed page is a combination of user interface elements, together with the extended objects for IPTV services to provide web-based multimedia interaction.

7 Core script profile

ECMAScript [ISO/IEC 16262] was originally designed to be a web scripting language, providing a mechanism to enliven webpages in browsers and to perform server computation as part of web-based client-server architecture. It is suitable for being applied in web-based IPTV services to perform computations and provide interoperability with the end users within a host environment, which is the user agent (e.g., a browser) in the IPTV terminal device. It should be noted that not all of the manipulating computational objects defined by ECMAScript are necessary for implementation of the functionality defined for resource-constrained devices. Therefore, an optimized set of objects has been identified to efficiently implement that category of TDs. In this context, the core script profile has been defined.

The following clauses list the core script profile for IPTV services. All the built-in objects listed below are the objects whose semantics are defined in the LIME-Script of [ITU-T H.762]. Built-in objects must be supplied by the implementation at the start of the execution of the user agent in the IPTV terminal device.

7.1 Global

Global properties and functions can be used with built-in objects.

Properties:

- `NaN`
The not-a-number value

Functions:

- `parseInt ( String, radix )`
Parses a string and returns an integer
Radix of `parseInt()` can be 8 for octal, 10 for decimal or 16 for hexadecimal (0 is interpreted as 10)
- `isNaN ( number )`
Tests whether a value is the not-a-number value (i.e., an illegal number)

7.2 Object

LIME-Script objects provide basic language and facilities. The "Object" is one of the built-in objects.

Properties:

- `prototype`

Functions:

- `Object ( [value] )`
Creates a new Object object

Constructor:

- `new Object ( [value] )`
When called with new, a constructor may create an object

7.2.1 Object.prototype

Properties:

- `constructor`
Returns the function that created the Object prototype

Functions:

- `toString()`
Returns the string value corresponding to the [Class] internal property of the object
- `valueOf()`
Returns the primitive value of an Object object

7.3 Function

A function is a callable object that may be invoked as a subroutine.

Properties:

- `prototype`
Value is used to initialize the prototype internal property
- `length`
Indicates the number of arguments expected by the function

7.3.1 Function.prototype

Properties:

- `constructor`
It returns the function that created the prototype (in this case Function)

Functions:

- `toString()`
It returns a representation of the function

7.4 Array

The Array object is used to store multiple values in a single variable.

Properties:

- `prototype`
Allows the adding of properties and methods to an object
- `length`
Sets or returns the number of elements in an array

Functions:

- `Array ( item0, item1, ... )`
Creates a new array
- `new Array ( item0, item1,... )`
Creates and initializes an array
- `new Array ( [len] )`
Creates and initializes an array with length set to len

7.4.1 Array.prototype

Properties:

- `constructor`
Returns the function that creates the Array object prototype

Functions:

- `toString()`
Converts an array to a String, and returns the result
- `join ( [separator] )`
Joins all elements of an array into a String
- `reverse()`
Reverses the order of the elements in an array
- `sort ( [comparefn] )`
Sorts the elements of an array

7.5 String

A String is used to manipulate a stored piece of text.

Properties:

- `prototype`
Allows properties and methods to be added to an object
- `length`
Returns the length of a String

Functions:

- `String ( [value] )`
When called as a function it performs a type conversion

Constructor:

- `new String ( [value] )`
Used to create a String object
- `String.fromCharCode ( char0[ ,char1, ...] )`
Returns a String value containing as many characters as arguments

7.5.1 String.prototype

Properties:

- `constructor`
Returns the function that creates the String object prototype

Functions:

- `toString()`
Returns this String value
- `valueOf()`
Returns the primitive value of a String object
- `charAt ( pos )`
Returns the character at the specified index

- `charCodeAt ( pos )`
Returns the unicode of the character at the specified index
- `indexOf ( searchString, position )`
Returns the position of the first found occurrence of a specified value in a String
- `lastIndexOf ( searchString, position )`
Returns the position of the last found occurrence of a specified value in a String
- `split( separator )`
Splits a String into an array of substrings
- `substring ( start[,end] )`
Extracts the characters from a String, between two specified indices
- `toLowerCase()`
Converts a String to lowercase
- `toUpperCase()`
Converts a String to uppercase

7.6 Boolean

Boolean is used to convert a non-Boolean value to a Boolean value.

Properties:

- `prototype`
Allows adding properties and methods to an object

Functions:

- `Boolean ( [value] )`
Returns a Boolean value

Constructor:

- `new Boolean ( [value] )`
Used to create a Boolean object with an initial value

7.6.1 Boolean.prototype

Properties:

- `constructor`
Returns the function that creates the Boolean object prototype

Functions:

- `toString()`
Converts a Boolean value to a String, and returns the result
- `valueOf()`
Returns the primitive value of a Boolean object

7.7 Number

Object wrapper for primitive numeric values.

Properties:

- `prototype`

Allows you to add properties and methods to an object

- `MAX_VALUE`
Returns the largest number possible
- `MIN_VALUE`
Returns the smallest number possible
- `NaN`
The not-a-number value

Functions:

- `Number ( [value] )`
Function used to create a Number object

Constructor:

- `new Number ( [value] )`
Used to create Number objects

7.7.1 **Number.prototype**

- `constructor`
Returns the function that creates the Number object prototype

Functions:

- `toString ( [radix] )`
Converts a Number object to a String
- `valueOf()`
Returns the primitive value of a Number object

7.8 **Date**

Date is used to work with dates and time.

Properties:

- `prototype`
Allows properties and methods to be added to an object

Functions:

- `Date ( [year, month [, date [, hours [, minutes [, seconds [, ms]]]]]] )`

Constructor:

- `new Date ( [year, month [, date [, hours [, minutes [, seconds [, ms]]]]]] )`
Used to create Date objects

7.8.1 **Date.prototype**

Properties:

- `constructor`
Returns the function that created the Date object prototype

Functions:

- `toString()`
Converts a Date object to a String

- `getFullYear()`
Returns the year (four digits)
- `getUTCFullYear()`
Returns the year, according to universal time (four digits)
- `getMonth()`
Returns the month (from 0-11)
- `getUTCMonth()`
Returns the month, according to universal time (from 0-11)
- `getDate()`
Returns the day of the month (from 1-31)
- `getUTCDate()`
Returns the day of the month, according to universal time (from 1-31)
- `getDay()`
Returns the day of the week (from 0-6)
- `getUTCDay()`
Returns the day of the week, according to universal time (from 0-6)
- `getHours()`
Returns the hour (from 0-23)
- `getUTCHours()`
Returns the hour, according to universal time (from 0-23)
- `getMinutes()`
Returns the minutes (from 0-59)
- `getUTCMinutes()`
Returns the minutes, according to universal time (from 0-59)
- `getSeconds()`
Returns the seconds (from 0-59)
- `getUTCSeconds()`
Returns the seconds, according to universal time (from 0-59)
- `getMilliseconds()`
Returns the milliseconds (from 0-999)
- `getUTCMilliseconds()`
Returns the milliseconds, according to universal time (from 0-999)
- `getTimezoneOffset()`
Returns the time difference between GMT and local time, in minutes
- `setMilliseconds ( ms )`
Sets the milliseconds (from 0-999)
- `setUTCMilliseconds ( ms )`
Sets the milliseconds, according to universal time (from 0-999)
- `setSeconds ( sec, [, ms] )`
Sets the seconds (from 0-59)

- `setUTCSeconds ( sec, [, ms] )`
Set the seconds, according to universal time (from 0-59)
- `setMinutes ( min [, sec, [, ms]] )`
Set the minutes (from 0-59)
- `setUTCMinutes ( min [, sec, [, ms]] )`
Set the minutes, according to universal time (from 0-59)
- `setHours ( hour [, min [, sec, [, ms]] ] )`
Sets the hour (from 0-23)
- `setUTCHours ( hour [, min [, sec, [, ms]] ] )`
Sets the hour, according to universal time (from 0-23)
- `setDate(date)`
Sets the day of the month (from 1-31)
- `setMonth ( mon [, date] )`
Sets the month (from 0-11)
- `setUTCMonth ( mon [, date] )`
Sets the month, according to universal time (from 0-11)
- `setFullYear ( year [, mon [, date]] )`
Sets the year (four digits)
- `setUTCFullYear ( year [, mon [, date]] )`
Sets the year, according to universal time (four digits)
- `setYear ( year )`
Use the `setFullYear()` method instead
- `toLocaleString()`
Converts a Date object to a String object using locale conventions
Produces the same output format as `Date.prototype.toString()`
- `toUTCString()`
Converts a Date object to a String, according to universal time
Produces the same output format as `Date.prototype.toString()`

8 Extended script profile

The core script profile for IPTV provides the basic native built-in objects implemented by resource-restrained IPTV TDs. Since web-based IPTV also requires video controlling and user interactivity, an extension is needed to the core script profile for IPTV. The extended script profile defined in this Recommendation provides for these additional manipulating computational objects for video and interactivity-related performing computations, which may be optionally implemented by IPTV TDs.

The procedure on how the core script profile and extended script profile coordinate with the web-based terminal middleware to implement interactivity amongst the end user, terminal device and IPTV service platform is given in [ITU-T H.730].

8.1 MediaController object

The MediaController object encapsulates the capabilities of the terminal device to play live channel, nPVR, VoD, music and other media types. The MediaController object can provide necessary methods and properties to control a media.

The essential behaviour of playing for live channel, nPVR, VoD, music and other media types are the same as for those which can be abstracted to the methods and properties of MediaController, including the status of play, trick mode, stop, etc.

The programmed MediaController object in the web document not only carries the necessary information of the current media with it, but also provides methods for user interactivity such as trick mode control and web document refreshment.

8.1.1 Media control

8.1.1.1 Properties

Since the MediaController object provides the ability to control and display the media for IPTV, the identifier, status and other properties of the player are listed in the following clauses.

8.1.1.1.1 allowTrickmode

The property controls trick mode (fast forward/fast rewind/pause) operation during the life cycle of the player. This property is in logic AND relationship with “playbackMode” property of the MediaController.

- Optionality: Mandatory
- DataType: `unsignedInt[0:1]`
- Reference value:
 - 0: TRICK mode disabled
 - 1 (default): TRICK mode enabled
- Read-write: R/W

8.1.1.1.2 currentTime

The current playing time of the media. For VoD, the time represents the relative time (NPT) since the start point of the media. For nPVR, it represents the current absolute time (ClockTime). For channel, it is meaningless and an empty string will be returned.

- Optionality: Mandatory
- DataType: `String(128)`
- Reference value:
 - Empty string
- Read-write: R

NOTE – NormalPlayTime(NPT) and AbsoluteTime(ClockTime) refer to [IETF RFC 2326].

8.1.1.1.3 channelNum

The number of the current channel.

- Optionality: Mandatory
- DataType: `int[-1:4294967295]`
- Reference value:
 - If the current media is not a channel, the property is set to "-1"
- Read-write: R

8.1.1.1.4 cycleFlag

The property controls media loop play.

- Optionality: Mandatory
- DataType: `unsignedInt[0:1]`

- Reference value:
 - 0: loop mode
 - 1 (default): Single play mode
- Read-write: R/W

8.1.1.1.5 height

The height of the video output (in pixels). Each time the property is changed, the video output remains unchanged until the "refreshVideoDisplay" method is called.

- Optionality: Mandatory
- DataType: `int[4]`
- Reference value:
 - The default value is set to the height of the browser window (e.g., for PAL the height is 576 pixels)
- Read-write: R/W

8.1.1.1.6 instanceId

The unique ID of the local media player instance created.

- Optionality: Mandatory
- DataType: `unsignedInt[0:255]`
- Reference value:
 - The default value is 0, indicating no MediaController object instance has been initialized
- Read-write: R/W

8.1.1.1.7 left

The relative excursion from the left-top corner of the browser window (in pixels). Each time the property is changed, the video output remains unchanged until the "refreshVideoDisplay" method is called.

- Optionality: Mandatory
- DataType: `int[4]`
- Reference value:
 - The default value is 0
- Read-write: R/W

8.1.1.1.8 playMode

The play mode of the MediaController. The property has to be defined since the initiation of the player. It cannot be changed during playing until the MediaController is stopped.

- Optionality: Mandatory
- DataType: `unsignedInt[0:1]`
- Reference value:
 - 0 (default): single media mode
 - 1: playlist mode
- Read-write: R/W

8.1.1.1.9 width

The width of the video output (in pixels). Each time the property is changed, the video output remains unchanged until the "refreshVideoDisplay" (see clause 8.1.1.2.15) method is called.

- Optionality: Mandatory
- DataType: `int[4]`
- Reference value:
 - The default value is set to the width of the browser window (e.g., for PAL, the width is 720 pixels)
- Read-write: R/W

8.1.1.1.10 mute

Set the mute/sound status of the MediaController. It works as soon as the property is set.

- Optionality: Mandatory
- DataType: `unsignedInt[0:1]`
- Reference value:
 - 0 (default): sound
 - 1: mute
- Read-write: R/W

8.1.1.1.11 mediaCode

The unique identifier of the current media played by the MediaController instance. If the "playMode" property of the MediaController is set to playlist mode, the property represents the "mediaCode" of the current played media.

- Optionality: Mandatory
- DataType: `String`
- Reference value:
 - None
- Read-write: R

8.1.1.1.12 mediaDuration

The duration (in seconds) of the current media.

- Optionality: Mandatory
- DataType: `unsignedInt`
- Reference value:
 - 0
- Read-write: R

8.1.1.1.13 playbackMode

The playback state of the MediaController object. The player functional block should notify the web document by Event object, whenever the status of the player changes. This property is encapsulated into a formatted string like JSON or XML, which contains the following 2 parameters.

- Optionality: Mandatory
- Current Mode: NormalPlay, Pause, Trickmode.
- Speed: 0x, 1x, 2x/-2x, 4x/-4x, 8x/-8x, 16x/-16x, 32x/-32x, etc.

- **DataType:** String(128)
- **Reference value:**
 - 0
- **Read-write:** R

8.1.1.1.14 top

The relative excursion from the left-top corner of the browser window (in pixels). Each time the property is changed, the video output remains unchanged until the "refreshVideoDisplay" method is called.

- **Optionality:** Mandatory
- **DataType:** int[4]
- **Reference value:**
 - 0
- **Read-write:** R/W

8.1.1.1.15 videoAlpha

It represents the transparency of the video.

- **Optionality:** Optional
- **DataType:** unsignedInt[0:100]
- **Reference value:**
 - 0 (default): Opaque
 - 1: Transparent
- **Read-write:** R/W

8.1.1.1.16 videoDisplayMode

The video display mode of the MediaController. Each time the "videoDisplayMode" is changed, the video output remains unchanged until the "refreshVideoDisplay" method is called.

- **Optionality:** Mandatory
- **DataType:** unsignedInt[0:3,255]
- **Reference value:**
 - 0: The video should be displayed according to the "left", "top", "width", "left" properties.
 - 1 (default): Full screen mode.
 - 2: The video should be displayed according to the height of the screen without changing the height video aspect ratio.
 - 3: The video should be displayed according to the width of the screen without changing the width video aspect ratio.
 - 255: The video output should be hidden while the audio is still playing.
- **Read-write:** R/W

8.1.1.2 Methods

8.1.1.2.1 Constructor

The constructor of the MediaController object with a default value.

- **Optionality:** Mandatory

- Syntax:
`MediaController()`
- Parameters:
 - None
- Return value:
 - None

8.1.1.2.2 **bindNativePlayerInstance**

Create a new `MediaController` object by binding an existing native player instance and take over the management to that media player or media stream.

- Optionality: Mandatory
- Syntax:
`number bindNativePlayerInstance ( input number instanceID )`
- Parameters:
 - `instanceID`: Refer to "instanceId".
- Return value:
 - 0: success
 - -1: failure
 - Others: undefined

NOTE – If the newly created player cannot match a local "instanceId", the operation will be reported with failure by returning -1.

8.1.1.2.3 **fastForward**

Media fast forwarding with the rate indicated by "speed".

- Optionality: Mandatory
- Syntax:
`fastForward ( input number speed )`
- Parameters:
 - `speed ( int )`: valid values are positive integers, e.g., 64, 32, 16, 8, 4, 2, etc.
- Return value:
 - None

8.1.1.2.4 **fastRewind**

Media fast rewinding with the rate indicated by "speed".

- Optionality: Mandatory
- Syntax:
`fastRewind ( input number speed )`
- Parameters:
 - `speed ( int )`: valid values are negative integers, e.g., -64, -32, -16, -8, -4, -2, etc.
- Return value:
 - None

8.1.1.2.5 **getVolume**

Get the current volume of the IPTV TD.