(SIMPLE)

Message Passing in Chrome Extensions

The Problem

- Very easy to do either/both of the following:

The Problem

- **Very easy to do either/both of the following:**

Run script on the tab and manipulate tab DOM

The Problem

- **Very easy to do either/both of the following:**

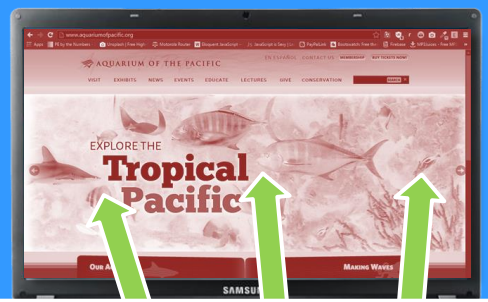
Run script on the tab and manipulate tab DOM

Run script on the extension/popup and manipulate extension DOM

The Problem

- **Very easy to do either/both of the following:**

Run script on the tab and manipulate tab DOM



Not As Easy to Interact Between Scripts/DOMs

Run script on the extension/popup and manipulate extension DOM

Basic File Structure / Directory

manifest.json

- assigns the responsibilities of the other files

content.js

- runs on tabs (as allowed by manifest)
- responsible for tab behavior

popup.html

- extension DOM

popup.js

- runs on extension popup.html
- responsible for extension behavior

Event Sequence

manifest assigns permissions

- On Chrome launch, manifest assigns responsibilities to other files (if extension is enabled)

manifest.json

```
{
  "manifest_version": 2,
  "name": "Example Extension",
  "version": "1.0",
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["jquery-2.2.2.min.js", "content.js"]
  }],
  "permissions": [
    "activeTab"
  ],
  "browser_action": {
    "default_popup": "popup.html"
  }
}
```

Event Sequence

manifest assigns permissions

- On Chrome launch, manifest assigns responsibilities to other files (if extension is enabled)

JS content.js:
Run on chrome tabs (specifically active tab in this case)

manifest.json

```
{
  "manifest_version": 2,
  "name": "Example Extension",
  "version": "1.0",
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["jquery-2.2.2.min.js", "content.js"]
  }],
  "permissions": [
    "activeTab"
  ],
  "browser_action": {
    "default_popup": "popup.html"
  }
}
```

Event Sequence

manifest assigns permissions

- On Chrome launch, manifest assigns responsibilities to other files (if extension is enabled)

content.js:
Run on chrome tabs (specifically active tab in this case)

popup.html:
Launch when the browser_action event occurs (extension icon in nav. ribbon is clicked)

manifest.json

```
{
  "manifest_version": 2,
  "name": "Example Extension",
  "version": "1.0",
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["jquery-2.2.2.min.js", "content.js"]
  }],
  "permissions": [
    "activeTab"
  ],
  "browser_action": {
    "default_popup": "popup.html"
  }
}
```

Event Sequence

manifest assigns permissions

- On Chrome launch, manifest assigns responsibilities to other files (if extension is enabled)

content.js:
Run on chrome tabs (specifically active tab in this case)

popup.html:
Launch when the browser_action event occurs (extension icon in nav. ribbon is clicked)

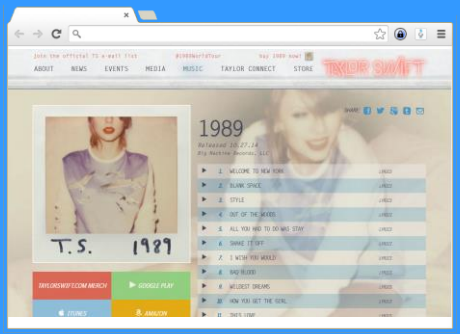
manifest.json

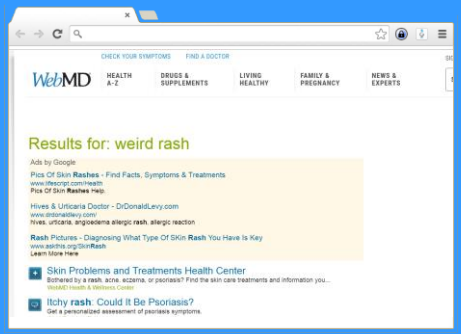
```
{
  "manifest_version": 2,
  "name": "Example Extension",
  "version": "1.0",
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["jquery-2.2.2.min.js", "content.js"]
  }],
  "permissions": [
    "activeTab"
  ],
  "browser_action": {
    "default_popup": "popup.html"
  }
}
```

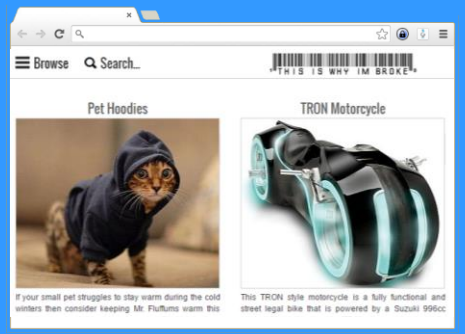
Event Sequence

manifest assigns permissions → content scripts active/listening

- Launch chrome







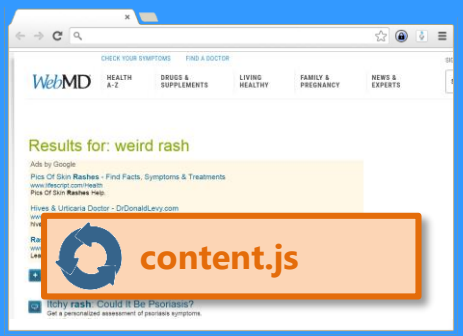
Event Sequence

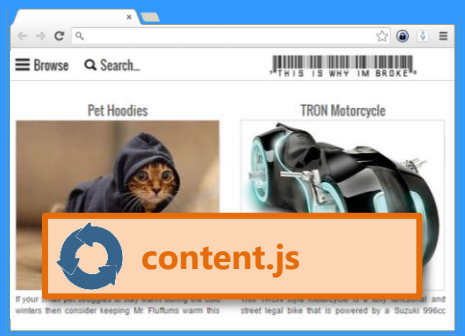
manifest assigns permissions → content scripts active/listening

- Launch chrome
- Assuming extension is enabled…

content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
  if (request.instructions === "giveMeFirstImgPlease") {
    var firstImgTag = 'http:' +
    $('body').find('img').attr('src').replace(/http:|https:/, '');
    var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
    sendResponse(msgBackToExt);
  }
});
```





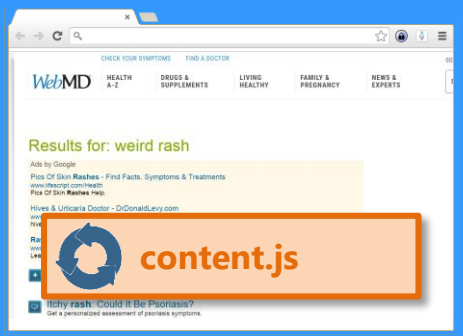
Event Sequence

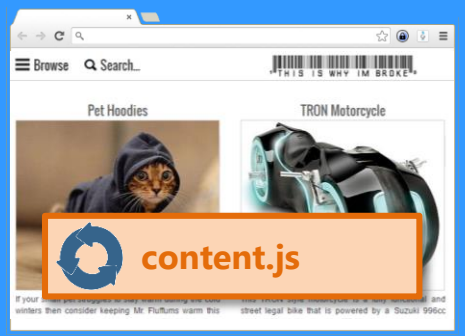
manifest assigns permissions → content scripts active/listening

- Content script running on all tabs where were allowed by manifest

manifest.json

```
{
  "manifest_version": 2,
  "name": "Example Extension",
  "version": "1.0",
  "content_scripts": [{
    "matches": ["<all_urls>"],
    "js": ["jquery-2.2.2.min.js", "content.js"]
  }],
  "permissions": [
    "activeTab"
  ],
  "browser_action": {
    "default_popup": "popup.html"
  }
}
```





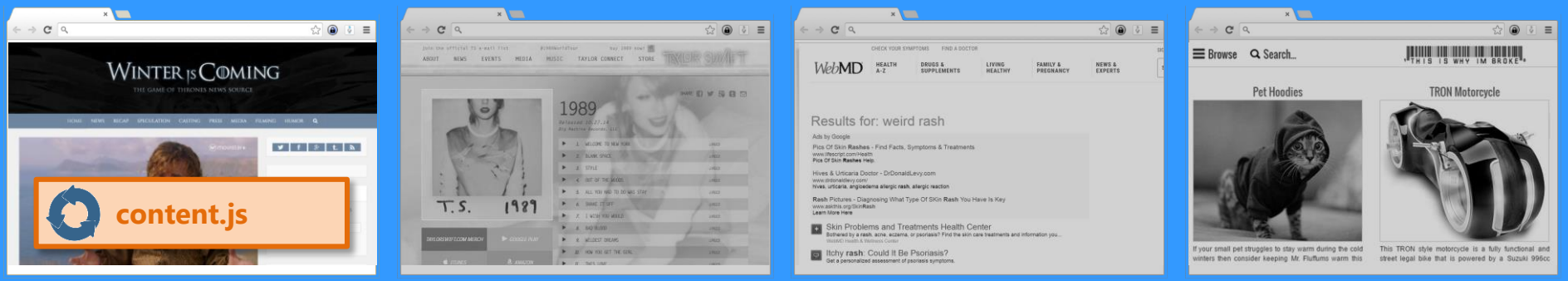
Event Sequence

manifest assigns permissions → content scripts active/listening

- **Content script running on all tabs where were allowed by manifest**

content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
  if (request.instructions === "giveMeFirstImgPlease") {
    var firstImgTag = 'http:' +
    $('body').find('img').attr('src').replace(/http:|https:/, '');
    var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
    sendResponse(msgBackToExt);
  }
});
```



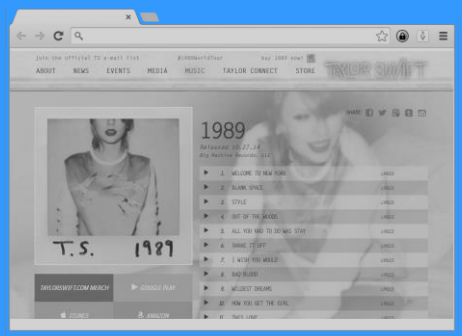
Event Sequence

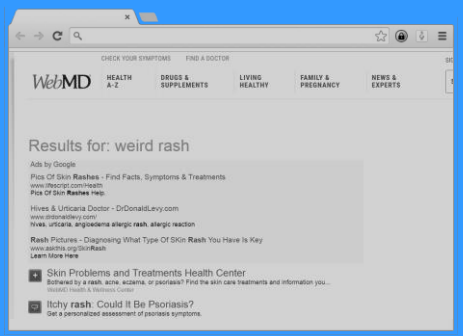
manifest assigns permissions → content scripts active/listening

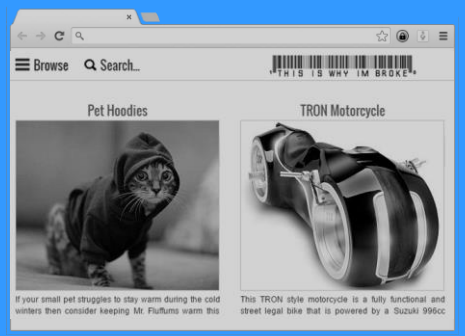
- Content script running on all tabs where were allowed by manifest
- Event listeners active

content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
    if (request.instructions === "giveMeFirstImgPlease") {
      var firstImgTag = 'http:' +
      $('body').find('img').attr('src').replace(/http:|https:/, '');
      var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
      sendResponse(msgBackToExt);
    }
});
```



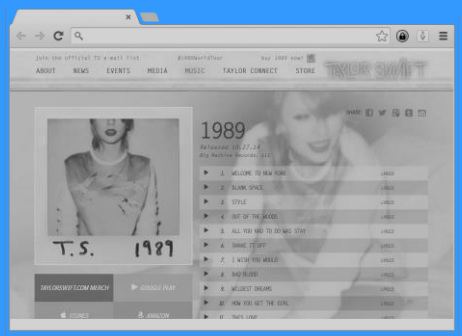


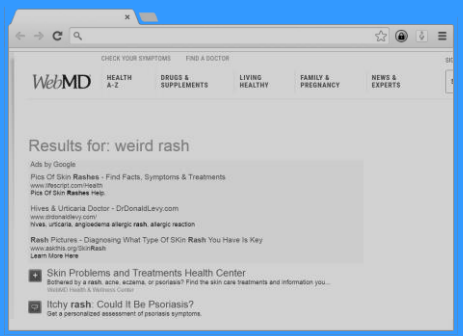


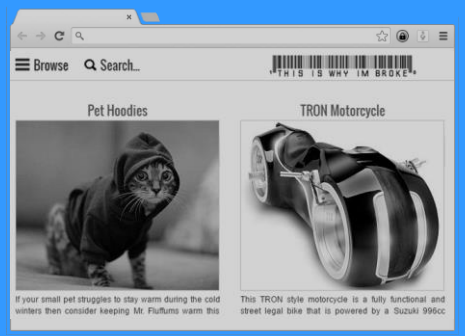
Event Sequence

manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

- **'browser action' renders popup.html**







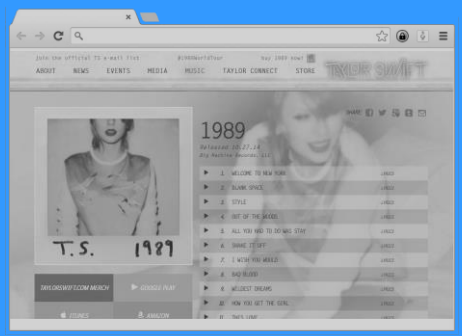
Event Sequence

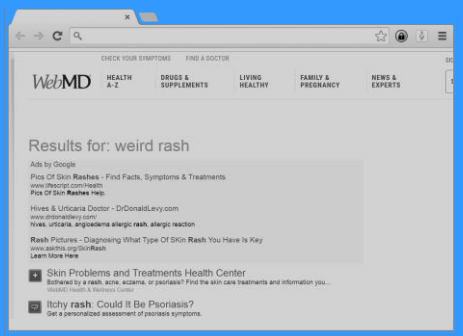
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

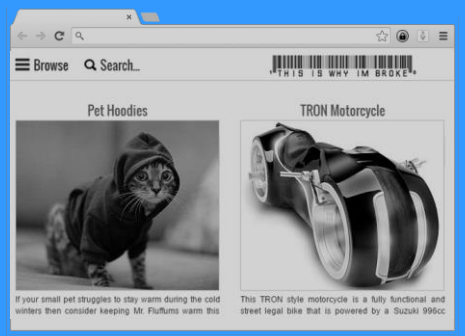
- **'browser action' renders popup.html**

popup.html

```
<!DOCTYPE html>
<html>
<head><title>Example Extension</title>
    <link rel="stylesheet" href="styles.css">
    <script src="jquery-2.2.2.min.js"></script>
    <script src="popup.js"></script>
</head>
<body><h1>Example Extension</h1>
    <button id="btn">Click To Get First Img From Page</button><br>
    <div id="infoFromTabDOM">Img sourced from tab DOM:</div></body>
</html>
```







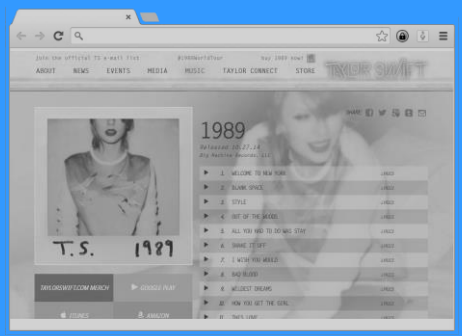
Event Sequence

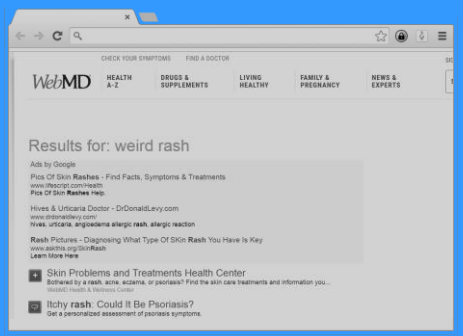
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

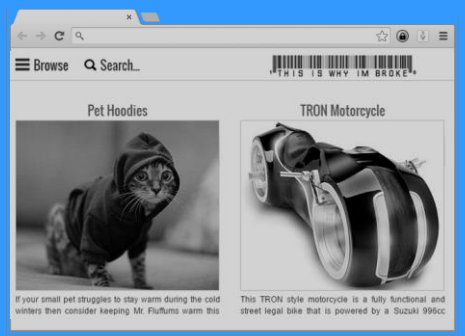
- 'browser action' renders popup.html
- popup.js called

popup.html

```
<!DOCTYPE html>
<html>
<head><title>Example Extension</title>
    <link rel="stylesheet" href="styles.css">
    <script src="jquery-2.2.2.min.js"></script>
    <script src="popup.js"></script>
</head>
<body><h1>Example Extension</h1>
    <button id="btn">Click To Get First Img From Page</button><br>
    <div id="infoFromTabDOM">Img sourced from tab DOM:</div></body>
</html>
```







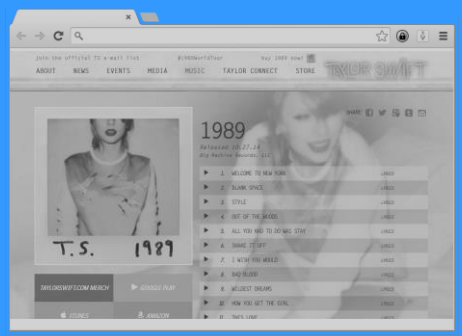
Event Sequence

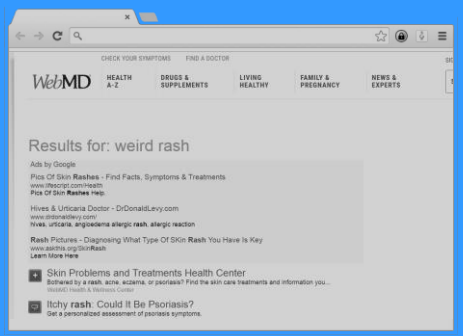
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

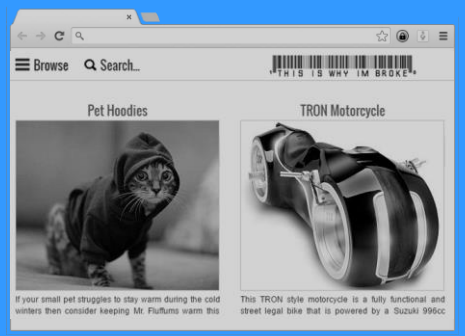
- **popup.js runs**

popup.js

```
1  console.log('hey');
2  $(document).ready(function() {
3    $('#btn').on('click', function() {
4      var messageToTab = {instructions: "giveMeFirstImgPlease"};
5      chrome.tabs.getSelected(null, function(tab) {
6        chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
7          $('img').remove();
8          $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
9        });
10     });
11   });
12 });
```







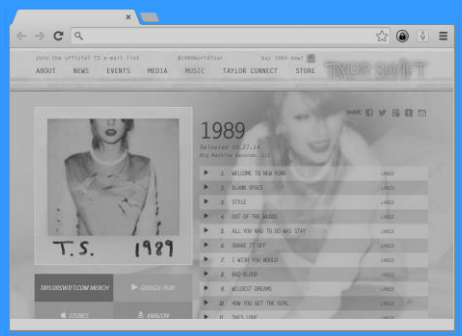
Event Sequence

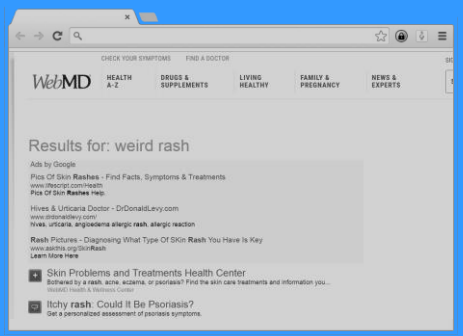
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

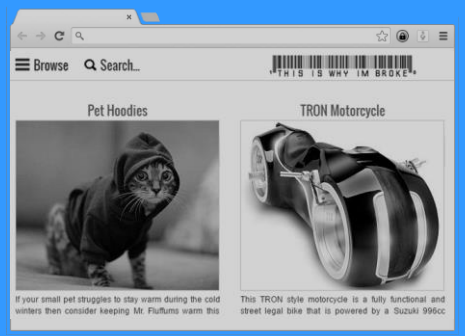
- popup.js runs
- sends out a message to the tab

popup.js

```
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```







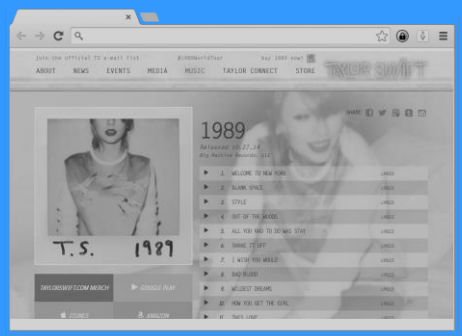
Event Sequence

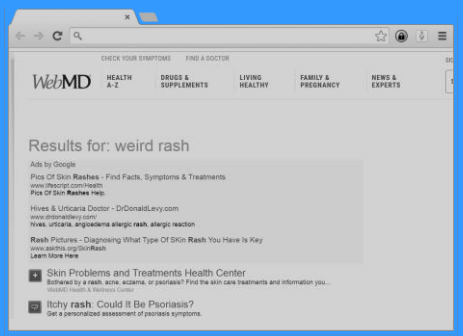
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

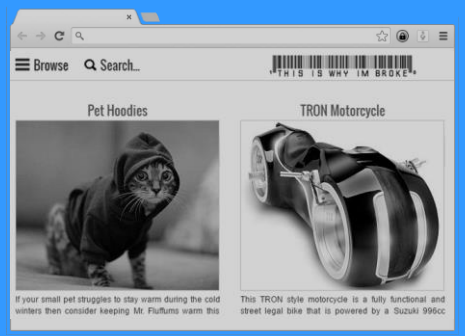
- popup.js runs
- sends out a message to the tab

popup.js

```
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('img').remove();
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```







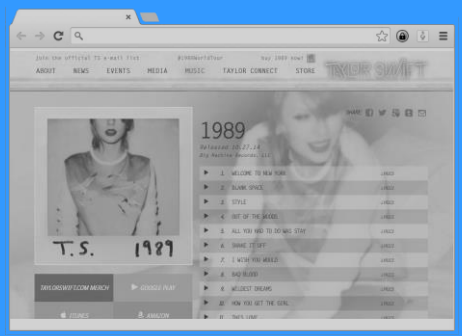
Event Sequence

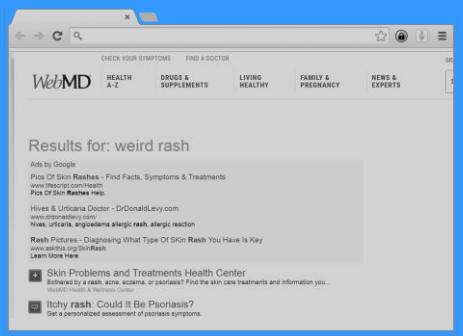
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

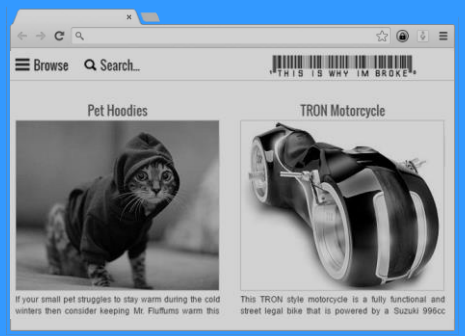
- popup.js runs
- sends out a message to the tab

popup.js

```
console.log('hey');
$(document).ready(function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
        chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
          $('img').remove();
          $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
        });
      });
    });
});
```







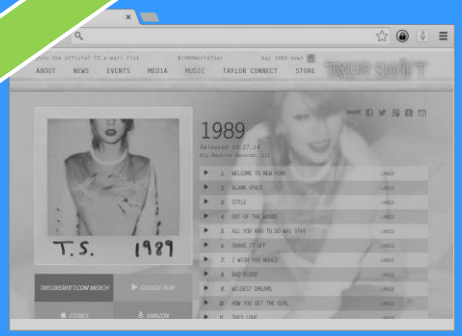
Event Sequence

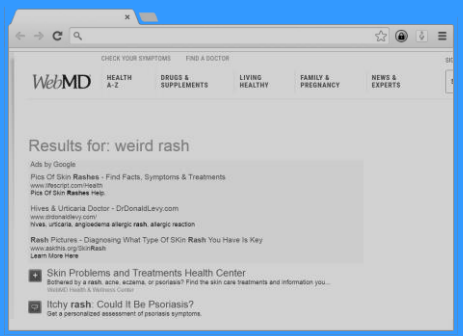
manifest assigns permissions → content scripts active/listening → **popup & popup.js request initiated**

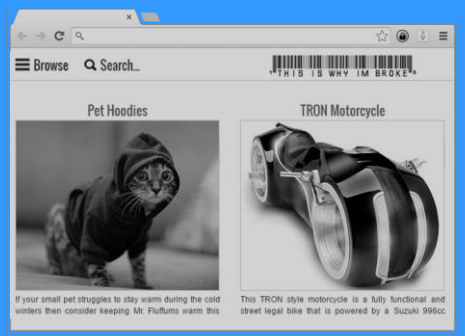
- popup.js runs
- sends out a message to the tab

popup.js

```
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```







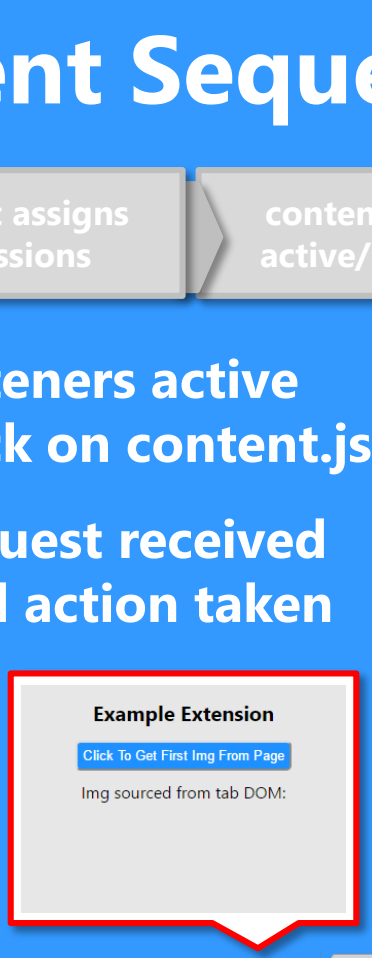
Event Sequence

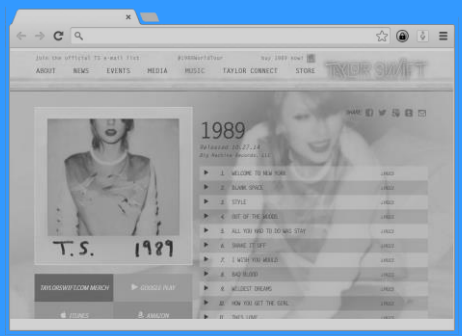
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → **content receives request, executes, and responds**

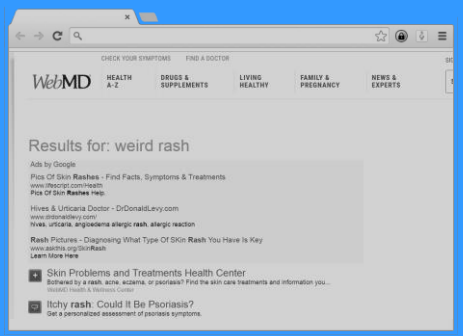
- **Listeners active back on content.js**
- **request received and action taken**

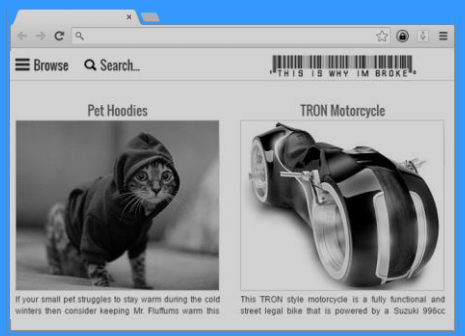
content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
  if (request.instructions === "giveMeFirstImgPlease") {
    var firstImgTag = 'http:' +
    $('body').find('img').attr('src').replace(/http:|https:/, '');
    var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
    sendResponse(msgBackToExt);
  }
});
```









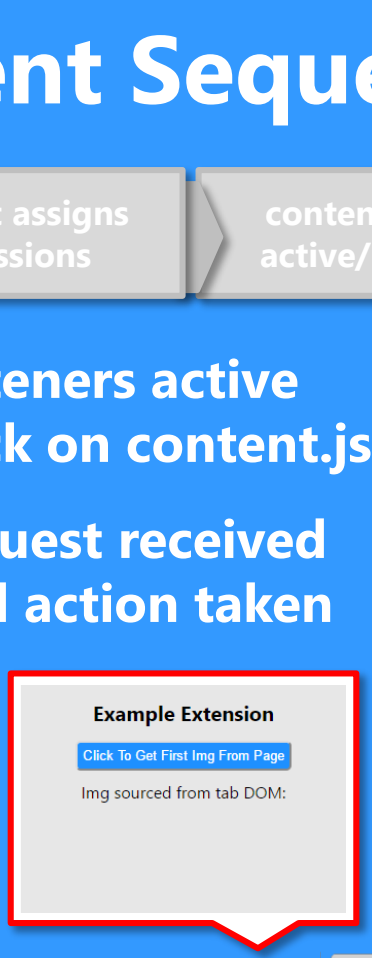
Event Sequence

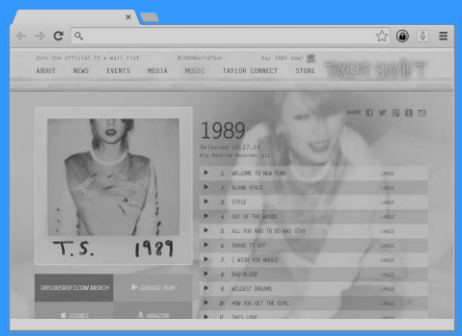
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → **content receives request, executes, and responds**

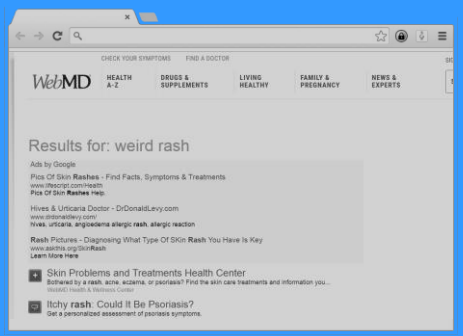
- Listeners active back on content.js
- request received and action taken

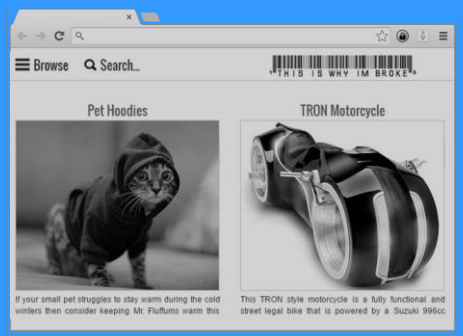
content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
    if (request.instructions === "giveMeFirstImgPlease") {
      var firstImgTag = 'http:' +
      $('body').find('img').attr('src').replace(/http:|https:/, '');
      var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
      sendResponse(msgBackToExt);
    }
});
```









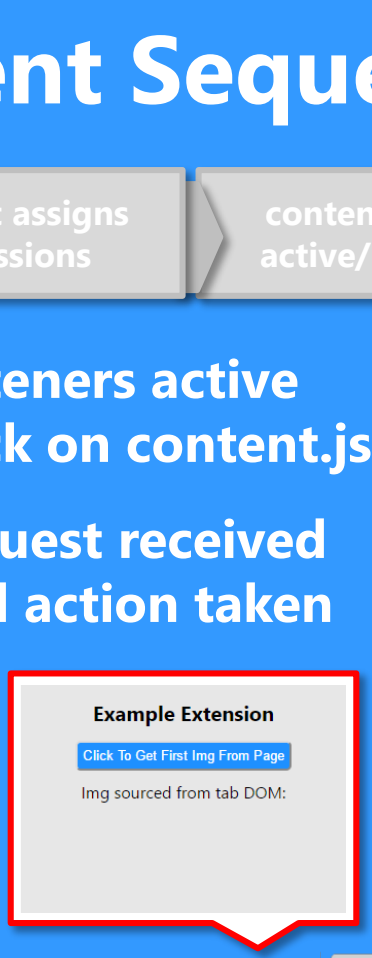
Event Sequence

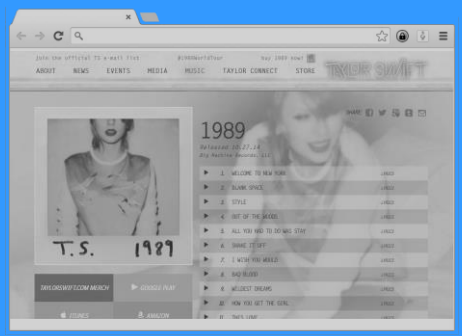
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → **content receives request, executes, and responds**

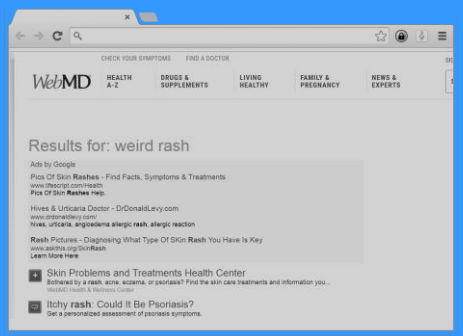
- Listeners active back on content.js
- request received and action taken

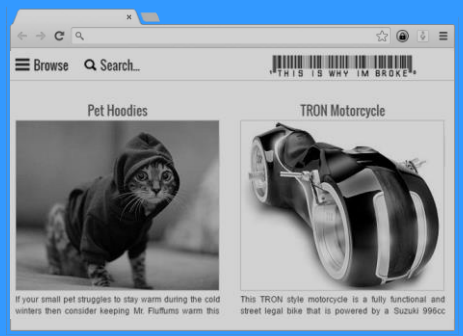
content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
  if (request.instructions === "giveMeFirstImgPlease") {
    var firstImgTag = 'http:' +
    $('body').find('img').attr('src').replace(/http:|https:/, '');
    var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
    sendResponse(msgBackToExt);
  }
});
```









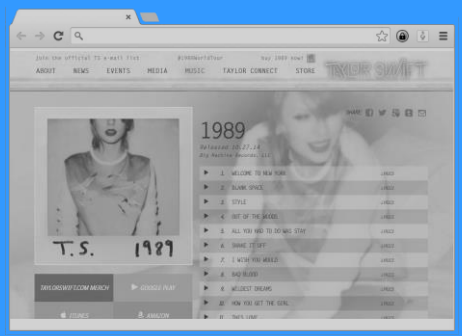
Event Sequence

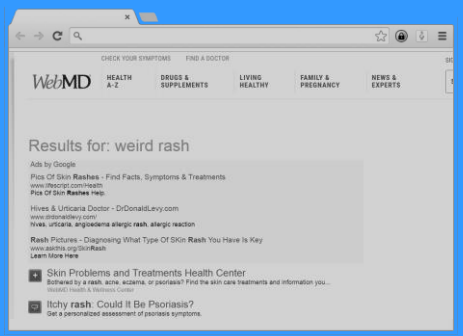
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → **content receives request, executes, and responds**

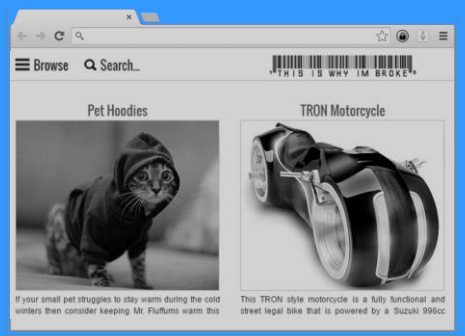
- **send message back to popup.js**

content.js

```
chrome.runtime.onMessage.addListener(function(request, sender, sendResponse) {
  if (request.instructions === "giveMeFirstImgPlease") {
    var firstImgTag = 'http:' +
    $('body').find('img').attr('src').replace(/http:|https:/, '');
    var msgBackToExt = {fromTab: "imgTagAsRequested", imgTagSrc: firstImgTag};
    sendResponse(msgBackToExt);
  }
});
```







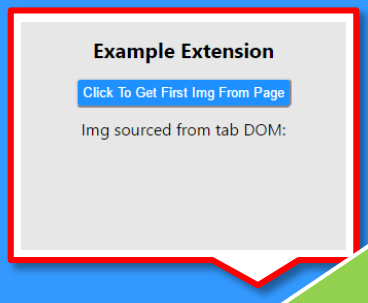
Event Sequence

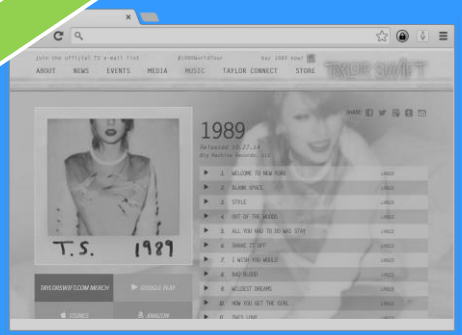
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → content receives request, executes, and responds → **popup.js receives response and continues execution**

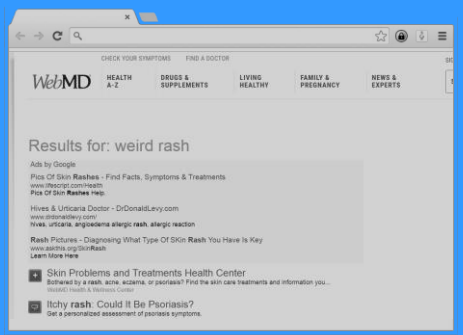
- **popup.js receives message**

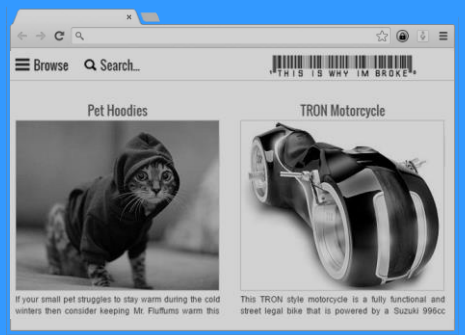
popup.js

```
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('img').remove();
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```







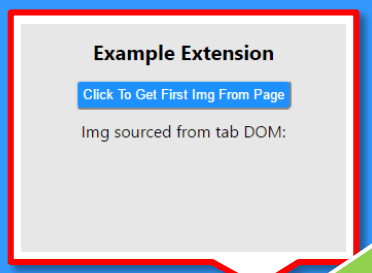


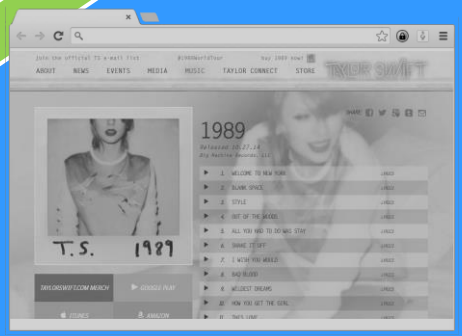
Event Sequence

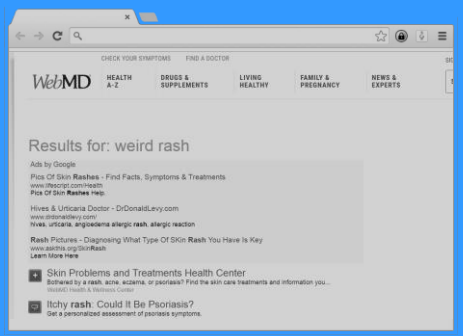
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → content receives request, executes, and responds → popup.js receives response and continues execution

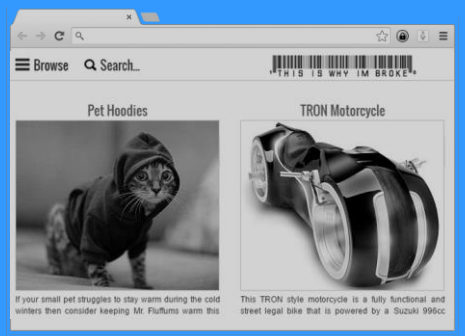
- popup.js receives message
- continues with callback execution

```
popup.js
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('img').remove();
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```









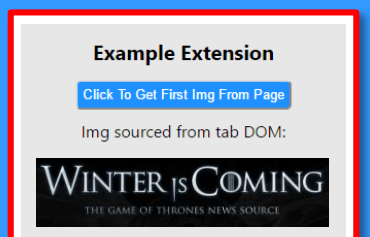
Event Sequence

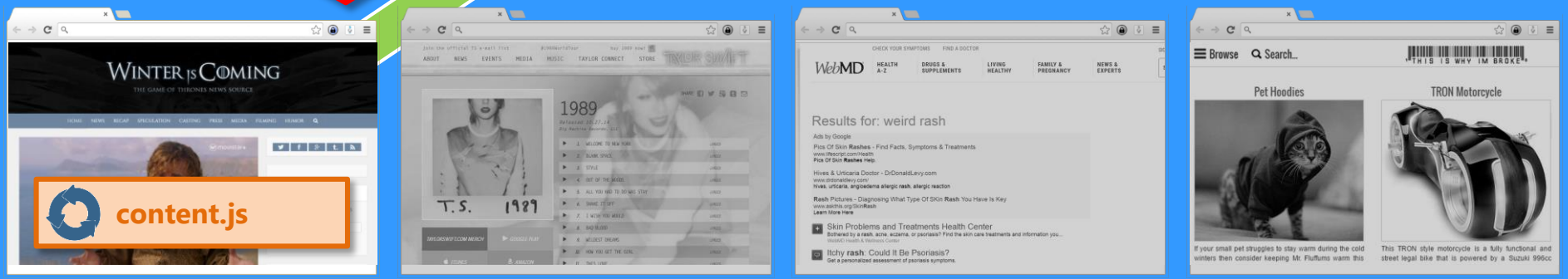
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → content receives request, executes, and responds → popup.js receives response and continues execution

- popup.js' running on popup.html allows use of data received from tab

popup.js

```
console.log('hey');
$(document).ready(function() {
  $('#btn').on('click', function() {
    var messageToTab = {instructions: "giveMeFirstImgPlease"};
    chrome.tabs.getSelected(null, function(tab) {
      chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
        $('img').remove();
        $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
      });
    });
  });
});
```





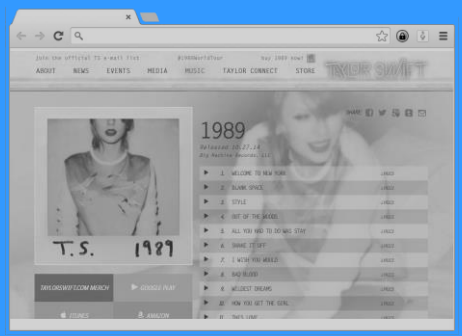
Event Sequence

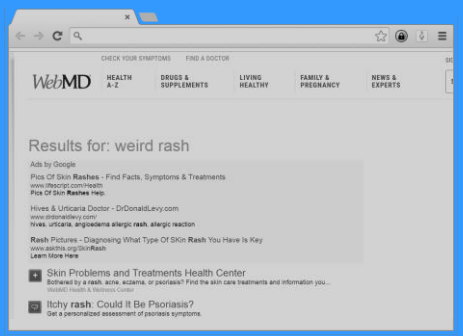
manifest assigns permissions → content scripts active/listening → popup & popup.js request initiated → content receives request, executes, and responds → popup.js receives response and continues execution

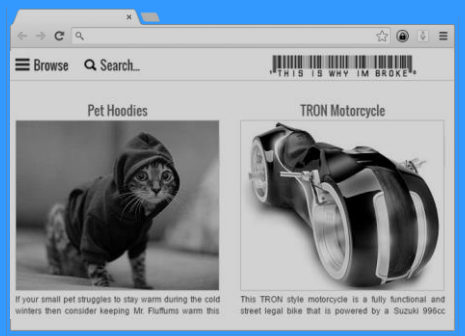
- popup.js' running on popup.html allows use of data received from tab

popup.js

```
1  console.log('hey');
2  $(document).ready(function() {
3    $('#btn').on('click', function() {
4      var messageToTab = {instructions: "giveMeFirstImgPlease"};
5      chrome.tabs.getSelected(null, function(tab) {
6        chrome.tabs.sendMessage(tab.id, messageToTab, function(res) {
7          $('img').remove();
8          $('#infoFromTabDOM').append('<img src="'+res.imgTagSrc+'"/>');
9        });
10     });
11   });
12 });
```







Other Message Passing

- Long-lived connections
  - chrome.runtime.connect or chrome.tabs.connect
  - onMessage.addListener
  - postMessage
- Cross-extension messaging (& LL equivalent)
  - runtime.onMessageExternal or runtime.onConnectExternal
  - addListener
  - sendResponse

- https://developer.chrome.com/extensions/messaging
- https://github.com/apeltz/chromeMessagePassing