Developing Google Chrome Extensions

Brian Kennish
November 10, 2009
Agenda

- Introduction
  - What Extensions Are
  - Why You Should Work on Extensions
  - When the Extension System Ships
- How to Build Extensions
  - Technical Overview
  - Step-by-Step Example
- Summary
- Q&A
Introduction
What Extensions Are

- Programs that modify and enhance Google Chrome's functionality
- Written in HTML, CSS, and JavaScript
- Integrated with browser features using a simple API
- Developed iteratively as webpages
What Extensions Are

- Installed instantly, without a browser restart
- Updated automatically like Google Chrome itself
- Transparent about their cross-origin and browser capabilities
- Run in separate processes like Google Chrome tabs
Demo: Gmail Checker

Shows how many unread messages are in your inbox.
Demo: Gmail Checker

Shows how many unread messages are in your inbox.
Demo: Subscribe in a Feed Reader

Displays a subscription button when a page has an available feed.
Demo: Subscribe in a Feed Reader

Displays a subscription button when a page has an available feed.
Demo: Qrome

Turns URLs and other text into QR codes to make them easy to transfer to mobile devices.
Demo: Qrome

Turns URLs and other text into QR codes to make them easy to transfer to mobile devices.
Why You Should Work on Extensions

- Part of an important and fast-growing platform
- Persistent presence on users' machines
- Source of traffic to your site
- Easy and fun
When the Extension System Ships

- In the Google Chrome Dev channel right now
- In the Beta channel later this quarter, along with a gallery
- In the Stable channel soon after
How to Build Extensions
Structure of an Extension

Compressed directory containing:

- `manifest file` (**manifest.json**) — metadata that describes the extension
Structure of an Extension

And at least one of these components:

- **browser action** or **page action** — UI surface
- **content scripts** — CSS and JavaScript injected into pages
- **background page** — long-running script that handles tasks or state
- **utility web files** — additional content
Structure of an Extension

And at least one of these components:

- **plugins** — NPAPI binaries (see https://developer.mozilla.org/en/Plugins)
- **theme** — custom browser skin (see http://code.google.com/chrome/extensions/themes.html)
Extension Communication

Internal:

External:

- Cross-origin XHR (requires permission)
chrome is the top-level object and exposes:

- chrome.extension.* — sends extension messages and resolves the URLs of extension files
- chrome.browserAction.* — sets the appearance of browser actions and their badges
- chrome.pageAction.* — enables and disables page actions
Layout of the Extension API

`chrome` is the top-level object and exposes:

- `chrome.windows.*` — manages windows (requires `tabs` permission)
- `chrome.tabs.*` — manages tabs (requires `tabs` permission)
- `chrome.bookmarks.*` — manages bookmarks (requires `bookmarks` permission)
Other APIs

Extensions can also access:

- standard DOM and JavaScript APIs (e.g., HTML traversal and manipulation)
- HTML5 APIs (e.g., localStorage)
- WebKit APIs (e.g., experimental CSS properties)
Other APIs

Extensions can also access:

- V8 APIs (e.g., JSON parsing and stringification)
- bundled JavaScript libraries (e.g., jQuery)
- more (e.g., Google AJAX APIs)
Step-by-Step Example: Chritter

A Twitter button for your toolbar.
Step One
Add UI

```json
{
   "name": "Chritter",
   "version": "1.0",
   "description": "A Twitter button for your toolbar.",
   "icons": {
      "128": "icon.png"
   },
   "browser_action": {
      "default_icon": "browseraction.png",
      "default_title": "Chritter",
      "popup": "popup.html"
   }
}

manifest.json
Step Two
Fetch Public Data with XHR

```javascript
req = new XMLHttpRequest();
req.open(
  'GET',
  'http://twitter.com/statuses/public_timeline.json'
);
req.onload = processTweets;
req.send();
```
Step Three
Refactor Non-Presentation Code

```javascript
var res = JSON.parse(req.responseText);
unreadCount += res.length;

if (unreadCount > 0) {
    chrome.browserAction.setBadgeBackgroundColor({
        color: [255, 0, 0, 255]
    });
    chrome.browserAction.setBadgeText({
        text: '' + unreadCount
    });
}

tweets = res.concat(tweets);
```

background.html
Step Four
Detect Successful Authorization and Fetch *Private* Data

```javascript
// look for oauth_pin
var pin = document.getElementById('oauth_pin');

// send pin to extension
var port = chrome.extension.connect();

if (pin) {
    pin = pin.innerHTML.replace(/\s*|\s*$/g, '');
    port.postMessage({success: true, pin: pin});
} else { port.postMessage({success: false}); }
```
Step Five
Autoupdate to a New Version

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<gupdate
   xmlns="http://www.google.com/update2/response"
   protocol="2.0">
   <app appid="loogiogdnjdgdnmbjdjbjbbonkcfpnjdp">
      <updatecheck
         version="6.0"
         codebase="http://localhost/chritter/6.crx"/>
   </app>
</gupdate>
```
Summary
Key Takeaways

- Small learning curve
- Over 30-million active Google Chrome users
- Upcoming GTUG and similar community events
Q&A
Online Resources

- Documentation: http://code.google.com/chrome/extensions/
- Blog: http://blog.chromium.org/
- Discussion group: http://groups.google.com/group/chromium-extensions