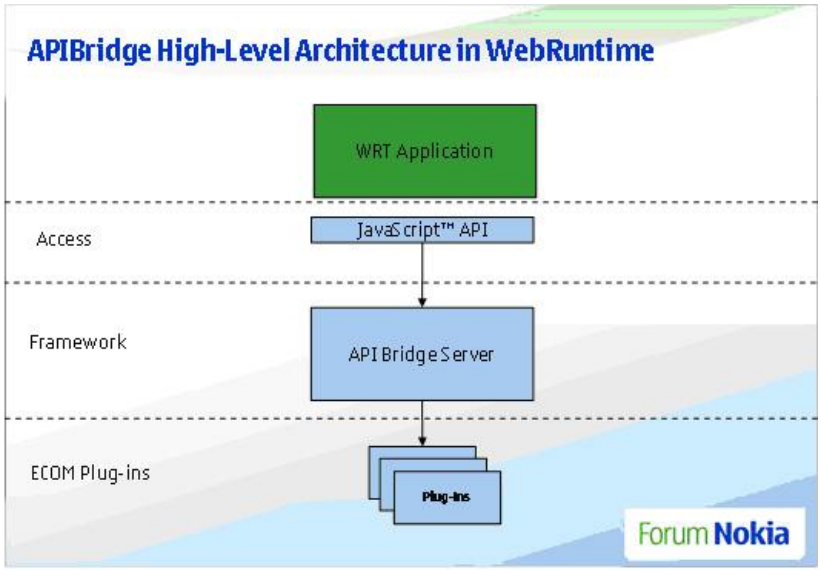
APIBridge Web Runtime API

Purpose of this document

This document explains how to use the services provided by the APIBridge component from the Web Runtime (WRT) environment. The document describes the API used to access the services provided by the APIBridge and how to set up the WRT application to make use of the APIBridge.

APIBridge High-Level Architecture

The APIBridge is a Symbian server that exposes an HTTP interface for communication between the server and its clients. The APIBridge can be used in the WRT environment by making Ajax calls to the local host port that the APIBridge listens to. The following architecture diagram explains the different parts of the system:



Here is a description of each of the components:

Component	Description
Access Layer JavaScript API	The JavaScript™ API provides WRT applications with a function-based interface to the services of the APIBridge. It performs the following activities: ▪ Internally generates the HTTP requests to the APIBridge from the function calls, abstracting this complexity from the application developer; ▪ Performs the required security checks and calls to the APIBridge in order to authenticate the application. This layer is represented by the apibridge.js file, which can be included and used in any WRT application that intends to use the APIBridge. Chapter 3 provides a description of the functions available.
Framework Layer APIBridge Server	The APIBridge Server is responsible for authenticating clients, receiving requests, routing requests to the appropriate APIBridge plug-in for execution, and returning the results.
ECOM Plug-ins Layer Plug-ins	Plug-ins are responsible for analysing the parameters in the request, calling the appropriate Symbian APIs to execute it, and creating the HTTP response.

APIBridge Web Runtime API

This chapter describes the JavaScript API available for WRT applications. The function API described here is contained in the apibridge.js file that developers will need to include in their WRT applications.

The required file can be found at the following:

<APIBridge Directory>\js\apibridge.js

A complete sample WRT application that uses the APIBridge APIs can be found in

(C) Copyright Nokia 2014. All rights reserved.

File Upload

Description

This function allows for the upload of a file to a web server using a multipart/form-data POST request. It also supports the addition of accompanying parameters in the same POST request:

```
fileUpload ( varName, fileName, url, parameters, onSuccess, onError )
```

Parameters

Parameter	Description
varName	Name of the variable that will contain the binary data of the uploaded file. This name will be used by the server.
fileName	Local path to the file to be uploaded.
url	Server URL where the request should be posted.
parameters	JavaScript object containing key-value pairs representing any additional variables that need to be posted to the server along with the file.
onSuccess	JavaScript callback function that will be called upon successful uploading of the server. It will contain the Ajax Request object as a parameter.
onError	JavaScript callback function that will be called when the operation is not successful. It will contain the Ajax Request object as a parameter.

Usage

```
var parameters = {
"userId" : "12345",
 "title" : "Upload Test"
};

APIBridge.fileUpload(
"image",
 "C:\Images\test.jpg",
 "http://www.test.com/fileupload.php",
 parameters,
 function(res) {
  alert("Done");
 },
 function(res) {
  alert("ERROR: " + res.status);
 }
);
```

New File Service

Description

This function allows WRT applications to embed native applications in order to capture images, videos, and audio:

```
newFileService ( fileType, onSuccess, onError )
```

Parameters

(C) Copyright Nokia 2014. All rights reserved.

Parameter	Description
fileType	Variable that indicates what type of file needs to be retrieved. The possible values are: ▪ APIBridge.NewFileType.Image ▪ APIBridge.NewFileType.Video ▪ APIBridge.NewFileType.Audio
onSuccess	JavaScript callback function that will be called upon successful file capture. It will contain the path of the new file or NULL if nothing was captured as a parameter.
onError	JavaScript callback function that will be called when the operation is not successful. It will contain the Ajax Request object as a parameter.

Usage

```
APIBridge.newFileService(
 APIBridge.NewFileType.Image,
 function(src) {
if (src && src.length)
   document.write("<img src='" + src + "' />");
else
   alert("No image taken...");
 },
 function(res) {
  alert(("Error "+res.status);
 }
);
```

File Reading

Description

This function allows WRT applications to read the contents of a local file:

```
readFile ( fileName, onSuccess, onError )
```

Parameters

Parameter	Description
readFile	Contains a path to the local file to be read.
onSuccess	JavaScript callback function that will be called upon successful file reading. It will contain the following two parameters: ▪ File contents: A string containing the contents of the file; ▪ MIME type: The MIME type of the file.
onError	JavaScript callback function that will be called when the operation is not successful. It will contain the Ajax Request object as a parameter.

Usage

```
APIBridge.readFile(
 "C:\test.xml",
 function(contents,mType) {
document.write("File Read. Mime-type: "+mType+"
");
document.write(contents);
 },
```


(C) Copyright Nokia 2014. All rights reserved.

```
  function(res) {
   alert(("Error "+res.status);
  }
 );
```

Image Resizing

Description

This function allows WRT applications to generate an on-the-fly image that is a different size from the original. This is useful when creating gallery views of the device image files.

This function will not return the actual resized image but a URL to the bridge, which will create the image upon call. This URL can be used as an src value of an IMG tag.

```
resizeImg (fileName, width, height, maintainAspectRatio, onSuccess, onError)
```

Parameters

Parameter	Description
fileName	Local path to the image file that needs to be resized.
width	Target width of the resized image in pixels.
height	Target height of the resized image in pixels.
maintainAspectRatio	Boolean parameters that indicate whether the developer would like to maintain the aspect ratio of the resized image. If yes, the original proportions of the image will be maintained. In this case the developer needs to indicate width or height but not both.
onSuccess	JavaScript callback function that will be called upon successful construction of the URL. It has one parameter that contains the URL to use for retrieving the resized image.
onError	JavaScript callback function that will be called when the operation is not successful. It will contain the Ajax Request object as a parameter.

Usage

```
APIBridge.resizeImg(
"C:\Images\test.jpg",
80,
80,
false,
 function(src) {
  document.write("<img src='" + src + "' />");
 },
 function(req) {
alert("ERROR "+req.status);
 }
);
```

Image Thumbnail Creation

Description

This function allows WRT applications to access the thumbnail for a particular image in the device's image gallery. It provides several performance advantages over the resize image feature, since thumbnails might already be generated by other applications or, if not, will only be generated once and stored. However it is not possible to specify the size of the thumbnail.

This function will not return the actual thumbnail image but a URL to the bridge, which will return the thumbnail upon call. This

(C) Copyright Nokia 2014. All rights reserved.

URL can be used as an src value of an IMG tag.

```
getThumbnail (fileName, onSuccess, onError)
```

Parameters

Parameter	Description
fileName	Local path to the image file.
onSuccess	JavaScript callback function that will be called upon successful construction of the URL. It has one parameter that contains the URL to use for retrieving the thumbnail.
onError	JavaScript callback function that will be called when the operation is not successful. It will contain the Ajax Request object as a parameter.

Usage

```
APIBridge.getThumbnail(
"C:\Images\test.jpg",
 function(src) {
  document.write("<img src='" + src + "' />");
 },
 function(req) {
alert("ERROR "+req.status);
 }
);
```

Logging Service

Description

This function allows WRT applications to access the device call log:

```
getLoggingService().getList (criteria, callback)
```

Parameters

Parameter	Description
criteria	Currently not used. Parameter is provided for future additions to the service.
callback	JavaScript callback function that will be called when the operations has been completed. This callback function follows the same rules used in the WRT 1.1 Service API. For more information: Web Developer's Library For a description of the Iterable object retrieved upon successful completion: Web Developer's Library

Usage

```
function displayResults(transId, eventCode, result) {
 if (hasError(result))
  return;

 var item;
 var out = "";

 var iterableList = result.ReturnValue;
```


(C) Copyright Nokia 2014. All rights reserved.

```
  if (!iterableList.getNext)
   iterableList = new APIBridge.IterableList([iterableList]);

  while ( item = iterableList.getNext() ) {
   for (var n in item) {
    out += n + "=" + item[n] + "<br />";
   }

   out += "<br />";
  }

  var output = document.getElementById("output");
  output.innerHTML = ( out.length > 0 ) ? out : "(Empty list)";
 }
 var ls = APIBridge.getLoggingService();
 ls.GetList(null, displayResults);
```

Location Service

Description

This function allows WRT applications to access the device's current location information:

```
getLocationService().GetLocation (criteria, callback)
```

Parameters

Parameter	Description
criteria	Currently not used. Parameter is provided for future additions to the service.
callback	JavaScript callback function that will be called when the operation has been completed. This callback function follows the same rules used in the WRT 1.1 Service API. For more information:Web Developer's Library For a description of the Iterable object retrieved upon successful completion: Web Developer's Library

Usage

```
function displayResults(transId, eventCode, result) {
```

See complete implementation of this function in APIBrigde_Web_Runtime_API#Usage_6

```
}
var ls = APIBridge.getLocationService();
ls.GetLocation(null, displayResults);
```

Media Management Service

Description

This function allows WRT applications to access the device media files:

```
getMediaManagementnService().GetList (criteria, callback)
```

Parameters

(C) Copyright Nokia 2014. All rights reserved.

Parameter	Description
criteria	Parameter follows the same syntax defined in the WRT 1.1 Service API: Web Developer's Library **Support Notes**: Currently only FileDate is supported as a filter.
callback	JavaScript callback function that will be called when the operation has been completed. This callback function follows the same rules used in the WRT 1.1 Service API. For more information: Web Developer's Library For a description of the Iterable object retrieved upon successful completion: Web Developer's Library

Usage

```
function displayResults(transId, eventCode, result) {
```

... See complete implementation of this function in APIBrigde_Web_Runtime_API#Usage_6

```
}
var mms = APIBridge.getMediaManagementService();

var criteria = {
sort: {
          key: "FileDate",
order: "Descending"
},
 Type: "FileInfo",
Filter: {
  FileType: "Image"
 }
};

mms.GetList(criteria,
function(transId, eventCode, result) {
  if (hasError(result))
   return;

  displayResults(transId, eventCode, result);
 }
);
```

Telephony Service

The telephony services requires the widget to add the "telephony.js" file in addition to "apibridge.js"

Description

This function allows WRT applications to send DTMF tones when there's an active phone call:

```
sendDTMF( string, onSuccess, onError)
```

Parameters

Parameter	Description
string	This contains a valid string of characters that can be transmitted over DTMF (0-9) and the symbols # *
onSucess	JavaScript callback function that will be called when the operation has been completed succesfully

(C) Copyright Nokia 2014. All rights reserved.

onError	JavaScript callback function that will be called when the operation has been completed with an error. The XMLHttpRequest object is passed as a parameter.

Usage

```
function onSendDTMFClicked()
{
 var dtmfStr = document.getElementById('dtmf').value;

 Telephony.sendDTMF(dtmfStr,
  function(text)
  {
   alert(text);
  },
  function (err)
  {
   alert(err.status);
  }
 );
}
```

How to Package A WRT Application that uses the APIBridge

Applications that use the APIBridge will need to be packaged as Symbian Installable Files (.SIS). This is required in order to install the APIBridge framework and its plug-ins in the device (in case they weren't already there).

Developers should have at least limited knowledge of Symbian programming as well as the build system and compiler in order to create the installation package for their widget. The APIBridge is delivered with a template source code of a Widget Installer executable. This program will take care of installing the .wgz file after all of the required Symbian native components are installed.

Download the APIBridge

Follow the steps outlined below to create the installation package.

Creating the WgzInstaller for the widget

The template sources for creating the WgzInstaller can be found at

<APIBridge Directory>\WgzInstaller

and the project can be imported to Carbide by using the *bld.inf* in the *group* directory.

To compile this utility, developers must install the SWInstaller Plug-in to their S60 SDK. For more information go to:

SW Installer Launcher API

Follow these steps to create the WgzInstaller:

1. Open the *WgzInstaller\group\WgzInstaller.mmp* file.

- Replace the UID2 with you own UID.

2. Open the *WgzInstaller\src\WgzInstaller.cpp* file.

- Change the value of the *KWidgetInstallerFileName* Literal with the name of your .wgz file.

3. Compile the project to generate the customised *WgzInstaller.exe.*

Creating the .sis file for the widget

Once the .exe has been generated, it is time to create the installation package. Follow these steps:

1. Copy the .wgz and the *ApiBridge_vx_x_x.sis* to the *WgzInstaller/content* directory.

2. Modify the WgzInstaller*/sis/WgzInstaller_template.pkg*:

- Application name
- Installation UID

(C) Copyright Nokia 2014. All rights reserved.

- Vendor name
- APIBridge version if not correct
- .wgz file name

3. Generate the .sis file using the modified .pkg file.

4. Sign the .sis file.

The signed .sis file is ready to be installed on the device.

(C) Copyright Nokia 2014. All rights reserved.