

Expanding the browser experience with web extensions

By [Chen Hui Jing](#) / [@huijing.bsky.social](#)



Surname First name

陈	慧	晶
Chen	Hui	Jing

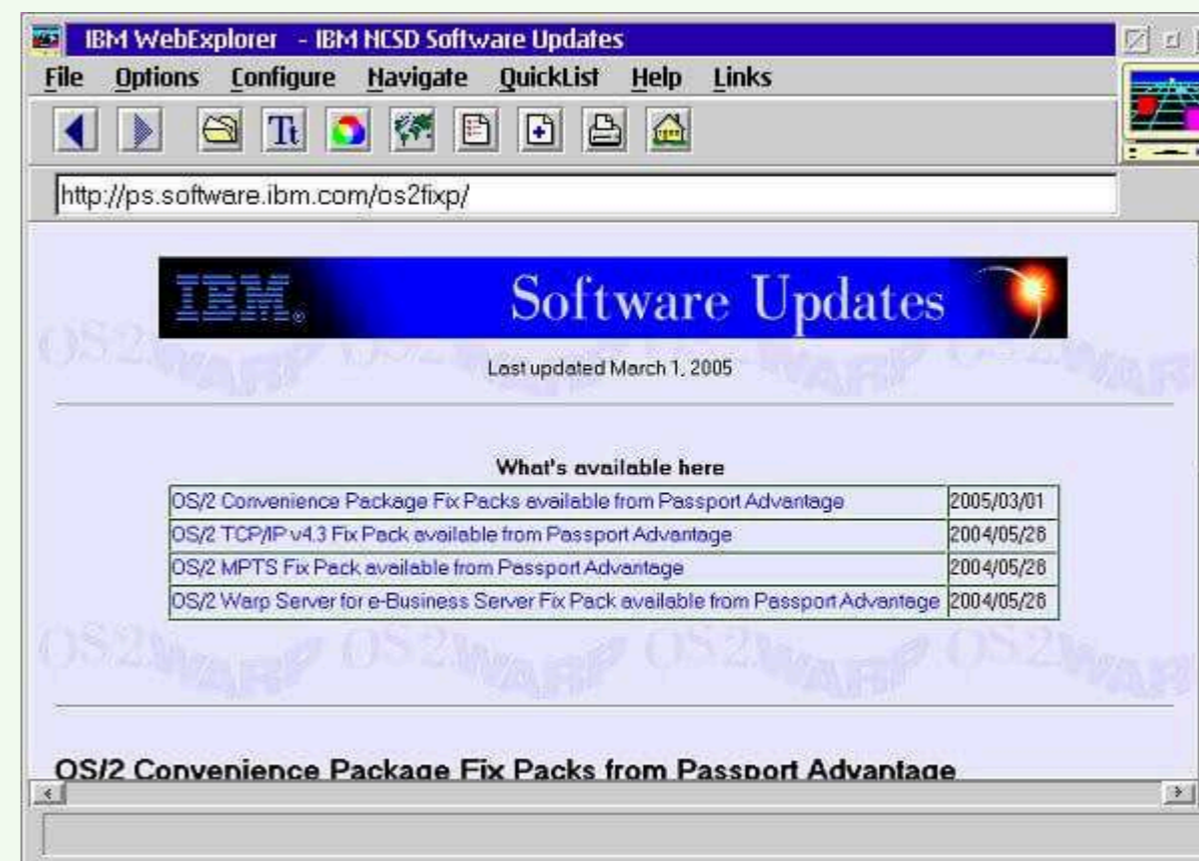
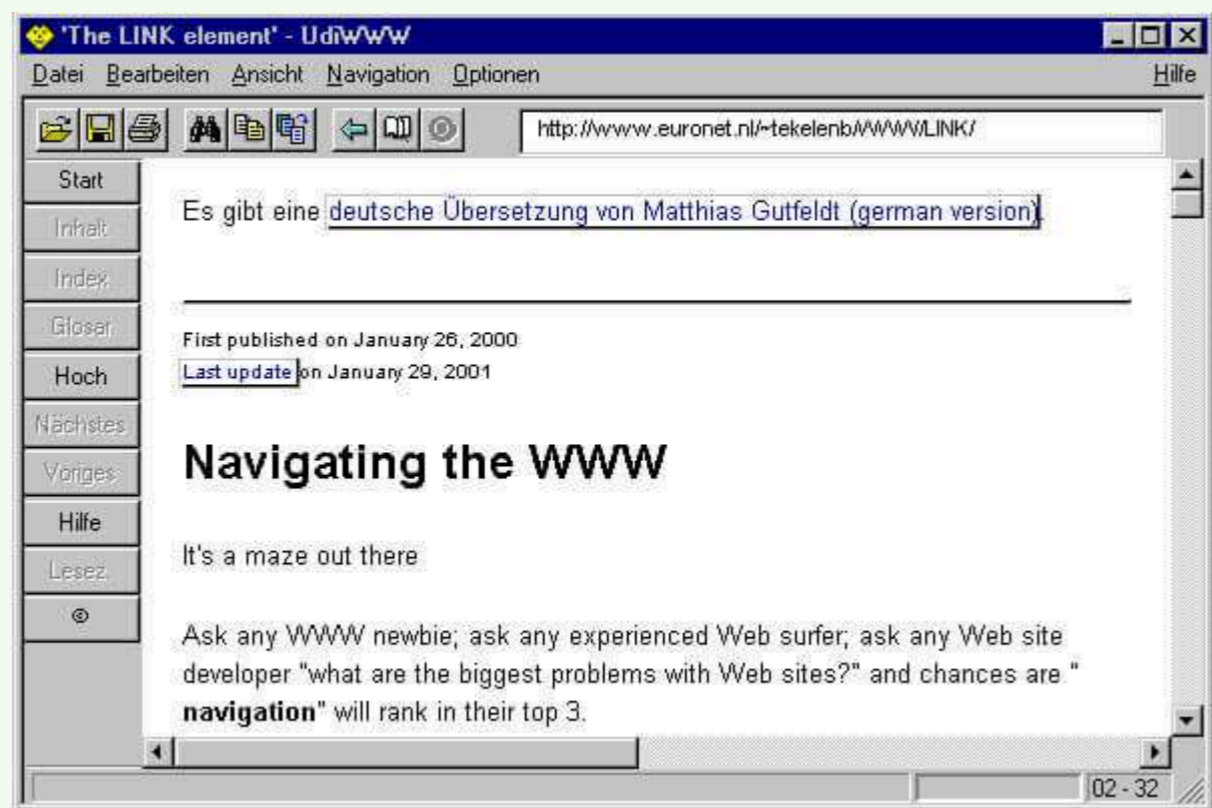
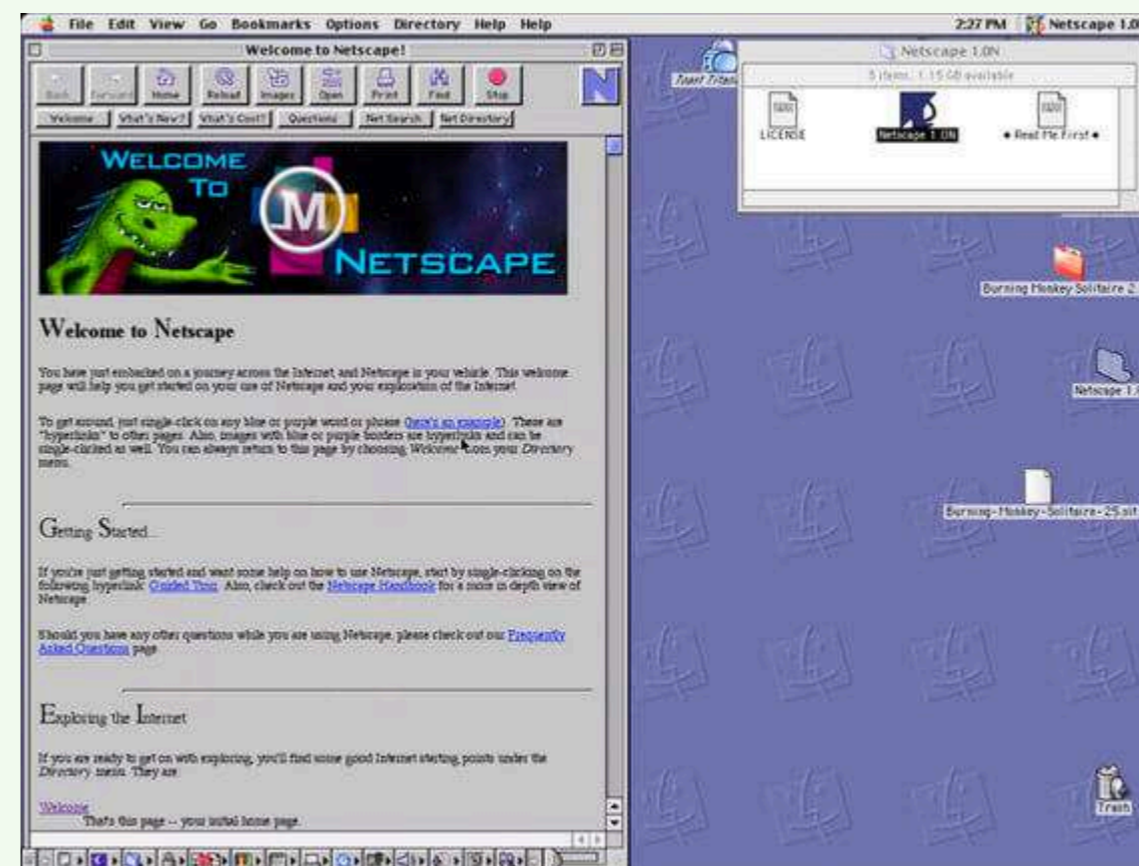


@huijing.bsky.social

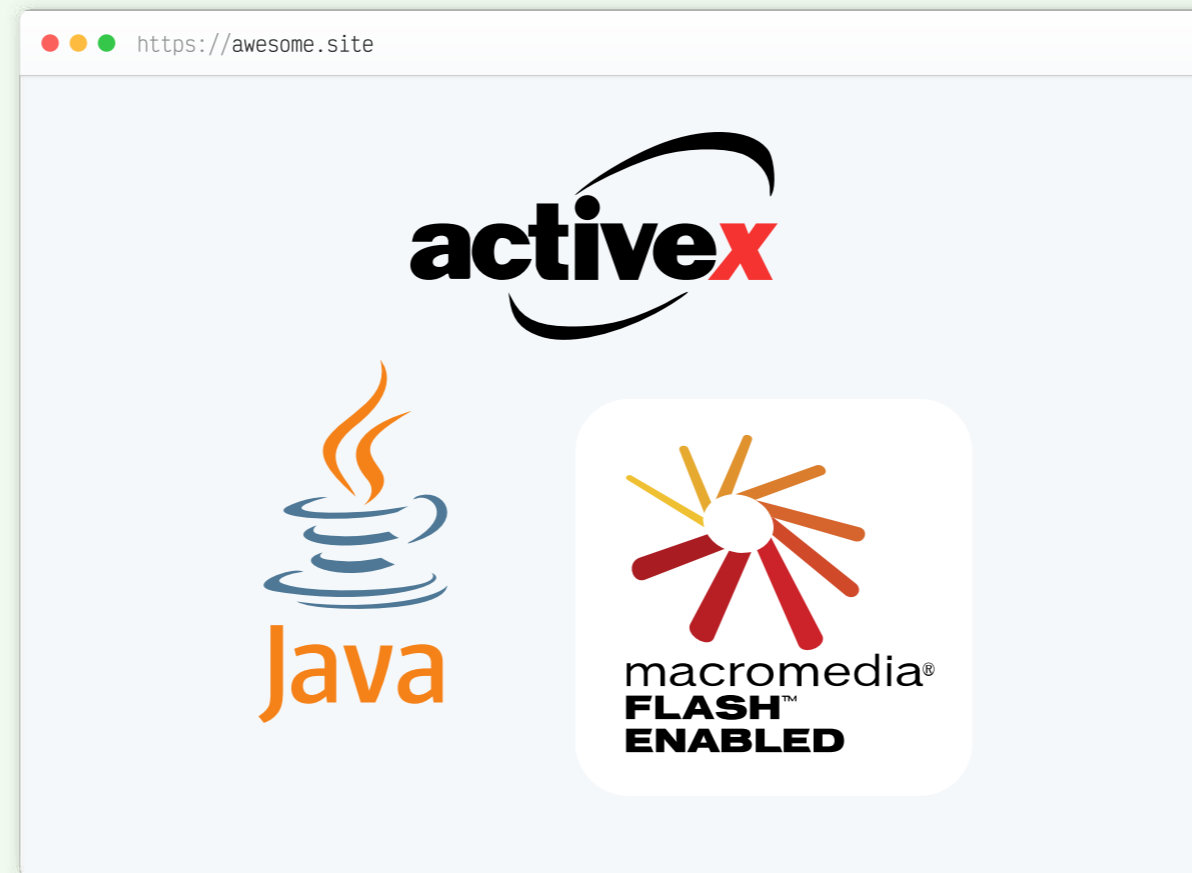




Interledger
FOUNDATION



Plug-ins





Download Help

Download Java for Windows

Recommended Version 7 Update 25 (filesize: 882 KB)



ORACLE

Status: Installing Java



3 Billion Devices Run Java

Computers, Printers, Routers, Cell Phones, BlackBerry.

Google Chrome Help

Java Setup - Progress

Chrome

Help Res

What

Remo

Disab

Error M

Troub

Other

Offline

Troub

USER

clicking

Trouble
Try the

Kindle, Parking Meters, Public Transportation Passes, ATMs,
Credit Cards, Home Security Systems, Cable Boxes, TVs...

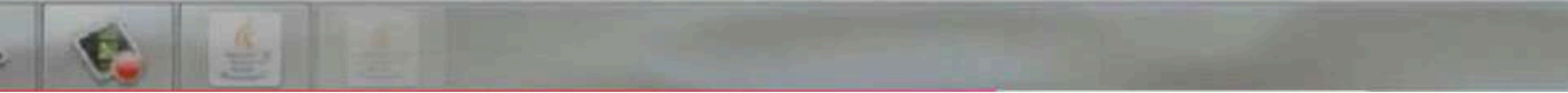
ORACLE[®]

rowser

VM,

[Select Language](#) | [About Java](#) | [Support](#) | [Developers](#)
[Privacy](#) | [Terms of Use](#) | [Trademarks](#) | [Disclaimer](#)

ORACLE[®]



User stylesheets

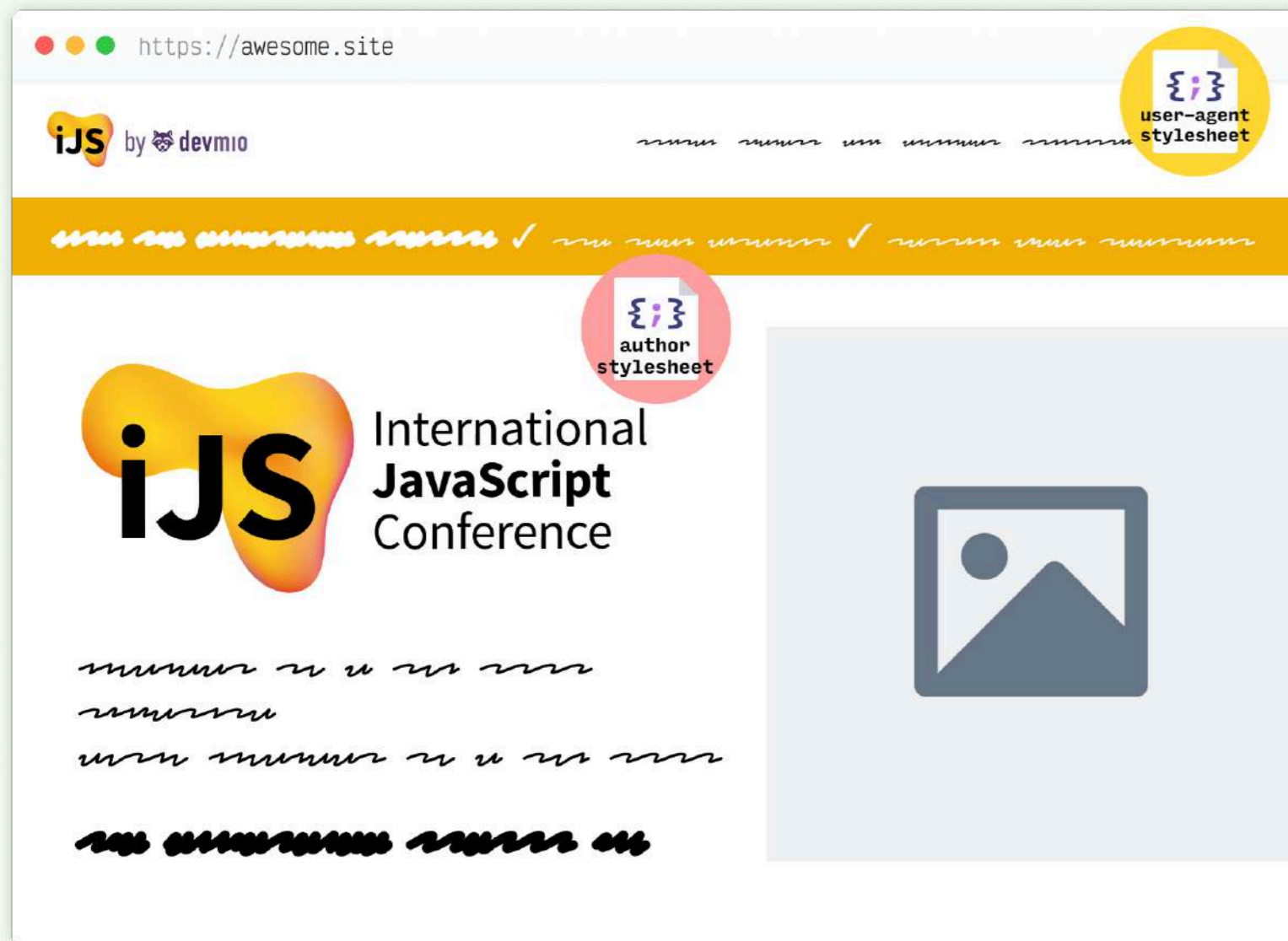
§ 6.2. Cascading Origins

Each style rule has a cascade origin, which determines where it enters the cascade. CSS defines three core origins:

Author Origin: The author specifies style sheets for a source document according to the conventions of the document language.

User Origin: The user may be able to specify style information for a particular document.

User-Agent Origin: Conforming user agents must apply a default style sheet (or behave as if they did).



Bookmarklets



Welcome to Microsoft's Homepage - Microsoft Internet Explorer

File Edit View Go Favorites Help


Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail Print Edit

Address <http://theoldnet.com/get?url=microsoft.com&year=1998&scripts=false&decode=false> Links

microsoft.com Home All Products | Support | Search | microsoft.com Home

Microsoft

Home | Events | Training | Downloads | Newsletters | International | About Our Site |

 Internet Explorer
[Download it free!](#)

Product Families
[BackOffice](#)
[Developer Tools](#)
[Office](#)
[MSN](#)
[Windows](#)

Business Solutions
[Industries](#)
[Small Business](#)

Developers
[Software Developers](#)
[Web Site Builders](#)


Education
[Academic Products](#)
[Education Resellers](#)
[Higher Education](#)
[K-12 Education](#)

IT Professionals
[Digital Nervous System](#)
[IT Professionals/Execs](#)
[Solution Providers](#)
[Year 2000](#)

Partners/Resellers
[Becoming a Partner](#)
[Find a Services Partner](#)
[Resellers Consultants](#)

Personal Use
[Games](#)
[Kids](#)
[Personal Computing](#)
[Seniors](#)

About Microsoft
[Company Overview](#)
[Jobs](#)
[Press Information](#)
[Privacy/Security](#)
[Stockholder Information](#)
[US Offices & Web Sites](#)



Join the Microsoft Office 2000 Preview Program
 You'll receive a pre-release version of Office 2000 Premium, extensive evaluation materials, and access to private product support newsgroups. The Consumer Preview Program is for Office enthusiasts and the Corporate Preview Program is for IT professionals.

order yours today

Microsoft Unveils Complete Television Software Platform and Services for Cable Industry

Build Database Applications with Visual FoxPro 6.0

Point, Click, Shop with the MSN Shopping Holiday Gift Finder

Thinking Globally: How Microsoft Localizes for the Web


Schools: Tune in to Y2K Interactive Teleconference on Dec. 7

Read About the Microsoft Trial

[Subscribe](#) to Our Free E-Mail Newsletter!

For a text-only version of the home page [click here](#).

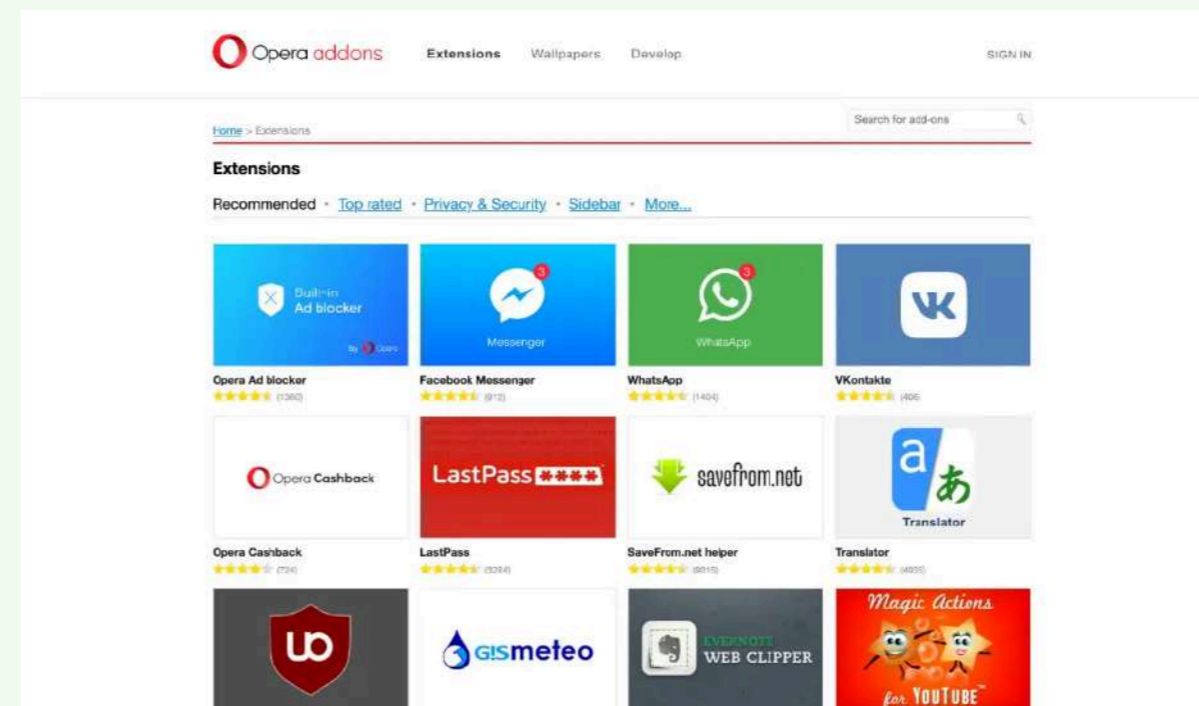
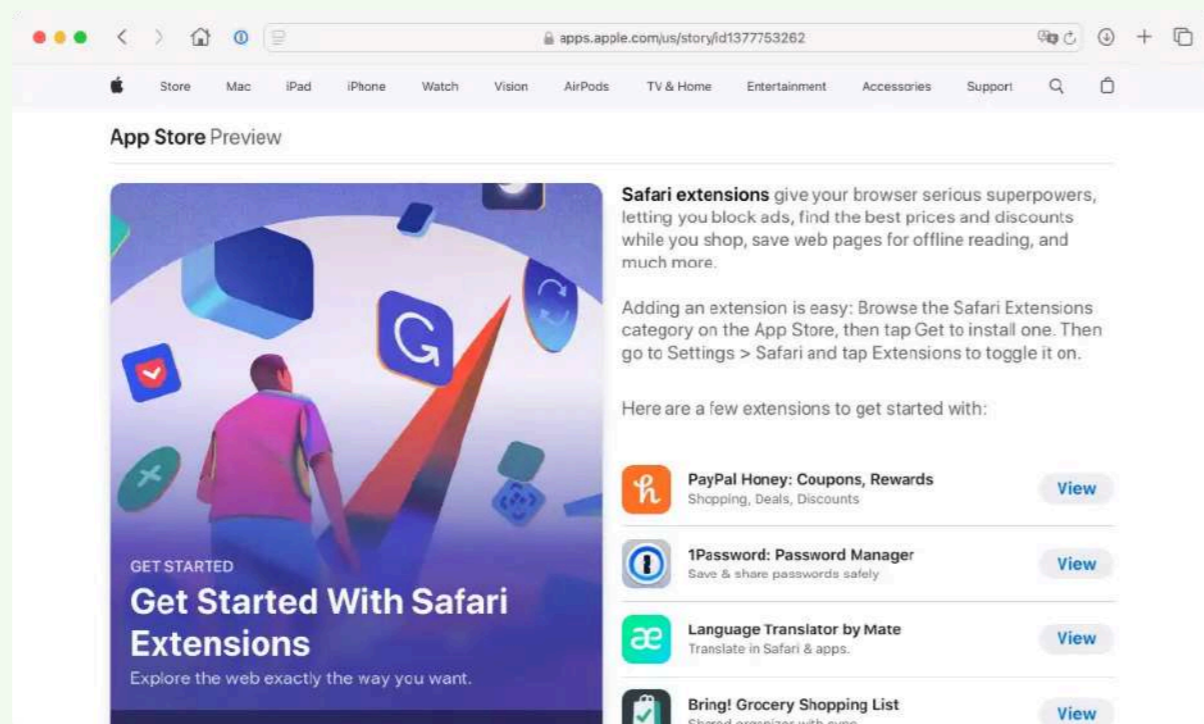
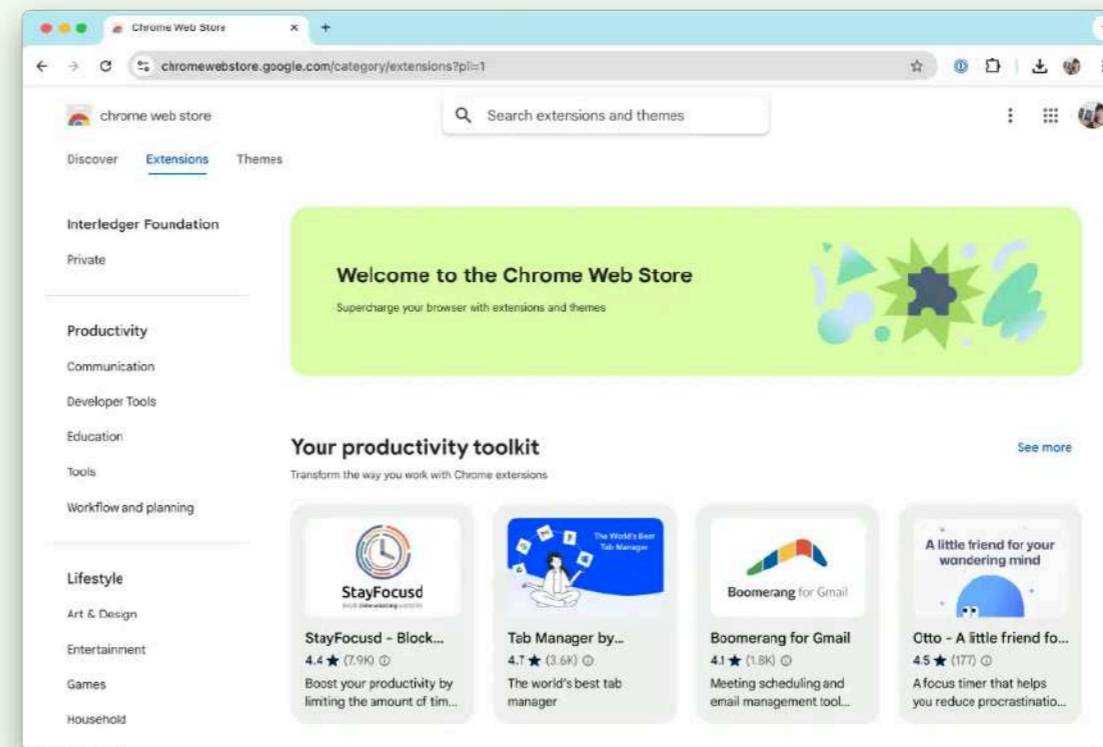
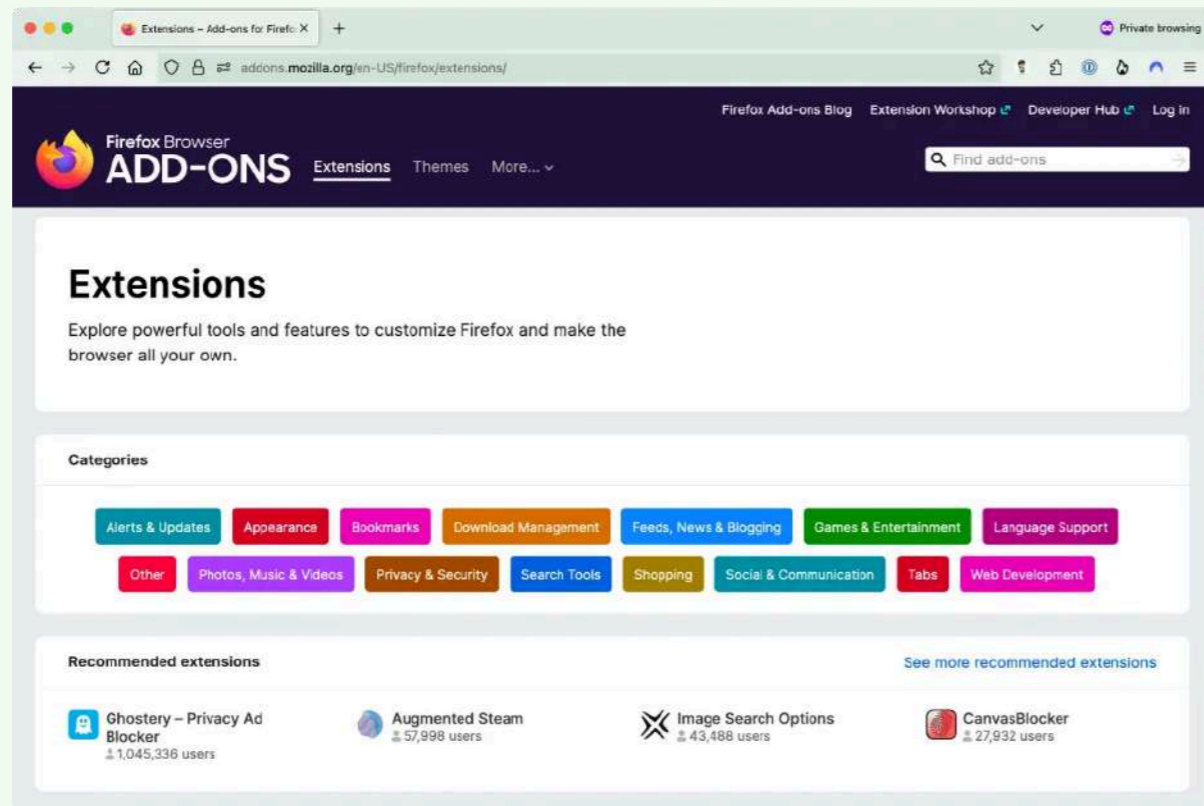
Microsoft and the freedom to innovate.



[What's your opinion?](#)

Last Updated: Monday, November 30, 1998
 ©1998 Microsoft Corporation. All rights reserved. [Terms of Use](#) [Privacy Policy](#)

Done Internet zone

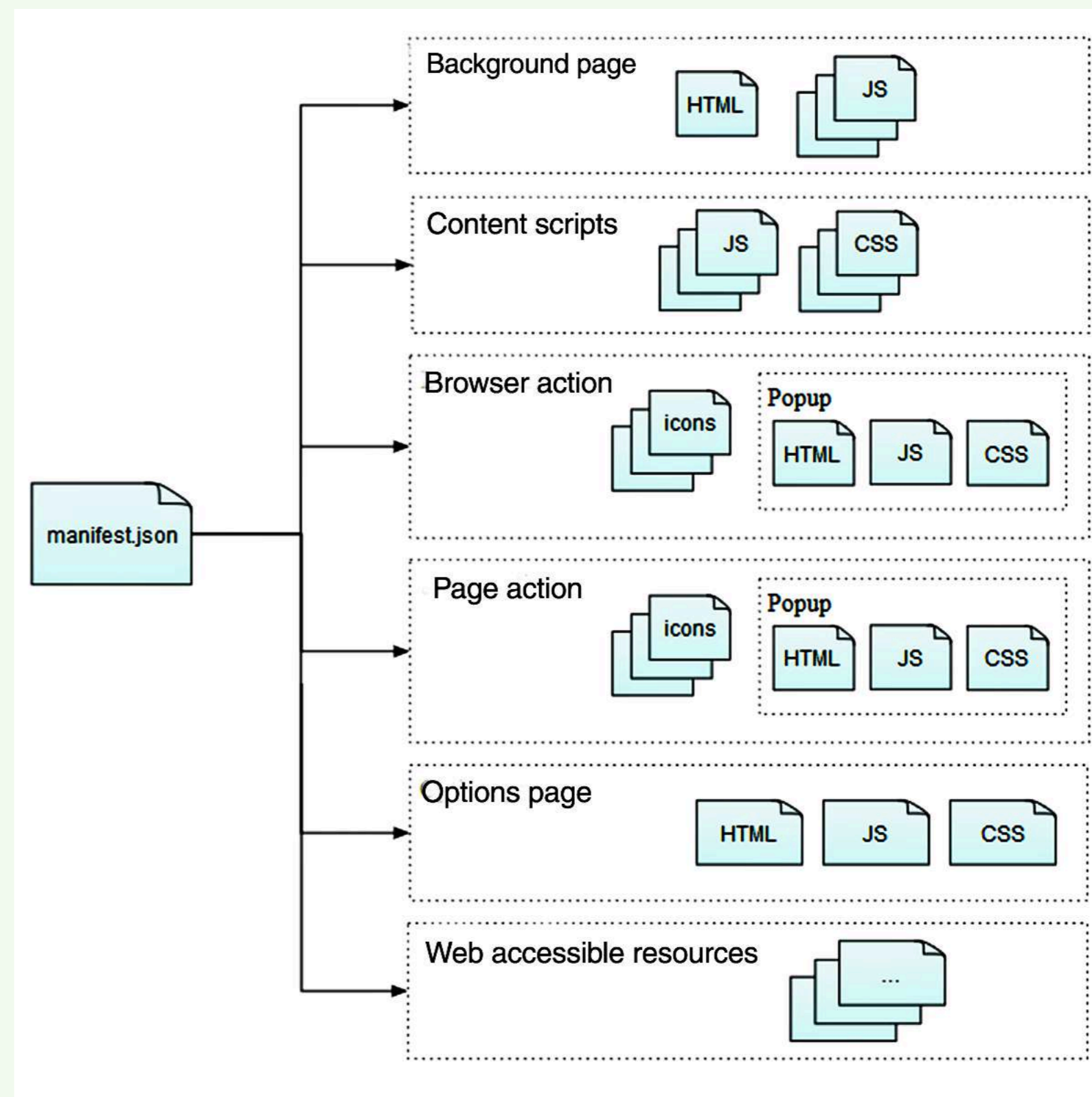


manifest.json

```
{  
  "manifest_version": 3,  
  "name": "Awesome Extension",  
  "version": "1.0.0",  
}
```

"manifest_version", "version", and "name" are the only mandatory keys.

See [manifest.json](#) page on MDN for full list of keys



AE1: Does nothing

Folder structure:

```
├── AE1/  
│   ├── manifest.json  
│   ├── nothing.html  
│   └── icons/  
│       ├── icon32.png  
│       └── icon48.png
```

nothing.html:

```
<html>  
  <body>  
    <h1>Nothing</h1>  
  </body>  
</html>
```

Icons:



```
{  
  "manifest_version": 3,  
  "name": "AE1",  
  "version": "1.0",  
  
  "description": "This extension doesn't actually do anything",  
  "icons": {  
    "32": "icons/icon32.png",  
    "48": "icons/icon48.png"  
  },  
  
  "action": {  
    "default_popup": "nothing.html"  
  }  
}
```

`about:debugging#/runtime/this-firefox`

Open the `about:debugging` page, click the This Firefox option, click the Load Temporary Add-on button, then select any file in your extension's directory.

The extension now installs, and remains installed until you restart Firefox.

`chrome://extensions`

Enable Developer Mode by clicking the toggle switch next to Developer mode.

Click the Load unpacked button and select the extension directory.

AE1.1: Does a tiny something

nothing.html:

```
<html>
  <body>
    <h1>Nothing</h1>
    <button>Something</button>
  </body>
  <style>
    body { text-align: center }
  </style>
  <script src="nothing.js"></script>
</html>
```

nothing.js:

```
document.querySelector("button").addEventListener("click", () => {
  document.querySelector("h1").style.color = "tomato";
});
```

Content scripts

1. Static declaration

Register the script using the `content_scripts` key in your `manifest.json`. This automatically loads the script on pages that match the specified pattern.

2. Dynamic declaration

The script is registered with the browser via the `scripting.registerContentScripts()` method. The difference with the static declaration method is you can add or remove content scripts at runtime.

3. Programmatic injection

Load the script via the `scripting.executeScript()` method into a specific tab based on particular triggers, e.g. an user action.

Content Script Environment

Firefox

Xray vision in Firefox

- The global scope (`globalThis`) is composed of standard JavaScript features as usual, plus `window` as the prototype of the global scope.
- Most DOM APIs are inherited from the page through `window`, through Xray vision to shield the content script from modifications by the web page.
- A content script may encounter JavaScript objects from its global scope or Xray-wrapped versions from the web page.

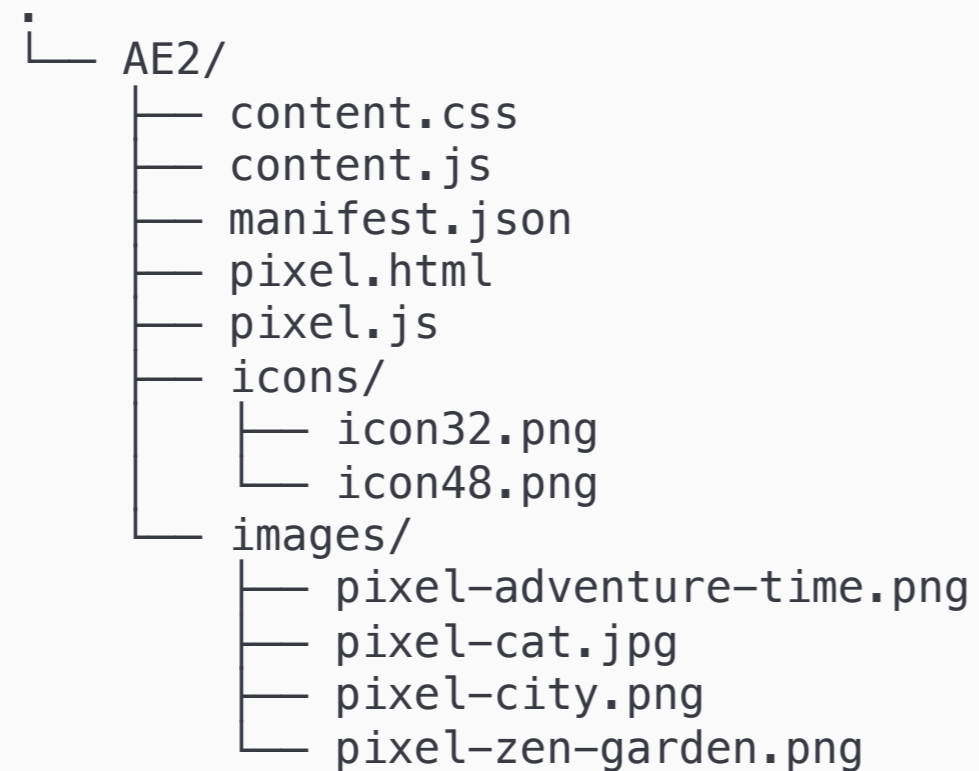
Chrome

Isolated worlds in Chrome

- An isolated world is a private execution environment that isn't accessible to the page or other extensions.
- The global scope is `window`, and the available DOM APIs are generally independent of the web page (other than sharing the underlying DOM).
- Content scripts cannot directly access JavaScript objects from the web page.

AE2: Click button, change page

Folder structure:



- Image source: [pixel-adventure-time.png](#)
- Image source: [pixel-cat.jpg](#)
- Image source: [pixel-city.png](#)
- Image source: [pixel-zen-garden.png](#)

```
{
  "manifest_version": 3,
  "name": "AE2",
  "version": "1.0",
  "description": "Activate pixel art",
  "icons": {
    "32": "icons/icon32.png",
    "48": "icons/icon48.png"
  },
  "permissions": ["activeTab", "scripting"],
  "action": {
    "default_popup": "pixel.html"
  },
  "web_accessible_resources": [
    {
      "resources": [
        "images/pixel-adventure-time.png",
        "images/pixel-cat.jpg",
        "images/pixel-city.png",
        "images/pixel-zen-garden.png"
      ],
      "extension_ids": ["*"],
      "matches": ["*://*/*"]
    }
  ]
}
```

AE2: Click button, change page

pixel.html:

```
<html>
  <head>
    <meta charset="UTF-8">
    <style>
      body { text-align: center }
      h1 { white-space: nowrap }
      button:first-of-type { margin-block-end: 0.5em }
    </style>
  </head>
  <body>
    <h1>Pixel-time</h1>
    <button id="pixelate">Pixelate</button>
    <button id="reset">Reset</button>
  </body>
  <script src="pixel.js"></script>
</html>
```

pixel.js:

```
window.browser = (function () {
  return window.msBrowser || window.browser || window.chrome;
})();

let id;
browser.tabs.query({ active: true, currentWindow: true }, (tabs) => {
  id = tabs[0].id;
  browser.scripting.executeScript({
    target: { tabId: tabs[0].id },
    files: ["content.js"],
  });
  browser.scripting.insertCSS({
    target: { tabId: tabs[0].id },
    files: ["content.css"],
  });
});

document.getElementById("pixelate").addEventListener("click", () => {
  browser.tabs.sendMessage(id, { message: "pixelate" });
});

document.getElementById("reset").addEventListener("click", () => {
  browser.tabs.sendMessage(id, { message: "reset" });
});
```

AE2: Click button, change page

content.js:

```
window.browser = (function () {
  return window.msBrowser || window.browser || window.chrome;
})();

function pickPixelArt(art) {
  switch (art) {
    case "a":
      return browser.runtime.getURL("images/pixel-adventure-time.png");
    case "b":
      return browser.runtime.getURL("images/pixel-cat.jpg");
    case "c":
      return browser.runtime.getURL("images/pixel-city.png");
    case "d":
      return browser.runtime.getURL("images/pixel-zen-garden.png");
  }
}

function pickRandomImage() {
  const array = ["a", "b", "c", "d"];
  let index = Math.floor(Math.random() * array.length);
  let random = array[index];
  return random;
}
```

content.css:

```
body:has(.pixel-time) {
  overflow: hidden;
}

.pixel-time {
  position: fixed;
  z-index: 9999;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: black;
  display: flex;
}

.pixel-time img {
  margin: auto;
  width: 100%;
}
```

Background scripts

Background scripts or a background page enable you to monitor and react to events in the browser, such as navigating to a new page, removing a bookmark, or closing a tab.

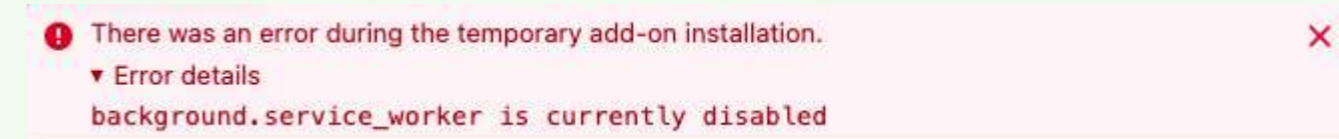
Source: [MDN: Background scripts](#)

AE3: Press key, change page

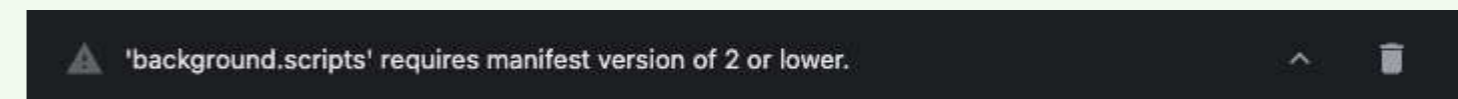
manifest.json:

```
{
  "background": {
    "service_worker": "background.js",
    "scripts": ["background.js"]
  },
  "commands": {
    "_execute_action": {
      "suggested_key": {
        "default": "Ctrl+Shift+Y"
      }
    },
    "pixelate": {
      "suggested_key": {
        "default": "Alt+A"
      },
      "description": "Send a 'pixelate' event to the extension"
    },
    "reset": {
      "suggested_key": {
        "default": "Ctrl+Shift+E"
      },
      "description": "Send a 'reset' event to the extension"
    }
  }
}
```

Firefox (when only `service_worker` is used):



Chrome (when `scripts` exist in the manifest):



Proposal: declaring background scripts in a neutral way

AE3: Press key, change page

background.js:

```
const API = chrome || browser;

API.tabs.onActivated.addListener((activeInfo) => {
  API.tabs.get(activeInfo.tabId, function (tab) {
    API.commands.onCommand.addListener((command) => {
      if (command === "pixelate") {
        API.tabs.sendMessage(tab.id, { message: "pixelate" });
      } else if (command === "reset") {
        API.tabs.sendMessage(tab.id, { message: "reset" });
      }
    });
  });
});
```

w3c / webextensions

Code Issues 351 Pull requests 11 Actions Projects 1 Wiki Security Insights

Use cases that are not well served by service workers #72

Open dotproto opened this issue on Sep 3, 2021 · 113 comments

dotproto commented on Sep 3, 2021 · edited

Near the end of the [2021-08-19](#) meeting we briefly discussed (but didn't capture) the idea of collecting background page use cases and design patterns that are not well served by service workers. My hope is that by collecting these use cases in a well known location, community members and browser vendors we will be able to more concretely discuss the challenges posed by this new background context and explore potential solutions.

Please use this issue to link to use use cases that are impossible to accomplish with or not well supported by service workers. If you'd like to report a new use case, please [create a new issue](#) and reference this one.

Use cases

Specific ways that background pages are used that cannot be accomplished with a service worker.

- HTML/XML parsing
 - [Issue 1056364: Add support for parsing XML and HTML documents to Manifest V3](#)
 - [Issue 1051597: Extension Manifest V3 support for SingleFileZ and similar extensions](#)
- Background page interaction
 - [Issue 1128240: Background page loading & manipulation in service worker browser extensions](#)
- Audio playback
 - [Issue 1131236: Playing audio in service worker-based browser extensions](#)
- Clipboard access
 - [Issue 1160302: Cannot read clipboard from service worker in a MV3 chrome extension](#)
- Blob URLs
 - [Issue 1051597, comment #5+](#)
- Requesting file:-scheme resources (aka file:// URLs)
 - [Issue 1051597, comment #12+](#) – Discussion of file resource request limitations in Fetch
- WebRTC streams
 - Extension "Snowflake" [Create RTCPeerConnections in workers](#) [webrtc-extensions#77](#) (comment)
 - [Issue 1207214: RTCDataChannel for ManifestV3](#)
- WebSockets
 - chromium-extensions: [MV3 Service Workers Don't Support Websocket](#)
 - [Create RTCPeerConnections in workers](#) [webrtc-extensions#77](#)
- Native messaging
 - [Issue 1152255: ServiceWorker is shut down every 5 minutes for manifest V3 extension](#)
- Wasm
 - [Issue 1173354: wasm does not work in extensions manifest v3](#)

Assignees: No one assigned

Labels: [topic: service worker](#)

Projects: [WECG Issue Management](#) Status: [Todo](#)

Milestone: No milestone

Development: No branches or pull requests

Notifications [Customize](#)

[Subscribe](#)

You're not receiving notifications from this thread.

49 participants

and others

<https://github.com/w3c/webextensions/issues/72>

Manifest v2 versus v3

v2

- Extension authors have a choice on whether background scripts or a page can be persistent or non-persistent
- Uses the `chrome.webRequest` API, a flexible API that lets extensions intercept and block or otherwise modify HTTP requests and responses
- Code can be hosted remotely
- Most API methods use callback functions

v3

- Background scripts are run with non-persistent service workers
- Uses the `declarativeNetRequest` API, declarative API to specify conditions and actions that describe how network requests should be handled
- Support is removed for remotely hosted code and execution of arbitrary strings
- Most API methods return promises

Migrate v2 to v3 🙄

Problem:

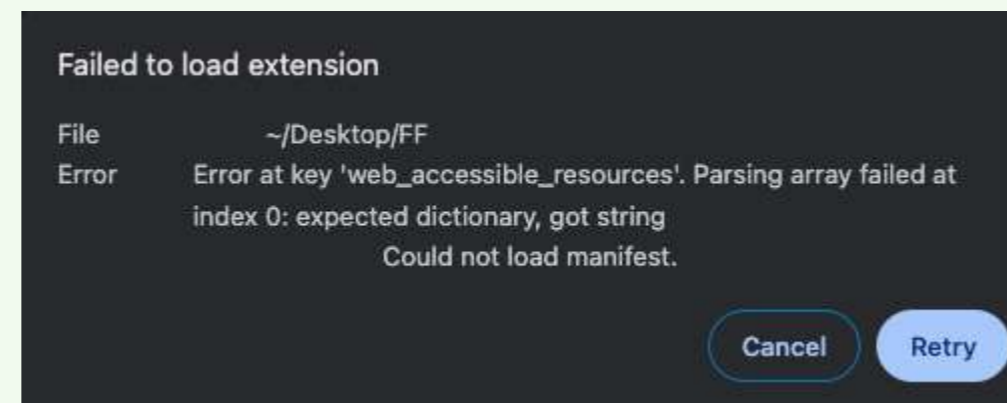
```
Manifest version 2 is deprecated, and support will be removed in 2024.  
See https://developer.chrome.com/docs/extensions/develop/migrate/mv2-deprecation-timeline for details.
```

Fix:

```
"manifest_version": 3
```

Migrate v2 to v3 🙄

Problem:



Fix:

```
"action": { ... },  
  
"web_accessible_resources": [  
  {  
    "resources": ["beasts/*.jpg"],  
    "extension_ids": ["*"],  
    "matches": ["*://*/*"]  
  }  
]
```

Migrate v2 to v3 🙄

Problem:

```
Uncaught ReferenceError: browser is not defined  
at choose_beast.js:100:1
```

Fix:

```
window.browser = (function () {  
  return window.msBrowser || window.browser || window.chrome;  
})();
```

Migrate v2 to v3 🙄

Problem:

```
Uncaught TypeError: Cannot read properties of undefined (reading 'then')  
at choose_beast.js:105:3
```

Fix:

```
/* Replace tabs.executeScript */  
browser.tabs  
  .executeScript({ file: "/content_scripts/beastify.js" })  
  .then(listenForClicks)  
  .catch(reportExecuteScriptError);
```

```
/* With scripting.executeScript */  
browser.tabs.query({ active: true, currentWindow: true }, (tabs) => {  
  browser.scripting  
    .executeScript({  
      target: { tabId: tabs[0].id },  
      files: ["/content_scripts/beastify.js"],  
    })  
    .then(listenForClicks)  
    .catch(reportExecuteScriptError);  
});
```

Migrate v2 to v3 🙄

Problem:

```
Error handling response: TypeError: Cannot read properties of undefined (reading 'executeScript')  
at chrome-extension://lejlkohkjhglbclhbnbpfjmljmkml/popup/choose_beast.js:105:6
```

Fix:

```
"permissions": ["activeTab", "scripting"]
```

Migrate v2 to v3 🙄

Problem:

```
choose_beast.js:63 Could not beastify: TypeError: browser.tabs.insertCSS is not a function
```

Fix:

```
/* Replace tabs.insertCSS */
function beastify(tabs) {
  browser.tabs.insertCSS({ code: hidePage }).then(() => {
    const url = beastNameToURL(e.target.textContent);
    browser.tabs.sendMessage(tabs[0].id, {
      command: "beastify",
      beastURL: url,
    });
  });
}
```

```
/* With scripting.insertCSS */
function beastify(tabs) {
  browser.scripting.insertCSS({
    target: { tabId: tabs[0].id },
    css: `body > :not(.beastify-image) { display: none; }`,
  })
  .then(() => {
    const url = beastNameToURL(e.target.textContent);
    browser.tabs.sendMessage(tabs[0].id, {
      command: "beastify",
      beastURL: url,
    });
  });
}
```

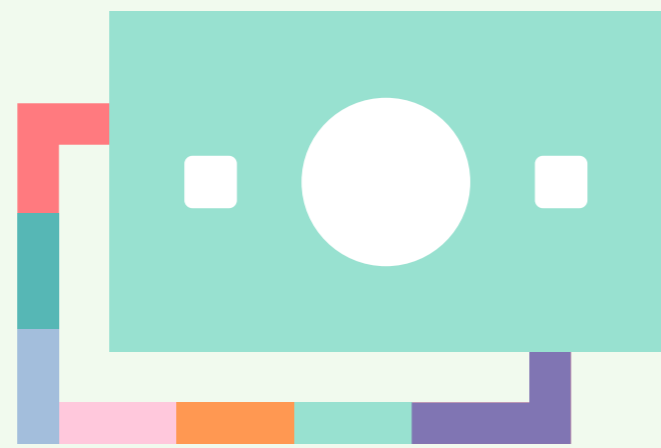
Migrate v2 to v3 🙄

Problem:

```
Uncaught (in promise) Error: Could not establish connection. Receiving end does not exist.
```

Fix:

```
/* Add to content script */  
window.browser = (function () {  
  return window.msBrowser || window.browser || window.chrome;  
})();
```

Web monetization

<https://webmonetization.org/>

<https://issues.chromium.org/issues/40110471>

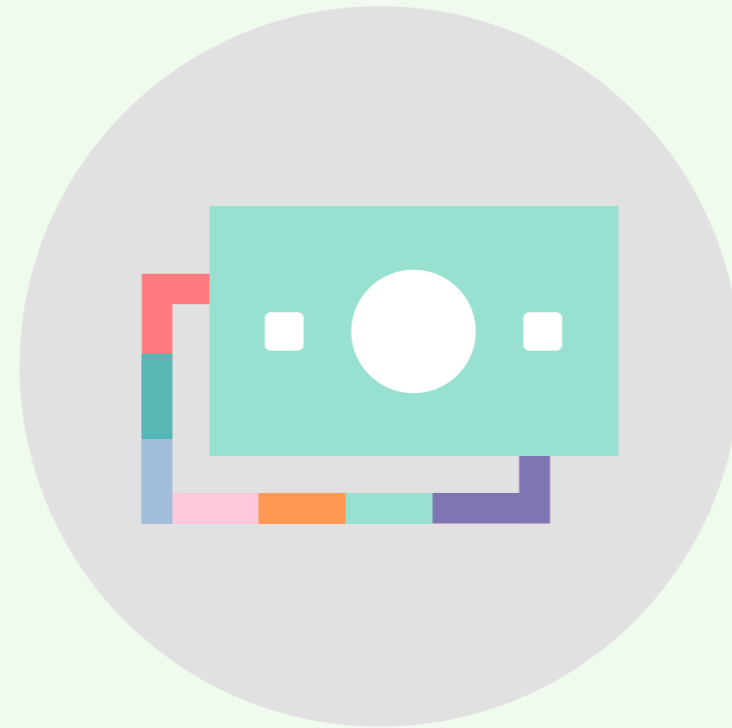
Payment Pointers

- A standardized identifier for payment accounts
- Used by an account holder to share the details of their account with a counter-party

```
https://ilp.gatehub.net/747467740/USD
```

To implement web monetization on a website:

```
<link rel="monetization" href="https://ilp.gatehub.net/747467740/USD">
```



<https://github.com/interledger/web-monetization-extension>

<https://github.com/interledger/web-monetization-extension/blob/eff212733de71444ff033f131ee3e3c4f000af29/src/background/services/background.ts#L70-L73>

```
async injectPolyfill() {
  try {
    await this.browser.scripting.registerContentScripts([
      {
        world: 'MAIN',
        id: 'polyfill',
        allFrames: true,
        js: ['polyfill/polyfill.js'],
        matches: PERMISSION_HOSTS.origins,
        runAt: 'document_start',
      },
    ]);
  } catch (error) {
    // Firefox <128 will throw saying world: MAIN isn't supported. So, we'll
    // inject via contentScript later. Injection via contentScript is slow,
    // but apart from WM detection on page-load, everything else works fine.
    if (!error.message.includes(`world`)) {
      this.logger.error(
        `Content script execution world \`${MAIN}\` not supported by your browser.\n` +
        `Check https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/API/scripting/ExecutionWorld#browser_comp
        error,
      );
    }
  }
}
```

<https://github.com/interledger/web-monetization-extension/blob/eff212733de71444ff033f131ee3e3c4f000af29/src/content/services/contentScript.ts#L70-L74>

```
// Todo: When Firefox has good support for `world: MAIN`, inject this directly  
// via manifest.json https://bugzilla.mozilla.org/show_bug.cgi?id=1736575 and  
// remove this, along with injectPolyfill from background  
// See: https://github.com/interledger/web-monetization-extension/issues/607  
  
async injectPolyfill() {  
  const document = this.window.document;  
  const script = document.createElement('script');  
  script.src = this.browser.runtime.getURL('polyfill/polyfill.js');  
  await new Promise<void>((resolve) => {  
    script.addEventListener('load', () => resolve(), { once: true });  
    document.documentElement.appendChild(script);  
  });  
  script.remove();  
}
```

<https://github.com/interledger/web-monetization-extension/blob/eff212733de71444ff033f131ee3e3c4f000af29/src/content/polyfill.ts#L103>

```
window.addEventListener(
  '__wm_ext_monetization',
  (event: CustomEvent<MonetizationEventPayload['details']>) => {
    if (!(event.target instanceof HTMLLinkElement)) return;
    if (!event.target.isConnected) return;

    const monetizationTag = event.target;
    monetizationTag.dispatchEvent(
      new MonetizationEvent('monetization', event.detail),
    );
  },
  { capture: true },
);

window.addEventListener(
  '__wm_ext_onmonetization_attr_change',
  (event: CustomEvent<{ attribute?: string }>) => {
    if (!event.target) return;

    const { attribute } = event.detail;
    // @ts-expect-error: we're defining this now
    event.target.onmonetization = attribute
      ? new Function(attribute).bind(event.target)
      : null;
  },
  { capture: true },
);
```



<https://webmonetization.org/>

References

- [A Brief History of Browser Extensibility](#)
- [Why Did Mozilla Remove XUL Add-ons?](#)
- [Safari web extensions](#)
- [Chromium removed support for user stylesheets](#)
- [How to Add a User Stylesheet in Firefox](#)
- [What are Bookmarklets? How to Use JavaScript to Make a Bookmarklet in Chromium and Firefox](#)
- [Creating a Safari web extension](#)
- [Use cases that are not well served by service workers](#)
- [Google's Manifest V3 Still Hurts Privacy, Security, and Innovation](#)

Thank you



<https://chenhuijing.com>



@huijing



@huijing@tech.lgbt



@huijing.bsky.social

Font is Figtree by Erik Kennedy.