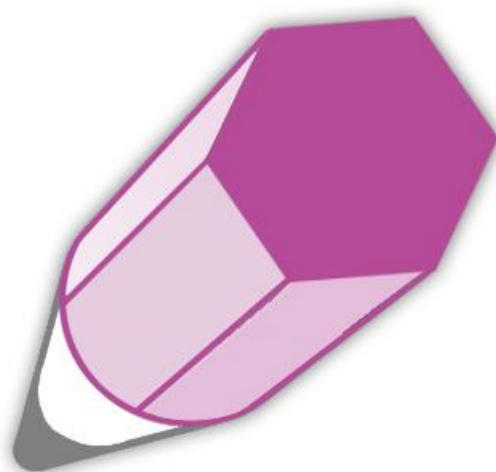




**IC** *EDIT*



## **Technical Documentation**

Structure and Reaction Editor

JavaScript Version 3.0.1.5

Spring 2014

# Table of Contents

<b>1</b>	<b>Preface.....</b>	<b>2</b>
<b>2</b>	<b>Configuration of ICEDIT.....</b>	<b>3</b>
2.1	Parameter description .....	4
<b>3</b>	<b>Changing the Layout of ICEDIT .....</b>	<b>5</b>
3.1	Configuration of ICEDIT atom button bar.....	5
3.2	Width and height of the drawing area .....	7
3.2.1	Drawing area width .....	7
3.2.2	Drawing area height.....	8
3.3	Button images .....	8
3.4	Button background .....	8
3.5	Button bar background color .....	9
3.5.1	Background of the main button bar.....	9
3.5.2	Background of the sub button bar.....	10
3.6	Add/Remove shadow effect .....	10
3.7	Select Text Font .....	11
3.8	Example .....	11
<b>4</b>	<b>Initialization.....</b>	<b>13</b>
<b>5</b>	<b>Programming Interface .....</b>	<b>14</b>
5.1	Functions.....	14
5.2	Demo webpage .....	15
<b>6</b>	<b>Enhanced functionality by Webservice integration .....</b>	<b>21</b>
6.1	ICEdit and ICNativeWS hosted by a Tomcat servlet container .....	21
6.2	Service configuration.....	21
<b>7</b>	<b>Customer Support.....</b>	<b>22</b>

# 1 Preface

This documentation gives a detailed technical description of IEDIT JavaScript.

Here you can find information about:

- Changing the Layout of IEDIT
- Initialization of IEDIT with a structure/reaction
- Programming interface

**Please note:** For the use of IEDIT JavaScript please refer to the IEDIT JavaScript user manual IEdit\_JavaScriptEditor.pdf.

## 2 Configuration of IC<sub>EDIT</sub>

It is recommended to use a separate configuration file instead of setting the parameters in the html file.

If your configuration file is called **iceditjs.config.js** you may embed it in your html page:

...

```
<title>ICEdit JavaScript Version</title>
<script type="text/javascript" language="javascript" src="./iceditjs.config.js"></script>
<script type="text/javascript" language="javascript" src="iceditjs/iceditjs.nocache.js"></script>
```

...

Example of a configuration file (iceditjs.config.js):

```
// Application directory in format [DIRECTORY]/ for example "app/"
/**
Controller command for ICNativeServiceWS
e.g. "ICNativeWS/icedit/" (default)
    "app/icnative/"
*/
var ICNativeService = {
    path: "" //default is ICNativeWS/icedit/
};
// Image directory
var ImageDirectory = {
    path: "./img/iceditjs/"
};
/**
Atom Modi:
0: Complete (default)
1: Some functionality is removed by disabling
2: Some functionality is removed by setting invisible
*/
var FunctionalityMode = {
    Mode: 0
};
/**
0: Off (default)
1: On Hetero
2: On Terminal
3: On All
*/
var ShowHydrogens = {
    set: 3
};
// Atoms toolbar configuration
var Atoms = {
    // 1. Button
    description1: "Draw carbon atoms",
    image1: "c_atom.png", symbol1: "C",
    // 2. Button
    description2: "Draw nitrogen atoms",
    image2: "n_atom.png",
    symbol2: "N",
    // 3. Button
    description3: "Draw oxygen atoms",
    image3: "o_atom.png",
    symbol3: "O",
    // 4. Button
    description4: "Draw sulfur atoms",
    image4: "s_atom.png",
    symbol4: "S"
};
```

Alternatively you can set the parameters directly in your html page. For missing parameters common defaults will be used.

## 2.1 Parameter description

<pre>var ImageDirectory = {     path: "./img/iceditjs" };</pre>	Path to the directory with button images.
<pre>var FunctionalityMode = {     Mode: 0 };</pre>	Offers 3 different modes: 0: Complete (default) 1: Some functionality is missing by disabling 2: Some functionality is missing by setting invisible In 1 and 2 some atom and bond properties are missing.
<pre>var ShowHydrogens = {     set: 3 };</pre>	Offers 4 different options to show hydrogens: 0: Off (default) 1: On <u>Hetero</u> 2: On <u>Terminal</u> 3: On <u>All</u>

## 3 Changing the Layout of ICEDIT

The layout of ICEDIT JavaScript can be modified. The following layout items can be adapted according to your needs:

1. Configuration of the atom button bar
2. Height and width of the drawing area
3. Button images
4. Button background
5. Button bar background color
6. Add/Remove shadow effect

### 3.1 Configuration of ICEDIT atom button bar

This brief description will show you how to arrange the atom buttons on the Atom button bar.

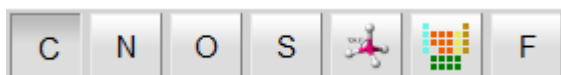


Fig. 1: Atom button bar

Figure 1 shows the standard atom button bar with its default atom buttons C, N, O and S. The button on the right of the atom button bar currently showing the F-atom is variable and shows the last selected atom from the periodic table of elements.

The atom buttons can be configured by changing the following JavaScript code which must be embedded in the HTML file containing the ICEDIT object:

```
// Atoms toolbar configuration
var Atoms = {
  // 1. Button
  description1: "Draw carbon atoms",
  symbol1: "C",

  // 2. Button
  description2: "Draw nitrogen atoms",
  symbol2: "N",

  // 3. Button
  description3: "Draw oxygen atoms",
  symbol3: "O",
```

```
        // 4. Button
        description4: "Draw sulfur atoms",
        symbol4: "S"
    };
```

If you want to add or change a button just add a new button section or change the appropriate button section in the atoms definition.

So for example if you want to have a P-atom button instead of N and you want to have an additional Si button, your atoms definition would look like this:

```
// Atoms toolbar configuration
var Atoms = {
    // 1. Button
    description1: "Draw carbon atoms",
    symbol1: "C",

    // 2. Button
    description2: "Draw phosphor atoms",
    symbol2: "P",

    // 3. Button
    description3: "Draw oxygen atoms",
    symbol3: "O",

    // 4. Button
    description4: "Draw sulfur atoms",
    symbol4: "S",

    // 5. Button
    description5: "Draw silicon atoms",
    symbol5: "Si"
};
```

**Please note:**

- The description definition will be used as a tool tip.
- If you want to add a new atom button, don't forget to increase the trailing number of the description and the symbol variable name.

After changing your ICEDIT HTML file your atom button bar will look like this:



Fig. 2: User-defined atom button bar

## 3.2 Width and height of the drawing area

### 3.2.1 Drawing area width

To change the width of the ICEDIT drawing area you need to change 3 values in the ICEditJS.css file. For example, if you want to increase the width from 600 to 700 pixels then you should make the following changes:

```
.subButtonPanel {  
  width: 700px;  
  height: 40px;  
  background-color: Lightgrey;  
  color: Lightgrey;  
}  
  
...  
  
.canvasPanel {  
  width: 700px;  
  height: 600px;  
  background-color: white;  
  color: black;  
}  
  
.mainPanel {  
  width: 700px;  
  height: 635px;  
  background-color: white;  
  color: black;  
  border-width: 1px;  
  border-style: solid;  
  border-color: black;  
  margin-left: 10px;  
  box-shadow: 10px 10px 5px #888888;  
}
```

**Please note:**

You should not use a width < 600 pixels because this space is needed for the button bar.



### 3.2.2 Drawing area height

Similar to the drawing area width you can change the height of ICEDIT but here you only have to change two values. In the following example the height was increased from 600 to 800 pixels:

```
.canvasPanel {
  width: 600px;
  height: 800px;
  background-color: white;
  color: black;
}

.mainPanel {
  width: 600px;
  height: 835px;
  background-color: white;
  color: black;
  border-width: 1px;
  border-style: solid;
  border-color: black;
  margin-left: 10px;
  box-shadow: 10px 10px 5px #888888;
}
```


**Please note:**

The height value in the *mainPanel* section must be greater than the drawing area height because of the button panel. In the example above the former value was 635.

You cannot reduce the height of the drawing area to 0 but if you want to dynamically hide or show the ICEDIT you can use the interface functions `hide()` and `show()`. Please refer to chapter 5 for further details.

### 3.3 Button images

The images for the buttons are png files which are located in the `img/iceditjs` directory. If you want to change one of these images just replace it with a 24x24 sized png image of your choice. For example the

image for the select button  is saved in the `img/iceditjs/select.png` file.

### 3.4 Button background

By default all ICEDIT buttons are 3d styled buttons with a grey color gradient. If you want to turn them for example into plain yellow buttons then please make the following changes (red text) in the ICEDITJS.css file:

```
.gwt-PushButton-up,  
.gwt-PushButton-up-hovering,  
.gwt-PushButton-up-disabled,  
.gwt-PushButton-down,  
.gwt-PushButton-down-hovering,  
.gwt-PushButton-down-disabled {  
  margin: 0;  
  text-decoration: none;  
  background: yellow;  
}  
...  
.gwt-ToggleButton-up,  
.gwt-ToggleButton-up-hovering,  
.gwt-ToggleButton-up-disabled,  
.gwt-ToggleButton-down,  
.gwt-ToggleButton-down-hovering,  
.gwt-ToggleButton-down-disabled {  
  margin: 0;  
  text-decoration: none;  
  background: yellow;  
}
```

Since there are two different kinds of buttons (push and toggle buttons) you have to make two changes in the css file.

## 3.5 Button bar background color

### 3.5.1 Background of the main button bar

The background of the main button bar can be set to an arbitrary color. If you want to have a purple background for example, you would have to make the following changes in the ICEditJS.css file:

```
.buttonBorder {  
  background-color: grey;  
  color: white;  
  border-width: 2px;  
  border-style: solid;  
  border-color: purple;  
}  
...  
.fillMainButtonPanel {  
  height: 33px;  
  width: 40px;  
  background-color: purple;  
  color: white;  
}  
...  
.mainButtonPanel {  
  width: 0px;  
  height: 40px;  
  background-color: purple;  
  color: white;  
}
```

```
...  
.mainPanel {  
  width: 600px;  
  height: 635px;  
  background-color: purple;  
  color: black;  
  border-width: 1px;  
  border-style: solid;  
  border-color: black;  
  margin-left: 10px;  
  box-shadow: 10px 10px 5px #888888;  
}
```

### 3.5.2 Background of the sub button bar

Similar to the main button bar, you can change the background color of the sub button bar by making the following changes:

```
.subButtonBorder {  
  background-color: grey;  
  color: white;  
  border-width: 2px;  
  border-style: solid;  
  border-color: purple;  
}  
  
.subButtonPanel {  
  width: 600px;  
  height: 40px;  
  background-color: purple;  
  color: lightgrey;  
}
```

### 3.6 Add/Remove shadow effect

If you want to remove the shadow behind the drawing area, just delete the *box-shadow* line:

```
.mainPanel {  
  width: 600px;  
  height: 635px;  
  background-color: white;  
  color: black;  
  border-width: 1px;  
  border-style: solid;  
  border-color: black;  
  margin-left: 10px;  
  box-shadow: 10px 10px 5px #888888;  
}
```

### 3.7 Select Text Font

You can select a different text font for the window title, dialog text and context menu entries by changing the `icEditTextFont` section of the `ICEditJS.css` file:

```
.icEditTextFont {  
    font-family: sans-serif, monospace;  
    font-size: 0.9em;  
}
```

### 3.8 Example

By default the *ICEDIT* JavaScript has the following 'Look & Feel':

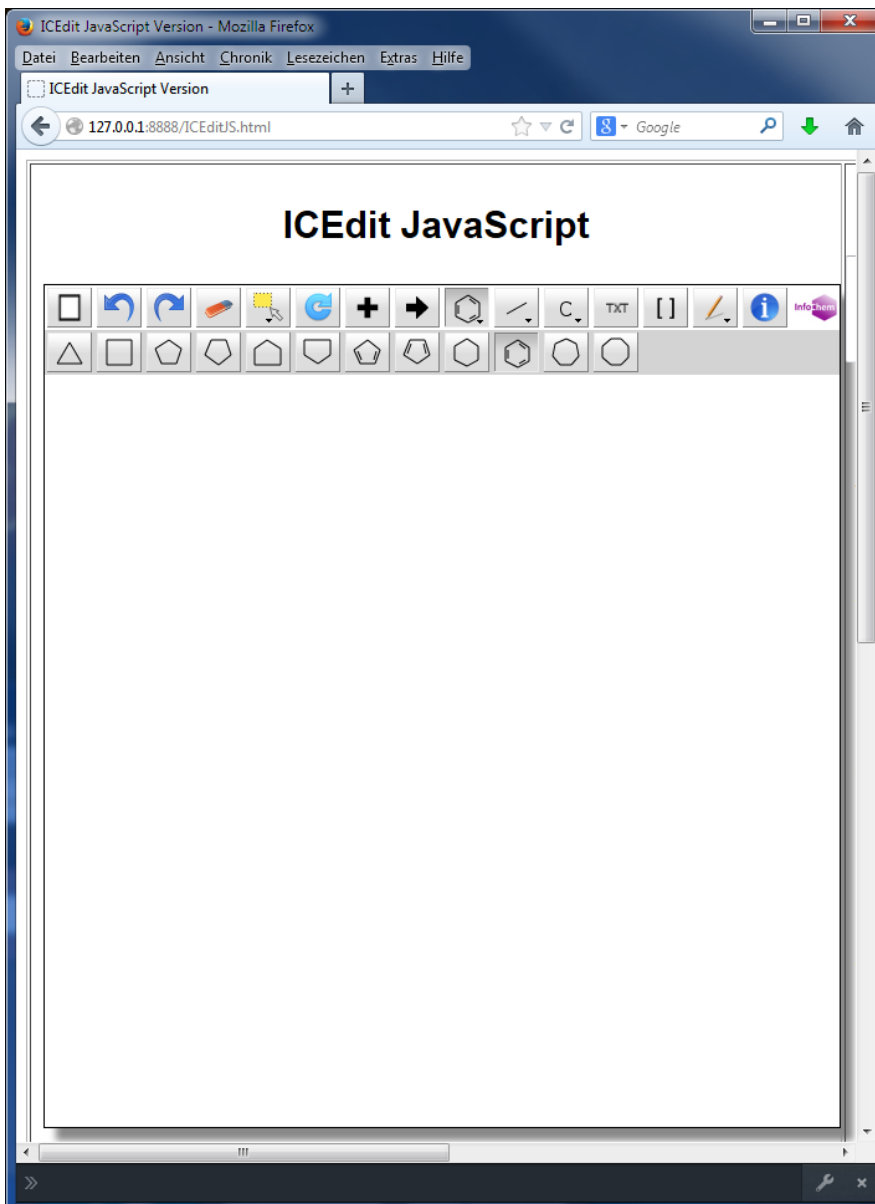
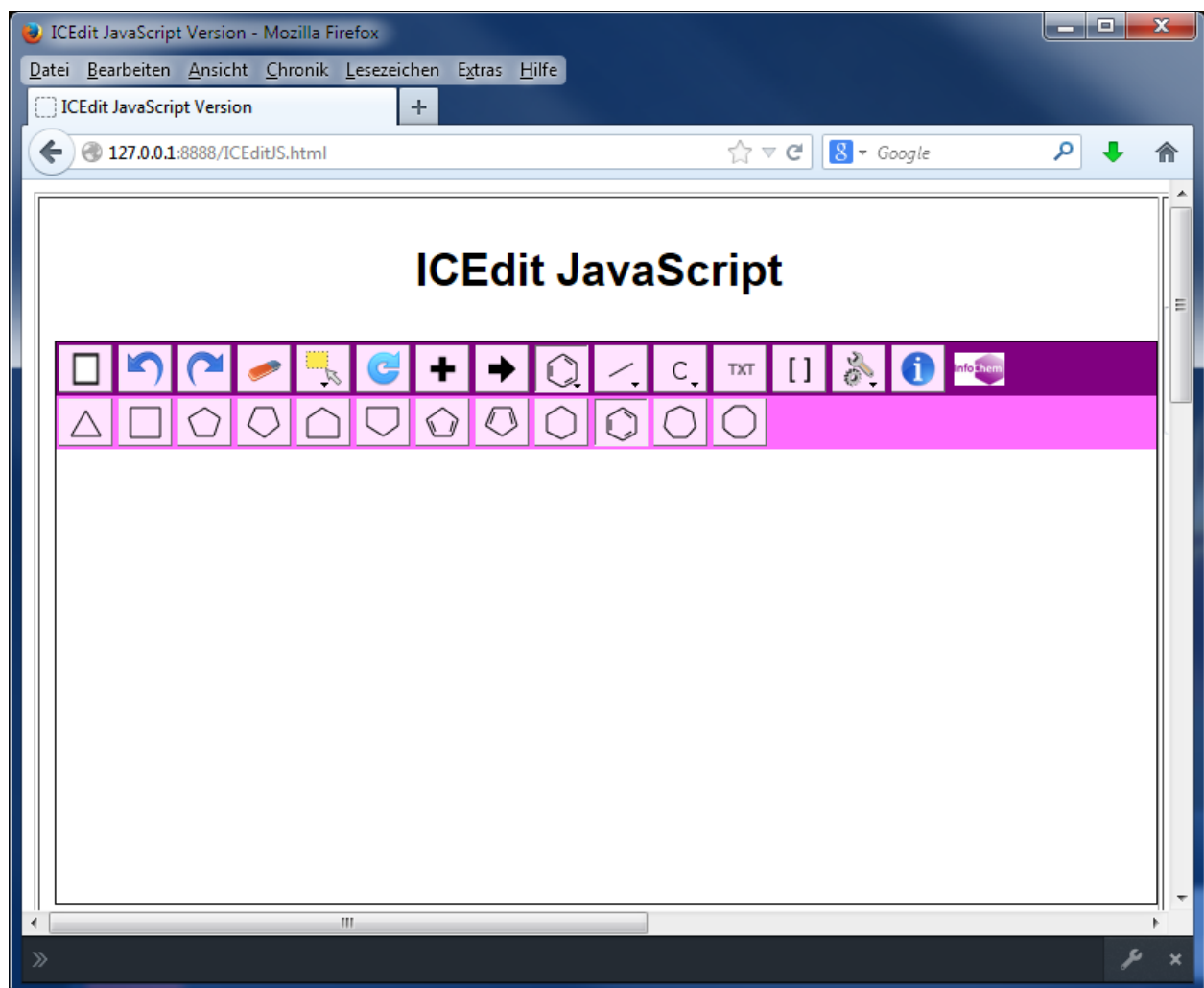


Fig. 3: *ICEDIT* default 'Look & Feel'

The changes listed below demonstrate how the 'Look & Feel' of ICEDIT can be modified (see above for the required changes):

- Width has been increased from 600 to 800 pixels.
- Height has been decreased from 600 to 300 pixels.
- Main button bar background color has been set to purple
- Sub button bar background color has been set to #FF6CFF
- The button color has been set to #FFE3FF (without color gradient)
- The shadow effect has been removed
- Pencil button image has been replaced

The modified 'Look & Feel' of ICEDIT:



**Fig. 4:** ICEDIT with modified 'Look & Feel'

## 4 Initialization

If you want to open the ICEDIT JavaScript application with an initial structure/reaction you can use the `init_rosdal` variable. When set, the ICEDIT shows the structure/reaction described by this variable.

### Example:

In this example the webpage containing the ICEDIT object has defined an `init_rosdal` variable which has been set to a benzene structure:

```
<script type="text/javascript">  
    var init_rosdal = "1 (X251,Y-106) , 2 (X251,Y-136) , 3 (X277,Y-151) , 4 (X303,Y-  
        136) , 5 (X303,Y-106) , 6 (X277,Y-91) , 1=2 (Q1024) , 2-  
        3 (Q1024) , 3=4 (Q1024) , 4-5 (Q1024) , 5=6 (Q1024) , 1-6 (Q1024) . ";  
  
    // ...  
</script>
```

The webpage shows an initial benzene structure when opened:

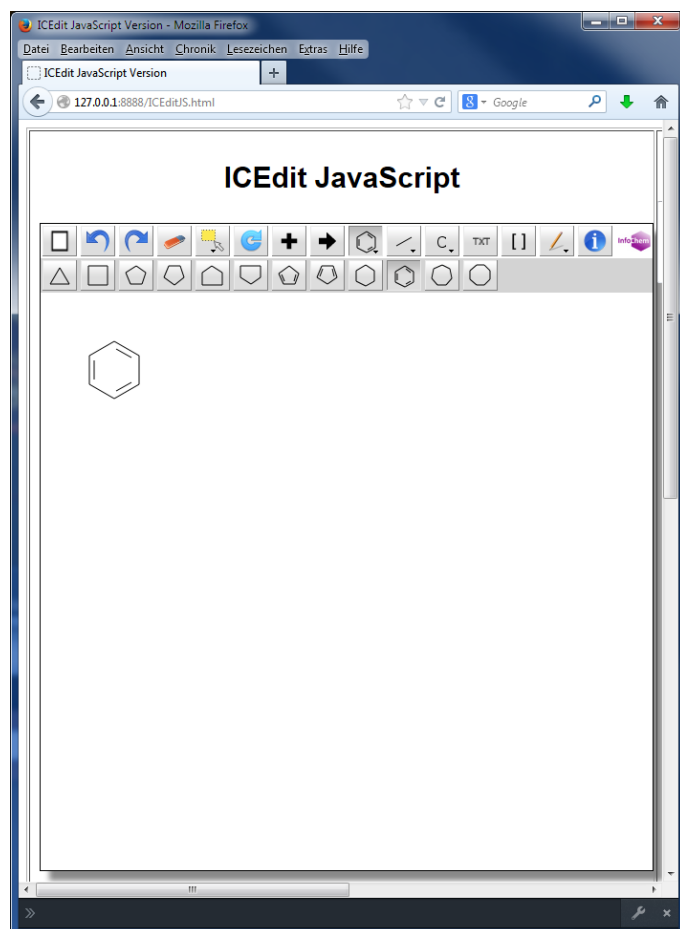


Fig. 5: ICEDIT with an initial benzene structure

## 5 Programming Interface

### 5.1 Functions

ICEDIT JavaScript offers a programming interface with the methods listed below. These methods can be accessed via the ICEDITJS object (e.g.: `ICEDITJS.getRosdal()`)

#### ***getRosdal()***:

The `getRosdal()` function returns the corresponding rosdal to the structure/reaction drawn in the ICEDIT drawing area.

#### ***setRosdal(rosdal)***:

You can use the `setRosdal()` function to draw a structure/reaction on the ICEDIT drawing area. The rosdal parameter is a string in rosdal format.

#### ***createPngFromRosdal(rosdal)***:

The `createPngFromRosdal()` function returns a png image as a data url of the type "image/png" which contains the structure/reaction represented by the rosdal parameter.

#### ***createPngFromRosdalWithBorder(rosdal, border)***:

Similar to `createPngFromRosdal()`, the `createPngFromRosdalWithBorder()` function creates a png image of the Rosdal but additionally provides the possibility to add a border around the structure/reaction. The border parameter represents the width of the border in pixels.

#### ***createPngFromRosdalToBox(rosdal, width, height, border)***:

This function creates a png image with user defined width and height values. The structure/reaction described by the rosdal parameter will be scaled so that it fits into the given size defined by the parameters width and height. In addition you can define a border around the structure/reaction.

#### ***setCanvasHeight(height)***:

You can dynamically change the height of the drawing area by calling the `setCanvasHeight()` function with the height in pixels as parameter.

#### ***hide()***:

The `hide()` function can be used to set the ICEDIT object to invisible.

#### ***show()***:

After setting the ICEDIT object to invisible you can set it to visible again with the `show()` function.

## 5.2 Demo webpage

The listing below is a demo webpage to demonstrate the interface functions:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <meta name="viewport" content="width=device-width, user-scalable=no" />
    <meta http-equiv="Cache-Control" content="no-cache" />
    <link type="text/css" rel="stylesheet" href="ICeditJS.css">

    <title>ICedit JavaScript Version</title>
    <script type="text/javascript" language="javascript"
      src="iceditjs/iceditjs.nocache.js"></script>
    <script type="text/javascript" language="javascript" src="./jquery.js"></script>
    <script type="text/javascript">
      //var init_rosdal = "1(X264,Y-169),2(X264,Y-199),3(X290,Y-214),4(X316,Y-199),
      //5(X316,Y-169),6(X290,Y-154),1=2(Q1024),2-3(Q1024),3=4(Q1024),4-5(Q1024),5=6(Q1024),
      //1-6(Q1024).";

      function windowOpen(address) {
        var myWindow = window.open(address, "ICeditJS", "width=600,height=600,scrollbars=yes");
        myWindow.focus();
      }

      function getRosdal2()
      {
        return document.getElementById("rosdal").value;
      }

      function copyRosdal()
      {
        var result = ICeditJS.getRosdal();
        document.getElementById("rosdal").value = result;
      }

      function copyExpandedRosdal()
      {
        var result = ICeditJS.getExpandedRosdal();
        document.getElementById("rosdal").value = result;
      }

      function getCanvasHeight()
      {
        return parseInt(document.getElementById("heightCanvas").value);
      }
    </script>
  </head>
  <body>
    <div id="rosdal">
      <input type="text" value="" />
    </div>
    <div id="heightCanvas">
      <input type="text" value="" />
    </div>
  </body>
</html>
```



```
function getPngBorder()
{
    return parseInt(document.getElementById("png_border").value);
}

function getPngWidth()
{
    return parseInt(document.getElementById("png_width").value);
}

function getPngHeight()
{
    return parseInt(document.getElementById("png_height").value);
}

$(function() {
    if (window.opener != null)
    {
        // When the Initialization button is pressed, this webpage is called again
        // and the calling window instance (window.opener) is set. In this case,
        // the init_rosdal variable will be set with the Rosdal of the parent window.
        init_rosdal = window.opener.document.getElementById("rosdal").value;
    }

    function createImage() {
        try {
            //var imgSource = ICEditJS.createPngFromRosdal(
            // document.getElementById("rosdal").value);
            //var imgSource = window.createPngFromRosdal(
            // document.getElementById("rosdal").value);
            var imgSource = ICEditJS.createPngFromRosdalWithBorder(
                document.getElementById("rosdal").value, getPngBorder());
            $("#image").attr({ src: imgSource });
        }
        catch(e) {
            console.log("error",e);
            alert(e);
        }

        return false;
    }

    $("#img-button").bind("click", function() { createImage(); });

    function createImageToBox() {
        var imgSource = ICEditJS.createPngFromRosdalToBox(
            document.getElementById("rosdal").value, getPngWidth(), getPngHeight(), 2);
        $("#image-box").attr({ src: imgSource });
        return false;
    }
}
```

```

$("#img-button-box").bind("click", function() { createImageToBox(); });

function createImageMany() {
  var start = new Date().getTime();

  for (var i=0;i<1000;i++) {
    var imgSource = ICEditJS.createPngFromRosdal('1N,1(X257,Y-159),2(X257,
      Y-189),3N,3(X283,Y-204),4(X309,Y-189),5(X309,Y-159),6N,6(X283,Y-144),1=2(Q1024),
      2-3(Q1024),3=4(Q1024),4-5(Q1024),5=6(Q1024),1-6(Q1024).');
    //var imgSource = window.createPngFromRosdal(document.getElementById("rosdal").value);
    $("#image-many").attr({ src: imgSource });
  }

  var end = new Date().getTime();
  alert(end-start + "ms")

  return false;
}

$("#img-button-many").bind("click", function() { createImageMany(); });
});
</script>
</head>
<body>
<!-- RECOMMENDED if your web app will not function without JavaScript enabled -->

<table width="100%" border="1" cellpadding="0" cellspacing="2">
  <tr>
    <td valign="top">

      <noscript>
        <div
          style="width: 22em; position: absolute; left: 50%; margin-left: -11em; color: red;
          background-color: white; border: 1px solid red; padding: 4px;
          font-family: sans-serif">
          Your web browser must have JavaScript enabled
          in order for this application to display correctly.
        </div>
      </noscript>

      <h1>
        <p align="center">ICEdit JavaScript</p>
      </h1>
      <div id="canvasholder"></div>
      <br />

    </td>
    <td>
      <h2>
        <p align="center">Interface Demo</p>
      </h2>
    </td>
  </tr>
</table>

```

```
<p align="center">
  <textarea id="rosdal" style="width:500px;height:80px"></textarea>
</p>

<table width="100%" border="0" cellpadding="4" cellspacing="4">
  <tr>
    <td>
      <p>
        <b>1. Create a test structure/reaction</b>
      </p>
      <p>
        Please draw a structure/reaction with
        <i>ICEdit JavaScript</i>
        which is needed for the steps below.
      </p>
    </td>
  </tr>
  <tr>
    <td>
      <p>
        <b>2. Display Rosdal</b>
      </p>
      <p>
        The getRosdal() function provides you the corresponding
        Rosdal to the test structure/reaction:
      </p>
      <input type="button" value="get Rosdal" onclick="copyRosdal()">
    </td>
  </tr>
  <tr>
    <td>
      <p>
        <b>3. Drawing a structure/reaction</b>
      </p>
      <p>
        You can use the setRosdal() function to draw a
        structure/reaction.
        <br />
        In this example the Rosdal in the text area will be used to
        draw a structure/Reaction in the ICEdit canvas:
      </p>
      <input type="button" value="set Rosdal"
        onclick="ICEditJS.setRosdal(getRosdal2());">
    </td>
  </tr>
  <tr>
    <td>
      <p>
        <b>4. Creating png images</b>
      </p>
      <p>
```

```

    The createPngFromRosdal()/createPngFromRosdalWithBorder()
    functions create a png file with a rosdal/rosdal,border as
    parameters:
</p>
<button id="img-button">create image</button>
<p>
    Border:
</p>
<input type="text" id="png_border" size="3" maxlength="3"
    value="10">
    <img id="image" style="border: 1px solid steelblue;" />
</td>
</tr>
<tr>
<td>
    <p>
        <b>5. Creating scaled png images</b>
    </p>
    <p>
        Similar to createPngFromRosdal() the
        createPngFromRosdalToBox() function creates a png file with
        user defined width and height values:
    </p>
    <button id="img-button-box">create scaled image</button>
    <p>
        Width:
    </p>
    <input type="text" id="png_width" size="3" maxlength="3"
        value="100">
        Height:
    <input type="text" id="png_height" size="3" maxlength="3"
        value="100">
    <img id="image-box" style="border: 1px solid steelblue;" />
    </td>
</tr>
<tr>
<td>
    <p>
        <b>6. Performance test</b>
    </p>
    <p>
        By clicking the button below 1000 png files will be
        generated. When the process has finished the time used to
        create the files will be displayed.
    </p>
    <button id="img-button-many">create many images</button>
    <img id="image-many" style="border: 1px solid steelblue;" />
    </td>
</tr>
<tr>
<td>
```

```
<p>
  <b>7. Layout</b>
</p>
<p>
  You can change the height of ICEdit by calling the
  setCanvasHeight(int height) function with the height in pixels
  as parameter.
</p>
<input type="button" value="set height"
  onclick="ICEditJS.setCanvasHeight(getCanvasHeight());">
<input type="text" id="heightCanvas" size="3" maxlength="3"
  value="300">
<p>
  You can hide/show the ICEdit component by calling the
  hide()/show() function.
</p>
<input type="button" value="hide" onclick="ICEditJS.hide();">
<input type="button" value="show" onclick="ICEditJS.show();">
</td>
</tr>
<tr>
<td>
  <p>
    <b>8. Initialization</b>
  </p>
  <p>
    An initial structure can be passed to the ICEdit by setting
    the init_rosdal variable. In case the ICEdit is opened in a new
    window the variable init_rosdal should be set by the child
    window by setting the variable with the value of a rosdal
    textarea of the parent window. In this demo the textarea above
    (set with step 2) is used to hand over the rosdal.
  </p>
  <input type="button" value="open ICEdit with rosdal initialization"
    onclick="windowOpen(document.URL);">
  </td>
</tr>
</table>

</td>
</tr>
</table>

</body>
</html>
```

## 6 Enhanced functionality by WebService integration

It is possible to extend ICEdit with some additional functionalities like full superatom support.

Therefore ICEdit needs a connection to Infochem's WebService **ICNativeWS**. This connection must follow the **Same-Origin-Policy** (SOP).

### 6.1 ICEdit and ICNativeWS hosted by a Tomcat servlet container

1. Follow **ICNativeWS\_Technical\_Documentation.pdf** to install ICNativeWS on a Tomcat servlet container.
2. Unpack **ICEditDemo.zip** to \$CATALINA\_HOME/webapps directory.
3. Navigate your browser to <http://127.0.0.1:8080/ICeditDemo/ICeditJS.html>. Draw a bond with COOH for one atom. This atom should be expandable by the context menu.

### 6.2 Service configuration

If you are using a proxy, you may have to configure the ICNativeService in your HTML file:

```
// Native service configuration
var ICNativeService = {
    path: "" //default is ICNativeWS/icedit/
    //path: "ICNativeWS/icedit/"
    //path: "app/icnative/"
};
```

## 7 Customer Support

Please contact us to receive technical support or license information:

INFOCHEM GESELLSCHAFT FÜR CHEMISCHE INFORMATION MBH  
Landsbergerstrasse 408/V  
D-81241 München  
Germany

Phone: +49 (0)89 583002

Fax: +49 (0)89 5803839

Email: [iceditsupport@infochem.de](mailto:iceditsupport@infochem.de)

For general information about InfoChem products, see the InfoChem homepage at <http://www.infochem.de/>.