Create Your First App

This tutorial walks you through creating your first Chrome App. Chrome Apps are structured similarly to

extensions so current developers will recognize the manifest and packaging methods. When you're done, you'll just need to produce a zip file of your code and assets in order to **publish** your app.

A Chrome App contains these components:

- The **manifest** tells Chrome about your app, what it is, how to launch it and the extra permissions that it requires.
- The **background script** is used to create the event page responsible for managing the app life cycle.
- All code must be included in the Chrome App package. This includes HTML, JS, CSS and Native Client modules.
- All **icons** and other assets must be included in the package as well.

API Samples: Want to play with the code? Check out the **hello-world** sample.

Step 1: Create the manifest

First create your `manifest.json` file (**Formats: Manifest Files** describes this manifest in detail):

```
{
  "name": "Hello World!",
  "description": "My first Chrome App.",
  "version": "0.1",
  "manifest_version": 2,
  "app": {
    "background": {
      "scripts": ["background.js"]
    }
  },
  "icons": { "16": "calculator-16.png", "128": "calculator-128.png" }
}
```

Important: Chrome Apps **must** use **manifest version 2**.

Step 2: Create the background script

Next create a new file called `background.js` with the following content:

```
chrome.app.runtime.onLaunched.addListener(function() {
  chrome.app.window.create('window.html', {
    'outerBounds': {
      'width': 400,
      'height': 500
    }
  });
});
```

In the above sample code, the **onLaunched event** will be fired when the user starts the app. It then immediately opens a window for the app of the specified width and height. Your background script may contain additional listeners, windows, post messages, and launch data, all of which are used by the event page to manage the app.

Step 3: Create a window page

Create your `window.html` file:

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <div>Hello, world!</div>
  </body>
</html>
```

Step 4: Create the icons

Copy these icons to your app folder:

- **calculator-16.png**
- **calculator-128.png**

Step 5: Launch your app

Enable flags

Many of the Chrome Apps APIs are still experimental, so you should enable experimental APIs so that you can try them out:

- Go to **chrome://flags**.
- Find "Experimental Extension APIs", and click its "Enable" link.
- Restart Chrome.

Load your app

To load your app, bring up the apps and extensions management page by clicking the settings icon  and choosing **Tools > Extensions**.

Make sure the **Developer mode** checkbox has been selected.

Click the **Load unpacked extension** button, navigate to your app's folder and click **OK**.

Open new tab and launch

Once you've loaded your app, open a New Tab page and click on your new app icon.

Or, load and launch from command line

These command line options to Chrome may help you iterate:

- `--load-and-launch-app=/path/to/app/` installs the unpacked application from the given path, and launches it. If the application is already running it is reloaded with the updated content.
- `--app-id=ajjhbohkjpincjgiieeomimlgnll` launches an app already loaded into Chrome. It does not restart any previously running app, but it does launch the new app with any updated content.

Back to top

content bug