MarkdownKit 1.0.3

Thank you so much for helping to support my small business by purchasing MarkdownKit. This document contains all you need to know about the module. If you have any issues with the module, please contact me either directly via email, through my website or the Xojo forums.

About MarkdownKit

MarkdownKit is a 100% CommonMark compliant Markdown parser for Xojo written in pure Xojo code. I needed a fast and robust parser that not only would reliably generate the correct output but would also run on iOS. After looking around I realised that there was no other solution available for Xojo and so I decided to write one myself. MarkdownKit is a labour love, taking 2 months of hard work, 730 GitHub commits and containing over 6000 lines of code.

MarkdownKit takes Markdown as input and generates an *abstract syntax tree* (AST). From the AST, it is then able to render the input as HTML.

Package Contents

Your download contains the following components:

1. This documentation
2. A demo application
3. Desktop test suites (both `String` and `Text` versions of the module)
4. iOS test suite

The Desktop Test Suites

There are two test suite projects (`src/Test Suite (String Version)` and `src/Test Suite (Text Version)`) These applications contains three windows:

one for running the HTML tests, one for the AST tests and a third as a simple editor. Clicking the "Run" button on the toolbar for test windows will run all 649 tests from the CommonMark 0.29 specification, proving the compliance of the parser. Feel free to click on an individual test to see the provided input, the expected output and the generated output.

The iOS Application Test Suite

This is a very basic iOS app which will run the 649 CommonMark HTML tests. It contains the `Text` based `MarkdownKit` module (for you to use in your own projects). It's designed to run on an iPad in landscape mode.

The Demo Application

The demo app is a fully functioning Markdown editor with a live preview. It has light and dark themes (user-selectable) and will even highlight syntax within code blocks in the provided Markdown input. I have deliberately kept it light on features as its purpose is really just to demonstrate what can be achieved with MarkdownKit.

Which MarkdownKit Module?

Your purchase actually includes two `MarkdownKit` modules. One uses the Xojo `String` datatype, the other uses `Text`. The module found in the test suite project in the `src/Test Suite (String Version)`) folder uses the `String` datatype and the module in the `src/Test Suite (Text Version)` uses the `Text` datatype. The iOS project also uses the `Text` datatype-based version of the module. Since Xojo's iOS framework does not support the `String` datatype I originally wrote `MarkdownKit` using `Text`. Whilst this worked fine on iOS and macOS, there are inherent issues with Xojo's implementation of the `Text` datatype on Windows and Linux. In short, the performance is poor. Therefore, I decided to port the module to use the `String` datatype.

You are free to use whichever module you like (they are both CommonMark compliant). I would recommend using the `String` based version whenever possible (i.e: in all non-iOS projects) as it is much faster. The `Text` datatype has already been deprecated by Xojo and I'm hopeful that they will add `String`

support to iOS soon.

Quick Start

1. For non-iOS projects, open the `MarkdownKit.xojo_project` file in the IDE (use either `src/Test Suite (String Version)` or `src/Test Suite (String Version)`). For iOS projects open the `iOS Demo.xojo_project` file in `src/iOS`. Copy the `MarkdownKit` module from the navigator and paste it into your own project.
2. Convert Markdown source to HTML with the `MarkdownKit.ToHTML()` method:

```
// Desktop, Console & Web projects should use String:
Dim html As String = MarkdownKit.ToHTML("Some **bold** text")

// iOS must use Text:
Dim html As Text = MarkdownKit.ToHTML("Some **bold** text")
```

Advanced Use

I imagine that most people will only ever need to use the simple `MarkdownKit.ToHTML()` method. However, if you want access to the abstract syntax tree created by `MarkdownKit` during parsing then you can, like so:

```
Dim ast As New MarkdownKit.Document("Some **bold** text")

// Parsing Markdown is done in two phases. First the block structur
e is
// determined and then inlines are parsed.
ast.ParseBlockStructure
ast.ParseInlines
```

Why might you want access to the AST? Well, maybe you want to do something as simple as render every soft linebreak in a document as a hard linebreak. Perhaps you want to output the Markdown source as something other than HTML.

`MarkdownKit` provides a class interface called `IRenderer` which must be implemented by any custom renderer you write. The built-in `MarkdownKit.HTMLRenderer` and `MarkdownKit.ASTRenderer` classes are examples of renderers which implement this interface. Take a look at their well-documented methods to learn how to write your own renderer.

14th July 2019