chrome.extension

Description:	The `chrome.extension` API has utilities that can be used by any extension page. It includes support for exchanging messages between an extension and its content scripts or between extensions, as described in detail in **Message Passing**.
Availability:	Since Chrome 27.
Content Scripts:	**getURL** , **inIncognitoContext** , **lastError** , **onRequest** and **sendRequest** are supported. **Learn more**

Summary

onRequestExternal

Types

ViewType

The type of extension view.

Enum
`"tab"`, or `"popup"`

Properties

object	`chrome.extension.lastError`	**Deprecated** since Chrome 58. Please use **runtime.lastError**. Set for the lifetime of a callback if an ansychronous extension api has resulted in an error. If no error has occured lastError will be *undefined*. **Properties** string — message — Description of the error that has taken place.
boolean	`chrome.extension.inIncognitoContext`	True for content scripts running inside incognito tabs, and for extension pages running inside an incognito process. The latter only applies to extensions with 'split' incognito_behavior.

Methods

sendRequest

```
chrome.extension.sendRequest(string extensionId, any request, function responseCallback)
```

Deprecated since Chrome 33. Please use **runtime.sendMessage**.

Sends a single request to other listeners within the extension. Similar to **runtime.connect**, but only sends a single request with an optional response. The **extension.onRequest** event is fired in each page of the extension.

Parameters

string	(optional) extensionId	Since Chrome 33. The extension ID of the extension you want to connect to. If omitted, default is your own extension.
any	request	Since Chrome 33.
function	(optional) responseCallback	If you specify the *responseCallback* parameter, it should be a function that looks like this: `function(any response) {...};` any — response — The JSON response object sent by the handler of the request. If an error occurs while connecting to the extension, the callback will be called with no arguments and **runtime.lastError** will be set to the error message.

getURL

```
string chrome.extension.getURL(string path)
```

Deprecated since Chrome 58. Please use **runtime.getURL**.

Converts a relative path within an extension install directory to a fully-qualified URL.

Parameters

string	path	A path to a resource within an extension expressed relative to its install directory.

getViews

```
array of Window chrome.extension.getViews(object fetchProperties)
```

Returns an array of the JavaScript 'window' objects for each of the pages running inside the current extension.

Parameters

object	(optional) fetchProperties	

ViewType	(optional) type	The type of view to get. If omitted, returns all views (including background pages and tabs). Valid values: 'tab', 'notification', 'popup'.
integer	(optional) windowId	The window to restrict the search to. If omitted, returns all views.
integer	(optional) tabId	Since Chrome 54. Find a view according to a tab id. If this field is omitted, returns all views.

getBackgroundPage

```
Window chrome.extension.getBackgroundPage()
```

Returns the JavaScript 'window' object for the background page running inside the current extension. Returns null if the extension has no background page.

Returns

getExtensionTabs

```
array of Window chrome.extension.getExtensionTabs(integer windowId)
```

Deprecated since Chrome 33. Please use **extension.getViews** `{type: "tab"}`.

Returns an array of the JavaScript 'window' objects for each of the tabs running inside the current extension. If `windowId` is specified, returns only the 'window' objects of tabs attached to the specified window.

Parameters		
integer	(optional) windowId	

isAllowedIncognitoAccess

```
chrome.extension.isAllowedIncognitoAccess(function callback)
```

Retrieves the state of the extension's access to Incognito-mode (as determined by the user-controlled 'Allowed in Incognito' checkbox.

Parameters		
function	callback	The *callback* parameter should be a function that looks like this: `function(boolean isAllowedAccess) {...};` boolean — isAllowedAccess — True if the extension has access to Incognito mode, false otherwise.

isAllowedFileSchemeAccess

```
chrome.extension.isAllowedFileSchemeAccess(function callback)
```

Retrieves the state of the extension's access to the 'file://' scheme (as determined by the user-controlled 'Allow access to File URLs' checkbox.

Parameters		
function	callback	The *callback* parameter should be a function that looks like this: `function(boolean isAllowedAccess) {...};` boolean — isAllowedAccess — True if the extension can access the 'file://' scheme, false otherwise.

setUpdateUrlData

```
chrome.extension.setUpdateUrlData(string data)
```

Sets the value of the ap CGI parameter used in the extension's update URL. This value is ignored for extensions that are hosted in the Chrome Extension Gallery.

Parameters		
string	data	

Events

onRequest

Deprecated since Chrome 33. Please use **runtime.onMessage**.

Fired when a request is sent from either an extension process or a content script.

addListener

```
chrome.extension.onRequest.addListener(function callback)
```

Parameters				
function	callback	The *callback* parameter should be a function that looks like this: `function(any request, runtime.MessageSender sender, function sendResponse) {...};`		
		any	(optional) request	Since Chrome 33. The request sent by the calling script.
		runtime.MessageSender	sender	Since Chrome 33.
		function	sendResponse	Function to call (at most once) when you have a response. The argument

				should be any JSON-ifiable object, or undefined if there is no response. If you have more than one `onRequest` listener in the same document, then only one may send a response. The *sendResponse* parameter should be a function that looks like this: `function() {...};`

onRequestExternal

Deprecated since Chrome 33. Please use **runtime.onMessageExternal**.

Fired when a request is sent from another extension.

addListener

```
chrome.extension.onRequestExternal.addListener(function callback)
```

Parameters

function	callback	The *callback* parameter should be a function that looks like this: `function(any request, runtime.MessageSender sender, function sendResponse) {...};`		
		any	(optional) request	The request sent by the calling script.
		runtime.MessageSender	sender	
		function	sendResponse	Function to call when you have a

response. The argument should be any JSON-ifiable object, or undefined if there is no response.

The *sendResponse* parameter should be a function that looks like this:

```
function() {...};
```