February 5, 2013 Version 1.1

EnGage HTML 5 Support

The EnGage player software and EnGage Content Manager both support the use of HTML5 content via the HTML 5 content widget. The HTML 5 content widget can be associated with a content package which contains all the files required to support the HTML 5 application.

HTML5 Package

When using HTML5 content all the assets required need to be included into a ZIP file (package) including the, html, java script, css, images, video and any other supporting files. See the "Content Support" section of this document for details about what content types are supported by HTML 5 on the EnGage player.

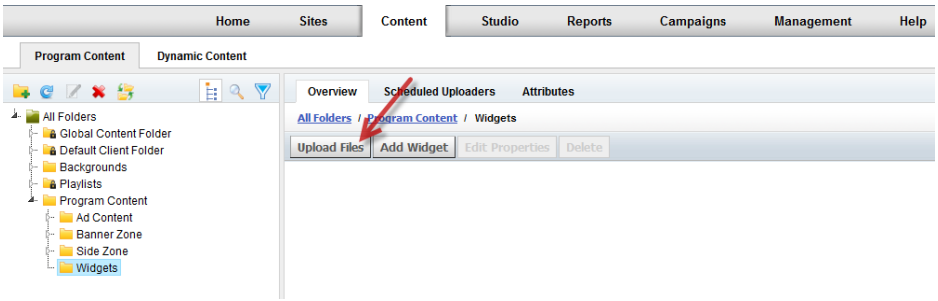
The ZIP file can then be uploaded to EnGage and referenced by an HTML 5 content widget. The HTML 5 package can contain folders and sub folders containing files. However, any code should reference the files with a relative path. See the "File Paths" section of this document for more information about file and content references from within the HTML 5 application.

The HTML 5 content widget will need an "entry point" which is the file the browser will initially load to start the application. This entry point should be "index.html" in the root of the ZIP file, however the entry point can be an alternate file name and can be contained within a directory, within the ZIP file. Be sure to specify the entry point when creating the HTML 5 content widget.

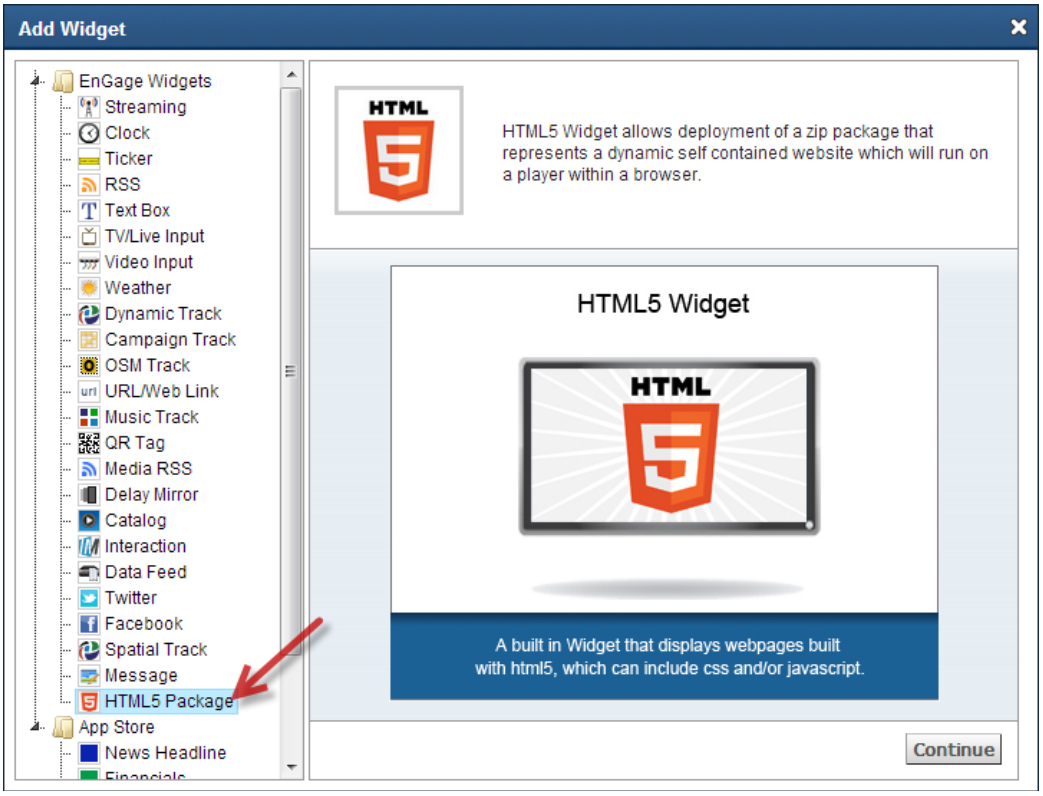
Creating an HTML5 widget

Before the HTML 5 content widget can be created, the HTML content package (ZIP) file needs to be uploaded to EnGage. Upload the ZIP file to any folder within the content tab similar to traditional video or image content.

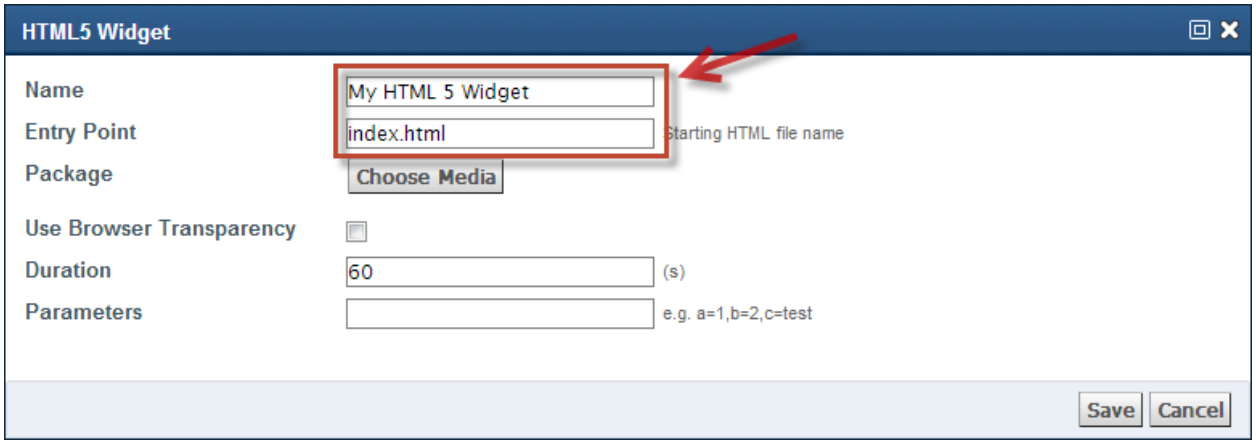
Step 1) Upload the content package (ZIP file contain all the HTML 5 application files) through the Content tab in Engage.



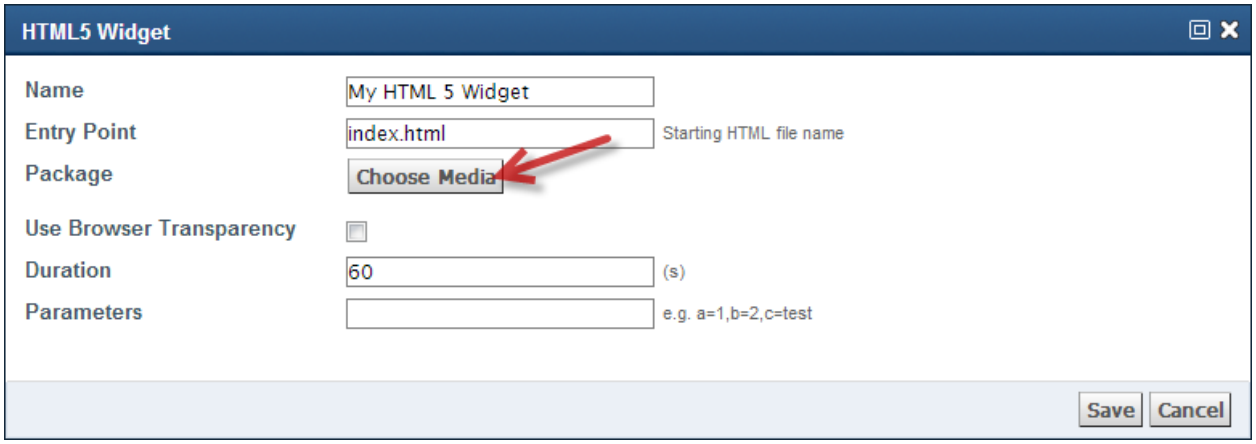
Step 2) Create the content widget by using the "Add Widget" button in the Content tab. Select the "HTML 5 Package" widget. Click "continue" to add the details for the widget.



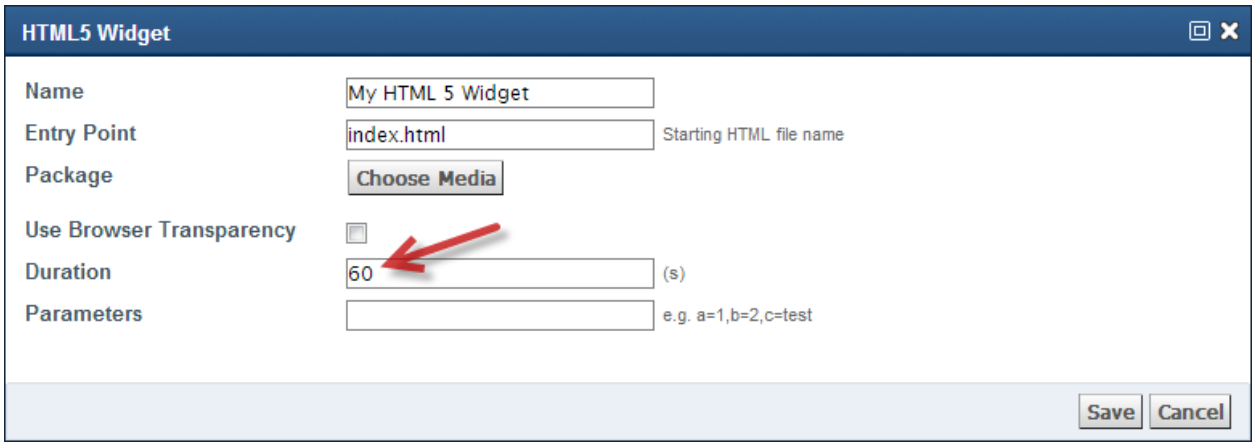
Step 3) Type in a “Name” for the HTML 5 widget and the “entry point”. The “entry point” is the first file the EnGage player loads to start the application. This filename should be “index.html”.



Step 4) Select the HTML5 Package by clicking on the “Choose Media” button. Browse the uploaded file in Step 1 and select the file. Click “OK” to select the file.



Step 5) Enter an appropriate duration for the application into the duration field. Click “Save” to finish the creation of the HTML 5 content widget.



Step 6) Now that the HTML 5 content widget is created, it can be added to a program or playlist.

NOTE: Be sure to test the HTML 5 application on a test player with the same player hardware as deployed in the field before fully deploying the application to live players.

Content Support

The EnGage player uses the WebKit browser engine and supports the known working features available in this engine. These supported features also include the support content types and codecs. If there is a need to integrate content into the HTML 5 application, the content must in an appropriate format. See the "Appendix A: HTML 5 Feature Support" section of this document for details about support content formats.

For example, in regards to video, the EnGage Player (which uses WebKit) supports WebM and "Ogg Theora" codecs.

> ***NOTE:*** The EnGage player does NOT support the use of H.264 and MPEG-4 video within an HTML 5 application since these formats are not supported by the WebKit browser engine.

Embedded JavaScript Functions

The HTML 5 content widget has functions available via JavaScript to access the EnGage player API that can be helpful to developers.

XML String EnGageAPI(XML String request)

Provides a direct link to the Engage Player API. *request* must be an xml string meeting the EnGage Player API specification. The return value will be the returned xml string from the Player API request.

String EnGageGetParameter(String name)

Provides access to the content parameters specified on the widget. *name* must be the name of one of the content parameters specified on the widget. If the value of the content parameter is a player variable reference ('i.e. name=$[player.var}'), then the current value of that player variable will be returned.
Return value is always a string, and will be null if no such parameter exists.

String EnGageGetVariable(String name)

Provides a convenience method to access the EnGage Player API's 'getVariable' method. *name* specifies the player variable being querried. The method will return the value of that player variable as a string, or null if no such parameter exists.

NULL EnGageSetVariable(String name, String Value)

Provides a convenience method to access the EnGage Player API's 'setVariable' method. *name* specifies the player variable being set, and *value* specifies the value to set. If no such player variable exists, a new one is created.
NULL EnGageLogMsg(String source, String msg)
Provides a convenience method to access the EnGage Player API's 'log' method. *source* is a user defined string used to identify the source of the log (i.e. a particular HTML application). *msg* is a string which contains the logged data. Logged data is available sent to the EnGage server, where it can be accessed via the EnGage web service API.

NULL EnGageSendCommand(String command)

Provides a generic callback mechanism to interface between web content and the Engage Player. command must be a specific command string understood and supported by the EnGage Player. The list of supported command and their actions is:

- "registerWatchdog"
  - Initiates the HTML watchdog. The watchdog timeout can be set by passing 'watchdogTimeout=N' as a content parameter, where *N* is the timeout in milliseconds.
- "resetWatchdog"
  - Resets the HTML watchdog counter. Send this periodically to prevent a timeout.
- "complete"
  - Indicate to the EnGage Player that playback should be terminated. This can be used to stop playback before the widget's defined duration.

In addition, the Engage Player may attempt to invoke a JavaScript function '**NULL EnGageCallback(String command)**'. The widget will invoke this command with a *command* value of 'startWatchdog' in response to a 'sendCommand("registerWatchdog")' invocation.

> ***NOTE:*** To use the embedded JavaScript functions the player API must be enabled from within the "Site Settings" object in EnGage. Be sure to Publish the "Site Settings" and issue a Reboot to the player to fully enable the Player API functionality.

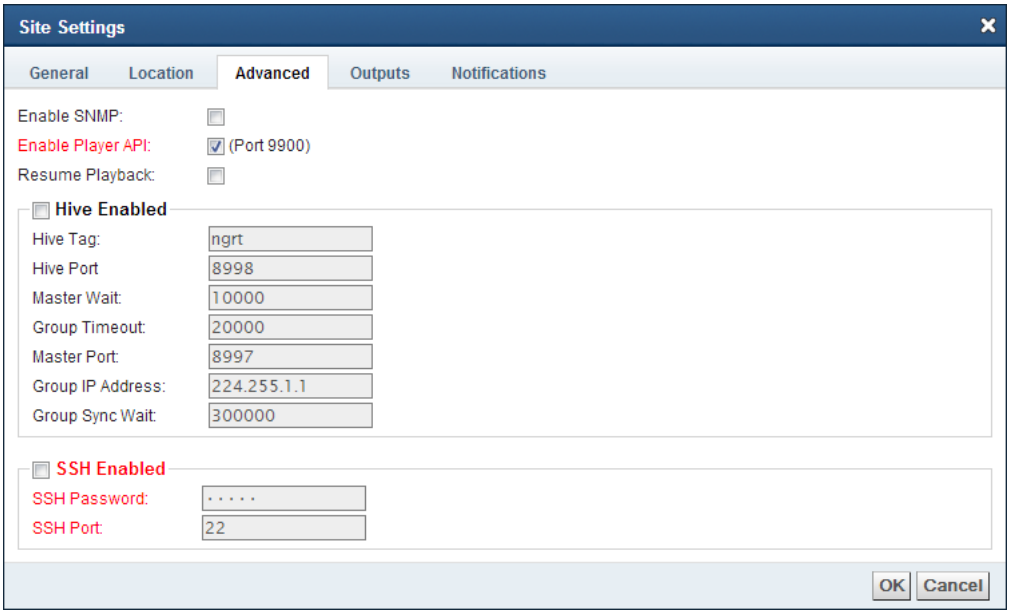


Figure 1: Enabling the API within the "Site Settings"

Supported Players

To use the HTML content widget on the EnGage player, it must be running a minimum firmware version of 8.4. The playback quality of the application will depend on the complexity of the HTML 5 application and the player hardware.

Note: Be sure to test the HTML 5 application on the same hardware as deployed in the field prior to deploying the application to live players.

> ***NOTE:*** Any changes to an entity after it has been scheduled for publishing will NOT be included in the scheduled publish.

File Paths

When referencing any file contained within the HTML 5 content package (ZIP file), the path should be relative to the root of the ZIP file. For example if referencing an image file in an "images" directory within the ZIP file, the path should be:

```
image/myimage.jpg
```

or

```
<img src="images/myimage.jpg">
```

When referencing any content or data files outside of the HTML 5 application the path should be absolute. For example, when referencing an XML file outside of the HTML 5 application the path should be:

/Media/data/xmldata.zip/mydata.xml

Supported Features

The EnGage player uses the WebKit browser engine and supports the known working features available in this engine. Any features that are currently "experimental" in the WebKit engine ***are not enabled on the EnGage player.*** Any feature that is not enabled can be enabled upon request with the ComQi Support team for addition in a future firmware release.

Please refer to "Appendix A: Feature Support" for a detailed list of all the HTML 5 features that are supported by the EnGage Player software.

EnGage Player Specific Details

There are some features specific to how HTML 5 operates on the EnGage player.

Scroll Bars

By default the WebKit browser will display the standard scroll bars within the HTML 5 application. To hide the scroll bars, use the appropriate CSS within the application.

Background Transparency

To use background transparency within the HTML 5 application, the “Use Browser Transparency” option must be enabled on the HTML 5 content widget. Once enabled the application background can be set to transparent or an opaque value by setting the value in the BODY tag of the HTML. If the “Use Browser Transparency” is enabled within a BODY background colour, the back will be transparent. If the “Use Browser Transparency” value is not enabled, the default background colour is white.

Appendix A: HTML 5 Feature Support

The following table is a list of supported features.

> ***NOTE*:** This list is based on results from html5test.com.

Feature	Supported?
Parsing rules	
<!DOCTYPE html> triggers standards mode	YES
HTML5 tokenizer	YES
HTML5 tree building	YES
SVG in text/html	YES
MathML in text/html	YES
Canvas	
canvas element	YES
2D context	YES
Text	YES
Video	
video element	YES
Subtitle support	NO
Poster image support	YES
MPEG-4 support	NO
H.264 support	NO
Ogg Theora support	YES
WebM support	YES
Audio	
audio element	YES
PCM audio support	YES
AAC support	YES

MP3 support	YES
Ogg Vorbis support	YES
Ogg Opus support	YES
WebM support	NO
Elements	
Embedding custom non-visible data	YES
Section elements	YES
Grouping content elements	Partial
Text-level semantic elements	Partial
Interactive elements	Partial
hidden attribute	YES
Dynamic markup insertion	YES
Forms	
Field types:	
input type=text	YES
input type=search	YES
input type=tel	YES
input type=url	YES
input type=email	YES
input type=datetime	Partial
input type=date	Partial
input type=month	Partial
input type=week	Partial
input type=time	Partial
input type=datetime-local	Partial
input type=number	YES
input type=range	YES
input type=color	NO
input type=checkbox	YES
input type=image	Partial
input type=file	YES

textarea	Partial
select	YES
fieldset	Partial
datalist	Partial
keygen	YES
output	YES
progress	YES
meter	YES
Fields:	
Field validation	YES
Association of controls and forms	YES
Other attributes	Partial
CSS selectors	YES
Events	YES
Forms:	
Form validation	YES
User Interaction	
Drag and Drop:	
Attributes	NO
Events	NO
HTML editing:	
Editing elements	NO
Editing documents	NO
APIs	NO
Spellcheck:	
spellcheck attribute	YES
History and navigation	
Session history	YES
Microdata	

Microdata	NO
Web applications	
Application Cache	YES
Custom scheme handlers	NO
Custom content handlers	NO
Custom search providers	NO
Security	
Sandboxed iframe	YES
Seamless iframe	NO
iframe with inline contents	NO
Various	
Scoped style element	NO
Asynchronous script execution	YES
Runtime script error reporting	YES
Base64 encoding and decoding	YES
Location and Orientation	
Geolocation	NO
Device Orientation	NO
WebGL	
3D context	NO
Native binary data	YES
Communication	
Cross-document messaging	YES
Server-Sent Events	YES
XMLHttpRequest Level 2	Partial
WebSocket	YES
Files	

ComQi

File API	YES
File API: Directories and System	NO
Storage	
Session Storage	YES
Local Storage	YES
IndexedDB	NO
Web SQL Database	YES
Workers	
Web Workers	YES
Shared Workers	YES
Local Multimedia	
Access the webcam	NO
Notifications	
Web Notifications	NO
Other	
Page Visibility	NO
Text selection	YES
Scroll into view	YES
Mutation Observer	NO
EXPERIMENTAL	
Audio	
Web Audio API	NO
Video and Animation	
Full screen support	YES
Pointer Lock support	NO
window.requestAnimationFrame	NO