Firefox Extension Development Tutorial :: Configuration Files

Table of contents

1. Overview

A Firefox Extension is a collection of files and folders that have been compressed into a file with a .xpi extension. The .xpi file (pronounced zippy) is nothing more than a .zip file that has been renamed.

Create a folder (wherever you feel most comfortable, such as the Desktop) named 'MyExt'. Truthfully, the name of this folder is up to you, but I will refer to it as 'MyExt' throughout this tutorial. Within this folder you will need to create the following directories:

```
MyExt/
      chrome/
      chrome/chromeFiles/
      chrome/chromeFiles/content/
      defaults/
      defaults/preferences/
```

These directories will remain empty for now. There are two text files that you will need to create at the root of the MyExt folder. These files are configuration files that provide information to the Firefox extension framework. Each of these are described below.

Alternatively, you may download a MyExt.zip file that already contains the empty directories.

2. Install Manifest

An Install Manifest is the file used to provide information about a Firefox addon (extension, plugin, component, ..) while it is being installed. The file contains metadata identifying the addon, providing information about who created it, where more information can be found about it, which applications and versions it is compatible with, and more.

To create your Install Manifest, create a text file called 'install.rdf' and place it in the root of your MyExt directory (i.e. MyExt/install.rdf). Paste the following information into this file:

```
<?xml version="1.0"?>
<RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:em="http://www.mozilla.org/2004/em-rdf#">

<Description about="urn:mozilla:install-manifest">
  <em:id>sample@sample.com</em:id>
  <em:version>1.0</em:version>
  <em:type>2</em:type>

  <!-- Target Application this extension can install into,
       with minimum and maximum supported versions. -->
```

```
                <em:targetApplication>
                  <Description>
                    <em:id>{ec8030f7-c20a-464f-9b0e-13a3a9e97384}</em:id>
                    <em:minVersion>1.5</em:minVersion>
                    <em:maxVersion>1.5</em:maxVersion>
                  </Description>
                </em:targetApplication>

                <!-- Front End MetaData -->
                <em:name>My Extension</em:name>
                <em:description>A sample extension.</em:description>
                <em:creator>Your Name Here</em:creator>
                <em:homepageURL>http://www.mypage.com/</em:homepageURL>
                </Description>
              </RDF>
```

At this time you should modify this generic information to match your needs. Here is a breakdown of the first section:

1. *id*: This is this value which uniquely identifies your extension. This ID should simply be your email address. This ensures that the ID will be different than those of other extensions. It is possible to use a GUID as well, but this is no longer necessary or recommended.
2. *version*: This string identifies the version of the extension being installed. The version number is completely up to you, however it is recommended that you use versions less than 1.0 when in development stages. When you feel confident your extension works well, you may make the version be 1.0. Future improvements should increase the version number as appropriate. For Firefox 1.5, a version string consists of one or more version sections, separated with dots. Each version section is a sequence of four parts : <number-a><string-b><number-c><string-d>, where each of the parts is optional. Numbers are integers base 10, and strings are ASCII. Example: 1b2a.
3. *type*: This integer value represents the type of addon being installed. For an extension, this number should always be 2.

The targetApplication section defines which application is targeted by this extension. The <em:id> field defines which application your addon is created for, and is currently set to Firefox in the sample file. This tutorial only provides Firefox details, so this GUID should remain constant.
The <em:minVersion> and <em:maxVersion> fields simply tell Firefox which versions of Firefox the extension is designed for. These values will be compared to the value of the app.extensions.version preference and can be different from the actual version of Firefox (yes, it is annoying). For example, Firefox versions 1.0 - 1.0.6 all have app.extensions.version of 1.0.
The last section of data describes your extension:

1. *name*: This line simply defines the name of your extension and is intended for display in the UI of Firefox.

2. *description*: This line should describe the functionality of your extension and is intended to be displayed in the UI. It should fit on one line in order to display properly in the user interface.
3. *creator*: This line should contain your name and will be used to display your name in the UI.
4. *homepageURL*: This line is optional and is simply just a link to the author's homepage. If you have a personal homepage that you would like to reference, put it here.

3. Chrome

Before we talk about the Chrome Manifest file we must learn what 'chrome' means. Chrome is the term used to refer to Interface Packages created for Firefox. The Firefox browser contains a component, the Chrome Manager, that handles the installation and loading of the various parts of Firefox. Everything from the guts (global, browser, etc.) to extensions (such as yours!) register themselves with this manager.

Chrome uses URIs just like you are used to on the web (http). However, in order for Firefox to know it is working with a chrome package, the 'chrome' prefix is used (instead of http). For example, there is a built-in package called 'browser'. This package can be referenced as 'chrome://browser'.

Let's look at the browser package a little more. This Chrome package, much like your extension will, provides User Interface and backend code for Firefox. We can inspect the browser package's primary User Interface file: browser.xul. Open up your DOM Inspector and type 'chrome://browser/content/browser.xul' into the location bar. Now press the inspect button (enter will not work). From the View menu make sure that 'Browser' is checked. You can see here what looks exactly like your browser! That is because you have loaded the layout file (inside the Browser Chrome package) and are currently inspecting it.

4. Chrome Manifest

As we proceed through the tutorial you will add many lines to the chrome.manifest file located at the root of the MyExt directory. For now, we will add one line that tells Firefox where to find the content needed to display and execute your extension:

```
content   sample   chrome/chromeFiles/content/
```

This line says that for the chrome package named 'sample', Firefox can find the content files at the location chrome/chromeFiles/content which is a path relative to the location of chrome.manifest (you created this directory above).

At this time you should come up with a unique package name for your extension (instead of

'sample'). For the Home Page Scheduler extension we used 'hpsched'. In the next few steps we will create the actual content that Firefox will be looking for.

5. Further Reading

Install Manifest Details: http://developer.mozilla.org/en/docs/Install_Manifests
Chrome Manifest Details: http://developer.mozilla.org/en/docs/Chrome_Manifest
All About Chrome: http://www.mozilla.org/xpfe/ConfigChromeSpec.html