Mobilising websites: building a WRT widget for the Nokia Developer website

This article describes the design and development process used to mobilise the Nokia Developer Community website as a Symbian Web Runtime widget.

Step 1. Choosing sections

16 Aug 2009

The Nokia Developer website offers many sections and sub-sections, with rich information covering all the aspects of mobile development. All this information is highly valuable in a standard website, since it allows users to easily find answers to all mobile-related questions and doubts. However, a mobile device represents a different environment, where users want to access information in a different way, and for different purposes:

- users want to quickly reach information, with very few interactions
- they want to easily read information, without the need to identify relevant content in a overly-populated interface
- also, not all content is suited for mobile users and devices, since too-rich or complex information is:
  - not easily representable on small displays
  - not of interest for users in mobility

For this reasons, the following sections are chosen for the mobile version of the Nokia Developer Community:

- the Discussion Boards: the discussion boards contains quite only textual data, and so are easily adaptable to mobile displays, without the need to create complex interfaces
- the Blogs: blogs are easily readable from users in mobility, since do not require high attention or concentration, and are also easily adaptable to small displays
- the Events: events, visible on Nokia Developer homepage, are few items with a short descriptive text

Step 2. Designing the user interface

The style

The Nokia Developer website has a well-known and established look-and-feel, consisting of clear colors (mainly blue and green-based) and a polite interface. When mobilizing a website, it is highly recommended to follow the style guidelines of the original website, so the Nokia Developer widget presented here will use the same style and colors of the Nokia Developer website.

Representing the sections

The widget has 3 types of content: blogs, boards and events.

- The first two, blogs and boards, are typically composed by longer textual information, structured in a descriptive title and a full HTML description, possibly containing images and other media. So, for blogs and boards it has sense to use a master-detail pattern, where the master view shows a list of titles, and the detail view shows the detail of the chosen item.
- Events, instead, are typically represented by simpler and shorter texts than the other two sections. So, in order also to create a more dynamic user interface, events will be represented by a news ticker positioned just below the widget's header, and will be always visible. So, the component shown in this Nokia Developer Wiki article can be used: News Ticker JavaScript component for Web Runtime.

First interface draft

The first user interface draft is represented by the following picture.

It has three main elements:

- a top header, containing the widget's logo and title
- a news ticker, showing the events' information
- a main view, where blogs and boards content is displayed

(C) Copyright Nokia 2013. All rights reserved.

Showing clear information

Users in mobility need clear and easily readable information. This means several things:

- colors must be carefully chosen, in order to have clearly readable text, with optimal contrast
- font size must not be too small. In order to take into account users with different visual needs, resizable fonts are an optimal choice. The approach described in this article is so used: How to dynamically change font size in Web Runtime widgets

In order to allow users to freely change the font size, a Settings view is added to the widget, that can potentially be used for other relevant actions and settings.

Different screens and input methods

Touch-enabled devices also have larger displays to represent the widget's information. For these reasons, it makes sense to design two slightly different interfaces depending on the user device:

- on touch-enabled devices, softkey option items can be replaced by buttons placed within the widget's interface
- on non touch-enabled devices, classical softkey items are used, in order to save space for the widget's main content

The layout will so be structured as described in this Nokia Developer Wiki article: How to build a Web Runtime layout with Header and Buttons Bar

The buttons' bar will contain 3 buttons, allowing users to switch between the main views of the widget:

- a button for the Blogs section
- a button for the Discussion Boards section
- a button for the Settings view

Final layout draft

The following pictures show the final widget's user interface drafts, both for small and large/touch-enabled displays:
Layout for small displays

Layout for large displays

Step 3. Retrieving information from network

 (C) Copyright Nokia 2013. All rights reserved.

The three widget's sections have RSS feeds that allow mobile applications to easily access their content. These feeds are so used to dynamically retrieve the information with Ajax network requests.

Updating the information

Information changes, and so needs to be refreshed within the widget itself. In this case there are typically two choices:

- use a "refresh/update" button to allow users to refresh the content of a view
- implement an auto-refresh mechanism that periodically retrieves and shows the updated information

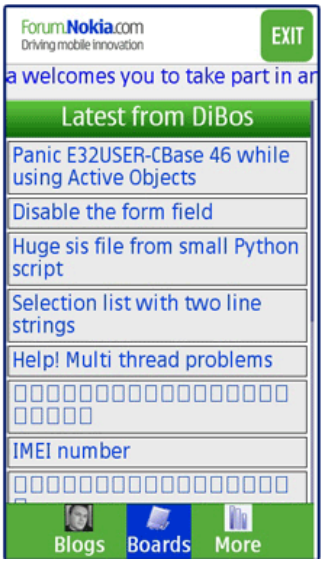
The last approach is, from a usability point of view, the best one, since allow users to have updated information without requiring any extra steps or user interactions. In order to allow the user to have control over the auto-refresh functionality, an option is added to the Settings screen, allowing users to change the refresh interval.

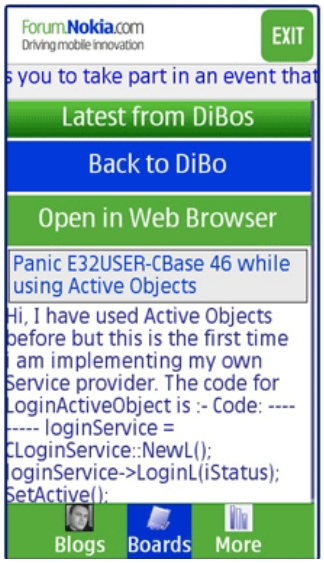
Step 4. Showing the information

Now that layout structure is ready, and information can be easily retrieved with Ajax calls, next step is to actually show the information to the user. The following screenshots show the blogs and discussion boards screenshots, of the touch-enabled version of the widget:

Views of Blogs section

Views of Discussion Boards section





In order to allow a better visualization of the retrieved RSS data, some extra handling of this data is performed:

- images are dynamically resized according to the display width
- links are modified, in order to use the widget.openURL() function, so to open them in the S60 web browser
- embedded objects are removed

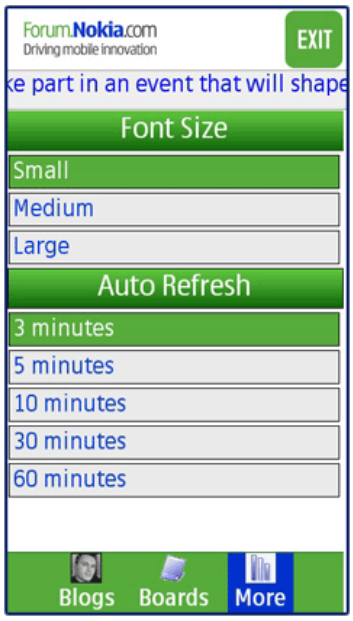
Step 5. Managing user settings

The settings view contains controls that allow users to manage the following settings:

- the font size
- the auto-refresh interval

The final layout of the settings view is shown by the following image:

(C) Copyright Nokia 2013. All rights reserved.



Since settings must be kept between different usage sessions, the widget setPreferenceForKey() and preferenceForKey() methods are used.

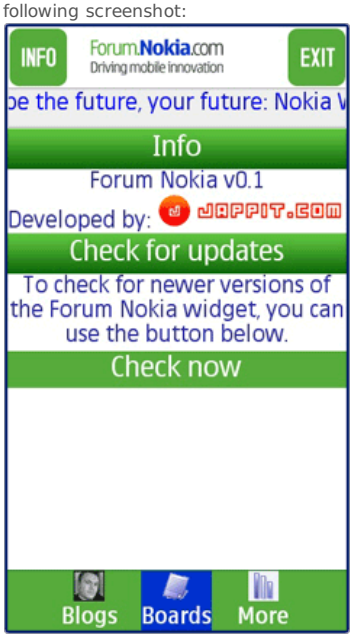
Step 6. Widget's update mechanism

Widgets, as any other mobile applications, could need to be updated in order to fix bugs, add features or simply do some code changes. In order to allow the Nokia Developer widget to be updated, two different techniques are used:

- a standard check and update mechanism, as described here: How-to create self-updating WRT widgets. With this approach, a new widget, if available, is downloaded and installed over the old version.
- an alternative code-update technique, described here: How to update WRT widgets without installing a new version. This technique allows to update the widget's code without requiring a new version to be downloaded over the old one, so improving the widget's usability and reducing network traffic.

Adding the about/update view

In order to allow the user to check for updates, another view is added to the widget. Since this view is not strictly related to widget's content, the button to access it, on touchscreen devices, is placed in the widget's header, on the right side of the Nokia Developer logo. The about screen is visible in the following screenshot:



Step 7. Usability refinements

Now that the main widget's graphics and layout are ready, some refinements can be done in order to improve the overall usability.

Adding tactile feedback

Tactile feedback allows to improve the user feelings about the interface responsiveness, by giving him immediate feedback about user interactions. In order to add tactile feedback to this widget, the following declarative approach is used: How to declaratively add tactile feedback in WRT widgets

(C) Copyright Nokia 2013. All rights reserved.

Optimizing spaces for small displays

The screenshots view in the previous sections show the widget running on touchscreen devices, with 360x640 resolution. In order to optimize spaces on devices with smaller displays, some elements can be adjusted and resized. The screenshots below show the various widget's views running in a 240x320 display:

Step 8. Create an icon

Before releasing the widget, an icon must be created, in order to identify the widget within the phone menu. An example icon (a 88x88 PNG image) is visible below:

Step 9. Deploy!

All is ready, so what remains to do is to do a lot of testing, on both touch enabled and non-touch enabled devices, to see if all works well.

Further development

Many adjustments and improvements are possible to the presented widget. Examples include:

- allowing the user to choose the discussion boards of interests, so filtering the RSS feed depending on his preferences
- adding a home screen version of the widget, that displays latest RSS data on the device home screen

Downloads

The Nokia Developer widget developed for this article is available for download here: Media:ForumNokiaWidget.zip

(C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.