

# Package ‘ChemmineOB’

November 27, 2024

**Type** Package

**Title** R interface to a subset of OpenBabel functionalities

**Date** 2024-8-02

**Version** 1.45.0

**Author** Kevin Horan, Thomas Girke

**Maintainer** Thomas Girke <thomas.girke@ucr.edu>

**Suggests** ChemmineR, BiocStyle, knitr, knitrBootstrap, BiocManager, rmarkdown, RUnit, codetools

**Imports** BiocGenerics, zlibbioc, Rcpp (>= 0.11.0)

**Description** ChemmineOB provides an R interface to a subset of cheminformatics functionalities implemented by the OpenBabel C++ project. OpenBabel is an open source cheminformatics toolbox that includes utilities for structure format interconversions, descriptor calculations, compound similarity searching and more. ChemmineOB aims to make a subset of these utilities available from within R. For non-developers, ChemmineOB is primarily intended to be used from ChemmineR as an add-on package rather than used directly.

**License** Artistic-2.0

**Depends** R (>= 2.15.1), methods

**SystemRequirements** OpenBabel (>= 3.0.0) with headers (<http://openbabel.org>). Eigen3 with headers.

**Enhances** ChemmineR (>= 2.13.0)

**URL** <https://github.com/girke-lab/ChemmineOB>

**biocViews** Cheminformatics, BiomedicalInformatics, Pharmacogenetics, Pharmacogenomics, MicrotitrePlateAssay, CellBasedAssays, Visualization, Infrastructure, DataImport, Clustering, Proteomics, Metabolomics

**VignetteBuilder** knitr

**LinkingTo** BH, Rcpp, zlibbioc

**git\_url** <https://git.bioconductor.org/packages/ChemmineOB>

**git\_branch** devel

**git\_last\_commit** e269adf

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2024-11-27

## Contents

canonicalNumbering_OB . . . . .	2
convertFormat . . . . .	3
convertFormatFile . . . . .	4
convertToImage . . . . .	6
exactMass_OB . . . . .	7
fingerprint_OB . . . . .	8
forEachMol . . . . .	9
OB-classes . . . . .	10
prop_OB . . . . .	10
smartsSearch_OB . . . . .	11
<b>Index</b>	<b>12</b>

---

canonicalNumbering\_OB *Canonical Numbering OB*

---

## Description

Computes a re-arrangement required to transform the atom numbering of the given compound into the canonical atom numbering. This function uses the `OBGraphSym` and `CanonicalLabels` classes of Open Babel to compute the re-arrangement.

## Usage

```
canonicalNumbering_OB(obmolRefs)
```

## Arguments

`obmolRefs` A list of `OBMol` references ( of class `'_p_OpenBabel__OBMol'`) representing the compounds.

## Value

A list of vectors of index values. Each item in the list corresponds to one of the given compounds. The values of a list item are the re-arrangement of the atoms. For example, if the value in item 1, column 1 is 25, that means that atom number 1 in the original compound should become atom number 25 in the canonical version of that compound.

## Author(s)

Kevin Horan

**References**

[http://openbabel.org/api/2.3/canonical\\_code\\_algorithm.shtml](http://openbabel.org/api/2.3/canonical_code_algorithm.shtml)

**See Also**

[convertFormat](#) to return a new compound in the canonical format.

**Examples**

```
## Not run:
library(ChemmineR)
data(sdfsampl.e)
labels = canonicalNumbering_OB(obmol(sdfsampl.e[[1]]))

## End(Not run)
```

---

convertFormat

*Convert Formats*

---

**Description**

Converts compound data from one format to another. Some formats are not supported on windows. This currently includes "INCHI" and "INCHIKEY".

**Usage**

```
convertFormat(from, to, source, options=data.frame(names="gen2D", args=""))
```

**Arguments**

from	The format that source is in. This should be a string supported by OpenBabel.
to	The format to convert source to.
source	The initial compound format, as a string. The format of the string should be identical to the file format of the same name. Tabs and newlines may be represented with \t and \n, respectively.
options	This is a data frame listing OpenBabel Options that should be applied while converting each compound. The "names" column should be the name of each option and the "args" column should be the arguments to the corresponding option. For example, the default value shown above will cause 2D coordinates to be generated from the input compound and saved in the output compound, assuming the output format supports it.  A full list of available options can be found by executing "obabel -L ops" at the command line. The current list is:  Oxout <file.xxx> Additional file output AddInIndex Append input index to title AddPolarH Adds hydrogen to polar atoms only canonical Canonicalize the atom order energy ForceField Energy Evaluation (not displayed in GUI) fillUC

<param> Fill the unit cell (strict or keepconnect) gen2D Generate 2D coordinates gen3D Generate 3D coordinates minimize ForceField Energy Minimization (not displayed in GUI) partialcharge <method> Calculate partial charges by specified method readconformer Adjacent conformers combined into a single molecule s Isomorphism filter(-s, -v options replacement)(not displayed in GUI) sort <desc> Sort by descriptor(~desc for reverse) unique [param] remove duplicates by descriptor;default inchi v Isomorphism filter(-s, -v options replacement)(not displayed in GUI)

### Value

Returns the compound given in source in the format specified by to.

### Author(s)

Kevin Horan

### References

OpenBabel <http://openbabel.org>

### See Also

[convertFormatFile](#)

### Examples

```
#create an SDF from a SMILES string and put the atoms in the canonical order
## Not run:
sdfStr = convertFormat("SMI", "SDF", "CC(=O)OC1=CC=CC=C1C(=O)O\\ttest_name",
                      options=data.frame(names="canonical", args=""))

## End(Not run)
```

---

convertFormatFile      *Convert Format of Files*

---

### Description

Convert a file from one format to another. Some formats are not supported on windows. This currently includes "INCHI" and "INCHIKEY".

### Usage

```
convertFormatFile(from, to, fromFile, toFile, options=data.frame(names="gen2D", args=""))
```

**Arguments**

from	The format that fromFile is in. This should be a string supported by OpenBabel.
to	The format to convert toFile to.
fromFile	The name of the file to be converted
toFile	The name of the new file to be created or overwritten
options	<p>This is a data frame listing OpenBabel Options that should be applied while converting each compound. The "names" column should be the name of each option and the "args" column should be the arguments to the corresponding option. For example, the default value shown above will cause 2D coordinates to be generated from the input compound and saved in the output compound, assuming the output format supports it.</p> <p>A full list of available options can be found by executing "obabel -L ops" at the command line. The current list is:</p> <p>Oxout &lt;file.xxx&gt; Additional file output AddInIndex Append input index to title AddPolarH Adds hydrogen to polar atoms only canonical Canonicalize the atom order energy ForceField Energy Evaluation (not displayed in GUI) fillUC &lt;param&gt; Fill the unit cell (strict or keepconnect) gen2D Generate 2D coordinates gen3D Generate 3D coordinates minimize ForceField Energy Minimization (not displayed in GUI) partialcharge &lt;method&gt; Calculate partial charges by specified method readconformer Adjacent conformers combined into a single molecule s Isomorphism filter(-s, -v options replacement)(not displayed in GUI) sort &lt;desc&gt; Sort by descriptor(~desc for reverse) unique [param] remove duplicates by descriptor;default inchi v Isomorphism filter(-s, -v options replacement)(not displayed in GUI)</p>

**Value**

No value is returned. toFile will be created with the compound in the new format.

**Author(s)**

Kevin Horan

**References**

OpenBabel <http://openbabel.org>

**See Also**

[convertFormat](#)

**Examples**

```
## Not run:  
convertFormatFile("SMI", "SDF", "test.smiles", "test.sdf")  
  
## End(Not run)
```

---

convertToImage	<i>Convert To Image</i>
----------------	-------------------------

---

### Description

Converts compound data to an image format

### Usage

```
convertToImage(from,to,source,toFile,options=data.frame(names="gen2D",args=""),
  out_options=data.frame(names=c("p"),args=c(300)))
```

### Arguments

from	The format that source is in. This should be a string supported by OpenBabel.
to	The image format to convert source to. This should be either "PNG", "POV", or "SVG".
source	The initial compound format, as a string. The format of the string should be identical to the file format of the same name. Tabs and newlines may be represented with \t and \n, respectively.
toFile	The name of the output file.
options	<p>This is a data frame listing OpenBabel Options that should be applied while converting each compound. The "names" column should be the name of each option and the "args" column should be the arguments to the corresponding option. For example, the default value shown above will cause 2D coordinates to be generated from the input compound and saved in the output compound, assuming the output format supports it.</p> <p>A full list of available options can be found by executing "obabel -L ops" at the command line. The current list is:</p> <p>0xout &lt;file.xxx&gt; Additional file output AddInIndex Append input index to title AddPolarH Adds hydrogen to polar atoms only canonical Canonicalize the atom order energy ForceField Energy Evaluation (not displayed in GUI) fillUC &lt;param&gt; Fill the unit cell (strict or keepconnect) gen2D Generate 2D coordinates gen3D Generate 3D coordinates minimize ForceField Energy Minimization (not displayed in GUI) partialcharge &lt;method&gt; Calculate partial charges by specified method readconformer Adjacent conformers combined into a single molecule s Isomorphism filter(-s, -v options replacement)(not displayed in GUI) sort &lt;desc&gt; Sort by descriptor(~desc for reverse) unique [param] remove duplicates by descriptor;default inchi v Isomorphism filter(-s, -v options replacement)(not displayed in GUI)</p>
out_options	<p>Similar to the options argument, but these options apply to the output format. For the "PNG" format, possible options are:</p> <p>"p" for the number of pixels on the square output file.</p> <p>For the "SVG" format, see the OpenBabel docs for a full list (<a href="https://openbabel.org/docs/dev/FileFormats/">https://openbabel.org/docs/dev/FileFormats/</a>)</p>

**Value**

Generates a square image with the given compounds arranged in a grid format. The width (and height, which must be the same) can be given using the out\_options with argument "p". Writes the resulting image to a file specified by to\_file.

**Author(s)**

Kevin Horan

**References**

OpenBabel <http://openbabel.org>

**Examples**

```
# Convert the given smiles string to a PNG image with size 500x500.
## Not run:
sdfStr = convertToImage("SMI", "PNG", "CC(=O)OC1=CC=CC=C1C(=O)O\ttest_name", "out_image.png",
                        out_options=data.frame(names="p", args="500"))

## End(Not run)
```

---

exactMass\_OB

*Exact Mass (Monoisotopic Mass)*

---

**Description**

Computes the exact mass of each compound given.

**Usage**

```
exactMass_OB(obmolRefs)
```

**Arguments**

obmolRefs      A list of OBMol references ( of class '\_p\_OpenBabel\_\_OBMol') representing the compounds.

**Value**

A vector of mass values.

**Author(s)**

Kevin Horan

### Examples

```
## Not run:
molRefs = foreachMol("SMILES", "C1CCCCC1\\ttest-compound-name", identity)
exactMass_OB(molRefs)

## End(Not run)
```

---

fingerprint\_OB                    *Fingerprints from OpenBabel*

---

### Description

Generates fingerprints using OpenBabel. The compound format can be specified as anything supported by OpenBabel. The fingerprint name can also be specified.

### Usage

```
fingerprint_OB(obmolRefs, fingerprintName, reverse=FALSE)
```

### Arguments

obmolRefs	A list of OBMol references ( of class <code>'_p_OpenBabel__OBMol'</code> ) representing the molecules to compute properites for. If you have your molecules in string format, you can create a list of OBMol references using the <code>foreachMol</code> function, see the example.
fingerprintName	The name of the fingerprint to generate. A list of available names can be found with <code>"obabel -L fingerprints"</code> . Currently that list is: <code>"FP2"</code> , <code>"FP3"</code> , <code>"FP4"</code> , and <code>"MACCS"</code> . Currently <code>"MACCS"</code> is not supported on windows.
reverse	Reverse the order of bits in each fingerprint. The fingerprints returned by this function are in reverse order to those returned by <code>obabel</code> when using the same fingerprint calculation. As long as all fingerprints being compared came from the same source (ie, all from <code>obabel</code> , or all from <code>ChemmineOB</code> ) the results should be correct. But if you need to mix fingerprints from different sources, care muse be taken to make sure they are all in the same direction. To make <code>ChemmineOB</code> match <code>obabel</code> output, set <code>reverse=TRUE</code> . We cannot set this to <code>TRUE</code> by default at this point as that could break code written prior to this change.

### Value

A matrix of binary values is returned. There is a row for each compound. The length of a row is determined by the fingerprint specified.

### Author(s)

Kevin Horan



**Examples**

```
## Not run:
molRefs = forEachMol("SMILES", "C1CCCCC1\test-compound-name", identity)
fingerprint_OB(molRefs, "FP3")

## End(Not run)
```

---

forEachMol

*For Each Mol*

---

**Description**

Reads in molecules from the given string in the given format and calls function *f* on each molecule. The results are then combined using the reduce function, if given.

**Usage**

```
forEachMol(inFormat, inString, f, reduce)
```

**Arguments**

<code>inFormat</code>	Format of string in source. This can be any OpenBabel format such as "SDF" or "SMILES". A full list can be found by executing "obabel -L formats".
<code>inString</code>	The compounds to generate fingerprints for. The format should be exactly what would be in a file of the same format. Newlines can be represented with "\n".
<code>f</code>	A function taking one OBMol reference and possibly returning a result.
<code>reduce</code>	This function will be passed to the Reduce function along with the results of all the <i>f</i> calls. This can be used to combine the results.

**Value**

The result will be a List of return values from the *f* function if not reduce function was given. Otherwise it will be the result of the reduce function applied to the results of the *f* function.

**Author(s)**

Kevin Horan

**Examples**

```
## Not run:
molRefs = forEachMol("SMILES", "C1CCCCC1\test-compound-name",
                    identity, c)

## End(Not run)
```

---

 OB-classes

*Classes from OB*


---

### Description

These are methods generated by SWIG for R. These should not generally be used outside of ChemmineOB, they are listed here to quite some warnings.

---

 prop\_OB

*Properties from OpenBabel*


---

### Description

Generates the following descriptors: "cansmi", "cansmiNS", "formula", "HBA1", "HBA2", "HBD", "InChI", "InChIKey", "logP", "MR", "MW", "nF", "title", "TPSA".

### Usage

```
prop_OB(obmolRefs)
```

### Arguments

obmolRefs      A list of OBMol references ( of class '\_p\_OpenBabel\_\_OBMol') representing the molecules to compute properites for. If you have your molecules in string format, you can create a list of OBMol references using the forEachMol function, see the example.

### Value

Returns a data frame with the following OpenBabel descriptors: "cansmi", "cansmiNS", "formula", "HBA1", "HBA2", "HBD", "InChI", "logP", "MR", "MW", "nF", "title", "TPSA". The "InChI" descriptor is not supported on windows currently, so that column will always be blank.

### Author(s)

Kevin Horan

### Examples

```
## Not run: # remove when ubuntu 16.04 bug fixed
molRefs = forEachMol("SMILES", "C1CCCC1\\ttest-compound-name", identity)
prop_OB(molRefs)

## End(Not run)
```

---

smartsSearch_OB	<i>SMARTS Search</i>
-----------------	----------------------

---

**Description**

Returns the number of matches found for each compound given.

**Usage**

```
smartsSearch_OB(obmolRefs, smartsPattern, uniqueMatches = TRUE)
```

**Arguments**

obmolRefs	A list of OBMol references ( of class '_p_OpenBabel__OBMol') representing the molecules to compute properites for. If you have your molecules in string format, you can create a list of OBMol references using the forEachMol function, see the example.
smartsPattern	Any valid SMARTS pattern.
uniqueMatches	Should only unique matches be counted?

**Value**

A vector of counts.

**Author(s)**

Kevin Horan

**Examples**

```
## Not run:  
molRefs = forEachMol("SMILES", "C1CCCC1\\ttest-compound-name", identity)  
smartsSearch_OB(molRefs, "[CH3X4]")  
  
## End(Not run)
```

# Index

\* **canonical**  
    canonicalNumbering\_OB, 2

\* **exact**  
    exactMass\_OB, 7

\* **mass**  
    exactMass\_OB, 7

\* **monoisotopic**  
    exactMass\_OB, 7

\* **morgan**  
    canonicalNumbering\_OB, 2

[, ExternalReference-method  
    (OB-classes), 10

[<-, ExternalReference-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_DoubleType, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_FptIndexHeader, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBAtom, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBAtomAtomIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBAtomBondIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBBond, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBFFCalculation2, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBFFCalculation3, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBFFCalculation4, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBFFConstraint, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBFFParameter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBInternalCoord, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolAtomBFSIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolAtomDFSIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolAtomIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolBondBFSIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolBondIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBMolRingIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBResidueAtomIter, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBRing, character-method  
    (OB-classes), 10

[[<-, \_p\_OpenBabel\_\_OBUnitCell, character-method  
    (OB-classes), 10

[[<-, \_p\_std\_\_pairT\_unsigned\_int\_unsigned\_int\_t, character-method  
    (OB-classes), 10

    \$, \_p\_OpenBabel\_\_AliasData-method  
    \$, \_p\_OpenBabel\_\_DoubleType-method  
    \$, \_p\_OpenBabel\_\_FastSearch-method  
    \$, \_p\_OpenBabel\_\_FastSearchIndexer-method  
    \$, \_p\_OpenBabel\_\_FptIndexHeader-method  
    \$, \_p\_OpenBabel\_\_OBAngle-method  
    \$, \_p\_OpenBabel\_\_OBAngleData-method  
    \$, \_p\_OpenBabel\_\_OBARomaticTyper-method  
    \$, \_p\_OpenBabel\_\_OBAtom-method  
    \$, \_p\_OpenBabel\_\_OBAtomAtomIter-method

- `$_p_OpenBabel__OBAtomBondIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBAtomClassData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBAtomTyper`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBBase`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBBitVec`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBBond`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBBuilder`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBChainsParser`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBChiralData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBCommentData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBConformerData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBConversion`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBDOSData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBDescriptor`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBElectronicTransitionData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBElement`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBElementTable`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBError`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBExternalBond`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBExternalBondData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFCalculation2`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFCalculation3`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFCalculation4`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFConstraint`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFConstraints`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFFParameter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFingerprint`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBForceField`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBFormat`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBGenericData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBGlobalDataBase`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBGridData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBInternalCoord`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBIsotopeTable`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMatrixData`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMessageHandler`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMol`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolAngleIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolAtomBFSIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolAtomDFSIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolAtomIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolBondBFSIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolBondIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolPairIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolRingIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBMolTorsionIter`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBOp`-method  
(OB-classes), 10
- `$_p_OpenBabel__OBOrbital`-method  
(OB-classes), 10

- `$_p_OpenBabel__OBOrbitalData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBPairData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBPlugin-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRTree-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRandom-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBResidue-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBResidueAtomIter-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBResidueData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBResidueIter-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRing-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRingData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRingSearch-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRingTyper-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBRotationData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSMatch-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSerialNums-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSetData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSmartsMatcher-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSmartsPattern-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSqrtTbl-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBStopwatch-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBSymmetryData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBTorsion-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBTorsionData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBTypeTable-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBUnitCell-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBVectorData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBVibrationData-method`  
 (OB-classes), 10
- `$_p_OpenBabel__OBVirtualBond-method`  
 (OB-classes), 10
- `$_p_OpenBabel__SpaceGroup-method`  
 (OB-classes), 10
- `$_p_OpenBabel__matrix3x3-method`  
 (OB-classes), 10
- `$_p_OpenBabel__transform3d-method`  
 (OB-classes), 10
- `$_p_OpenBabel__vector3-method`  
 (OB-classes), 10
- `$_p_std__pairT_unsigned_int_unsigned_int_t-method`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBBond_std__allocatorT_OpenBa`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBGenericData_p_std__allocato`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBInternalCoord_p_std__alloca`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBMol_std__allocatorT_OpenBab`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBResidue_std__allocatorT_Ope`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBRing_p_std__allocatorT_Open`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__OBRing_std__allocatorT_OpenBa`  
 (OB-classes), 10
- `$_p_std__vectorT_OpenBabel__vector3_std__allocatorT_OpenB`  
 (OB-classes), 10
- `$_p_std__vectorT_double_std__allocatorT_double_t_t-method`  
 (OB-classes), 10
- `$_p_std__vectorT_int_std__allocatorT_int_t_t-method`  
 (OB-classes), 10
- `$_p_std__vectorT_std__pairT_unsigned_int_unsigned_int_t_s`  
 (OB-classes), 10
- `$_p_std__vectorT_std__string_std__allocatorT_std__string_`  
 (OB-classes), 10
- `$_p_std__vectorT_std__vectorT_OpenBabel__vector3_std__all`  
 (OB-classes), 10
- `$_p_std__vectorT_std__vectorT_int_std__allocatorT_int_t_t`  
 (OB-classes), 10

`$_p_std__vectorT_std__vectorT_std__pairT_unsigned_int_t_std__allocatorT_std__pairT_unsigned_int_t` (OB-classes), 10  
`$_p_std__vectorT_unsigned_int_std__allocatorT_unsigned_int_t` (OB-classes), 10  
`$_p_stringp-method` (OB-classes), 10  
`$<-,_p_OpenBabel__DoubleType-method` (OB-classes), 10  
`$<-,_p_OpenBabel__FptIndexHeader-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBAtom-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBAtomAtomIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBAtomBondIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBBond-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBFFCalculation2-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBFFCalculation3-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBFFCalculation4-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBFFConstraint-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBFFParameter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBInternalCoord-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolAtomBFSIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolAtomDFSIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolAtomIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolBondBFSIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolBondIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBMolRingIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBResidueAtomIter-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBRing-method` (OB-classes), 10  
`$<-,_p_OpenBabel__OBUnitCell-method` (OB-classes), 10  
`$<-,_p_std__pairT_unsigned_int_unsigned_int_t-method` (OB-classes), 10  
`canonicalNumbering-OB-2`  
`convertFormat`, 3, 3, 5  
`convertFormatFile`, 4, 4  
`convertToImage`, 6  
`exactMass-OB`, 7  
`fingerprint-OB`, 8  
`forEachMol`, 9  
`length, SWIGArray-method` (OB-classes), 10  
`OB-classes`, 10  
`prop-OB`, 10  
`smartsSearch-OB`, 11