chrome.devtools.panels

Description:	Use the chrome.devtools.panels API to integrate your extension into Developer Tools window UI: create your own panels, access existing panels, and add sidebars.
Availability:	Since Chrome 31.

See **DevTools APIs summary** for general introduction to using Developer Tools APIs.

Overview

Each extension panel and sidebar is displayed as a separate HTML page. All extension pages displayed in the Developer Tools window have access to all modules in chrome.devtools API, as well as to **chrome.extension** API. Other extension APIs are not available to the pages within Developer Tools window, but you may invoke them by sending a request to the background page of your extension, similarly to how it's done in the **content scripts**.

You can use the devtools.panels.setOpenResourceHandler method to install a callback function that handles user requests to open a resource (typically, a click on a resource link in the Developer Tools window). At most one of the installed handlers gets called; users can specify (using the Developer Tools Settings dialog) either the default behavior or an extension to handle resource open requests. If an extension calls setOpenResourceHandler() multiple times, only the last handler is retained.

Examples

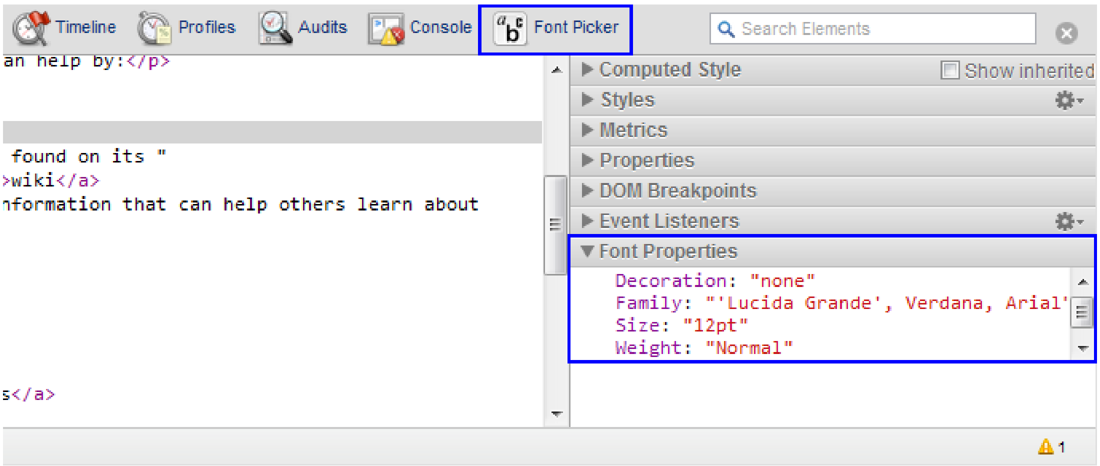
The following code adds a panel contained in Panel.html, represented by FontPicker.png on the Developer Tools toolbar and labeled as *Font Picker*:

```
chrome.devtools.panels.create("Font Picker",
                              "FontPicker.png",
                              "Panel.html"
                              function(panel) { ... });
```

The following code adds a sidebar pane contained in Sidebar.html and titled *Font Properties* to the Elements panel, then sets its height to 8ex:

```
chrome.devtools.panels.elements.createSidebarPane("Font Properties",
    function(sidebar) {
      sidebar.setPage("Sidebar.html");
      sidebar.setHeight("8ex");
    });
```

This screenshot demonstrates the effect the above examples would have on Developer Tools window:



```
an help by:</p>

 found on its "
>wiki</a>
nformation that can help others learn about

s</a>
```

```
Decoration: "none"
Family: "'Lucida Grande', Verdana, Arial'
Size: "12pt"
Weight: "Normal"
```

You can find examples that use this API in **Samples**.

Summary

Types
ElementsPanel
SourcesPanel
ExtensionPanel
ExtensionSidebarPane
Button
Properties
elements
sources
themeName
Methods
create − `chrome.devtools.panels.create(string title, string iconPath, string pagePath, function callback)`
setOpenResourceHandler − `chrome.devtools.panels.setOpenResourceHandler(function callback)`
openResource − `chrome.devtools.panels.openResource(string url, integer lineNumber, function callback)`

Types

ElementsPanel

Represents the Elements panel.

onSelectionChanged

Fired when an object is selected in the panel.

methods
createSidebarPane

```
ElementsPanel.createSidebarPane(string title, function callback)
```

Creates a pane within panel's sidebar.

<table>
<tr><th colspan="3">Parameters</th></tr>
<tr><td>string</td><td>title</td><td>Text that is displayed in sidebar caption.</td></tr>
<tr><td>function</td><td>(optional) callback</td><td>A callback invoked when the sidebar is created.
If you specify the <i>callback</i> parameter, it should be a function that looks like this:
<code>function(ExtensionSidebarPane result) {...};</code><table><tr><td>ExtensionSidebarPane</td><td>result</td><td>An ExtensionSidebarPane object for created sidebar pane.</td></tr></table></td></tr>
</table>

events

addListener

```
onSelectionChanged.addListener(function callback)
```

<table>
<tr><th colspan="3">Parameters</th></tr>
<tr><td>function</td><td>callback</td><td>The <i>callback</i> parameter should be a function that looks like this:
<code>function() {...};</code></td></tr>
</table>

SourcesPanel

Since Chrome 41.

Represents the Sources panel.

onSelectionChanged

Fired when an object is selected in the panel.

methods

createSidebarPane

```
SourcesPanel.createSidebarPane(string title, function callback)
```

Creates a pane within panel's sidebar.

<table>
<tr><th colspan="3">Parameters</th></tr>
<tr><td>string</td><td>title</td><td>Text that is displayed in sidebar caption.</td></tr>
<tr><td>function</td><td>(optional) callback</td><td>A callback invoked when the sidebar is created.
If you specify the <i>callback</i> parameter, it should be a function that looks like this:
<code>function(ExtensionSidebarPane result) {...};</code><table><tr><td>ExtensionSidebarPane</td><td>result</td><td>An ExtensionSidebarPane object for created sidebar pane.</td></tr></table></td></tr>
</table>

events

addListener

```
onSelectionChanged.addListener(function callback)
```

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

ExtensionPanel

Represents a panel created by extension.

onSearch

Fired upon a search action (start of a new search, search result navigation, or search being canceled).

onShown

Fired when the user switches to the panel.

onHidden

Fired when the user switches away from the panel.

methods

createStatusBarButton

```
Button ExtensionPanel.createStatusBarButton(string iconPath, string tooltipText, boolean
disabled)
```

Appends a button to the status bar of the panel.

Parameters		
string	iconPath	Path to the icon of the button. The file should contain a 64x24-pixel image composed of two 32x24 icons. The left icon is used when the button is inactive; the right icon is displayed when the button is pressed.
string	tooltipText	Text shown as a tooltip when user hovers the mouse over the button.
boolean	disabled	Whether the button is disabled.

events

addListener

```
onSearch.addListener(function callback)
```

Parameters				
function	callback	The *callback* parameter should be a function that looks like this: `function(string action, string queryString) {...};`		
		string	action	Type of search action being performed.
		string	(optional) queryString	Query string (only for 'performSearch').

addListener

```
onShown.addListener(function callback)
```

Parameters		
function	callback	The *callback* parameter should be a function that looks like this: `function(global window) {...};` global | window | The JavaScript `window` object of panel's page.

addListener

```
onHidden.addListener(function callback)
```

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

ExtensionSidebarPane

A sidebar created by the extension.

onShown

Fired when the sidebar pane becomes visible as a result of user switching to the panel that hosts it.

onHidden

Fired when the sidebar pane becomes hidden as a result of the user switching away from the panel that hosts the sidebar pane.

methods

setHeight

```
ExtensionSidebarPane.setHeight(string height)
```

Sets the height of the sidebar.

Parameters		
string	height	A CSS-like size specification, such as `'100px'` or `'12ex'`.

setExpression

```
ExtensionSidebarPane.setExpression(string expression, string rootTitle, function callback)
```

Sets an expression that is evaluated within the inspected page. The result is displayed in the sidebar pane.

Parameters		
string	expression	An expression to be evaluated in context of the inspected page. JavaScript objects and DOM nodes are displayed in an expandable tree similar to the console/watch.
string	(optional) rootTitle	An optional title for the root of the expression tree.
function	(optional) callback	A callback invoked after the sidebar pane is updated with the expression evaluation results. If you specify the *callback* parameter, it should be a function that looks like this:

		`function() {...};`

setObject

`ExtensionSidebarPane.setObject(string jsonObject, string rootTitle, function callback)`

Sets a JSON-compliant object to be displayed in the sidebar pane.

Parameters		
string	jsonObject	An object to be displayed in context of the inspected page. Evaluated in the context of the caller (API client).
string	(optional) rootTitle	An optional title for the root of the expression tree.
function	(optional) callback	A callback invoked after the sidebar is updated with the object. If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`

setPage

`ExtensionSidebarPane.setPage(string path)`

Sets an HTML page to be displayed in the sidebar pane.

Parameters		
string	path	Relative path of an extension page to display within the sidebar.

events

addListener

`onShown.addListener(function callback)`

<table>
<tr><th colspan="3">Parameters</th></tr>
<tr><td>function</td><td>callback</td><td>The <i>callback</i> parameter should be a function that looks like this:
<code>function(global window) {...};</code>
<table>
<tr><td>global</td><td>window</td><td>The JavaScript <code>window</code> object of the sidebar page, if one was set with the <code>setPage()</code> method.</td></tr>
</table>
</td></tr>
</table>

addListener

`onHidden.addListener(function callback)`

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

Button

A button created by the extension.

onClicked

Fired when the button is clicked.

methods

update

`Button.update(string iconPath, string tooltipText, boolean disabled)`

Updates the attributes of the button. If some of the arguments are omitted or `null`, the corresponding attributes are not updated.

Parameters		
string	(optional) iconPath	Path to the new icon of the button.
string	(optional) tooltipText	Text shown as a tooltip when user hovers the mouse over the button.
boolean	(optional) disabled	Whether the button is disabled.

events

addListener

`onClicked.addListener(function callback)`

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

Properties

ElementsPanel	`chrome.devtools.panels.elements`	Elements panel.
SourcesPanel	`chrome.devtools.panels.sources`	Since Chrome 38. Sources panel.
string	`chrome.devtools.panels.themeName`	Since Chrome 59. The name of the color theme set in user's DevTools settings. Possible values: `default` (the default) and `dark`.

Methods

create

`chrome.devtools.panels.create(string title, string iconPath, string pagePath, function callback)`

Creates an extension panel.

Parameters		
string	title	Title that is displayed next to the extension icon in the Developer Tools toolbar.

<table>
<tr><td>string</td><td>iconPath</td><td>Path of the panel's icon relative to the extension directory.</td></tr>
<tr><td>string</td><td>pagePath</td><td>Path of the panel's HTML page relative to the extension directory.</td></tr>
<tr><td>function</td><td>(optional) callback</td><td>A function that is called when the panel is created.
If you specify the <i>callback</i> parameter, it should be a function that looks like this:
<code>function(ExtensionPanel panel) {...};</code>
<table>
<tr><td>ExtensionPanel</td><td>panel</td><td>An ExtensionPanel object representing the created panel.</td></tr>
</table></td></tr>
</table>

setOpenResourceHandler

```
chrome.devtools.panels.setOpenResourceHandler(function callback)
```

Specifies the function to be called when the user clicks a resource link in the Developer Tools window. To unset the handler, either call the method with no parameters or pass null as the parameter.

<table>
<tr><th colspan="3">Parameters</th></tr>
<tr><td>function</td><td>(optional) callback</td><td>A function that is called when the user clicks on a valid resource link in Developer Tools window. Note that if the user clicks an invalid URL or an XHR, this function is not called.
If you specify the <i>callback</i> parameter, it should be a function that looks like this:
<code>function(devtools.inspectedWindow.Resource resource) {...};</code>
<table>
<tr><td>devtools.inspectedWindow.Resource</td><td>resource</td><td>A devtools.inspectedWindow.Resource object for the resource that was clicked.</td></tr>
</table></td></tr>
</table>

openResource

```
chrome.devtools.panels.openResource(string url, integer lineNumber, function callback)
```

Since Chrome 38.

Requests DevTools to open a URL in a Developer Tools panel.

Parameters		
string	url	The URL of the resource to open.
integer	lineNumber	Specifies the line number to scroll to when the resource is loaded.
function	(optional) callback	A function that is called when the resource has been successfully loaded. If you specify the *callback* parameter, it should be a function that looks like this: `function() {...};`