**Web Profiling and Scoping Tool – New Extension Integrations**

**Important Components of the Chrome extension**

* **Business Logic:**

It is an important part of any application which needs to be intact and latest. It is suggested to avoid making any direct changes to business logic. That way extension files can be directly updated with latest files if any revisions come. If anything needs to be specific try to overwrite extensions original behavior in custom files.

* **Background HTML/js :**

Almost every chrome extension needs background page. It is basically an HTML page which imports required javascripts. These javascript files run every time new url is visited.

* **Communicate with background page:**

Our extension need to talk to background javascripts to get access to gathered data. Chrome provides a few APIs which can be directly employed to do the same.

* **User Interface:**

Most of the extensions has user interface. The collected data is rendered on the extension page.

* **Manifest.Json:**

It is a configuration file for the extension. It holds settings for manifest information, background page, tab permissions and security policies.

{

 "manifest\_version": 2,

 "name": "Getting started example",

 "description": "This extension shows a Google Image search result for the current page",

 "version": "1.0",

 "browser\_action": {

 "default\_icon": "icon.png",

 "default\_popup": "popup.html"

 },

 "permissions": [

 "activeTab",

 "https://ajax.googleapis.com/"

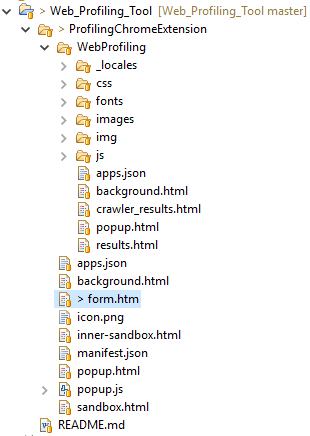
     ]

}

Few important elements of Manifest.Json:

* + manifest\_version: version of manifest file
  + name: name of the extension
  + default\_icon: icon displayed next to the Omnibox, waiting for user interaction.
  + default\_popup: extension landing page
  + page: background page
  + permissions: specific permissions for extension
  + content\_security\_policy: custom content security policies for extension

**Project Directory Structure**



Each folder contains specific files. Manifest file should be at root level, whenever extension is loaded chrome searches for manifest.json to read defined configuration. Without valid manifest.json one cannot load chrome extension.

**Steps to integrate new extension**

*Let’s assume we are integrating extension demo in our extension.*

* Get the latest code base and try to bundle required javascript files in extension specific folder under **“js”** directory (demojs).
* Most of the extensions require background javascript, try to locate this file in demo extension and import into our background.html page. Doing so extension can load appropriate file on startup.
* Each extension would have the code to carry out communication between user interface and background javascripts. Try to locate that piece of code and integrate as is to avoid complications. Following code snippets shows how to establish this link and communicate over the same.

Sending message to specific tab.

chrome.tabs.sendMessage(

TABID,

*//message to be sent*

{action: "demo"}

);

Sending message to chrome runtime i.e. background javascript.

chrome.runtime.sendMessage({

*//message to be sent*

});

Listing to messages from tabs in background javascript.

chrome.extension.onMessage.addListener(function(request, sender, sendResponse) {

*//handle incoming message*

});

Listening to messages from chrome runtine on tab.

chrome.runtime.onMessage.addListener(function (request, sender, sendResponse) {

*//access and manipulate data message as required*

});

* Locate html page within the extension, try to understand the user interface and javascript. Modify these files as per the requirement.
* Design custom user interface, use above code to send and receive data between extension and chrome runtime (which has access to background javascripts, it conveys incoming messages to corresponding listener).

**Debugging**

Debugging chrome extension page is quite simple and straightforward. Right click on extension page and click on inspect page.

To debug background javascripts, go to chrome setting, click on extension, locate our extension and find background.html.  
Click on background.html (first html file in “Inspect view : “). It will open chrome developer tool for background page from which all background javascripts are loaded.