coolriver2685
1年前 浏览
2 评论
1 收藏

chrome插件 DIY

浏览器

> 本文作者：imweb coolriver 原文出处：imweb社区 **未经同意，禁止转载**

1 前言

对于一个web前端开发者，chrome浏览器是一个工作，学习和生活的必备工具。除了chrome本身的基本能力（控制台等）外，能大幅提高这个神器的使用体验的是，可扩展能力（插件）以及丰富的插件生态。

每个人根据使用习惯会有自己的一套插件配置（鼠标手势、代理配置等等），这些

作者其它文章

在新窗口中打开页面？... (1104浏览)

React入门心得及使用t... (1005浏览)

ES6学习笔记 (5272浏览)

setTimeout的那些事 (2969浏览)

页面守护者：Service ... (5078浏览)

推荐文章

基于vue2.0+vuex+loc... (362浏览)

基于vue2+vuex2+vu... (507浏览)

编写graphql服务 (309浏览)

pm2模块编写入门 (499浏览)

设计模式初探-代理模式 (524浏览)

插件包括具体的插件的配置信息，甚至可以和你的google账号绑定，当你换一台电脑，只要使用相同的google账号登录chrome，就会享受到一致的使用体验。

当你打开chrome的“扩展程序”界面，看着琳琅满目的插件，有没有想过亲自动手，打造一个自己的插件呢？当然，这种想法不应该是闲着某个部位疼，刻意的去开发一个连自己都不会实际使用的插件。而应该是发现现在的插件库里，没有一个能解决自已在使用chrome过程中某个痛点的插件。

好，假设现在你在chrome的使用上想要一个扩展功能，但用各种关键字在各种可能找到答案的地方都搜索了，仍然没有看到想要的插件。这个时候，就可以考虑自己开发了。如果刚好你是一个web前端开发者，那么恭喜你，几乎没有门槛（只要有能看懂chrome extentions API文档就行），因为所有用到的技术都是你所熟悉的： json配置, js逻辑， css+html展示。

2 初步探索

chrome extension的官方文档上，有着简单的demo指引，全面的API文档，以及对于各种API的丰富例子。对于初次开发chrome插件的人来说，建议先看入门指引小demo，篇幅很小，很快可以看完。看完之后对chrome插件的基本配置和文件结构会有一个大致的认识，同时也学会了如何在chrome上加载自己在本地开发的插件。

chrome extention支持的扩展点以及扩展功能很多，对于初学者不可能一下子看完官方所有文档再去动手。而应该是先根据自己用过的插件，脑海中有个大致的印象：插件可以在哪些地方起到效果。chrome插件最常见的功能莫过到url栏右侧的那些小图标了，就是这些：

如果看完上入门指引小demo后，你肯定知道了怎么样实现这样一个功能。这里就不讲官方文档里已经讲过的细节，主要讲一下比较重要的配置文件 `manifest.json` 。官方小demo的配置文件内容如下：

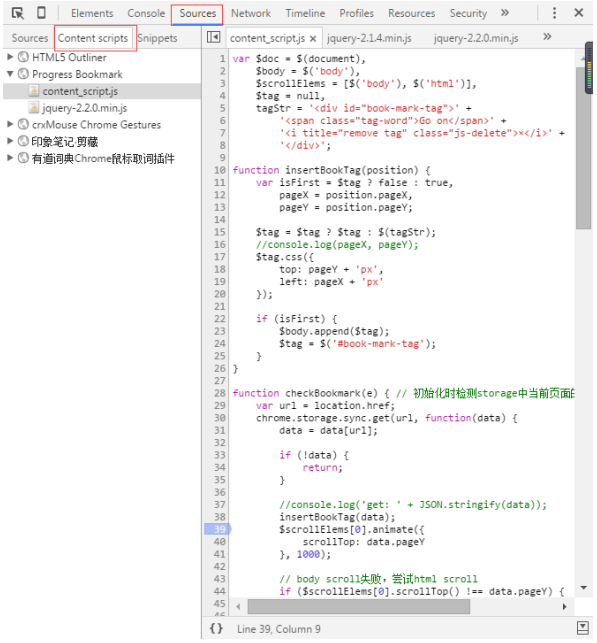
```
{
  "manifest_version": 2, // 插件标准版本

  "name": "Getting started example", //
  "description": "This extension shows
  "version": "1.0", // 插件版本,发布到插

  "browser_action": { // 就是url栏右侧那
    "default_icon": "icon.png", // 展示
    "default_popup": "popup.html" // 点
  },
  "permissions": [ // 插件权限设置，规则
    "activeTab",
    "https://ajax.googleapis.com/"
  ]
}
```

插件配置文件里配置的文件路径，都是相对于 `mainfest.json` 的相对路径。所以当插件逻辑并不复杂时，通常就将配置文件和插件代码直接放在同一层目录下。插件配置文件中，配置了该插件扩展的描述信息、扩展的功能，以及插件的访问权限。插件功能除了 `browser_action` 配置的popup页面外，还支持什么功能呢？用过鼠标手势类插件的肯定知道，插件可以**访问和修改页面dom**，因为这类插件本质就是在dom上绘制鼠标路径。那么在哪里配置控制页面dom的代码呢？
在 `content_scripts` 中：

```
{
  "name": "My extension",
  ...
```

```
  "content_scripts": [ // 这里配置的脚本
    {
      "matches": ["http://www.google.cc
      "css": ["mystyles.css"],
      "js": ["jquery.js", "myscript.js'
    }
  ],
  ...
}
```

`content_scripts` 实际上是运行在受插件影响的页面中，在devtool中可以看到插件注入的这些内容:



目前我们已经掌握了插件两个功能点的配置了。如果用过evernote剪裁插件的人应该知道，插件可以改变chrome在页面上的右键菜单：

这里在哪里实现的呢？以上的两个配置点

无法实现，是在 `background` 项中配置的：

```
{
  "name": "My extension",
  ...
  "background": { // 在浏览器运行环境中运
    "scripts": ["background.js"]
  }
```

了解了上面三种扩展点的配置方法后，就可以开始开发功能丰富的插件了。

3 插件开发示例

本人开发了一个插件，通过介绍这个插件的开发流程，让大家熟悉几个常用的API。

> **初衷/痛点**：平时看一些文章/博客，要处理手头上其他事情，不得不中断。为了下次再看，通常是保存到书签/笔记。使用书签/笔记的方法，有两个弊端：一个是下次再进入时，不记得上次看到哪里了，又得重头开始看，或者拼命回忆上次看到哪里；另一个是，下次根本不记得哪些是没有看完的（除非专门建一个分类标签）。
> **期望**：有个插件，能够记录那些没看完，但又非常想继续看完的文章，即使关闭浏览器，换个浏览器，也能够获取到这些记录，并且打开再次打开文章时，能自动跳转到上次看到的位置。

- 安装：https://chrome.google.com/webstore/detail/bookmark/ediaiaoabgoimfjpmegbhlhmpajmegoj

这个插件的主要功能是：

- 记录没有看完的文章/博客（在页面上通过右键菜单添加标记）,保存进度（按高度百分

比）。

- 下次从记录（url栏右侧插件功能点）中进入文章页面时，页面会滚动到上次标记的位置。

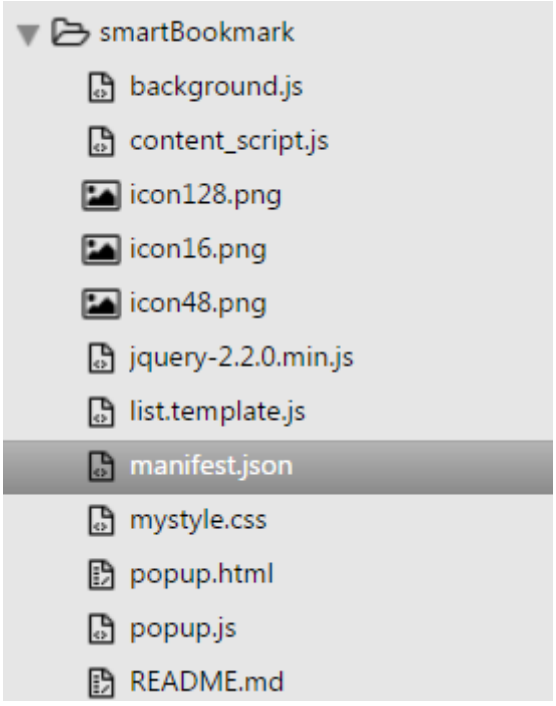
3.1 配置文件

这个插件用到了2节中讲到了3个插件扩展点： `browser_action` ， `content_scripts` 和 `background` 。配置文件如下：

```
{
    "manifest_version": 2,

    "name": "Progress Bookmark",
    "description": "save progress of ar
    "version": "1.1.5",

    "browser_action": {
        "default_icon": "icon48.png",
        "default_popup": "popup.html",
        "default_title": "Progress Book
        "default_badgetext": "test"
    },
    "icons": {
        "16": "icon16.png",
        "48": "icon48.png",
        "128": "icon128.png"
    },
    "background": {
        "scripts": ["background.js"]
    },
    "content_scripts": [{
        "matches": [
            "http://*/*",
            "https://*/*"
        ],
        "js": [
            "jquery-2.2.0.min.js",
            "content_script.js"
        ],
        "css": ["mystyle.css"]
    }],
    "permissions": [
        "contextMenus",
```

```
        "storage",
        "tabs",
        "http://*/*",
        "https://*/*"
    ]
}
```

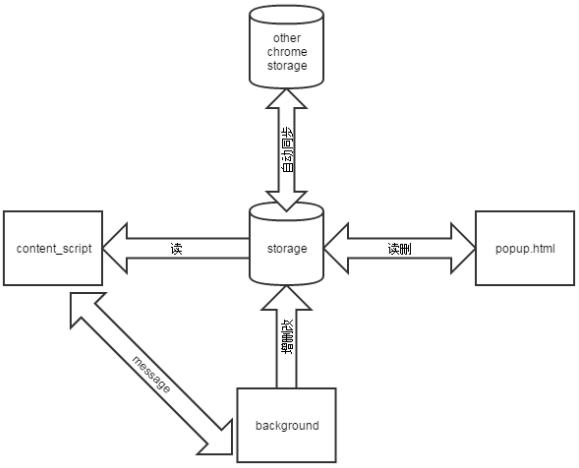
上面的配置文件已经包含了大部分需要开发的文件名： `popup.html`， `background.js`， `content_script.js` `mystyle.css`。另外，各种尺寸的icon图片，用于插件在不同地方（插件logo，pop图标，插件商店等等）展示。目录结构如下：



3.2 数据存储和数据流

本插件的功能类似于书签，需要保存目标页面的一些信息（标题, url, 进度）。那么有没有一种好的方法，可以保存这些数据，并且在同一个google账号上共享呢？还真有： `chrome.storage`。官方文档中详细介绍了其用法，以及如何在同账号不同浏览器上自动同步数据。

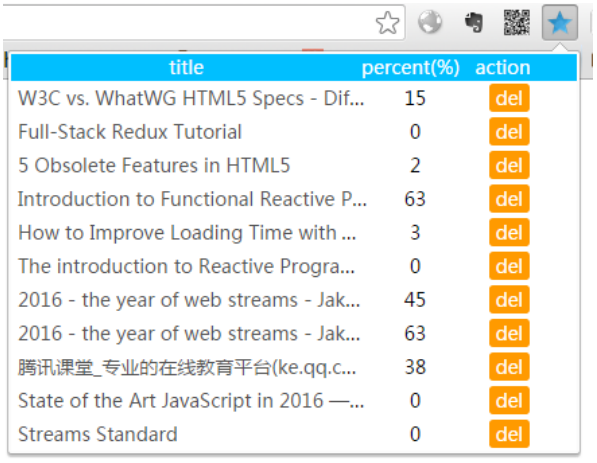
基于 `chrome.storage` ，本插件的各种扩展点的数据流操作图如下：



上图中，content_script和background的脚本运行环境不一样，通过message来进行通讯。

3.3 popup页面

本插件的popup页面用于展示已经保存记录的未读完页面，页面展示效果如下:



title	percent(%)	action
W3C vs. WhatWG HTML5 Specs - Dif...	15	del
Full-Stack Redux Tutorial	0	del
5 Obsolete Features in HTML5	2	del
Introduction to Functional Reactive P...	63	del
How to Improve Loading Time with ...	3	del
The introduction to Reactive Progra...	0	del
2016 - the year of web streams - Jak...	45	del
2016 - the year of web streams - Jak...	63	del
腾讯课堂_专业的在线教育平台(ke.qq.c...	38	del
State of the Art JavaScript in 2016 —...	0	del
Streams Standard	0	del

该页面负责展示和删除保存的记录，页面主要逻辑如下:

```
// popup.js in popup.html
document.addEventListener('DOMContentLc
    loadStorage();
    bindEvents();
});

function loadStorage() { // 读取storage
    chrome.storage.sync.get(null, funct
        var list = [],
            prop,
            html;
        //console.log(data);
        for (prop in data) {
```

```
            if (data.hasOwnProperty(prc
                if (data[prop].pageX !=
                    list.push($.extend(
                        url: prop
                    }));
                }
            }
        }

        html = renderList(list);
        $('#content').html(html);
    });
}

function bindEvents() {
    $(document).on('click', '.js-remove
        //console.log('click remove');
        var $this = $(this),
            $link = $this.closest('tr')
            url = $link.attr('href');
        chrome.storage.sync.remove(url,
            loadStorage();
        });
    });
}
```

3.4 content_script

content_script主要有两个功能：

- 在页面中操作（新增 、滚动到指定位置、删除）记录坐标的元素 ；
- 向background发送坐标消息和删除坐标的消息。 主要功能代码如下：

```
// content_script.js

function checkBookmark(e) { // 初始化时
    var url = location.href;
    chrome.storage.sync.get(url, functi
        data = data[url];

        if (!data) {
            return;
        }
```

```
        //console.log('get: ' + JSON.st
        insertBookTag(data);
        $scrollElems[0].animate({
            scrollTop: data.pageY
        }, 1000);

        // body scroll失败，尝试html scr
        if ($scrollElems[0].scrollTop()
            $scrollElems[1].animate({
                scrollTop: data.pageY
            }, 1000);
        }
    });
}

function bindEvents() { // 事件和消息
    $doc.on('mouseup', function(e) { //
            //console.log(e.which);

            if (e.which === 3) {
                chrome.runtime.sendMess
                    type: 'bookmark-pos
                    pageX: e.pageX,
                    pageY: e.pageY,
                    title: document.tit
                    progress: Math.floo
                });
            }
        })
        .on('ready', checkBookmark)
        .on('click', '#book-mark-tag .j
            chrome.runtime.sendMessage(
                type: 'remove-bookmark'
            });
        });

    chrome.runtime.onMessage.addListene
        if (request.type === 'add-bookm
            insertBookTag(request);
        } else if (request.type === 're
            deleteTag();
        }
    });
}
```

3.5 background

background负责的任务有：

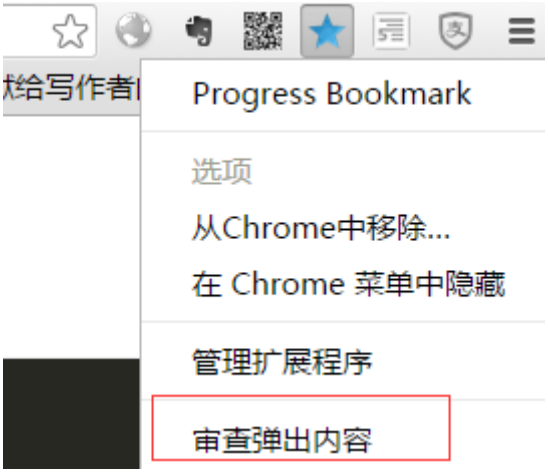
- 添加右键菜单
- 接收content_script发来的消息并处理

```
// background.js
function createMenu() {      // 添加右键
    var contexts = ["page", "selection"
        "audio"
    ];
    contextId = chrome.contextMenus.cre
        "title": 'Set progress tag here
        "contexts": contexts,
        "onclick": menuHandle
    });
}

function menuHandle() { // 右键菜单点击时
    //console.log('click');
    var value = {},
        key = senderCur.url,
        obj = {};

    value.pageX = requestCur.pageX;
    value.pageY = requestCur.pageY;
    value.progress = requestCur.progres
    value.title = requestCur.title;

    obj[key] = value;
    //console.log(obj);
    chrome.storage.sync.set(obj, functi
        //console.log('send callback');
        chrome.tabs.query({ // 查找当前
            active: true,
            currentWindow: true
        }, function(tabs) { // 发送消息
            chrome.tabs.sendMessage(tab
                type: 'add-bookmark-cb'
                pageX: requestCur.pageX
                pageY: requestCur.pageY
            });
        });
    });
}
```

```
function bindEvents() {
    // 响应来自content_script的message
    chrome.runtime.onMessage.addListene
        if (request.type === 'bookmark-
            requestCur = request;
            senderCur = sender;
        } else if (request.type === 're
            //console.log('sendRequest:
            chrome.storage.sync.remove(
                chrome.tabs.query({ //
                    active: true,
                    currentWindow: true
                }, function(tabs) { //
                    chrome.tabs.sendMes
                        type: 'remove-b
                    });
                });
            });
        }
    });
}
```

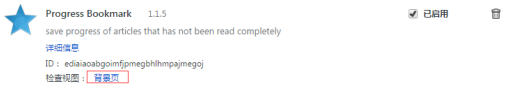
4 插件调试方法

- popup.html调试： 右键点击popup图标，选择“审查弹出内容”：



- content_script调试：devtool -> sources -> Content scripts
- background调试：chrome://extensions/ 激活开发者模式，点击

对应插件“检查视图”后的“背景页”



5 插件发布

需要google账户+信用卡（为了成为google认证的开发者）
参考链接：https://segmentfault.com/a/119000000

2 条评论

张颖 1年前

学习了

coverguo 9个月前

很不错的教程哈