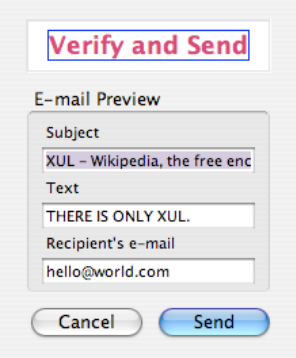
Introduction to Firefox Extensions
Section 3: Creating a Dialog Box
CS493 – University of Virginia

The full guide will lead to the creation of an extension that lets a user e-mail text that has been highlighted on a webpage. In the last section, we added hooks to the extension in the Tools and right click menus. We will now make it so that a pop-up dialog box appears when one of these menu items is clicked. The dialog box's functionality will be finished in the next section.



This dialog box will use three files: `menubox.xul` (the XUL structure code); `menu.js` (the JavaScript file); and `css.css` (the style data).

The XUL code will set up three text input fields (subject, text, and recipient's e-mail). Two of the fields (subject and text) will be initialized to arguments retrieved by the JavaScript code. The behavior for Cancel and Send is also determined by the JavaScript code. The style sheet is used to change the text color of the dialog title and demonstrate the relationship between CSS and XUL items.

To begin, change the definition of `pop()` in `example.xul` to:

```
<script type="application/x-javascript">
    function pop() {
        window.openDialog(
            'chrome://example/content/menubox.xul',
            'showmore',
            'chrome',
            'hello');
    }
    </script>
```

This code will load a new XUL file for the dialog window and pass it the argument string "hello". The XUL file is found by Firefox by the path that was set in the first section of this guide. You should make a new file, `menubox.xul`, in `~/extensions/chrome/content/`. The code for `menubox.xul` follows, with an explanation beneath.

```
1   <?xml version="1.0"?>
2   <?xml-stylesheet href="chrome://global/skin/global.css" type="text/css"?>
3   <?xml-stylesheet href="chrome://example/content/css.css" type="text/css"?>
4
5   <dialog id="checkemail" title="Example Dialog"
6       xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul"
7       buttons="accept,cancel"
8       buttonlabelcancel="Cancel"
9       buttonlabelaccept="Send"
10      ondialogaccept="return onSend();"
11      ondialogcancel="return onCancel();">
12
13      <dialogheader title="Example Dialog" description="Verify and Send"
```

```
14  id="title_header" />
15      <groupbox>
16          <caption label="E-mail Preview"/>
17          <label id="subject" value="Subject" />
18              <textbox id="user_sub" />
19          <label id="text" value="Text" />
20              <textbox id="user_txt" />
21          <label id="em" value="Recipient's e-mail" />
22              <textbox id="email" value="hello@world.com" />
23      </groupbox>
24
25      <script type="application/x-javascript"
26  src="chrome://example/content/menu.js"></script>
27
28  </dialog>
```

Lines 13-23 are the core of the dialog. This code section establishes the dialog elements. Each `label` corresponds to a line of text and each `textbox` corresponds to an input text section. JavaScript and CSS can refer to the elements by their IDs.

You should place the CSS file – `css.css` – at `~/extensions/chrome/content/`, as specified by the path in line 3. As shown in the screen shot at the beginning of this guide, this CSS makes the text "Verify and Send" be large, bold, and pink with a white background. It does this by editing the appearance of the element with the type `dialogheader` and ID `title_header`. CSS can be applied to other elements in the same fashion.

```
dialogheader#title_header {
    background-color: #ffffff;
    color: #cc3366;
    font-weight: bold;
    font-size: 18px;
}
```

The JavaScript file `menu.js` is imported on lines 25 and 26. By the pathname there, you should place `menu.js` in the same directory as all of the other files, with the following code.

```
1   var txt = window.arguments[0];
2
3   function onSend() {
4       alert("Send: " + txt);
5       return true;
6   }
7
8   function onCancel() {
9       alert("Cancel: " + txt);
10      return true;
11  }
```

Line 1 fetches the argument we passed to the dialog window's constructor in `example.xul`'s `pop()` function. The functions `onSend()` and `onCancel()` will display that argument in a pop-up alert box. Lines 10 and 11 in `menubox.xul` make associate these functions with the dialog box's Send and Cancel buttons. In the next section, we will make this JavaScript a little more fancy.