cnrs

Université de Lille

CRIStAL
Centre de Recherche en Informatique,
Signal et Automatique de Lille

Inria
inventors for the digital world
50 YEARS
1967 › 2017 » 2067
Shaping The Future

Microdown

clean and extensible Markdown to support Pharo documentation

S. Ducasse, L. Dargaud, G. Polito

Context: The Pillar Text Edition Toolchain

https://github.com/pillar-markup

- Original Goals
  - Book generation
  - Static Website generation

https://github.com/SquareBracketAssociates/

Problem

New goals

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

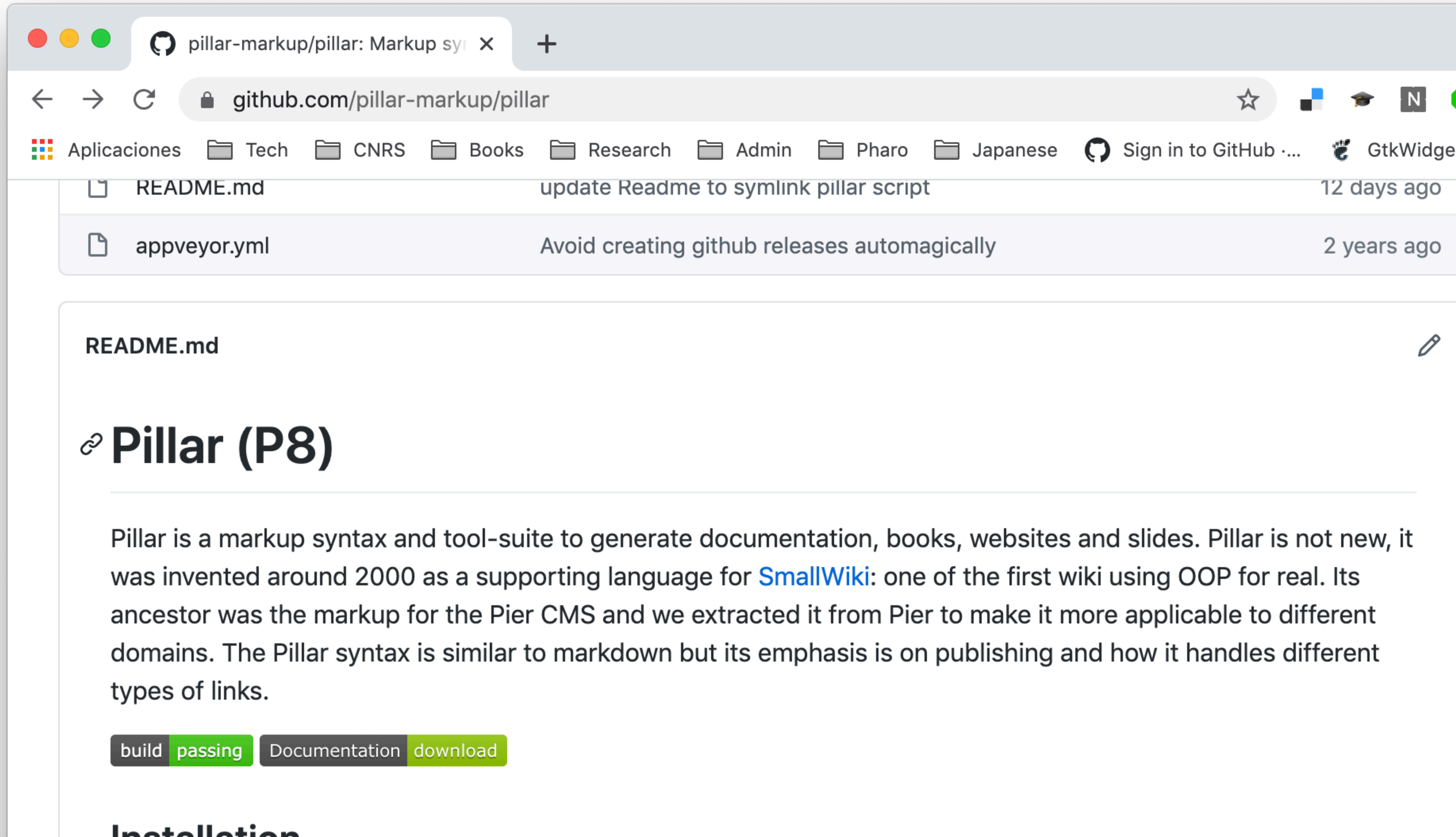
M↓?

Problem

Markdown is a de-facto “standard”

Well-know, 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)



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.

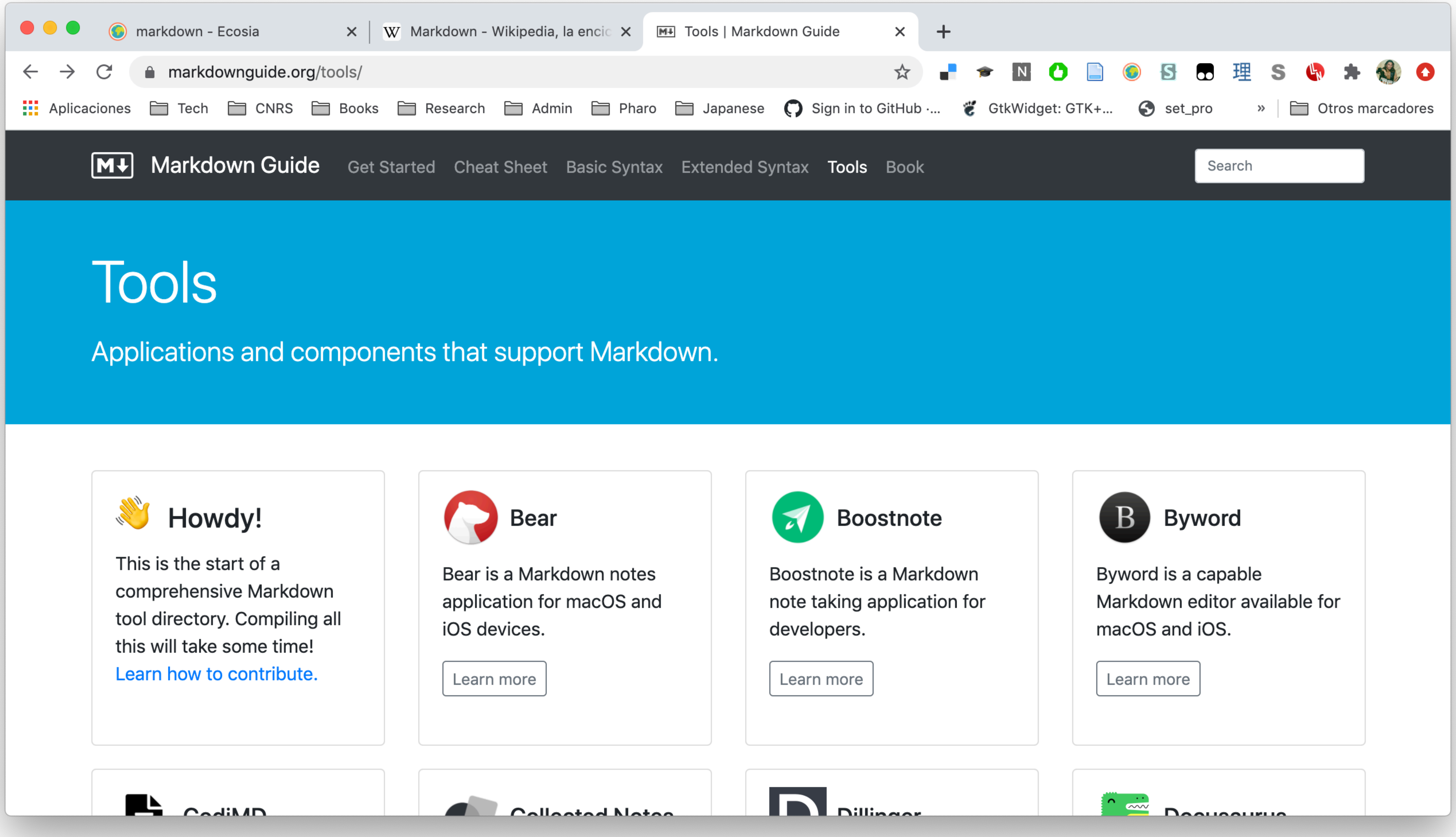
build passing Documentation download

Installation

Problem

But… Markdown is **many** standards

Each tool has a variant



Tools

Applications and components that support Markdown.

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.

Boostnote

Boostnote is a Markdown note taking application for developers.

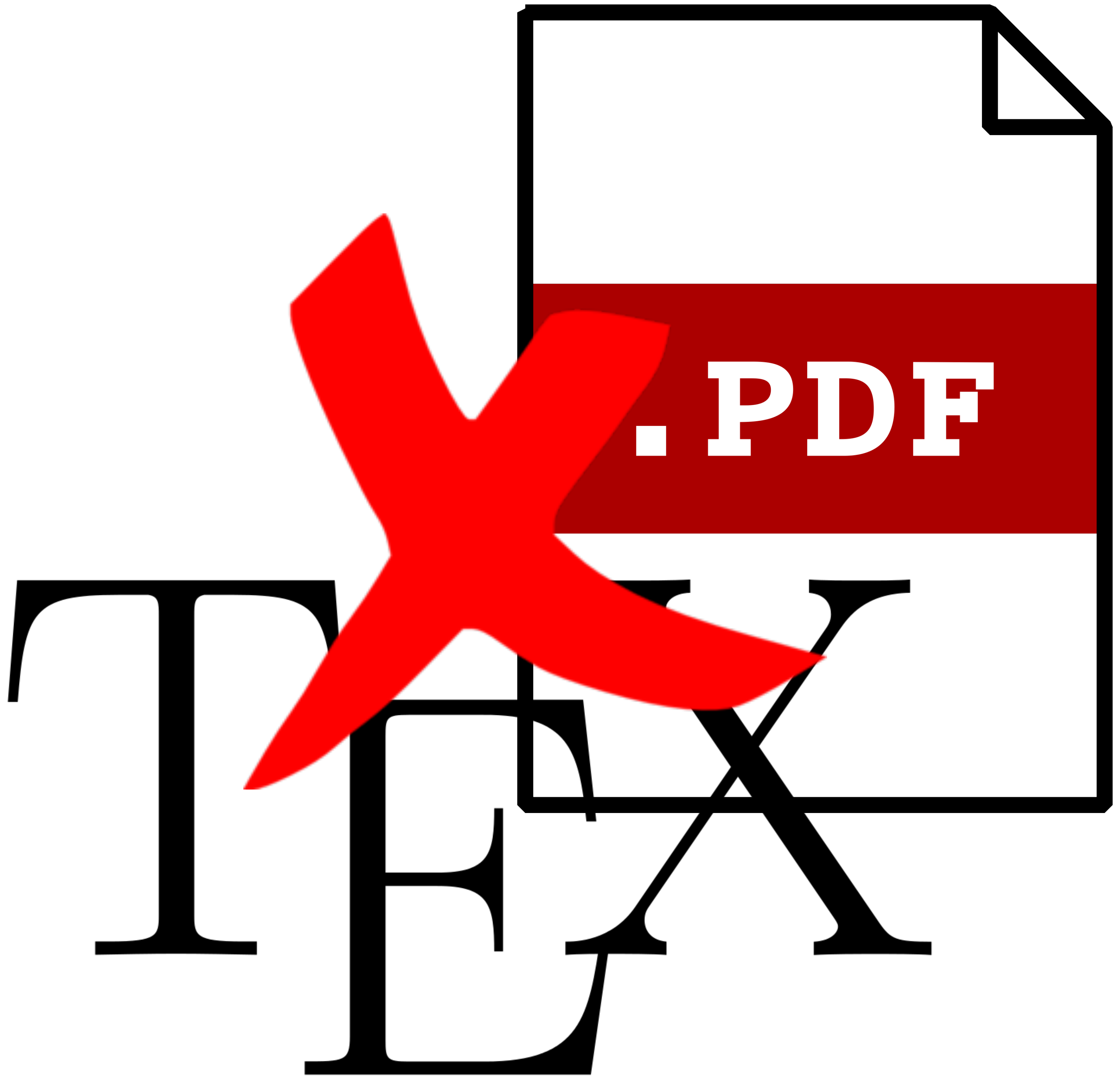
Byword

Byword is a capable Markdown editor available for macOS and iOS.

Problem

And no proper book support

- No explicit anchors
- No figures or code references
- No captions
- Not extensible



Microdown

a clean and extensible Markdown implementation

yes, yet another one!

- Markdown clean and non-ambiguous **subset**

  => compatibility Microdown —> Markdown

- **Extensible**

  => support for books

- A **robust parser** tolerates non-supported syntax

  => compatibility Microdown <— Markdown

A Markdown Compatible Subset

~~~
*** horizontal lines

# … ###### headers

```
code blocks
```

1. ordered
2. lists

* unordered
* lists
~~~

````
[link](https://example.com)

![figure](image.jpg)
````

+ Blockquotes, tables, bold, italics…

Microdown Extensions

(ignored by Markdown)

```
% Comments


# This title has an anchor
@anchor


Extensible annotations
<?footnote | value=footnote is an annotation.?>


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


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

+ inline math

Solution: Microdown

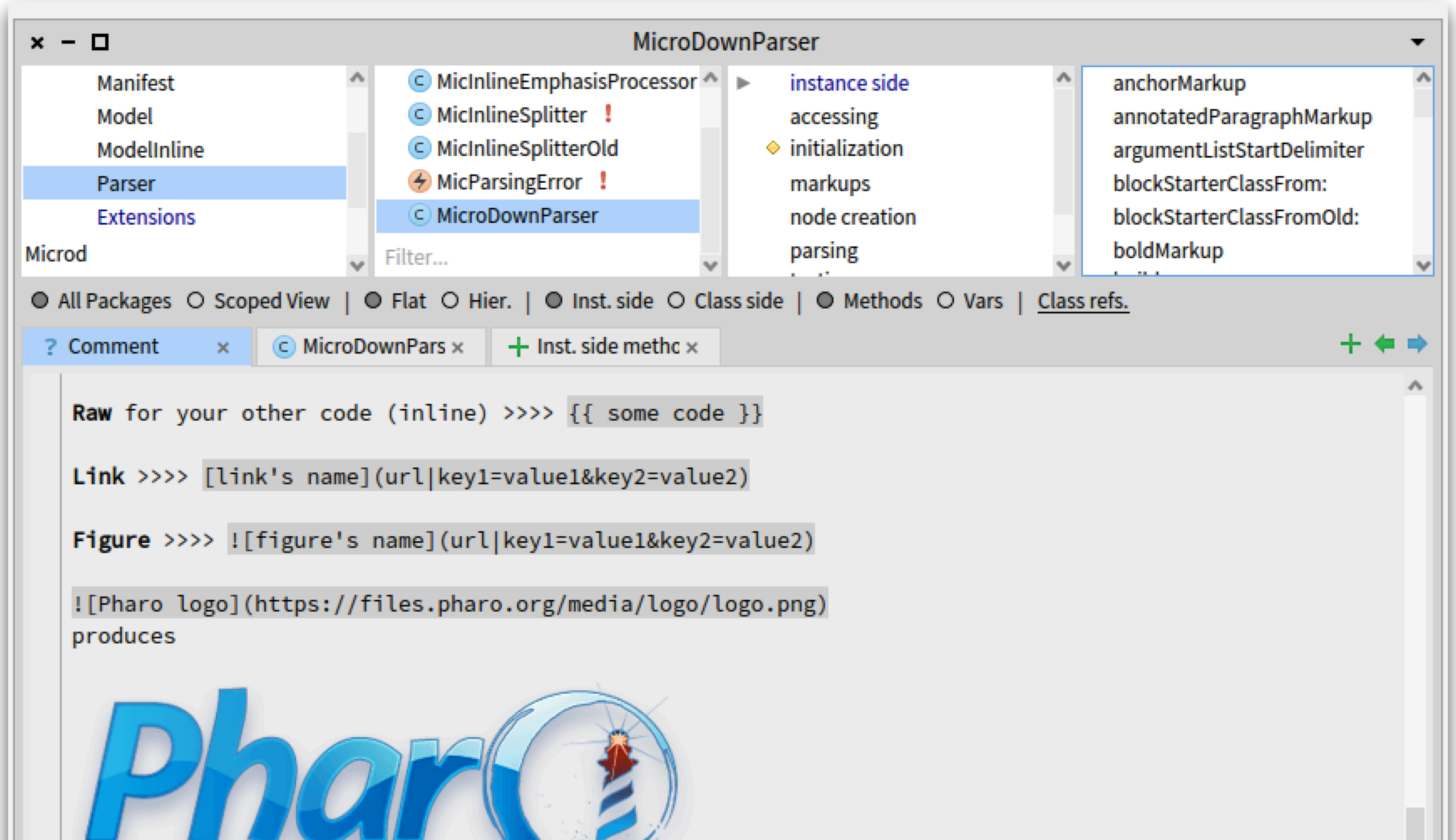
Microdown Robust Parser

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



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

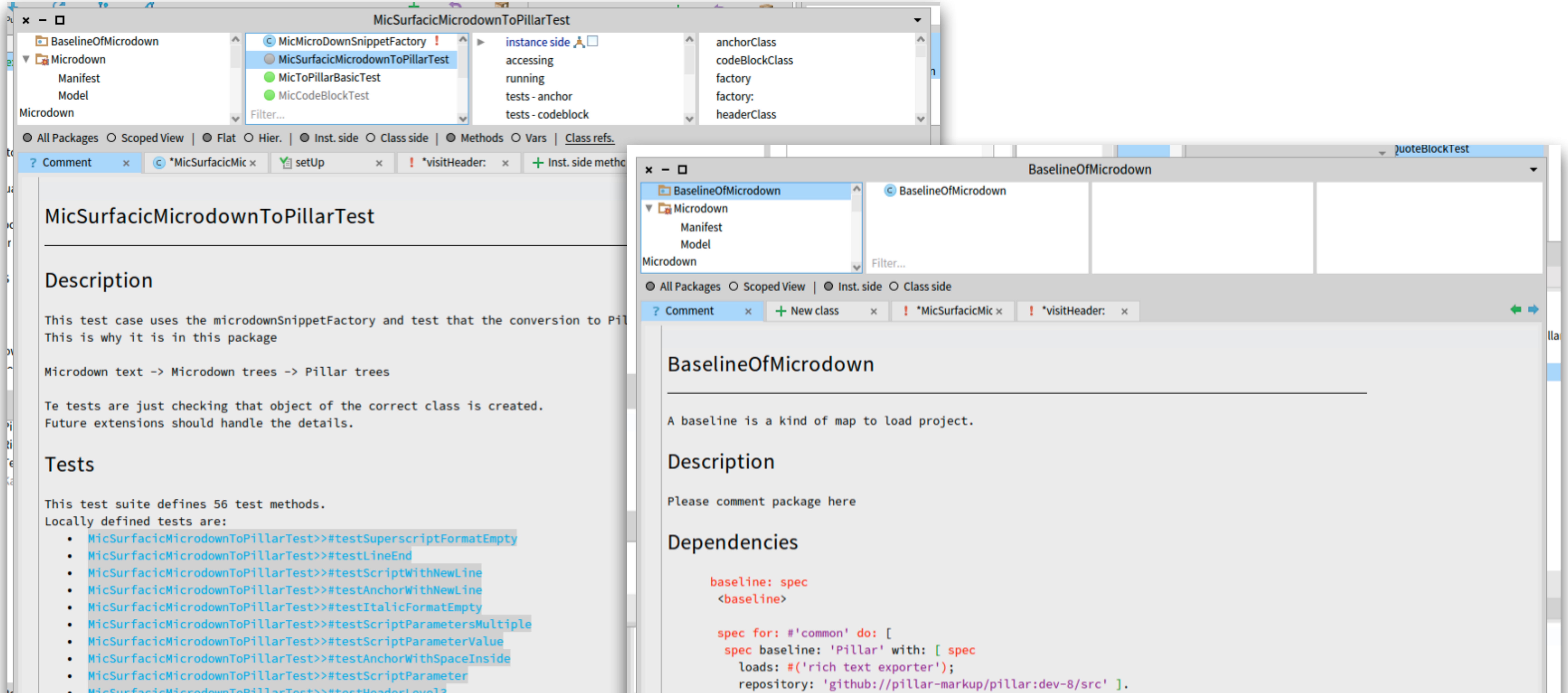
Link >>>> `[link's name](url|key1=value1&key2=value2)`

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

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

Applications

Comment Templates



MicSurfacicMicrodownToPillarTest

Description

```
This test case uses the microdownSnippetFactory and test that the conversion to Pil
This is why it is in this package

Microdown text -> Microdown trees -> Pillar trees

Te 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
```

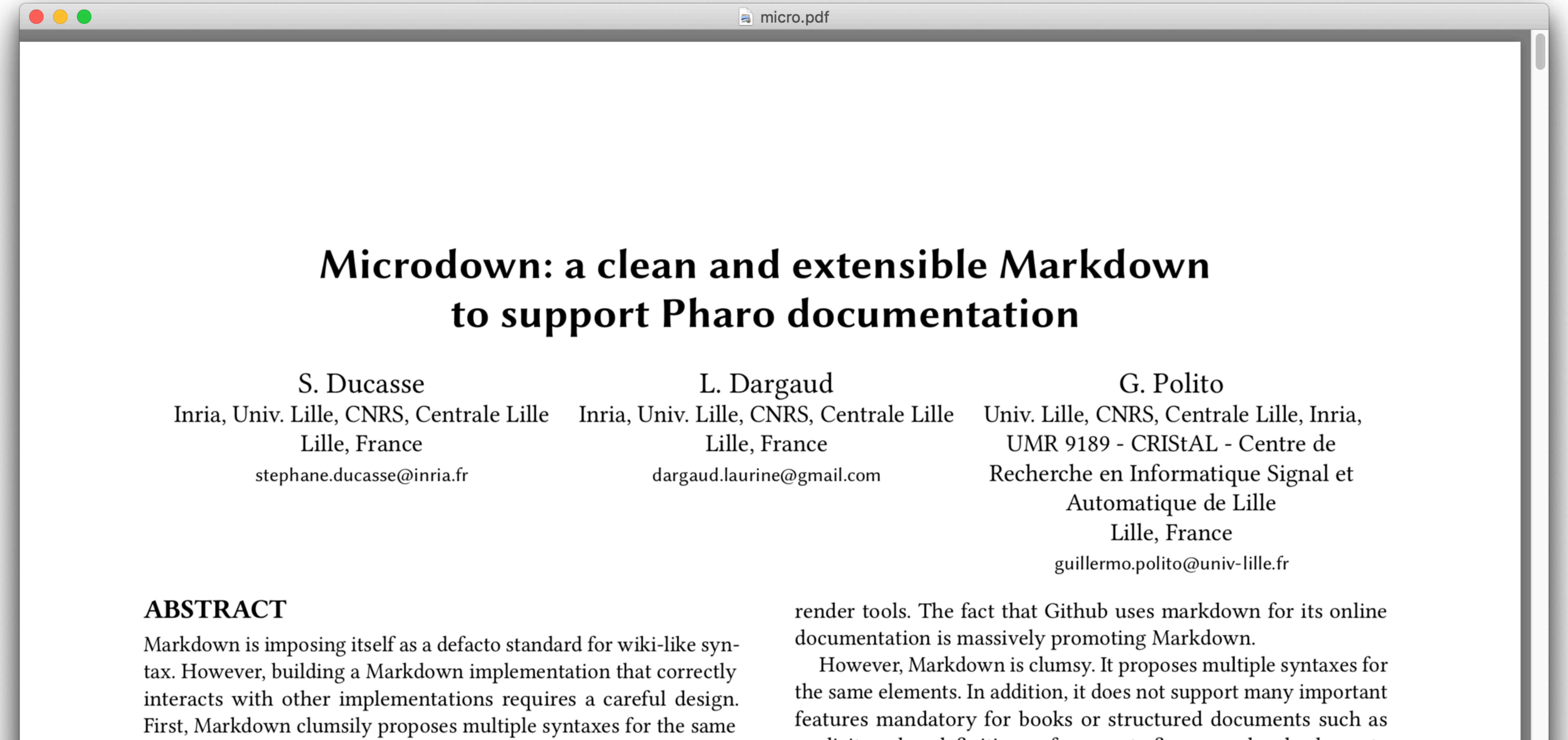
Dependencies

```
baseline: spec
 <baseline>

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

More in the Article!

- Template customisations
- Microdown AST
- Parser details



Microdown: a clean and extensible Markdown to support Pharo documentation

S. Ducasse
Inria, Univ. Lille, CNRS, Centrale Lille
Lille, France
stephane.ducasse@inria.fr

L. Dargaud
Inria, Univ. Lille, CNRS, Centrale Lille
Lille, France
dargaud.laurine@gmail.com

G. Polito
Univ. Lille, CNRS, Centrale Lille, Inria, UMR 9189 - CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille
Lille, France
guillermo.polito@univ-lille.fr

ABSTRACT

Markdown is imposing itself as a defacto standard for wiki-like syntax. However, building a Markdown implementation that correctly interacts with other implementations requires a careful design. First, Markdown clumsily proposes multiple syntaxes for the same

render tools. The fact that Github uses markdown for its online documentation is massively promoting Markdown.

However, Markdown is clumsy. It proposes multiple syntaxes for the same elements. In addition, it does not support many important features mandatory for books or structured documents such as

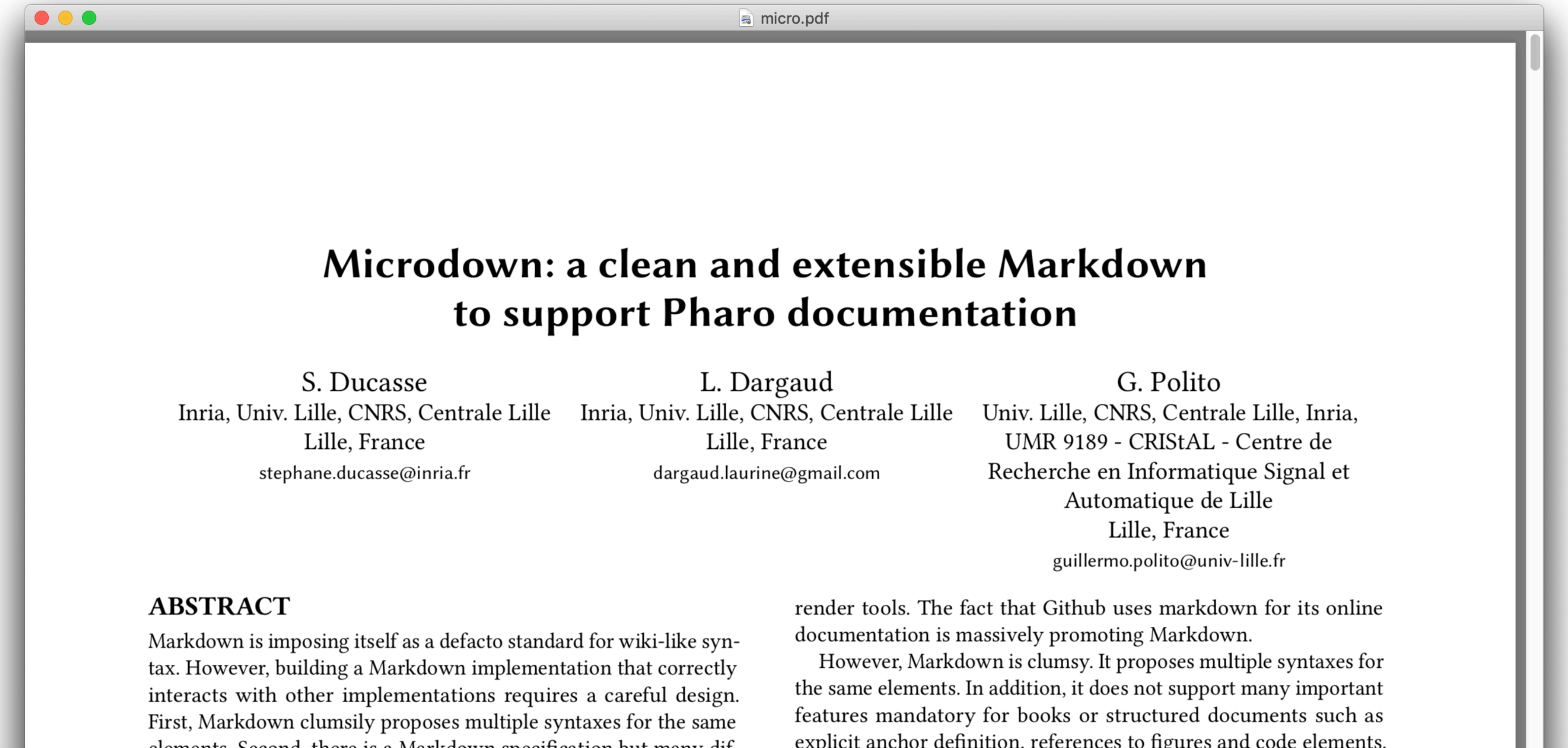
Microdown

clean and extensible Markdown to support Pharo documentation

S. Ducasse, L. Dargaud, G. Polito

- Markdown clean and lean subset
- A robust parser
- Book extensions

- Integration in the Pharo IDE
- And soon Books and Iceberg integration!



Microdown: a clean and extensible Markdown to support Pharo documentation

S. Ducasse
Inria, Univ. Lille, CNRS, Centrale Lille
Lille, France
stephane.ducasse@inria.fr

L. Dargaud
Inria, Univ. Lille, CNRS, Centrale Lille
Lille, France
dargaud.laurine@gmail.com

G. Polito
Univ. Lille, CNRS, Centrale Lille, Inria,
UMR 9189 - CRIStAL - Centre de
Recherche en Informatique Signal et
Automatique de Lille
Lille, France
guillermo.polito@univ-lille.fr

ABSTRACT

Markdown is imposing itself as a defacto standard for wiki-like syntax. However, building a Markdown implementation that correctly interacts with other implementations requires a careful design. First, Markdown clumsily proposes multiple syntaxes for the same elements. Second, there is a Markdown specification but many dif-

render tools. The fact that Github uses markdown for its online documentation is massively promoting Markdown.

However, Markdown is clumsy. It proposes multiple syntaxes for the same elements. In addition, it does not support many important features mandatory for books or structured documents such as explicit anchor definition, references to figures and code elements.