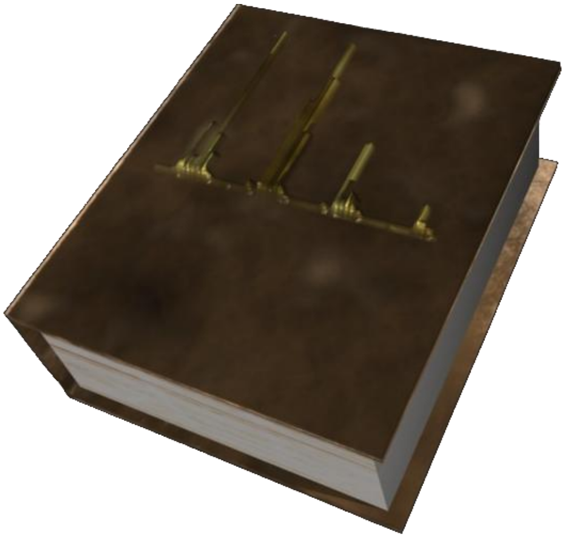
# mSpecs Editor Documentation



http://mSpecs.tu-bs.de

Last modified: Saturday, December 13, 2008

# Table of Contents

1. Introduction ........ 3
   1.1. What is mSpecs Editor? ........ 3
   1.2. Credits and Copyright Notice ........ 3
2. Installation ........ 3
   2.1. Windows ........ 3
   2.2. Linux ........ 3
   2.3. MacOSX ........ 3
3. User Interface ........ 3
   3.1. Identifiers ........ 5
   3.2. Name, Synonyms and Derivative ........ 5
   3.3. Molecular Weight and Monoisotopic Mass ........ 5
   3.4. Chemical Identifiers ........ 6
   3.5. Retention Time, Retention Index, Retention Index Drift and Alkanes ........ 6
   3.6. Mass Traces ........ 6
   3.7. Structures ........ 6
   3.8. Spectrum ........ 7
   3.9. Measurement Data ........ 7
   3.10. Systems Biology Data ........ 7
4. Menus and Dialogs ........ 7
   4.1. File Menu ........ 7
   4.2. Edit Menu ........ 7
   4.3. Tools Menu ........ 7
      4.3.1. Automatic Calculations ........ 8
      4.3.2. Settings Dialog ........ 8
   4.4. Scripts Menu ........ 9
   4.5. Help Menu ........ 9
5. Scripts ........ 9
   5.1. Available Objects and Functions ........ 9
   5.2. The ‘library’ Object ........ 10
   5.3. Accessing the Elements of a Library ........ 10
   5.4. The ‘tools’ Object ........ 12

# 1. Introduction

## 1.1. What is mSpecs Editor?

mSpecs Editor is designed to help managing and editing mass spectral libraries. It provides capabilities to enter diverse information in addition to spectra such as identification numbers, formulas, molecular weights, structures, comments etc. Its scripting functionalities help to manage large libraries in a short time.

## 1.2. Credits and Copyright Notice

mSpecs Editor is © 2008 by Dr. rer. nat. Bernhard Thielen and Dr. rer. nat. Stephanie Heinen (both former members of the Department of Biochemistry, University of Cologne). The application is build upon the Qt GUI Toolkit (Copyright © 2008 Nokia Corporation, see http://www.trolltech.com/qt). The Qt icon is © by Nokia Corporation; some of the other icons were taken from the NuoveXT KDE-desktop theme (http://nuovext.pwsp.net/).

mSpecs Editor is licensed under the terms of the GNU General Public License 2 (see License.txt).

# 2. Installation

The latest version of mSpecs Editor is available at http://mSpecs.tu-bs.de.

## 2.1. Windows

Download the installer, start the installation by double-clicking the file and follow the installation instructions.

## 2.2. Linux

The Linux version can be compiled from source code. Ensure that the latest version of Qt and a recent compiler version are installed. Qt is available from http://trolltech.com/downloads/opensource/appdev/linux-x11-cpp. Most Linux versions also include ready-to-use Qt binaries within their package managers.

In order to compile open a shell and navigate to the folder where the source code of mSpecs is unpacked. Type ‘qmake’ followed by ‘make’ and the executable should be present in the “build” subdirectory.

## 2.3. MacOSX

There is currently no package for MaxOSX available, so mSpecs has to be build from source code. The procedure is similar to the one used under Linux explained in the section above.

# 3. User Interface

The user interface is subdivided into two parts (see figure 1), from which the left one is visible after opening or creating a library. It contains buttons and text fields to sort and filter their entries. On the right side data of the selected compound can be entered. This part of the screen is further subdivided into several tab-pages that can be accessed by clicking onto the registers in the upper part. There are multiple data fields available (see table 1), from which some are explained in more detail below.

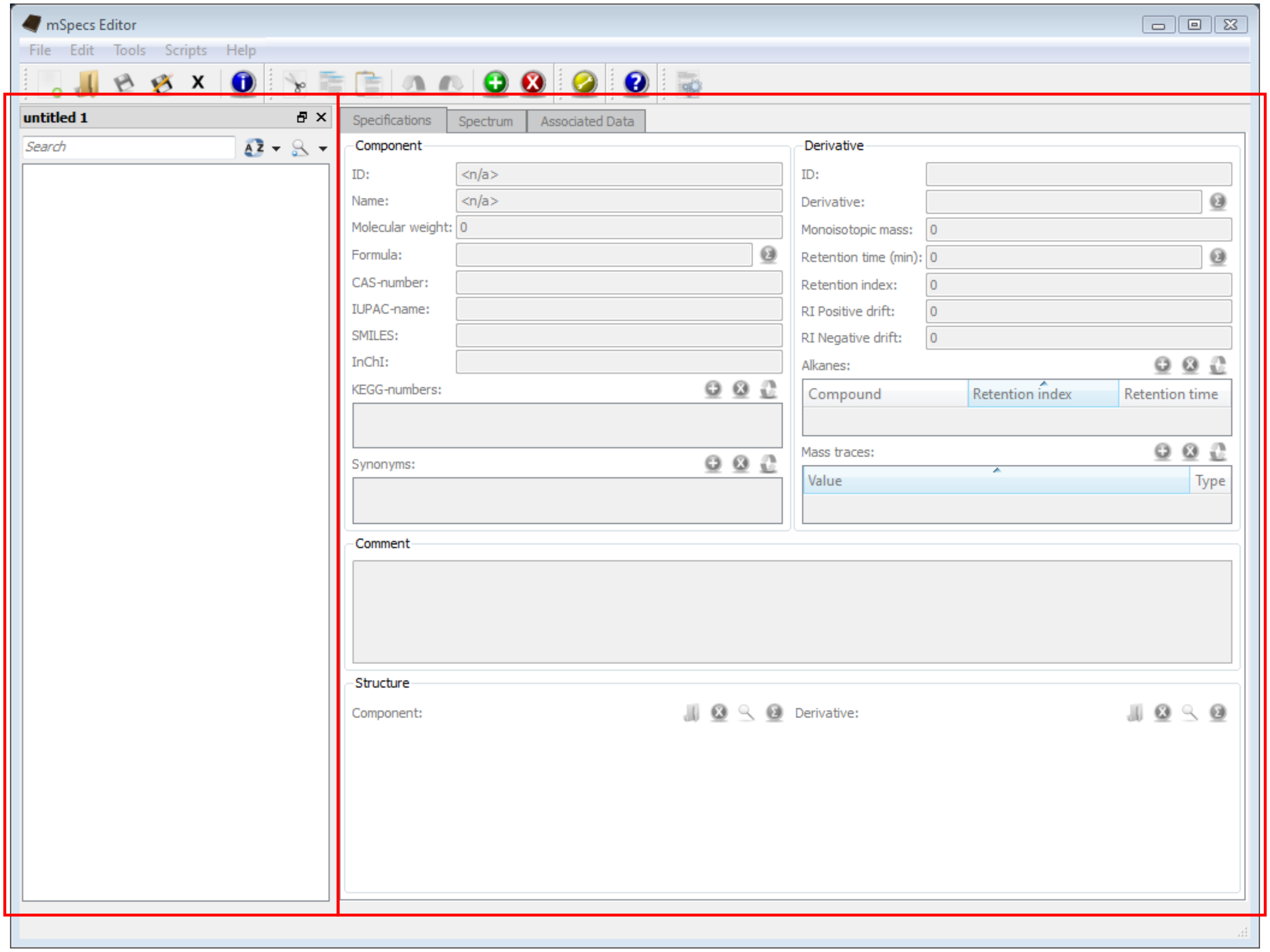


Figure 1: The user interface of mSpecs is divided into two parts (noted by red boxes). The left part shows currently opened libraries and on the right side the data of the selected compound can be entered.

Table 1: Available data field, their type and if they are sortable and filterable, respectively.

| Field | Type | Comment | Sortable/Filterable |
|---|---|---|---|
| *Specifications* | | | |
| Component | | | |
| ID | string | identification number | yes / yes |
| Name | string | identifier | yes / yes |
| Molecular weight | double | molecular weight | yes / yes |
| Formula | string | chemical formula | no / yes |
| CAS-number | string | CAS-number | no / yes |
| IUPAC-name | string | IUPAC-name | no / yes |
| SMILES | string | SMILES-code [29] | no / yes |
| InChI | string | InChI-code [13] | no / yes |
| KEGG-numbers | array of strings | KEGG compound number [10] | no / yes |
| Synonyms | array of strings | synonyms | no / yes |
| Derivatives | | | |

| Field | Type | Comment | Sortable/Filterable |
|---|---|---|---|
| ID | string | identification number | yes / yes |
| Derivative | string | identifier | yes / yes |
| Monoisotopic mass | double | monoisotopic mass [30] | yes / yes |
| Retention time (min) | double | Retention time | yes / yes |
| Retention index | double | Kováts retention index [14] | yes / yes |
| RI positive drift | double | Positive drift of retention index | yes / yes |
| RI negative drift | double | Negative drift of retention index | yes / yes |
| Alkanes | 2D array[1] | reference list of alkanes | no / no |
| Mass traces | array of doubles | list of reference ions for quantification | no / yes |
| Comment | | | |
| Comment | string | commentaries | no / yes |
| Structure | | | |
| Component | molecule[2] | chemical structure of the component | no / no |
| Derivative | molecule[2] | chemical structure of the derivative | no / no |
| *Spectrum* | | | |
| Visual representation | – | graphical representation | no / no |
| Tabular representation | 2D array[1] | tabular representation | no / no |
| *Associated data* | | | |
| Measurement | | | |
| Author | string | author of the measurement | no / yes |
| Date | date | date of measurement | yes / yes |
| Device | string | device used | no / yes |
| Method | string | explanation to the method | no / yes |
| Column | string | used chromatographical column | no / yes |
| Experimental conditions | string | further experimental conditions | no / yes |
| Systems Biology Data | | | |
| Reactions | array of strings | associated reactions | no / yes |
| Enzymes | 2D array[1] | associated enzymes | no / yes |
| Genes | 2D array[1] | associated genes | no / yes |

[1] Table with given number of columns and arbitrary number of rows.

[2] Object storing a molecular structure.

## 3.1. Identifiers

The ID fields of the component and derivative are used to identify the component and the derivative, respectively. It is recommended to use unique numbers or strings for every component on the component-ID field and for every derivative on the derivative-ID field, e.g. “Ala” for the compound Alanine and “Ala01”, “Ala02” etc. for its derivatives.

## 3.2. Name, Synonyms and Derivative

The name of the component and its derivative can be entered here. Besides the derivative field there is a button to calculate the monoisotopic mass (see below) of the derivative using the molecular formula of the component and the string stored in the derivative field (see 4.3.1 for explanations about the calculation mechanism). Additional synonyms of the substance can be entered in the list of synonyms in the lower part of the component-box.

## 3.3. Molecular Weight and Monoisotopic Mass

Use these fields to enter the molecular weight of the component and the monoisotopic mass of the derivative. The monoisotopic mass is the sum of the masses of the atoms in a molecule using the unbound mass of the most abundant isotope. This results in the peak with the highest mass when leaving out the higher isotope-peaks in the spectrum. Using the button next to the formula field the molecular weight of the component can be calculated. Additionally, the monoisotopic mass of the derivative can be calculated using the molecular formula and the derivative name (see above).

## 3.4. Chemical Identifiers

The CAS-number, IUPAC-name, SMILES and InChI-fields can be used to enter corresponding data relevant to the component. After entering KEGG-numbers in the corresponding area, one can retrieve information online by right-clicking on the entry and choosing “Retrieve online information”. The Url called for displaying the online information can be changed in the settings (see section 4.3.2).

## 3.5. Retention Time, Retention Index, Retention Index Drift and Alkanes

The retention time during chromatography is used to discriminate metabolites with a similar mass spectrum like e.g. monosaccharides. In order to increase independence of the retention time behavior to the length of the column used and experimental conditions the Kováts retention index can be calculated automatically using the reference list entered in the alkanes field. Following formula is used for the calculation:

$$RI_c = \left(RT_c + \frac{RI_1 \cdot RT_2 - RI_2 \cdot RT_1}{RI_2 - RI_1}\right) \cdot \frac{RI_2 - RI_1}{RT_2 - RT_1}$$

Abbreviations $RI_c$: retention index of the compound; $RT_c$: retention time of the compound; $RT_1$: retention time of the preceding alkane; $RT_2$: retention time of the subsequent alkane; $RI_1$: retention index of the preceding alkane; $RI_2$: retention index of the subsequent alkane.

If a given retention time is smaller than the smallest given reference compound or bigger than the biggest given reference substance the retention index is extrapolated from the two reference compounds with the smallest and the biggest retention time, respectively.

The retention index drift fields are used to note if a derivative has a sharp or diffuse appearance in the chromatogram.

## 3.6. Mass Traces

Mass traces are usually used to further identify and quantify derivatives. A mass to charge ratio and a type can be entered here.

## 3.7. Structures

Structures for the components as well as for the derivative can be imported using the “Load structure” command. The supported file formats include the Chemical Markup Language (CML) Version 1 and 2 and the MDL Mol file. Please note that not all structure like e.g. showing radicals are supported currently.

With a given structure the molecular formula can be calculated. Hereby the number of hydrogen is calculated by saturating the carbon- and nitrogen-atoms.

A structure can also be exported into various formats including Scalable Vector Graphics (SVG) and several bitmap types using the “Export structure” command.

## 3.8. Spectrum

A mass spectrum is usually composed out of a list of mass-to-charge and intensity numbers. On the spectrum page, the mass spectrum can be viewed in the upper part and edited in the lower part. Every mass-to-charge and intensity pair can be further specified by entering textual data in the table.

## 3.9. Measurement Data

Data regarding the author of the measurement, the date, the device, method and the chromatographic column can be entered here. The experimental conditions field is supposed to be used for additional information concerning the measurement.

## 3.10. Systems Biology Data

The data fields available in the systems biology area are provided to link information regarding reactions, enzymes and genes to the compound. Online information can be retrieved by using the context menu. The Urls called here can be modified using the settings-dialog (see section 4.3.2). Per default the reactions are parsed as KEGG-reaction numbers and the enzymes as EC-numbers using the KEGG web page. For the genes, the sequence is used for initiating a blast search.

# 4. Menus and Dialogs

## 4.1. File Menu

Besides the common load-, save- and new-commands the file menu offers the possibility to import, export and print libraries as well as showing a dialog containing its properties.

In the properties dialog, the maintainer, contact information and notes can be added to the currently active library. On the second page the number and ratio of filled data of each data field is shown.

For import, the AMDIS/NIST and JCAMP-DX library files are supported. In addition to that, the user may export to these formats, to the Portable Document Format (pdf) and as tab-delimited text file. Please note that not all fields can be exported into every file format (e.g. exporting structures to the tab-delimited text file is not possible). Lastly mSpecs offers its own XML-specification in order to provide an easy adaptable format for other tools.

## 4.2. Edit Menu

The edit menu contains commands to handle library entries. The undo- and redo-menu items can handle every action changing the library including those committed by using a script (see section 5).

## 4.3. Tools Menu

In the tools menu, the automatic calculations explained in section 3 and a link the settings dialog are available.

### 4.3.1. Automatic Calculations

In the following list, the available automatic calculations are specified together with the data fields necessary and affected:

- Calculate component's formula:

  ***needed:*** component's structure

  ***affected:*** component's formula

  The formula is calculated by counting the weights and the number of atoms contained in the structure. Hereby the carbon- and nitrogen-atoms are saturated with hydrogen.

- Calculate component's molecular weight:

  ***needed:*** component's formula

  ***affected:*** component's molecular weight

- Calculate derivative's formula:

  ***needed:*** derivative's structure, perhaps the component's structure

  ***affected:*** derivative field

  The calculation is done as explained for the component's formula. Depending on the settings (see below), further steps can be carried out. Either the difference between the derivative's and the component's formula can be stored, or the user can display the difference in the common MeOX (Methoxyamine) / TMS (trimethylsily) notation.

- Calculate derivative's monoisotopic mass:

  ***needed:*** derivative field, perhaps the component's formula

  ***affected:*** derivative's monoisotopic mass

  The monoisotopic mass of the derivative is calculated depending on the basis of the notation adjusted in the settings dialog and explained before.

- Calculate retention index from alkane mix:

  ***needed:*** retention time, alkanes

  ***affected:*** retention index

  The retention index can be calculated using the mechanism explained in section 3.5.

### 4.3.2. Settings Dialog

The settings dialog contains options to customize the behavior of mSpecs.

- The calculation way of the derivative's formula and monoisotopic mass can be adjusted as noted above.
- During export as XML-file, the user can choose in which way the structures are exported. Please note that by exporting the structures in SVG they cannot be imported anymore.
- The urls used for retrieving information on the internet can be entered in the lower part of the dialog. The data field used for the dynamic part of the url is drawn from the corresponding line in case of the KEGG-numbers and

reactions. For the enzymes and genes, the user can additionally choose the column. In order to mark the position where the string is inserted, the sequence '%1' (without the apostrophes) is used.

- On the second page the user can choose which data fields should be visible and therefore customize the look of the application. Please note that no data is deleted upon hiding a data field.

## 4.4. Scripts Menu

The scripts menu shows a command leading to the scripts dialog as well as menu items to start recently used scripts. The manage scripts dialog box offers commands to run, create, import, edit and delete scripts. Please note that there is no editor provided for editing the scripts. Upon pressing the edit-button, mSpecs will ask for an editor. Choose your preferred text editor, ideally with Javascript as highlighter. The scripts are stored in either in the scripts folder of the application (for the zip-version) or the users directory (e.g. "C:\Users\Computer\AppData\Roaming\mSpecs Editor\scripts" in Windows Vista). For further information regarding the script language, refer to section 5.

## 4.5. Help Menu

The help menu leads to this file and further information concerning the application.

# 5. Scripts

The script engine included in mSpecs offers a tool for maintaining and analyzing even large libraries in a short time by automating the editing of the entries. The language used is build upon the ECMAScript scripting language, which is also the basis of e.g. JavaScript. This documentation skips information concerning the syntax and the usage (see e.g. http://en.wikibooks.org/wiki/Javascript) and concentrates on the available functions. Commented examples are supplied together with the application.

The comment visible in the scripts dialog is done by adding the following code to the top of the script:

```
/* description:
This is a description of the script.
You can also enter text in additional lines.
*/
```

In the following, the phrases in brackets hint to the type of the object and the type of the value returned, respectively.

## 5.1. Available Objects and Functions

On the top level, there are four objects available:

- `current` (Number) The 'current' object points to the index of the current element in the library. Please note that this index is not valid anymore if you change the number of entries in the library within the script. Returns -1 if there is no current element in the library.
- `library` (Object) Offers options to access the entries and do operations such as inserting and removing elements (5.2).

- `tools` (Object) Contains functions to write to a log window, use automatic calculations (see 4.3.1) and to interact with the user (5.4).
- `ok` (Boolean) Is used for some of the methods of the tools object (5.4).

## 5.2. The 'library' Object

The library object offers the following fields:

- `maintainer` (String) read and write the maintainer field of the library.
- `contactInformation` (String) read and write the contact information field of the library.
- `notes` (String) read and write the notes field of the library.
- `length` (Number) read the size of the library.

Additionally the following functions are supplied:

- `at(index)` (Object) access the element of the library at the given *index* (see the next section). Returns a null value if the index is outside of the boundaries of the library.
- `push()` (Number) adds an empty element to the library. Returns the index of the newly added element.
- `removeAt(index)` removes the element at the given *index*. Does nothing if the index is outside of the boundaries of the library.

## 5.3. Accessing the Elements of a Library

The elements of a library are accessed through the `at()` function of the library. They offer following fields. Notice that there is no check if a field (e.g. a molecular formula) is stored in a meaningful way in terms of chemical correctness.

- `alkaneMix` (special) read and write the reference substances. The alkane mix is read and stored as a two-dimensional array, where each row of the first array is composed by another array with three elements. The first element contains a string with the compound's name, the second a number with the retention index and the third a number with the retention time.
- `associatedEnzymes` (special) read and write the associated enzymes. The associated enzymes are stored as a two-dimensional array, where each row of the first array is composed by another array with two elements. The first element contains a string with the enzyme name and the second a string with the EC-number.
- `associatedGenes` (special) read and write the associated genes. The associated genes are stored as a two-dimensional array, where each row of the first array is composed by another array with three elements. The first element contains a string with the gene name, the second a string with the location and the third a string with the sequence.

- `associatedReactions` (Array of Strings) read and store the associated reactions.
- `author` (String) read and store the measurement's author.
- `casNumber` (String) read and store the CAS-number of the component.
- `column` (String) read and store the chromatographic column used.
- `comment` (String) read and store commentaries.
- `conditions` (String) read and write the experimental conditions.
- `date` (Date) read and write the date of measurement.
- `derivative` (String) read and store the derivative's name.
- `device` (String) read and store the device used.
- `formula` (String) read and store the molecular formula.
- `idComponent` (String) read and store the ID of the component.
- `idDerivative` (String) read and store the ID of the derivative.
- `inchi` (String) read an store the component's InChI string.
- `iupacName` (String) read and store the IUPAC name.
- `keggNumbers` (Array of Strings) read and store the component's KEGG-numbers.
- `massTraces` (special) read and write the mass traces. The mass traces are read and stored as a two-dimensional array, where each row of the first array is composed by another array with two elements. The first element is a number (the mass trace) and the second a string with the type ("Quantifier", "Confirmed" and "Excluded", respectively). Notice that "Quantifier" will be taken as type if a wrong string is provided.
- `method` (String) read and store the method used.
- `molecularWeightComponent` (Number) read and store the molecular weight of the component.
- `molecularWeightDerivative` (Number) read and store the molecular weight of the derivative.
- `moleculeComponent` (Object) read and store the component's molecular structure. There are no methods available to handle single elements of the molecule such as atoms or bonds. However, structures can be imported and exported from and to strings (see section 5.4).
- `moleculeDerivative` (Object) read and store the derivative's molecular structure. There are no methods available to handle single elements of the molecule such as atoms or bonds. However, structures can be imported and exported from and to strings (see section 5.4).
- `name` (String) read and store the name of the component.
- `retentionIndex` (Number) read and store the retention index.

- `retentionIndexNegativeDrift` (Number) read and store the negative drift of the retention index.
- `retentionIndexPositiveDrift` (Number) read and store the positive drift of the retention index.
- `retentionTime` (Number) read and store the retention time.
- `signals` (special) read and store the spectrum's signals. The spectrum's signals are stored as a two-dimensional array, where each row of the first array is composed by another array with three elements. The first element contains a number with the mass-to-charge value, the second a number with the intensity and the third a string with the comment.
- `smiles` (String) read and store the SMILES of the component.
- `synonyms` (Array of Strings) read and store the component's synonyms.

## 5.4. The 'tools' Object

The tools object contains various methods to interact with the user, execute automatic calculations and showing the progress of a script. The functions are accessed as followed:

- `loadTextFile(path)` (String) reads a text file at the `path` given as a string. The path can be absolute (e.g. "C:/Users/Computer/textfile.txt") or relative starting from the folder where the scripts are stored (e.g. "subfolder/subfolder2/textfile.txt"). Note to windows users: be aware to use slash (/) as a directory-separator, not backslash.
- `getOpenFileName(caption, filter, path)` (String) shows a dialog with a given `caption` where the user can select a file. The `filter` is a string used to restrict the number of files shown. A filter for a single type of files could be e.g. "Images (*.png *.bmp *.jpg)". Multiple filters are separated using two semicolon (e.g. "Images (*.png *.xpm *.jpg);;Text files (*.txt);;All files (*)"). The `path` is an optional string containing the folder that is shown on startup of the dialog. If no path is provided, the default directory of the scripts is used. Returns an empty string if the user cancels the dialog.
- `saveTextFile(path, text)` (Boolean) writes a string provided in the `text`-parameter into the given path. The path should contain the filename and can be provided either in an absolute or a relative manner (see above). Returns true when the file is saved successfully.
- `getSaveFilename(caption, filter, path)` (String) shows a dialog where the user can select a file. See `getOpenFileName()` for additional information.
- `messageBoxInformation(title, text)` shows a message box with the given `title` and `text`.

- `importStructure(text)` (Object) converts a structure from a given string `text` into a molecular structure.
- `exportStructure(molecule, path, type, width, height)` (Boolean) exports a given `molecule` to the `path` into the given `type`. Valid types are: “mol”, “cml1”, “cml2”, “svg”, “bmp”, “jpg” and “png”. If you use one of the latter three types, the optional parameters `width` and `height` are used to determine the size in pixels of the resulting image. If no width or height is provided, 1500 is taken as a default.
- `calculateFormulaFromStructure(molecule)` (String) calculates the molecular structure from the given `molecule`-object.
- `calculateMolecularWeightFromFormula(formula)` (Number) calculates the molecular weight from the given `formula`.
- `calculateMonoisotopicMassFromFormula(formula)` (Number) calculates the monoisotopic mass from the given `formula`.
- `calculateRetentionIndex(rt, mix)` (Number) calculates the retention index from the given retention time `rt` and the alkane mix provided through `mix`.
- `logInsertHtml(text)` inserts the string `text`, which is interpreted as html, into the log window. The log window gets visible upon inserting text for the first time.
- `logInsertPlainText(text)` inserts the string `text`, which is interpreted as plain text, into the log window. The log window gets visible upon inserting text for the first time.
- `logClear()` clears the log.
- `getFloat(title, label, value, minValue, maxValue, decimals)` (Number) shows a dialog with a `title` and the text label where the user is able to enter a floating point number. The remaining parameters are optional. `value` contains the default value that the dialog is set to, `minValue` and `maxValue` are the minimal and maximal values, respectively (default: -2147483647 and 2147483647) and `decimals` is the number of decimals the value may have. The object `ok` (see section 5.1) is set to true if the user presses ok and false, if cancel is pressed.
- `getInteger(title, label, value, minValue, maxValue, step)` (Number) show a dialog with a `title` and the text `label` where the user is able to enter a integer. The remaining parameters are optional. `value` contains the default value that the dialog is set to, `minValue` and `maxValue` are the minimal and maximal values, respectively (default: -2147483647 and 2147483647) and `step` is the amount the number changes when the user presses the arrow buttons. The object `ok`

(see section 5.1) is set to true if the user presses ok and false, if cancel is pressed.

- `getItem(title, label, list, current, editable)` (String) shows a dialog with a `title` and the text `label` where the user is able to choose one of several options given by the array of strings `list`. The index of the default item can be given through the `current`-parameter. If `editable` is true, the user can enter their own text. If `editable` is false, the user may only select one of the items. The object `ok` (see section 5.1) is set to true if the user presses ok and false, if cancel is pressed.
- `getText(title, label, text)` (String) shows a dialog with a `title` and the text `label` where the user is able to enter some text. The default text can be given though the optional parameter `text`. The object `ok` (see section 5.1) is set to true if the user presses ok and false, if cancel is pressed.
- `progressBarShow()` shows the progress bar in the lower part of the application.
- `progressBarHide()` hides the progress bar in the lower part of the application.
- `progressBarSet(value)` sets the value of the progress bar to `value`. A number between 0 and 100 is accepted, where the progress bar is fully filled when set to 100.