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

chrome.**tabs.sendMessage**(
  tabId: number,指定页签
  message: any,
  options?: object,可
  callback?: function,可回调
)
只有content_scripts可以接收到，必须之名tabId；

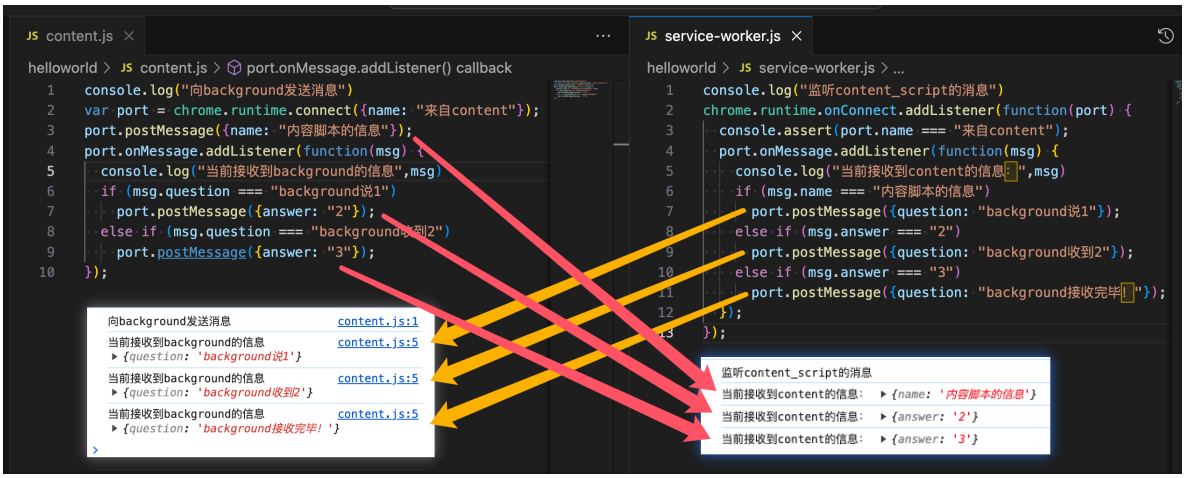
3、监听接收消息的方法

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



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(tabId, connectInfo?): Port

1. 只有 content_scripts 可以接收到，必须之名 tabId；

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