

HUMAN HEALTH

ENVIRONMENTAL HEALTH

PerkinElmer

ChemDraw

JavaScript API Reference Guide 1.6



Last Updated: 26/10/2022

Table of Contents

ChemDraw JavaScript API Reference Guide	9
Global Objects	10
ChemDrawAPI	10
activeDocument : Document	10
Example	10
window : Window	10
Example	10
version : string	10
Example	10
openURLInDefaultBrowser(url) : void	11
Parameters	11
Returns	11
Throws Exceptions	11
Example	11
registerURLTriggeredCallback(callbackFunction) : string	11
Parameters	11
Returns	11
Throws Exceptions	11
Example	12
copyToClipboard(text) : void	12
Parameters	12
Returns	12
Throws Exceptions	12
Example	12
Classes	13

Document	13
selection : Selection	13
Example	13
addCDXML(cdxmlText) : void	13
Parameters	13
Returns	13
Throws Exceptions	13
Example	13
getCDXML() : string	13
Parameters	14
Returns	14
Throws Exceptions	14
Example	14
addCDXBase64Encoded(base64EncodedCDXText) : void	14
Parameters	14
Returns	14
Throws Exceptions	14
Example	14
getCDXBase64Encoded() : string	14
Parameters	15
Returns	15
Throws Exceptions	15
Example	15
addSMILES(smilesText) : void	15
Parameters	15
Returns	15
Throws Exceptions	15
Example	15

getSMILES() : string	16
Parameters	16
Returns	16
Throws Exceptions	16
Example	16
addInChI(inChIText) : void	16
Parameters	16
Returns	16
Throws Exceptions	16
Example	16
getInChI() : string	17
Parameters	17
Returns	17
Throws Exceptions	17
Example	17
getInChIKey() : string	17
Parameters	17
Returns	17
Throws Exceptions	18
Example	18
addMolV2000(molV2000Text) : void	18
Parameters	18
Returns	18
Throws Exceptions	18
Example	18
getMolV2000() : string	18
Parameters	19
Returns	19

Throws Exceptions	19
Example	19
addMolV3000(molV3000Text) : void	19
Parameters	19
Returns	19
Throws Exceptions	19
Example	19
getMolV3000() : string	20
Parameters	20
Returns	20
Throws Exceptions	20
Example	20
addMol(molText) : void	20
Parameters	20
Returns	20
Throws Exceptions	20
Example	21
addRXNV2000(rxnV2000Text) : void	21
Parameters	21
Returns	21
Throws Exceptions	21
Example	21
getRXNV2000() : string	21
Parameters	21
Returns	22
Throws Exceptions	22
Example	22
addRXNV3000(rxnV3000Text) : void	22

Parameters	22
Returns	22
Throws Exceptions	22
Example	22
getRXNV3000() : string	23
Parameters	23
Returns	23
Throws Exceptions	23
Example	23
getPNGBase64Encoded([options]) : string	23
Parameters	23
Returns	23
Throws Exceptions	23
Example	23
Window	24
resizeTo(width, height) : void	24
Parameters	24
Returns	24
Throws Exceptions	24
Example	24
setDefaultSize(width, height) : void	24
Parameters	25
Returns	25
Throws Exceptions	25
Example	25
close() : void	25
Parameters	25
Returns	25

Throws Exceptions	25
Example	25
onClose(callback) : void	25
Parameters	26
Returns	26
Example	26
Selection	26
containsPartialStructure() : boolean	26
Parameters	26
Returns	26
Example	26
getCDXML() : string	26
Parameters	26
Returns	27
Example	27
getSVG([options]) : string	27
Parameters	27
Returns	27
Example	27
getInChIKey() : string	27
Parameters	27
Returns	27
Example	28
getSMILES() : string	28
Parameters	28
Returns	28
Example	28
getMolV2000() : string	28

Parameters	28
Returns	28
Throws Exceptions	28
Example	28
getMolV3000() : string	29
Parameters	29
Returns	29
Throws Exceptions	29
Example	29
isEmpty() : boolean	29
Parameters	29
Returns	29
Example	29
onChange(callback) : void	29
Parameters	30
Returns	30
Example	30
JSON Structures	30
ImageOption	30
Properties	30

ChemDraw JavaScript API Reference Guide

This guide provides information about the ChemDraw JavaScript API.

The ChemDraw JavaScript API enables ChemDraw add-ins to access the various features of ChemDraw with JavaScript. With the latest version, the API can support limited access to the active document and the add-in container window.

To access the guide and tutorials for creating ChemDraw add-ins and using ChemDraw JavaScript API in ChemDraw add-ins, go to **Help>Contents>ChemDraw Add-ins**.

Note: *The ChemDraw JavaScript API works only in ChemDraw add-ins. To access the add-in examples that demonstrate the usage of the API, go to [ChemDraw Add-in Examples](#).*

Global Objects

ChemDrawAPI

The singleton global object to access the active document, the add-in container window, etc.

activeDocument : Document

Gets the active document that is currently open in ChemDraw.

Example

```
try {
    var activeDocument = ChemDrawAPI.activeDocument;
}
catch (err) {
    alert(err.message);
}
```

window : Window

Gets the add-in container window that contains the add-in that is currently running.

Example

```
try {
    var addinContainerWindow = ChemDrawAPI.window;
}
catch (err) {
    alert(err.message);
}
```

version : string

Gets the version of the ChemDraw JavaScript API.

Example

```
try {
    document.write(ChemDrawAPI.version);
}
catch (err) {
    alert(err.message);
}
```

Note: The add-in example **Hello World** demonstrates the usage of **ChemDrawAPI.version**. To access it, go to [ChemDraw Add-in Examples](#).

openURLInDefaultBrowser(url) : void

Opens the specified URL in system's default browser. Only web URLs are supported i.e. The URL must start with an "http" or "https" otherwise an exception will be thrown.

Parameters

Name	Type	Description
url	string	The URL to open in system's default browser

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.openURLInDefaultBrowser (  
        "http://www.perkinelmer.com/category/chemdraw");  
}  
catch (err) {  
    alert(err.message);  
}
```

Note: The add-in example **Open URL** demonstrates the usage of **ChemDrawAPI.openURLInDefaultBrowser**. To access it, go to [ChemDraw Add-in Examples](#).

registerURLTriggeredCallback(callbackFunction) : string

Registers a callback function with ChemDraw that will be called when ChemDraw is invoked by the URL handler. This API takes a callback function and returns a callback key that uniquely identifies this callback.

Parameters

Name	Type	Description
callbackFunction	Function	The callback function that will be invoked

Returns

Return Type	Description
string	A callback key that uniquely identifies this callback

Throws Exceptions

Yes

Example

```
try {
    const callbackKey = ChemDrawAPI.registerURLTriggeredCallback((arg) => {
        alert(arg);
    });
}
catch (err) {
    alert(err.message);
}
```

Note: The add-in example **ChemDrawOAuthExample** demonstrates the usage of **ChemDrawAPI.registerURLTriggeredCallback**. To access it, go to [ChemDraw Add-in Examples](#). More information on the usage of this API can be found in the ChemDraw User Guide.

copyToClipboard(text) : void

Copies the specified text to the clipboard.

Parameters

Name	Type	Description
text	string	The text to be copied

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.copyToClipboard("Hello World");
}
catch (err) {
    alert(err.message);
}
```

Classes

Document

Provides an API to add and get data in a ChemDraw document.

Note: The add-in example **Document Data Importer and Exporter** demonstrates the usage of the API for adding and getting data in the active document. To access it, go to [ChemDraw Add-in Examples](#).

selection : Selection

Gets the selection in the document.

Example

```
var selection = ChemDrawAPI.activeDocument.selection;
if (!selection.isEmpty()) {
    // Do something
}
```

addCDXML(cdxmlText) : void

Adds a CDXML document to the active document. The active document honors the style and coordinates of the supplied CDXML document.

Parameters

Name	Type	Description
cdxmlText	string	The CDXML document to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.activeDocument.addCDXML(cdxmlText);
}
catch (err) {
    alert(err.message);
}
```

getCDXML() : string

Gets the contents of the active document as CDXML.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a CDXML string

Throws Exceptions

Yes

Example

```
try {  
    var cdxmlText = ChemDrawAPI.activeDocument.getCDXML();  
}  
catch (err) {  
    alert(err.message);  
}
```

addCDXBase64Encoded(base64EncodedCDXText) : void

Adds a CDX document, encoded as [Base64](#), to the active document. The active document honors the style and coordinates of the supplied CDX document.

Parameters

Name	Type	Description
base64EncodedCDXText	string	The CDX document to add to the active document. The string must be encoded as Base64

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.activeDocument.addCDXBase64Encoded(base64EncodedCDXText);  
}  
catch (err) {  
    alert(err.message);  
}
```

getCDXBase64Encoded() : string

Gets the contents of the active document as [Base64](#) encoded CDX.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a Base64 encoded string

Throws Exceptions

Yes

Example

```
try {
    var base64EncodedCDXText = ChemDrawAPI.activeDocument.getCDXBase64Encoded();
}
catch (err) {
    alert(err.message);
}
```

addSMILES(smilesText) : void

Adds a structure, encoded in a [SMILES](#) string, to the active document. The document retains its original styling. The structure will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
smilesText	string	The SMILES encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.activeDocument.addSMILES(smilesText);
}
catch (err) {
    alert(err.message);
}
```

getSMILES() : string

Gets the contents of the active document as a [SMILES](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [SMILES](#) string will be returned in dot-separated notation.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a SMILES string

Throws Exceptions

Yes

Example

```
try {  
    var smilesText = ChemDrawAPI.activeDocument.getSMILES();  
}  
catch (err) {  
    alert(err.message);  
}
```

addInChI(inChIText) : void

Adds a structure, encoded in an [InChI](#) string, to the active document. The document retains its original styling. The structure will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
inChIText	string	The InChI encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.activeDocument.addInChI(inChIText);  
}
```



```
catch (err) {  
    alert(err.message);  
}
```

getInChI() : string

Gets the contents of the active document as an [InChI](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [InChI](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as an InChI string

Throws Exceptions

Yes

Example

```
try {  
    var inChIText = ChemDrawAPI.activeDocument.getInChI();  
}  
catch (err) {  
    alert(err.message);  
}
```

getInChIKey() : string

Gets the contents of the active document as an [InChIKey](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [InChIKey](#) string will return them in its native format.

Note: The InChIKey is a one-way format. A key can be generated from a structure, but a structure cannot be generated from a key. Thus there is no Add method for InChIKey.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as an InChIKey string

Throws Exceptions

Yes

Example

```
try {
    var inChIKeyText = ChemDrawAPI.activeDocument.getInChIKey();
}
catch (err) {
    alert(err.message);
}
```

addMolV2000(molV2000Text) : void

Adds a structure, encoded in a [MolV2000](#) string to the active document. The document retains its original styling. If present, 2D coordinates will be honored for the structure, otherwise the structure will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
molV2000Text	string	The MolV2000 encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.activeDocument.addMolV2000(molV2000Text);
}
catch (err) {
    alert(err.message);
}
```

getMolV2000() : string

Gets the contents of the active document as a [MolV2000](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [MolV2000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a MolV2000 string

Throws Exceptions

Yes

Example

```
try {
    var molV2000Text = ChemDrawAPI.activeDocument.getMolV2000();
}
catch (err) {
    alert(err.message);
}
```

addMolV3000(molV3000Text) : void

Adds a structure, encoded in a [MolV3000](#) string to the active document. The document retains its original styling. If present, 2D coordinates will be honored for the structure, otherwise they will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
molV3000Text	string	The MolV3000 encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.activeDocument.addMolV3000(molV3000Text);
}
catch (err) {
    alert(err.message);
}
```

getMolV3000() : string

Gets the contents of the active document as a [MolV3000](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [MolV3000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a MolV3000 string

Throws Exceptions

Yes

Example

```
try {  
    var molV3000Text = ChemDrawAPI.activeDocument.getMolV3000();  
}  
catch (err) {  
    alert(err.message);  
}
```

addMol(molText) : void

This is a convenience method that will automatically detect whether the string is in [MolV2000](#) or [MolV3000](#) format, and add the structure to the active document. The document retains its original styling. If present, 2D coordinates will be honored for the structure, otherwise the structure will be placed at the center of the currently visible view of the document. If the supplied string is in neither [MolV2000](#) or [MolV3000](#) format, an exception will be thrown.

Note: There is no equivalent *getMol()* method. Call the *getMolV2000()* or *getMolV3000()* methods instead.

Parameters

Name	Type	Description
molText	string	The MolV2000 or MolV3000 encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.activeDocument.addMol (molText);  
}  
catch (err) {  
    alert (err.message);  
}
```

addRXNV2000(rxnV2000Text) : void

Adds a reaction, encoded in an [RXNV2000](#) string, to the active document. The document retains its original styling. If present, 2D coordinates will be honored for the reaction, otherwise the reaction will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
rxnV2000Text	string	The RXNV2000 encoded structure to add to the active document

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.activeDocument.addRXNV2000 (rxnV2000Text);  
}  
catch (err) {  
    alert (err.message);  
}
```

getRXNV2000() : string

Gets the contents of the active document as an [RXNV2000](#) string. If there is a mixture of reactions and non-reactions (e.g. clip-art, text etc), only the reactions will be returned. If there are multiple reactions, the [RXNV2000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a RXNV2000 string

Throws Exceptions

Yes

Example

```
try {
    var rxnV2000Text = ChemDrawAPI.activeDocument.getRXNV2000();
}
catch (err) {
    alert(err.message);
}
```

addRXNV3000(rxnV3000Text) : void

Adds a reaction, encoded in an [RXNV3000](#) string, to the active document. The document retains its original styling. If present, 2D coordinates will be honored for the reaction, otherwise the reaction will be placed at the center of the currently visible view of the document.

Parameters

Name	Type	Description
rxnV3000Text	string	The RXNV3000 encoded structure to add to the active document.

Returns

None

Throws Exceptions

Yes

Example

```
try {
    ChemDrawAPI.activeDocument.addRXNV3000(rxnV3000Text);
}
catch (err) {
    alert(err.message);
}
```

getRXNV3000() : string

Gets the contents of the active document as a [RXNV3000](#) string. If there is a mixture of reactions and non-reactions (e.g. clip-art, text etc), only the reactions will be returned. If there are multiple reactions, the [RXNV3000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a RXNV3000 string

Throws Exceptions

Yes

Example

```
try {  
    var rxnV3000Text = ChemDrawAPI.activeDocument.getRXNV3000();  
}  
catch (err) {  
    alert(err.message);  
}
```

getPNGBase64Encoded([options]) : string

Gets the contents of the active document as a PNG image with the specified options. This can be used to generate a PNG image as a preview of the document.

Parameters

Name	Type	Description
options	Image options	Optional Options are for getting the PNG image.

Returns

Return Type	Description
string	The contents of the active document as a Base64 encoded PNG string

Throws Exceptions

Yes

Example

```
try {  
    var pngBase64EncodedText = ChemDrawAPI.activeDocument.getPNGBase64Encoded({  
        transparent: false,  
    });  
}
```

```
        scalePercent: 100,  
        borderSizeInPixels: 100  
    });  
}  
catch (err) {  
    alert(err.message);  
}
```

Window

resizeTo(width, height) : void

Resizes the window using the specified width and height.

This API is deprecated.

Parameters

Name	Type	Description
width	integer	The new width of the window. A value greater than zero is expected, otherwise an exception will be thrown
height	integer	The new height of the window. A value greater than zero is expected, otherwise an exception will be thrown

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.window.resizeTo(width, height);  
}  
catch (err) {  
    alert(err.message);  
}
```

setDefaultSize(width, height) : void

Set default size for the window using the specified width and height.

Parameters

Name	Type	Description
width	integer	The new width of the window. A value greater than zero is expected, otherwise an exception will be thrown
height	integer	The new height of the window. A value greater than zero is expected, otherwise an exception will be thrown

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.window.setDefaultSize(width, height);  
}  
catch (err) {  
    alert(err.message);  
}
```

close() : void

Closes the add-in container window.

Parameters

None

Returns

None

Throws Exceptions

Yes

Example

```
try {  
    ChemDrawAPI.window.close();  
}  
catch (err) {  
    alert(err.message);  
}
```

onClose(callback) : void

Registers a callback function which will be called when the add-in container window is about to close.

Parameters

Name	Type	Description
callback	function	(Optional) The callback function which will be called when the add-in container window is about to close.

Returns

None

Example

```
ChemDrawAPI.window.onClose(function () {
    // Do clean up here when the add-in container window is about to close
});
```

Selection

Provides API to get the selected structures in a ChemDraw document.

Note: The add-in example Selection Monitor demonstrates the usage of the API for getting the preview image of the selected structures in the active document. To access it, go to [ChemDraw Add-in Examples](#).

containsPartialStructure() : boolean

Determines whether the selection contains partially selected structure.

Parameters

None

Returns

Return Type	Description
boolean	true if part of a single structure or grouped structures is selected, false otherwise

Example

```
if ChemDrawAPI.activeDocument.selection.containsPartialStructure() {
    // Do something
}
```

getCDXML() : string

Gets the contents of the selection as a CDXML document.

Parameters

None

Returns

Return Type	Description
string	The contents of the selection as a CDXML document

Example

```
var cdxmlText = ChemDrawAPI.activeDocument.selection.getCDXML();  
  
// Do something with the CDXML text here
```

getSVG([options]) : string

Gets the contents of the selection as an SVG image with the specified options. This can be used to generate an SVG image as a preview of the selection.

Parameters

Name	Type	Description
options	Image options	Optional Options for getting the SVG image

Returns

Return Type	Description
string	The contents of the selection as an SVG image

Example

```
var svg = ChemDrawAPI.activeDocument.selection.getSVG({  
  transparent: false,  
  scalePercent: 100,  
  borderSizeInPixels: 20  
});  
  
// Do something with the svg image here
```

getInChIKey() : string

Gets the contents of the selection as an [InChIKey](#) string.

Parameters

None

Returns

Return Type	Description
string	The contents of the selection as an InChIKey string

Example

```
var inChIKey = ChemDrawAPI.activeDocument.selection.getInChIKey();  
  
// Do something with the inChIKey here
```

getSMILES() : string

Gets the contents of the selection as a [SMILES](#) string.

Parameters

None

Returns

Return Type	Description
string	The contents of the selection as a SMILES string

Example

```
var smilesText = ChemDrawAPI.activeDocument.selection.getSMILES();  
  
// Do something with the SMILES here
```

getMolV2000() : string

Gets the contents of the active document as a [MolV2000](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [MolV2000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a MolV2000 string

Throws Exceptions

Yes

Example

```
try {  
    var molV2000Text = ChemDrawAPI.activeDocument.selection.getMolV2000();  
}  
catch (err) {  
    alert(err.message);  
}
```

getMolV3000() : string

Gets the contents of the active document as a [MolV3000](#) string. If there is a mixture of structures and non-structures (e.g. clip-art, text etc), only the structures will be returned. If there are multiple structures, the [MolV3000](#) string will return them in its native format.

Parameters

None

Returns

Return Type	Description
string	The contents of the active document as a MolV3000 string

Throws Exceptions

Yes

Example

```
try {  
    var molV3000Text = ChemDrawAPI.activeDocument.selection.getMolV3000();  
}  
catch (err) {  
    alert(err.message);  
}
```

isEmpty() : boolean

Determines whether any object has been selected in the document.

Parameters

None

Returns

Return Type	Description
boolean	true if nothing has been selected, false otherwise

Example

```
if (!ChemDrawAPI.activeDocument.selection.isEmpty()) {  
    // Do something  
}
```

onChange(callback) : void

Registers a callback function which will be called when the selection is changed in the document.

Parameters

Name	Type	Description
callback	function	The callback function which will be called when the selection is changed

Returns

None

Example

```
// Set the callback (or handler) for the selection change event
ChemDrawAPI.activeDocument.selection.onChange(function () {
    if (!ChemDrawAPI.activeDocument.selection.isEmpty()) {
        cdxmlText = ChemDrawAPI.activeDocument.selection.getCDXML();
        svg = ChemDrawAPI.activeDocument.selection.getSVG();

        // Do something with the cdxmlText and the svg image here
    }
});
```

JSON Structures

ImageOption

Options for getting an image. This can be used to specify the properties of the preview image for generating a preview of a document or a selection.

Properties

Name	Type	Default Value	Description
transparent	boolean	true	(Optional) If true, the image will have a transparent background. If false, the background is generated along with the document contents
scalePercent	integer	100	(Optional) The scaling applied to the image. 100% is the actual size. A value greater than zero is expected, otherwise an exception will be thrown
borderSizeInPixels	integer	0	(Optional) The border (gap) to be placed around the image in pixels. A value of zero or greater is expected,

Name	Type	Default Value	Description
			otherwise an exception will be thrown