**Common Extras**

**Menu Metadata**

**Contents**

1 Introduction 1

1.1 Overview 1

1.1.1 Technical Approach 1

1.1.2 Extras Architecture 1

1.1.3 Relationship of Extras to Common Metadata 1

1.2 Document Organization 1

1.3 Document Notation and Conventions 2

1.3.1 XML Conventions 2

1.3.2 General Notes 4

1.4 Normative References 4

1.5 Informative References 5

1.6 General Types 5

1.6.1 PlayabackStart-type 5

2 Extras Menu Top Level Elements 6

2.1 ExtrasMenuExperience-type 6

2.2 ExtrasMenu-type 6

3 Compatibility 8

3.1.1 CompatibilityFixedLayout-type 8

4 Menus 11

4.1 Menus 11

4.1.1 Menus-type 11

4.1.2 Menu-type 12

4.1.3 MenuElement-type 12

4.1.4 MenuUserAction-type and MenuElementUserAction-type 13

4.1.5 MenuBehavior-type 15

4.1.6 MenuScroll-type 16

4.1.7 AcquireAsset-type 16

4.2 Menu Components 17

4.2.1 Position 17

4.2.2 Button 17

4.2.3 Background 18

4.2.4 Image Positioned 18

5 Galleries 19

5.1 Gallery 19

5.1.1 Gallery-type 19

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
|  |  |  |
|  |  |  |

# Introduction

This document builds on Media Manifest Metadata by adding a simple menu system.

This specification is designed as a resource. Those using this specification may extend the definition with additional data element specific for their needs. They may replace elements with others perhaps more suitable to their needs; however, for interoperability all are highly encouraged to use the data elements exactly as defined.

Common Extras Metadata is part of the Common Metadata family of specifications.

## Overview

### Technical Approach

This document builds on Common Extras Manifest Metadata by adding a simple menu system.

### Extras Architecture

The Extras Menu architecture has the following data objects

* [TBS]

From these components an Extras Menu can be created.

### Relationship of Extras to Common Metadata

Common Extras is an extension to Common Metadata and may be used in conjunction with Common Metadata, or as its own entity.

Common Metadata includes elements that cover typical definitions of media, particularly movies and television. Common Metadata has two parts: Basic Metadata and Digital Asset Metadata. Basic Metadata includes descriptions such as title and artists. It describes information about the work independent of encoding. Digital Asset metadata describes information about individual encoded audio, video and subtitle streams, and other media included. Package and File Metadata describes one possible packaging scenario and ties in other metadata types. Ratings and Parental Control information is described.

Common Metadata is designed to provide definitions to be inserted into other metadata systems. A given metadata scheme, for example, the Entertainment Merchant’s Association (EMA) may select element of the Common Metadata to be used within its definitions. EMA would then define additional metadata to cover areas not included in Common Metadata.

## Document Organization

This document is organized as follows:

1. Introduction—Provides background, scope and conventions
2. [TBS]

## Document Notation and Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119]. That is:

* “MUST”, “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification.
* “MUST NOT” or “SHALL NOT” means that the definition is an absolute prohibition of the specification.
* “SHOULD” or “RECOMMENDED” mean that there may be valid reasons to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
* “SHOULD NOT” or “NOT RECOMMENDED” mean that there may be valid reasons when the particular behavior is acceptable, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
* “MAY” or “OPTIONAL” mean the item is truly optional, however a preferred implementation may be specified for OPTIONAL features to improve interoperability.

Terms defined to have a specific meaning within this specification will be capitalized, e.g. “Track”, and should be interpreted with their general meaning if not capitalized.

Normative key words are written in all caps, e.g. “SHALL”

### XML Conventions

XML is used extensively in this document to describe data. It does not necessarily imply that actual data exchanged will be in XML. For example, JSON may be used equivalently.

This document uses tables to define XML structure. These tables may combine multiple elements and attributes in a single table. Although this does not align with schema structure, it is much more readable and hence easier to review and to implement.

Although the tables are less exact than XSD, the tables should not conflict with the schema. Such contradictions should be noted as errors and corrected.

#### Naming Conventions

This section describes naming conventions for Common Metadata XML attributes, element and other named entities. The conventions are as follows:

* Names use initial caps, as in InitialCaps.
* Elements begin with a capital letter, as in InitialCapitalElement.
* Attributes begin with a lowercase letter, as in initiaLowercaseAttribute.
* XML structures are formatted as Courier New, such as md:rightstoken
* Names of both simple and complex types are followed with “-type”

#### Structure of Element Table

Each section begins with an information introduction. For example, “The Bin Element describes the unique case information assigned to the notice.”

This is followed by a table with the following structure.

The headings are

* Element—the name of the element.
* Attribute—the name of the attribute
* Definition—a descriptive definition. The definition may define conditions of usage or other constraints.
* Value—the format of the attribute or element. Value may be an XML type (e.g., “string”) or a reference to another element description (e.g., “See Bar Element”). Annotations for limits or enumerations may be included (e.g.,” int [0..100]” to indicate an XML xs:int type with an accepted range from 1 to 100 inclusively)
* Card—cardinality of the element. If blank, then it is 1. Other typical values are 0..1 (optional), 1..n and 0..n.

The first row of the table after the header is the element being defined. This is immediately followed by attributes of this element, if any. Subsequent rows are child elements and their attributes. All child elements (i.e., those that are direct descendents) are included in the table. Simple child elements may be fully defined here (e.g., “Title”, “ ”, “Title of work”, “xs:string”), or described fully elsewhere (“POC”, “ ”, “Person to contact in case there is a problem”, “md:ContactInfo-type”). In this example, if POC was to be defined by a complex type defined as md:ContactInfo-type. Attributes immediately follow the containing element.

Accompanying the table is as much normative explanation as appropriate to fully define the element, and potentially examples for clarity. Examples and other informative descriptive text may follow. XML examples are included toward the end of the document and the referenced web sites.

### General Notes

All required elements and attributes must be included.

When enumerations are provided in the form ‘enumeration’, the quotation marks (‘’) should not be included.

The term “Device” refers to an entity playing the interactive material specified here. It may be a standalone physical device, such as a Blu-ray player, or it might be an application running on a general purpose computer, a table, phone or as part of another device. The term ‘User’ refers to the person using the Device.

## Normative References

[CM] Common Metadata, [www.movielabs.com/md/md](http://www.movielabs.com/md/md)

[Manifest] Common Metadata Media Manifest Metadata, [www.movielabs.com/md/manifest](http://www.movielabs.com/md/manifest)

[CPE] Cross-Platform Extras, [www.movielabs.com/md/cpe](http://www.movielabs.com/md/cpe)

[Ratings] Common Ratings Metadata, [www.movielabs.com/md/ratings](http://www.movielabs.com/md/ratings)

[RFC4646] Philips, A, et al, *RFC 4646, Tags for Identifying Languages*, IETF, September, 2006. <http://www.ietf.org/rfc/rfc4646.txt>

[ISO639] ISO 639-2 Registration Authority, Library of Congress. <http://www.loc.gov/standards/iso639-2/>

[ISO3166-1] Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes, 2007.

[ISO3166-2] ISO 3166-2:2007Codes for the representation of names of countries and their subdivisions -- Part 2: Country subdivision code

[ISO4217] Currency shall be encoded using ISO 4217 Alphabetic Code. <http://www.iso.org/iso/currency_codes_list-1>

[ISO8601] ISO 8601:2000 Second Edition, *Representation of dates and times, second edition*, 2000-12-15.

[TTML] Timed Text Markup Language (TTML) 1.0, W3C Proposed Recommendation 14 September 2010, <http://www.w3.org/TR/ttaf1-dfxp/>

## Informative References

## General Types

### PlayabackStart-type

Playback can come from a Presentation or Playable Sequence as defined in [Manifest].

By default, playback begins at the beginning of the Presentation on Playable Sequence. If desired, ChapterIndex or EntryTimecode can be provided to indicate playback is to start at the referenced point.

Note that playback will continue to the end of the Presentation or PlayableSequence, or until stopped. If a playback stop time is required, then a Playable Sequence with the appropriate stop time should be created and referenced.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **PlaybackStart-type** |  |  |  |  |
| PlayableSequenceID |  | ID for PlayableSequence to be played | manifest:PlayableSequenceID-type | (choice) |
| PresentationID |  | ID for Presentation to be played | manifest:Presentation-type |
| ChapterIndex |  | If applicable, Chapter Index from reference into the Presentation or Playable Sequence | xs:integer | 0..1  (optional choice) |
| EntryTimecode |  | If applicable, Start timecode to begin playback | manfest:Timecode-type |

# Extras Menu Top Level Elements

ExtrasMenuSet is a collection of related ExtrasMenus.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **ExtrasMenuCollection-type** |  |  |  |  |
|  | MenuCollectionID | Unique identifier for this Extras Menu | extrasmenu:ExtrasMenuICollectionD-type | 0..1 |
| SpecVersion |  | Version of this specification. | xs:string |  |
| ExtrasMenuExperience |  | Mapping of Experiences to Menus. | extrasmenu:ExtrasMenuExperience-type | 0..n |
| ExtrasMenu |  |  | extrasmenu:ExtrasMenu-type | 1..n |

## ExtrasMenuExperience-type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **ExtrasMenuExperience-type** |  |  |  |  |
| ExperienceID |  | ID for the Experience or Experiences that map to a particular set of Menus. | manifest:ExperienceID-type | 1..n |
| ExtrasMenuID |  | Set of Extras Menus that correspond with the Experiences in ExperienceID. | extrasmenu:ExtrasMenuID-type | 1..n |

## ExtrasMenu-type

ExtrasMenu contains identification information, information on compatibility, the menus themselves and galleries (a special kind of menu).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **ExtrasMenus-type** |  |  |  |  |
|  | ExtrasMenuID | Unique identifier for this Extras Menu | Extrasmenu:ExtrasMenuID-type | 0..1 |
|  | updateNum | Version of this Extras Menu instance. Initial release should be 1. This is a value assigned by the manifest creator that should only be incremented if a new version of manifest is released. If absent, 1 is to be assumed. This is generally only used for selective update workflows. | xs:positiveInteger | 0..1 |
| Compatibility |  | Information to player on compatibility. | extrasmenu:Compatibility-type |  |
| Menus |  | A menu. Together, these elements constitute the set of menus. | extrasmenu:Menus-type |  |
| Gallery |  | Picture galleries | extrasmenu:Gallery-type | 0..n |

ExtraMenu is can be authored for display environments, with an instance for each environment. The player is assumed to use Compatibility to choose an ExtrasMenu instance that is compatible with its display environment.

Navigation begins by displaying the menu associated with Menus/StartMenuID.

# Compatibility

Compatibility provides information so a player knows whether or not this ExtrasMenu is playable. Compatibility-type is an extension of extras:Capability-type which defines specification-level compatibility.

Players are assumed to play all menus that are less than or equal then the specification to which they were designed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Compatibility-type** |  |  |  |  |
| SpecVersion |  | Version of this specification, to which the XML document is written | xs:string |  |
| DeviceClass |  | General Device category | xs:string | 0..n |
|  | subClass | Specialization of DeviceClass | xs:string | 0..1 |
| FixedLayout |  | Information describing compatible fixed layout devices. | extrasmenu:CompatibiltyFixedLayout-type | (choice) |

DeviceClass is encoded as follows:

* ‘Computer’ – general purpose computer such as a PC or Mac. Display could be built-in or external monitor
* ‘TV’ – any large screen display device. This refer to the “10 foot experience”.
* ‘Phone’ – Smartphone devices. Although lines are blurred with tablets (phablets), a phone is typically 6” or smaller.
* ‘Tablet’ – Tablet devices, typically 7” or larger. Note that 6”-7” can use the Phone or Tablet moniker.

DeviceClass/@subClass is used to provide additional detail on DeviceClass. There are currently no pre-defined values for @subClass.

Currently, only Fixed Layout is defined. Other layouts may be defined in the future.

### CompatibilityFixedLayout-type

Players are limited in display capability. However, there is a range of devices for which a given resolution and aspect ratio playable.

As Menus are designed to be associated with video, certain assumptions are made

* Menus are targeted for high definition devices (720p and higher)
* Square pixels

The current design is fixed layout. However, other layout models may be supported in the future.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **CompatibilityResolution-type** |  |  |  |  |
|  | rotate | Indicates whether rotation is allowed. | xs:boolean |  |
|  | crop | Indicates whether size can be cropped to no less than Safe Area. | xs:boolean |  |
| WidthPixels |  | Number of pixels horizontally in background area | xs:positiveInteger |  |
| HeightPixels |  | Number of pixels horizontally in background area | xs:positiveInteger |  |
| SafeWidthPixels |  | Number of pixels horizontally in safe area | xs:positiveInteger | 0..1  (if any are included all must be included) |
| SafeHeightPixels |  | Number of pixels horizontally in safe area | xs:positiveInteger |
| SafePixelsFromLeft |  | Start of safe area from left of background area | xs:positiveInteger |
| SafePixelsfromBottom |  | Start of safe area from bottom of background area | xs:positiveInteger |

The rotation attribute indicates whether the layout must remain in the orientation of the Width and Height Pixels or whether it can be rotated 90 (or 270) degrees. Rotating a menu not designed for rotation will generally result in unacceptable layout.

The crop attribute indicates whether the background extends beyond the safe area has been defined for active content. If crop is ‘true’, the safe area must be defined.

The safe area is the area within the WidthPixels by HeightPixels area that all players must display. The safe area size is SafePixelsWidth wide, SafePixelsHeight pixels high. The safe area location starts on the pixel SafePixelsLeft from the left of the background and SafePixelsFromBottom from the bottom of background. The following illustrations shows these value.



For example, if the background is 1080p and the author wishes to leave 20 pixels outside the safe area, values would be set as follows

WidthPixels = 1920

HeighPixels = 1080

SafePixelsWidth = 1880 [i.e., 1920 – 20 pixels left – 20 pixels right)

SafePixelsHeight = 1040

SafePixelsFromLeft = 20

SafePixelsFromRight = 20

Note that the Safe Area defined in Cross Platform Extras (CPE) has a different meaning and that does not apply here; or vice versa.

# Menus

Menus is a generalized mechanism for displaying simple menus. The following illustrates the components of a menu:

**Chapters**

**Background**

* image/video
* audio

**Menu Object**

* Image/video
* Trigger
  + Rollover
  + Select
  + Up/down/left/right/etc.
* Action
  + Another Menu
  + Start Video
* Transition

**Play Setup Chapters**

**Main Menu**

Menus are designed to be independent of human interface modalities (navigation using a remote control, keyboard/mouse, gesture, etc.) (left, right, up, down, select, numbers) would be a different modality than a mouse and keyboard.

## Menus

### Menus-type

TBS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Menus-type** |  |  |  |  |
| StartMenuID |  | Initial Menu to be displayed | extrasmenu:MenuID-type |  |
| Menu |  | A menu. Together, these elements constitute the set of menus. | extrasmenu:MenuID-type | 1..n |

### Menu-type

TBS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Menu-type** |  |  |  |  |
|  | menuID | Identifier for this menu | extrasmenu:MenuID-type |  |
| Background |  | Background for the menu. This is displayed whenever the menu is active. | extrasmenu:Background-type |  |
| FirstElementID |  | Element to start with. | extrasmenu:MenuElementID-type |  |
| Element |  | Each element. Together these are all menu elements on the menu. | extrasmenu:MenuElement-type | 1..n |
| MenuActions |  | Actions to be taken based on user interaction. | extrasmenu:MenuUserAction-type | 0..1 |
| Scrollbar |  | Description of how information scrolls on this menu (if applicable) | extrasmenu:MenuScroll-type | 0..1 |

### MenuElement-type

Every element has both visual appearance and behavior that occurs upon user interaction. The Icon is the appearance. The action behaviors are similar to what would be found in in JavaScript (e.g., onClick). TBS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **MenuElement-type** |  |  |  |  |
|  | MenuElementID | ID uniquely identifying this menu element. | extrasmenu:MenuElementID-type |  |
| Icon |  | Visual, accessible audio and position information associated with element. | extrasmenu:ButtonPositioned-type |  |
| ElementActions |  | Action taken based on user interaction | extrasmenu:MenuElementAction-type |  |

### MenuUserAction-type and MenuElementUserAction-type

These elements define the response to user interaction with various user-interface modalities (touch, mouse, remote, etc.). Some actions apply to the entire menu (swipes, timeout), while others are specific to a menu item. These are assigned to MenuUserAction-type and MenuElementUserAction-type accordingly.

* Menu
  + Finger – SwipeLeft/SwipeRight/SwipeUp/SwipeDown. These are associated with a finger on a screen, such as a tablet.
  + Timeout – An action can be defined, either for inactivity or a specific time.
* Element
  + Select – Universal across modalities, although it may occur with different user interaction (e.g., mouse click, finger tap, remote select button, keyboard enter)
  + Double-click—universal across modalities (e.g., mouse double-click, finger double-tap)
  + Arrows – Left/right/up/down. These may be associated with a remote or with a keyboard,
  + Mouse – MouseOver/DoubleClick. Associated with a mouse or equivalent device

#### MenuUserAction-type

This type defines actions that can be taken from a Menu background. Actions taken on a menu element are listed under MenuElementUserAction-type.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **MenuUserAction-type** |  |  |  |  |
| OnSwipeUp |  | Action taken with a swipe up (bottom to top). | extrasmenu:MenuBehavior-type | 0..1 |
| OnSwipeDown |  | Action taken with a swipe down (top to bottom). | extrasmenu:MenuBehavior-type | 0..1 |
| OnSwipeLeft |  | Action taken with a swipe left (right to left). | extrasmenu:MenuBehavior-type | 0..1 |
| OnSwipeRight |  | Action taken with a swipe right (left to right). | extrasmenu:MenuBehavior-type | 0..1 |
| OnTimeout |  | Action taken with a swipe up (bottom to top). | extrasmenu:MenuBehavior-type | 0..1 |
|  | timeout | Idle time duration after which action is taken | xs:duration |  |

#### MenuElementUserAction-type

This element type defines the actions that can be taken on a Menu item. This is slightly different than the actions that can be taken on a backround. For example, there is not swipe behavior on a menu item.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **MenuElementUserAction-type** |  |  |  |  |
| OnSelect |  | Action taken when selected | extrasmenu:MenuBehavior-type | 0..1 |
| OnDoubleClick |  | Action taken when selected with a double-stroke (double-click, double-tap, etc.) | extrasmenuextrasmenu::MenuBehavior-type | 0..1 |
| OnUp |  | Action taken when focus removed toward top (e.g., up arrow) | extrasmenu:MenuBehavior-type | 0..1 |
| OnDown |  | Action taken when focus removed towards bottom (e.g., down arrow) | extrasmenu:MenuBehavior-type | 0..1 |
| OnLeft |  | Action taken when focus removed to left (e.g., left arrow) | extrasmenu:MenuBehavior-type | 0..1 |
| OnRight |  | Action taken when focus removed to right (e.g., right arrow) | extrasmenu:MenuBehavior-type | 0..1 |
| OnMouseover |  | Action taken when cursor hovers over element. Note that RolloverIcon behavior is independent of this action. | extrasmenu:MenuBehavior-type | 0..1 |

### MenuBehavior-type

Indicates what happens ‘next’ after a menu item has been selected or when focused is removed from a menu item (i.e., left, right, down or up).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **MenuBehavior-type** |  |  |  |  |
| MenuID |  | Jump to this menu | extrasmenu:MenuID-type | (choice) |
| MenuElementID |  | Jump to this element on the same Menu | extrasmenu:MenuElement-type |  |
| PlaybackStart |  | Begin Playback at the specified starting point. | extrasmenu:PlaybackStart-type |  |
| ChapterUnique |  | Jump to this chapter. | extrasmenu:ChapterUnique-type |  |
| ChooseAudioTrackRef |  | Select audio track for subsequent playback. This overrides default track selection. | manifest:SubtitleTrackID-type |  |
| ChooseVideoTrackRef |  | Select video track for subsequent playback. This overrides default track selection. | manifest:SubtitleTrackID-type |  |
| ChooseSubTrackRef |  | Select subtitle track for subsequent playback. This overrides default track selection. | manifest:SubtitleTrackID-type |  |
| AcquireAsset |  | Acquire (Buy) the asset. | extrasmenu:AcquireAsset-type |  |

[CHS: Should we have a way of selecting system language and subtitle type?]

### MenuScroll-type

Provides for scrolling through options, such as traversing a gallery. Scrolling may be horizontal (left/right) or vertical (up/down). When a user selects to scroll left, text moves right. When a user selects to scroll right, text moves left. And, so forth.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **MenuScroll-type** |  |  |  |  |
|  | horizontal | If present and ‘true’ menu scrolls horizontally. Otherwise, menu scroll vertically. | xs:boolean | 0..1 |
| ScrollBarImage |  | Image for scroll bar. | extrasmenu:ImagePositioned-type |  |
| PositionMarker |  | Image at location of selected text, typically within ScrollBarImage. [CHS: this needs work for positioning.] | extrasmenu:ImageID-type |  |
| LeftUpIcon |  | Icon to display for left scrolling (horizontal) or up scrolling (not horizontal). | extrasmenu:Button-type |  |
| RightDownIcon |  | Icon to display for right scrolling (horizontal) or down scrolling (not horizontal) | extrasmenu:Button-type |  |

### AcquireAsset-type

This type allows an asset to be acquired. This corresponds with CPE’s Content Access APIs acquire method; [CPE], Section 5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **AcquireAsset-type** |  |  |  |  |
| ALID |  | Logical Asset Identifier associated with the asset. (ContentID in [CPE]). | md:AssetLogicalID-type |  |
| RequestType |  | Acquisition intent (e.g., buy or rent). If absent, this should be resolved by the retailer. | xs:string | 0..1 |

Request is encoded as follows

* ‘Buy’ – indicates a purchase intent
* ‘Rent’ – indicates a rent intent

## Menu Components

### Position

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Position-type** |  |  |  |  |
| PixelsFromLeft |  | Pixels from left. First pixel is ‘1’. | xs:integer |  |
| PixelsFromTop |  | Pixels from top of area. First pixel is ‘1’ | xs:integer |  |

### Button

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Button-type** |  |  |  |  |
| Icon |  | ID of Image for the button icon. | extras:ImageID-type |  |
| RolloverIcon |  | ID of Image for the button icon when cursor is on the icon. | extras:ImageID-type | 0..1 |
| LabelText |  | Alternative text representing button when images cannot be displayed. Also, for accessibility, especially text-to-speech. | xs:string | 0..1 |
| AlternateAudio |  | Audio clip associated with button, typically for accessibility | extras:AudioClipRef-type |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **ButtonPositioned-type** |  |  | extrasmenu:Button-type (extension) |  |
| Position |  | Position of Button | extrasmenu:Position-type |  |

### Background

Backgrounds may contain at most one visual and one audio element. If a ChainID element is included, neither an AudioLoopID nor an ImageID element should be included.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Background-type** |  |  |  |  |
| AudioLoopID |  | Audio to play with background. Audio specified should loop. | manifest:AudioTrackID-type | 0..1 |
| ImageID |  | ImageID | manifest:ImageID-type | 0..1 |
| Clip |  | Chain that includes video and audio for background. Chain should loop. | manifest:AudiovisualClipRef-type | 0..1 |

### Image Positioned

Provides the ability to precisely position an image on the display.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Background-type** |  |  |  |  |
| ImageID |  | Image to be positioned | manifest:ImageD-type |  |
| Position |  | Position for image | extrasmenu:Position-type |  |

# Galleries

A gallery is a special type of menu. It is referenced by a MenuID and can be invoked as any other menu.

## Gallery

The user interface for the presentation of images is called a Gallery. The Gallery contains enough information to provide a simple display of images.

A gallery contains

* Name – Used for gallery selection)
* Picture Group – Images associated with Gallery. The Gallery will include all images in the Picture Group.
* Background – Image or video background with optional audio.
* Auto-advance timing – If system is to display images automatically, how long to dwell on each slide.

### Gallery-type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **Gallery-type** |  |  |  |  |
|  | MenuID | MenuID associated with this gallery. | extrasmenu:MenuID-type |  |
| Type |  |  |  |  |
| PictureGroupID |  | Picture Group containing Pictures for gallery | manifest:PictureGroupID |  |
| GalleryName |  | Title of Gallery | xs:string | 0..n |
|  | language | Language of gallery | xs:language | 0..1 |
| GalleryNameAlternateAudio |  | Audio corresponding GalleryName. Anticipated use is accessibility. | manifest:AudioClipRef-type | 0..n |
|  | language | Language of GalleryNameAlternateAudio | xs:language | 0..1 |
| Layout |  | Layout and behavior for images | extrasmenu:GalleryLayout-type |  |
| AutoNextSlideTime |  | Time each Picture dwells on screen before switching to next Picture. If ‘0’, images should be switched manually. If absent, Device may select its own time, or choose not to switch images automatically. | xs:duration | 0..1 |
| Loop |  | Should images be displayed in a loop? That is, should first image be displayed after last image? If absent or ‘false’ images are not looped. If ‘true’ images are looped. | xs:boolean | 0..1 |
| ExitMenuID |  | MenuID for menu to be invoked upon exiting the Gallery. | extrasmenu:MenuID-type |  |

#### GalleryLayout-type

This definition allows the author to specify Gallery layout including background image, image placement and buttons.

The following illustrates the elements of a layout.

**< Previous Exit Next >**

**Gallery**

**Background**

* image/video
* audio

**Title burned in**

**Borders burned in**

**Positioned Images**

**Prev, Exit and Next buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Element** | **Attribute** | **Definition** | **Value** | **Card.** |
| **GalleryLayout-type** |  |  |  |  |
| Background |  | Background image for gallery. Pictures will be overlaid on this image. If absent, Device may use a background of its choice. | extrasmenu:Background-type | 0..1 |
| PicturePosition |  | For each image that can be displayed simultaneously, the position of that image. One entry must exist for each position. For example, in a 4-up gallery, there must be 4 instances. | extrasmenu:Position-type | 1…n |
| NextButton |  | Button that goes to next page of Gallery | extrasmenu:ButtonPositioned-type | 0..1 |
| PreviousButton |  | Button that goes to previous page of Gallery | extrasmenu:ButtonPositioned-type | 0..1 |
| ExitButton |  | Button to exit Gallery | extrasmenu:ButtonPositioned-type | 0..1 |

[CHS: I’m concerned the buttons are too complicated. Perhaps we can have something simpler.]