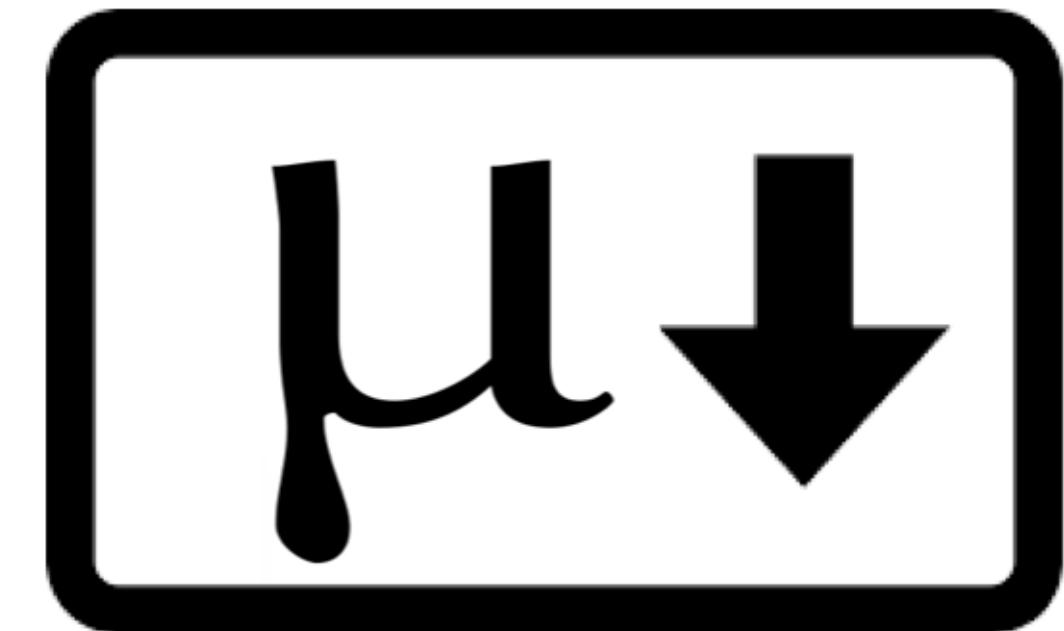




Microdown

clean and extensible markup language to support Pharo documentation

S. Ducasse & K. Osterbye



License and other admins

<https://github.com/pillar-markup/Microdown>

MIT

Copyright © 2019 - 2022 Stéphane Ducasse, Kasper Osterbye

with contributions of Guillermo Polito, Leo Frere, Gaylord Delporte, Laurine Dargaud, Lina Grangaud, Hernan Morales Durand, Esteban Lorenzano

Context

Context: The Pillar Text Edition Toolchain

<https://github.com/pillar-markup>

Goals and used for

- Book generation <http://books.pharo.org>
- Slides <http://mooc.pharo.org>
- Static Website generation (books)

Pharo: an immersive object-oriented system

Damien CASSOU, Stéphane DUCASSE and Luc FABRESSE
www.pharo.org
W1S02

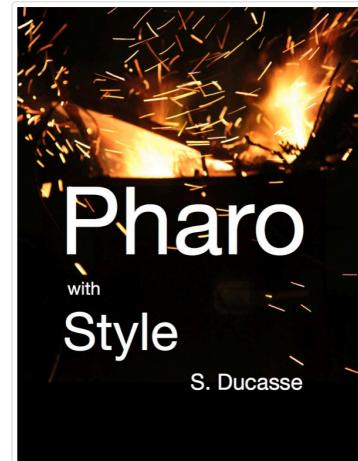


Pharo Books

Pharo is a clean, innovative, open-source, live-programming environment. Get immersed in a world of living objects!

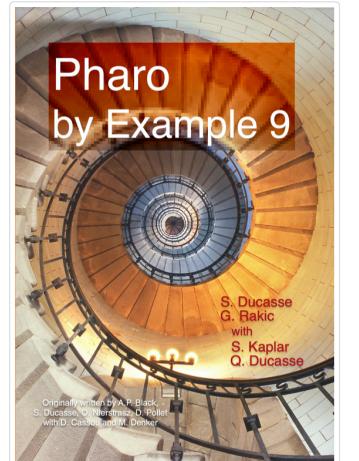
Contribute to the SquareBracketsAssociates community free books <https://github.com/SquareBracketAssociates/>

Recent books



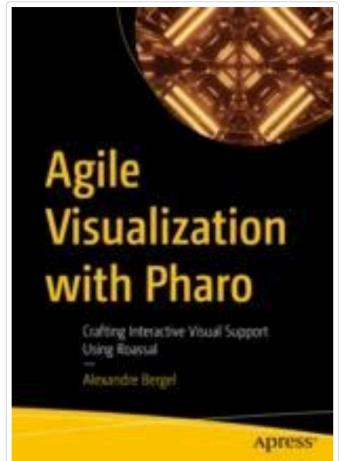
Pharo with Style
S. Ducasse

Pharo with Style (New 2022 Edition) explains the difference between code that runs and code that talks.



Pharo by Example 9
S. Ducasse, G. Bakic, with S. Kaplar, O. Ducassee

Pharo by Example 9 (New 2022 Edition) is a new version of Pharo by Example!. New material inside.



Agile Visualization with Pharo
Crafting Interactive Visual Support Using Pharo
Alessandro Bergel

Agile Visualization covers aspects that are relevant for practitioners, businesses, and academics to successfully

Pillar: One word Two concepts

- Syntax of the markdown

!! Header

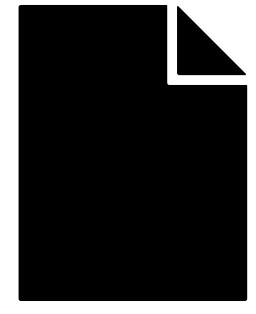
[[[

]]]

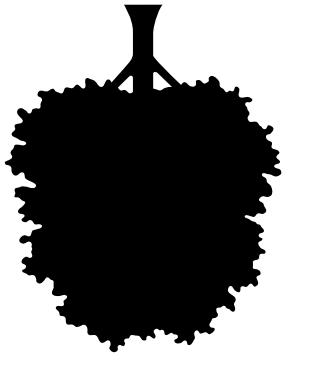
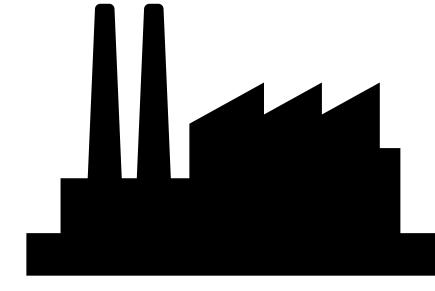
- Document compilation Chain

Context

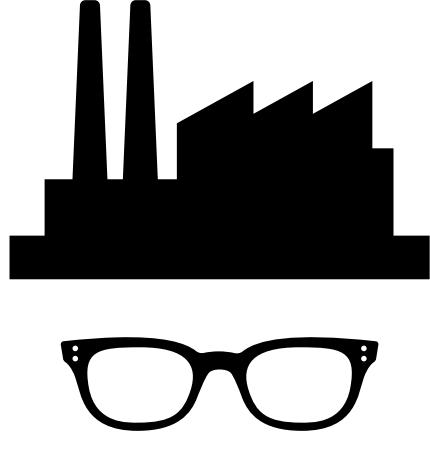
Pillar Compilation Chain



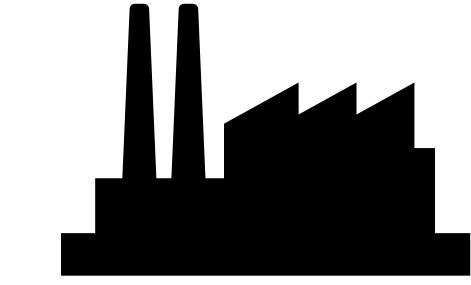
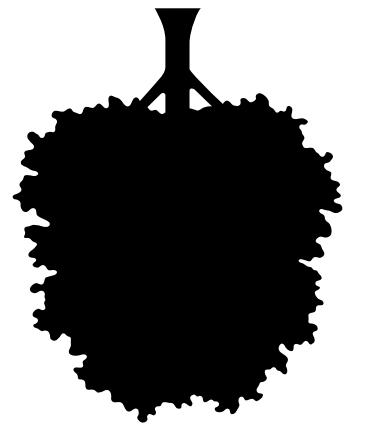
Pillar



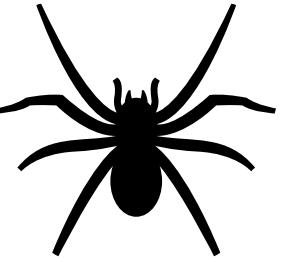
Pillar Trees



Pillar Visitors



PDF



Web



Slides

Problem

New goals

- Better integration with external tools: text editors, websites
- In-Image documentation

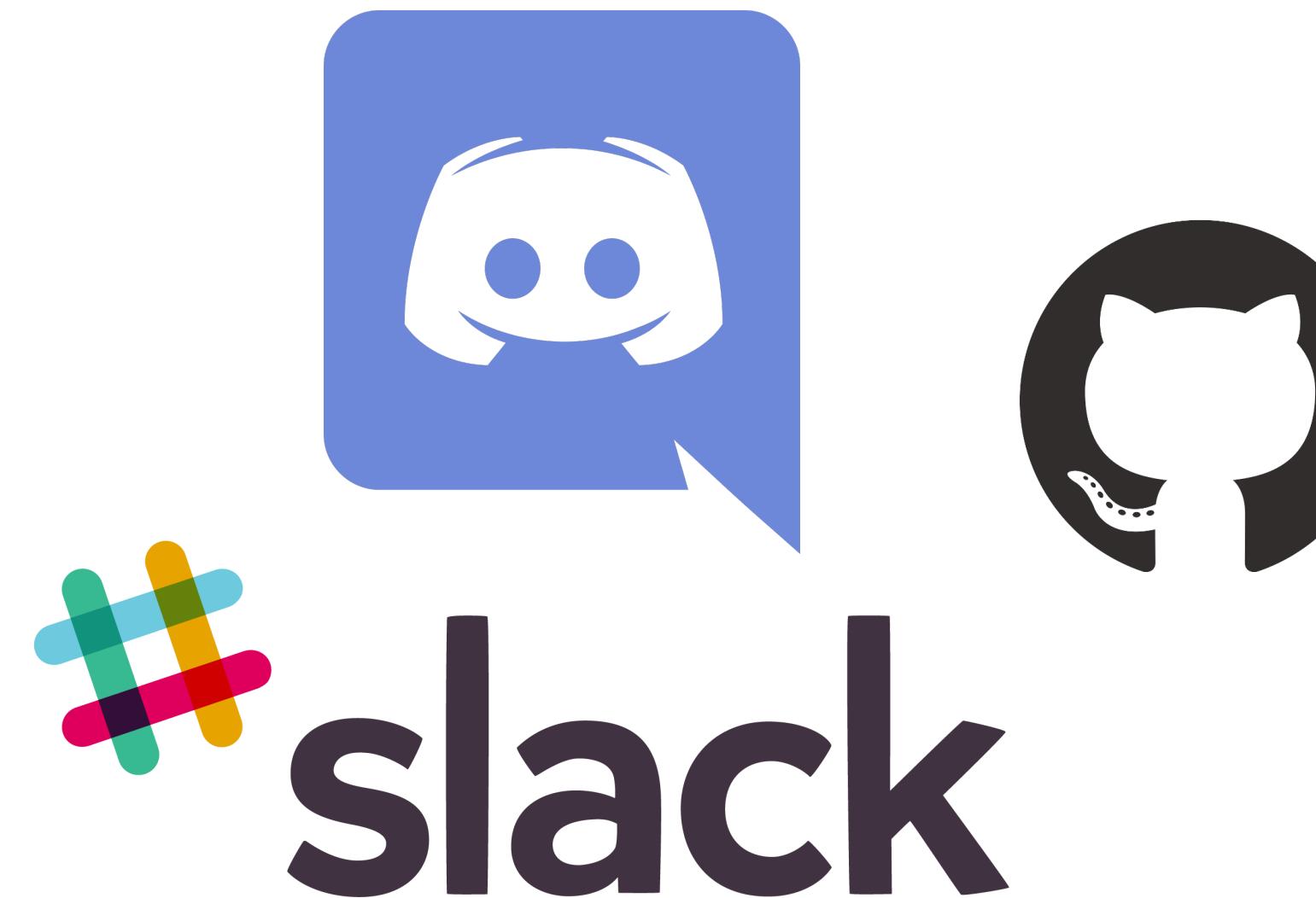


Problem

Markdown is a de-facto “standard”

Well-known, very used => low entry barrier

- Project documentation (e.g., Github readme files)
- User discussion (e.g., Slack, Discord, Stack overflow)
- Static site generation (e.g., Jekyll)



A screenshot of a GitHub repository page for 'pillar-markup/pillar'. The page shows the repository's README.md and appveyor.yml files. The README.md file contains a section titled 'Pillar (P8)' which describes the tool as a markup syntax and tool-suite for generating documentation, books, websites, and slides. It notes that Pillar was invented around 2000 for SmallWiki and was later extracted from Pier CMS. The GitHub interface includes standard navigation elements like back, forward, and search, along with a sidebar for other repos and user profile links.

pillar-markup/pillar: Markup syntax and tool-suite for generating documentation, books, websites and slides.

github.com/pillar-markup/pillar

Aplicaciones Tech CNRS Books Research Admin Pharo Japanese Sign in to GitHub ... GtkWidget

README.md update Readme to symlink pillar script 12 days ago

appveyor.yml Avoid creating github releases automagically 2 years ago

README.md

Pillar (P8)

Pillar is a markup syntax and tool-suite to generate documentation, books, websites and slides. Pillar is not new, it was invented around 2000 as a supporting language for [SmallWiki](#): one of the first wiki using OOP for real. Its ancestor was the markup for the Pier CMS and we extracted it from Pier to make it more applicable to different domains. The Pillar syntax is similar to markdown but its emphasis is on publishing and how it handles different types of links.

build passing Documentation download

Problem

But... Markdown is ****many**** standards

Each tool has a variant

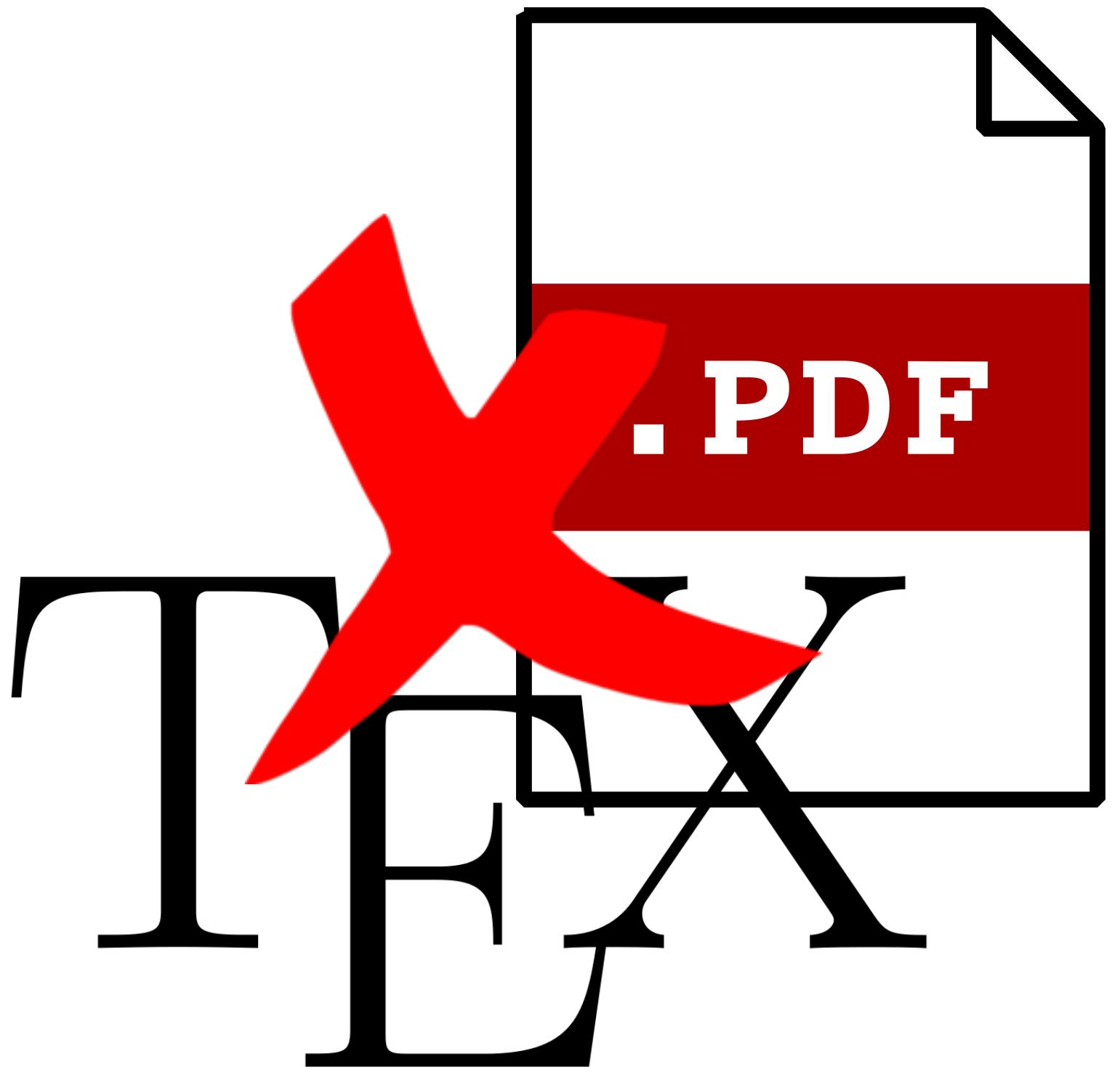
The screenshot shows a web browser window with three tabs open: 'markdown - Ecosia', 'Markdown - Wikipedia, la enciclopedia libre', and 'Tools | Markdown Guide'. The main content area displays the 'Tools' section of the [Markdown Guide](https://markdownguide.org/tools/) website. The page has a blue header with the title 'Tools' and a subtitle 'Applications and components that support Markdown.' Below this, there are four cards, each featuring a logo and a brief description of a Markdown tool:

- Howdy!** This is the start of a comprehensive Markdown tool directory. Compiling all this will take some time! [Learn how to contribute.](#)
- Bear** Bear is a Markdown notes application for macOS and iOS devices. [Learn more](#)
- Boostnote** Boostnote is a Markdown note taking application for developers. [Learn more](#)
- Byword** Byword is a capable Markdown editor available for macOS and iOS. [Learn more](#)

At the bottom of the page, there are partial views of other tools: CodiMD, Collected Notes, Dillinger, and DocUSAHC.

And no proper book support

- No explicit anchors
- No figures or code references
- No captions
- Not extensible
- Since June 22 Math support but
 - cannot refer to your equation :)

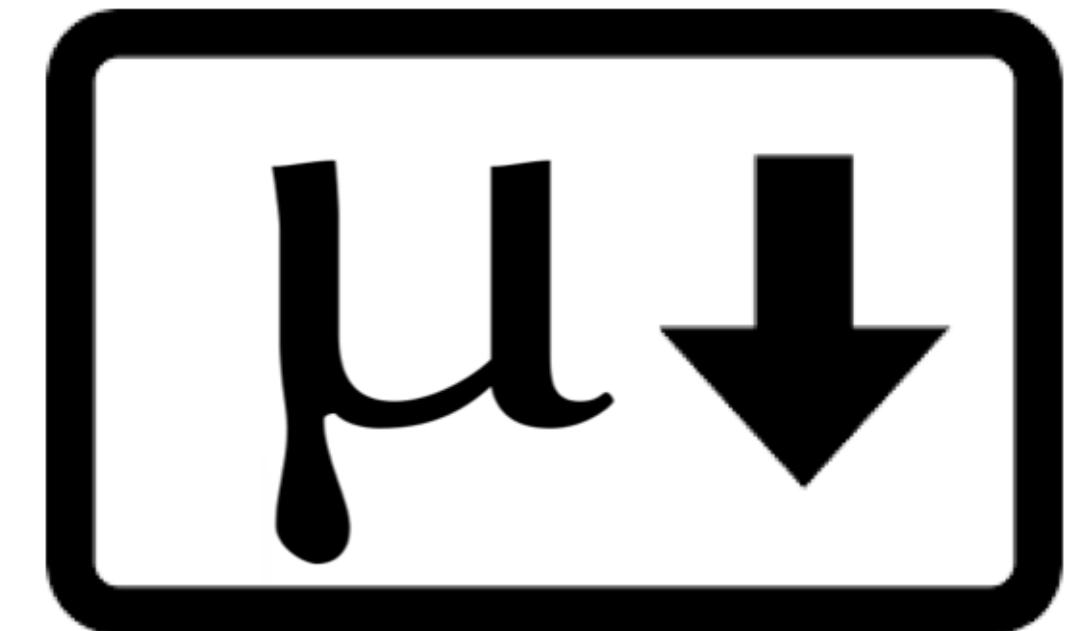


Microdown

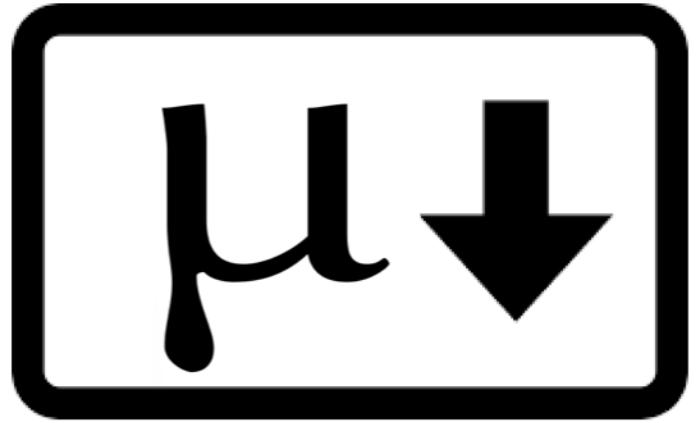
a clean and extensible markup language

Yes, yet another one!

- Markdown clean and non-ambiguous **subset**
=> compatibility Microdown → Markdown
- **Extensible**
=> support for books
=> slide, calendar,
- A **robust parser** tolerates non-supported syntax
=> compatibility Microdown ← Markdown



A Markdown Compatible Subset



*** horizontal lines

[link](<https://example.com>)

... ##### headers

![figure](image.jpg)

```

code blocks

```

1. ordered
2. lists

* unordered
* lists

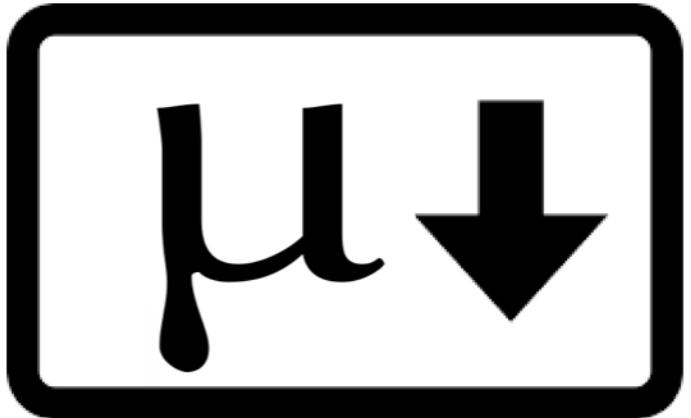
+ Blockquotes, tables,
bold, italics, columns

+ slides,...

Solution: Microdown

Microdown Extensions (1)

(ignored by Markdown)



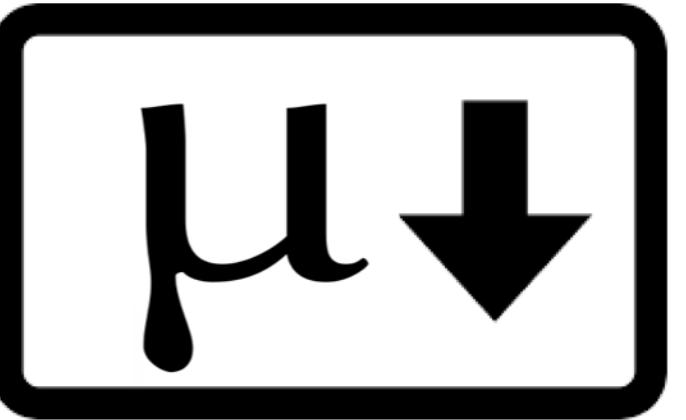
% Comments

Meta data

```
{  
  "author": "Tintin et Milou",  
  "title": "Tintin chez les picaros"  
}
```

Solution: Microdown

Anchors and References (ignored by Markdown)



This title has an anchor

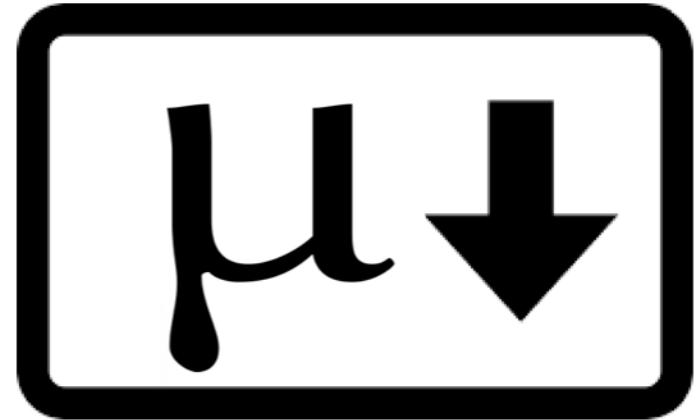
@anchor

![Our Logo](logo.png size=80&anchor=logo1)

References to anchors ***@anchor@*** and figures ***@logo1@***.

Solution: Microdown

Extensible Annotations (ignored by Markdown)



Extensible annotations (in text)

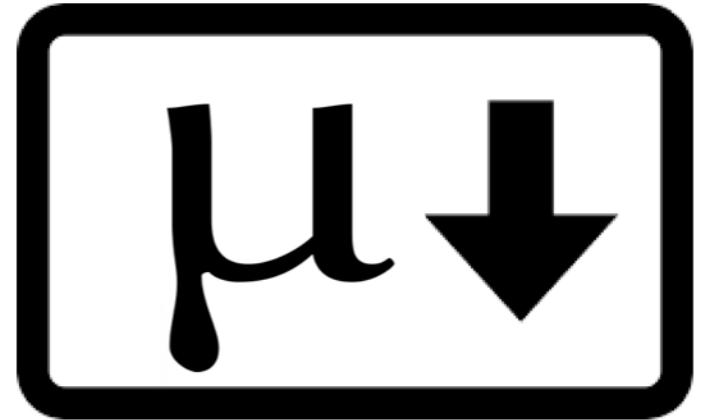
```
{!footnote | value=footnote is an annotation.!}
```

```
{!citation|ref=Duca99a!}
```

Solution: Microdown

Extensible environments

(ignored by Markdown)



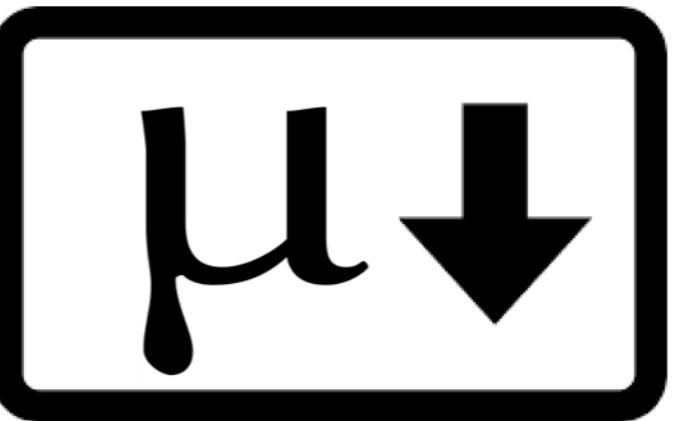
```
<!slide|title=This is a cool title&tag=nh5p
```

- a list of bullet
 - bullet 2
 - bullet 3
- !>

```
<!inputFile|path=Chapters/withStyle.md!>
```

Solution: Microdown

Extensible environments (ignored by Markdown)



```
<!columns
```

```
<!column|width=80
```

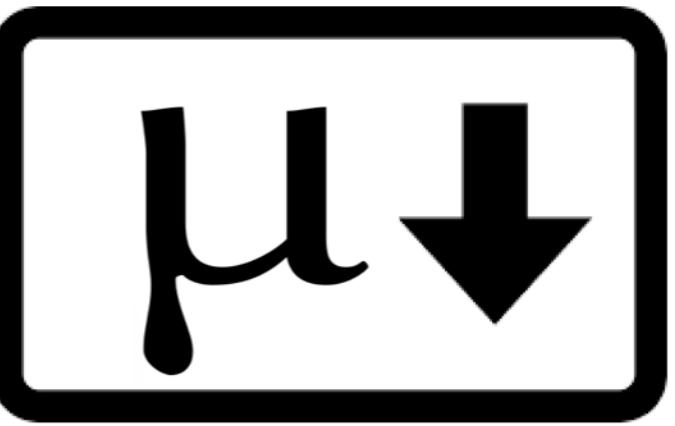
- col 1 item1 a first list
- col 1 item2 a first list

```
!>
```

```
!>
```

Solution: Microdown

Math



Math in paragraph

This is a math $\frac{1+3}{2+5}$

Math equations

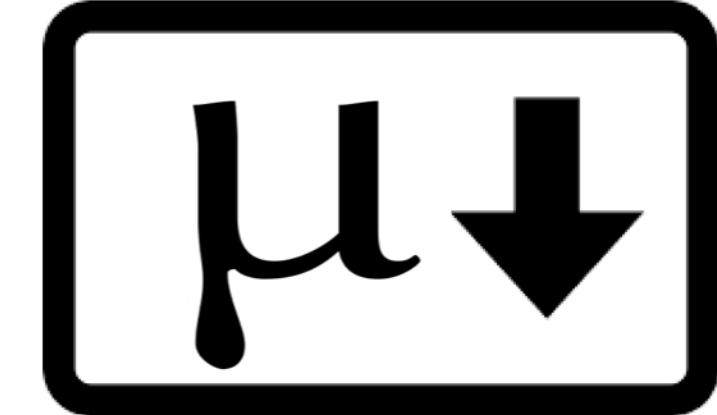
```
$$ %anchor=Eq1
\frac{1+3}{2+5}
$$
```

As you can see in Equation *@Eq1@*

Solution: Microdown

Microdown Robust Parser

Inspired by CommonMark's specification



- Elements are either block elements (paragraphs, blockquotes, lists...) or inline elements (bold, italics, links...)
- Blocks form a tree
- When a block opener is detected a new block is open in the tree
- A line is added to the current block if it accepts it. Otherwise the block is closed and it retries with the parent.

Invalid syntax is then added as verbatim text

Rendering of Class and Package Comments

The screenshot shows the Pharo IDE interface with the following details:

- Title Bar:** MicroDownParser
- Left Sidebar:** A tree view of packages:
 - Manifest
 - Model
 - ModelInline
 - Parser** (selected)
 - Extensions
- Right Sidebar:** A list of class-side comments for `MicroDownParser`.
 - instance side
 - accessing
 - initialization
 - markups
 - node creation
 - parsing
- Bottom Panel:** A text area displaying rendered comments:

```
Raw for your other code (inline) >>> {{ some code }}
```

```
Link >>> [link's name](url|key1=value1&key2=value2)
```

```
Figure >>> ! [figure's name](url|key1=value1&key2=value2)
```

```
! [Pharo logo](https://files.pharo.org/media/logo/logo.png)
```

produces


- Toolbar:** Includes buttons for Comment, MicroDownPars, and Inst. side methc.
- Status Bar:** Shows navigation icons (+, ←, →).

Applications

Comment Templates

MicSurfacicMicrodownToPillarTest

Description

This test case uses the microdownSnippetFactory and test that the conversion to Pillar is correct. This is why it is in this package.

Microdown text -> Microdown trees -> Pillar trees

The tests are just checking that object of the correct class is created. Future extensions should handle the details.

Tests

This test suite defines 56 test methods.

Locally defined tests are:

- MicSurfacicMicrodownToPillarTest>>#testSuperscriptFormatEmpty
- MicSurfacicMicrodownToPillarTest>>#testLineEnd
- MicSurfacicMicrodownToPillarTest>>#testScriptWithNewLine
- MicSurfacicMicrodownToPillarTest>>#testAnchorWithNewLine
- MicSurfacicMicrodownToPillarTest>>#testItalicFormatEmpty
- MicSurfacicMicrodownToPillarTest>>#testScriptParametersMultiple
- MicSurfacicMicrodownToPillarTest>>#testScriptParameterValue
- MicSurfacicMicrodownToPillarTest>>#testAnchorWithSpaceInside
- MicSurfacicMicrodownToPillarTest>>#testScriptParameter
- MicSurfacicMicrodownToPillarTest>>#testHeaderLevel3

BaselineOfMicrodown

A baseline is a kind of map to load project.

Description

Please comment package here

Dependencies

```
baseline: spec
<baseline>

spec for: #'common' do: [
  spec baseline: 'Pillar' with: [ spec
    loads: #('rich text exporter');
    repository: 'github://pillar-markup/pillar:dev-8/src' ].
```

Applications

MicrodownBuilder

- No you should not write microdown manually
- You can script the Microdown builder
-

Applications

Comment Templates

The screenshot shows a software interface for viewing class hierarchies and their associated comments. The main window displays a tree view of classes under 'Spec' and a detailed view of the 'SpMenuItemPresenter' class.

Class Hierarchy:

- Spec2-Adapters-Morphic
- Spec2-Adapters-Morphic-Tests
- Spec2-Adapters-Stub
- Spec2-Backend-Tests
- Spec2-Code
- Spec2-Code-Backend-Tests
- Spec2-Code-Commands
- Spec2-Code-Diff
- Spec2-Code-Diff-Morphic
- Spec2-Code-Diff-Tests
- ManifestSpec2Core
- SpAbstractAdapter
- SpAbstractPresenter
- SpPresenter
- SpAbstractWidgetPresenter
- SpAbstractButtonPresenter
- SpButtonPresenter
- SpMenuItemPresenter
- SpAbstractFormButtonPresenter
- SnCheckBoxPresenter

SpMenuItemPresenter Class View:

Comments:

- initialize
- menu
- api
- api - events
- initialization
- overrides
- whenMenuChangedDo:

Code Examples:

```
addItem: [ :item | item name: '3:', loremIpsumWords atRandom ] ;  
yourself ].
```

Factory method:
You can use `SpMenuItemPresenter` in your presenters by sending `SpPresenter>>#newMenuItem`.

Examples:

- `SpMenuItemPresenter class>>#example`

API Methods:

- `SpMenuItemPresenter>>#menu`
- `SpMenuItemPresenter>>#menu:`

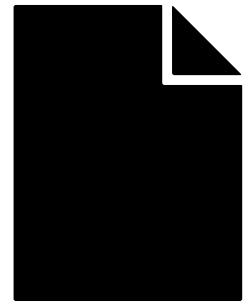
Events:

- `SpMenuItemPresenter>>#whenMenuChangedDo:`

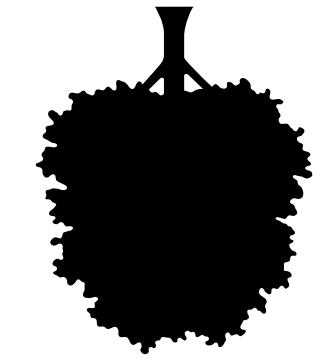
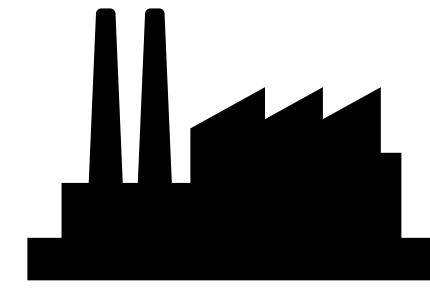
Hierarchy:

```
SpAbstractPresenter  
└─ SpPresenter  
└─ SpAbstractWidgetPresenter  
└─ SpAbstractButtonPresenter  
└─ SpMenuItemPresenter (this is me)
```

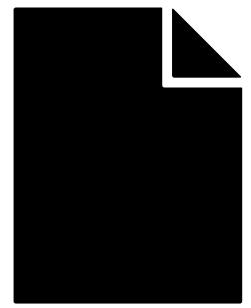
Pillar can eat uDown Documents



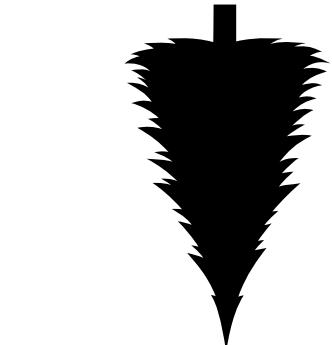
Pillar



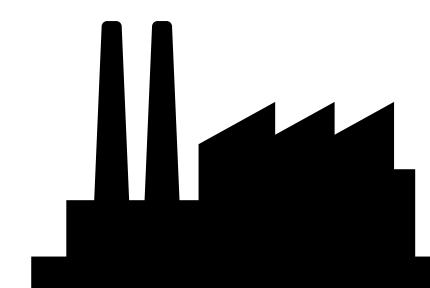
Pillar Trees



uDown



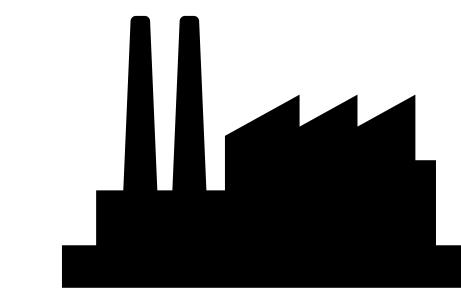
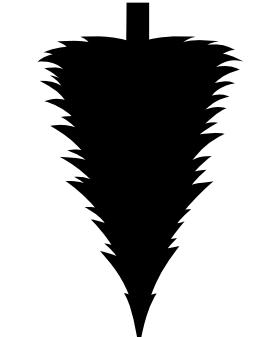
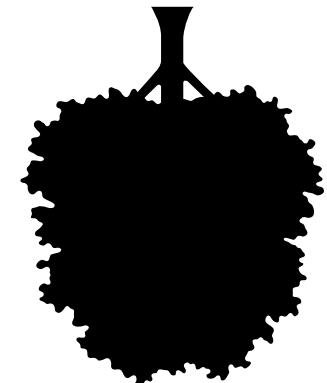
uDown Trees



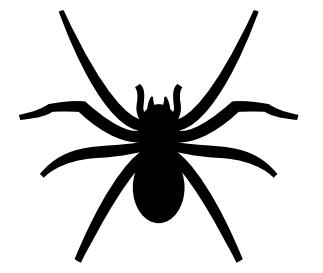
Pillar Visitors



uDown Visitors



PDF



Web



Slides

Books (state of union)

- convert a book into Microdown
 - pillar convertBook index.pillar
- compile a md book
 - pillar build pdf index.md

Microdown HTML Styler

- Let you pick and configure a HTML export style
- Choose an HTML document properties
- Shipped with (> 20) class-less CSS frameworks.
- Don't need to hand-craft your tags to add classes.
- Live source preview so you can see what's being exported.
- Some CSS Frameworks include themes, so it's also supported.

Microdown HTML Styler

Getting started

In this little book we will show that Pharo is easy to learn for a Pythonist.

- Dynamically typed
- REPL everywhere
- Objects everywhere

Pharo dynamic nature is close to the one of Python.

Now Pharo is just uniform: the computation is expressed using only objects answering messages and lambdas (lexical closure).

Arithmetic

Let's start with basic messages for arithmetic operations.

Basic operators

```
1 + 2  
>>> 3  
  
1.1 * 2.3  
>>> 2.53
```

Tips: in Pharo all the mathematic operators (+, '-', ...) can be defined on any class.

Power

```
4 ** 2
```

Getting started

In this little book we will show that Pharo is easy to learn for a Pythonist.

- Dynamically typed
- REPL everywhere

GETTING STARTED

In this little book we will show that Pharo is easy to learn for a Pythonist.

- Dynamically typed
- REPL everywhere
- Objects everywhere

Pharo dynamic nature is close to the one of Python.

Now Pharo is just uniform: the computation is expressed using only objects answering messages (lexical closure).

ARITHMETIC

Let's start with basic messages for arithmetic operations.

Microdown HTML Styler @ Work

x - □ Microdown HTML Styler on: PharoForThePythonists.mic

Open Preview Export Help Update Quit

Styles

AttricSS
AwmCSS
Axist
Chota
ClasslessCSS
ConcreteCSS
LaTeX
MVP
MercuryCSS
NewCSS
PicnicCSS
Sakura
SimpleCSS
SpCSS
Splendor
StylizeCSS
Tufte
W3C
WaterCSS
Wing
Yorha

HTML Options CSS Details Export Options

Document Type: HTML 5
Encoding: UTF-8
Language: en - English
Embed CSS:
Links open a new page:

CSS

```
/*!
 * LaTeX.css (https://latex.now.sh/)
 *
 * Source:
 https://github.com/vincentdoerig/latex-
 css
 * Licensed under MIT
 (https://github.com/vincentdoerig/latex-
 -css/blob/master/LICENSE)
 */@font-face{font-family:'Latin
 Modern';font-style:normal;font-weight:n
 ormal;font-display:swap;src:url('./font
 s/LM-regular.woff2')
 format('woff2'),url('./fonts/LM-regular
 .woff')
 format('woff'),url('./fonts/LM-regular.
 File size: 6 KB
```

HTML

```
<!DOCTYPE html><html lang="en"><head><meta http-equiv="Content-Type" content="text/html; charset=utf-8"><meta name="generator" content="Microdown"><meta name="viewport" content="width=device-width,initial-scale=1.0,user-scalable=yes"><title>Untitled document</title><link rel="stylesheet" href="css/style.css"></head><body><main role="main">
<h2>Getting started</h2>
<p>In this little book we will show that Pharo is easy to learn for a
```

File size: 13 KB

x - □ Microdown HTML Styler on: PharoForThePythonists.mic

Open Preview Export Help Update Quit

Styles

AttricSS
AwmCSS
Axist
Chota
ClasslessCSS
ConcreteCSS
LaTeX
MVP
MercuryCSS
NewCSS
PicnicCSS
Sakura
SimpleCSS
SpCSS
Splendor
StylizeCSS
Tufte
W3C
WaterCSS
Wing
Yorha

HTML Options CSS Details Export Options

Themes

Big Stone
Black
Gondola
Mischnka
Pastel Pink
Pearl Lusta
Tasman
White

Repository: Visit project web
CSS: <https://igoradamenko.github.io/awsm.css/css/awsm.css>
Minified CSS: <https://igoradamenko.github.io/awsm.css/css/awsm.min.css>
Normalize CSS: Not Available
Reset CSS: Not Available
Versions: Use normal Use minified

CSS

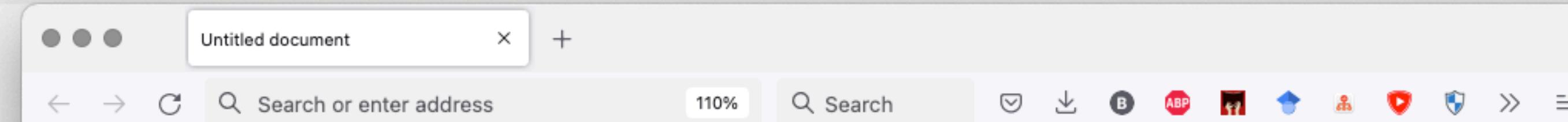
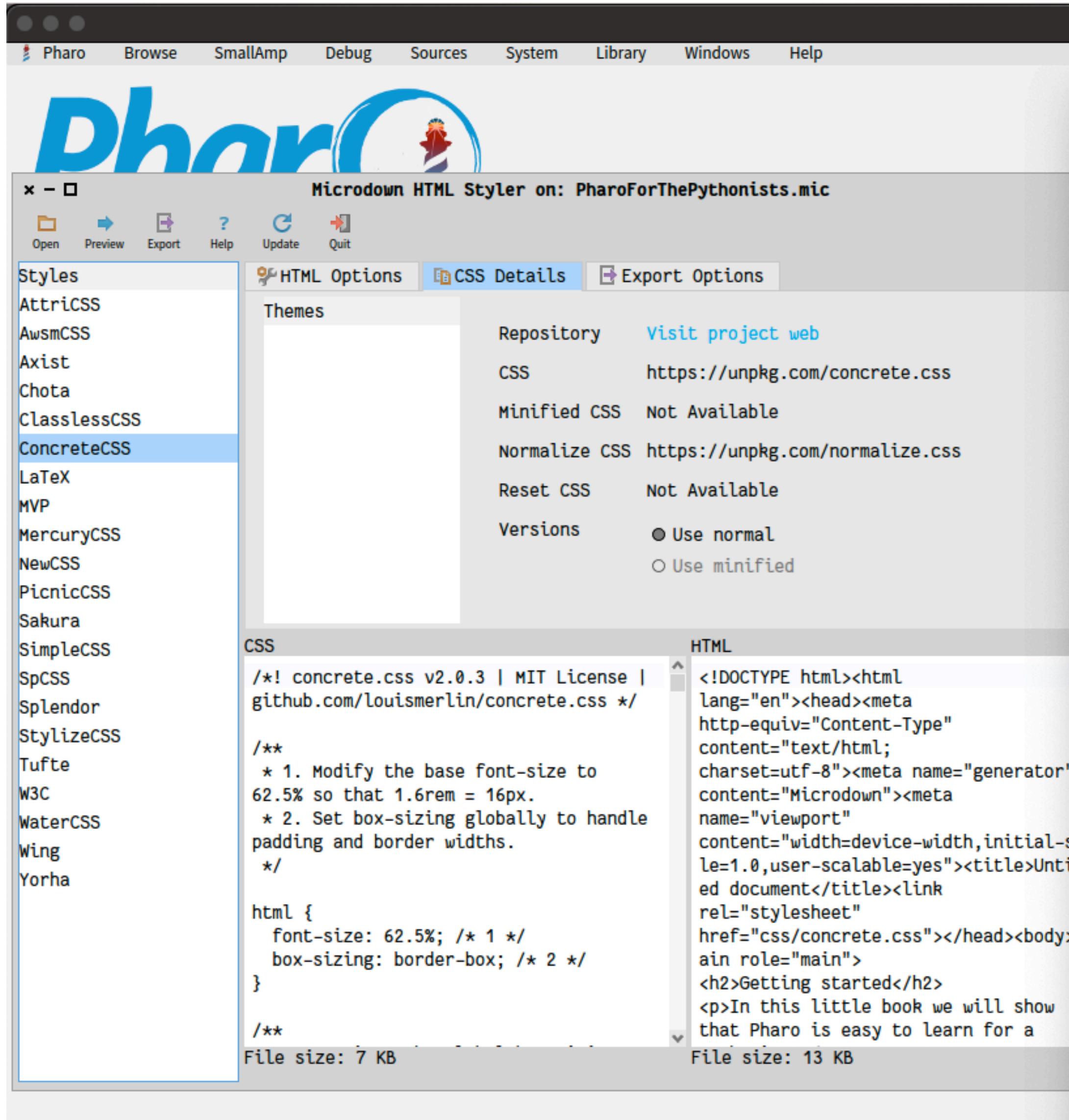
```
@charset "UTF-8";
/*
 * awsm.css v3.0.7
 (https://igoradamenko.github.io/awsm.css/)
 * Copyright 2015 Igor Adamenko
 <mail@igoradamenko.com>
 (https://igoradamenko.com)
 * Licensed under MIT
 (https://github.com/igoradamenko/awsm.css/b
 lob/master/LICENSE.md)
 */
html{font-family:system-ui,-apple-system,Bl
 inkMacSystemFont,"Segoe
 UI",Roboto,Oxygen,Ubuntu,Cantarell,"PT
 Sans","Open Sans","Fira Sans","Droid
 Sans","Helvetica
 File size: 13 KB
```

HTML

```
<!DOCTYPE html><html lang="en"><head><meta http-equiv="Content-Type" content="text/html; charset=utf-8"><meta name="generator" content="Microdown"><meta name="viewport" content="width=device-width,initial-scale=1
 .0,user-scalable=yes"><title>Untitled document</title><link rel="stylesheet" href="css/awsm_theme_gondola.css"></head><body><main role="main">
<h2>Getting started</h2>
<p>In this little book we will show that Pharo is easy to learn for a Pythonist.</p>
<ul>
<li>Dynamically typed</li>
<li>REPL everywhere</li>
</ul>
```

File size: 13 KB

Microdown HTML Styler @ Work



Getting started

In this little book we will show that Pharo is easy to learn for a Pythonist.

- Dynamically typed
- REPL everywhere
- Objects everywhere

Pharo dynamic nature is close to the one of Python.

Now Pharo is just uniform: the computation is expressed using only objects answering messages and lambdas (lexical closure).

Arithmetic

Let's start with basic messages for arithmetic operations.

Basic operators

```
1 + 2  
>>> 3
```

```
1.1 * 2.3  
>>> 2.53
```

Tips: in Pharo all the mathematic operators (+, '-', ...) can be defined on any class.

Pharo Document Browser

Document Browser

Microdown Roadmap

- **Books:**
 - **convert them all** (check if math is fully working)
 - browsable on github AND in HTML AND from Pharo (see next item)
- **Document Browser** in Pharo11
- Resurrect ecstatic for web page