

Creating a Chrome Extension

A Browser Action Extension

John Sonmez

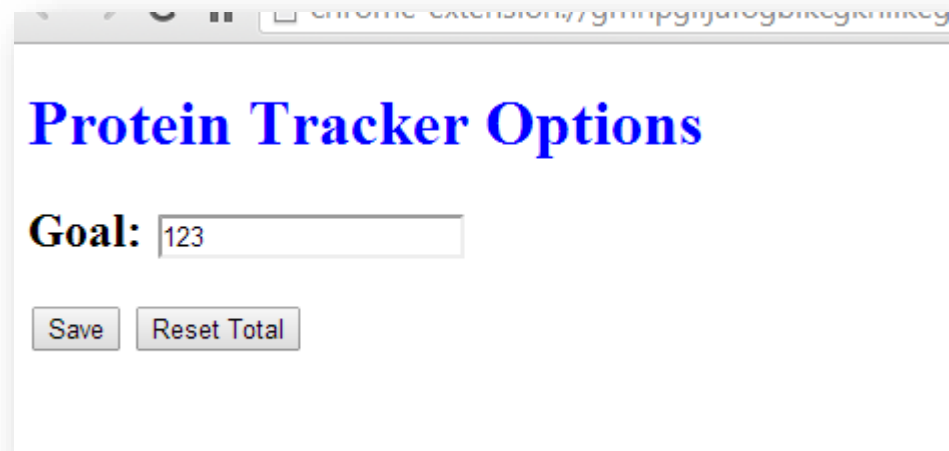
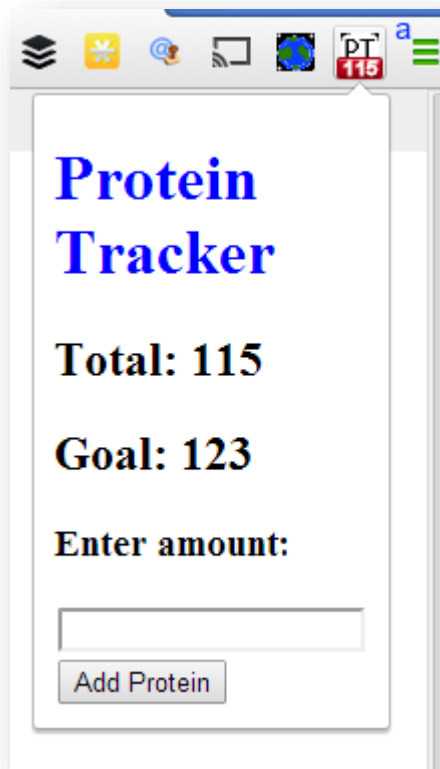
<http://simpleprogrammer.com>

@jsonmez



pluralsight 
hardcore developer training

What We Are Going To Build



Tracking protein right inside your browser

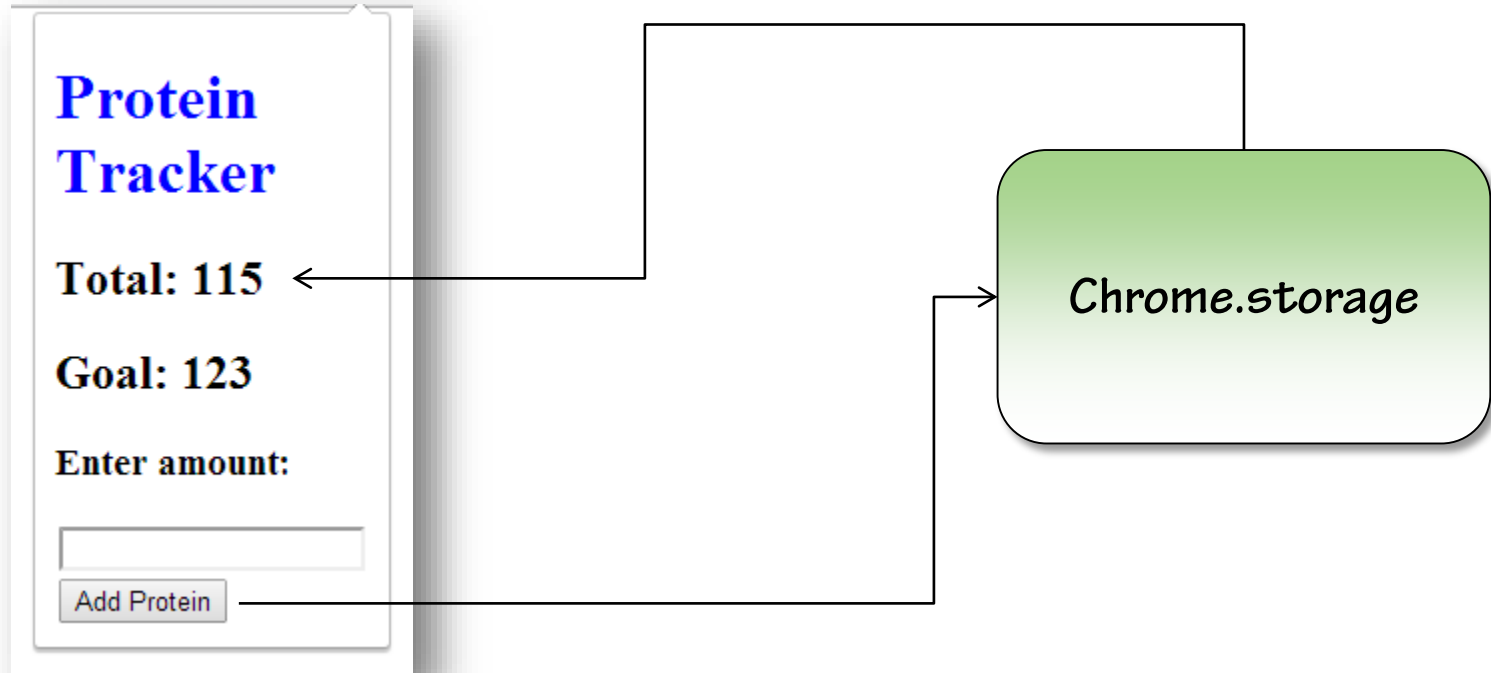
Breaking It Down

1. Create directory and manifest
2. Create popup UI
3. Add protein total saving and displaying
4. Create an options page
5. Add notifications for reset and goal met
6. Add badge to display current total
7. Add context menu item



We'll be adding much more functionality to this extension

Adding Protein



Chrome.storage provides an abstraction over local storage and lets us sync data

Options Page

Protein Tracker Options

Goal:

In the manifest we can specify an options page for our extension

Event Page

The image shows a screenshot of the Chrome Extensions documentation page for Event Pages. A green rounded rectangle labeled `eventPage.js` has two lines extending from it. One line points to the 'Event Pages' link in the 'Developer's Guide' sidebar. The other line points to the 'Manifest' section in the main content area. In the 'Manifest' section, a context menu is open over the `"scripts": ["eventPage.js"]` line in the manifest JSON, with the 'Inspect element' option selected. The manifest JSON snippet is as follows:

```
{
  "name": "My extension",
  ...
  "background": {
    "scripts": ["eventPage.js"],
    "persistent": false
  }
}
```

The documentation page also includes a sidebar with sections like 'Getting Started', 'Overview', 'What's New?', and 'Developer's Guide'. The main content area explains that Event Pages are very similar to background pages but are only loaded when needed, freeing memory and other system resources. It also mentions that Event Pages are available in the stable channel as of Chrome 42.

We'll use an event page to register to events and add a context menu item

Up Next

Getting Started

Overview

What's New?

Developer's Guide

Browser UI

Browser Actions

Context Menus

Desktop Notifications

Omnibox

Options Pages

Override Pages

Page Actions

Browser Interaction

Implementation

Finishing

Tutorials

Manifest V2

Debugging

Google Analytics

chrome.pageAction

Description: Use the `chrome.pageAction` API to put icons inside the address bar. Page actions represent actions that can be taken on the current page, but that aren't applicable to all pages.

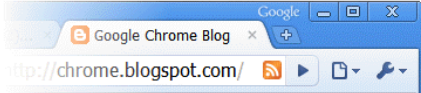
Availability: Stable since Chrome 5.

Manifest: `"page_action": {...}`

Some examples:

- Subscribe to this page's RSS feed
- Make a slideshow out of this page's photos

The RSS icon in the following screenshot represents a page action that lets you subscribe to the RSS feed for the current page.



If you want the extension's icon to always be visible, use a [browser action](#) instead.

Manifest

Register your page action in the [extension manifest](#) like this:

Manifest

Parts of the UI

Tips

Examples

Reference

Types

ImageDataType

Methods

show

hide

setTitle

getTitle

setIcon

setPopup

getPopup

Events

onClicked

Sample Extensions

We'll be creating a Page Action extension that makes use of content scripts