

Markdownで書く電子書籍開発環境

自己紹介

- Name : **azu**
- Twitter : @azu_re
- Website: [Web scratch](#), [JSer.info](#)



JavaScript Plugin Architecture	
README	✓
1. Introduction	✓
2. jQuery	✓
3. ESLint	✓
4. Connect	✓
Published with GitBook	

JavaScript Plugin Architecture

build passing

この書籍はJavaScriptのライブラリやツールにおけるプラグインアーキテクチャを見ていく事を目的としたものです。

Introduction

JavaScriptの世界では一つの大きなライブラリよりも小さいなものを組み合わせていくようなスタイルが多く見られます。

小さなものを組み合わせて使えるようなエコシステムの土台となるものを書こうとした際に、プラグインアーキテクチャが重要となると言えます。

ソフトウェアの構造に「プラグイン機構」を設け、ユーザコミュニティから開発者コミュニティへの質的な転換を図るのは、ソフトウェア設計からエコシステム設計へとつながる -- OSS開発の活発さの維持と良いソフトウェア設計の間には緊張関係があるのだろうか? - t-wadaのブログ

この書籍では、そのプラグインアーキテクチャや仕組み、エコシステムを形成してるライブラリやツールなどの実装から学ぶことを目的にしています。

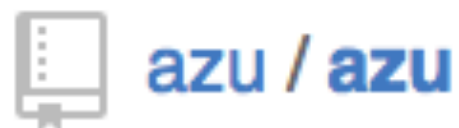
Installation

JavaScript Plugin Architecture

JavaScript Plugin Architecture

電子書籍の開発中にやったこと

やりたかったこと



👁 Unwatch ▾

5

★ Unstar

[Markdown] 電子書籍開発環境 #42

🔔 Open

azu opened this issue 7 months ago · 6 comments



azu commented 7 months ago

Owner



コンセプト: もっと気軽に書ける電子書籍

もっと簡単に色々自動化できる電子書籍の書き方についてを考える。

できればASTを扱うツールの作り方のようなテーマで実際に書いてみて、コンセプトを実証したい。

[Markdown] 電子書籍開発環境・Issue #42

- Markdown/文章のLint
- インラインコードのLint
- ファイルのincludeするMarkdown拡張
- Markdown to HTML or PDF
- エディタ

[Markdown] 電子書籍開発環境・Issue #42

- Markdown/文章のLint => [textlint](#)
- インラインコードのLint => [ESLint](#)
- ファイルのincludeするMarkdown拡張 => [GitBook](#)+プラグイン
- Markdown to HTML or PDF => [GitBook](#)
- エディタ => [GitBook Editor](#)

GitBook



GitBook

- Markdownで電子書籍を書けるツール/プラットフォーム
- [GitbookIO/gitbook](https://gitbook.io/gitbook)
 - Markdown -> HTML/PDF/Epubの変換
 - 各章を書いてSUMMARY.mdにリンクを書くだけで作れる
 - プラグインで拡張できる

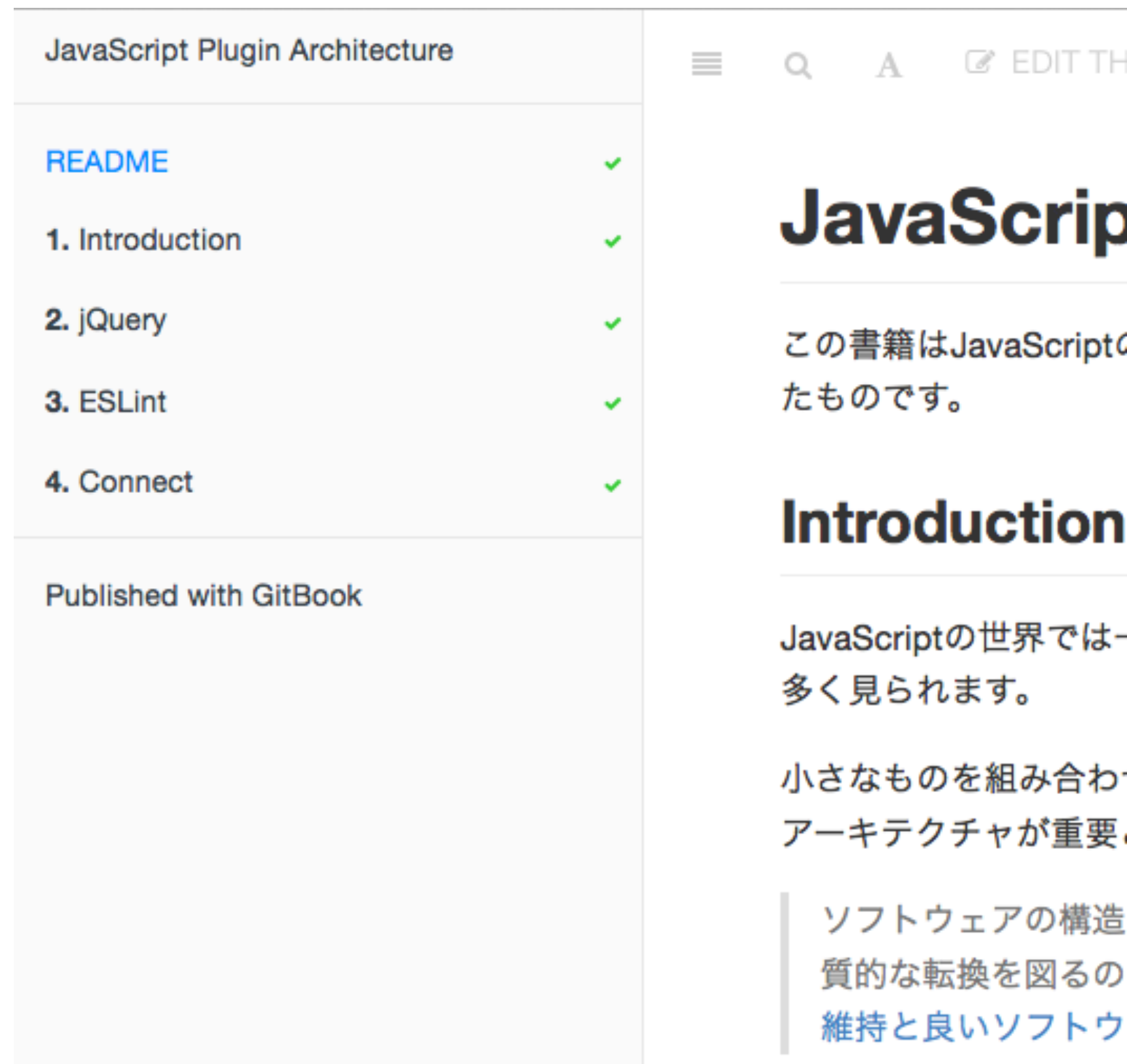
gitbook.com

- GitBookの公開プラットフォーム
- とても良く出来てる
- HTML/PDF/Epubの自動生成、コミット毎プレビュー、販売/寄付、Organization、アップデート通知、オンラインエディタ
- GitHubとDeployment APIでhookして自動的に反映できる

✓ GitBook連携の設定・Issue #4・
azu/JavaScript-Plugin-Architecture

GitBookの構造

- - └─ README.md
 - └─ SUMMARY.md
 - └─ book.json
- SUMMARY.md から各章へのMarkdown
ファイルへリンク
- gitbook build => 静的なHTMLが生
成される



JavaScript Plugin Architectureの構造

- └─ README.md
- └─ SUMMARY.md
- └─ book.json
- └─ src(*.js)
 - └─ jQuery
- └─ ja (*.md, *.png 文章関係)
 - └─ jQuery
- └─ test (*-test.js)
 - └─ jQuery
- └─ package.json

✓ プロジェクト構造について・Issue

#5 ・ azu/JavaScript-Plugin-
Architecture

textlint

- Markdownやテキスト向けのLintツール
- テキスト版ESLintツール
 - ルールをJavaScriptで簡単に追加できる
- [textlintで日本語の文章をチェックする | Web Scratch](#)

textlint rule

- [textlint-rule-max-ten](#)
 - 一文に利用できる、の数をチェックするルール
- [textlint-rule-spellcheck-tech-word](#)
 - WEB+DB用語統一ルールベースの単語チェックするルール
- [textlint-rule-no-mix-dearu-desumasu](#)
 - 「ですます」調と「である」調の混在をチェックするルール

textlint rule

- [textlint-rule-no-start-duplicated-conjunction](#)
 - 「しかし、～ 。 しかし、～。」 など同じ接続詞が連続していないかをチェックするルール
- textlintのルールは以下にまとめられている
 - [Collection of textlint rule](#) ・ [azu/textlint Wiki](#)

プロジェクト固有の表記揺れ

- 表記揺れのチェックに汎用的な辞書/ルールはない
 - 全ての表現が一意ならそもそも表記揺れなんて起きない
 - 書籍の中で一貫した表現を保証するためのもの
- プロジェクト固有のルールで表記揺れのチェックが必要

textlint-rule-prh

- [vvakame/prh](#) を利用したtextlint rule
- yamlでルールを簡単に追加できる(正規表現や大文字小文字などよくある表記揺れは簡単に書ける仕組みがある)

rules:

- expected: プラグインアーキテクチャ

pattern:

- プラグイン機構
- プラグインのアーキテクチャ

textlint-rule-prh

- textlint-rule-prhについては詳しくは以下の記事を参照
- [textlint + prhで表記ゆれを検出する | Web Scratch](#)

なぜプロジェクト毎に表記揺れルール？

- typoなどを見つけた場合にルールを追加して**から**修正できる
- **Connect**に統一しよう・Issue #48・azu/JavaScript-Plugin-Architecture
 - リグレッションテストと同じ意味合い
- 表記がルールとして明文化できるので**Contribute**しやすい

GitBook + textlint

- GitBookはSUMMARY.mdから各章の.mdへのリンクがある
- [azu/gitbook-summary-to-path](#)
- SUMMARY.mdに書かれているファイルをtextlintする

```
$ summary-to-path SUMMARY.md | xargs textlint
```

```
# 全ての章がtextlintでLintできる
```

✓ textlintの導入・Issue #1・azu/
JavaScript-Plugin-Architecture

コードのLint

- コードをESLintでチェックしたい
- 技術書に載せるコードを書く方法は2種類
 - コードを外部ファイルとして書いて読み込む
 - インラインにコードを書く

外部ファイルのコード

- [azu/gitbook-plugin-include-codeblock](#)
- いい感じに外部ファイルをCodeBlockとして読み込むGitBookプラグイン

```
[include, test.js](fixtures/test.js)
```

と書けば、CodeBlockとして展開される。

=> GitHub上ではただのリンクとなる(fallback)

外部ファイルのコードをLint

- ESLintを使い単純にJavaScriptとしてLintを通す
 - ✓ ESLintの導入・Issue #6・azu/JavaScript-Plugin-Architecture

インラインコードのLint

これは`a`という変数を定義している。

```
```js  
var a = 1;
```
```

- インラインに書かれているコードに対してもLintを行う
- インラインコードは実行されないなのでtypoし易い

インラインコードのLint

- [eslint/eslint-plugin-markdown](#)を利用
- ESLintのプラグインとしてインラインコードをLintできる
 - jsやjavascriptといったCodeBlockに対してLint

インラインコードのLintの問題

- 問題: インラインコードは実行できないのが正常というケース！
- 説明するためにコードの一部を取り出す場合
 - コードとしては実行できない
 - コードブロックのみで見ると変数が未定義となっている
- => インラインコード専用のゆるいルールを作る

インラインコード専用のゆるいルール

- 設定ファイルを分けることで解決！
- 通常のコード用: [.eslintrc](#)
- インラインコード用: [.md.eslintrc](#)
 - [.eslintrc](#)を継承
 - `no-undef`や`no-unused-vars`などを無効化

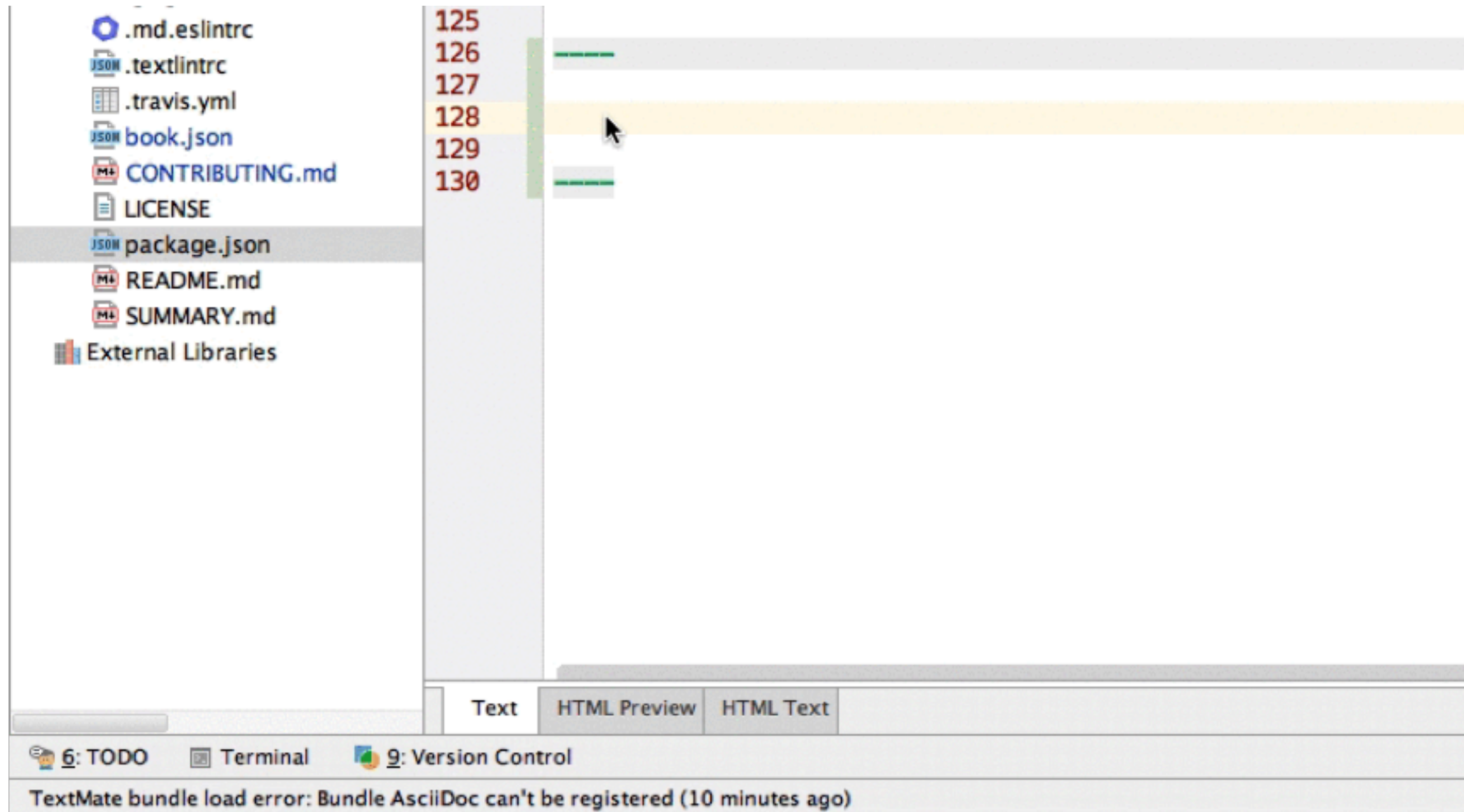
✓ ESLintでインラインコードのLint

• Issue #7 • [azu/JavaScript-Plugin-Architecture](#)

エディタ

- 殆ど素のMarkdownなので好きなエディタが使える
- GitBookの[公式エディタ](#)もある
- WebStorm、Atom、Markdownエディタなどでよい

WebStorm + File Watch + textlint



Atom + 1000ch/linter-textlint



The screenshot shows the Atom text editor interface. The main editor area displays a text file with line numbers 126 through 133 on the left margin. Line 128 contains the text `[jquery.com] (http://jquery.com) is safe.`. The `jquery.com` part is highlighted in blue, and the `(http://jquery.com)` part is highlighted in purple. A red dashed underline is visible under `jquery.com`. The status bar at the bottom shows `File 0`, `Project 0`, a green checkmark icon followed by `No Issues`, the file path `ja/jquery/README.md*`, and the cursor position `130:1`.

```
126  -----  
127  
128  [jquery.com] (http://jquery.com) is safe.  
129  
130  
131  
132  -----  
133
```

File 0 Project 0 ✓ No Issues ja/jquery/README.md* 130:1

CONTRIBUTING.md

- [Contributing Guidelines](#)を書いてみる
- 書こうと思うと、どういう手順でプレビューできるか、修正するか、文章を書いていくかの整理が必要になる
- 自分のためでもあり、Contributingする人のためになる

✓ CONTRIBUTING.md · Issue #12 · azu/
JavaScript-Plugin-Architecture

August 21, 2015 – September 21, 2015

Period: 1 month

Overview



42 Active Pull Requests



29 Active Issues

42

Merged Pull Requests

0

Proposed Pull Requests

23

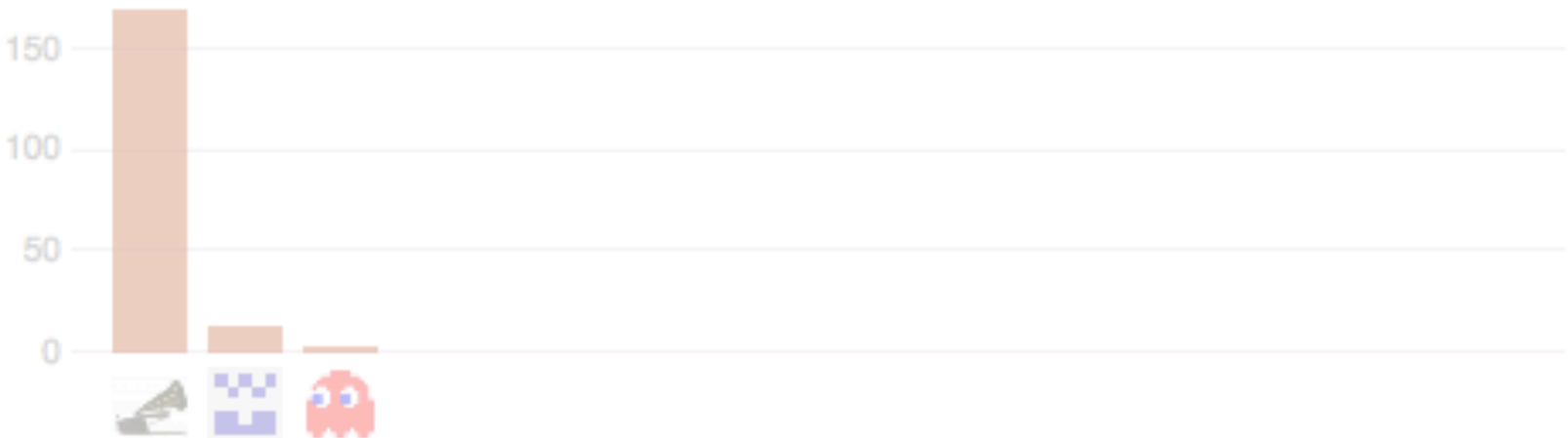
Closed Issues

6

New Issues

Issue/Pull Request 駆動

Excluding merges, **3 authors** have pushed **179 commits** to master and **182 commits** to all branches. On master, **41 files** have changed and there have been **1,599 additions** and **8 deletions**.



Issue/Pull Request駆動

- 文章の正しさは人により異なるので根拠を残す
 - コードと違って曖昧成分が多い
 - 文章の自動チェックを入れた理由を残す
- Pull Request駆動で文章もCIを通してから
 - 検証済みマージ - マージされるとGitBookに自動反映

Issue

- Issueで設計をしてから文章を書く
- 気になったことはとりあえずIssueとしてメモ書き
- 参考リンクとかをコメントにどんどん書いていく
- ちょっとづつ進められるようにタスクをIssueとして細分化
 - => 飽きやすいのを防止するため

GitHubと電子書籍

- Promise本での話
- Githubで書く電子書籍
- Promise本で取り組んだ電子書籍の開発ツール、CI、継続的リリースについて

まとめ

- [JavaScript Plugin Architecture](#)という電子書籍で実践してる事
- Markdownで書いてGitBookでビルド、公開してる
- textlintで文章をチェック
- JavaScriptやインラインコードをESLintでチェック
- Issue/Pull Request駆動でモチベーションを保つ

疑問点

- 登場してきたIssueに書いてみるといいかも!?