HUMAN HEALTH

ENVIRONMENTAL HEALTH

ChemDraw JS 20.0

API Reference Guide

PerkinElmer®

For the Better

Table of Contents

Global Objects

perkinelmer.ChemdrawWebManager

The singleton global object to manage ChemDraw JS instances.

attach(options) : void

Creates a new ChemDraw JS tool and attaches it to the specified DOM node. The attach function may be used to attach several instances of ChemDraw to the DOM and may be called at any time. The ChemDraw instance cannot be used immediately once this functions returns. A callback function can be specified in the input JSON object. Then it can be called after the instance becomes available. Please refer to the Developers Guide for details.

Parameters

Name	Type	Description
options	AttachOptions	which is object contains some parameters to configure ChemDraw JS

Returns

void

Example

```
var chemdrawjs;
function chemDrawAttached(chemdrawweb) {
  // Implement your own logic here, call API
  chemdrawjs = chemdrawweb;
}

perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: chemDrawAttached
});
```

perkinelmer.DataFormats

The data formarts used to specify the type of data as used by the *dataFormat* field in the attach options.

Name	Type	Description
B64CDX	string	Base64 encoded CDX format data

Name	Type	Description
CDXML	string	CDXML format data
MOLFILE	string	MolFile format data
SMILES	string	SMILES format data

Classes

ChemDrawDirect

Provides APIs to manage the ChemDraw document, canvas and drawings on the canvas.

addProduct(cdxml) : number

Appends a new product to the reaction present in the current drawing. If there are products present, the new product will be added to the right of the right-most product. If no reaction is present, a new arrow will be added to the center of the canvas along with the product.

Parameters

Name	Type	Description
cdxml	string	the CDXML string

Returns

number the ID of the new product

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});
chemdrawjs.addProduct("CDXML content...");
```

addReactant(cdxml) : number

Appends a new reactant to the reaction present in the current drawing. If there are reactants present, the new reactant is added to the left of the left-most reactant. If no reaction is present, a new arrow is added to the center of the canvas along with the reactant.

Parameters

Name	Type	Description
cdxml	string	the CDXML string

Returns

number the ID of the new reactant

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.addReactant("CDXML content...");
```

addReagent(cdxml) : number

Appends a new reagent to the reaction present in the current drawing. The reagent is added to the top of the reaction arrow and above the top-most reagent if there are any already. If no reaction is present, a new arrow is added to the center of the canvas, along with the reagent.

Parameters

Name	Type	Description
cdxml	string	the CDXML string

Returns

number the ID of the new reagent

Example

```
var chemdrawjs;
```

```
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.addReagent("CDXML content...");
```

clear() : void

Clears all drawings on the ChemDraw canvas. This operation is not undoable. And ChemDraw will not prompt the user to confirm the operation.

Parameters

None

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.clear();
```

convertToCDXML(content, mimeType, callback) : void

Converts structure string to CDXML format.

Parameters

Name	Type	Description
content	string	the structure string
mimeType	string	the string format type, ex: chemical/x-cdx, chemical/x-mdl-molfile, chemical/x-helm
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.convertToCDXML("Structure content...", "Structure mine type...", function (cdxml,
error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

findReactions(callback) : void

Returns reactions information of the current drawing. This call is asynchronous and a callback function must be provided that is informed when the results are available.

Parameters

Name	Type	Description
callback	GetTextContentCallback	function when finish loading

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.findReactions(function (reactions, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

fitToContainer() : void

Resizes ChemDraw instance to fit to the container.

Parameters

None

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.fitToContainer();
```

getAllFragmentsInfo([options]) : Array<FragmentInfo>

Gets information of all fragments on the canvas which contains CDXML string and preview PNG picture.

Parameters

Name	Type	Description
options	GetFragmentsInfoOptions	**Optional** the options for getting fragments information

Returns

Array<FragmentInfo> an array of fragments information which contains CDXML string and preview PNG picture

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var fragmentsInfo = chemdrawjs.getAllFragmentsInfo();
var cdxml = fragmentsInfo.cdxml;
var image = fragmentsInfo.image;
// ...
```

getB64CDX() : string

Returns a CDX base64 string representing the current drawing.

Parameters

None

Returns

string a CDX base64 string representing the current drawing

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var b64cdx = chemdrawjs.getB64CDX();
```

getCDXML() : string

Returns a CDXML string representing the current drawing.

Parameters

None

Returns

string a CDXML string representing the current drawing

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
```

```
    chemdrawjs = chemdrawweb;
  }
});

var cdxml = chemdrawjs.getCDXML();
```

getCanvasSize() : Size

Gets ChemDraw JS's current canvas size.

Parameters

None

Returns

Size ChemDraw JS's current canvas size

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var size = chemdrawjs.getCanvasSize();
var width = size.width;
var height = size.height;
```

getDocumentName() : string

Gets the underlying document name which is saved in the CDXML.

Parameters

None

Returns

string the underlying document name which is saved in the CDXML

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var name = chemdrawjs.getDocumentName();
```

getErrors() : Array<string>

Returns an array of strings describing the chemical errors that are present in the current drawings.

Parameters

None

Returns

Array<string> an array of strings describing the chemical errors that are present in the current drawings

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var errors = chemdrawjs.getErrors();
```

getFragmentsInfo([options]) : Array<FragmentInfo>

Gets fragments information which contains CDXML string and preview PNG picture.

Parameters

Name	Type	Description
options	GetFragmentsInfoOptions	**Optional** the options for getting fragments information

Returns

Array<FragmentInfo> an array of fragments information which contains CDXML string and preview PNG picture

deprecated

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var fragmentsInfo = chemdrawjs.getFragmentsInfo();
var cdxml = fragmentsInfo.cdxml;
var image = fragmentsInfo.image;
// ...
```

getImgUrl([options]) : string

Returns a URL string which points to a PNG representation of the current drawing. The image contents are encoded in the URL so that it can be used offline.

Note: *To get an image without border, you should use getImgUrl({'borderSize': 0}).*

Parameters

Name	Type	Description
options	ImageOptions	**Optional** options for getting PNG information

Returns

string a URL string which points to a PNG representation of the current drawing

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var imageUrl = chemdrawjs.getImgUrl();
```

getInChI(callback) : void

Translates the current drawing into an InChI representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});
```

```
chemdrawjs.getInChI(function (inchI, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getInChIKey(callback) : void

Translates the current drawing into an InChIKey representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getInChIKey(function (inchIKey, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getMOL(callback) : void

Translates the current drawing into a MOL v2000 representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getMOL(function (mol, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getMOLV3000(callback) : void

Translates the current drawing into a MOL v3000 representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getMOLV3000(function (molv3000, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getMolecularWeight(content, mimeType, callback) : void

Gets molecular weight of the structure.

Parameters

Name	Type	Description
content	string	the structure string
mimeType	string	the string format type, ex: chemical/x-cdx, chemical/x-mdl-molfile, chemical/x-helm
callback	GetNumberCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getMolecularWeight("Structure content...", "Structure mine type...", function
(weight, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getProperties(callback) : void

Returns properties of the current drawing. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
```

```
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getProperties(function (props, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getRXN(callback) : void

Translates the current drawing into a RXN v2000 representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});
```

```
chemdrawjs.getRXN(function (rxn, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getRXNV3000(callback) : void

Translates the current drawing into a RXN v3000 representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getRXNV3000(function (rxnv3000, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getSMILES(callback) : void

Translates the current drawing into a SMILES representation. This call is asynchronous. A callback function can to be provided and it will be invoked after the result is generated.

Parameters

Name	Type	Description
callback	GetTextContentCallback	the callback function to be called after the result is generated

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.getSMILES(function (smiles, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

getSVG([options[, callbackOptions]]) : string

Translates the current drawing into a SVG representation.

Parameters

Name	Type	Description
options	ImageOptions	**Optional** image options for getting SVG information, also could be

Name	Type	Description
	GetTextContentCallback	a callback function for legacy support.
callbackOptions	ImageOptions	**Optional** image options for getting SVG information. Only effective when first parameter is a callback function.

Returns

string

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var svg = chemdrawjs.getSVG();
```

getSelectedFragmentsInfo([options]) : Array<FragmentInfo>

Gets information of the selected fragments on the canvas which contains CDXML string and preview PNG picture.

Parameters

Name	Type	Description
options	GetFragmentsInfoOptions	Optional the options for getting fragments information

Returns

Array<FragmentInfo> an array information of the selected fragments on the canvas which contains CDXML string and preview PNG picture

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
```

```
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var fragmentsInfo = chemdrawjs.getSelectedFragmentsInfo();
var cdxml = fragmentsInfo.cdxml;
var image = fragmentsInfo.image;
// ...
```

getVersion() : string

Returns the current version of ChemDraw as a string.

Parameters

None

Returns

string the current version of ChemDraw

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var version = chemdrawjs.getVersion();
```

isBlankStructure() : boolean

Returns true if nothing has been drawn in ChemDraw and false if something has been drawn.

Parameters

None

Returns

boolean true if nothing has been drawn in ChemDraw and false if something has been drawn

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var isBlank = chemdrawjs.isBlankStructure();
```

isSaved() : boolean

Determines if there is any unsaved change.

Parameters

None

Returns

boolean true if there is any unsaved change, otherwise false

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

var isSaved = chemdrawjs.isSaved();
```

labelFragments(ids) : void

Auto-number the fragments of the reaction with Roman numeral labels. Any existing labels will be replaced with the new ones.

Parameters

Name	Type	Description
ids	Array<number>	the array of the fragments to be labelled

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.labelFragments([1, 2, 3]);
```

loadB64CDX(b64CDX, callback) : void

Reads a BASE64 encoded CDX string and loads its contents into ChemDraw. This call is asynchronous. A callback function can to be provided and it will be invoked after the loading is done.

Parameters

Name	Type	Description
b64CDX	string	the CDX base64 encoded string
callback	LoadDocumentCallback	function when finish loading

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadB64CDX("CDX base64 content...");
```

loadCDXML(cdxml) : void

Reads a CDXML string and loads its contents into ChemDraw.

Parameters

Name	Type	Description
cdxml	string	the CDXML string

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadCDXML("CDXML content...");
```

loadConfig(config) : void

Applies a new configuration into ChemDraw.

Parameters

Name	Type	Description
config	string ¦ Configuration	the configuration json object or json string

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadConfig("Configuration JSON string");
// chemdrawjs.loadConfig({}); or Configuration JSON Object
```

loadConfigFromUrl(url, callback) : void

Applies a new configuration from remote url into ChemDraw.

Parameters

Name	Type	Description
url	string	the remote configuration file url
callback	ConfigLoadCallback	when finished, this function will be called

Returns

void

Example

```
var chemdrawjs;
```

```
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadConfigFromUrl("remote configuration file url", function (error) {
  // if error is undefined configuration is done
  // if error isn't undefined, it shows the error message during the loading progress
});
```

loadMOL(mol, callback) : void

Reads a MOL string and loads its contents into ChemDraw. This call is asynchronous. A callback function can to be provided and it will be invoked after the loading is done.

Parameters

Name	Type	Description
mol	string	the MOL string
callback	LoadDocumentCallback	function when finish loading

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadMOL("MOL content...", function (cdxml, error) {
  if (!error) {
```

```
    // deal with result
  } else {
    // deal with error
  }
});
```

loadSMILES(smiles, callback) : void

Reads a SMILES string and loads its contents into ChemDraw. This call is asynchronous. A callback function can to be provided and it will be invoked after the loading is done.

Parameters

Name	Type	Description
smiles	string	the SMILES string
callback	LoadDocumentCallback	function when finish loading

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.loadSMILES("SMILES content...", function (cdxml, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

markAsSaved() : void

Sets the dirty flag to indicate the canvas is saved and clean.

Parameters

None

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.markAsSaved();
```

nameToStructure(name, callback) : void

Reads a structure name and returns a CDXML string representing the structure.

Parameters

Name	Type	Description
name	string	the name of the structure to be converted
callback	GetTextContentCallback	the callback function to be called after converting is done

Returns

void

Example

```
var chemdrawjs;
```

```
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.nameToStructure("structureName", function (cdxml, error) {
  if (!error) {
    // deal with result
  } else {
    // deal with error
  }
});
```

selection : ChemDrawDirectSelection

ChemDrawDirectSelection Object, which provides APIs to manage the selected objects on the canvas.

setCanvasSize(size[, options]) : void

Sets ChemDraw JS canvas size.

Parameters

Name	Type	Description
size	Size	the new canvas size
options	SetCanvasSizeOptions	**Optional** the options to configure the canvas size

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
```

```
  }
});

var size = {
  width: 100,
  height: 100
};

var options = {
  doNotCropDrawings: false
};

chemdrawjs.setCanvasSize(size, options);
```

setContentChangedHandler(handler) : void

Sets a callback function which will be invoked when user makes any change on the canvas.

Parameters

Name	Type	Description
handler	ContentChangedHandler	the callback function

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.setContentChangedHandler(function (event) {
  // Implement your own logic here
```

```
});
```

setDocumentName(name) : void

Sets the document name to the resulting CDXML.

Parameters

Name	Type	Description
name	string	the document name to set

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.setDocumentName("document name");
```

setViewOnly(viewOnly) : void

Sets whether the user is able to interact with ChemDraw.

Parameters

Name	Type	Description
viewOnly	boolean	indicator the ChemDraw JS instance is view only or not

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.setViewOnly();
```

structureToName(cdxml, callback) : void

Reads a CDXML string and returns the name of the structure.

Parameters

Name	Type	Description
cdxml	string	the CDXML string to be converted
callback	GetTextContentCallback	the callback function to be called after converting is done

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.structureToName("CDXML content...", function (name, error) {
  if (!error) {
    // deal with result
```

```
  } else {
    // deal with error
  }
});
```

ChemDrawDirectSelection

Provides APIs to manage the selected objects on the canvas.

containsPartialStructure() : boolean

Returns true if part of a single molecule or grouped molecules is selected.

Parameters

None

Returns

boolean

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty() && chemdrawjs.selection.containsPartialStructure()) {
    // do something like sub-structure search
  }
});
```

getCDXML() : string

Returns a CDXML string representing current selected drawing, throw an exception when nothing is selected.

Parameters

None

Returns

string

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
    var cdxml = chemdrawjs.selection.getCDXML();
  }
});
```

getImageUrl([options]) : string

Returns PNG picture encoded in Base64 representing current selected drawing, throw an exception when nothing is selected.

Parameters

Name	Type	Description
options	ImageOption	**Optional** options for getting PNG information

Returns

string

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});
```

```
chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
    var imageUrl = chemdrawjs.selection.getImageUrl();
  }
});
```

getSVG([options]) : string

Return SVG string representing current selected drawing, throw an exception when nothing is selected.

Parameters

Name	Type	Description
options	ImageOption	**Optional** options for getting PNG information

Returns

string

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
    var svg = chemdrawjs.selection.getSVG();
  }
});
```

isEmpty() : boolean

Returns true if nothing has been selected in ChemDraw and false if something has been selected.

Parameters

None

Returns

boolean

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
    // do something when selected drawings
  }
});
```

numberOfObjects() : number

Returns the count of fully or partially selected objects or grouped objects, including molecules, arrows and etc.

Parameters

None

Returns

number

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
```

```
    var selectedObjectsCount = chemdrawjs.selection.numberOfObjects();
  }
});
```

numberOfStructures() : number

Returns the count of fully or partially selected molecules/grouped molecules.

Parameters

None

Returns

number

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  if (!chemdrawjs.selection.isEmpty()) {
    var selectedObjectsCount = chemdrawjs.selection.numberOfObjects();
  }
});
```

onChange(handler) : void

Sets a callback function which will be invoked when selection changed on the canvas.

Parameters

Name	Type	Description
handler	SelectionChangedHandler	the callback function

Returns

void

Example

```
var chemdrawjs;
perkinelmer.ChemdrawwebManager.attach({
  id: "chemdrawjsContainer",
  callback: function (chemdrawweb) {
    chemdrawjs = chemdrawweb;
  }
});

chemdrawjs.selection.onChange(function () {
  // Implement your own logic here
});
```

JSON Structures

ApplicationProperties

Application properties

Properties

Name	Type	Description
StyleSheet	string	The name of the default document stylesheet
chemservice	string	The end point of the ChemDraw web service

AttachOptions

The options for creating and attaching a new ChemDraw JS instance

Properties

Name	Type	Description
callback	AttachCallback	**Optional** The callback function to be invoked after the new ChemDraw JS instance is attached
config	Configuration	**Optional** The configuration JSON object
configUrl	string	**Optional** The URL to a remote configuration file
data	string	**Optional** The data to open a document while initializing ChemDraw JS
dataFormat	string	**Optional** The format of the data. The value should be one of the constants defined in DataFormatEnum.
element	HTMLElement	**Optional** The object of the HTML element to be attached to
errorCallback	ErrorCallback	**Optional** The callback function when ChemDraw JS is failed to load or attach
id	string	**Optional** The ID of the HTML element to be attached to
license	string	**Optional** The license file content to initialize ChemDraw JS
licenseUrl	string	**Optional** The license file URL to initialize ChemDraw JS
preservePageInfo	boolean	**Optional** Determines the view mode of ChemDraw JS
viewonly	boolean	**Optional** Determines the initial state of ChemDraw JS (view only or editable)

Configuration

The ChemDraw JS configuration

Properties

Name	Type	Description
appProperties	ApplicationProperties	Other application properties
features	Features	Feature toggles
layout	ToolsLayout	Toolbar layout

ContentChangedEvent

The event for content change handler

Properties

Name	Type	Description
action	string	The action name

Features

Features

Properties

Name	Type	Description
disabled	Array<string>	The list of disabled features
enabled	Array<string>	The list of enabled features

FragmentInfo

Structure information on the canvas

Properties

Name	Type	Description
cdxml	string	The structure CDXML string

Name	Type	Description
height	number	The structure height
id	number	The fragment id
image	string	The structure preview PNG picture encoded in BASE64
width	number	The structure width
x	number	The structure location x
y	number	The structure location y

GetFragmentsInfoOptions

Options for getting fragments information

Properties

Name	Type	Description
borderSize	number	**Optional** The size of the border
transparentBackground	boolean	**Optional** Determines if the background should be transparent or not

ImageOptions

Options for getting the image information.

Properties

Name	Type	Description
borderSize	number	**Optional** The size of the border
embedCDXML	boolean	**Optional** Determines CDXML string is embedded in SVG string. Only works for SVG rendering
transparentBackground	boolean	**Optional** Determines if the background should be transparent or not

SetCanvasSizeOptions

The option for setting canvas size

Properties

Name	Type	Description
doNotCropDrawings	boolean	**Optional** Determines whether crop the drawing when change the canvas size.

Size

The size object contains width and height

Properties

Name	Type	Description
height	number	height in pixel
width	number	width in pixel

Tool

Tool

Properties

Name	Type	Description
order	Array<string ¦ ToolGroup>	Configure tool item order, it could be single tool or group tool

ToolGroup

Group tool instance

Properties

Name	Type	Description
collapse	string	Collapse mode："auto", "always", "never"
columns	string	The number of columns of the expanded tool group panel
group	string	The name of the tool group
order	Array<string>	The order of the sub tools
writingMode	string	**Optional** The layout mode of the expanded tool group panel: "horizontal", "vertical"

ToolsLayout

Tools layout

Properties

Name	Type	Description
orientation	string	Toolbar orientation. "Horizontal", "Vertical"
tools	Array<Tool>	The list of tools or tool groups

Callbacks

AttachCallback

A callback function which is called after ChemDraw JS is attached

Parameters

Name	Type	Description
cdd	ChemDrawDirect	the new instance of ChemDraw JS

ConfigLoadCallback

A callback function which is called after a new configuration is loaded

Parameters

Name	Type	Description
error	string	**Optional** the error message during the loading process

ContentChangedHandler

The callback function which will return the event

Parameters

Name	Type	Description
event	ContentChangedEvent	the action description when does changes on canvas

ErrorCallback

Callback function indicating error happens.

Parameters

Name	Type	Description
error	Error	The error detail

GetNumberCallback

The callback function which will return the number and error

Parameters

Name	Type	Description
value	number	the result number
error	string	the error message if any exception occurs

GetTextContentCallback

The callback function which will return the string and error

Parameters

Name	Type	Description
content	string	the result string
error	string	the error message if some exceptions occur

LoadDocumentCallback

The callback function which will return the CDXML string and error

Parameters

Name	Type	Description
cdxmlStr	string	the CDXML string
error	string	the error message if some exceptions occur

SelectionChangedHandler

The callback function of selection changed event.

Parameters

None