



Chrome Extension Project Structure

Modern Manifest V3 Chrome extension with popup, content scripts, and service worker background script.

Updated 2025-12-22

#chrome #extension #browser #javascript #manifest-v3

PNG

PDF

Copy

Prompt

Project Directory

my-extension/

- manifest.json Extension confi...
- src/ Source code
 - background/ Service worker
 - service-worker.js Background scri...
 - messages.js Message handlers
 - alarms.js Scheduled tasks
 - content/ Content scripts
 - content.js Injected into p...
 - content.css Injected styles
 - popup/ Browser action ...
 - popup.html
 - popup.js
 - popup.css
 - options/ Extension setti...
 - options.html
 - options.js
 - options.css
 - lib/ Shared utilities
 - storage.js chrome.storage ...
 - api.js External API ca...
 - constants.js
 - assets/ Static resources
 - icons/
 - icon-16.png
 - icon-48.png
 - icon-128.png
 - images/
 - _locales/ i18n translati...
 - en/
 - messages.json Default locale
 - package.json
 - .gitignore
 - README.md

Why This Structure?

This structure follows Manifest V3 requirements: a service worker replaces persistent background pages, content scripts are clearly separated, and the popup/options UI lives in dedicated folders. The `src/lib/` folder centralizes shared utilities like storage wrappers.

Key Directories

src/background/ - Service worker with modular message handlers

src/content/ - Scripts and styles injected into web pages

src/popup/ - Browser action popup (HTML/JS/CSS)

src/lib/ - Shared utilities across all contexts

Manifest Structure

```
// manifest.json (key fields)
{
  "manifest_version": 3,
  "name": "__MSG_extensionName__",
  "version": "1.0.0",
  "action": { "default_popup": "src/popup/popup.html" },
  "background": { "service_worker": "src/background/service-worker.js" },
  "permissions": ["storage", "activeTab"],
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["src/content/content.js"]
  }]
}
```

Getting Started

- Create project folder and `manifest.json`
- Add icons in `assets/icons/` (16, 48, 128px)
- Open `chrome://extensions` in Chrome
- Enable Developer mode and click Load unpacked
- Select your project folder

When To Use This

- Adding features to websites you visit
- Automating browser tasks
- Modifying page content or appearance
- Building productivity tools
- Creating developer tools

Manifest V3 Concepts

Service Workers - Replace background pages, wake on events, sleep when idle

Content Scripts - Isolated world, communicate via `chrome.runtime.sendMessage`

Permissions - Request minimal permissions, prefer `activeTab`

Host Permissions - Declared separately from API permissions in V3

Best Practices

- Use `chrome.storage.local` instead of `localStorage`
- Prefer `activeTab` permission over `*`
- Split service worker into modules for maintainability
- Use `chrome.i18n` for user-facing strings
- Handle service worker wake-up gracefully

Message Passing

Use `chrome.runtime.sendMessage` for popup/options to background. Use `chrome.tabs.sendMessage` for background to content scripts. Content scripts can respond with `sendResponse` or return a Promise.

Trade-offs

No bundler - Simple setup, but no TypeScript or modern imports

Service worker limits - Can't use DOM APIs, limited to 5 min active time

No persistent state - Service worker terminates, must use storage