How to port Opera Widgets to Nokia WRT

Preface

There are many different widget technologies and runtimes out there. One common denominator between many of the is the fact that they are, in effect, renamed zip packages containing common web technologies like HTML, CSS and Javascript etc.

The following contains guidelines for porting an Opera widget to the S60 Web Runtime.

Target Audience

This document is aimed at developers who wish to port their own Opera Widgets to Nokia WRT.

Widget Suitability

In order for the widget to be ported successfully, the widget must be originally designed for a mobile device with consideration for factors such as screen size, orientation and navigation methods.

Technology Compatibility

Web standard technologies

Nokia WRT supports the following technologies:

- HTML 4.01 Specification, W3C Recommendation 24 December 1999,
- XHTML Mobile Profiles Specification, XHTML Mobile Profile Version 29-Oct-2001, WAP-277-XHTMLMP-20011029-a
- CSS Level 2 revision 1 (CSS 2.1), WAP CSS Specification Version 26-Oct-2001, WAP-239-WCSS-20011026-a
- JavaScript 1.5 (ECMA-262 3rd Edition), ECMAScript Language Specification, 3rd edition (December 1999)
- The combination of XML, XHTML/HTML, CSS, DOM, and the XMLHttpRequest object to add Ajax capability to a widget

If the widget you wish to port uses technologies that are not in this list, then it will not function as intended without modifications to the source code.

Opera Widget specific Javascript APIs

If only standard Javascript has been used in the widget, then no additional Javascript libraries need to be added to the widget.

However, the Opera Widget environment provides Javascript APIs that allow widgets to access special functionality. Some of that functionality is not available in the Nokia WRT environment. If the built-in Opera Widget APIs have been used, then an additional Javascript library needs to be added to the widget. Step 4 explains how to do this.

Be aware that the library is only partially supported and as such will have reduced functionality. The following is a list of the currently supported and unsupported functionality.

Supported attributes and methods

- `identifier`
- `openURL()`
- `preferenceForKey()`
- `setPreferenceForKey()`
- `onshow`
- `onhide`

Wrapper supported attributes and methods

- `originURL`
- `widgetMode`
- `showNotification()` (no callback on acknowledgement available)
- `show()`
- `hide()`

Unsupported attributes and methods

- `getAttention()`

Opera Geolocation API

The Geolocation API supported by a Geolocation-enabled build of the Opera web browser is partially can be partially used by wrapping the Nokia Location Service Object functionality with this OperaGeolocationWrapper.txt Javascript wrapper. Remember to rename the file extension to .js before including it in your widget.

Porting other runtime specific APIs to Nokia WRT

Although beyond the scope of this guide, if there are other runtime specific APIs that the widget you wish to port needs, it may be possible to implement them by using Javascript to wrap Nokia WRT functionality inside an object that mimics the API you wish to implement. For more information about this, refer to the following article.

Implementing widget runtime APIs with Nokia WRT functionality

---

The Method

1. Unzip the Opera Widget package

The widget file will need to be unzipped to make modifications to its contents. To do this, its file extension may need to be changed to .zip before being extracted.

2. Create info.plist manifest file

When installing a .wgz file the system looks for a manifest file called info.plist. The Opera Widget equivalent is called config.xml.

Here's how to use the Opera Widget config.xml data to create the WRT manifest file.

1. **Download and unzip this** info.plist **template file and save it in the widget's root directory.**
2. **Add the `<DisplayName>` (mandatory)**
   This can be found from the <widgetname> element of the config.xml file.
3. **Add the `<Identifier>` (mandatory)**
   An identifier is not explicitly declared in the config.xml. However, Opera widgets have an optional `<id>` element which contains a child `<host>` element containing the url of the widget and it's name. These can be formatted into the same form as the conventional WRT Identifier, which is the URL in reverse with the widget name appended to the end.

   eg. `widget.example.com/MyWidget` → `com.example.widget.MyWidget`

   Otherwise, a 30 digit random integer can be assigned as the identifier, which is what Opera does when it installs a widget.

   eg. `111222333444555666777888999000`
4. **Add the `<MainHTML>` (mandatory)**
   Opera widget config.xml files contain an optional `<widgetfile>` element that contains the HTML file to be opened by the HTML renderer. If this element exists, it's value should be assigned to the `MainHTML` key in the info.plist file.

   If it does not exist, then the default value, `index.html` should be assigned to the `MainHTML` key.
5. **5. `<AllowNetworkAccess>` (optional)**
   Nokia's WRT does not allow fine tooth control over network access. Either all network resources are available, or none are.

   Opera's config.xml contains a `<security>` element that contains a child `<access>` element, the presence of which allows the widget access to network resources. If this tag is present, then the `<AllowNetworkAccess>` should contain a self closing tag `<true />`. If not, then it should be `<false />`.
6. **`<Version>` (optional)**
   This tag facilitates updating of the widget, allowing it to check if there is a newer version available. As there is no direct equivalent property in Opera Widgets, this tag can be left out.
7. **Save the file.**

3. Find, Move and Rename the icon

The icon file must be in PNG format, therefore if your icon is in GIF or SVG format, then you need to convert it to PNG.

It then needs to be saved in the root directory as icon.png.

4. Add Javascript Library (optional)

Download this OperaWidget.txt file, change the extension to .js and save it to the root directory. Add the following line to the head of your main HTML file.

```
<script type=”Javascript” src=”OperaWidget.js”/ >
```

Opera Widgets provide special functionality via a built-in object named "widget". Nokia WRT also contains a built-in object named "widget" that cannot be extended.

As a result of this, to use the functionality provided by the Opera Widget wrapper, it needs to be referenced as `operaWidget` rather than `widget`.

5. Packaging

Re-zip the containing folder and change the file extension to .wgz.

6. Deploy it

Upload the widget to your Nokia phone or to Nokia's Ovi portal to share it with the rest of the world.

Examples

- Hello World!
- Google News
- Chinese Abacus

References

- Create your first WRT widget
- How to port Adobe AIR to WRT
- How to port Bondi Widgets to Symbian Web Runtime
- Web Developer's Library 1.7
- Opera Widgets
- Opera Widgets SDK
- Opera Widgets Specification