PlayIt - User Guide

The PlayIt User Guide gives a full breakdown of all the parameters that drive how PlayIt works and functions. This guide will cover everything you need to know to create your own PlayIt system.

Version 1.0 Hide All ☐

Index

Overview

LAYOUT

PlayIt consists of two components, the user interface (UI) and a datasource. The user interface drives all visual cues of PlayIt and the datasource provides PlayIt with it's dynamic data content. A full break down of all the PlayIt features and functionality can be seen in the demo "All About PlayIt" which comes bundled with the PlayIt source code.

DATASOURCES

PlayIt works with the concept of a datasource. The datasource is what provides PlayIt with all its data. Currently PlayIt supports two types of datasources. The first type of datasource is from an XML configuration file. This type of data source can be generated dynamically with a server side scripting language such as ASP.net, PHP, Ruby or any other type of technology. The XML file can also be a purely static XML file that you pre-populate yourself.

The second type of datasource is much more technical and currently experimental. It derives from a Web Service that returns a JSON object. The JSON responses are then fed into PlayIt. This experimental Web Service datasource will contain much more documentation and guidance as PlayIt continues to rollout new types of datasources.

Index

Setting Up

The easiest and fastest way to get PlayIt working in your environment is to just make a copy of the about-playit.html and about-playit.xml files and begin to modify the configuration file to point to your content. The instructions below contain the step by step instructions for setting up PlayIt on your site.

STEP 1. Install files

- Extract the jquery.playit.zip archive to your web server
- The PlayIt demos will need to be served from the 'http://' path and can be removed if needed

STEP 2. Include References

- If you ONLY need to support HTML 5 browsers then the mediaelementjs references are not needed.
- Insert the code in Figure 1-1 into your *<head>* tag
- Update the paths below. The themename 'cupertino' is interchangeable.
  - **{THEME-CDN}**
    - http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.16/cupertino/jquery-ui.css

        - http://ajax.aspnetcdn.com/ajax/jquery.ui/1.8.16/themes/cupertino/jquery-ui.css
    - **{JQUERY-CDN}**
        - http://ajax.googleapis.com/ajax/libs/jquery/1.7/jquery.min.js
        - http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.7.min.js
    - **{JQUERY-UI-CDN}**
        - http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.16/jquery-ui.min.js
        - http://ajax.aspnetcdn.com/ajax/jquery.ui/1.8.16/jquery-ui.min.js

```
<link rel="stylesheet" href="{THEME-CDN}" id="playit-jquery-ui-css" />
<link rel="stylesheet" href="[path-to-playit]/jquery.playit.css" />
<link rel="stylesheet" href="[path-to-playit]/lib/mediaelementjs/mediaelementplayer.css" />
<link rel="stylesheet" href="[path-to-playit]/lib/mediaelementjs/mejs-skins.css" />
<script type="text/javascript" src="{JQUERY-CDN}"></script>
<script type="text/javascript" src="{JQUERY-UI-CDN}"></script>
<script type="text/javascript" src="[path-to-playit]/jquery.playit.min.js" id="playit-source"></script>
<script type="text/javascript" src="[path-to-playit]/lib/mediaelementjs/mediaelement-and-player.min.js"></script>
```

Figure 1-1 required head tags

CDNs can be your own server.

STEP 3. Call PlayIt Code

- Insert the following code after your *<body>* tag
- If using the XML datasource (used by default) then make sure xmlConfig param points to your configuration file.

```
<div id="playit"></div>
<script type="text/javascript">
   jQuery(document).ready(function () {
      jQuery("#playit").playit({
         dataSourceOptions: {
            xmlConfig: 'my-playit-config.xml',
            cacheXml: false
         }
      });
   });
</script>
```

STEP 4. Configure XML Datasource

When using the XML datasource keep the *cacheXml* parameter false. This is important otherwise your data will not show correctly. Set this value to true once you have completed your set up and development.

Note: If you're dynamically generating the XML file then you can point the xmlConfig value at any extension such as: http://mysite/path/dynamic.aspx | php | jsp | etc...The output of any of these dynamic pages must be a valid PlayIt XML file.

Index

Datasource: XML File

The following settings pertain to a PlayIt XML configuration file. This is one of the main datasources that can power PlayIt and what ultimately drives the layout and content of PlayIt. Please note that all attributes and nodes are case sensitive and should use lower case for all values. Because XML is a strict markup it is important that you handle every setting with precise care. If the PlayIt configuration file doesn't load correctly then you can validate your XML file with a tool such as the W3C Markup Validation Service to make sure you have your configuration file setup correctly.

Navigation: playit - section - category

<playit>

The playit tag is used to control many of the global aspects of the plugin. This root level tag only contains section nodes and only one playit tag is to be used within an xml datasource configuration file.

children: section

attribute	type	overview
General Settings		
accordion-animate	bool	Allows the accordion menu to animate when clicked *[default:true]*
accordion-autoselect	bool	Auto selects the first or first *isdefault* category when a section is clicked *[default:false]*
accordion-minwidth	int	The minimum width the accordion menu can be resized to *[default:200]*
accordion-maxwidth	int	The maximum width the accordion menu can be resized to *[default:800]*
autofill	bool	The height of the plugin will fill its current parent upto 100% *[default:false] see notice below for useage*
ios-iframeasobject	bool	Forces iOS devices such as iPad to use the object tag over the iframe *[default:true] see notice below.*
ios-iframeasobjectlink	bool	Creates a small message over the object tag with a link to the souce *[default:true] see notice below.*
toolbar-title	text	The text that displays in the center of the toolbar and defaults to *[default:'PlayIt Media Player']*
toolbar-searchtext	text	The text that displays in the search box when no search as been entered *[default:'Search']*
toolbar-searchwidth	int	The width of the search box measured in pixels *[default:auto]*
toolbar-showsearch	bool	Shows or hides the search box *[default:true]*
view-default	enum	Determines which data view is active when the control is loaded *[options:grid,list,tile default:grid]*
Audio Specific		
audio-autoplay	bool	Auto plays the first audio node when a category of type audio is loaded *[default:false]*
audio-info-enable	bool	Shows or hides the info button on the audio player *[default:true]*.
audio-infobox-modal	bool	Opens the audio information dialog as a modal dialog *[default:true]*
audio-infobox-height	int	Controls the height of the audio information dialog *[default:300]*
audio-infobox-width	int	Controls the width of the audio information dialog *[default:400]*
audio-infobox-resizable	int	Opens the audio information dialog as a resizable dialog *[default:true]*
Image Specific		
image-autoplay	bool	When the image player is loaded it will start to play through the images *[default:true]*
image-dialog-height	int	The height in pixels the image dialog window will open*[default:500]*
image-dialog-width	int	The width in pixels the image dialog window will open*[default:700]*
image-info-enable	bool	Shows or hides the info button on the image player *[default:true]*
image-slides-interval	int	The duration in which each image is displayed in milliseconds *[default:3000]*
image-slides-thumbcount	int	How many thumbnails show in the image player thumb view *[default:7]*
Video Specific		
video-autoplay	bool	Auto plays a locally hosted video *[default:false]* Services like YouTube requires a query string to that service.
video-dialog-height	int	Controls the height of the video dialog window and *[default:500]*
video-dialog-width	int	Controls the width of the video dialog window and *[default:700]*
video-info-enable	bool	Shows or hides the video information button on the navigation bar.
video-navbar-enable	bool	Shows or hides the video navigation bar.

⚠ The **autofill** attribute only fills to its current parent. Additional CSS is required to fill the entire window. Elements from the players default locatoin upto the html tag will need to have their respective heights set to 100%. See the

demo file about-playit.html for an overview.

The **ios-iframeasobject** attribute only applies to iOS devices. Currently devices like the iPad don't allow iframes to be resized, therefore the alternative is to render a fragment of the page and provide a link to the corresponding category link. This setting may eventually become obsolete if the iOS browser changes it's behavior.

<section>

The section node creates the highest level of organization in the navigation menu of PlayIt. The section node supports an unlimited number of category nodes per section.

children: category

attribute	type	overview
name	text	Displays the name of the section
isdefault	bool	Sets this section as the default section. If isdefault is set on multiple sections, then the last one will be selected *[default:false]*

<category>

The category node contains the collection of media data elements such as audio, image, link, and video nodes. A category can contain many but only one type of the media data elements per category. When a category is selected all of its children nodes will be visible the main view panel and searchable. If the category type='link' then it can point to a web asset itself, such as a web page or be a collection of many link nodes.

children: audio || img || link || video || grid || list || tile

attribute	type	overview
type *	enum	The type of data the category will display **required attribute [options:audio,image,link,video,separator]*
name	text	Displays the name of the category
isdefault	bool	Sets this category as the default category. If isdefault is set on multiple categories, then the last one will be selected. The categories parent section must have isdefault set to true before its isdefault property to be accepted. *[default:false]*
link-src	text	Url or path to valid web asset, such as a web page, pdf or image. This attribute requires that the type is 'link'. If the link-src attribute is empty then PlayIt will attempt to load any <link> elements that are present
link-scroll	enum	If the 'link-src' element is set then the viewable category iframe can be scrollable. *[options:auto,on,off default:auto]*
view-default	enum	Determines which data view is active when the control is first loaded. This option will override the playit@view-default option if set. *[options:grid,list,tile default:grid]*

Category Views: grid - list - tile - field(s)

<grid>

The grid tag controls the viewable aspects of the grid. For readability it is recommended to place this tag before any of the media element tags. A grid contains fields which drive which columns are shown and how they sort.

children: fields

attribute	type	overview
enable	bool	Allows for the grid view to be shown or hidden on a per category basis *[default:true]*

<list>

The list tag controls all aspects of how the list view is displayed. A list contains fields which drive which items are shown and how they sort. The list view incorporates an additional 'sortdisplay' attribute which is tied only to the list view. This allows the sort button to have a custom name apart from what is viewed in the list item.

children: fields

attribute	type	overview
enable	bool	Allows for the list view to be shown or hidden on a per category basis *[default:true]*

<tile>

The tile tag controls all aspects of how the tile view is displayed. A tile contains no fields and all sortable content fields are fixed. When a categories type is audio the tile view will display the 'Album' and 'Artist' attributes as sortable data, all other category types will only display the 'Title'. The tile view does contain a slider that will allow you change the size of the active poster image from 64px up to 256px.

If you want to change fixed text sort items (i.e. 'Artist', 'Album', 'Title') you can do this through the plait.lang file settings. See the localizing section for details on changing the internal text settings.

children: *no-children*

attribute	type	overview
enable	bool	Allows for the list view to be shown or hidden on a per category basis *[default:true]*
item-size	enum	Set the default image (poster) item-size when the Tile view is loaded. The tile view only supports images up to 256x256, for performance it is recommended to scale your images no larger than 256x256. *[options:64, 128, 192, 256 default:64]*

<fields>

The fields node are useable only to the grid and list views and is used to control what columns and labels are viewable.

children: *field*

attribute	type	overview
sort-column	text	The column to sortby when the view is first opened. This value must be a valid attribute that is tied to any of the category types (audio, image, link, video). If the category type has a fixed attribute or custom attribute then sortcolumn must match that value. For example you could use the fixed audio@title attribute which would equate to sortcolumn='title' for the value. For custom attributes such as audio@mycustom you would then use sortcolumn='mycustom' as the sortcolumn value.
sort-direction	enum	The direction the 'sortcolumn' should short by *[options:asc, desc default:asc]*

<field>

The field node controls the display and visibility of which columns and labels are seen in the grid and list views.

children: *no-children*

attribute	type	overview
display	text	The display name you want the required *match* attribute to be displayed as. For example you may have chosen to match audio@title but you want the grid column or list label to show something different such as 'Song Name'. In this context you would set the field as: <field match="title" display="Song Name" /> This would display all title attributes but the column header or label would show 'Song Name'.
match *	text	The category types (audio, image, link, video) attribute that should be shown in the column for the grid view or as a label for the list view. For example you could match the fixed attribute image@title which would equate to match='title' for the value. Used in the grid fields context 'title' will show as a column in the grid. Used in the list fields context 'title' will show as a label for each item. The match

		attribute can be applied to any category types attribute fixed or custom. **required attribute [options:any category type nodes attributes]*
sort-display	text	This value only applies to the <list> view and allows the sort button to have a different name from the label display name.
width	int	This value only applies to the <grid> view and controls the width of each column

Category Types: audio - img - link - video

<audio>

The audio tag controls all aspects of an individual audio file. Audio files can be grouped together by using the exact same album and artist values across multiple audio tags.

children: info

attribute	type	overview
album	text	The album or group of audio tracks. The album and artist attributes together form a unique key for grouping when used on the tile view. If your not working with the traditional definition of an album it can still be used to group shows, podcasts or any type of audio.
artist	text	The artist or producer of the audio. The album and artist attributes together form a unique key for grouping when used on the tile view.
info-enable	bool	Enables or disables the audio information button on the audio player. If the audio <info> tag is empty the information button will be disabled. The attribute playit@audio-info-enable must be set to true before this value is evaluated. *[default:true, when info tag is filled with content]*
poster	text	The path to an image file used as a display asset on the tile view. The image size for the poster image ranges from 64x64 to 256x256.
src	text	The path to an audio source file such as mp3.
title *	text	The required name for the audio file **required attribute*
thumb	text	The path to an image file used as a display on the audio control panel and list view. The image size range is from 32x32 to 48x48 unless overridden in the css style sheet.
track	int	The track number within an album set
{dynamic}	var	Any dynamic meta-data attribute needed to support your audio file. For example if you wanted to add your own custom data such as release year, composer, and file size then you could attach the following attributes to the audio tag as such: <audio ... **release**='2002' **composer**='Cory Lamle;Ryan Heideman' **size**='2.5MB' ... /> These dynamic meta data fields can now be displayed as new columns in your grid or labels in your tile view. They can be named whatever you want and you can add as many as you see fit.

Please check your browsers documentation if the supplied audio format is not working, some browsers do not support specific formats.

The image tag controls all aspects of an image file

children: info

attribute	type	overview
height	int	The default height of the image when viewed in the image player
		Enables or disables the image information button on the image player. If the img <info> tag

info-enable	bool	is empty the information button will be disabled. The attribute playit@image-info-enable must be set to true before this value is evaluated. *[default:true, when info tag is filled with content]*
poster	text	The path to an image file used as a display asset on the tile view. The image size for the poster image ranges from 64x64 to 256x256.
src	text	The path to an image source file such as jpeg.
title *	text	The required name for the image file **required attribute*
thumb	text	The path to an image file used as a display on the audio control panel and list view. The image size range is from 32x32 to 48x48 unless overridden in the css style sheet.
width	int	The default width of the image when viewed in the image player
{dynamic}	var	Any dynamic meta-data attribute needed to support your image file. For example if you wanted to add your own custom data such as date or location taken then you could attach the following attributes to the img tag as such: <audio ... **date**='2002' **location**='San Diego, California' ... /> These dynamic meta data fields can now be displayed as new columns in your grid or labels in your tile view. They can be named whatever you want and you can add as many as you see fit.

<link>

The link tag controls all aspects of a link reference.

children: info

attribute	type	overview
features	text	A comma-separated list of items for a new window object. The following values are supported: height=pixels, left=pixels, resizable=yes|no|1|0, scrollbars=yes|no|1|0, status=yes|no|1|0, top=pixels, width=pixels
poster	text	The path to an image file used as a display asset on the tile view. The image size for the poster image ranges from 64x64 to 256x256.
src	text	The path to any web asset such as web page, pdf file, image or any item your browser can render.
target	text	The location where the window will open. The following values are supported: • _blank - URL is loaded into a new window. This is default • _parent - URL is loaded into the parent frame • _self - URL replaces the current page • _top - URL replaces any framesets that may be loaded • name - The name of the window
title *	text	The required name for the image file **required attribute*
thumb	text	The path to an image file used as a display on the audio control panel and list view. The image size range is from 32x32 to 48x48 unless overridden in the css style sheet.
{dynamic}	var	Any dynamic meta-data attribute needed to support your link tag. For example if you wanted to add your own custom data such as a custom group then you could attach the following attributes to the link tag as such: <link ... **group**='technology' ... /> These dynamic meta data fields can now be displayed as new columns in your grid or labels in your tile view. They can be named whatever you want and you can add as many as you see fit.

<video>

The video tag controls all aspects of a video file.

children: info | source

attribute	type	overview
info-enable	bool	Enables or disables the video information button on the video navigation bar. If the video <info> tag is empty the information button will be disabled. The attribute playit@video-info-enable must be set to true before this value is evaluated. *[default:true, when info tag is filled with content]*
player	enum	The type of video player needed for this video source. The 'local' option uses a default HTML 5 or flash player. The 'remote' option will use an external video player such as youtube. *[options:local, remote default:local]*
poster	text	The path to an image file used as a display asset on the tile view. The image size for the poster image ranges from 64x64 to 256x256.
src	text	The path to any video file your browser can render or the path to a video source like YouTube or Vimeo. If **player='remote'** then this value is required.
title *	text	The required name for the image file **required attribute*
thumb	text	The path to an image file used as a display on the audio control panel and list view. The image size range is from 32x32 to 48x48 unless overridden in the css style sheet.
{dynamic}	var	Any dynamic meta-data attribute needed to support your video tag. For example if you wanted to add your own custom data such as play time and a description then you could attach the following attributes to the video tag as such: <video ... **time**='5:00' **desc**='Trailer for movie XYZ' ... /> These dynamic meta data fields can now be displayed as new columns in your grid or labels in your tile view. They can be named whatever you want and you can add as many as you see fit.

<source>

The source tag specifies a physical video file. Used in the video tag this tag allows for multiple video sources. This tag is preferred over using the video 'src' attribute.

parent: video
children: no-children

attribute	type	overview
src	text	The path to any video file your browser can render. This should be a physical video file and not the link to a 3rd party service like YouTube. Include as many source tags as needed. The first source tag found will be attempted to play. If your browser cannot render the video then the next video source will be attempted.
type	text	The type of video that is being rendered, examples include .mp4, webm, ogg, mov

The fallback video for older browsers without HTML5 support should either be defined in the src attribute on the video node or be the first video in the set of sources.

If you don't include the following reference for the MediaElements fallback player:

```
<script type="text/javascript" src="lib/mediaelementjs/mediaelement-and-player.min.js" ></script>
```

Then some videos may not play on some browsers. Its important to always provide fallback source tags for the various browsers. Please check your browsers documentation if the supplied video format is not working, some browsers do not support specific formats. For a list of compatible video formats see the following link: HTML 5 Video Overview.

When setting the attribute **player='remote'** only the **src** attribute value is applied and all **source** tag(s) are ignored. The remote player should be a player provider by another site.

<info>

children: no-children

The info tag is used for custom information. Used by the category media types audio, image, and video this tag allows for custom information to be bound to the specific media type. This data can be text/html/javascript. The data inside this tag should be surrounded by:

<![CDATA[<b> My Markup <b>]]>

The CDATA is only needed if the content contains any markup, plain text does not require the CDATA.

Poster & Thumb Image Placement

Below is an overview of the order for how the thumb and poster attributes get interpreted. Each location uses a fallback approach, where if the first attribute in the list is not set then the next value will be attempted until it reaches the default image.

Audio Player Active Thumbnail: audio@thumb, audio@poster, then PlayIt music note image

Image Player Thumbnails: img@thumb, img@poster, then PlayIt default camera image

List View: [media-type]@thumb, [media-type]@poster, then PlayIt default [media-type] image

Tile View: [media-type]@poster, then PlayIt default [media-type] image

Index

Datasource: Web Services

The Web Services datasource is currently experimental. In future versions of PlayIt the documentation for this particular data source will be completely filled out. Currently the only supported documentation is through the demo files. See websrv-demo.asmx and websrv-demo.html for a quick example of hooking up PlayIt to a web service.

Index

Styling & Theming

PlayIt uses a layout philosophy mechanism centered heavily on stylesheets. Since it's impossible to know the different variations in which this plugin could be used, the implementation of CSS selectors in order for you to customize PlayIt to your liking. We strongly encourage you to use a DOM inspector to locate the styles you want to change.

PlayIt is ThemeRoller Ready which means you can easily create a theme to integrate into your site and fit your exact color schema. To create your own theme follow these simple steps:

1. Browse to the ThemeRoller Tool
2. On the right-hand side begin to create your theme, you will see your progress as you modify values.
3. When done click the "Download Theme" link then the "Download" button be sure the drop-down says "Custom Theme"
4. Inside the zip archive you just downloaded, you will see a custom theme folder with a CSS file named something like jquery-ui-1.8.13.custom.css
5. Include this custom theme as a CSS link tag in the pages where you plan to use PlayIt

PlayIt also comes with its own stylesheet jquery.playit.css which is used to lay out the basic elements of the plug-in. You can edit this stylesheet to your liking. The easiest way to style or change an element is to use one of the DOM inspectors to find out which element maps to a specific style.

Common Distribution Networks (CDN)

If you want to link to one of the CDN's you can do so by looking at the following URLs:

```
// Google
http://ajax.googleapis.com/ajax/libs/jqueryui/[UI.VERSION]/themes/[THEME-NAME]/jquery-ui.css
// Microsoft
http://ajax.aspnetcdn.com/ajax/jquery.ui/[UI.VERSION]/themes/[THEME-NAME]/jquery-ui.css
```

Here is a list of the most common themes via CDN.

base: Google | Microsoft
black-tie: Google | Microsoft
blitzer: Google | Microsoft
cupertino: Google | Microsoft
dark-hive: Google | Microsoft
dot-luv: Google | Microsoft
eggplant: Google | Microsoft
excite-bike: Google | Microsoft
flick: Google | Microsoft
hot-sneaks: Google | Microsoft
humanity: Google | Microsoft
mint-choc: Google | Microsoft
overcast: Google | Microsoft
pepper-grinder: Google | Microsoft
redmond: Google | Microsoft
smoothness: Google | Microsoft
south-street: Google | Microsoft
start: Google | Microsoft
sunny: Google | Microsoft
swanky-purse: Google | Microsoft
trontastic: Google | Microsoft
ui-darkness: Google | Microsoft
ui-lightness: Google | Microsoft
vader: Google | Microsoft

Index

Globalization

USING LANGUAGES

The data that PlayIt renders is all localizable. There are two parts to localizing PlayIt which are the internal and external content. The internal content is the content that drives the user interface elements, such as the buttons text. The external content is driven by the data source such as the XML configuration file.

Internal Content:
To change the internal content by language just add the following to your <head> tag.
<script type="text/javascript" src="source/lang/playit.de.js" id="playit-lang"></script>

External Content:
For external content just place any language in the PlayIt datasource.

Index

Performance

XML DATASOURCE CONSIDERATIONS

When creating your configuration file remember to always set the **cacheXml:** property to true when your done with your settings. See Setting Up for more details. By default PlayIt only renders a category when it is called. However the initial request to the configuration file will pull down the entire file. So anything you can do to keep a tight streamline configuration file is important. If your not using an attribute then don't include it. Only use the dynamic meta data attributes when they are really needed.

IMAGE FILES

One of the biggest performance increases you can get when using the attributes **thumb** and **poster** is to use compressed jpeg files where the thumb source is no larger than 64x64 and poster images are no larger than 256x256. Its also very important that the thumb attribute be used and point to a true 64x64 or smaller image.

When using the **img** tag it is very important that image being referenced is as tightly compressed as possible. It is highly recommended that your images do not exceed 1920x1080 or images stay below 400K if possible.

Index