HTML5 and JavaScript Framework

QNX Software Systems Limited
1001 Farrar Road
Ottawa, Ontario
K2K 0B3
Canada

Voice: +1 613 591-0931
Fax: +1 613 591-3579
Email: info@qnx.com
Web: http://www.qnx.com/

Electronic edition published: Friday, February 21, 2014

Table of Contents

About This Guide

This document describes the HTML5 development framework for the QNX CAR platform. It also includes a set of Cordova-compliant JavaScript plugins you can use in your apps.

Although you can use various technologies to build apps for your in-car system, HTML5 offers a number of advantages, including portability, flexibility, feature set, and performance.

To find out about:	See:
Overview of the tools you need	*HTML5 and JavaScript Development Tools* (p. 9)
Downloading and installing the tools needed to write HTML5 apps	*Obtaining the Development Tools* (p. 11)
Creating BAR package files for your apps	*Creating BAR packages for Cordova apps* (p. 15) and *Creating BAR packages for WebWorks apps* (p. 17)
Automotive-relevant Cordova plugins that help you write HTML5 apps	*Cordova JavaScript Plugins*
WebWorks extensions from the 2.0 release	*WebWorks JavaScript Extensions (CAR 2.0—Deprecated)*
Converting a WebWorks application to Cordova for use with the QNX CAR platform	*Porting WebWorks Applications to Cordova*

Typographical conventions

Throughout this manual, we use certain typographical conventions to distinguish technical terms. In general, the conventions we use conform to those found in IEEE POSIX publications.

The following table summarizes our conventions:

Reference	Example
Code examples	`if( stream == NULL )`
Command options	`-lR`
Commands	`make`
Environment variables	***PATH***
File and pathnames	`/dev/null`
Function names	*exit()*
Keyboard chords	**Ctrl –Alt –Delete**
Keyboard input	`Username`
Keyboard keys	`Enter`
Program output	`login:`
Variable names	*stdin*
Parameters	*parm1*
User-interface components	**Navigator**
Window title	**Options**

We use an arrow in directions for accessing menu items, like this:

You'll find the Other... menu item under **Perspective → Show View** .

We use notes, cautions, and warnings to highlight important messages:

Notes point out something important or useful.

Cautions tell you about commands or procedures that may have unwanted or undesirable side effects.

Warnings tell you about commands or procedures that could be dangerous to your files, your hardware, or even yourself.

Note to Windows users

In our documentation, we use a forward slash (/) as a delimiter in all pathnames, including those pointing to Windows files. We also generally follow POSIX/UNIX filesystem conventions.

Technical support

Technical assistance is available for all supported products.

To obtain technical support for any QNX product, visit the Support area on our website (*www.qnx.com*). You'll find a wide range of support options, including community forums.

Chapter 1
HTML5 and JavaScript Development Tools

The QNX CAR platform ships with many development aids for writing your own HTML5 apps, including a set of Cordova JavaScript plugins. You can mix HTML5 apps with apps written with other technologies, such as Qt. HTML5 apps can access PPS objects (via Cordova), as can Qt apps.

Apache Cordova is required for HTML5 application development. The Apache Cordova framework gives you access to several automotive APIs, including vehicle sensors, the navigation route, climate control, multimedia, and other useful APIs for the vehicle HMI. For the full API details, see the *Cordova JavaScript Plugins*.

This version of the QNX CAR platform also includes a set of WebWorks extensions with similar functionality, but note that these are now being deprecated. For details, see *WebWorks JavaScript Extensions (CAR 2.0—Deprecated)*.

We strongly discourage testing with the `file:///` protocol because this may cause unintended behavior.

HTML5 app support

The HMI layer of the QNX CAR platform includes a browser engine that runs HTML5 apps independently of the HMI.

This release is shipped with two versions of the HMI: a Qt5 version (the default) and an HTML5 version. You can run HTML5 apps in both HMI versions in the same manner—by launching them from the **Apps Section** screen (see "A Guided Tour of the HMI" in the *User's Guide* for details). Each HTML5 app runs in its own browser engine instance. This design provides fault isolation for the HMI and lets you run HTML5 apps regardless of which HMI version is running. The *HTML5 Developer's Guide* in the QNX SDK for Apps and Media provides more information on the browser engine.

The HMI serves as a reference UI that provides programming samples of front-end controls that allow the user to configure vehicular settings. You can customize the HMI or replace it altogether. The apps in the HTML5 version of the HMI are built with HTML5, version 2.0.1.1 of Sencha Touch 2 and with multiple versions of jQuery.

Chapter 2
Obtaining the development tools

If you're an existing customer of QNX Software Systems Ltd., use the following instructions to obtain the development tools you'll need to work with the QNX CAR platform.

To obtain and set up the tools you need for HTML5 development:

1. Download the HTML5 SDK Installation package for the QNX CAR platform from the QNX *Download* site.
2. Extract the zip file to your development folder.
3. Download and install a Webkit browser, such as Chrome, Safari, or Chromium, on your development computer.
4. If you don't have Java installed, get the latest version of Java from: *www.java.com* and install it.
5. If you don't have `node.js` (0.99 or higher) installed, follow these instructions:
   - For Ubuntu Linux, run the following command:
     ```
     sudo apt-get install nodejs
     ```
   - For Red Hat Enterprise Linux, run the following command as the root user (run `su -`) before executing the following command:
     ```
     yum install nodejs
     ```
   - For Windows:
     1. Download the installer for `node.js` from *www.nodejs.org/*.
     2. Run the installer and follow the instructions to install `node.js`.
6. Add the directory containing the packaging tools included with the QNX SDK for HTML5 to your system path, use the following instructions:
   - For Linux, run the following commands in a command line shell:
     1. `cd <HTML5_SDK_location>`
     2. `cd html5sdk`
     3. `export PATH=`pwd`/tools/packaging/linux/bin:$PATH`

     where *<HTML5_SDK_location>* is the location where you extracted the QNX HTML5 SDK.

For Linux, running these commands will temporarily modify your path until you close the console session. To permanently modify your path, add the following command to export your path to your bash profile, located at `~/.profile`:

```
exportPATH=<HTML5_SDK_location>/tools/packaging/linux/bin:$PATH
```

- For Windows, follow these steps:

  1. Open a Command Prompt window.
  2. Change to the folder where you extracted the QNX SDK for HTML5.
  3. Change to the `html5sdk` folder.
  4. Run the following command:

     ```
     set PATH=%CD%\tools\packaging\win32\bin;%PATH%
     ```

When you've completed these tasks, you'll have obtained all the tools you'll need for HTML5 development.

Chapter 3
Modifying CSS Styles

You can modify the CSS3 styles included in the QNX CAR platform by using Compass and Sass.

Overview

If you want to customize the CSS files on your system, you can start by inspecting the HTML source for each application so you can see each CSS file that is being loaded. The Web Inspector tool is ideal for this. For details, see the "Analyzing page resources" section in the "Debugging Web Apps" chapter of the *HTML5 Developer's Guide*, which is in the documentation set for the QNX SDK for Apps and Media.

The modification of CSS styles applies only to the reference applications.

You can directly access any CSS (`.scss`) files that reside on your target. For example, you'll find the base settings for the fonts and colors of the Midnight Blue HMI theme here:

```
/apps/common/themes/midnightblue/base.scss
```

Setting up Compass and Sass

To modify existing `.scss` files or to create `.scss` files of your own, we recommend using the following:

- *Compass*—an open-source CSS authoring framework
- *Sass*—a CSS extension language, which gets installed when you install Compass

In order to install and use Compass, you need to have **Ruby** already installed on your host computer. If you don't have Ruby installed, you'll find an installer (for Windows) here:

http://rubyinstaller.org/

To install Compass on your Windows host:

1. Run the following command in a Command Prompt window:

   ```
   gem install compass
   ```

   The above command will install the Sass extension language as well.
2. To verify that compass is installed, run this command:

```
compass version
```

3. To add Compass project files to your working directory, run the following command:

```
compass init
```

For more Compass commands, see the following:

http://compass-style.org/help/tutorials/command-line/

In your working directory you should now see a file called `config.rb`, which you can use to configure the output for your project. For more information on configuring Compass, see the following:

http://compass-style.org/help/tutorials/configuration-reference/

Where to go for more information

The Compass site has a set of tutorials covering configuration, commands, best practices, and more:

http://compass-style.org/help/tutorials/

The Sass site includes a comprehensive set of reference documentation covering variables, mixins, and so on:

http://www.sass-lang.com/documentation/file.SASS_REFERENCE.html

The following post also has some useful information on Compass and Sass:

Syntactically Awesome Stylesheets: Using Compass in Sass

Chapter 4
Creating BAR packages for Cordova apps

To create a Cordova BAR package file for your apps:

1. Navigate to the folder where you installed the your QNX software (***$QNX_CAR_DEPLOYMENT***, ***%QNX_CAR_DEPLOYMENT***%).
2. Go to the `html5/cordova/tools/gen-cordovapkg/` folder.
3. Run the following script to generate a BAR package for one or more apps specified:

   ```
   gen-cordovapkg.py -o output_path --qnx-sdk html5_sdk_path
   [appname_1] .. [appname_N]
   ```

   where:

   - *output_path* points to an existing directory to put the generated BAR file
   - *html5_sdk_path* is a path to the HTML5 SDK. For additional information about the HTML5 SDK, see SDK Overview.
   - *appname_1*, *appname_2*, .. *appname_N* are the applications you want to put in a BAR package

   For example:

   ```
   gen-cordovapkg.py -o /tmp/ --qnx-sdk /tmp/html5sdk Home Com
   munication
   ```

If you don't specify any apps, the script will automatically build all the Cordova apps in $***QNX_CAR_DEPLOYMENT***`/html5/apps/` for Linux and ***%QNX_CAR_DEPLOYMENT%***`\html5\apps` for Windows.

You can specify an alternative directory by using the --application-path option.

The script generates a BAR package file for each of the apps specified. For the example above, the `tmp` directory will contain two `.bar` files:

- `Home.cordova.bar`
- `Communication.cordova.bar`

For information on packaging, installing, and launching your HTML5 app, see the *Application and Window Management* guide.

Chapter 5
Creating BAR packages for WebWorks apps

To create a WebWorks BAR package file for your apps:

1. Navigate to the folder where you installed the your QNX software (***$QNX_CAR_DEPLOYMENT***, ***%QNX_CAR_DEPLOYMENT***%).
2. Go to the `html5/webworks/tools/gen-wwpkg/` folder.
3. Run the following script to generate a BAR package file for one or more apps specified:

   ```
   gen-wwpkg.py -o output_path [appname_1] .. [appname_N]
   ```

   where:

   - *output_path* points to an existing directory to put the generated BAR file
   - *appname_1*, *appname_2*, .. *appname_N* are the applications you want to put in a BAR package

   For example, the following command generates BAR package files for the Home and Communication apps:

   ```
   gen-wwpkg.py -o /tmp/ Home Communication
   ```

You can specify an alternative directory by using the --application-path option.

The script generates a BAR package file for each of the apps specified. For the example above, the `tmp` directory will contain two `.bar` files:

- `Home.bar`
- `Communication.bar`

For information on packaging, installing, and launching your HTML5 app, see the *Application and Window Management* guide.

Index