Pygments Markdown Lexer Documentation

Release 0.1.0

Jürgen Hermann

July 06, 2015

Contents

A Markdown lexer for Pygments to highlight *Markdown* code snippets.

Here's a short example:

````
Enables _Pygments_ to handle
[Markdown](https://daringfireball.net/projects/markdown/syntax)
in *Sphinx* **code blocks**.

```
Preformatted, GitHub style!
```

```sh
echo "GitHub style with lexer"
```
````

See Markdown Syntax Examples for the full range of Markdown syntax elements.

CHAPTER 1

Documentation Contents

1.1 Markdown Syntax Examples

The following are some pygmentized examples from the Markdown syntax reference.

```
<h2 id="overview">Overview</h2>

<h3 id="philosophy">Philosophy</h3>

<ul id="ProjectSubmenu">
    <li><a href="/projects/markdown/" title="Markdown Project Page">Main</a></li>
</ul>

HTML <!-- comment one-liner -->
HTML <!-- comment
          on 2 lines -->
This --> not a comment
```

```
*   [Overview](#overview)
    *   [Philosophy](#philosophy)
    *   [Inline HTML](#html)
    *   [Automatic Escaping for Special Characters](#autoescape)

**Note:** This document is itself written using Markdown; you
can [see the source for it by adding '.text' to the URL][src].

  [src]: /projects/markdown/syntax.text
```

```
* * *
```

```
... including [Setext] [1], [atx] [2], ...

  [1]: http://docutils.sourceforge.net/mirror/setext.html
  [2]: http://www.aaronsw.com/2002/atx/
```

```
... *asterisks* around a word actually look like \*emphasis\*.
```

```
This is a regular paragraph.

<table>
    <tr>
        <td>Foo</td>
    </tr>
```

```
</table>

This is another regular paragraph.
```

```
Span-level HTML tags -- e.g. `<span>`, `<cite>`, or `<del>` -- can be
used anywhere in a Markdown paragraph, list item, or header.
```

```
Copyright symbol &copy;, but AT&T vs. AT&amp;T and 4 < 5.
```

```

============

    This is a H1
    ============


------------

Normal text.
```

```
#

    # This is a H1

##

######

Normal text.
```

```
#              #

    # This is a H1 #

##              ##

###              ######

Normal text.
```

```
    > This is a blockquote with two paragraphs. ...
    >
    > Donec sit amet nisl. Aliquam semper ipsum sit amet velit. Suspendisse
    > id sem consectetuer libero luctus adipiscing.

    > This is a blockquote with only a leading indicator.

    > Donec sit amet nisl. Aliquam semper ipsum sit amet velit. Suspendisse
    id sem consectetuer libero luctus adipiscing.

Blockquotes can be nested.

    > This is the first level of quoting.
    >
    > > This is nested blockquote.
    >
    > Back to the first level.
```

```
Blockquotes can contain other Markdown elements, including headers, lists,
and code blocks:

    > ## This is a header.
    >
    > 1.   This is the first list item.
    > 2.   This is the second list item.
    >
    > Here's some example code:
    >
    >     return shell_exec("echo $input | $markdown_script");
```

```
*   Red
*   Green
*   Blue

+   Red
+   Green
+   Blue

-   Red
-   Green
-   Blue

1.  Bird
2.  McHale
3.  Parish

*   A list item with a blockquote:

    > This is a blockquote
    > inside a list item.

*   A list item with a code block:

        <code goes here>

1986\. What a great season.
```

```
This is a normal paragraph:

    _This_ is a *code block*.

    * still code
    > also code

    tell application "Foo"
        beep
    end tell

Regular Markdown syntax is not processed within code blocks.
```

```
You can produce a horizontal rule tag (`<hr />`) by placing three or
more hyphens, asterisks, or underscores on a line by themselves. If you
wish, you may use spaces between the hyphens or asterisks. Each of the
following lines will produce a horizontal rule:

* * *
```

```
***

*****

- - -

---------------------------------------
```

```
<h2 id="span">Span Elements</h2>

<h3 id="link">Links</h3>

Markdown supports two style of links: *inline* and *reference*.

In both styles, the link text is delimited by [square brackets].

To create an inline link, use a set of regular parentheses immediately
after the link text's closing square bracket. Inside the parentheses,
put the URL where you want the link to point, along with an *optional*
title for the link, surrounded in quotes. For example:

    This is [an example](http://example.com/ "Title") inline link.

    [This link](http://example.net/) has no title attribute.

Will produce:

    <p>This is <a href="http://example.com/" title="Title">
    an example</a> inline link.</p>

    <p><a href="http://example.net/">This link</a> has no
    title attribute.</p>

If you're referring to a local resource on the same server, you can
use relative paths:

    See my [About](/about/) page for details.

Reference-style links use a second set of square brackets, inside
which you place a label of your choosing to identify the link:

    This is [an example][id] reference-style link.

You can optionally use a space to separate the sets of brackets:

    This is [an example] [id] reference-style link.

Then, anywhere in the document, you define your link label like this,
on a line by itself:

    [id]: http://example.com/  "Optional Title Here"

That is:

*   Square brackets containing the link identifier (optionally
    indented from the left margin using up to three spaces);
*   followed by a colon;
*   followed by one or more spaces (or tabs);
```

```
*   followed by the URL for the link;
*   optionally followed by a title attribute for the link, enclosed
    in double or single quotes, or enclosed in parentheses.

The following three link definitions are equivalent:

    [foo]: http://example.com/  "Optional Title Here"
    [foo]: http://example.com/  'Optional Title Here'
    [foo]: http://example.com/  (Optional Title Here)

**Note:** There is a known bug in Markdown.pl 1.0.1 which prevents
single quotes from being used to delimit link titles.

The link URL may, optionally, be surrounded by angle brackets:

    [id]: <http://example.com/>  "Optional Title Here"

You can put the title attribute on the next line and use extra spaces
or tabs for padding, which tends to look better with longer URLs:

    [id]: http://example.com/longish/path/to/resource/here
        "Optional Title Here"

Link definitions are only used for creating links during Markdown
processing, and are stripped from your document in the HTML output.

Link definition names may consist of letters, numbers, spaces, and
punctuation -- but they are *not* case sensitive. E.g. these two
links:

    [link text][a]
    [link text][A]

are equivalent.

The *implicit link name* shortcut allows you to omit the name of the
link, in which case the link text itself is used as the name.
Just use an empty set of square brackets -- e.g., to link the word
"Google" to the google.com web site, you could simply write:

    [Google][]

And then define the link:

    [Google]: http://google.com/

Because link names may contain spaces, this shortcut even works for
multiple words in the link text:

    Visit [Daring Fireball][] for more information.

And then define the link:

    [Daring Fireball]: http://daringfireball.net/

Link definitions can be placed anywhere in your Markdown document. I
tend to put them immediately after each paragraph in which they're
used, but if you want, you can put them all at the end of your
```

```
document, sort of like footnotes.

Here's an example of reference links in action:

    I get 10 times more traffic from [Google] [1] than from
    [Yahoo] [2] or [MSN] [3].

      [1]: http://google.com/        "Google"
      [2]: http://search.yahoo.com/  "Yahoo Search"
      [3]: http://search.msn.com/    "MSN Search"

Using the implicit link name shortcut, you could instead write:

    I get 10 times more traffic from [Google][] than from
    [Yahoo][] or [MSN][].

      [google]: http://google.com/        "Google"
      [yahoo]:  http://search.yahoo.com/  "Yahoo Search"
      [msn]:    http://search.msn.com/    "MSN Search"

Both of the above examples will produce the following HTML output:

    <p>I get 10 times more traffic from <a href="http://google.com/"
    title="Google">Google</a> than from
    <a href="http://search.yahoo.com/" title="Yahoo Search">Yahoo</a>
    or <a href="http://search.msn.com/" title="MSN Search">MSN</a>.</p>

For comparison, here is the same paragraph written using
Markdown's inline link style:

    I get 10 times more traffic from [Google](http://google.com/ "Google")
    than from [Yahoo](http://search.yahoo.com/ "Yahoo Search") or
    [MSN](http://search.msn.com/ "MSN Search").

The point of reference-style links is not that they're easier to
write. The point is that with reference-style links, your document
source is vastly more readable. Compare the above examples: using
reference-style links, the paragraph itself is only 81 characters
long; with inline-style links, it's 176 characters; and as raw HTML,
it's 234 characters. In the raw HTML, there's more markup than there
is text.

With Markdown's reference-style links, a source document much more
closely resembles the final output, as rendered in a browser. By
allowing you to move the markup-related metadata out of the paragraph,
you can add links without interrupting the narrative flow of your
prose.


<h3 id="em">Emphasis</h3>

Markdown treats asterisks (`*`) and underscores (`_`) as indicators of
emphasis. Text wrapped with one `*` or `_` will be wrapped with an
HTML `<em>` tag; double `*`'s or `_`'s will be wrapped with an HTML
`<strong>` tag. E.g., this input:

    *single asterisks*
```

```
    _single underscores_

    **double asterisks**

    __double underscores__

will produce:

    <em>single asterisks</em>

    <em>single underscores</em>

    <strong>double asterisks</strong>

    <strong>double underscores</strong>

You can use whichever style you prefer; the lone restriction is that
the same character must be used to open and close an emphasis span.

Emphasis can be used in the middle of a word:

    un*frigging*believable

But if you surround an `*` or `_` with spaces, it'll be treated as a
literal asterisk or underscore.

To produce a literal asterisk or underscore at a position where it
would otherwise be used as an emphasis delimiter, you can backslash
escape it:

    \*this text is surrounded by literal asterisks\*



<h3 id="code">Code</h3>

To indicate a span of code, wrap it with backtick quotes (`` ` ``).
Unlike a pre-formatted code block, a code span indicates code within a
normal paragraph. For example:

    Use the `printf()` function.

will produce:

    <p>Use the <code>printf()</code> function.</p>

To include a literal backtick character within a code span, you can use
multiple backticks as the opening and closing delimiters:

    ``There is a literal backtick (`) here.``

which will produce this:

    <p><code>There is a literal backtick (`) here.</code></p>

The backtick delimiters surrounding a code span may include spaces --
one after the opening, one before the closing. This allows you to place
literal backtick characters at the beginning or end of a code span:
```

```
    A single backtick in a code span: `` ` ``

    A backtick-delimited string in a code span: `` `foo` ``

will produce:

    <p>A single backtick in a code span: <code>`</code></p>

    <p>A backtick-delimited string in a code span: <code>`foo`</code></p>

With a code span, ampersands and angle brackets are encoded as HTML
entities automatically, which makes it easy to include example HTML
tags. Markdown will turn this:

    Please don't use any `<blink>` tags.

into:

    <p>Please don't use any <code>&lt;blink&gt;</code> tags.</p>

You can write this:

    `&#8212;` is the decimal-encoded equivalent of `&mdash;`.

to produce:

    <p><code>&amp;#8212;</code> is the decimal-encoded
    equivalent of <code>&amp;mdash;</code>.</p>



<h3 id="img">Images</h3>

Admittedly, it's fairly difficult to devise a "natural" syntax for
placing images into a plain text document format.

Markdown uses an image syntax that is intended to resemble the syntax
for links, allowing for two styles: *inline* and *reference*.

Inline image syntax looks like this:

    ![Alt text](/path/to/img.jpg)

    ![Alt text](/path/to/img.jpg "Optional title")

That is:

*   An exclamation mark: `!`;
*   followed by a set of square brackets, containing the `alt`
    attribute text for the image;
*   followed by a set of parentheses, containing the URL or path to
    the image, and an optional `title` attribute enclosed in double
    or single quotes.

Reference-style image syntax looks like this:

    ![Alt text][id]
```

```
Where "id" is the name of a defined image reference. Image references
are defined using syntax identical to link references:

    [id]: url/to/image  "Optional title attribute"

As of this writing, Markdown has no syntax for specifying the
dimensions of an image; if this is important to you, you can simply
use regular HTML `<img>` tags.


* * *


<h2 id="misc">Miscellaneous</h2>

<h3 id="autolink">Automatic Links</h3>

Markdown supports a shortcut style for creating "automatic" links for URLs and email addresses: simpl

    <http://example.com/>

Markdown will turn this into:

    <a href="http://example.com/">http://example.com/</a>

Automatic links for email addresses work similarly, except that
Markdown will also perform a bit of randomized decimal and hex
entity-encoding to help obscure your address from address-harvesting
spambots. For example, Markdown will turn this:

    <address@example.com>

into something like this:

    <a href="&#x6D;&#x61;i&#x6C;&#x74;&#x6F;:&#x61;&#x64;&#x64;&#x72;&#x65;
    &#115;&#115;&#64;&#101;&#120;&#x61;&#109;&#x70;&#x6C;e&#x2E;&#99;&#111;
    &#109;">&#x61;&#x64;&#x64;&#x72;&#x65;&#115;&#115;&#64;&#101;&#120;&#x61;
    &#109;&#x70;&#x6C;e&#x2E;&#99;&#111;&#109;</a>

which will render in a browser as a clickable link to "address@example.com".

(This sort of entity-encoding trick will indeed fool many, if not
most, address-harvesting bots, but it definitely won't fool all of
them. It's better than nothing, but an address published in this way
will probably eventually start receiving spam.)



<h3 id="backslash">Backslash Escapes</h3>

Markdown allows you to use backslash escapes to generate literal
characters which would otherwise have special meaning in Markdown's
formatting syntax. For example, if you wanted to surround a word
with literal asterisks (instead of an HTML `<em>` tag), you can use
backslashes before the asterisks, like this:

    \*literal asterisks\*
```

```
Markdown provides backslash escapes for the following characters:

    \   backslash
    `   backtick
    *   asterisk
    _   underscore
    {}  curly braces
    []  square brackets
    ()  parentheses
    #   hash mark
    +    plus sign
    -    minus sign (hyphen)
    .   dot
    !   exclamation mark
```

1.2 Complete API Reference

The following is a complete API reference generated from source.

1.2.1 pygments_markdown_lexer package

Pygments Markdown Lexer – A Markdown lexer for Pygments to highlight Markdown code snippets.

class pygments_markdown_lexer.**MarkdownLexer**(***options*)

Bases: pygments.lexer.RegexLexer

A Markdown lexer for Pygments.

Some rules adapted from code in pygments.lexers.markup (BSD-licensed).

aliases = [u’md’, u’markdown’]

closers = u’\’”)]}>\u2019\u201d\xbb!?’

end_string_suffix = u’((?=$)|(?=[-/:.,; \\n\\x00\\u2010\\u2011\\u2012\\u2013\\u2014\\xa0\\’\\”\\)\\]\\}\\>\\u2019\\u

filenames = [u’*.md’, u’*.mkd’, u’*.markdown’]

flags = 8

mimetypes = [u’text/x-markdown’]

name = u’Markdown’

tokens = {‘inline’: [(u’\\\\.’, Token.Literal.String.Escape), (u’&[-a-z0-9]+;’, Token.Literal.String.Symbol), (u’&#[0-9]{1,9

unicode_delimiters = u’\u2010\u2011\u2012\u2013\u2014\xa0’

Submodules

pygments_markdown_lexer.lexer module

Markdown lexer for Pygments.

See Write your own lexer and Builtin Tokens.

class pygments_markdown_lexer.lexer.**Markdown**
Bases: object

Symbolic names for Markdown tokens.

CodeBlock = Token.Comment.Preproc

Heading = Token.Generic.Heading

HtmlBlock = Token.Comment.Preproc

HtmlComment = Token.Comment.MultiLine

HtmlEntity = Token.Literal.String.Symbol

HtmlSingle = Token.Comment.Single

Markup = Token.Keyword

SubHeading = Token.Generic.Heading

class pygments_markdown_lexer.lexer.**MarkdownLexer**(***options*)
Bases: pygments.lexer.RegexLexer

A Markdown lexer for Pygments.

Some rules adapted from code in pygments.lexers.markup (BSD-licensed).

aliases = [u'md', u'markdown']

closers = u'\'")]}>\u2019\u201d\xbb!?'

end_string_suffix = u'((?=$)|(?=[-/:.,; \\n\\x00\\u2010\\u2011\\u2012\\u2013\\u2014\\xa0\\'\\"\\)\\]\\}\\>\\u2019\\u

filenames = [u'*.md', u'*.mkd', u'*.markdown']

flags = 8

mimetypes = [u'text/x-markdown']

name = u'Markdown'

tokens = {'inline': [(u'\\\\.', Token.Literal.String.Escape), (u'&[-a-z0-9]+;', Token.Literal.String.Symbol), (u'&#[0-9]{1,9

unicode_delimiters = u'\u2010\u2011\u2012\u2013\u2014\xa0'

1.3 Contribution Guidelines

1.3.1 Overview

Contributing to this project is easy, and reporting an issue or adding to the documentation also improves things for every user. You don't need to be a developer to contribute.

Reporting issues

Please use the *GitHub issue tracker*, and describe your problem so that it can be easily reproduced. Providing relevant version information on the project itself and your environment helps with that.

Improving documentation

The easiest way to provide examples or related documentation that helps other users is the *GitHub wiki*.

If you are comfortable with the Sphinx documentation tool, you can also prepare a pull request with changes to the core documentation. GitHub's built-in text editor makes this especially easy, when you choose the *"Create a new branch for this commit and start a pull request"* option on saving. Small fixes for typos and the like are a matter of minutes when using that tool.

Code contributions

Here's a quick guide to improve the code:

1. Fork the repo, and clone the fork to your machine.
2. Add your improvements, the technical details are further below.
3. Run the tests and make sure they're passing (`invoke test`).
4. Check for violations of code conventions (`invoke check`).
5. Make sure the documentation builds without errors (`invoke build --docs`).
6. Push to your fork and submit a pull request.

Please be patient while waiting for a review. Life & work tend to interfere.

1.3.2 Details on contributing code

This project is written in Python, and the documentation is generated using Sphinx. setuptools and Invoke are used to build and manage the project. Tests are written and executed using pytest and tox.

Set up a working development environment

To set up a working directory from your own fork, follow these steps, but replace the repository `https` URLs with SSH ones that point to your fork.

For that to work on Debian type systems, you need the `git`, `python`, and `python-virtualenv` packages installed. Other distributions are similar.

Add your changes to a feature branch

For any cohesive set of changes, create a *new* branch based on the current upstream `master`, with a name reflecting the essence of your improvement.

```
git branch "name-for-my-fixes" origin/master
git checkout "name-for-my-fixes"
... make changes...
invoke ci # check output for broken tests, or PEP8 violations and the like
... commit changes...
git push origin "name-for-my-fixes"
```

Please don't create large lumps of unrelated changes in a single pull request. Also take extra care to avoid spurious changes, like mass whitespace diffs. All Python sources use spaces to indent, not TABs.

Make sure your changes work

Some things that will increase the chance that your pull request is accepted:

- Follow style conventions you see used in the source already (and read PEP8).
- Include tests that fail *without* your code, and pass *with* it. Only minor refactoring and documentation changes require no new tests. If you are adding functionality or fixing a bug, please also add a test for it!
- Update any documentation or examples impacted by your change.
- Styling conventions and code quality are checked with `invoke check`, tests are run using `invoke test`, and the docs can be built locally using `invoke build --docs`.

Following these hints also expedites the whole procedure, since it avoids unnecessary feedback cycles.

1.4 Software License

1.4.1 Full License Text

```
   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
   excluding communication that is conspicuously marked or otherwise
   designated in writing by the copyright owner as "Not a Contribution."

   "Contributor" shall mean Licensor and any individual or Legal Entity
   on behalf of whom a Contribution has been received by Licensor and
   subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
```

```
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
       notices within Derivative Works that You distribute, alongside
       or as an addendum to the NOTICE text from the Work, provided
       that such additional attribution notices cannot be construed
       as modifying the License.

   You may add Your own copyright statement to Your modifications and
   may provide additional or different license terms and conditions
   for use, reproduction, or distribution of Your modifications, or
   for any such Derivative Works as a whole, provided Your use,
   reproduction, and distribution of the Work otherwise complies with
   the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
```

```
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "{}"
   replaced with your own identifying information. (Don't include
   the brackets!)  The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright {yyyy} {name of copyright owner}

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

CHAPTER 2

Indices & Tables

- genindex
- modindex
- search

p

A

C

E

F

H

M

N

P

S

T

U