Contents

1 mtoc++ 1.5 Software Documentation 2
   1.1 Introduction ............................................................. 2
   1.2 Contents ............................................................... 2
   1.3 Downloads .............................................................. 2

2 Changes and new features in mtoc++ 2
   2.1 New features in mtoc++ 0.1 .......................................... 4
   2.2 Changes in mtoc++ 0.1 ................................................ 4
   2.3 New features in mtoc++ 1.2 .......................................... 4
   2.4 Changes in mtoc++ 1.2 ................................................ 5
   2.5 New features in mtoc++ 1.3 .......................................... 5
   2.6 Changes in mtoc++ 1.3 ................................................ 5
   2.7 New features in mtoc++ 1.4 .......................................... 6
   2.8 Changes in mtoc++ 1.4 ................................................ 7
   2.9 New features in mtoc++ 1.5 .......................................... 7
   2.10 Changes in mtoc++ 1.5 ............................................... 7

3 Downloading mtoc++ 8
   3.1 Obtaining mtoc++ via the GIT versioning system .................. 8
   3.2 mtoc++ 1.5 .............................................................. 8

4 mtoc++ license conditions 8

5 mtoc++ installation instructions 9
   5.1 Software requirements and recommendations ........................ 9
   5.2 Using precompiled binaries ......................................... 9
      5.2.1 Windows users .................................................... 10
      5.2.2 Unix binaries .................................................... 10
   5.3 Compiling mtoc++ from source ..................................... 10
      5.3.1 Windows platforms .............................................. 10
      5.3.2 Unix platforms ................................................ 11
      5.3.3 Apple hints ..................................................... 11
      5.3.4 CMake options: Installation folders and customization ....... 11
   5.4 Testing ................................................................. 12

6 Tips for doxygen usage with mtoc 12
   6.1 Feature and change tracking information .......................... 12
      6.1.1 New feature and change log commands ........................ 12

7 Configuration and use of mtoc++ 13
   7.1 Documentation creation ............................................. 13
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1.1</td>
<td>Using the MatlabDocMaker</td>
<td>13</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Using mtoc++ directly</td>
<td>14</td>
</tr>
<tr>
<td>7.1.3</td>
<td>Using the python script from a unix shell</td>
<td>14</td>
</tr>
<tr>
<td>7.2</td>
<td>Configuring mtoc++ and doxygen</td>
<td>15</td>
</tr>
<tr>
<td>7.2.1</td>
<td>Configuration options for doxygen</td>
<td>15</td>
</tr>
<tr>
<td>7.2.2</td>
<td>Configuration options for the mtoc++ filter</td>
<td>15</td>
</tr>
<tr>
<td>7.2.3</td>
<td>Extending default LaTeX environment for doxygen</td>
<td>17</td>
</tr>
<tr>
<td>7.2.4</td>
<td>Fake classes for typical MatLab data types</td>
<td>17</td>
</tr>
<tr>
<td>8</td>
<td>Troubleshooting mtoc++</td>
<td>18</td>
</tr>
<tr>
<td>8.1</td>
<td>Configuration</td>
<td>18</td>
</tr>
<tr>
<td>8.2</td>
<td>Issues finding binaries (MAC)</td>
<td>18</td>
</tr>
<tr>
<td>8.3</td>
<td>Debugging mtoc++</td>
<td>18</td>
</tr>
<tr>
<td>9</td>
<td>mtoc++ Developers</td>
<td>18</td>
</tr>
<tr>
<td>9.1</td>
<td>Martin Drohmann</td>
<td>18</td>
</tr>
<tr>
<td>9.2</td>
<td>Daniel Wirtz</td>
<td>19</td>
</tr>
<tr>
<td>10</td>
<td>Todo List</td>
<td>19</td>
</tr>
<tr>
<td>11</td>
<td>Namespace Index</td>
<td>19</td>
</tr>
<tr>
<td>11.1</td>
<td>Packages</td>
<td>19</td>
</tr>
<tr>
<td>12</td>
<td>Hierarchical Index</td>
<td>19</td>
</tr>
<tr>
<td>12.1</td>
<td>Class Hierarchy</td>
<td>19</td>
</tr>
<tr>
<td>13</td>
<td>Class Index</td>
<td>20</td>
</tr>
<tr>
<td>13.1</td>
<td>Class List</td>
<td>20</td>
</tr>
<tr>
<td>14</td>
<td>Namespace Documentation</td>
<td>21</td>
</tr>
<tr>
<td>14.1</td>
<td>examples Namespace Reference</td>
<td>21</td>
</tr>
<tr>
<td>14.1.1</td>
<td>Detailed Description</td>
<td>21</td>
</tr>
<tr>
<td>15</td>
<td>Class Documentation</td>
<td>21</td>
</tr>
<tr>
<td>15.1</td>
<td>AccessStruct Struct Reference</td>
<td>21</td>
</tr>
<tr>
<td>15.1.1</td>
<td>Detailed Description</td>
<td>22</td>
</tr>
<tr>
<td>15.2</td>
<td>cell Class Reference</td>
<td>22</td>
</tr>
<tr>
<td>15.2.1</td>
<td>Detailed Description</td>
<td>22</td>
</tr>
<tr>
<td>15.3</td>
<td>char Class Reference</td>
<td>22</td>
</tr>
<tr>
<td>15.3.1</td>
<td>Detailed Description</td>
<td>22</td>
</tr>
<tr>
<td>15.4</td>
<td>colvec Class Reference</td>
<td>22</td>
</tr>
<tr>
<td>15.4.1</td>
<td>Detailed Description</td>
<td>22</td>
</tr>
<tr>
<td>15.5</td>
<td>ConfFileScanner Class Reference</td>
<td>23</td>
</tr>
</tbody>
</table>
1 mtoc++ 1.5 Software Documentation

1.1 Introduction

The mtoc++ software comprises two programs to build nice Doxygen documentation for Matlab projects.

The filter program 'mtocpp' transforms relevant parts of the M-Files into C++ syntax, which can be parsed by doxygen. The generated html files can be processed by the program 'mtocpp_post' in order to generate documentation looking more like Matlab.

Short mtoc++ feature list:

- Transforms function m-files into standalone functions
- Supports most of the OOP features from Matlab.

See the examples.Class class for detailed examples on how to use mtoc++ with Matlab. The documentation has been created using mtoc++.

1.2 Contents

- Downloading mtoc++
- mtoc++ installation instructions
- Configuration and use of mtoc++
- Tips for doxygen usage with mtoc
- Changes and new features in mtoc++
- mtoc++ license conditions
- Troubleshooting mtoc++

1.3 Downloads

For all available downloads check Downloading mtoc++. The most current (official) release can be downloaded below.

Documentation

This documentation can also be downloaded as (doxygen/mtoc++ generated) PDF.

Release 1.5 (2013/06/25)

- GitHub
- Source tarball
- Windows binaries and tools

2 Changes and new features in mtoc++

Changelog and new feature list for mtoc++

Here are all the changes/new features sorted by versions of mtoc++:

- New features in mtoc++ 0.1 - Changes in mtoc++ 0.1
- New features in mtoc++ 1.2 - Changes in mtoc++ 1.2
• New features in mtoc++ 1.3 - Changes in mtoc++ 1.3
• New features in mtoc++ 1.4 - Changes in mtoc++ 1.4
• New features in mtoc++ 1.5 - Changes in mtoc++ 1.5

Attention

The repeated occurrence of the new features/changes in this specific site below is just due to the fact that the mtoc++ features/changes themselves have to be written down somewhere. Under usual circumstances those tags below would be placed inside the MatLab files/functions/classes where the actual change happened; see the comments from the MatlabDocMaker as an example. So the list below is not necessarily complete, but the sites referenced above contain all new features / changes!

New in 1.5 (Daniel Wirtz, 2013-07-22) Added a FAQ: faq

Change in 1.5 (Daniel Wirtz, 2013-06-25) mtocpp_post/postprocess.rl: Increased robustness. If processing a directory, the postprocessor does not completely stop on (rare) parse errors but continues to loop through the directory. This way all files are at least tried to be processed.

New in 1.5 (Daniel Wirtz, 2013-02-21) Added '.*.mex' files to the types of files parsed by default in Doxyfile.template

Change in 1.4 (Daniel Wirtz, 2012-11-19) mtocpp_post/postprocess.rl: Re-Added the possibility to directly specify a file target instead of a whole folder.

Change in 1.4 (Daniel Wirtz, 2012-10-17)

  • Using the new css-style from doxygen 1.8 for own docs
  • Added some troubleshooting feedback
  • Checked the CMake procedure on a Mac platform (MacBook pro), worked neatly.
  • Added an extra section Using mtoc++ directly for instructions on how to directly use mtoc++
  • Fixed broken links to MatLab method/property attributes in mtoc++ output

Change in 1.4 (Daniel Wirtz, 2012-09-27) Optimized compilation under Visual Studio 2010, now can also build the mtoc++ documentation locally.

Change in 1.4 (Martin Drohmann, 2012-09-27)

  • Added check for HAVE_DOT in doxygen.conf.in
  • Bugfix for function-only M-files reported by Francois Rongere

New in 1.4 (Martin Drohmann, 2012-02-17)

  • Started mtoc++ 1.4.
  • Added alias for an "events" tag, creating a page of all events in default documentation
  • Changed naming convention for alias-tags new and change as newer doxygen versions seem not to recognize `\1\2`-like combinations of arguments any more (?) now pages named "newfeat\_1 \_2" with underscore are created, please update your static references in your misc documentation files

Change in 1.3 (Daniel Wirtz, 2012-01-16) Changed the SHOW_FILES default value in the doxygen configuration file from "NO" to "YES".

Change in 1.3 (Daniel Wirtz, 2012-01-16) Bugfix: The setting EXTRA_PACKAGES in the doxygen configuration file was given the wrong path format, as latex follows the unix file separation using only forward-slashes "/", so the inclusion failed on Windows platforms. We fixed this by passing the correctly transformed path. Also a new placeholder "_FileSep_" which is being processed by MatlabDocMaker (any any tools to come) and set to the correct file separator character for your platform.
Change in 1.3  (Daniel Wirtz, 2012-01-14) Bugfix: Moved the mtoc++ developers page declaration into a separate file inside the tools/config folder, so that error messages like "changelog1:13: warning: unable to resolve reference to 'dw' for \ref command" do not appear anymore.

Change in 1.3  (Daniel Wirtz, 2011-12-08) Bugfix: The CUSTOM_DOC_DIR path is not longer extended by a docs/ folder.

Change in 1.3  (Daniel Wirtz, 2011-11-29) Added the new fake classes varargin and varargout to the class_substitutes.c file with links to the MatLab online documentation.

New in 1.3  (Daniel Wirtz, 2011-11-28) Started mtoc++ 1.3.

New in 1.2  (Daniel Wirtz, 2011-11-27) Reordered the Doxyfile.m4 so that changes from our side are all collected to the bottom. This makes keeping custom settings over different versions easier.

New in 1.2  (Daniel Wirtz, 2011-11-25) Included a file class_substitutes.c into the config directory that introduces fake classes for common matlab data types.

Change in 1.2  (Martin Drohmann, 2011-11-17) Updated the test reference files

New in 1.2  (Daniel Wirtz, 2011-11-07) Created the initial mtoc++ documentation structure

Change in 1.2  (Daniel Wirtz, 2011-11-07) Reordered the source code files and tools in more concise folders.

2.1  New features in mtoc++ 0.1

Demo features of the demo classes and examples

See also Changes in mtoc++ 0.1

Page Tips for doxygen usage with mtoc

   (Daniel Wirtz, undated) Added a fancy new feature! (New feature Example)
   (Daniel Wirtz, 2011-01-01) Added a fancy new feature on new year’s! (New feature Example)

2.2  Changes in mtoc++ 0.1

Demo changes of the demo classes and examples

See also New features in mtoc++ 0.1

Page Tips for doxygen usage with mtoc

   (Daniel Wirtz, undated) Changed foo to bar! (Changelog Example)

2.3  New features in mtoc++ 1.2

First "stable" release with windows/unix support.

See also Changes in mtoc++ 1.2

Page Changes and new features in mtoc++

   (Daniel Wirtz, 2011-11-27) Reordered the Doxyfile.m4 so that changes from our side are all collected to the bottom. This makes keeping custom settings over different versions easier.
   (Daniel Wirtz, 2011-11-25) Included a file class_substitutes.c into the config directory that introduces fake classes for common matlab data types.
   (Daniel Wirtz, 2011-11-07) Created the initial mtoc++ documentation structure
Class **MFileScanner**

(Martin Drohmann, 2011-11-17) New config flag COPY_TYPIFIED_FIELD_DOCU which allows to toggle the automatic insertion of required fields for method parameters. This flag sets whether the documentation of fields in 'Required fields of param', 'Optional fields of param' or 'Generated fields of retval' shall be copied in case the Parameter 'param' or 'retval' have a type.

2.4 Changes in mtoc++ 1.2

First "stable" release with windows/unix support.

See also New features in mtoc++ 1.2

Page **Changes and new features in mtoc++**

(Martin Drohmann, 2011-11-17) Updated the test reference files

(Daniel Wirtz, 2011-11-07) Reordered the source code files and tools in more concise folders.

Class **MFileScanner**

(Martin Drohmann, 2011-11-28) Allow long (including line breaks) default values for properties


(Martin Drohmann, 2011-11-17) Fixed a bug that messed up the documentation if a new line was started after a @type tag and added a test case to classA.m

(Martin Drohmann, 2011-11-17) Non-standard access modifier strings are now separated by a comma

(Martin Drohmann, 2011-11-17) Fixed a parse error occurring with the new ∼-notation in newer MatLab versions. Calls like foo = bar(par1, ∼, par3) now work.

(Martin Drohmann, 2011-11-17) The order of @default and @type tags in parameters (if occurring) is no longer fixed.

2.5 New features in mtoc++ 1.3

Improved stability for Windows platforms, event handling

See also Changes in mtoc++ 1.3

Page **Changes and new features in mtoc++**

(Daniel Wirtz, 2011-11-28) Started mtoc++ 1.3.

Class **MFileScanner**

(Martin Drohmann, 2012-02-03) Print a warning message to stderr when optional parameter in methods of functions are not documented with default values.

(Martin Drohmann, 2012-01-10) "Bugfix": Allowing the use of the AbortSet tag in property declarations, however, to extra action (e.g. inserting a note in documentation) is taken so far.

(Martin Drohmann, 2011-12-16) Allowing multiple lines for default values in property comments & code and added a test case.

2.6 Changes in mtoc++ 1.3

Improved stability for Windows platforms, event handling

See also New features in mtoc++ 1.3

Page **Changes and new features in mtoc++**

(Daniel Wirtz, 2012-01-16) Changed the SHOW_FILES default value in the doxygen configuration file from "NO" to "YES".

(Daniel Wirtz, 2012-01-16) Bugfix: The setting EXTRA_PACKAGES in the doxygen configuration file was given the wrong path format, as latex follows the unix file separation using only forwardslashes "/", so the inclusion
failed on Windows platforms. We fixed this by passing the correctly transformed path. Also a new placeholder 
"_FileSep_" which is being processed by MatlabDocMaker (any any tools to come) and set to the correct file 
separator character for your platform.

(Daniel Wirtz, 2012-01-14) Bugfix: Moved the mtoc++ developers page declaration into a separate file inside
the tools/config folder, so that error messages like "changelog1:13: warning: unable to resolve reference to ‘dw’
for ‘ref command” do not appear anymore.

(Daniel Wirtz, 2011-12-08) Bugfix: The CUSTOM_DOC_DIR path is not longer extended by a docs/ folder.

(Daniel Wirtz, 2011-11-29) Added the new fake classes varargin and varargout to the class_substitutes.c file
with links to the MatLab online documentation.

Class MFileScanner

(Martin Drohmann, 2012-02-17) added events section
(Martin Drohmann, 2012-02-15) improved documentation for dependent flags
(Martin Drohmann, 2012-02-15) remove =0 for purely virtual class methods
(Martin Drohmann, 2012-02-15) We totally ignore functions which are locally defined inside another function.
(Martin Drohmann, 2012-02-15) Added config GENERATE_SUBFUNTION_DOCUMENTATION and format for
output of subfunctions.
(Martin Drohmann, 2012-02-03) Bugfix: Default values were printed twice if documented in the documenta-
tion block and given as property default values. Now, the documented default value is preferred.
(Martin Drohmann, 2012-02-03) Improved the automatic documentation text for MATLAB specific attributes of
properties and methods, add a link to the online MATLAB documentation.
(Martin Drohmann, 2012-01-10) Some minor modifications for the postprocessor regarding dots ‘.’ and ‘::’
(Martin Drohmann, 2011-12-16) Bugfix: On Windows platforms the wrong getcwd command was issued and
is now fixed.
(Martin Drohmann, 2011-12-13) Bugfix: Now handling the Abstract property correctly (was previously added
for SetObservable declarations due to copy&paste)
(Martin Drohmann, 2012-01-13) Added a test case for default properties containing semicolons
(Martin Drohmann, 2012-01-13) Changed format for documentation of default properties and parameters
(Martin Drohmann, 2012-01-13) Default arguments for properties are added to the properties documentation
block
(Martin Drohmann, 2012-01-13) Bugfix: observable properties have been documented as abstract ones.
(Martin Drohmann, 2011-12-13) Adding a bold "Default:" line in property documentation blocks if a default
value/default tag is set in either code or property comment.
(Martin Drohmann, 2011-12-04) Bugfix reported by Evgeny Pr on mathworks: allow property definitions not
ended by semicolons.

2.7 New features in mtoc++ 1.4

Included basic source browsing, handling of varargin-parameters, basic LaTeX generation support

See also Changes in mtoc++ 1.4

Page Changes and new features in mtoc++

(Martin Drohmann, 2012-02-17)

• Started mtoc++ 1.4.
• Added alias for an "events" tag, creating a page of all events in default documentation
• Changed naming convention for alias-tags new and change as newer doxygen versions seem not to recog-
nize \1\2-like combinations of arguments any more (?) now pages named "newfeat\1_\2" with underscore
are created, please update your static references in your misc documentation files

Generated on Mon Jul 27 2015 12:53:15 for mtoc++ by Doxygen
2.8 Changes in mtoc++ 1.4

Many bugfixes due to detailed feedback! Thanks!
See also New features in mtoc++ 1.4

Page Changes and new features in mtoc++
(Daniel Wirtz, 2012-11-19) mtocpp_post/postprocess.rl: Re-Added the possibility to directly specify a file target instead of a whole folder.
(Daniel Wirtz, 2012-10-17)
- Using the new css-style from doxygen 1.8 for own docs
- Added some troubleshooting feedback
- Checked the CMake procedure on a Mac platform (MacBook pro), worked neatly.
- Added an extra section Using mtoc++ directly for instructions on how to directly use mtoc++
- Fixed broken links to MatLab method/property attributes in mtoc++ output

(Daniel Wirtz, 2012-09-27) Optimized compilation under Visual Studio 2010, now can also build the mtoc++ documentation locally.
(Martin Drohmann, 2012-09-27)
- Added check for HAVE_DOT in doxygen.conf.in
- Bugfix for function-only M-files reported by Francois Rongere

Class MFileScanner
(Daniel Wirtz, 2012-10-22) Added an additional comment about indentation in matlab on parse errors
(Martin Drohmann, 2012-10-19) prettify the output of postprocessed source for source code browsing in doxygen. Now, we recommend the usage of the FILTER_SOURCE_FILES doxygen switch.
(Martin Drohmann, 2012-10-17) implemented varargin handling by Matlab inputParser, as suggested here:
http://www.mathworks.de/de/help/matlab/ref/inputparser.parse.html
(Martin Drohmann, 2012-02-24) ignore comments in front of classdef (fixes Grrr message from Jesse Hopkins)

2.9 New features in mtoc++ 1.5

Current development
See also Changes in mtoc++ 1.5

Page Changes and new features in mtoc++
(Daniel Wirtz, 2013-07-22) Added a FAQ: faq
(Daniel Wirtz, 2013-02-21) Added ’*.mex’ files to the types of files parsed by default in Doxyfile.template

2.10 Changes in mtoc++ 1.5

Current development
See also New features in mtoc++ 1.5

Page Changes and new features in mtoc++
(Daniel Wirtz, 2013-06-25) mtocpp_post/postprocess.rl: Increased robustness. If processing a directory, the postprocessor does not completely stop on (rare) parse errors but continues to loop through the directory. This way all files are at least tried to be processed.
Class **MFileScanner**

(Martin Drohmann, 2013-08-20) Bugfix: incorrect usage of addOptional does not break mtocpp (pointed out by Alexander on FEX)

(Martin Drohmann, 2013-07-08) Bugfix: End of line comment in line before function ending broke the parser.

(Martin Drohmann, 2013-05-16) Added code removal feature as requested by Maxime.

(Martin Drohmann, 2013-04-01) Parse event attributes (ListenAccess, NotifyAccess, Hidden) correctly - as pointed out by Evgeny Pr on FEX

(Daniel Wirtz, 2013-03-28) Merged together simultaneous contributions from Martin and me regarding the F-EX feedback from Pete. Now also allowing e.g. ‘protected’ as access modifier string and the ‘Abstract’ class modifier is also recognized.

## 3 Downloading mtoc++

**mtoc++ download sources and variants**

The sources for mtoc++ are accessible as

- Source tarball
- Windows binaries
- Via the GIT versioning system

Once you’ve obtained your copy check out the **mtoc++ installation instructions**

**Note**

For Windows binaries from this page you might need to install the Microsoft Visual C++ 2012 redistributables.

### 3.1 Obtaining mtoc++ via the GIT versioning system

In order to always stay up-to-date with the latest (development) mtoc++ versions, simply pull it from our GIT repository and recompile it when a new release is available. For more information about GIT checkout their website

The GIT repository can currently be found at https://github.com/mdrohmann and a direct clone is possible via

```
git clone https://github.com/mdrohmann/mtocpp.git mtocpp
```

### 3.2 mtoc++ 1.5

- This documentation as PDF.
- Source tarball
- Windows binaries (32/64bit) and tools
- Windows binaries (32/64bit)

## 4 mtoc++ license conditions

This software is available under the BSD license http://www.opensource.org/licenses/bsd-license.php

Copyright (c) 2010-2013, Martin Drohmann, Daniel Wirtz All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5 mtoc++ installation instructions

Make sure you have the latest version of mtoc++, see Downloading mtoc++. Next step after installation is Configuration and use of mtoc++

5.1 Software requirements and recommendations

The following programs need to be available on your machine in order to use mtoc++:

- doxygen (>=1.8.1): mtoc++ is a filter for doxygen. If not yet available, get it at http://www-doxygen.org

The following programs will highly improve your documentation creation experience if available:

- dot: A Graphviz tool that allows doxygen to create nice graphics for inheritance trees and collaboration diagrams.
- latex: Required to use LaTeX processing capabilities of doxygen (e.g. http://www-latex-project.org/ftp.html). mtoc++ comes with some markups for better latex inclusion into the text flow. Also, easy inclusion of external latex sources and styles is included in mtoc++'s tools.
- ghostscript: If using formulas with doxygen and you are not using pdflatex or are on a windows machine, this is a prerequisite (see http://www.stack.nl/~dimitri/doxygen/install-bin_windows)

If you want to build mtoc++ from source, you will also need:

- dirent.h (We included a Visual Studio API implementation by Tony Ronkko for Windows)

5.2 Using precompiled binaries
5.2.1 Windows users

If you are a windows user you can directly download the binaries at Downloading mtoc++. Then simply place the binaries in a folder of your choice and add the folder to the PATH environment variable. If you intend to use the MatlabDocMaker, you can also copy the mtoc++ binaries into the "documentation configuration files" folder for your/each project, this path will be added to PATH by MatLab locally.

Note

Depending on your system setup, you might need to install the Microsoft Visual C++ 2010 redistributables, which can be found here.

Attention

ttoc++ as well as doxygen expect all required programs (see Software requirements and recommendations) to be available via the PATH environment variable, e.g. latex.exe or gswin32c.exe must be present in order for doxygen to work with LaTeX output. Make sure that you have all requirements available, otherwise doxygen or the MatlabDocMaker will complain soon enough. You can check/change your Windows PATH environment variable via the sequence

Computer \ Properties \ Advanced system settings \ Environment Variables \ Edit Path

We are trying to always compile current Windows binaries for mtocpp and mtocpp_post and include them for direct download.

5.2.2 Unix binaries

For unix users we recommend to compile the sources following Compiling mtoc++ from source. However, we also plan to provide some precompiled linux binaries/packages soon. If you find a matching choice you can use it and all you have to do is to ensure that the binaries can be found on the environment PATH.

5.3 Compiling mtoc++ from source

Please check the Software requirements and recommendations when you intend to build mtoc++ yourself.

mtoc++ is built using the cmake (cross-platform make) tool. This tool is available for both unix and Windows, however, we only tested compiling our sources on linux and MS Visual Studio 2010.

5.3.1 Windows platforms

For Windows compilation, you need a Windows C++ compiler (e.g. MinGW or Visual Studio). Then running the CMake GUI allows you to choose a compiler, specify any CMake configuration settings and create the makefiles/- Visual Studio projects needed for compilation.

Furthermore, we’re using the dirent.h library for file access. As this is a linux library we’ve included a file dirent_msvc.h in our source, which implements the dirent api for Microsoft Visual Studio and was written by Tony Ronkko. More information and downloads can be found at http://www.softagalleria.-net/dirent.php.

Note

On Windows, you can build both 32bit and 64bit versions. If you build with Visual Studio, in recent CMake versions you need to specify the target architecture already when choosing the generator ("Visual Studio 10 / Visual Studio 10 Win64"). This sets up the VS2010 project with the correct platforms. In general, you can of course also use 64bit binaries from ragel and doxygen, but this is not required for successful 64bit-compilation of mtoc++. 
5.3 Compiling mtoc++ from source

5.3.2 Unix platforms

The following procedure is an example of how to compile mtoc++ on a linux machine:

```bash
# Compiling mtoc++ from source

tar -xcvf mtocpp.tar.gz
\* cd mtocpp
\* # Create build folder (optional, but more clean)
\* mkdir build
\* cd build
\* # Run cmake
\* cmake ..
\* make install
```

Attention

Please be aware that, depending on your installation location, you might need different access/write permissions. For most cases, a

```bash
sudo make install
```

will do the job if the above snippet fails.

5.3.3 Apple hints

For installation under recent Apple OS like 10.8.2, the MacPorts project is a very useful tool to obtain prerequisites for mtoc++ compilation. Once installed, get ragel and doxygen via

```bash
sudo port
> install ragel
> install doxygen
```

5.3.4 CMake options: Installation folders and customization

Note

These options are explained for the linux case, for windows the CMake GUI allows to set the relevant options.

The default value for the install prefix is /usr/local, so the mtocpp binaries mtocpp and mtocpp_post go to /usr/local/bin and the documentation is created inside /usr/local/share/doc/mtocpp.

If you want the "make install" command to copy the binaries and documentation to different locations, you can choose them by setting the following variables:

- **CMAKE_INSTALL_PREFIX**: Set this to whatever location you want mtoc++ to be installed. Note that the binaries are effectively copied into "CMAKE_INSTALL_PREFIX/bin" in order to comply with linux standards.

- **CUSTOM_DOC_DIR**: This value is "CMAKE_INSTALL_PREFIX/share/doc/mtocpp" per default.

So typing

```bash
cmake -DCMAKE_INSTALL_PREFIX="/my/root/dir" -DCUSTOM_DOC_DIR="/my/docs"
```

will copy the binaries to /my/root/dir/bin and the documentation to /my/docs.

If you left the **CUSTOM_DOC_DIR** flag empty the documentation would have gone to /my/root/dir/share/doc/mtocpp
5.4 Testing

mtoc++ comes with some unit tests to check for e.g. successful compilation. Run the tests by typing

```
make test
```

in the same folder where you called cmake.

On Windows, dependent on your compiler, you will either have makefiles for the test cases or a separate Visual Studio project to run the tests.

Have fun!

6 Tips for doxygen usage with mtoc

6.1 Feature and change tracking information

6.1.1 New feature and change log commands

New features can be tracked version-based via using

```
@new{<mainversionnumber>, <mainversionnumber>, <developerkey>
[, <date>]} <description>
```

For example, writing

```
@new{0,1,dw} Added a fancy new feature! (New feature Example)
```

results in

**New in 0.1** (Daniel Wirtz, undated) Added a fancy new feature! (New feature Example)

To include a date write

```
@new{0,1,dw,2011-01-01} Added a fancy new feature on new year's! (New feature Example)
```

results in

**New in 0.1** (Daniel Wirtz, 2011-01-01) Added a fancy new feature on new year's! (New feature Example)

and a new related page called newfeat01 listing these items. To refer to that Changelog page, use the keyword 'newfeat' together with both plainly concatenated numbers:

```
@ref newfeat01
```

gives newfeat01

Changes can be tracked version-based via using

```
@change{<mainversionnumber>, <mainversionnumber>, <developerkey>
[, <date>]} <change-text>
```

For example, writing

```
@change{0,1,dw} Changed foo to bar! (Changelog Example)
```

results in

**Change in 0.1** (Daniel Wirtz, undated) Changed foo to bar! (Changelog Example)

The optional date works same as with the '@new' command. The related page keys for changes are composed by the keyword 'changelog' and both plainly concatenated numbers (similar to the new feature keys).
7 Configuration and use of mtoc++

Help on how to use the tools coming with mtoc++

Make sure you have followed the mtoc++ installation instructions.

Contents

- Documentation creation
  - Using the MatlabDocMaker
  - Using mtoc++ directly
  - Using the python script from a unix shell
- Configuring mtoc++ and doxygen
  - Configuration options for doxygen
  - Configuration options for the mtoc++ filter
  - Extending default LaTeX environment for doxygen

7.1 Documentation creation

As mtoc++ itself is only a filter to plug into doxygen, there is little sense in calling the binaries directly.

Thus, mtoc++ comes with a series of tools that take over the documentation generation process for different interfaces.

Those tools can be found inside the \(<\text{mtoc++-source-dir}>\)/tools folder.

Note

At some stage you will need to have access to the involved binaries like doxygen, mtocpp, mtocpp_post or latex. It is your responsibility to ensure the availability of the binaries in whatever environment you want to create the documentation. The most obvious way is to place all binaries inside a directory contained in your local PATH variable (both unix/windows). We’ve had reported issues with MAC users, that dont have the environment set when launching MatLab from the Dock. See Troubleshooting mtoc++ for more information.

7.1.1 Using the MatlabDocMaker

The most convenient way of using mtoc++ within your matlab project is to use the MatlabDocMaker class coming with mtoc++. The MatlabDocMaker is a MatLab native class that can be directly used from within MatLab in order to create the project documentation.

Follow these simple steps in order to quickly get your first documentation:

- Place the MatlabDocMaker.m file somewhere on your project’s MatLab path.
- Change the MatlabDocMaker.getProjectName method to return your project’s name
- Copy the contents of the \(<\text{mtoc++-source-dir}>\)/tools/config folder into e.g. a subfolder of your MatLab project
- Call the MatlabDocMaker.setup method and use the folder from the previous step as your “documentation configuration files directory”.
- Use the MatlabDocMaker.create method to generate your documentation and look at it in a web browser.

See the MatlabDocMaker class description for more details on how to use it.
Note

You may of course keep the MatlabDocMaker.m and the configuration files where you initially placed your mtoc++ source and point to the appropriate directories during setup. However, if you want to use multiple projects with mtoc++ you probably want to have different configurations for each project, so that is why we recommend to create local copies of your tools and configuration within each project. (The MatlabDocMaker stores its setting dependent on the name you specify for the project!) The way the MatlabDocMaker works it can be easily inserted into whatever versioning system your project uses. As it stores important folders in MatLab preferences each developer will still have his local documentation settings (after running MatlabDocMaker.setup on each machine, of course).

7.1.2 Using mtoc++ directly

Okay, so you’re a crack and want to control everything. That’s fine with us! In this case we also assume you’re familiar with whatever your operating environment is and you have solid knowledge of what’s going on. First, you could simply reverse-engineer what the MatlabDocMaker is doing (it automatically generates and inserts the correct scripts read by doxygen), otherwise, here are the basic steps required to get started with mtoc++ directly. In short, this happens by including mtoc++ as a filter for *.m files:

- Compile things as necessary and make binaries accessible
- Modify your doxygen configuration file:
  - Setup your doxygen as usual, including the sources and output directories
  - Make doxygen parse Matlab files
  - Register mtoc++ as a filter for those files
  - If you have a custom mtocpp.conf you want mtoc++ to use, you need to create a shell/batch script that passes this file to mtoc++ and use this file as filter executable
  - Check if you are using latex-features of mtoc++, if so, add latex-support and provide necessary style files
- Run doxygen
- Run mtocpp_post passing the folder containing your HTML output as argument
- Look at some nice documentation, be happy!
- If you’re not happy, try starting with the provided Doxyfile.template in the tools/ directory and inserting proper values for all the placeholders we’re using. Everything related to mtoc++ has been put to the very bottom of the file, most critically:

```
* EXTENSION_MAPPING = .m=C++
* INPUT = _SourceDir_ _ConfDir_
* FILE_PATTERNS = *.m
* FILTER_PATTERNS = *.m="_ConfDir_"_FileSep_"_MTOCFILTER_"
```

Here, the underscored values need to be replaced manually in order to insert the correct values. Essentially:

- EXTENSION_MAPPING tells doxygen to regard .m-Files as if they were C++ files, style-wise.
- INPUT tells doxygen where to look for files.
- FILE_PATTERNS lets doxygen also look for .m-Files.
- FILTER_PATTERNS is the most important line of the configuration. Here, you need to define scripts that should be called by doxygen before certain files are processed.

7.1.3 Using the python script from a unix shell

Todo python script, yet to come
7.2 Configuring mtoc++ and doxygen

As the configuration of doxygen/mtoc++ is independent from the actual tool used we will explain it separately. The involved files can again be found inside the /tools/config folder.

- Doxyfile.template - Configuration options for doxygen
- mtocpp.conf - Configuration options for the mtoc++ filter
- latexextras.template - Extending default LaTeX environment for doxygen
- class_substitutes.c - Fake classes for typical MatLab data types

Attention

USING MTOC++ DOES NOT EXCLUDE THE REQUIREMENT TO KNOW AND UNDERSTAND DOXYGEN ITSELF!

The settings in the "Doxygen.template" file inside the /tools/config folder are a default configuration for Doxygen which we thought might be useful in a MatLab setting/project and contains some changes in order to make mtoc++ run together with doxygen. We've had lots of feedback and problem reports which actually had to do with settings purely regarding doxygen, so we strongly recommend having a look through Configuration options for doxygen and the references therein before contacting us. Thanks!

7.2.1 Configuration options for doxygen

The Doxyfile.template file uses placeholders for specific folders etc. and contains any other configuration settings you want doxygen to use. This way, the configuration files can be included into the versioning system as local developers paths are stored outside the configuration file and are provided by the different tools coming with mtoc++.

See http://www.stack.nl/~dimitri/doxygen/config.html for more information on doxygen configuration.

7.2.2 Configuration options for the mtoc++ filter

The file mtocpp.conf contains additional configuration for the mtoc++ parser.

Note

The mtoc++ filter takes exactly two arguments, of which the first is the file to process, and the second is an optional configuration file. So if you dont want to customize mtoc++ because the default settings are just fine, there is nothing to do (you simply can set the filter target in doxygen to the mtocpp binary for the manual config case). Otherwise, if you want to provide a config file to mtoc++, depending on your platform, you have to write a shell/batch script that is included as filter callback in doxygen's configuration file. Inside the script, the first argument is forwarded to mtocpp and the second configuration file path is provided statically in the script.

We recommend to use the MatlabDocMaker tool described in Using the MatlabDocMaker, as it does all that for you.

The following is a short list of options that can be specified in the configuration file for the mtoc++ filter. All options are declared by the syntax

<option> := <value>

and are optional, as the default values are hardcoded into mtoc++.

- ALL - File Patterns
• PRINT_FIELDS - Flag indicating whether automatic struct fields or object member documentation is generated. Default true.

• AUTO_ADD_FIELDS - Flag indicating whether undocumented field names are added to documentation. Default false.

• AUTO_ADD_PARAMETERS - Flag indicating whether undocumented parameters and return values are added to documentation with documentation text equal to the parameter / return value name. Default false.

• AUTO_ADD_CLASS_PROPERTIES - Flag indicating whether undocumented member variables are added to documentation with documentation text equal to the parameter / return value name. Default false.

• AUTO_ADD_CLASSES - Flag indicating whether undocumented classes are added to documentation with documentation text equal to the class name. Default true.

• REMOVE_FIRST_ARG_IN_ABSTRACT_METHODS - Flag indication whether the first argument in abstract non-static methods shall be a this pointer, and therefore removed in the C++ output. Default true.

• ENABLE_OF_TYPE_PARSING - Flag indicating whether the string "of type" is parsed in the the first two lines of comments. This is equivalent to the @type tag, but makes the code more readable at some places. Default true.

• VOID_TYPE_IN_RETURN_VALUES - Flag indicating whether the typename void shall be inserted for return values with no specified type. Default false. PRINT_RETURN_VALUE_NAME - Integer flag indicating whether return value names shall be printed in the function synopsis. If this flag is deactivated only the type names are written. The flag can be set to either 0, 1 or 2 and has default value 2:
  – 0: means that no return value names shall be printed at all.
  – 1: means that return value names shall be printed for return value lists with more than one element only.
  – 2: means that return value names shall be printed always.

Moreover, default descriptions/values for recurring entries like parameters or field names can be specified.

Attention
Note that the configuration file sections for variables above and rules below have to be separated by a single line containing only a double hash `##`. ONLY use `##` for that purpose.

Parameter default descriptions
Use the syntax

\[
\text{add(params) = } \langle \text{param1} \rangle \Rightarrow \"\"\text{Your param1 description text in triple quotes}\"\";
\]

\[
\text{add(params) = } \langle \text{param2} \rangle \Rightarrow \"\"\text{Your param2 description text in triple quotes}\"\";
\]

to add default descriptions to parameters of functions or class members.

Struct field default descriptions
Use the syntax

\[
\text{add(fields) = } \langle \text{field} \rangle \Rightarrow \"\"\text{Your field description text in triple quotes}\"\";
\]

to add default descriptions to fields of any struct or class (identified by a ".fieldname" syntax in the MatLab code)

Extra documentation
Use

\[
\text{add(doc) = \"\"\text{<some extra doc for all files> \"\"));}
\]

to append some extra documentation to each class or files documentation. Use

\[
\text{add(extra) = \"\"\text{<text at end of comments> \"\"));}
\]
7.2 Configuring mtoc++ and doxygen

To append text at the end of any comment.

Global settings for specific files or folder groups

More advanced, those settings above can also be made on a group-based setting. The syntax

* glob = <folder or filename (regexp allowed)> {
*   <expressions as above>
* } glob = <subfolder or files> {
*   <expressions as above>
* }}


* can be used to specify groups of rules that are applied to any matching file or files in folders. Nesting is possible, too.

So for example,

glob = myfile.m { add(params) = param1 => """ param 1 description """; }

would cause mtoc++ to add the description "param 1 description" to any parameter called param1 of a method/function inside the file myfile.m.

Attention

Having common field names specified centrally is a quite convenient way to autogenerate documentation. However, if you use e.g. the same parameter name in a different meaning and forget to explicitly specify the parameter documentation, the default values will be inserted. This possibly leads to more confusion for users than it does help. Furthermore, not specifying the parameters in the local comments decreases readability of the code. One of mtoc++'s main advantages in combination with doxygen is that code can be commented highly readable in-place!

See the file itself for more detailed configuration options and examples.

7.2.3 Extending default LaTeX environment for doxygen

The latexextras.template file is processed and included into the latex environment available to doxygen during the documentation creation. Insert here any commands or packages that you want latex to know for your documentation formulas.

Attention

When having errors inside an LaTeX formula, doxygen will complain upon finishing and tell you to look into the _formulas/log/.tex file in the documentation output folder. THIS WARNING COMES ONLY ONE TIME! Upon the next creation run, only changed/new formulas will be re/generated. We considered deleting all formula pngs before each re-creation, but decided not to do this for performance issues. So just make sure you react to latex typos/errors immediately.

The default packages that are included by the latexextras.template are

\usepackage{amsmath}
\usepackage{amssymb}
\usepackage{amsfonts}
\usepackage{subfig}
\usepackage{bbm}

7.2.4 Fake classes for typical MatLab data types

The file class_substitutes.c includes some class descriptions for typical MatLab data types like handle or logical, but also introduces custom types like colvec or rowvec that can be used with the @type tag for property, parameter or return value types.

Add new classes to this file or change existing ones as you need.
8 Troubleshooting mtoc++

Some hopefully useful hints when things don't go as they should!

8.1 Configuration

Attention

The first and most important message: **KNOWLEDGE OF DOXYGEN IS ESSENTIAL!**

mtoc++ is designed as a filter for MatLab m-files, so that they can be processed by doxygen as if they were C source files. Everything else regarding tags, conventions and possible formatting of display is completely defined by Doxygen. So, unless explicitly explained as "feature" of mtoc++ here, one should look into Doxygen's documentation pages first before complaining about some stuff that mtoc++ surprisingly cannot do.

Check out the Using mtoc++ directly section for details on how mtoc++ works.

8.2 Issues finding binaries (MAC)

Thanks to a report from K. Kearney to resolve path issues on MAC platforms:

"After building, I added `<mydir>/mtoc++_1.4/tools` to my Matlab path and tried to run MacDocMaker.setup. I encountered an issue where Matlab couldn’t locate either mtocpp or latex. I think this is a Mac-specific issue; when you start Matlab in the standard way, from the Dock (rather than through the command-line matlab command), the shell it starts doesn’t run any configuration scripts (.bashrc, .back_profile, etc) or set system paths. I think some versions of Matlab have a .matlabrc.sh file that can be modified to set a PATH, and I’ve seen something on the newsgroup about .plist files, but I just force the Matlab shell it to match my other Terminal sessions by adding the following lines to the top of the matlab shell script (`<matlabroot>/bin/matlab`):

* source ~/.bash_profile
* source /etc/bashrc
* source /etc/profile

With that change (and after restarting Matlab), I was able to successfully run MatlabDocMaker.setup, and then MatlabDocMaker.create on a test directory."

8.3 Debugging mtoc++

For hard cases like segfaults there is also hope!

You can build your mtoc++ binaries with the `Debug` build type (starting in the source folder):

* mkdir build
* cd build
* cmake -DCMAKE_BUILD_TYPE=Debug ..
* make

Then, send the compiled binaries along with the used source code to us and we will try to figure out what the heck is wrong with it!

9 mtoc++ Developers

9.1 Martin Drohmann

[http://wwwmath.uni-muenster.de/u/martin.drohmann](http://wwwmath.uni-muenster.de/u/martin.drohmann)
- Lead Programmer
- Ragel expert :-)

9.2 Daniel Wirtz

http://www.agh.ians.uni-stuttgart.de/orga/people/wirtz

- Programming
- Maintenance Matlab File Exchange
- Windows binaries (VS2010)

10 Todo List

Page Configuration and use of mtoc++

python script, yet to come

11 Namespace Index

11.1 Packages

Here are the packages with brief descriptions (if available):

- examples
  A namespace/package for example files

12 Hierarchical Index

12.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- AccessStruct
- cell
- char
- colvec
- ConfFileScanner
- double
- function_handle
- handle
- integer
- logical
- matrix
13 Class Index

13.1 Class List

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

AccessStruct 21

cell
   A MatLab cell array or matrix 22

char
   A MatLab character array 22

colvec
   A matlab column vector 22

ConfFileScanner 23

double
   A double value 23

function_handle
   A MatLab function handle 23

handle
   Matlab’s base handle class (documentation generation substitute) 24

integer
   An integer value 25

ordered_map< ST > 29

ordered_map< DocuBlock > 29

ordered_map< DocuList > 29

ordered_map< std::pair< int, std::string > > 29
14 Namespace Documentation

14.1 examples Namespace Reference

A namespace/package for example files.

14.1.1 Detailed Description

Note

If you use the 'SHOW_NAMESPACES' option in your 'doxygen' configuration file, you need to document all your namespaces/packages at least with a brief description text. This should be done in a separate file.

See Also

'namespaces.c' in the 'share/doc/mtoccpp' directory for details.

15 Class Documentation

15.1 AccessStruct Struct Reference

Public Attributes

- AccessEnum full
• AccessEnum get
• AccessEnum set
• std::vector<std::pair<MatlabAccessEnum, std::string>> classMemberAccess
• MatlabAccessEnum state

Friends

• std::ostream & operator<< (std::ostream &os, AccessStruct &as)

15.1.1 Detailed Description

Definition at line 104 of file mfilescanner.h.
The documentation for this struct was generated from the following file:

• src/mfilescanner.h

15.2 cell Class Reference

A MatLab cell array or matrix.

15.2.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.3 char Class Reference

A MatLab character array.

15.3.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations and represents string-like types.
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.4 colvec Class Reference

A matlab column vector.

15.4.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c
15.5 ConfFileScanner Class Reference

Public Member Functions

• ConfFileScanner (const std::string &filename, const std::string &conffilename)
• int execute ()
• const char * get_conffile ()

Public Attributes

• DocuList param_list_
• DocuList return_list_
• DocuListMap field_docu_
• DocuBlock docuheader_
• DocuBlock docubody_
• DocuBlock docubody_
• DocuBlock docubody_
• GroupSet groupset_
• DocuList vars_

15.5.1 Detailed Description

Definition at line 17 of file confscanner.h.
The documentation for this class was generated from the following file:

• src/confscanner.h

15.6 double Class Reference

A double value.

15.6.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations. The MatLab type
associated with this class is double.
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.7 function_handle Class Reference

A MatLab function handle.

15.7.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations.
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c
15.8 handle Class Reference

Matlab's base handle class (documentation generation substitute)

Public Attributes

- `matlabtypesubstitute addlistener`
  
  Creates a listener for the specified event and assigns a callback function to execute when the event occurs.

- `matlabtypesubstitute notify`
  
  Broadcast a notice that a specific event is occurring on a specified handle object or array of handle objects.

- `matlabtypesubstitute delete`
  
  Handle object destructor method that is called when the object's lifecycle ends.

- `matlabtypesubstitute disp`
  
  Handle object disp method which is called by the display method. See the MATLAB disp function.

- `matlabtypesubstitute display`
  
  Handle object display method called when MATLAB software interprets an expression returning a handle object that is not terminated by a semicolon. See the MATLAB display function.

- `matlabtypesubstitute findobj`
  
  Finds objects matching the specified conditions from the input array of handle objects.

- `matlabtypesubstitute findprop`
  
  Returns a meta.property objects associated with the specified property name.

- `matlabtypesubstitute fields`
  
  Returns a cell array of string containing the names of public properties.

- `matlabtypesubstitute fieldnames`
  
  Returns a cell array of string containing the names of public properties. See the MATLAB fieldnames function.

- `matlabtypesubstitute isvalid`
  
  Returns a logical array in which elements are true if the corresponding elements in the input array are valid handles. This method is Sealed so you cannot override it in a handle subclass.

- `matlabtypesubstitute eq`
  
  Relational functions example. See details for more information.

- `matlabtypesubstitute transpose`
  
  Transposes the elements of the handle object array.

- `matlabtypesubstitute permute`
  
  Rearranges the dimensions of the handle object array. See the MATLAB permute function.

- `matlabtypesubstitute reshape`
  
  Changes the dimensions of the handle object array to the specified dimensions. See the MATLAB reshape function.

- `matlabtypesubstitute sort`
  
  Sort the handle objects in any array in ascending or descending order.

15.8.1 Detailed Description

As doxygen does not know the class "handle" from itself, many classes do not get rendered within the documentation and the correct root class is not even displayed. This workaround guarantees a correct (also graphical) representation of the class hierarchy.

Note here that by having the type handle it could also mean to have a vector or matrix of handles.

Definition at line 91 of file class_substitutes.c.
15.9 integer Class Reference

An integer value.

15.9.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations. Matlab types associated with this class are all int-types (int8, uint8 etc).

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

- tools/config/class_substitutes.c

15.10 logical Class Reference

A boolean value.

15.10.1 Detailed Description

This class can be seen as synonym for boolean values/flags used inside classes. In order to stick with matlab conventions/datatypes, this class was named logical instead of bool or boolean.

This class is an artificially created class in doxygen to allow more precise type declarations.

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

- tools/config/class_substitutes.c
15.11  matrix Class Reference

A matlab matrix.
Inherited by sparsematrix.

15.11.1  Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations
Definition at line 73 of file class_substitutes.c.
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.12  MethodParams Struct Reference

Public Member Functions

• std::string ccprefix ()
• std::string ccpostfix ()
• std::string print_list ()

Public Attributes

• bool abstr
• bool statical
• bool hidden
• bool sealed
• bool test
• bool testMethodSetup
• bool testMethodTeardown

Friends

• std::ostream & operator<< (std::ostream &os, MethodParams &mp)

15.12.1  Detailed Description

Definition at line 201 of file mfilescanner.h.
The documentation for this struct was generated from the following file:

• src/mfilescanner.h

15.13  MFileScanner Class Reference

#include <mfilescanner.h>
Public Types

- typedef std::vector<std::string> DocuBlock
- typedef ordered_map<DocuBlock> DocuList
- typedef ordered_map<DocuList> DocuListMap
- typedef std::map<std::string, DocuBlock> AltDocuList
- typedef std::map<std::string, AltDocuList> AltDocuListMap
- typedef std::set<std::string> GroupSet
- typedef ordered_map<std::pair<int, std::string>> VararginParserValuesType

Public Member Functions

- MFileScanner (std::istream &fin, std::ostream &fout, const std::string &filename, const std::string &confilename, RunMode runmode)
- int execute ()
- DocuList &getParamList ()
- MethodParams &getMethodParams ()
- void end_function ()

15.13.1 Detailed Description

Change in 1.5 (Martin Drohmann, 2013-08-20) Bugfix: incorrect usage of addOptional does not break mtocpp (pointed out by Alexander on FEX)

Change in 1.5 (Martin Drohmann, 2013-07-08) Bugfix: End of line comment in line before function ending broke the parser.

Change in 1.5 (Martin Drohmann, 2013-05-16) Added code removal feature as requested by Maxime.

Change in 1.5 (Martin Drohmann, 2013-04-01) Parse event attributes (ListenAccess, NotifyAccess, Hidden) correctly - as pointed out by Evgeny Pr on FEX

Change in 1.5 (Daniel Wirtz, 2013-03-28) Merged together simultaneous contributions from Martin and me regarding the FEX feedback from Pete. Now also allowing e.g. ‘protected’ as access modifier string and the ‘Abstract’ class modifier is also recognized.

Change in 1.4 (Daniel Wirtz, 2012-10-22) Added an additional comment about indentation in matlab on parse errors

Change in 1.4 (Martin Drohmann, 2012-10-19) prettify the output of postprocessed source for source code browsing in doxygen. Now, we recommend the usage of the FILTER_SOURCE_FILES doxygen switch.

Change in 1.4 (Martin Drohmann, 2012-10-17) implemented varargin handling by Matlab inputParser, as suggested here: http://www.mathworks.de/de/help/matlab/ref/inputparser.-parse.html

Change in 1.4 (Martin Drohmann, 2012-02-24) ignore comments in front of classdef (fixes Grrr message from Jesse Hopkins)

Change in 1.3 (Martin Drohmann, 2012-02-17) added events section
Change in 1.3 (Martin Drohmann, 2012-02-15) improved documentation for dependent flags

Change in 1.3 (Martin Drohmann, 2012-02-15) remove =0 for purely virtual class methods

Change in 1.3 (Martin Drohmann, 2012-02-15) bugfix: no Grrr! messages for complex property declarations.

Change in 1.3 (Martin Drohmann, 2012-02-15) We totally ignore functions which are locally defined inside another function.

Change in 1.3 (Martin Drohmann, 2012-02-15) Added config GENERATE_SUBFUNTION_DOCUMENTATION and format for output of subfunctions.

Change in 1.3 (Martin Drohmann, 2012-02-03) Bugfix: Default values were printed twice if documented in the documentation block and given as property default values. Now, the documentated default value is preferred.

Change in 1.3 (Martin Drohmann, 2012-02-03) Improved the automatic documentation text for MATLAB specific attributes of properties and methods, add a link to the online MATLAB documentation.

New in 1.3 (Martin Drohmann, 2012-02-03) Print a warning message to stderr when optional parameter in methods of functions are not documented with default values.

New in 1.3 (Martin Drohmann, 2012-01-10) "Bugfix": Allowing the use of the AbortSet tag in property declarations, however, to extra action (e.g. inserting a note in documentation) is taken so far.

Change in 1.3 (Martin Drohmann, 2012-01-10) Some minor modifications for the postprocessor regarding dots `'` and `'::`

New in 1.3 (Martin Drohmann, 2011-12-16) Allowing multiple lines for default values in property comments & code and added a test case.

Change in 1.3 (Martin Drohmann, 2011-12-16) Bugfix: On Windows platforms the wrong getcwd command was issued and is now fixed.

Change in 1.3 (Martin Drohmann, 2011-12-13) Bugfix: Now handling the Abstract property correctly (was previously added for SetObservable declarations due to copy&paste)

Change in 1.3 (Martin Drohmann, 2012-01-13) Added a test case for default properties containing semicolons

Change in 1.3 (Martin Drohmann, 2012-01-13) Changed format for documentation of default properties and parameters

Change in 1.3 (Martin Drohmann, 2012-01-13) Default arguments for properties are added to the properties documentation block

Change in 1.3 (Martin Drohmann, 2012-01-13) Bugfix: observable properties have been documented as abstract ones.

Change in 1.3 (Martin Drohmann, 2011-12-13) Adding a bold "Default:" line in property documentation blocks if a default value/default tag is set in either code or property comment.

Change in 1.3 (Martin Drohmann, 2011-12-04) Bugfix reported by Evgeny Pr on mathworks: allow property definitions not ended by semicolons.
Change in 1.2 (Martin Drohmann, 2011-11-28) Allow long (including line breaks) default values for properties

Change in 1.2 (Daniel Wirtz, 2011-11-27) Included a new option -verbose for short mtoc++ version output.

Change in 1.2 (Martin Drohmann, 2011-11-17) Fixed a bug that messed up the documentation if a new line was started after a @type tag and added a test case to classA.m

Change in 1.2 (Martin Drohmann, 2011-11-17) Non-standard access modifier strings are now separated by a comma

Change in 1.2 (Martin Drohmann, 2011-11-17) Fixed a parse error occurring with the new -notation in newer MatLab versions. Calls like foo = bar(par1, ~, par3) now work.

Change in 1.2 (Martin Drohmann, 2011-11-17) The order of @default and @type tags in parameters (if occurring) is no longer fixed.

New in 1.2 (Martin Drohmann, 2011-11-17) New config flag COPY_TYPIFIED_FIELD_DOCU which allows to toggle the automatic insertion of required fields for method parameters. This flag sets whether the documentation of fields in 'Required fields of param', 'Optional fields of param' or 'Generated fields of retval' shall be copied in case the Parameter 'param' or 'retval' have a type.

Definition at line 431 of file mfilescanner.h.

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

• src/mfilescanner.h

15.14 ordered_map< ST >  Class Template Reference

Inherits vector< std::pair< std::string, ST > >.

Public Types

• typedef std::pair< std::string, ST > item
• typedef std::vector< item > base_type
• typedef base_type::iterator iterator
• typedef base_type::const_iterator const_iterator

Public Member Functions

• ST & operator[] (const std::string &key)
• iterator find (const std::string &key)

15.14.1 Detailed Description

template<class ST>class ordered_map< ST >

Definition at line 277 of file mfilescanner.h.

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

• src/mfilescanner.h
15.15 PropExtraInformation Struct Reference

Public Attributes

- bool dependent
- bool setter
- bool getter

15.15.1 Detailed Description

Definition at line 187 of file mfilescanner.h.

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

- src/mfilescanner.h

15.16 PropParams Struct Reference

Public Member Functions

- std::string ccprefix ()
- std::string print_list ()

Public Attributes

- bool constant
- bool transient
- bool dependent
- bool hidden
- bool setObservable
- bool abstr
- bool abortSet
- bool event

Friends

- std::ostream & operator<< (std::ostream &os, PropParams &pp)

15.16.1 Detailed Description

Definition at line 123 of file mfilescanner.h.

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

- src/mfilescanner.h

15.17 rowvec Class Reference

A matlab row vector.
15.17.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations.

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

- tools/config/class_substitutes.c

15.18 RunMode Struct Reference

Public Types

- enum Mode { Normal = 0, ParseParams, ParseMethodParams }

Public Attributes

- Mode mode
- std::string methodname
- bool latex_output
- bool print_fields
- bool auto_add_fields
- bool auto_add_params
- bool auto_add_class_properties
- bool auto_add_class
- bool copy_typified_field_docu
- bool remove_first_arg_in_abstract_methods
- bool parse_of_type
- bool void_type_in_return_values
- int print_return_value_name
- bool generate_subfunction_documentation

15.18.1 Detailed Description

Definition at line 30 of file mfilescanner.h.

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

- src/mfilescanner.h

15.19 sparsematrix Class Reference

A matlab sparse matrix.

Inherits matrix.

15.19.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations.

Definition at line 81 of file class_substitutes.c.

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

- tools/config/class_substitutes.c
15.20 struct Class Reference

A MatLab struct.

15.20.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations. 
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.21 varargin Class Reference

A variable number of input arguments.

15.21.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations. 
For more information about the varargin argument see the MatLab documentation on varargin. 
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c

15.22 varargout Class Reference

A variable number of output arguments.

15.22.1 Detailed Description

This class is an artificially created class in doxygen to allow more precise type declarations. 
For more information about the varargout argument see the MatLab documentation on varargout. 
The documentation for this class was generated from the following file:

• tools/config/class_substitutes.c
Index
AccessStruct, 21
addlistener
    handle, 25

cell, 22
char, 22
colvec, 22
ConfFileScanner, 23
double, 23
eq
    handle, 25
examples, 21

function_handle, 23
handle, 24
    addlistener, 25
eq, 25
    sort, 25
integer, 25

logical, 25

MFileScanner, 26
matrix, 26
MethodParams, 26

ordered_map< ST >, 29

PropExtraInformation, 30
PropParams, 30

rowvec, 30
RunMode, 31

sort
    handle, 25
sparsematrix, 31
struct, 32

varargin, 32
varargout, 32