Adding complex HTML using a Chrome content script

Asked 8 years ago Active 4 years, 2 months ago Viewed 16k times

22

21

I am working with Chrome extension's content script to create a complex display that is added on web pages.

I have first tested it directly integrated on a website, but now I need to put it in an extension.

The thing is that the content script API for Chrome only allows to inject javascript. That means that, to inject complex HTML layouts I would need to write it entirely with JS objects, which is long to write, hard to maintain and absolutely not designer-friendly.

I'm wondering if anyone know or can think of a clever way to get a better workflow on this.

javascript html google-chrome google-chrome-extension content-script

Share Improve this question Follow

edited Apr 8 '13 at 10:23
Brock Adams
82k ● 19 ● 202 ● 264

asked Apr 8 '13 at 7:48
Malharhak
1,482 ● 2 ● 15 ● 25

3 Answers

Active Oldest Votes

37

✓

It's relatively easy to add whole web pages by having your content script inject them in an iframe. Just follow these guidelines:

1. Place the `*.htm` or `*.html` files in your extension's source folder(s).
2. Place any `*.css` and `*.js` files, that the HTML uses, in the extension folder(s) too.
3. Declare the HTML file(s) as resources. EG:

```
"web_accessible_resources": ["Embedded_Hello_world.htm"]
```

4. Do not use any ***inline***, or external server, javascript in your HTML files. This avoids problems with [*the Content Security Policy* (CSP)](https://developer.chrome.com/extensions/contentSecurityPolicy).
5. This question doesn't cover communicating with the page/iframe, but if you want to do that, it is a bit more involved. Search here on SO; it's been covered many times.

2. Download jQuery into it.
3. Create the 5 files as specified below.
4. Load the unpacked extension (You can see similar steps in this answer.)
5. Reload this page in Chrome; you'll see the "Hello World" page, embedded at the top.

Create these files in the extension folder:

manifest.json:

```
{
    "manifest_version":         2,
    "content_scripts":          [ {
        "js":       [ "iframeInjector.js" ],
        "matches":  [   "https://stackoverflow.com/questions/*"
        ]
    } ],
    "description":              "Inject a complete, premade web page",
    "name":                     "Inject whole web page",
    "version":                  "1",
    "web_accessible_resources": ["Embedded_Hello_world.htm"]
}
```

iframeInjector.js:

```
var iFrame  = document.createElement ("iframe");
iFrame.src  = chrome.extension.getURL ("Embedded_Hello_world.htm");

document.body.insertBefore (iFrame, document.body.firstChild);
```

Embedded_Hello_world.htm:

```
<!DOCTYPE html>
<html><head>
    <title>Embedded Hello World</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">

    <link href="HelloWorld.css" rel="stylesheet" type="text/css">

    <script type="text/javascript" src="jquery.min.js"></script>
    <script type="text/javascript" src="HelloWorld.js"></script>
</head><body>
<p>Hello World!</p>
</body></html>
```

HelloWorld.js:

```
$(document).ready (jQueryMain);

function jQueryMain () {
    $("body").append ('<p>Added by jQuery</p>');
}
```

Share Improve this answer Follow

edited May 23 '17 at 12:02
Community ♦
1 ● 1

answered Apr 8 '13 at 10:17
Brock Adams
82k ● 19 ● 202 ● 264

1 Thanks for this answer. But the problem is that I need a lot of communication between the page and my script. I think that messaging through an iFrame each UI event (I need to detect mouvemoves, keystrokes etc) is gonna be really complicated. – Malharhak Apr 10 '13 at 11:03

@Malharhak did you find a answer? how communication between the iframe and content script? – NamNamNam May 13 '16 at 9:25

Sorry I really don't remember, this was a long time ago. I did find a way to make everything work though. I don't think I went with iFrames though, I was just injecting html in js. – Malharhak May 16 '16 at 15:14

Suppose if i need to query from background.js from this html. How do i do? I get chrome.tabs.query as undefined? Could you please help? – uhs Dec 12 '16 at 10:03

2 ATTENTION: This works as of September 2019, but use `chrome.runtime.getURL` instead of `chrome.extension.getURL`. See stackoverflow.com/questions/32344868/… – Pavindu Sep 22 '19 at 19:12

3

This may be better, no external library and no iframe. Is nearly the same as iautomation solution.

```
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
        var div = document.createElement('div');
        div.innerHTML = this.responseText;
        document.body.insertBefore(div, document.body.firstChild);
    } else {
        console.log('files not found');
    }
};
xhttp.open("GET", chrome.extension.getURL("/content.htm"), true);
xhttp.send();
```

Share Improve this answer Follow

answered Jan 17 '17 at 8:59
Photonic
1,054 ● 13 ● 27

I had the same issue, that my extension heavily relies on script templates

2

Here's what I did:

- create `templates.html` to store script templates in
- add `templates.html` to the `web_accessible_resources` as in the the above answer^^
- access `templates.html` from `content.js` with xhr and parse with jQuery

manifest.json

```
"web_accessible_resources": ["templates.html"]
```

templates.html

```
<script id="template1" type="text/template">
    <div class="template1">template1</div>
</script>
<script id="template2" type="text/template">
    <div class="template2">template2</div>
</script>
```

content.js

```
function getTemplates(){
    return new Promise(function(resolve){
        $.ajax({
            url: chrome.extension.getURL('/templates.html'),
            success: function(data) {
                var $templates = $('<div>
</div>').append($.parseHTML(data)).find('script'),
                    templates = {};
                $templates.each(function(){
                    templates[this.id] = this.innerHTML;
                });
                return resolve(templates);
            }
        });
    });
}
getTemplates().then(function(templates){
    console.log(templates.template1); //<div class="template1">template1</div>
});
```

Share Improve this answer Follow

answered May 10 '16 at 0:01

iautomation

820 ● 10 ● 16