

## Firefox Extension: Firefox Toolbar Tutorial - How to make your own toolbar in Firefox/Mozilla

In addition to the Mozilla XUL read write local files tutorial - where we created a simple XUL page with a textfield, which text can be saved to your local harddisk.  
Also see: Firefox Statusbar Tutorial.

Now we create a simple toolbar for the Firefox browser. This toolbar is installed as regular extension via a XPI installer file.

The toolbar has similar capabilities as the XUL page:

When the toolbar is in its normal size, we just see one line of the text. This is useful for some important notes/tasks/todo's:

Click on the "SWITCH" button and the toolbar expands to its full size and more of the textfield is visible:

As stated before, this is extremely useful for notes/tasks/todo's, aswell as temporary text copied from webpages and temporarily stored in the textfield for later use. Just type in some text, press "SAVE" and the text is stored in a file called "mozdat\_captainbar.txt" in your Firefox profile directory.

The creation of such a toolbar is pretty straightforward:

First we need to create the directory-structure:

[project-directory] [project-directory]/chrome [project-directory]/chrome/content [project-directory]/chrome/skin (the directory skin is not really used here - i.e. stylesheets (CSS) and graphics/icons would be stored in this directory)

If you have created this directory-structure, copy the following files in their given locations:

```
[project-directory]/install.rdf
<?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>{a1577a80-
3943-4362-a66d-858f01e2196c}</em:id> <em:name>CaptainBar</em:name> <em:version>0.1</em:version>
<em:description>Toolbar for important notes</em:description> <em:file> <Description
about="urn:mozilla:extension:file:captain.jar"> <em:package>content</em:package> <em:skin>skin</em:skin>
</Description> </em:file> <em:targetApplication> <Description> <em:id>{ec8030f7-c20a-464f-9b0e-
13a3a9e97384}</em:id> <em:minVersion>1.0</em:minVersion> <em:maxVersion>1.5</em:maxVersion> </Description>
</em:targetApplication> </Description> </RDF> Notes: minVersion and maxVersion is the version information. It tells the
browser, which versions of the browser this extension supports.
```

```
[project-directory]/chrome/content/contents.rdf
<?xml version="1.0"?> <RDF:RDF xmlns:RDF="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:chrome="http://www.mozilla.org/rdf/chrome#"> <RDF:Description about="urn:mozilla:package:captain"
chrome:extension="true" chrome:name="captain"/> <RDF:Seq RDF:about="urn:mozilla:package:root"> <RDF:li
RDF:resource="urn:mozilla:package:captain"/> </RDF:Seq> <RDF:Seq RDF:about="urn:mozilla:overlays"> <RDF:li
RDF:resource="chrome://browser/content/browser.xul"/> <RDF:li
RDF:resource="chrome://navigator/content/navigator.xul"/> </RDF:Seq> <RDF:Seq
RDF:about="chrome://browser/content/browser.xul"> <RDF:li>chrome://captain/content/captainOverlay.xul</RDF:li>
</RDF:Seq> <RDF:Seq about="chrome://navigator/content/navigator.xul">
<RDF:li>chrome://captain/content/captainOverlay.xul</RDF:li> </RDF:Seq> </RDF:RDF>
```

```
[project-directory]/chrome/content/captainOverlay.xul
<?xml version="1.0"?> <overlay id="captainToolbarOverlay"
xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"> <script src="captain.js"/> <toolbarpalette
id="BrowserToolbarPalette"> <toolbaritem id="capbar" flex="1"> <label value="CaptainBar: " control="blog"/>
<toolbarseparator/> <textbox id="blog" flex="1" multiline="true"/> <toolbarbutton id="savebutton" label="SAVE"
oncommand="save();"/> <toolbarbutton id="switchbutton" label="SWITCH" oncommand="toggleheight();"/>
<toolbarbutton id="readbutton" label="READ" oncommand="read();"/> </toolbaritem> </toolbarpalette> <toolbox
id="navigator-toolbox"> <toolbar accesskey="R" hidden="false" chromeclass="toolbar" class="chromeclass-toolbar"
customizable="false" id="captaintoolbar" mode="full" toolbartname="CaptainToolbar [shift+F1]"
inherits="collapsed,hidden" persist="collapsed,hidden" defaultset="capbar" height="30"/> </toolbox> <keyset
id="mainKeyset"> <key id="key_fgtoggle" keycode="VK_F1" modifiers="shift" command="cmd_toggleToolbar"/>
</keyset> <commandset id="mainCommandSet"> <command id="cmd_toggleToolbar"
```

oncommand="goToggleToolbar('captaintoolbar','cmd\_toggleToolbar');"/> </commandset> </overlay> Notes: This is pretty straightforward XUL code. See xulplanet.com and google for more details.

```
[project-directory]/chrome/content/captain.js
window.addEventListener("load", read, false); var toolbarheightflag = false; function toggleheight() { var toolbar =
document.getElementById("captaintoolbar"); if (toolbarheightflag) { toolbar.height = 30; toolbarheightflag = false; } else {
toolbar.height = 300; toolbarheightflag = true; } } var savefile = "mozdat_captainbar.txt"; try {
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect"); } catch (e) { alert("Permission to save file
was denied."); } // get the path to the user's home (profile) directory const DIR_SERVICE = new
Components.Constructor("@mozilla.org/file/directory_service;1", "nsIProperties"); try { path=(new
DIR_SERVICE()).get("ProfD", Components.interfaces.nsIFile).path; } catch (e) { alert("error"); } // determine the file-
separator if (path.search(\\) != -1) { path = path + "\\"; } else { path = path + "/"; } savefile = path+savefile; function
save() { try { netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect"); } catch (e) { alert("Permission
to save file was denied."); } var file = Components.classes["@mozilla.org/file/local;1"]
.createInstance(Components.interfaces.nsILocalFile); file.initWithPath( savefile ); if ( file.exists() == false ) { alert(
"Creating file... "); file.create( Components.interfaces.nsIFile.NORMAL_FILE_TYPE, 420 ); } var outputStream =
Components.classes["@mozilla.org/network/file-output-stream;1"] .createInstance(
Components.interfaces.nsIFileOutputStream ); /* Open flags #define PR_RDONLY 0x01 #define PR_WRONLY
0x02 #define PR_RDWR 0x04 #define PR_CREATE_FILE 0x08 #define PR_APPEND 0x10 #define
PR_TRUNCATE 0x20 #define PR_SYNC 0x40 #define PR_EXCL 0x80 */ /* ** File modes .... ** ** CAVEAT:
'mode' is currently only applicable on UNIX platforms. ** The 'mode' argument may be ignored by PR_Open on other
platforms. ** ** 00400 Read by owner. ** 00200 Write by owner. ** 00100 Execute (search if a directory) by
owner. ** 00040 Read by group. ** 00020 Write by group. ** 00010 Execute by group. ** 00004 Read by
others. ** 00002 Write by others ** 00001 Execute by others. ** */ outputStream.init( file, 0x04 | 0x08 | 0x20, 420, 0
); var output = document.getElementById('blog').value; var result = outputStream.write( output, output.length );
outputStream.close(); } function read() { try {
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect"); } catch (e) { alert("Permission to read file
was denied."); } var file = Components.classes["@mozilla.org/file/local;1"]
.createInstance(Components.interfaces.nsILocalFile); file.initWithPath( savefile ); if ( file.exists() == false ) { alert("File
does not exist"); } var is = Components.classes["@mozilla.org/network/file-input-stream;1"] .createInstance(
Components.interfaces.nsIFileInputStream ); is.init( file,0x01, 00004, null); var sis =
Components.classes["@mozilla.org/scriptableinputstream;1"] .createInstance(
Components.interfaces.nsIScriptableInputStream ); sis.init( is ); var output = sis.read( sis.available() );
document.getElementById('blog').value = output; } Notes: This is the Javascript file for resizing the toolbar and
reading/writing to the local file to store the textbox information.
```

```
[project-directory]/chrome/skin/contents.rdf
<?xml version="1.0"?> <RDF:RDF xmlns:RDF="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:chrome="http://www.mozilla.org/rdf/chrome#"> <RDF:Seq about="urn:mozilla:skin:root"> <RDF:li
resource="urn:mozilla:skin:classic/1.0" /> </RDF:Seq> <RDF:Description about="urn:mozilla:skin:classic/1.0">
<chrome:packages> <RDF:Seq about="urn:mozilla:skin:classic/1.0:packages"> <RDF:li
resource="urn:mozilla:skin:classic/1.0:captain" /> </RDF:Seq> </chrome:packages> </RDF:Description> </RDF:RDF>
And last but not least, we need to do some zipping:
```

```
[project-directory]/make.sh
#!/bin/sh cd chrome zip -r captain.jar content/ skin/ cd .. zip captain.xpi install.rdf chrome/captain.jar
As you see, the jar file and the xpi file are just regular zip files. If you are on Windows, you need to figure out yourself
how to zip the structure via WinZIP or some command line zip tool.
```

Now we have the ready-to-install XPI file. Open the Firefox browser, and simply load the XPI file (Menu: File -> Open File). The browser will tell you that the XPI is not signed, but that doesn't matter in our case (signing the XPI will be part of a future tutorial). Restart Firefox and the toolbar should appear.

Download the ready-to-install CAPTAIN TOOLBAR - captain.xpi (5.3kB)  
(tested with Firefox 1.0.x and 1.5)

## TROUBLESHOOTING

HELP: FIREFOX DOESN'T LOAD (Segfaults):

Start Firefox in save-mode: # firefox -safe-mode In this mode no extension is loaded and the faulty extension can be removed with:

Menu: Tools -> Extensions

There is some window at startup saying the extension can't be installed:

Click on "details" and you should see something like this:

Chrome Registration failed for Extension '{a1577a80-3943-4362-a66d-858f01e2196c}' when calling

nsIXULChromeRegistry::installSkin with this chrome path: jar:file:///some-path/extensions/%7Ba1577a80-3943-4362-a66d-858f01e2196c%7D/chrome/captain.jar!/skin/ (profile extension = true). Perhaps this path does not exist within the chrome JAR file, or the contents.rdf file at that location is malformed?  
Check if all files are in the right location and if all files are actually present.

UPDATE 04-AUG-2005:

Since the overlay has no onload event, loading of the data file was done with setTimeout: self.setTimeout('read()', 10000) But we can add an EventListener with addEventListener to the window, which will trigger when the document (->our toolbar) has loaded: window.addEventListener("load", read, false); All files have been updated to reflect these changes.