blog #1

<[原文](https://stackoverflow.com/questions/13777887/call-background-function-of-chrome-extension-from-a-site)>

##### 問題：

在網頁中，啟動chrome extension.例如，

檔案：http://www.example.com/test.html 如下：

**<script>**

hello();

**</script>**

而且background page 中有下列的定義:

function hello() {

 alert("test");

}

如何確定，test.html 中的hello() 呼叫的是背景網頁中定義的函數hello() ?

##### 提示：

Before a web page is able to call a background page's function, the following problems need to be solved:

* Be able to use hello(); from a web page. This is done by [injecting](https://stackoverflow.com/questions/9515704/building-a-chrome-extension-inject-code-in-a-page-using-a-content-script/9517879#9517879) a script defining hellousing Content scripts. The injected function communicates with the content script using a custom event or postMessage.
* The content script needs to communicate with the background. This is implemented through [chrome.runtime.sendMessage](http://developer.chrome.com/extensions/runtime#method-sendMessage).
If the web page needs to receive a reply as well:
* Send a reply from the background page (sendMessage / onMessage, see below).
* In the content script, create a custom event or use postMessage to send a message to the web page.
* In the web page, handle this message.

**All of these methods are asynchronous, and have to be implemented through callback functions.**

These steps need to be designed carefully. Here's a generic implementation which implements all of the above steps. What you need to know about the implementation:

* In the code-to-be-injected, use the sendMessage method whenever the content script need to be contacted.
Usage: sendMessage(<mixed message> [, <function callback>])

contentscript.js

// Random unique name, to be used to minimize conflicts:

**var** EVENT\_FROM\_PAGE **=** '\_\_rw\_chrome\_ext\_' **+** **new** Date().getTime();

**var** EVENT\_REPLY **=** '\_\_rw\_chrome\_ext\_reply\_' **+** **new** Date().getTime();

**var** s **=** document.createElement('script');

s.textContent **=** '(' **+** **function**(send\_event\_name, reply\_event\_name) {

 // NOTE: This function is serialized and runs in the page's context

 // Begin of the page's functionality

 window.hello **=** **function**(string) {

 sendMessage({

 type: 'sayhello',

 data: string

 }, **function**(response) {

 alert('Background said: ' **+** response);

 });

 };

 // End of your logic, begin of messaging implementation:

 **function** sendMessage(message, callback) {

 **var** transporter **=** document.createElement('dummy');

 // Handles reply:

 transporter.addEventListener(reply\_event\_name, **function**(event) {

 **var** result **=** **this**.getAttribute('result');

 **if** (**this**.parentNode) **this**.parentNode.removeChild(**this**);

 // After having cleaned up, send callback if needed:

 **if** (**typeof** callback **==** 'function') {

 result **=** JSON.parse(result);

 callback(result);

 }

 });

 // Functionality to notify content script

 **var** event **=** document.createEvent('Events');

 event.initEvent(send\_event\_name, **true**, **false**);

 transporter.setAttribute('data', JSON.stringify(message));

 (document.body||document.documentElement).appendChild(transporter);

 transporter.dispatchEvent(event);

 }

} **+** ')(' **+** JSON.stringify(/\*string\*/EVENT\_FROM\_PAGE) **+** ', ' **+**

 JSON.stringify(/\*string\*/EVENT\_REPLY) **+** ');';

document.documentElement.appendChild(s);

s.parentNode.removeChild(s);

// Handle messages from/to page:

document.addEventListener(EVENT\_FROM\_PAGE, **function**(e) {

 **var** transporter **=** e.target;

 **if** (transporter) {

 **var** request **=** JSON.parse(transporter.getAttribute('data'));

 // Example of handling: Send message to background and await reply

 chrome.runtime.sendMessage({

 type: 'page',

 request: request

 }, **function**(data) {

 // Received message from background, pass to page

 **var** event **=** document.createEvent('Events');

 event.initEvent(EVENT\_REPLY, **false**, **false**);

 transporter.setAttribute('result', JSON.stringify(data));

 transporter.dispatchEvent(event);

 });

 }

});

background.js

chrome.runtime.onMessage.addListener(**function**(message, sender, sendResponse) {

 **if** (message && message.type **==** 'page') {

 **var** page\_message **=** message.message;

 // Simple example: Get data from extension's local storage

 **var** result **=** localStorage.getItem('whatever');

 // Reply result to content script

 sendResponse(result);

 }

});

【manifest.json】

{

 "name": "Page to background and back again",

 "version": "1",

 "manifest\_version": 2,

 "background": {

 "scripts": ["background.js"]

 },

 "content\_scripts": [{

 "matches": ["<all\_urls>"],

 "js": ["contentscript.js"],

 "all\_frames": **true**,

 "run\_at": "document\_start"

 }]

}

執行測試網頁test.html ，會出現對話框說：

"Background said: null" 。

只要 打開背景網頁：然後在console 中打入：

localStorage.setItem('whatever', 'Hello!');

就可以正確顯示。