# 十一、消息传递

## 一、一次性请求

1、向扩展程序的各个部分发消息

```
chrome.runtime.sendMessage(
    extensionId?: string, 可任何扩展程序 (自身或其他)
    message: any,
    options?: object, 可
    callback?: function, 可回调
)
```

任何扩展部分都可以接收到,除了content\_scripts不能接收到;

### 2、某个浏览器的某个页签发消息的

#### 3、监听接收消息的方法

```
chrome.runtime.onMessage.addListener(
  (message:any, sender:MessageSender, sendResponse:func) => boolean |
  undefined
)
```

## 二、长期连接

1、内容脚本向扩展程序的各个部分:发长连接消息 chrome.runtime.connect(extensionId?, connectInfo?): Port

```
JS content.js ×
                                                                                     JS service-worker.js ×
       console.log("向background发送消息"
                                                                                             console.log("监听content_script的消息")
                                                                                              chrome.runtime.onConnect.addListener(function(port) {
       port.postMessage({name: "内容脚本的信息"});
                                                                                               console.assert(port.name === "来自content");
       port.onMessage.addListener(function(msg)
                                                                                               port.onMessage.addListener(function(msg) {
         ·console.log("当前接收到background的信息",msg
·if (msg.question ==== "background说1")
                                                                                                 console.log("当前接收到content的信息<mark>:</mark>",msg)
if (msg.name ==== "内容脚本的信息")
           port.postMessage({answer: "2"});
                                                                                                 port.postMessage({question: "background说1"});
else if (msg.answer === "2")
         else if (msg.question === "backgrou
                                                                                                 port.postMessage({question: "background收到2"});
else if (msg.answer === "3")
                                                                                                   port.postMessage({question: "background接收完毕<mark>[</mark>"});
           向background发送消息
           当前接收到background的信息
▶ {question: 'background说1'}
                                          content.js:5
                                                                                                监听content_script的消息
           当前接收到background的信息
                                          content.js:5
                       'background收到2'}
                                                                                                当前接收到content的信息: ▶ {name: '内容脚本的信息'}
           当前接收到content的信息: ▶ {answer: '2'}
                                                                                                当前接收到content的信息: ▶ {answer: '3'}
```

1. 建立连接时,系统会为每一端分配一个 runtime.Port 对象,用于通过该连接发 送和接收消息

### 2、扩展程序向内容脚本:某个浏览器的某个页签发消息的

chrome.tabs.connect(tabld, connectInfo?): Port

1. 只有 content\_scripts 可以接收到,必须之名 tabld;

### 3、监听长连接消息

chrome.runtime.onConnect.addListener(function(port) {port//来自发消息方的信息}

## 三、端口

1、tabs.connect()、runtime.connect() 或 runtime.connectNative() 时,返回一个 port可以直接调用 postMessage() 发送消息。

# 网络请求

- 1. 内容脚本也受同源政策的约束。
- 2. 扩展程序的来源不受限制。在扩展程序 Service Worker 或前台标签页中执行的脚本可以与其源之外的远程服务器通信
  - 1. 前提是该扩展程序请求跨源权限