

# Reference Manual

Generated by Doxygen 1.6.3

Tue Feb 8 08:09:46 2011



# Contents

<b>1</b>	<b>TinkerCell C API</b>	<b>1</b>
<b>2</b>	<b>Module Index</b>	<b>3</b>
2.1	Modules . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>Module Documentation</b>	<b>7</b>
4.1	Basic operations . . . . .	7
4.1.1	Detailed Description . . . . .	9
4.1.2	Function Documentation . . . . .	9
4.1.2.1	tc_appendColumns . . . . .	9
4.1.2.2	tc_appendRows . . . . .	9
4.1.2.3	tc_createItemsArray . . . . .	9
4.1.2.4	tc_createMatrix . . . . .	9
4.1.2.5	tc_createStringsArray . . . . .	10
4.1.2.6	tc_createTable . . . . .	10
4.1.2.7	tc_deleteItemsArray . . . . .	10
4.1.2.8	tc_deleteMatrix . . . . .	10
4.1.2.9	tc_deleteStringsArray . . . . .	11
4.1.2.10	tc_deleteTable . . . . .	11
4.1.2.11	tc_getColumnName . . . . .	11
4.1.2.12	tc_getItem . . . . .	11
4.1.2.13	tc_getMatrixValue . . . . .	11
4.1.2.14	tc_getRowName . . . . .	12
4.1.2.15	tc_getString . . . . .	12
4.1.2.16	tc_getTableValue . . . . .	12
4.1.2.17	tc_printMatrixToFile . . . . .	12

4.1.2.18	<a href="#">tc_printOutMatrix</a>	13
4.1.2.19	<a href="#">tc_printOutTable</a>	13
4.1.2.20	<a href="#">tc_printTableToFile</a>	13
4.1.2.21	<a href="#">tc_setColumnName</a>	13
4.1.2.22	<a href="#">tc_setItem</a>	13
4.1.2.23	<a href="#">tc_setMatrixValue</a>	14
4.1.2.24	<a href="#">tc_setRowName</a>	14
4.1.2.25	<a href="#">tc_setString</a>	14
4.1.2.26	<a href="#">tc_setTableValue</a>	14
4.2	<a href="#">Appearance</a>	15
4.2.1	<a href="#">Detailed Description</a>	16
4.2.2	<a href="#">Function Documentation</a>	16
4.2.2.1	<a href="#">tc_changeArrowHead</a>	16
4.2.2.2	<a href="#">tc_changeNodeImage</a>	16
4.2.2.3	<a href="#">tc_getAngle</a>	16
4.2.2.4	<a href="#">tc_getColor</a>	16
4.2.2.5	<a href="#">tc_getHeight</a>	17
4.2.2.6	<a href="#">tc_getPos</a>	17
4.2.2.7	<a href="#">tc_getWidth</a>	17
4.2.2.8	<a href="#">tc_getX</a>	17
4.2.2.9	<a href="#">tc_getY</a>	18
4.2.2.10	<a href="#">tc_moveSelected</a>	18
4.2.2.11	<a href="#">tc_setAngle</a>	18
4.2.2.12	<a href="#">tc_setColor</a>	18
4.2.2.13	<a href="#">tc_setPos</a>	18
4.2.2.14	<a href="#">tc_setPosMulti</a>	19
4.2.2.15	<a href="#">tc_setSize</a>	19
4.3	<a href="#">Get items</a>	20
4.3.1	<a href="#">Detailed Description</a>	21
4.3.2	<a href="#">Function Documentation</a>	21
4.3.2.1	<a href="#">tc_alignParts</a>	21
4.3.2.2	<a href="#">tc_allItems</a>	21
4.3.2.3	<a href="#">tc_find</a>	21
4.3.2.4	<a href="#">tc_findItems</a>	22
4.3.2.5	<a href="#">tc_getChildren</a>	22
4.3.2.6	<a href="#">tc_getName</a>	22

4.3.2.7	<a href="#">tc_getNames</a>	22
4.3.2.8	<a href="#">tc_getParent</a>	23
4.3.2.9	<a href="#">tc_getUniqueName</a>	23
4.3.2.10	<a href="#">tc_getUniqueNames</a>	23
4.3.2.11	<a href="#">tc_itemsOfFamily</a>	23
4.3.2.12	<a href="#">tc_itemsOfFamilyFrom</a>	24
4.3.2.13	<a href="#">tc_partsDownstream</a>	24
4.3.2.14	<a href="#">tc_partsIn</a>	24
4.3.2.15	<a href="#">tc_partsUpstream</a>	24
4.3.2.16	<a href="#">tc_rename</a>	24
4.3.2.17	<a href="#">tc_select</a>	25
4.3.2.18	<a href="#">tc_selectedItems</a>	25
4.3.2.19	<a href="#">tc_setSequence</a>	25
4.4	<a href="#">Annotations</a>	26
4.4.1	<a href="#">Detailed Description</a>	26
4.4.2	<a href="#">Function Documentation</a>	27
4.4.2.1	<a href="#">tc_getAllTextNamed</a>	27
4.4.2.2	<a href="#">tc_getAnnotation</a>	27
4.4.2.3	<a href="#">tc_getFamily</a>	27
4.4.2.4	<a href="#">tc_getName</a>	27
4.4.2.5	<a href="#">tc_getNames</a>	27
4.4.2.6	<a href="#">tc_getTextAttribute</a>	28
4.4.2.7	<a href="#">tc_getUniqueName</a>	28
4.4.2.8	<a href="#">tc_getUniqueNames</a>	28
4.4.2.9	<a href="#">tc_isA</a>	28
4.4.2.10	<a href="#">tc_rename</a>	29
4.4.2.11	<a href="#">tc_setAnnotation</a>	29
4.4.2.12	<a href="#">tc_setSequence</a>	29
4.4.2.13	<a href="#">tc_setTextAttribute</a>	29
4.5	<a href="#">Input and Output</a>	30
4.5.1	<a href="#">Detailed Description</a>	32
4.5.2	<a href="#">Function Documentation</a>	32
4.5.2.1	<a href="#">tc_addInputWindowCheckbox</a>	32
4.5.2.2	<a href="#">tc_addInputWindowOptions</a>	32
4.5.2.3	<a href="#">tc_askQuestion</a>	32
4.5.2.4	<a href="#">tc_clear</a>	32

4.5.2.5	<a href="#">tc_createInputWindow</a>	33
4.5.2.6	<a href="#">tc_createInputWindowFromFile</a>	33
4.5.2.7	<a href="#">tc_createSliders</a>	33
4.5.2.8	<a href="#">tc_displayNumber</a>	33
4.5.2.9	<a href="#">tc_displayText</a>	34
4.5.2.10	<a href="#">tc_errorReport</a>	34
4.5.2.11	<a href="#">tc_getFilename</a>	34
4.5.2.12	<a href="#">tc_getNumber</a>	34
4.5.2.13	<a href="#">tc_getNumbers</a>	34
4.5.2.14	<a href="#">tc_getStringFromList</a>	35
4.5.2.15	<a href="#">tc_highlight</a>	35
4.5.2.16	<a href="#">tc_messageDialog</a>	35
4.5.2.17	<a href="#">tc_openFile</a>	35
4.5.2.18	<a href="#">tc_openNewWindow</a>	36
4.5.2.19	<a href="#">tc_print</a>	36
4.5.2.20	<a href="#">tc_printFile</a>	36
4.5.2.21	<a href="#">tc_printMatrix</a>	36
4.5.2.22	<a href="#">tc_saveToFile</a>	36
4.5.2.23	<a href="#">tc_screenHeight</a>	37
4.5.2.24	<a href="#">tc_screenshot</a>	37
4.5.2.25	<a href="#">tc_screenWidth</a>	37
4.5.2.26	<a href="#">tc_screenX</a>	37
4.5.2.27	<a href="#">tc_screenY</a>	37
4.5.2.28	<a href="#">tc_setDisplayLabelColor</a>	37
4.5.2.29	<a href="#">tc_zoom</a>	38
4.6	<a href="#">System information</a>	39
4.6.1	<a href="#">Detailed Description</a>	39
4.6.2	<a href="#">Function Documentation</a>	39
4.6.2.1	<a href="#">tc_appDir</a>	39
4.6.2.2	<a href="#">tc_homeDir</a>	39
4.6.2.3	<a href="#">tc_isLinux</a>	40
4.6.2.4	<a href="#">tc_isMac</a>	40
4.6.2.5	<a href="#">tc_isWindows</a>	40
4.7	<a href="#">Network data</a>	41
4.8	<a href="#">Graphing</a>	42
4.8.1	<a href="#">Detailed Description</a>	42

---

4.8.2	Function Documentation	42
4.8.2.1	tc_errorBars	42
4.8.2.2	tc_getPlotData	43
4.8.2.3	tc_gnuplot	43
4.8.2.4	tc_hist	43
4.8.2.5	tc_multiplot	43
4.8.2.6	tc_plot	43
4.8.2.7	tc_savePlot	44
4.8.2.8	tc_scatterplot	44
4.8.2.9	tc_surface	44
4.9	Modeling	45
4.9.1	Detailed Description	46
4.9.2	Function Documentation	46
4.9.2.1	tc_addEvent	46
4.9.2.2	tc_addForcingFunction	47
4.9.2.3	tc_getEventResponses	47
4.9.2.4	tc_getEventTriggers	47
4.9.2.5	tc_getFixedVariables	47
4.9.2.6	tc_getForcingFunctionAssignments	47
4.9.2.7	tc_getForcingFunctionNames	48
4.9.2.8	tc_getInitialValues	48
4.9.2.9	tc_getParameter	48
4.9.2.10	tc_getParameters	48
4.9.2.11	tc_getParametersAndFixedVariables	49
4.9.2.12	tc_getParametersExcept	49
4.9.2.13	tc_getParametersNamed	49
4.9.2.14	tc_getRate	49
4.9.2.15	tc_getRates	50
4.9.2.16	tc_getStoichiometry	50
4.9.2.17	tc_getStoichiometryFor	50
4.9.2.18	tc_setInitialValues	50
4.9.2.19	tc_setParameter	50
4.9.2.20	tc_setRate	51
4.9.2.21	tc_setRates	51
4.9.2.22	tc_setStoichiometry	51
4.9.2.23	tc_setStoichiometryFor	51

---

4.9.2.24	<code>tc_writeModel</code>	51
4.10	Connections	52
4.10.1	Detailed Description	53
4.10.2	Function Documentation	53
4.10.2.1	<code>tc_getCenterPointX</code>	53
4.10.2.2	<code>tc_getCenterPointY</code>	53
4.10.2.3	<code>tc_getConnectedNodes</code>	53
4.10.2.4	<code>tc_getConnectedNodesWithRole</code>	53
4.10.2.5	<code>tc_getConnections</code>	54
4.10.2.6	<code>tc_getConnectionsWithRole</code>	54
4.10.2.7	<code>tc_getControlPointX</code>	54
4.10.2.8	<code>tc_getControlPointY</code>	55
4.10.2.9	<code>tc_insertConnection</code>	55
4.10.2.10	<code>tc_setAllStraight</code>	55
4.10.2.11	<code>tc_setCenterPoint</code>	55
4.10.2.12	<code>tc_setControlPoint</code>	56
4.10.2.13	<code>tc_setLineWidth</code>	56
4.10.2.14	<code>tc_setStraight</code>	56
4.11	Import/Export	57
4.11.1	Detailed Description	57
4.11.2	Function Documentation	57
4.11.2.1	<code>tc_exportSBML</code>	57
4.11.2.2	<code>tc_importSBML</code>	57
4.12	Simulation	58
4.12.1	Detailed Description	59
4.12.2	Function Documentation	59
4.12.2.1	<code>tc_elementaryFluxModes</code>	59
4.12.2.2	<code>tc_getEigenvalues</code>	59
4.12.2.3	<code>tc_getJacobian</code>	59
4.12.2.4	<code>tc_getScaledConcentrationCC</code>	60
4.12.2.5	<code>tc_getScaledElasticities</code>	60
4.12.2.6	<code>tc_getScaledFluxCC</code>	60
4.12.2.7	<code>tc_getSteadyState</code>	60
4.12.2.8	<code>tc_getUnscaledConcentrationCC</code>	60
4.12.2.9	<code>tc_getUnscaledElasticities</code>	60
4.12.2.10	<code>tc_getUnscaledFluxCC</code>	61



4.12.2.11 tc_KMatrix . . . . .	61
4.12.2.12 tc_LMatrix . . . . .	61
4.12.2.13 tc_reducedStoichiometry . . . . .	61
4.12.2.14 tc_simulateDeterministic . . . . .	61
4.12.2.15 tc_simulateHybrid . . . . .	62
4.12.2.16 tc_simulateStochastic . . . . .	62
4.12.2.17 tc_simulateTauLeap . . . . .	62
4.12.2.18 tc_steadyStateScan . . . . .	62
4.12.2.19 tc_steadyStateScan2D . . . . .	63
4.13 Modules . . . . .	64
4.13.1 Detailed Description . . . . .	64
4.13.2 Function Documentation . . . . .	64
4.13.2.1 tc_listOfPossibleModels . . . . .	64
4.13.2.2 tc_substituteModel . . . . .	64
<b>5 Class Documentation</b>	<b>65</b>
5.1 tc_items Struct Reference . . . . .	65
5.1.1 Detailed Description . . . . .	65
5.2 tc_matrix Struct Reference . . . . .	66
5.2.1 Detailed Description . . . . .	66
5.3 tc_strings Struct Reference . . . . .	67
5.3.1 Detailed Description . . . . .	67
5.4 tc_table Struct Reference . . . . .	68
5.4.1 Detailed Description . . . . .	68



# Chapter 1

## TinkerCell C API

The TinkerCell C API is a collection of functions that allow C programs to directly interact with TinkerCell's visual interface. SWIG is used to extend this API to other languages, such as Python, Perl, R, etc. The functions provided in this API are converted to Signals, which are much slower than function calls. But they can be used to communicate between threads, which is the main reason why they are used in TinkerCell. The API uses **six main data structures**:

**item**: just a reference to a TinkerCell object. Items are represented as integers in Python and Octave and as long ints in C.

**string**: a string of characters used. Represented as const char\* in C.

**tc\_items** array of items

```
tc_items A = tc_allItems()
A.length
tc_getItem(A,3)

long x = tc_find("x")
tc_setItem(A,3,x)
tc_items A2 = tc_createItemsArray(10) //array of length 10
```

**tc\_strings**: array of strings

```
tc_items A = tc_allItems()
tc_strings S = tc_getNames( A )
S.length
tc_getString(S,3)
tc_setString(S,3,"hello")
tc_strings S2 = tc_createStringsArray(10) //array of length 10
```

**tc\_matrix**: Two dimensional array of reals with row and column names. The rownames and colnames fields are **tc\_strings** objects

```
long x = tc_find("x")
tc_matrix M = tc_getNumericalData( x, "Parameters" )
int r = M.rows
int c = M.cols
tc_getColumnName(M,2)
tc_setColumnName(M,2,"col2")
tc_getRowName(M,1)
tc_setRowName(M,1,"row1")
tc_getMatrixValue(M,2,3)
tc_setMatrixValue(M,2,3,0.5)

tc_matrix M2 = tc_createMatrix(5,4)
```

**tc\_table**: Two dimensional array of Strings with row and column names. The rownames and colnames fields are [tc\\_strings](#) objects

```
long x = tc_find("x")
tc_table S = tc_getTextData( x, "Text Attributes" )
S.rows
S.cols
tc_getString( S.rownames, 1)
tc_getString( S.colnames, 2)
tc_getTableValue(S,2,3)
tc_setTableValue(S,2,3,"hello")
tc_table S2 = tc_createTable(4,5)
```

# Chapter 2

## Module Index

### 2.1 Modules

Here is a list of all modules:

Basic operations . . . . .	7
Appearance . . . . .	15
Get items . . . . .	20
Annotations . . . . .	26
Input and Output . . . . .	30
System information . . . . .	39
Network data . . . . .	41
Graphing . . . . .	42
Modeling . . . . .	45
Connections . . . . .	52
Import/Export . . . . .	57
Simulation . . . . .	58
Modules . . . . .	64



# Chapter 3

## Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

- [tc\\_items](#) (An array of int objects with length information. Use tc\_getItem(M,i) to get the i-th item ) 65
- [tc\\_matrix](#) (A 2D table of doubles with row and column names. Use tc\_getMatrixValue(M,i,j) to get the i,j-th value in [tc\\_matrix](#) M ) . . . . . 66
- [tc\\_strings](#) (An array of strings with length information. Use tc\_getString(M,i) to get the i-th string ) 67
- [tc\\_table](#) (A 2D table of strings with row and column names. Use tc\_getTableValue(M,i,j) to get the i,j-th value in [tc\\_matrix](#) M ) . . . . . 68





# Chapter 4

## Module Documentation

### 4.1 Basic operations

basic functions for getting and setting matrices, arrays, tables, etc.

#### Functions

- TCAPIEXPORT `tc_matrix tc_createMatrix` (int rows, int cols)  
*Create a matrix with the given rows and columns.*
- TCAPIEXPORT `tc_table tc_createTable` (int rows, int cols)  
*Create a strings table with the given rows and columns.*
- TCAPIEXPORT `tc_strings tc_createStringsArray` (int len)  
*Create an array of strings.*
- TCAPIEXPORT `tc_items tc_createItemsArray` (int len)  
*Create an array of items.*
- TCAPIEXPORT double `tc_getMatrixValue` (`tc_matrix` M, int i, int j)  
*get i,jth value from a `tc_matrix`*
- TCAPIEXPORT void `tc_setMatrixValue` (`tc_matrix` M, int i, int j, double d)  
*set i,jth value of a `tc_matrix`*
- TCAPIEXPORT const char \* `tc_getRowName` (`tc_matrix` M, int i)  
*get ith row name from a `tc_matrix`*
- TCAPIEXPORT void `tc_setRowName` (`tc_matrix` M, int i, const char \*s)  
*set ith row name for a `tc_matrix`*
- TCAPIEXPORT const char \* `tc_getColumnName` (`tc_matrix` M, int j)  
*get jth column name of a `tc_matrix`*
- TCAPIEXPORT void `tc_setColumnName` (`tc_matrix` M, int j, const char \*s)

*set jth column name of a `tc_matrix`*

- TCAPIEXPORT const char \* `tc_getTableValue` (`tc_table` S, int i, int j)  
*get i,j-th string in a table*
- TCAPIEXPORT void `tc_setTableValue` (`tc_table` S, int i, int j, const char \*s)  
*set i,jth string in a table*
- TCAPIEXPORT const char \* `tc_getString` (`tc_strings` S, int i)  
*get ith string in array of strings*
- TCAPIEXPORT void `tc_setString` (`tc_strings` S, int i, const char \*c)  
*set ith string in array of strings*
- TCAPIEXPORT long `tc_getItem` (`tc_items` A, int i)  
*get ith long item in array of items*
- TCAPIEXPORT void `tc_setItem` (`tc_items` A, int i, long o)  
*set ith long item in array of items*
- TCAPIEXPORT void `tc_deleteMatrix` (`tc_matrix` M)  
*delete a matrix*
- TCAPIEXPORT void `tc_deleteTable` (`tc_table` M)  
*delete a strings table*
- TCAPIEXPORT void `tc_deleteItemsArray` (`tc_items` A)  
*delete an array of items*
- TCAPIEXPORT void `tc_deleteStringsArray` (`tc_strings` C)  
*delete an array of strings*
- TCAPIEXPORT `tc_matrix` `tc_appendColumns` (`tc_matrix` A, `tc_matrix` B)  
*combine two matrices by appending their columns. row size must be equal for both matrices*
- TCAPIEXPORT `tc_matrix` `tc_appendRows` (`tc_matrix` A, `tc_matrix` B)  
*combine two matrices by appending their row. column sizes must be equal for both matrices*
- TCAPIEXPORT void `tc_printMatrixToFile` (const char \*file, `tc_matrix` M)  
*print a matrix to file*
- TCAPIEXPORT void `tc_printOutMatrix` (`tc_matrix` M)  
*print a matrix to stdout*
- TCAPIEXPORT void `tc_printTableToFile` (const char \*file, `tc_table` M)  
*print a table to file*
- TCAPIEXPORT void `tc_printOutTable` (`tc_table` M)  
*print a table to stdout*

### 4.1.1 Detailed Description

basic functions for getting and setting matrices, arrays, tables, etc.

### 4.1.2 Function Documentation

#### 4.1.2.1 TCAPIEXPORT tc\_matrix tc\_appendColumns (tc\_matrix *A*, tc\_matrix *B*)

combine two matrices by appending their columns. row size must be equal for both matrices

##### Parameters

*tc\_matrix* first matrix

*tc\_matrix* fsecond matrix

##### Returns

*tc\_matrix* new combined matrix

#### 4.1.2.2 TCAPIEXPORT tc\_matrix tc\_appendRows (tc\_matrix *A*, tc\_matrix *B*)

combine two matrices by appending their row. column sizes must be equal for both matrices

##### Parameters

*tc\_matrix* first matrix

*tc\_matrix* fsecond matrix

##### Returns

*tc\_matrix* new combined matrix

#### 4.1.2.3 TCAPIEXPORT tc\_items tc\_createItemsArray (int *len*)

Create an array of items.

##### Parameters

*int* number of items

##### Returns

*tc\_items*

#### 4.1.2.4 TCAPIEXPORT tc\_matrix tc\_createMatrix (int *rows*, int *cols*)

Create a matrix with the given rows and columns.

##### Parameters

*int* number of rows

*int* number of columns

#### Returns

[tc\\_matrix](#)

#### 4.1.2.5 TCAPIEXPORT tc\_strings tc\_createStringsArray (int *len*)

Create an array of strings.

#### Parameters

*int* length

#### Returns

[tc\\_strings](#)

#### 4.1.2.6 TCAPIEXPORT tc\_table tc\_createTable (int *rows*, int *cols*)

Create a strings table with the given rows and columns.

#### Parameters

*int* number of rows

*int* number of columns

#### Returns

[tc\\_table](#)

#### 4.1.2.7 TCAPIEXPORT void tc\_deleteItemsArray (tc\_items *A*)

delete an array of items

#### Parameters

*&tc\_items* pointer to array

#### 4.1.2.8 TCAPIEXPORT void tc\_deleteMatrix (tc\_matrix *M*)

delete a matrix

#### Parameters

*&tc\_matrix* pointer to matrix

**4.1.2.9 TCAPIEXPORT void tc\_deleteStringsArray (tc\_strings C)**

delete an array of strings

**Parameters**

*&tc\_strings* pointer to array

**4.1.2.10 TCAPIEXPORT void tc\_deleteTable (tc\_table M)**

delete a strings table

**Parameters**

*&tc\_table* pointer to table

**4.1.2.11 TCAPIEXPORT const char\* tc\_getColumnName (tc\_matrix M, int j)**

get jth column name of a [tc\\_matrix](#)

**Parameters**

[tc\\_matrix](#) matrix

*int* column

**Returns**

string column name

**4.1.2.12 TCAPIEXPORT long tc\_getItem (tc\_items A, int i)**

get ith long item in array of items

**Parameters**

[tc\\_items](#) array

*int* index

**Returns**

long value

**4.1.2.13 TCAPIEXPORT double tc\_getMatrixValue (tc\_matrix M, int i, int j)**

get i,jth value from a [tc\\_matrix](#)

**Parameters**

[tc\\_matrix](#) matrix

*int* row

*int* column

**Returns**

double value at the given row, column

**4.1.2.14 TCAPIEXPORT const char\* tc\_getRowName (tc\_matrix *M*, int *i*)**

get ith row name from a [tc\\_matrix](#)

**Parameters**

[tc\\_matrix](#) matrix

*int* row

**Returns**

string row name

**4.1.2.15 TCAPIEXPORT const char\* tc\_getString (tc\_strings *S*, int *i*)**

get ith string in array of strings

**Parameters**

[tc\\_strings](#) array

*int* index

**Returns**

string value

**4.1.2.16 TCAPIEXPORT const char\* tc\_getTableValue (tc\_table *S*, int *i*, int *j*)**

get i,j-th string in a table

**Parameters**

[tc\\_table](#) table

*int* row

*int* column

**Returns**

string value at row,column

**4.1.2.17 TCAPIEXPORT void tc\_printMatrixToFile (const char \**file*, tc\_matrix *M*)**

print a matrix to file

**Parameters**

*char\** file name

[tc\\_matrix](#)

**4.1.2.18 TCAPIEXPORT void tc\_printOutMatrix (tc\_matrix *M*)**

print a matrix to stdout

**Parameters**

*char\** file name

*tc\_matrix*

**4.1.2.19 TCAPIEXPORT void tc\_printOutTable (tc\_table *M*)**

print a table to stdout

**Parameters**

*tc\_table*

**4.1.2.20 TCAPIEXPORT void tc\_printTableToFile (const char \**file*, tc\_table *M*)**

print a table to file

**Parameters**

*char\** file name

*tc\_table*

**4.1.2.21 TCAPIEXPORT void tc\_setColumnName (tc\_matrix *M*, int *j*, const char \**s*)**

set *j*th column name of a *tc\_matrix*

**Parameters**

*tc\_matrix* matrix

*int* column

*string* column name

**4.1.2.22 TCAPIEXPORT void tc\_setItem (tc\_items *A*, int *i*, long *o*)**

set *i*th long item in array of items

**Parameters**

*tc\_items* array

*int* index

*long* value

**4.1.2.23 TCAPIEXPORT void tc\_setMatrixValue (tc\_matrix *M*, int *i*, int *j*, double *d*)**

set *i,j*th value of a [tc\\_matrix](#)

**Parameters**

[tc\\_matrix](#) matrix  
*int* row  
*int* column  
*double* value at the given row, column

**4.1.2.24 TCAPIEXPORT void tc\_setRowName (tc\_matrix *M*, int *i*, const char \* *s*)**

set *i*th row name for a [tc\\_matrix](#)

**Parameters**

[tc\\_matrix](#) matrix  
*int* row  
*string* row name

**4.1.2.25 TCAPIEXPORT void tc\_setString (tc\_strings *S*, int *i*, const char \* *c*)**

set *i*th string in array of strings

**Parameters**

[tc\\_strings](#) array  
*int* index  
*string* value

**4.1.2.26 TCAPIEXPORT void tc\_setTableValue (tc\_table *S*, int *i*, int *j*, const char \* *s*)**

set *i,j*th string in a table

**Parameters**

[tc\\_table](#) table  
*int* row  
*int* column  
*string* value at row,column



## 4.2 Appearance

get/set position, color, size, etc

### Functions

- TCAPIEXPORT double [tc\\_getY](#) (long item)  
*get the x location of an item*
- TCAPIEXPORT double [tc\\_getX](#) (long item)  
*get the y location of an item*
- TCAPIEXPORT [tc\\_matrix](#) [tc\\_getPos](#) ([tc\\_items](#) items)  
*get the y location of a list item. Output is a N x 2 matrix*
- TCAPIEXPORT void [tc\\_setPos](#) (long item, double x, double y)  
*set the x and y location of an item*
- TCAPIEXPORT void [tc\\_setPosMulti](#) ([tc\\_items](#) items, [tc\\_matrix](#) positions)  
*set the x and y location of a list of N items. Input a matrix of positions, with N rows and 2 columns (x,y)*
- TCAPIEXPORT void [tc\\_moveSelected](#) (double dx, double dy)  
*move all the selected items by a given amount*
- TCAPIEXPORT void [tc\\_setSize](#) (long item, double width, double height, int permanent)  
*Change the size of an item.*
- TCAPIEXPORT double [tc\\_getWidth](#) (long item)  
*get the width of an item*
- TCAPIEXPORT double [tc\\_getHeight](#) (long item)  
*get the width of an item*
- TCAPIEXPORT void [tc\\_setAngle](#) (long item, double t, int permanent)  
*get the width of an item*
- TCAPIEXPORT double [tc\\_getAngle](#) (long item)  
*get the angle of an item*
- TCAPIEXPORT const char \* [tc\\_getColor](#) (long item)  
*get the color of the item*
- TCAPIEXPORT void [tc\\_setColor](#) (long item, const char \*name, int permanent)  
*set the color of the item and indicate whether or not the color is permanenet*
- TCAPIEXPORT void [tc\\_changeNodeImage](#) (long item, const char \*filename)  
*change the graphics file for drawing one of the nodes*
- TCAPIEXPORT void [tc\\_changeArrowHead](#) (long connection, const char \*filename)  
*change the graphics file for drawing the arrowheads for the given connection*

### 4.2.1 Detailed Description

get/set position, color, size, etc

### 4.2.2 Function Documentation

#### 4.2.2.1 TCAPIEXPORT void tc\_changeArrowHead (long *connection*, const char \* *filename*)

change the graphics file for drawing the arrowheads for the given connection

##### Parameters

*int* address of connection, e.g. obtained using tc\_find

*string* file name of the new graphics file

#### 4.2.2.2 TCAPIEXPORT void tc\_changeNodeImage (long *item*, const char \* *filename*)

change the graphics file for drawing one of the nodes

##### Parameters

*int* address of item, e.g. obtained using tc\_find

*string* file name of the new graphics file

#### 4.2.2.3 TCAPIEXPORT double tc\_getAngle (long *item*)

get the angle of an item

##### Parameters

*int* address of item, e.g. obtained using tc\_find

##### Returns

double angle

#### 4.2.2.4 TCAPIEXPORT const char \* tc\_getColor (long *item*)

get the color of the item

##### Parameters

*int* address of item, e.g. obtained using tc\_find

##### Returns

string Hex code for color

**4.2.2.5 TCAPIEXPORT double tc\_getHeight (long *item*)**

get the width of an item

**Parameters**

*int* address of item, e.g. obtained using tc\_find

**Returns**

double height

**4.2.2.6 TCAPIEXPORT tc\_matrix tc\_getPos (tc\_items *items*)**

get the y location of a list item. Output is a N x 2 matrix

**Parameters**

*tc\_items* addresses of items

**Returns**

*tc\_matrix* x,y positions of items

**4.2.2.7 TCAPIEXPORT double tc\_getWidth (long *item*)**

get the width of an item

**Parameters**

*int* address of item, e.g. obtained using tc\_find

**Returns**

double width

**4.2.2.8 TCAPIEXPORT double tc\_getX (long *item*)**

get the y location of an item

**Parameters**

*int* address of item

**Returns**

double y position

**4.2.2.9 TCAPIEXPORT double tc\_getY (long *item*)**

get the x location of an item

**Parameters**

*int* address of item

**Returns**

double x position

**4.2.2.10 TCAPIEXPORT void tc\_moveSelected (double *dx*, double *dy*)**

move all the selected items by a given amount

**Parameters**

*double* change in x

*double* change in y

**4.2.2.11 TCAPIEXPORT void tc\_setAngle (long *item*, double *t*, int *permanent*)**

get the width of an item

set the angle of an item

**Parameters**

*int* address of item, e.g. obtained using tc\_find

*double* angle

**4.2.2.12 TCAPIEXPORT void tc\_setColor (long *item*, const char \* *name*, int *permanent*)**

set the color of the item and indicate whether or not the color is permanent

set the rgb color of the item and indicate whether or not the color is permanent

**Parameters**

*int* address of item, e.g. obtained using tc\_find

*string* Hex code for color

*int* 0(temporary) or 1 (permanent color change)

**4.2.2.13 TCAPIEXPORT void tc\_setPos (long *item*, double *x*, double *y*)**

set the x and y location of an item

**Parameters**

*int* address of item

*double* x position

*double* y position

**4.2.2.14 TCAPIEXPORT void tc\_setPosMulti (tc\_items *items*, tc\_matrix *positions*)**

set the x and y location of a list of N items. Input a matrix of positions, with N rows and 2 columns (x,y)

**Parameters**

*tc\_items* addresses of items

*tc\_matrix* x,y positions

**4.2.2.15 TCAPIEXPORT void tc\_setSize (long *item*, double *width*, double *height*, int *permanent*)**

Change the size of an item.

**Parameters**

*int* address of item, e.g. obtained using tc\_find

*double* width

*double* height

*int* 0 (temporary size change) or 1 (permanent size change)

## 4.3 Get items

get selected items or items of a family

### Functions

- TCAPIEXPORT [tc\\_items tc\\_partsIn](#) (long o)  
*Get all DNA parts inside the given container or module.*
- TCAPIEXPORT [tc\\_items tc\\_partsUpstream](#) (long o)  
*Get all DNA parts upstream of the given part.*
- TCAPIEXPORT [tc\\_items tc\\_partsDownstream](#) (long o)  
*Get all DNA parts downstream of the given part.*
- TCAPIEXPORT void [tc\\_alignParts](#) ([tc\\_items](#) a)  
*Align the given DNA parts in the order given.*
- TCAPIEXPORT void [tc\\_setSequence](#) (long o, const char \*s)  
*Assign DNA sequence to a part.*
- TCAPIEXPORT [tc\\_items tc\\_allItems](#) ()  
*get all visible items*
- TCAPIEXPORT [tc\\_items tc\\_selectedItems](#) ()  
*get all selected items*
- TCAPIEXPORT [tc\\_items tc\\_itemsOfFamily](#) (const char \*family)  
*get all items of the given family items*
- TCAPIEXPORT [tc\\_items tc\\_itemsOfFamilyFrom](#) (const char \*family, [tc\\_items](#) itemsToSelect-From)  
*get subset of items that belong to the given family*
- TCAPIEXPORT long [tc\\_find](#) (const char \*fullname)  
*get the first item with the given name (full name)*
- TCAPIEXPORT [tc\\_items tc\\_findItems](#) ([tc\\_strings](#) names)  
*get all items with the given names (full names)*
- TCAPIEXPORT void [tc\\_select](#) (long item)  
*select an item*
- TCAPIEXPORT void [tc\\_deselect](#) ()  
*deselect all items*
- TCAPIEXPORT [tc\\_items tc\\_getChildren](#) (long o)  
*get child items of the given item*

- TCAPIEXPORT long [tc\\_getParent](#) (long o)  
*get parent item of the given item*
- TCAPIEXPORT const char \* [tc\\_getName](#) (long item)  
*get the name of an item*
- TCAPIEXPORT const char \* [tc\\_getUniqueName](#) (long item)  
*get the full name of an item*
- TCAPIEXPORT void [tc\\_rename](#) (long item, const char \*name)  
*set the name of an item (not full name)*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getNames](#) ([tc\\_items](#) items)  
*get the names of several items*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getUniqueNames](#) ([tc\\_items](#) items)  
*get the full names of several items*

### 4.3.1 Detailed Description

get selected items or items of a family

### 4.3.2 Function Documentation

#### 4.3.2.1 TCAPIEXPORT void [tc\\_alignParts](#) ([tc\\_items](#) a)

Align the given DNA parts in the order given.

##### Parameters

[tc\\_items](#) a list of items

#### 4.3.2.2 BEGIN\_C\_DECLS TCAPIEXPORT [tc\\_items](#) [tc\\_allItems](#) ()

get all visible items

##### Returns

[tc\\_items](#) list of all items in the network

#### 4.3.2.3 TCAPIEXPORT long [tc\\_find](#) (const char \* *name*)

get the first item with the given name (full name)

##### Parameters

*string* name of an item. use full name whenever possible

##### Returns

int address of item with the name

#### 4.3.2.4 TCAPIEXPORT tc\_items tc\_findItems (tc\_strings *names*)

get all items with the given names (full names)

##### Parameters

*tc\_string* names of one or more items

##### Returns

[tc\\_items](#) addresses of all the items. For nonexistent names, a 0 will be placed in the list

#### 4.3.2.5 TCAPIEXPORT tc\_items tc\_getChildren (long *o*)

get child items of the given item

##### Parameters

*int* address of item

##### Returns

[tc\\_items](#) list of child items

#### 4.3.2.6 TCAPIEXPORT const char\* tc\_getName (long *item*)

get the name of an item

##### Parameters

*int* address of the item

##### Returns

string name (not full name)

#### 4.3.2.7 TCAPIEXPORT tc\_strings tc\_getNames (tc\_items *items*)

get the names of several items

##### Parameters

[tc\\_items](#) addresses of the items

##### Returns

*tc\_string* list of names (not full names)



**4.3.2.8 TCAPIEXPORT long tc\_getParent (long *o*)**

get parent item of the given item

**Parameters**

*int* address of item

**Returns**

int address of parent item (0 if no parent)

**4.3.2.9 TCAPIEXPORT const char\* tc\_getUniqueName (long *item*)**

get the full name of an item

**Parameters**

*int* address of the item

**Returns**

string full name of the item (always unique)

**4.3.2.10 TCAPIEXPORT tc\_strings tc\_getUniqueNames (tc\_items *items*)**

get the full names of several items

**Parameters**

*tc\_items* addresses of the items

**Returns**

tc\_string list of names (unique names)

**4.3.2.11 TCAPIEXPORT tc\_items tc\_itemsOfFamily (const char \**family*)**

get all items of the given family items

**Parameters**

*string* name of a type

**Returns**

*tc\_items* list of all items in network belonging under the given type

#### 4.3.2.12 TCAPIEXPORT tc\_items tc\_itemsOfFamilyFrom (const char \* *family*, tc\_items *itemsToSelectFrom*)

get subset of items that belong to the given family

##### Parameters

*string* name of a type

*tc\_items* list of items to select from

##### Returns

*tc\_items* list of all items in the list belonging under the given type

#### 4.3.2.13 TCAPIEXPORT tc\_items tc\_partsDownstream (long *o*)

Get all DNA parts downstream of the given part.

##### Parameters

*int* address of an item in the network

#### 4.3.2.14 BEGIN\_C\_DECLS TCAPIEXPORT tc\_items tc\_partsIn (long *o*)

Get all DNA parts inside the given container or module.

##### Parameters

*int* address of an item in the network

#### 4.3.2.15 TCAPIEXPORT tc\_items tc\_partsUpstream (long *o*)

Get all DNA parts upstream of the given part.

##### Parameters

*int* address of an item in the network

#### 4.3.2.16 TCAPIEXPORT void tc\_rename (long *item*, const char \* *name*)

set the name of an item (not full name)

##### Parameters

*int* address of item

##### Returns

string new name (not full name)

**4.3.2.17 TCAPIEXPORT void tc\_select (long *item*)**

select an item

**Parameters**

*int* address of the item

**4.3.2.18 TCAPIEXPORT tc\_items tc\_selectedItems ()**

get all selected items

**Returns**

[tc\\_items](#) list of all items currently selected by user

**4.3.2.19 TCAPIEXPORT void tc\_setSequence (long *o*, const char \* *s*)**

Assign DNA sequence to a part.

Align the given DNA parts in the order given.

## 4.4 Annotations

get annotation information about items

### Functions

- TCAPIEXPORT void [tc\\_setSequence](#) (long o, const char \*)  
*Align the given DNA parts in the order given.*
- TCAPIEXPORT const char \* [tc\\_getTextAttribute](#) (long item, const char \*attribute)  
*get the text attribute with the given name for the given item*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getAllTextNamed](#) ([tc\\_items](#) a, [tc\\_strings](#) attributes)  
*get all text Modeling with the given name for the given items*
- TCAPIEXPORT void [tc\\_setTextAttribute](#) (long item, const char \*attribute, const char \*value)  
*set text attribute for the given item*
- TCAPIEXPORT const char \* [tc\\_getName](#) (long item)  
*get the full name of an item*
- TCAPIEXPORT const char \* [tc\\_getUniqueName](#) (long item)  
*get the full name of an item*
- TCAPIEXPORT void [tc\\_rename](#) (long item, const char \*name)  
*set the name of an item (not full name)*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getNames](#) ([tc\\_items](#) items)  
*get the full names of several items*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getUniqueNames](#) ([tc\\_items](#) items)  
*get the full names of several items*
- TCAPIEXPORT const char \* [tc\\_getFamily](#) (long item)  
*get the family name of an item*
- TCAPIEXPORT int [tc\\_isA](#) (long item, const char \*family)  
*check is an item belongs in a family (or in a sub-family)*
- TCAPIEXPORT [tc\\_strings](#) [tc\\_getAnnotation](#) (long o)  
*get annotation for this item, i.e. family, author, descriptions, etc.*
- TCAPIEXPORT void [tc\\_setAnnotation](#) (long o, [tc\\_strings](#) annot)  
*set annotation for this item, i.e. family, author, descriptions, etc.*

### 4.4.1 Detailed Description

get annotation information about items

## 4.4.2 Function Documentation

### 4.4.2.1 TCAPIEXPORT tc\_strings tc\_getAllTextNamed (tc\_items *a*, tc\_strings *attributes*)

get all text Modeling with the given name for the given items

#### Parameters

*tc\_items* a list of items

*tc\_strings* a list of text attribute name that exists in each of the given items

#### Returns

*tc\_strings* the set of all text attribute values, one for each item in the input

### 4.4.2.2 BEGIN\_C\_DECLS TCAPIEXPORT tc\_strings tc\_getAnnotation (long *o*)

get annotation for this item, i.e. family, author, descriptions, etc.

#### Parameters

*int* address of item, e.g. obtained from tc\_find

### 4.4.2.3 TCAPIEXPORT const char \* tc\_getFamily (long *item*)

get the family name of an item

#### Parameters

*int* address of the item

#### Returns

string type of the item

### 4.4.2.4 TCAPIEXPORT const char\* tc\_getName (long *item*)

get the full name of an item

get the name of an item

### 4.4.2.5 TCAPIEXPORT tc\_strings tc\_getNames (tc\_items *items*)

get the full names of several items

get the names of several items

#### 4.4.2.6 TCAPIEXPORT const char\* tc\_getTextAttribute (long *item*, const char \* *attribute*)

get the text attribute with the given name for the given item

##### Parameters

*int* item in the model, e.g. something returned from tc\_find  
*string* name of the attribute

##### Returns

string attribute

#### 4.4.2.7 TCAPIEXPORT const char\* tc\_getUniqueName (long *item*)

get the full name of an item

##### Parameters

*int* address of the item

##### Returns

string full name of the item (always unique)

#### 4.4.2.8 TCAPIEXPORT tc\_strings tc\_getUniqueNames (tc\_items *items*)

get the full names of several items

##### Parameters

*tc\_items* addresses of the items

##### Returns

tc\_string list of names (unique names)

#### 4.4.2.9 TCAPIEXPORT int tc\_isA (long *item*, const char \* *family*)

check is an item belongs in a family (or in a sub-family)

##### Parameters

*int* address of the item  
*string* name of the family type

##### Returns

int 0(no) or 1(yes)

**4.4.2.10 TCAPIEXPORT void tc\_rename (long *item*, const char \* *name*)**

set the name of an item (not full name)

**Parameters**

*int* address of item

**Returns**

string new name (not full name)

**4.4.2.11 TCAPIEXPORT void tc\_setAnnotation (long *o*, tc\_strings *annot*)**

set annotation for this item, i.e. family, author, descriptions, etc.

**Parameters**

*int* address of item, e.g. obtained from tc\_find

*tc\_strings* pair of annotations, e.g. "name", "Don", "age", "93", "place", "Hawaii"

**4.4.2.12 TCAPIEXPORT void tc\_setSequence (long *o*, const char \* *s*)**

Align the given DNA parts in the order given.

**Parameters**

*tc\_items* a list of items

**4.4.2.13 TCAPIEXPORT void tc\_setTextAttribute (long *item*, const char \* *attribute*, const char \* *value*)**

set text attribute for the given item

**Parameters**

*int* item in model

*string* name of text attribute

## 4.5 Input and Output

display dialogs or get user inputs

### Functions

- TCAPIEXPORT void [tc\\_displayText](#) (long item, const char \*text)  
*displays the given text on the given item (the text is temporary)*
- TCAPIEXPORT void [tc\\_displayNumber](#) (long item, double number)  
*displays the given number on the given item (the text is temporary)*
- TCAPIEXPORT void [tc\\_setDisplayLabelColor](#) (const char \*a, const char \*b)  
*set the color for the number or text when using [tc\\_displayNumber](#) and [tc\\_displayText](#)*
- TCAPIEXPORT void [tc\\_highlight](#) (long item, const char \*color)  
*highlights an item (the highlight is temporary) with the given color (hex)*
- TCAPIEXPORT void [tc\\_print](#) (const char \*text)  
*show text in the output window.*
- TCAPIEXPORT void [tc\\_errorReport](#) (const char \*text)  
*show error text in the output window.*
- TCAPIEXPORT void [tc\\_printMatrix](#) ([tc\\_matrix](#) data)  
*show table in the output window.*
- TCAPIEXPORT void [tc\\_printFile](#) (const char \*filename)  
*show file contents in the output window.*
- TCAPIEXPORT void [tc\\_clear](#) ()  
*cleat the contents in the output window.*
- TCAPIEXPORT void [tc\\_createInputWindowFromFile](#) ([tc\\_matrix](#) input, const char \*filename, const char \*functionname, const char \*title)  
*create an input window that can call a dynamic library*
- TCAPIEXPORT void [tc\\_createInputWindow](#) ([tc\\_matrix](#) input, const char \*title, void(\*f)([tc\\_matrix](#)))  
*create an input window that can call a dynamic library*
- TCAPIEXPORT void [tc\\_addInputWindowOptions](#) (const char \*title, int i, int j, [tc\\_strings](#) options)  
*add options to an existing input window at the i,j-th cell. Options will appear in a list*
- TCAPIEXPORT void [tc\\_addInputWindowCheckbox](#) (const char \*title, int i, int j)  
*add a yes or no type of option to an existing input window at the i,j-th cell*
- TCAPIEXPORT void [tc\\_openNewWindow](#) (const char \*title)  
*open a new graphics window*



- TCAPIEXPORT void [tc\\_zoom](#) (double factor)  
*zoom by the given factor (0 - 1)*
- TCAPIEXPORT const char \* [tc\\_getStringDialog](#) (const char \*title)  
*get a text from the user (dialog)*
- TCAPIEXPORT const char \* [tc\\_getFilename](#) ()  
*get a file from the user (dialog)*
- TCAPIEXPORT int [tc\\_getStringFromList](#) (const char \*title, [tc\\_strings](#) list, const char \*selectedString)  
*get a text from the user (dialog) from a list of selections*
- TCAPIEXPORT double [tc\\_getNumber](#) (const char \*title)  
*get a number from the user (dialog)*
- TCAPIEXPORT void [tc\\_getNumbers](#) ([tc\\_strings](#) labels, double \*result)  
*get a list of numbers from the user (dialog) into the argument array*
- TCAPIEXPORT int [tc\\_askQuestion](#) (const char \*message)  
*display a dialog with a text and a yes and no button*
- TCAPIEXPORT void [tc\\_messageDialog](#) (const char \*message)  
*display a dialog with a text message and a close button*
- TCAPIEXPORT void [tc\\_openFile](#) (const char \*message)  
*open file*
- TCAPIEXPORT void [tc\\_saveToFile](#) (const char \*message)  
*save to file*
- TCAPIEXPORT void [tc\\_createSliders](#) ([tc\\_matrix](#) input, void(\*f)([tc\\_matrix](#)))  
*create a window with several sliders. when the sliders change, the given function will be called with the values in the sliders*
- TCAPIEXPORT void [tc\\_screenshot](#) (const char \*filename, int width, int height)  
*save screenshot in a file*
- TCAPIEXPORT void [tc\\_showProgress](#) (int progress)  
*show progress of current operation*
- TCAPIEXPORT int [tc\\_screenWidth](#) ()  
*get width of current canvas*
- TCAPIEXPORT int [tc\\_screenHeight](#) ()  
*get height of current canvas*
- TCAPIEXPORT int [tc\\_screenX](#) ()  
*get x position of current canvas*

- TCAPIEXPORT int `tc_screenY()`  
*get y position of current canvas*

### 4.5.1 Detailed Description

display dialogs or get user inputs

### 4.5.2 Function Documentation

#### 4.5.2.1 TCAPIEXPORT void `tc_addInputWindowCheckbox (const char * title, int i, int j)`

add a yes or no type of option to an existing input window at the i,j-th cell

##### Parameters

- int* row number
- int* column number

#### 4.5.2.2 TCAPIEXPORT void `tc_addInputWindowOptions (const char * title, int i, int j, tc_strings options)`

add options to an existing input window at the i,j-th cell. Options will appear in a list

##### Parameters

- string* name of an input window that was just created
- int* row number
- int* column number
- tc\_string* place these options (drop-down meny) at the (row,column) location of the table

#### 4.5.2.3 TCAPIEXPORT int `tc_askQuestion (const char * message)`

display a dialog with a text and a yes and no button

##### Parameters

- const char\** displayed message or question
- string* displayed message or question

#### 4.5.2.4 TCAPIEXPORT void `tc_clear ()`

cleat the contents in the output window.

cleat the contents in the output window

**4.5.2.5 TCAPIEXPORT void tc\_createInputWindow (tc\_matrix *input*, const char \* *title*, void(\*) (tc\_matrix) *f*)**

create an input window that can call a dynamic library

create an input window that will call a function

**Parameters**

*tc\_matrix* input window's arguments a default values

*string* name of this program

*void\** pointer to a 1-argument function that takes *tc\_matrix* argument

**4.5.2.6 TCAPIEXPORT void tc\_createInputWindowFromFile (tc\_matrix *input*, const char \* *filename*, const char \* *functionname*, const char \* *title*)**

create an input window that can call a dynamic library

create an input window that will run a function inside a C library

**Parameters**

*tc\_matrix* input window's arguments a default values

*string* C library file

*string* function inside the C library that takes *tc\_matrix* argument

*string* name of this program

**4.5.2.7 TCAPIEXPORT void tc\_createSliders (tc\_matrix *input*, void(\*) (tc\_matrix) *f*)**

create a window with several sliders. when the sliders change, the given function will be called with the values in the sliders

**Parameters**

*tc\_matrix* names of variables and initial values for the sliders

*void\** callback function with *tc\_matrix* as the argument

**4.5.2.8 TCAPIEXPORT void tc\_displayNumber (long *item*, double *number*)**

displays the given number on the given item (the text is temporary)

**Parameters**

*int* address of item in model, e.g. obtained from *tc\_find*

*double* number to display

#### 4.5.2.9 TCAPIEXPORT void tc\_displayText (long *item*, const char \* *text*)

displays the given text on the given item (the text is temporary)

##### Parameters

*int* address of item  
*string* text to display

#### 4.5.2.10 TCAPIEXPORT void tc\_errorReport (const char \* *text*)

show error text in the output window.

show error text in the output window

##### Parameters

*string* error message

#### 4.5.2.11 TCAPIEXPORT const char \* tc\_getFilename ()

get a file from the user (dialog)

popup dialog asking user to select a file

##### Returns

string the filename selected by the user

#### 4.5.2.12 TCAPIEXPORT double tc\_getNumber (const char \* *title*)

get a number from the user (dialog)

popup dialog asking user for a number

##### Parameters

*string* text presented to the user

##### Returns

double user's response

#### 4.5.2.13 TCAPIEXPORT void tc\_getNumbers (tc\_strings *labels*, double \* *result*)

get a list of numbers from the user (dialog) into the argument array

popup dialog asking user for several numbers (with labels)

##### Parameters

*tc\_strings* labels for each number to get  
*double\** array that will store the results

**4.5.2.14 TCAPIEXPORT int tc\_getStringFromList (const char \* *title*, tc\_strings *list*, const char \* *selectedString*)**

get a text from the user (dialog) from a list of selections

popup dialog asking user to select one item from a list

**Parameters**

*string* title of dialog

*tc\_string* list of options

*string* the option that is selected by default

**Returns**

int index of the user's selection, -1 if canceled

**4.5.2.15 TCAPIEXPORT void tc\_highlight (long *item*, const char \* *color*)**

highlights an item (the highlight is temporary) with the given color (hex)

highlights an item (the highlight is temporary) with the given color

**Parameters**

*int* address of item in model, e.g. obtained from tc\_find

*string* HEX code for color

**4.5.2.16 TCAPIEXPORT void tc\_messageDialog (const char \* *message*)**

display a dialog with a text message and a close button

**Parameters**

*const* char\* displayed message

*string* displayed message

**4.5.2.17 TCAPIEXPORT void tc\_openFile (const char \* *message*)**

open file

open a file

**Parameters**

*const* char\* file

*string* file name

**4.5.2.18 TCAPIEXPORT void tc\_openNewWindow (const char \* *title*)**

open a new graphics window

**Parameters**

*string* title of the new window

**4.5.2.19 TCAPIEXPORT void tc\_print (const char \* *text*)**

show text in the output window.

show text in the output window

**Parameters**

*string* text message

**4.5.2.20 TCAPIEXPORT void tc\_printFile (const char \* *filename*)**

show file contents in the output window.

show file contents in the output window

**Parameters**

*string* file name

**4.5.2.21 TCAPIEXPORT void tc\_printMatrix (tc\_matrix *data*)**

show table in the output window.

show table in the output window

**Parameters**

*tc\_matrix* table

**4.5.2.22 TCAPIEXPORT void tc\_saveToFile (const char \* *message*)**

save to file

save current network

**Parameters**

*const* char\* file

*string* filename

**4.5.2.23 TCAPIEXPORT int tc\_screenHeight ()**

get height of current canvas

**Returns**

int height

**4.5.2.24 TCAPIEXPORT void tc\_screenshot (const char \**filename*, int *width*, int *height*)**

save screenshot in a file

**Parameters**

*string* filename (PNG)

*int* width of image

*int* height of image

**4.5.2.25 TCAPIEXPORT int tc\_screenWidth ()**

get width of current canvas

**Returns**

int width

**4.5.2.26 TCAPIEXPORT int tc\_screenX ()**

get x position of current canvas

**Returns**

int x

**4.5.2.27 TCAPIEXPORT int tc\_screenY ()**

get y position of current canvas

**Returns**

int y

**4.5.2.28 TCAPIEXPORT void tc\_setDisplayLabelColor (const char \**color1*, const char \**color2*)**

set the color for the number or text when using tc\_displayNumber and tc\_displayText

**Parameters**

*string* HEX code for text color

*string* HEX code for background color

**4.5.2.29 TCAPIEXPORT void tc\_zoom (double *factor*)**

zoom by the given factor (0 - 1)

**Parameters**

*double* zoom factor between 0 and 1



## 4.6 System information

get information about the OS and program directory

### Functions

- TCAPIEXPORT int `tc_isWindows ()`  
*is this running in MS windows?*
- TCAPIEXPORT int `tc_isMac ()`  
*is this running in a Mac?*
- TCAPIEXPORT int `tc_isLinux ()`  
*is this running in Linux?*
- TCAPIEXPORT const char \* `tc_appDir ()`  
*TinkerCell application folder.*
- TCAPIEXPORT const char \* `tc_homeDir ()`  
*TinkerCell home folder.*

### 4.6.1 Detailed Description

get information about the OS and program directory

### 4.6.2 Function Documentation

#### 4.6.2.1 TCAPIEXPORT const char \* `tc_appDir ()`

TinkerCell application folder.

#### Returns

string application folder path

#### 4.6.2.2 TCAPIEXPORT const char \* `tc_homeDir ()`

TinkerCell home folder.

#### Returns

string home folder path

**4.6.2.3 TCAPIEXPORT int tc\_isLinux ()**

is this running in Linux?

is this running in a Unix system (excluding Mac)?

**Returns**

0 (not Linux) or 1 (is Linux)

**4.6.2.4 TCAPIEXPORT int tc\_isMac ()**

is this running in a Mac?

**Returns**

0 (not Mac OS ) or 1 (is Mac OS)

**4.6.2.5 TCAPIEXPORT int tc\_isWindows ()**

is this running in MS windows?

**Returns**

0 (not windows OS ) or 1 (is windows OS)

## 4.7 Network data

get/set information about the individual items in the network

get/set information about the individual items in the network

## 4.8 Graphing

display graphs, save graphs, get graph values

### Functions

- TCAPIEXPORT void [tc\\_surface](#) ([tc\\_matrix](#) z, const char \*title)  
*plot 3D data. Input matrix has x,y on the first two columns and z on the third column*
- TCAPIEXPORT void [tc\\_plot](#) ([tc\\_matrix](#) data, const char \*title)  
*plot the data in the matrix (with headers) with the given x-axis and title*
- TCAPIEXPORT void [tc\\_scatterplot](#) ([tc\\_matrix](#) data, const char \*title)  
*plot the 2-columns in the matrix (with headers) as a scatter plot*
- TCAPIEXPORT void [tc\\_errorBars](#) ([tc\\_matrix](#) data, const char \*title)  
*plot the data in the matrix (with headers) with the given x-axis and title. For each column i, the i+1 and i+2 columns should contain the upper and lower bounds (errors).*
- TCAPIEXPORT void [tc\\_hist](#) ([tc\\_matrix](#) data, const char \*title)  
*plot histogram for each column of the given matrix with the given bin size.*
- TCAPIEXPORT void [tc\\_multiplot](#) (int rows, int cols)  
*enable multi-plot, i.e. multiple plots on one screen. specify the number of rows and columns for the layout.*
- TCAPIEXPORT [tc\\_matrix](#) [tc\\_getPlotData](#) (int whichPlot)  
*get the data that is currently in the plot window*
- TCAPIEXPORT void [tc\\_gnuplot](#) (const char \*s)  
*gnuplot*
- TCAPIEXPORT void [tc\\_savePlot](#) (const char \*filename)  
*save plot*

### 4.8.1 Detailed Description

display graphs, save graphs, get graph values

### 4.8.2 Function Documentation

#### 4.8.2.1 TCAPIEXPORT void [tc\\_errorBars](#) ([tc\\_matrix](#) data, const char \* title)

plot the data in the matrix (with headers) with the given x-axis and title. For each column i, the i+1 and i+2 columns should contain the upper and lower bounds (errors).

#### Parameters

[tc\\_matrix](#) data  
*string* title of plot

**4.8.2.2 TCAPIEXPORT tc\_matrix tc\_getPlotData (int *whichPlot*)**

get the data that is currently in the plot window

get the data in the plot window

**Parameters**

*int* index of the plot (if multiple plots are being displayed)

**Returns**

*tc\_matrix* data

**4.8.2.3 TCAPIEXPORT void tc\_gnuplot (const char \*)**

gnuplot

plot the specific script using gnuplot

**Parameters**

*string* gnuplot commands

**4.8.2.4 TCAPIEXPORT void tc\_hist (tc\_matrix *data*, const char \* *title*)**

plot histogram for each column of the given matrix with the given bin size.

**Parameters**

*tc\_matrix* data

*string* title of plot

**4.8.2.5 TCAPIEXPORT void tc\_multiplot (int *rows*, int *cols*)**

enable multi-plot, i.e. multiple plots on one screen. specify the number of rows and columns for the layout.

**Parameters**

*int* number of rows

*int* number of columns

**4.8.2.6 TCAPIEXPORT void tc\_plot (tc\_matrix *data*, const char \* *title*)**

plot the data in the matrix (with headers) with the given x-axis and title

**Parameters**

*tc\_matrix* data with first column being the x-axis

*string* title of plot

**4.8.2.7 TCAPIEXPORT void tc\_savePlot (const char \* *filename*)**

save plot

save the current plot as a PDF file

**Parameters**

*string* filename (PDF suffix)

**4.8.2.8 TCAPIEXPORT void tc\_scatterplot (tc\_matrix *data*, const char \* *title*)**

plot the 2-columns in the matrix (with headers) as a scatter plot

plot the data in the matrix (with headers) as a scatter plot

**Parameters**

*tc\_matrix* data with first column as x-axis

*string* title of plot

**4.8.2.9 BEGIN\_C\_DECLS TCAPIEXPORT void tc\_surface (tc\_matrix *z*, const char \* *title*)**

plot 3D data. Input matrix has x,y on the first two columns and z on the third column

**Parameters**

*tc\_matrix* tree column matrix

*string* title of plot

## 4.9 Modeling

get/set parameters, equations, and so on

### Functions

- BEGIN\_C\_DECLS TCAPIEXPORT `tc_matrix tc_getParameters (tc_items a)`  
*get all the parameters for the given items. use `tc_allItems()` as argument to get all parameters*
- TCAPIEXPORT `tc_matrix tc_getInitialValues (tc_items a)`  
*get initial values of the given items. Fixed variables are included. use `tc_allItems()` for all items in the model.*
- TCAPIEXPORT void `tc_setInitialValues (tc_items items, tc_matrix values)`  
*set initial values of the given items.*
- TCAPIEXPORT `tc_matrix tc_getFixedVariables (tc_items a)`  
*get all fixed variables*
- TCAPIEXPORT `tc_matrix tc_getParametersAndFixedVariables (tc_items a)`  
*get all the parameters and fixed variables*
- TCAPIEXPORT double `tc_getParameter (long item, const char *attribute)`  
*get the parameter with the given name for the given item*
- TCAPIEXPORT `tc_matrix tc_getParametersNamed (tc_items a, tc_strings attributes)`  
*get all numerical Modeling with the given names for the given items*
- TCAPIEXPORT `tc_matrix tc_getParametersExcept (tc_items a, tc_strings attributes)`  
*get all numerical Modeling EXCEPT the given names*
- TCAPIEXPORT void `tc_setParameter (long item, const char *attribute, double value)`  
*set a parameter value for the given item*
- BEGIN\_C\_DECLS TCAPIEXPORT `tc_strings tc_getEventTriggers ()`  
*get the event triggers for a set of items*
- TCAPIEXPORT `tc_strings tc_getEventResponses ()`  
*get the event responses for a set of items*
- TCAPIEXPORT void `tc_addEvent (const char *trigger, const char *event)`  
*set the event trigger and response*
- TCAPIEXPORT `tc_strings tc_getForcingFunctionNames (tc_items a)`  
*get the forcing function names for a set of items*
- TCAPIEXPORT `tc_strings tc_getForcingFunctionAssignments (tc_items a)`  
*get the forcing function definitions for a set of items*

- TCAPIEXPORT void `tc_addForcingFunction` (long item, const char \*variable, const char \*formula)  
*set the forcing function for an item*
- TCAPIEXPORT int `tc_writeModel` (const char \*file, `tc_items` items)  
*write the ODE, stoichiometry, and rates functions to a file*
- BEGIN\_C\_DECLS TCAPIEXPORT `tc_matrix` `tc_getStoichiometry` (`tc_items` A)  
*get Modeling for the given items*
- TCAPIEXPORT void `tc_setStoichiometry` (`tc_items` A, `tc_matrix` N)  
*set Modeling for the given items (must be labeled)*
- TCAPIEXPORT `tc_strings` `tc_getRates` (`tc_items` A)  
*get rates for the given items*
- TCAPIEXPORT void `tc_setRates` (`tc_items` A, `tc_strings` rates)  
*set rates for the given items (same order as N)*
- TCAPIEXPORT `tc_matrix` `tc_getStoichiometryFor` (long x)  
*get Modeling for the given items*
- TCAPIEXPORT const char \* `tc_getRate` (long x)  
*get rate for the given items*
- TCAPIEXPORT void `tc_setRate` (long x, const char \*r)  
*set rate for the given items*
- TCAPIEXPORT void `tc_setStoichiometryFor` (long x, `tc_matrix` N)  
*set Modeling for the given items*
- TCAPIEXPORT void `tc_StoichiometryTool_api` (`tc_matrix`(\*getStoichiometry)(`tc_items`), void(\*setStoichiometry)(`tc_items`, `tc_matrix`), `tc_strings`(\*getRates)(`tc_items`), void(\*setRates)(`tc_items`, `tc_strings`))  
*initialize stiochiometry plug-in*

## 4.9.1 Detailed Description

get/set parameters, equations, and so on

## 4.9.2 Function Documentation

### 4.9.2.1 TCAPIEXPORT void `tc_addEvent` (const char \* *trigger*, const char \* *event*)

set the event trigger and response

#### Parameters

*string* trigger, e.g. `a > 2`

*string* response to trigger, e.g. `x = 5`



**4.9.2.2 TCAPIEXPORT void tc\_addForcingFunction (long *item*, const char \* *variable*, const char \* *formula*)**

set the forcing function for an item

**Parameters**

*int* address of an item, e.g. obtained from tc\_find

*string* name of existing variable or new variable

*string* formula for the variable

**4.9.2.3 TCAPIEXPORT tc\_strings tc\_getEventResponses ()**

get the event responses for a set of items

**Returns**

[tc\\_strings](#) all event trigger responses, e.g. A = 10; B = 2

**4.9.2.4 BEGIN\_C\_DECLS TCAPIEXPORT tc\_strings tc\_getEventTriggers ()**

get the event triggers for a set of items

**Returns**

[tc\\_strings](#) all event trigger equations, e.g. A > 10

**4.9.2.5 TCAPIEXPORT tc\_matrix tc\_getFixedVariables (tc\_items *a*)**

get all fixed variables

**Parameters**

[tc\\_items](#) list of items for which fixed attribute are set

[tc\\_matrix](#) matrix with 1 (fixed) or 0 (floating) in the same order as the list of items

**4.9.2.6 TCAPIEXPORT tc\_strings tc\_getForcingFunctionAssignments (tc\_items *a*)**

get the forcing function definitions for a set of items

**Parameters**

[tc\\_items](#) list of items. use [tc\\_allItems\(\)](#) to get all forcing functions

**Returns**

[tc\\_strings](#) list of assignment equations

#### 4.9.2.7 TCAPIEXPORT tc\_strings tc\_getForcingFunctionNames (tc\_items a)

get the forcing function names for a set of items

##### Parameters

*tc\_items* list of items. use *tc\_allItems()* to get all forcing functions

##### Returns

*tc\_strings* list of variable names

#### 4.9.2.8 TCAPIEXPORT tc\_matrix tc\_getInitialValues (tc\_items a)

get initial values of the given items. Fixed variables are included. use *tc\_allItems()* for all items in the model.

##### Parameters

*tc\_items* list of items for which the initial values are returned

##### Returns

*tc\_matrix* initial values in the same order as the input list

#### 4.9.2.9 TCAPIEXPORT double tc\_getParameter (long item, const char \* attribute)

get the parameter with the given name for the given item

##### Parameters

*int* item in the model, e.g. something returned from *tc\_find*

*string* name of the parameter

##### Returns

double value

#### 4.9.2.10 BEGIN\_C\_DECLS TCAPIEXPORT tc\_matrix tc\_getParameters (tc\_items a)

get all the parameters for the given items. use *tc\_allItems()* as argument to get all parameters

##### Parameters

*tc\_items* list of items for which the parameters are returned

##### Returns

*tc\_matrix* parameter values in the same order as the input list

**4.9.2.11 TCAPIEXPORT tc\_matrix tc\_getParametersAndFixedVariables (tc\_items a)**

get all the parameters and fixed variables

**Parameters**

*tc\_items* list of items. use *tc\_allItems()* to get all items in the model

**Returns**

*tc\_matrix* list of parameters and fixed variables. order is not preserved from the input

**4.9.2.12 TCAPIEXPORT tc\_matrix tc\_getParametersExcept (tc\_items a, tc\_strings attributes)**

get all numerical Modeling EXCEPT the given names

**Parameters**

*tc\_items* a list of items

*tc\_strings* a list of parameter names that exist in one or more of the given items

**Returns**

*tc\_matrix* the set of parameters with rownames as parameter names

**4.9.2.13 TCAPIEXPORT tc\_matrix tc\_getParametersNamed (tc\_items a, tc\_strings attributes)**

get all numerical Modeling with the given names for the given items

**Parameters**

*tc\_items* a list of items

*tc\_strings* a list of parameter names that exist in one or more of the given items

**Returns**

*tc\_matrix* the set of parameters with rownames as parameter names

**4.9.2.14 TCAPIEXPORT const char\* tc\_getRate (long x)**

get rate for the given items

**Parameters**

*int* address of a connection item

**Returns**

*tc\_matrix* reaction rate equations for given item

#### 4.9.2.15 TCAPIEXPORT tc\_strings tc\_getRates (tc\_items A)

get rates for the given items

##### Parameters

*tc\_items* list of items to get reaction rate equations from. use *tc\_allItems()* for whole model.

##### Returns

*tc\_strings* reaction rate equations for given items

#### 4.9.2.16 BEGIN\_C\_DECLS TCAPIEXPORT tc\_matrix tc\_getStoichiometry (tc\_items A)

get Modeling for the given items

##### Parameters

*tc\_items* list of items to get stoichiometry matrix from. use *tc\_allItems()* for whole model.

##### Returns

*tc\_matrix* stoichiometry matrix with rownames (molecules) and column names (reactions)

#### 4.9.2.17 TCAPIEXPORT tc\_matrix tc\_getStoichiometryFor (long x)

get Modeling for the given items

##### Parameters

*int* address of a connection item

##### Returns

*tc\_matrix* stoichiometry matrix for the item

#### 4.9.2.18 TCAPIEXPORT void tc\_setInitialValues (tc\_items items, tc\_matrix values)

set initial values of the given items.

##### Parameters

*tc\_items* list of items for which initial values are set

*tc\_matrix* the initial values in the same order as the list of items

#### 4.9.2.19 TCAPIEXPORT void tc\_setParameter (long item, const char \* attribute, double value)

set a parameter value for the given item

##### Parameters

*int* item in model

*string* name of parameter

**4.9.2.20 TCAPIEXPORT void tc\_setRate (long *x*, const char \* *r*)**

set rate for the given items

**Parameters**

*int* address of a connection item  
*tc\_matrix* reaction rate equations for given item

**4.9.2.21 TCAPIEXPORT void tc\_setRates (tc\_items *A*, tc\_strings *rates*)**

set rates for the given items (same order as *N*)

**Parameters**

*tc\_items* list of items to set reaction rate equations for. use *tc\_allItems()* for whole model.

**Returns**

*tc\_strings* reaction rate equations for given items

**4.9.2.22 TCAPIEXPORT void tc\_setStoichiometry (tc\_items *A*, tc\_matrix *N*)**

set Modeling for the given items (must be labeled)

**Parameters**

*tc\_items* list of items to set stoichiometry matrix for. use *tc\_allItems()* for whole model.  
*tc\_matrix* new stoichiometry matrix with rownames (molecules) and column names (reactions) \

**4.9.2.23 TCAPIEXPORT void tc\_setStoichiometryFor (long *x*, tc\_matrix *N*)**

set Modeling for the given items

**Parameters**

*int* address of a connection item  
*tc\_matrix* stoichiometry matrix for given item

**4.9.2.24 TCAPIEXPORT int tc\_writeModel (const char \* *file*, tc\_items *items*)**

write the ODE, stoichiometry, and rates functions to a file

**Parameters**

*string* output filename  
*tc\_items* items to include in the model. use *tc\_allItems* for the whole model

## 4.10 Connections

change appearance of connection arcs

### Functions

- TCAPIEXPORT long [tc\\_insertConnection](#) (tc\_items parts, const char \*name, const char \*family)  
*connect a set of parts (in) to another (out). give the connection name and family. returns the inserted connection*
- TCAPIEXPORT tc\_items [tc\\_getConnectedNodes](#) (long connection)  
*get the connected parts for a connection*
- TCAPIEXPORT tc\_items [tc\\_getConnectedNodesWithRole](#) (long connection, const char \*role)  
*get the parts with a role in a connection, such as reactants*
- TCAPIEXPORT tc\_items [tc\\_getConnections](#) (long part)  
*get connections for a part*
- TCAPIEXPORT tc\_items [tc\\_getConnectionsWithRole](#) (long part, const char \*role)  
*get connections where the given part has the given role, e.g. reactant*
- BEGIN\_C\_DECLS TCAPIEXPORT double [tc\\_getControlPointX](#) (long connection, long part, int whichPoint)  
*get x position of a control point*
- TCAPIEXPORT double [tc\\_getControlPointY](#) (long connection, long part, int whichPoint)  
*get y position of a control point*
- TCAPIEXPORT void [tc\\_setControlPoint](#) (long connection, long part, int whichPoint, double x, double y)  
*set x and y position of a control point*
- TCAPIEXPORT void [tc\\_setCenterPoint](#) (long connection, double y, double x)  
*set x and y position of the central control point*
- TCAPIEXPORT double [tc\\_getCenterPointX](#) (long connection)  
*get x position of the central control point*
- TCAPIEXPORT double [tc\\_getCenterPointY](#) (long connection)  
*get y position of the central control point*
- TCAPIEXPORT void [tc\\_setStraight](#) (long item, int straight)  
*switch between beziers and lines for drawing the connector, where 1 = line, 0 = bezier*
- TCAPIEXPORT void [tc\\_setAllStraight](#) (int straight)  
*switch between beziers and lines for drawing ALL connectors*
- TCAPIEXPORT void [tc\\_setLineWidth](#) (long item, double width, int permanent)  
*set the line width. Indicate whether the change should be temporary or permanent.*

### 4.10.1 Detailed Description

change appearance of connection arcs

### 4.10.2 Function Documentation

#### 4.10.2.1 TCAPIEXPORT double tc\_getCenterPointX (long *connection*)

get x position of the central control point

##### Parameters

*int* address of a connection, e.g. obtained using tc\_find

##### Returns

double x position

#### 4.10.2.2 TCAPIEXPORT double tc\_getCenterPointY (long *connection*)

get y position of the central control point

##### Parameters

*int* address of a connection, e.g. obtained using tc\_find

##### Returns

double y position

#### 4.10.2.3 TCAPIEXPORT tc\_items tc\_getConnectedNodes (long *connection*)

get the connected parts for a connection

##### Parameters

*int* address of a connection, e.g. obtained using tc\_find

##### Returns

[tc\\_items](#) all nodes connection by the given connection

#### 4.10.2.4 TCAPIEXPORT tc\_items tc\_getConnectedNodesWithRole (long *connection*, const char \* *role*)

get the parts with a role in a connection, such as reactants

get the parts with a specific role in the given connection, such as reactant

##### Parameters

*int* address of a connection, e.g. obtained using tc\_find

*string* a role, e.g. Reactant

### Returns

[tc\\_items](#) all nodes in the given connection with the given role

#### 4.10.2.5 TCAPIEXPORT *tc\_items* tc\_getConnections (long *part*)

get connections for a part

### Parameters

*int* address of a node, e.g. obtained using tc\_find

### Returns

[tc\\_items](#) all connections linked to the given node

#### 4.10.2.6 TCAPIEXPORT *tc\_items* tc\_getConnectionsWithRole (long *part*, const char \* *role*)

get connections where the given part has the given role, e.g. reactant

get connections where the given parts has a specific role, such as reactant

### Parameters

*int* address of a node, e.g. obtained using tc\_find

*string* a role, such as reactant

### Returns

[tc\\_items](#) connections linked to the given node with the given role

#### 4.10.2.7 BEGIN\_C\_DECLS TCAPIEXPORT double tc\_getControlPointX (long *connection*, long *part*, int *whichPoint*)

get x position of a control point

### Parameters

*int* address of a connection, e.g. obtained using tc\_find

*int* address of a node, e.g. obtained using tc\_find

*int* index of the control point related to the given connection and the given node

### Returns

double x position



**4.10.2.8 TCAPIEXPORT double tc\_getControlPointY (long *connection*, long *part*, int *whichPoint*)**

get y position of a control point

**Parameters**

*int* address of a connection, e.g. obtained using tc\_find

*int* address of a node, e.g. obtained using tc\_find

*int* index of the control point related to the given connection and the given node

**Returns**

double y position

**4.10.2.9 BEGIN\_C\_DECLS TCAPIEXPORT long tc\_insertConnection (tc\_items *parts*, const char \* *name*, const char \* *family*)**

connect a set of parts (in) to another (out). give the connection name and family. returns the inserted connection

connect a set of parts. The role of each part is automatically determined by its type. Give the connection name and family. returns the inserted connection

**Parameters**

*tc\_items* nodes to be connected

*string* name of new connection

*string* type of the new connection, i.e. one of the connection types in the catalog

**4.10.2.10 TCAPIEXPORT void tc\_setAllStraight (int *straight*)**

switch between beziers and lines for drawing ALL connectors

**Parameters**

*int* 0 (Bezier) or 1 (straight lines)

**4.10.2.11 TCAPIEXPORT void tc\_setCenterPoint (long *connection*, double *y*, double *x*)**

set x and y position of the central control point

**Parameters**

*int* address of a connection, e.g. obtained using tc\_find

*double* x position

*double* y position

**4.10.2.12 TCAPIEXPORT void tc\_setControlPoint (long *connection*, long *part*, int *whichPoint*, double *x*, double *y*)**

set x and y position of a control point

**Parameters**

*long* the connection

*long* the node that is associated with the particular curve of interest

*int* the index of the point on that curve of interest

*double* x value

*double* y value

**4.10.2.13 TCAPIEXPORT void tc\_setLineWidth (long *item*, double *width*, int *permanent*)**

set the line width. Indicate whether the change should be temporary or permanent.

**Parameters**

*int* address of a connection, e.g. obtained using tc\_find

*double* line width

*int* 0 (temporary change) or 1 (permanent change)

**4.10.2.14 TCAPIEXPORT void tc\_setStraight (long *item*, int *straight*)**

switch between beziers and lines for drawing the connector, where 1 = line, 0 = bezier

**Parameters**

*int* address of a connection, e.g. obtained using tc\_find

*int* 0 (Bezier) or 1 (straight lines)

## 4.11 Import/Export

Import/Export different file formats.

### Functions

- TCAPIEXPORT void [tc\\_exportSBML](#) (const char \*s)  
*save sbml format to a file*
- TCAPIEXPORT void [tc\\_importSBML](#) (const char \*s)  
*load sbml model as string*

### 4.11.1 Detailed Description

Import/Export different file formats.

### 4.11.2 Function Documentation

#### 4.11.2.1 BEGIN\_C\_DECLS TCAPIEXPORT void tc\_exportSBML (const char \* s)

save sbml format to a file

##### Parameters

*const* char\* file name

#### 4.11.2.2 TCAPIEXPORT void tc\_importSBML (const char \* s)

load sbml model as string

##### Parameters

*const* char\* sbml model file or string

## 4.12 Simulation

Simulations and other numerical analysis.

### Functions

- `BEGIN_C_DECL` `TCAPIEXPORT tc_matrix tc_simulateDeterministic` (double startTime, double endTime, int numSteps)  
*simulate using LSODA numerical integrator*
- `TCAPIEXPORT tc_matrix tc_simulateStochastic` (double startTime, double endTime, int numSteps)  
*simulate using exact stochastic algorithm*
- `TCAPIEXPORT tc_matrix tc_simulateHybrid` (double startTime, double endTime, int numSteps)  
*simulate using Hybrid algorithm/deterministic algorithmparam double start time*
- `TCAPIEXPORT tc_matrix tc_simulateTauLeap` (double startTime, double endTime, int numSteps)  
*simulate using Tau Leap stochastic algorithm*
- `TCAPIEXPORT tc_matrix tc_getSteadyState` ()  
*bring the system to steady state*
- `TCAPIEXPORT tc_matrix tc_steadyStateScan` (const char \*param, double start, double end, int numSteps)  
*calculate steady state for each value of a parameter*
- `TCAPIEXPORT tc_matrix tc_steadyStateScan2D` (const char \*param1, double start1, double end1, int numSteps1, const char \*param2, double start2, double end2, int numSteps2)  
*calculate steady state for each value of two parameters*
- `TCAPIEXPORT tc_matrix tc_getJacobian` ()  
*get the Jacobian at the current state*
- `TCAPIEXPORT tc_matrix tc_getEigenvalues` ()  
*get the eigenvalues of the Jacobian at the current state*
- `TCAPIEXPORT tc_matrix tc_getUnscaledElasticities` ()  
*unscaled elasticities*
- `TCAPIEXPORT tc_matrix tc_getUnscaledConcentrationCC` ()  
*unscaled elasticities*
- `TCAPIEXPORT tc_matrix tc_getUnscaledFluxCC` ()  
*unscaled flux control coefficients*
- `TCAPIEXPORT tc_matrix tc_getScaledElasticities` ()  
*scaled elasticities*
- `TCAPIEXPORT tc_matrix tc_getScaledConcentrationCC` ()

*scaled concentration control coefficients*

- TCAPIEXPORT `tc_matrix tc_getScaledFluxCC ()`

*scaled flux control coefficients*

- TCAPIEXPORT `tc_matrix tc_reducedStoichiometry ()`

*reduced stoichiometry*

- TCAPIEXPORT `tc_matrix tc_elementaryFluxModes ()`

*elementary flux modes*

- TCAPIEXPORT `tc_matrix tc_LMatrix ()`

*left nullspace of the stoichiometry matrix*

- TCAPIEXPORT `tc_matrix tc_KMatrix ()`

*right nullspace of the stoichiometry matrix*

### 4.12.1 Detailed Description

Simulations and other numerical analysis.

### 4.12.2 Function Documentation

#### 4.12.2.1 TCAPIEXPORT `tc_matrix tc_elementaryFluxModes ()`

elementary flux modes

##### Returns

`tc_matrix`

#### 4.12.2.2 TCAPIEXPORT `tc_matrix tc_getEigenvalues ()`

get the eigenvalues of the Jacobian at the current state

##### Returns

`tc_matrix` matrix with 1 row and n columns, each containing an eigenvalue

#### 4.12.2.3 TCAPIEXPORT `tc_matrix tc_getJacobian ()`

get the Jacobian at the current state

##### Returns

`tc_matrix` matrix with n rows and n columns, where n = number of species

**4.12.2.4 TCAPIEXPORT tc\_matrix tc\_getScaledConcentrationCC ()**

scaled concentration control coefficients

**Returns**

[tc\\_matrix](#)

**4.12.2.5 TCAPIEXPORT tc\_matrix tc\_getScaledElasticities ()**

scaled elasticities

**Returns**

[tc\\_matrix](#)

**4.12.2.6 TCAPIEXPORT tc\_matrix tc\_getScaledFluxCC ()**

scaled flux control coefficients

**Returns**

[tc\\_matrix](#)

**4.12.2.7 TCAPIEXPORT tc\_matrix tc\_getSteadyState ()**

bring the system to steady state

**Returns**

[tc\\_matrix](#) matrix with 1 row and n columns, where n = number of species

**4.12.2.8 TCAPIEXPORT tc\_matrix tc\_getUnscaledConcentrationCC ()**

unscaled elasticities

unscaled concentration control coefficients

**Returns**

[tc\\_matrix](#)

**4.12.2.9 TCAPIEXPORT tc\_matrix tc\_getUnscaledElasticities ()**

unscaled elasticities

**Returns**

[tc\\_matrix](#)

**4.12.2.10 TCAPIEXPORT tc\_matrix tc\_getUnscaledFluxCC ()**

unscaled flux control coefficients

**Returns**

[tc\\_matrix](#)

**4.12.2.11 TCAPIEXPORT tc\_matrix tc\_KMatrix ()**

right nullspace of the stoichiometry matrix

**Returns**

[tc\\_matrix](#)

**4.12.2.12 TCAPIEXPORT tc\_matrix tc\_LMatrix ()**

left nullspace of the stoichiometry matrix

**Returns**

[tc\\_matrix](#)

**4.12.2.13 TCAPIEXPORT tc\_matrix tc\_reducedStoichiometry ()**

reduced stoichiometry

**Returns**

[tc\\_matrix](#)

**4.12.2.14 BEGIN\_C\_DECLS TCAPIEXPORT tc\_matrix tc\_simulateDeterministic (double  
startTime, double endTime, int numSteps)**

simulate using LSODA numerical integrator

**Parameters**

*double* start time

*double* end time

*int* number of steps in the output

**Returns**

[tc\\_matrix](#) matrix of concentration or particles

#### 4.12.2.15 TCAPIEXPORT tc\_matrix tc\_simulateHybrid (double *startTime*, double *endTime*, int *numSteps*)

simulate using Hybrid algorithm/deterministic algorithmparam double start time

##### Parameters

*double* end time

*int* number of steps in the output

##### Returns

[tc\\_matrix](#) matrix of concentration or particles

#### 4.12.2.16 TCAPIEXPORT tc\_matrix tc\_simulateStochastic (double *startTime*, double *endTime*, int *numSteps*)

simulate using exact stochastic algorithm

##### Parameters

*double* start time

*double* end time

*int* number of steps in the output

##### Returns

[tc\\_matrix](#) matrix of concentration or particles

#### 4.12.2.17 TCAPIEXPORT tc\_matrix tc\_simulateTauLeap (double *startTime*, double *endTime*, int *numSteps*)

simulate using Tau Leap stochastic algorithm

##### Parameters

*double* start time

*double* end time

*int* number of steps in the output

##### Returns

[tc\\_matrix](#) matrix of concentration or particles

#### 4.12.2.18 TCAPIEXPORT tc\_matrix tc\_steadyStateScan (const char \* *param*, double *start*, double *end*, int *numSteps*)

calculate steady state for each value of a parameter

##### Parameters

*char* \* parameter name



*double* start value

*double* end value

*int* number of steps in the output

#### Returns

[tc\\_matrix](#) matrix of concentration or particles

**4.12.2.19 TCAPIEXPORT tc\_matrix tc\_steadyStateScan2D (const char \*param1, double start1, double end1, int numSteps1, const char \*param2, double start2, double end2, int numSteps2)**

calculate steady state for each value of two parameters

#### Parameters

*char* \* first parameter name

*double* start value for parameter 1

*double* end value for parameter 1

*int* number of steps in parameter 1

*char* \* second parameter name

*double* start value for parameter 2

*double* end value for parameter 2

*int* number of steps in parameter 2

#### Returns

[tc\\_matrix](#) matrix of concentration or particles

## 4.13 Modules

Functions for listing and swapping sub-models.

### Functions

- **BEGIN\_C\_DECLS TCAPIEXPORT void tc\_substituteModel** (long item, const char \*filename)  
*load a sub-model to represent the processes inside an existing connection*
- **TCAPIEXPORT tc\_strings tc\_listOfPossibleModels** (long item)  
*get the list of possible model files that can be used as a sub-model to represent the processes inside an existing connection*

### 4.13.1 Detailed Description

Functions for listing and swapping sub-models.

### 4.13.2 Function Documentation

#### 4.13.2.1 TCAPIEXPORT tc\_strings tc\_listOfPossibleModels (long item)

get the list of possible model files that can be used as a sub-model to represent the processes inside an existing connection

#### Parameters

*long* connection that will be the parent of the new model

#### Returns

tc\_list list of file names

#### 4.13.2.2 BEGIN\_C\_DECLS TCAPIEXPORT void tc\_substituteModel (long item, const char \*filename)

load a sub-model to represent the processes inside an existing connection

#### Parameters

*long* connection that will be the parent of the new model

*const* char\* file name of new model

# Chapter 5

## Class Documentation

### 5.1 tc\_items Struct Reference

An array of int objects with length information. Use tc\_getItem(M,i) to get the i-th item.

```
#include <TC_structs.h>
```

#### Public Attributes

- int **length**
- long \* **items**

#### 5.1.1 Detailed Description

An array of int objects with length information. Use tc\_getItem(M,i) to get the i-th item.

The documentation for this struct was generated from the following file:

- TC\_structs.h

## 5.2 tc\_matrix Struct Reference

A 2D table of doubles with row and column names. Use `tc_getMatrixValue(M,i,j)` to get the *i,j*-th value in `tc_matrix` *M*.

```
#include <TC_structs.h>
```

### Public Attributes

- `int` **rows**
- `int` **cols**
- `double *` **values**
- `tc_strings` **rownames**
- `tc_strings` **colnames**

### 5.2.1 Detailed Description

A 2D table of doubles with row and column names. Use `tc_getMatrixValue(M,i,j)` to get the *i,j*-th value in `tc_matrix` *M*.

The documentation for this struct was generated from the following file:

- `TC_structs.h`

## 5.3 tc\_strings Struct Reference

An array of strings with length information. Use tc\_getString(M,i) to get the i-th string.

```
#include <TC_structs.h>
```

### Public Attributes

- int **length**
- char \*\* **strings**

#### 5.3.1 Detailed Description

An array of strings with length information. Use tc\_getString(M,i) to get the i-th string.

The documentation for this struct was generated from the following file:

- TC\_structs.h

## 5.4 tc\_table Struct Reference

A 2D table of strings with row and column names. Use `tc_getTableValue(M,i,j)` to get the *i,j*-th value in [tc\\_matrix](#) *M*.

```
#include <TC_structs.h>
```

### Public Attributes

- `int` **rows**
- `int` **cols**
- `char **` **strings**
- [tc\\_strings](#) **rownames**
- [tc\\_strings](#) **colnames**

### 5.4.1 Detailed Description

A 2D table of strings with row and column names. Use `tc_getTableValue(M,i,j)` to get the *i,j*-th value in [tc\\_matrix](#) *M*.

The documentation for this struct was generated from the following file:

- `TC_structs.h`

# Index

## Annotation

- `tc_getAllTextNamed`, [27](#)
- `tc_getAnnotation`, [27](#)
- `tc_getFamily`, [27](#)
- `tc_getName`, [27](#)
- `tc_getNames`, [27](#)
- `tc_getTextAttribute`, [27](#)
- `tc_getUniqueName`, [28](#)
- `tc_getUniqueNames`, [28](#)
- `tc_isA`, [28](#)
- `tc_rename`, [28](#)
- `tc_setAnnotation`, [29](#)
- `tc_setSequence`, [29](#)
- `tc_setTextAttribute`, [29](#)

## Annotations, [26](#)

## Appearance, [15](#)

- `tc_changeArrowHead`, [16](#)
- `tc_changeNodeImage`, [16](#)
- `tc_getAngle`, [16](#)
- `tc_getColor`, [16](#)
- `tc_getHeight`, [16](#)
- `tc_getPos`, [17](#)
- `tc_getWidth`, [17](#)
- `tc_getX`, [17](#)
- `tc_getY`, [17](#)
- `tc_moveSelected`, [18](#)
- `tc_setAngle`, [18](#)
- `tc_setColor`, [18](#)
- `tc_setPos`, [18](#)
- `tc_setPosMulti`, [18](#)
- `tc_setSize`, [19](#)

## Basic

- `tc_appendColumns`, [9](#)
- `tc_appendRows`, [9](#)
- `tc_createItemsArray`, [9](#)
- `tc_createMatrix`, [9](#)
- `tc_createStringsArray`, [10](#)
- `tc_createTable`, [10](#)
- `tc_deleteItemsArray`, [10](#)
- `tc_deleteMatrix`, [10](#)
- `tc_deleteStringsArray`, [10](#)
- `tc_deleteTable`, [11](#)
- `tc_getColumnName`, [11](#)
- `tc_getItem`, [11](#)

- `tc_getMatrixValue`, [11](#)
- `tc_getRowName`, [11](#)
- `tc_getString`, [12](#)
- `tc_getTableValue`, [12](#)
- `tc_printMatrixToFile`, [12](#)
- `tc_printOutMatrix`, [12](#)
- `tc_printOutTable`, [13](#)
- `tc_printTableToFile`, [13](#)
- `tc_setColumnName`, [13](#)
- `tc_setItem`, [13](#)
- `tc_setMatrixValue`, [13](#)
- `tc_setRowName`, [14](#)
- `tc_setString`, [14](#)
- `tc_setTableValue`, [14](#)

## Basic operations, [7](#)

## Connections, [52](#)

- `tc_getCenterPointX`, [53](#)
- `tc_getCenterPointY`, [53](#)
- `tc_getConnectedNodes`, [53](#)
- `tc_getConnectedNodesWithRole`, [53](#)
- `tc_getConnections`, [54](#)
- `tc_getConnectionsWithRole`, [54](#)
- `tc_getControlPointX`, [54](#)
- `tc_getControlPointY`, [54](#)
- `tc_insertConnection`, [55](#)
- `tc_setAllStraight`, [55](#)
- `tc_setCenterPoint`, [55](#)
- `tc_setControlPoint`, [55](#)
- `tc_setLineWidth`, [56](#)
- `tc_setStraight`, [56](#)

## Export

- `tc_exportSBML`, [57](#)
- `tc_importSBML`, [57](#)

## Get

- `tc_alignParts`, [21](#)
- `tc_allItems`, [21](#)
- `tc_find`, [21](#)
- `tc_findItems`, [21](#)
- `tc_getChildren`, [22](#)
- `tc_getName`, [22](#)
- `tc_getNames`, [22](#)
- `tc_getParent`, [22](#)

- tc\_getUniqueName, 23
- tc\_getUniqueNames, 23
- tc\_itemsOfFamily, 23
- tc\_itemsOfFamilyFrom, 23
- tc\_partsDownstream, 24
- tc\_partsIn, 24
- tc\_partsUpstream, 24
- tc\_rename, 24
- tc\_select, 24
- tc\_selectedItems, 25
- tc\_setSequence, 25
- Get items, 20
- Graphing, 42
- Import/Export, 57
- Input
  - tc\_addInputWindowCheckbox, 32
  - tc\_addInputWindowOptions, 32
  - tc\_askQuestion, 32
  - tc\_clear, 32
  - tc\_createInputWindow, 32
  - tc\_createInputWindowFromFile, 33
  - tc\_createSliders, 33
  - tc\_displayNumber, 33
  - tc\_displayText, 33
  - tc\_errorReport, 34
  - tc\_getFilename, 34
  - tc\_getNumber, 34
  - tc\_getNumbers, 34
  - tc\_getStringFromList, 34
  - tc\_highlight, 35
  - tc\_messageDialog, 35
  - tc\_openFile, 35
  - tc\_openNewWindow, 35
  - tc\_print, 36
  - tc\_printFile, 36
  - tc\_printMatrix, 36
  - tc\_saveToFile, 36
  - tc\_screenHeight, 36
  - tc\_screenshot, 37
  - tc\_screenWidth, 37
  - tc\_screenX, 37
  - tc\_screenY, 37
  - tc\_setDisplayLabelColor, 37
  - tc\_zoom, 37
- Input and Output, 30
- Modeling, 45
  - tc\_addEvent, 46
  - tc\_addForcingFunction, 46
  - tc\_getEventResponses, 47
  - tc\_getEventTriggers, 47
  - tc\_getFixedVariables, 47
  - tc\_getForcingFunctionAssignments, 47
  - tc\_getForcingFunctionNames, 47
  - tc\_getInitialValues, 48
  - tc\_getParameter, 48
  - tc\_getParameters, 48
  - tc\_getParametersAndFixedVariables, 48
  - tc\_getParametersExcept, 49
  - tc\_getParametersNamed, 49
  - tc\_getRate, 49
  - tc\_getRates, 49
  - tc\_getStoichiometry, 50
  - tc\_getStoichiometryFor, 50
  - tc\_setInitialValues, 50
  - tc\_setParameter, 50
  - tc\_setRate, 50
  - tc\_setRates, 51
  - tc\_setStoichiometry, 51
  - tc\_setStoichiometryFor, 51
  - tc\_writeModel, 51
- Module
  - tc\_listOfPossibleModels, 64
  - tc\_substituteModel, 64
- Modules, 64
- Network data, 41
- Plotting
  - tc\_errorBars, 42
  - tc\_getPlotData, 42
  - tc\_gnuplot, 43
  - tc\_hist, 43
  - tc\_multiplot, 43
  - tc\_plot, 43
  - tc\_savePlot, 43
  - tc\_scatterplot, 44
  - tc\_surface, 44
- Simulation, 58
  - tc\_elementaryFluxModes, 59
  - tc\_getEigenvalues, 59
  - tc\_getJacobian, 59
  - tc\_getScaledConcentrationCC, 59
  - tc\_getScaledElasticities, 60
  - tc\_getScaledFluxCC, 60
  - tc\_getSteadyState, 60
  - tc\_getUnscaledConcentrationCC, 60
  - tc\_getUnscaledElasticities, 60
  - tc\_getUnscaledFluxCC, 60
  - tc\_KMatrix, 61
  - tc\_LMatrix, 61
  - tc\_reducedStoichiometry, 61
  - tc\_simulateDeterministic, 61
  - tc\_simulateHybrid, 61
  - tc\_simulateStochastic, 62
  - tc\_simulateTauLeap, 62



- tc\_steadyStateScan, 62
- tc\_steadyStateScan2D, 63
- System
  - tc\_appDir, 39
  - tc\_homeDir, 39
  - tc\_isLinux, 39
  - tc\_isMac, 40
  - tc\_isWindows, 40
- System information, 39
- tc\_addEvent
  - Modeling, 46
- tc\_addForcingFunction
  - Modeling, 46
- tc\_addInputWindowCheckbox
  - Input, 32
- tc\_addInputWindowOptions
  - Input, 32
- tc\_alignParts
  - Get, 21
- tc\_allItems
  - Get, 21
- tc\_appDir
  - System, 39
- tc\_appendColumns
  - Basic, 9
- tc\_appendRows
  - Basic, 9
- tc\_askQuestion
  - Input, 32
- tc\_changeArrowHead
  - Appearance, 16
- tc\_changeNodeImage
  - Appearance, 16
- tc\_clear
  - Input, 32
- tc\_createInputWindow
  - Input, 32
- tc\_createInputWindowFromFile
  - Input, 33
- tc\_createItemsArray
  - Basic, 9
- tc\_createMatrix
  - Basic, 9
- tc\_createSliders
  - Input, 33
- tc\_createStringsArray
  - Basic, 10
- tc\_createTable
  - Basic, 10
- tc\_deleteItemsArray
  - Basic, 10
- tc\_deleteMatrix
  - Basic, 10
- tc\_deleteStringsArray
  - Basic, 10
- tc\_deleteTable
  - Basic, 11
- tc\_displayNumber
  - Input, 33
- tc\_displayText
  - Input, 33
- tc\_elementaryFluxModes
  - Simulation, 59
- tc\_errorBars
  - Plotting, 42
- tc\_errorReport
  - Input, 34
- tc\_exportSBML
  - Export, 57
- tc\_find
  - Get, 21
- tc\_findItems
  - Get, 21
- tc\_getAllTextNamed
  - Annotation, 27
- tc\_getAngle
  - Appearance, 16
- tc\_getAnnotation
  - Annotation, 27
- tc\_getCenterPointX
  - Connections, 53
- tc\_getCenterPointY
  - Connections, 53
- tc\_getChildren
  - Get, 22
- tc\_getColor
  - Appearance, 16
- tc\_getColumnName
  - Basic, 11
- tc\_getConnectedNodes
  - Connections, 53
- tc\_getConnectedNodesWithRole
  - Connections, 53
- tc\_getConnections
  - Connections, 54
- tc\_getConnectionsWithRole
  - Connections, 54
- tc\_getControlPointX
  - Connections, 54
- tc\_getControlPointY
  - Connections, 54
- tc\_getEigenvalues
  - Simulation, 59
- tc\_getEventResponses
  - Modeling, 47
- tc\_getEventTriggers
  - Modeling, 47

- tc\_getFamily
  - Annotation, [27](#)
- tc\_getFilename
  - Input, [34](#)
- tc\_getFixedVariables
  - Modeling, [47](#)
- tc\_getForcingFunctionAssignments
  - Modeling, [47](#)
- tc\_getForcingFunctionNames
  - Modeling, [47](#)
- tc\_getHeight
  - Appearance, [16](#)
- tc\_getInitialValues
  - Modeling, [48](#)
- tc\_getItem
  - Basic, [11](#)
- tc\_getJacobian
  - Simulation, [59](#)
- tc\_getMatrixValue
  - Basic, [11](#)
- tc\_getName
  - Annotation, [27](#)
  - Get, [22](#)
- tc\_getNames
  - Annotation, [27](#)
  - Get, [22](#)
- tc\_getNumber
  - Input, [34](#)
- tc\_getNumbers
  - Input, [34](#)
- tc\_getParameter
  - Modeling, [48](#)
- tc\_getParameters
  - Modeling, [48](#)
- tc\_getParametersAndFixedVariables
  - Modeling, [48](#)
- tc\_getParametersExcept
  - Modeling, [49](#)
- tc\_getParametersNamed
  - Modeling, [49](#)
- tc\_getParent
  - Get, [22](#)
- tc\_getPlotData
  - Plotting, [42](#)
- tc\_getPos
  - Appearance, [17](#)
- tc\_getRate
  - Modeling, [49](#)
- tc\_getRates
  - Modeling, [49](#)
- tc\_getRowName
  - Basic, [11](#)
- tc\_getScaledConcentrationCC
  - Simulation, [59](#)
- tc\_getScaledElasticities
  - Simulation, [60](#)
- tc\_getScaledFluxCC
  - Simulation, [60](#)
- tc\_getSteadyState
  - Simulation, [60](#)
- tc\_getStoichiometry
  - Modeling, [50](#)
- tc\_getStoichiometryFor
  - Modeling, [50](#)
- tc\_getString
  - Basic, [12](#)
- tc\_getStringFromList
  - Input, [34](#)
- tc\_getTableValue
  - Basic, [12](#)
- tc\_getTextAttribute
  - Annotation, [27](#)
- tc\_getUniqueName
  - Annotation, [28](#)
  - Get, [23](#)
- tc\_getUniqueNames
  - Annotation, [28](#)
  - Get, [23](#)
- tc\_getUnscaledConcentrationCC
  - Simulation, [60](#)
- tc\_getUnscaledElasticities
  - Simulation, [60](#)
- tc\_getUnscaledFluxCC
  - Simulation, [60](#)
- tc\_getWidth
  - Appearance, [17](#)
- tc\_getX
  - Appearance, [17](#)
- tc\_getY
  - Appearance, [17](#)
- tc\_gnuplot
  - Plotting, [43](#)
- tc\_highlight
  - Input, [35](#)
- tc\_hist
  - Plotting, [43](#)
- tc\_homeDir
  - System, [39](#)
- tc\_importSBML
  - Export, [57](#)
- tc\_insertConnection
  - Connections, [55](#)
- tc\_isA
  - Annotation, [28](#)
- tc\_isLinux
  - System, [39](#)
- tc\_isMac
  - System, [40](#)

- tc\_isWindows
  - System, 40
- tc\_items, 65
- tc\_itemsOfFamily
  - Get, 23
- tc\_itemsOfFamilyFrom
  - Get, 23
- tc\_KMatrix
  - Simulation, 61
- tc\_listOfPossibleModels
  - Module, 64
- tc\_LMatrix
  - Simulation, 61
- tc\_matrix, 66
- tc\_messageDialog
  - Input, 35
- tc\_moveSelected
  - Appearance, 18
- tc\_multiplot
  - Plotting, 43
- tc\_openFile
  - Input, 35
- tc\_openNewWindow
  - Input, 35
- tc\_partsDownstream
  - Get, 24
- tc\_partsIn
  - Get, 24
- tc\_partsUpstream
  - Get, 24
- tc\_plot
  - Plotting, 43
- tc\_print
  - Input, 36
- tc\_printFile
  - Input, 36
- tc\_printMatrix
  - Input, 36
- tc\_printMatrixToFile
  - Basic, 12
- tc\_printOutMatrix
  - Basic, 12
- tc\_printOutTable
  - Basic, 13
- tc\_printTableToFile
  - Basic, 13
- tc\_reducedStoichiometry
  - Simulation, 61
- tc\_rename
  - Annotation, 28
  - Get, 24
- tc\_savePlot
  - Plotting, 43
- tc\_saveToFile
  - Input, 36
- tc\_scatterplot
  - Plotting, 44
- tc\_screenHeight
  - Input, 36
- tc\_screenshot
  - Input, 37
- tc\_screenWidth
  - Input, 37
- tc\_screenX
  - Input, 37
- tc\_screenY
  - Input, 37
- tc\_select
  - Get, 24
- tc\_selectedItems
  - Get, 25
- tc\_setAllStraight
  - Connections, 55
- tc\_setAngle
  - Appearance, 18
- tc\_setAnnotation
  - Annotation, 29
- tc\_setCenterPoint
  - Connections, 55
- tc\_setColor
  - Appearance, 18
- tc\_setColumnName
  - Basic, 13
- tc\_setControlPoint
  - Connections, 55
- tc\_setDisplayLabelColor
  - Input, 37
- tc\_setInitialValues
  - Modeling, 50
- tc\_setItem
  - Basic, 13
- tc\_setLineWidth
  - Connections, 56
- tc\_setMatrixValue
  - Basic, 13
- tc\_setParameter
  - Modeling, 50
- tc\_setPos
  - Appearance, 18
- tc\_setPosMulti
  - Appearance, 18
- tc\_setRate
  - Modeling, 50
- tc\_setRates
  - Modeling, 51
- tc\_setRowName
  - Basic, 14
- tc\_setSequence

- Annotation, [29](#)
- Get, [25](#)
- tc\_setSize
  - Appearance, [19](#)
- tc\_setStoichiometry
  - Modeling, [51](#)
- tc\_setStoichiometryFor
  - Modeling, [51](#)
- tc\_setStraight
  - Connections, [56](#)
- tc\_setString
  - Basic, [14](#)
- tc\_setTableValue
  - Basic, [14](#)
- tc\_setTextAttribute
  - Annotation, [29](#)
- tc\_simulateDeterministic
  - Simulation, [61](#)
- tc\_simulateHybrid
  - Simulation, [61](#)
- tc\_simulateStochastic
  - Simulation, [62](#)
- tc\_simulateTauLeap
  - Simulation, [62](#)
- tc\_steadyStateScan
  - Simulation, [62](#)
- tc\_steadyStateScan2D
  - Simulation, [63](#)
- tc\_strings, [67](#)
- tc\_substituteModel
  - Module, [64](#)
- tc\_surface
  - Plotting, [44](#)
- tc\_table, [68](#)
- tc\_writeModel
  - Modeling, [51](#)
- tc\_zoom
  - Input, [37](#)