chrome.pageAction

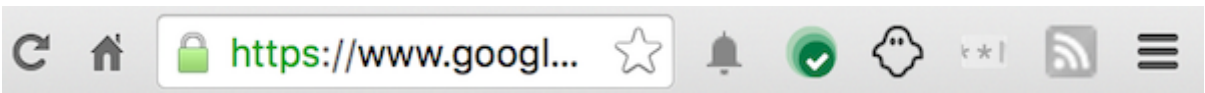
Description:	Use the `chrome.pageAction` API to put icons in the main Google Chrome toolbar, to the right of the address bar. Page actions represent actions that can be taken on the current page, but that aren't applicable to all pages. Page actions appear grayed out when inactive.
Availability:	Since Chrome 27.
Manifest:	`"page_action": {...}`

Some examples:

- Subscribe to this page's RSS feed
- Make a slideshow out of this page's photos

The RSS icon in the following screenshot represents a page action that lets you subscribe to the RSS feed for the current page.

Hidden page actions appear grayed out. For example, the RSS feed below is grayed out, as you can't subscribe to the feed for the current page:



Please consider using a **browser action** instead, so that users can always interact with your extension.

Manifest

Register your page action in the **extension manifest** like this:

```
{
  "name": "My extension",
  ...
  "page_action": {
```

```
  "page_action": {
    "default_icon": {                    // optional
      "16": "images/icon16.png",           // optional
      "24": "images/icon24.png",           // optional
      "32": "images/icon32.png"            // optional
    },
    "default_title": "Google Mail",      // optional; shown in tooltip
    "default_popup": "popup.html"        // optional
  },
  ...
}
```

Since devices with less-common scale factors like 1.5x or 1.2x are becoming more common, you are encouraged to provide multiple sizes for your icons. Chrome will select the closest one and scale it to fill the 16-dip space. This also ensures that if the icon display size is ever changed, you don't need to do any more work to provide different icons! However, if the size difference is too extreme, this scaling can cause the icon to lose detail or look fuzzy.

The old syntax for registering the default icon is still supported:

```
{
  "name": "My extension",
  ...
  "page_action": {
    ...
    "default_icon": "images/icon32.png"  // optional
    // equivalent to "default_icon": { "32": "images/icon32.png" }
  },
  ...
}
```

Parts of the UI

Like browser actions, page actions can have an icon, a tooltip, and popup; they can't have badges, however. In addition, page actions can be grayed out. You can find information about icons, tooltips, and popups by reading about the **browser action UI**.

You make a page action appear and be grayed out using the **pageAction.show** and **pageAction.hide** methods, respectively. By default, a page action appears grayed out. When you show it, you specify the tab in which the icon should appear. The icon remains visible until the tab is closed or starts displaying a different URL (because the user clicks a link, for example).

Tips

For the best visual impact, follow these guidelines:

- **Do** use page actions for features that make sense for only a few pages.
- **Don't** use page actions for features that make sense for most pages. Use **browser actions** instead.
- **Don't** constantly animate your icon. That's just annoying.

Examples

You can find simple examples of using page actions in the **examples/api/pageAction** directory. For other examples and for help in viewing the source code, see **Samples**.

Summary

Types
ImageDataType
Methods
show – `chrome.pageAction.show(integer tabId, function callback)`
hide – `chrome.pageAction.hide(integer tabId, function callback)`
setTitle – `chrome.pageAction.setTitle(object details, function callback)`
getTitle – `chrome.pageAction.getTitle(object details, function callback)`
setIcon – `chrome.pageAction.setIcon(object details, function callback)`
setPopup – `chrome.pageAction.setPopup(object details, function callback)`
getPopup – `chrome.pageAction.getPopup(object details, function callback)`
Events
onClicked

Types

ImageDataType

Pixel data for an image. Must be an ImageData object (for example, from a `canvas` element).

Methods

show

```
chrome.pageAction.show(integer tabId, function callback)
```

Shows the page action. The page action is shown whenever the tab is selected.

Parameters		
integer	tabId	The id of the tab for which you want to modify the page action.
function	(optional) callback	If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`

hide

```
chrome.pageAction.hide(integer tabId, function callback)
```

Hides the page action. Hidden page actions still appear in the Chrome toolbar, but are grayed out.

Parameters		
integer	tabId	The id of the tab for which you want to modify the page action.
function	(optional) callback	If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`

setTitle

```
chrome.pageAction.setTitle(object details, function callback)
```

Sets the title of the page action. This is displayed in a tooltip over the page action.

Parameters

object	details	integer tabId — The id of the tab for which you want to modify the page action. string title — The tooltip string.
function	(optional) callback	If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`

getTitle

```
chrome.pageAction.getTitle(object details, function callback)
```

Gets the title of the page action.

Parameters

object	details	integer tabId — Specify the tab to get the title from.
function	callback	The *callback* parameter should be a function that looks like this: `function(string result) {...};` string result

setIcon

```
chrome.pageAction.setIcon(object details, function callback)
```

Sets the icon for the page action. The icon can be specified either as the path to an image file or as the pixel data from a canvas element, or as dictionary of either one of those. Either the **path** or the **imageData** property must be specified.

Parameters

object	details	integer tabId — The id of the tab for which you want to modify the

				page action.
		ImageDataType or object	(optional) imageData	Either an ImageData object or a dictionary {size -> ImageData} representing icon to be set. If the icon is specified as a dictionary, the actual image to be used is chosen depending on screen's pixel density. If the number of image pixels that fit into one screen space unit equals `scale`, then image with size `scale` * n will be selected, where n is the size of the icon in the UI. At least one image must be specified. Note that 'details.imageData = foo' is equivalent to 'details.imageData = {'16': foo}'
		string or object	(optional) path	Either a relative image path or a dictionary {size -> relative image path} pointing to icon to be set. If the icon is specified as a dictionary, the actual image to be used is chosen depending on screen's pixel density. If the number of image pixels that fit into one screen space unit equals `scale`, then image with size `scale` * n will be selected, where n is the size of the icon in the UI. At least one image must be specified. Note that 'details.path = foo' is equivalent to 'details.path = {'16': foo}'
		integer	(optional) iconIndex	**Deprecated.** This argument is ignored.
function	(optional) callback	If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`		

setPopup

```
chrome.pageAction.setPopup(object details, function callback)
```

Sets the html document to be opened as a popup when the user clicks on the page action's icon.

Parameters

object	details	integer tabId — The id of the tab for which you want to modify the page action. string popup — The html file to show in a popup. If set to the empty string (''), no popup is shown.
function	(optional) callback	If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`

getPopup

```
chrome.pageAction.getPopup(object details, function callback)
```

Gets the html document set as the popup for this page action.

Parameters

object	details	integer tabId — Specify the tab to get the popup from.
function	callback	The *callback* parameter should be a function that looks like this: `function(string result) {...};` string result

Events

onClicked

Fired when a page action icon is clicked. This event will not fire if the page action has a popup.

addListener

```
chrome.pageAction.onClicked.addListener(function callback)
```

Parameters		
function	callback	The *callback* parameter should be a function that looks like this: `function( tabs.Tab tab) {...};` **tabs.Tab** tab