

Themes and localization

Laurent Jouanneau
Course on Mozilla Education and Technologies @ Evry
October, 2009



Creative Commons Attribution-Share Alike 3.0 licence
<http://creativecommons.org/licenses/by-sa/3.0/>





Comete, 2009

Themes



Chrome url for themes

- A theme is a set of files (css, images...) acceded with this kind of URL:
 - `chrome:// « package » /skin/ « a file »`
- No need to indicate the theme you use, it automatically load the given file from the package corresponding to the active theme
- To register a set of files for a theme in the chrome.manifest file:
 - `skin « package » « theme name » « path of the directory »`

```
skin myapp classic/1.0 jar:myapp.jar!/skin/classic/myapp/
```



themes

- « classic/1.0 » is the default theme (classic.jar)
- If you want to create a new theme, you have to create all css and images like in the classic/1.0 theme
- the active theme is indicated in the preference `general.skins.selectedSkin`



Adding a CSS file

To add a CSS in a XUL file:

```
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>  
<?xml-stylesheet href="chrome://myapp/skin/mystyle.css" type="text/css"?>
```

If you want to apply a css file to an existing xul page without modifying it, add this rule in your chrome.manifest:

```
style chrome://browser/content/browser.xul chrome://myapp/skin/mystyle.css
```



Comete, 2009

Localization



Chrome url for locales

- Files dedicated to a specific lang are acceded with this kind of URL:
 - `chrome:// « package » /locale/ « a file »`
- No need to indicate the lang you want in the URL, it automatically load the given file from the package corresponding to the active language
- To register a set of files for a locale in the chrome.manifest file:
 - `locale « package » « lang code » « path of the directory »`

```
locale  myapp  en-US  jar:myapp.jar!/locale/en-US/myapp/  
locale  myapp  fr     jar:myapp.jar!/locale/fr/myapp/
```



Localizing a XUL file

- Use a DTD to store strings of a XUL file. Strings should be declared with entities
- Store this DTD into a package for locales
- Link the DTD into the XUL file
- Use the entities in the XUL file



Using a DTD

myapp.dtd

```
<!ENTITY window.title      "Super Application">
<!ENTITY main.title        "A title">
<!ENTITY summary           "this is a summary.">
```

myapp.xul

```
<?xml version="1.0"?>

<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<?xml-stylesheet href="chrome://myapp/skin/" type="text/css"?>

<!DOCTYPE window SYSTEM "chrome://myapp/locale/myapp.dtd">
<window
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
  xmlns:html="http://www.w3.org/1999/xhtml"
  title="&window.title;">
  <html:h1>&main.title;</html:h1>
  <description>&summary;</description>
</window>
```



Using several DTD a the same time

```
<?xml version="1.0"?>

<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>
<?xml-stylesheet href="chrome://myapp/skin/" type="text/css"?>

<!DOCTYPE window [

<!ENTITY % myFirstDTD SYSTEM "chrome://myapp/locale/first.dtd" >
%myFirstDTD;
<!ENTITY % mySecondDTD SYSTEM "chrome://myapp/locale/second.dtd" >
%mySecondDTD;

]>

<window
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
  xmlns:html="http://www.w3.org/1999/xhtml"
  title="&window.title;">

  ...

</window>
```



Using localized strings in javascript

- An other type of locale files: *.properties files
 - It declares keys and values. Values are localized strings
- Accessed through a string bundle object
- A string bundle is retrieved with a <stringbundle> element or with the XPCOM service "@mozilla.org/intl/stringbundle;1".
- Useful to manipulate dynamically a localized string inside a global script or inside an XPCOM component, a JSM... (ex: message of an alert box)



Properties file

myapp.properties:

```
message.hello.world = Hello World !  
status.upload.file=Upload progress: %s of %s KB
```

myapp.xul:

```
<window  
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"  
  xmlns:html="http://www.w3.org/1999/xhtml"  
  title="&window.title;">  
  
  <script type="application/javascript" src="myapp.js"/>  
  
  <stringbundle id="mainStringBundleSet">  
    <stringbundle id="app:strings"  
      src="chrome://myapp/locale/myapp.properties"/>  
  </stringbundleset>  
  
</window>
```



Retrieving a locale

With a `<stringbundle>`:

```
var sb = document.getElementById('app:strings');  
var mystr = sb.getString('message.hello.world');  
var mystr2 = sb.getFormattedString('status.upload.file', [ '34', '1290' ] );
```

With the string bundle service:

```
var sbs = Components.classes["@mozilla.org/intl/stringbundle;1"]  
           .getService(Components.interfaces.nsIStringBundleService);  
var sb = sbs.createBundle("chrome://myapp/locale/myapp.properties");  
var mystr = sb.getString('message.hello.world');  
var mystr2 = sb.getFormattedString('status.upload.file', [ '34', '1290' ] );
```