

Markdown

でプレゼンテーション！

このプレゼンテーションはMARKDOWNで書いてます

例えば↓が・・・

```
# おまけ
```

```
おまけです
```

```
----
```

```
## このプレゼンテーションはMarkdownで書いてます
```

- 説明
 - + 説明
 - + 説明

おまけ

おまけです

このプレゼンテーションはMARKDOWNで書
いてます

- 説明
 - 説明
 - 説明

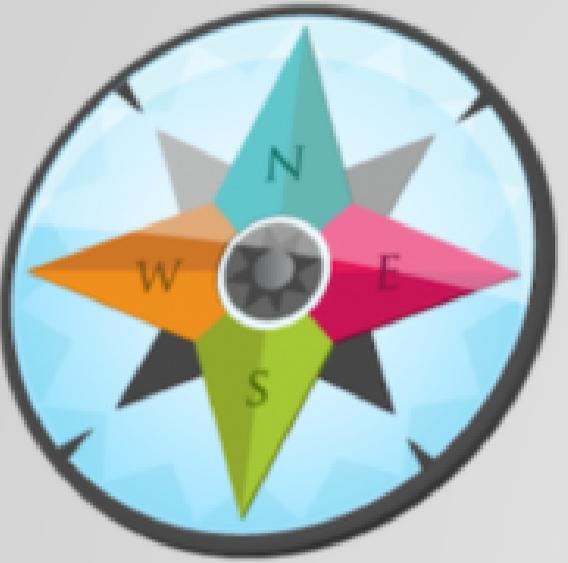
例えば↓が・・・

```
## expectJS
```

```
! [expectJS] (image/LearnBoost.png) ( 作者のLearnBoostさんのアイコン )
```

- ****should.js**** をベースに開発されたミニマムなBDDアサーションライブラリ
- ****mongoose**** や ****stylus**** の作者が作成
- クロスプラウザ: IE6+, Firefox, Safari, Chrome, Operaで動作
- 全てのティングフレームワークと併用可能
- Node.JSで使用可能(`require('expect.js')`)
- スタンドアローン

EXPECTJS



(作者のLearnBoostさんのアイコン)

- **should.js** をベースに開発されたミニマムなBDDアサーションライブラリ
- **mongoose** や **stylus** の作者が作成
- クロスプラウザ: IE6+, Firefox, Safari, Chrome, Operaで動作
- 全てのテスティングフレームワークと併用可能
- Node.JSで使用可能(`require('expect.js')`)
- スタンドアローン

MDPRESS



(作者のAditya Bhargavaさん・・・読みない汗)

- MarkDown文書からプレゼンテーションを生成するgem

```
$ mdpress readme.md
```

- これだけで **readme** というフォルダが作成され、その中の **index.html** を開くとプレゼンテーションが始まります

DEMO: このプレゼンテーションのソース (Sublime)

MDPRESSの良いところ

- Markdownラクチン！
- とりあえずLT用に見出しだけ作って、
- LT後に加筆すれば記事になります
- できた記事をQiita・はてなブログに貼り付ければパブリッシュ完了
- 無駄がないワークフロー

シンタックスハイライト : RUBY

```
configure :production do
  set :cache, Dalli::Client.new(
    ENV['MEMCACHE_SERVERS'],
    :username => ENV['MEMCACHE_USERNAME'],
    :password => ENV['MEMCACHE_PASSWORD'],
    :expires_in => 60 * 30
  )
end
```

シンタックスハイライト : OBJECTIVE-C

```
NSBundle* bundle = [NSBundle mainBundle];
NSString* path = [bundle pathForResource:@"Questions" ofType:@"plist"];
NSArray* questions = [NSArray arrayWithContentsOfFile:path];
NSUInteger QuestionsMax = questions.count;

for(NSDictionary* question in questions) {
    NSLog(@"%@", [question objectForKey:@"Question"]);
    NSLog(@"%@", [question objectForKey:@"CorrectAnswer"]);
    NSLog(@"%@", [question objectForKey:@"IncorrectAnswer"]);
    NSLog(@"%@", [question objectForKey:@"backgroundImage"]);
}
```

- 言語別のシンタックスハイライトには未対応
- しかたないので**Online syntax highlighter like TextMate**でシンタックスハイライトします

シンタックスハイライト : OBJECTIVE-C

```
NSBundle* bundle = [NSBundle mainBundle];
NSString* path = [bundle pathForResource:@"Questions" ofType:@"plist"];
questions = [NSArray arrayWithContentsOfFile:path];
QuestionsMax = questions.count;

for(NSDictionary* question in questions) {
    NSLog(@"%@", [question objectForKey:@"Question"]);
    NSLog(@"%@", [question objectForKey:@"CorrectAnswer"]);
    NSLog(@"%@", [question objectForKey:@"IncorrectAnswer"]);
    NSLog(@"%@", [question objectForKey:@"backgroundImage"]);
}
```

しかし・・・

Markdown文書を編集

→mdpressコマンドを実行

→ブラウザに移る

→ブラウザを更新

→Markdown文書の編集に戻る

めんどくさい

MDPRESS-GENERATOR



(作者のBrian Holtさん)

- mdpressの作業を自動化します

```
$ yo mdpress
```

- これで自動化準備完了

```
$ grunt server
```

これで

- プрезентーションをブラウザで開き
- mdpressのソースファイルを監視し
- 更新があるとプレゼンテーションを更新し
- ブラウザの更新（LiveReload）までしてくれます

つまり全自动

2画面あると、片方で編集、もう片方で仕上がりのチェックができます

では作ったプレゼンテーションをどこに置くか？

GITHUB PAGESでプレゼンテーション

DEMO : このプレゼンテーションのリポジトリ

つくり方

1. まず、普通にMarkdown文書を作る
2. \$ mdpress readme.mdでプレゼンテーションが *readme* フォルダに作られる
3. originブランチにpush
4. Setting -> GitHub Pagesで空のGitHub Pagesをつくる
5. gh-pagesブランチがGitHub上にできる
6. pullするとgh-pagesブランチもローカルに付いてくる
7. \$ git checkout gh-pages
8. \$ cp -rf readme/* .
9. \$ git push -u origin gh-pages (最初だけ)
10. \$ git push (2回目以降)

良いところ

- 改訂したとき、diffが見れる
- プルリクも送ることができる
- ソース(Markdown)とプレゼンテーション(HTML)を同じ一つのリポジトリで管理できる
- Qiitaやはてなブログのように、画像を一枚一枚アップしなくて良い
- LT後の質疑などの内容を、筆者だけでなく聞いた人もプルリクで追記できる

悪いところ

- 更新がややこしい

更新のやり方

1. [origin] まずoriginブランチであるか確認
2. [origin] Markdownに追記
3. [origin] Markdownエディタを終了
4. [origin] git push
5. [origin] mdpressでリポジトリとは別のフォルダにプレゼン作成
6. [gh-pages] gh-pagesブランチに切り替える
7. [gh-pages] プrezentをリポジトリにコピー
8. [gh-pages] git push
9. [origin] originブランチに戻しておく

スクリプト化してみる：

PUSH.SH

```
# レポジトリに入る
# フォルダ名は引数にしたい
cd 140127-2013-soukatsu-2014-houshin

# Markdownをpush
git add .
git commit -m "committed automatically by push.sh"
git push

# mdpressコマンドでreadmeフォルダを生成
cd ..
mdpress 140127-2013-soukatsu-2014-houshin/readme.md
```

```
# gh-pages ブランチに切り替える
cd 140127-2013-soukatsu-2014-houshin
git checkout gh-pages

# 先ほど生成したreadmeフォルダの中身をレポジトリにコピーする
cp -rf ../readme/* .

# 自動的にcommit+push
git add .
git commit -m "committed automatically by push.sh"
git push

# origin ブランチに戻す
git checkout master

# 元いたディレクトリに戻る
cd ..
```

めんどくさい

GULP-GH-PAGES



(作者のMichael Benedictさん。Twitter社に勤務。)

```
$ gulp deploy
```

これで全部やってくれます
ひえー

さて、プレゼンが終わって

- 作ったプレゼンをSpeakerDeckやSlideShareにアップしたい
- 今まででは・・・HTMLページを一枚ずつPDFにして、
- あとでたばねて一つのPDFファイルにしていた

めんどくさい

DECK2PDF



(作者のCedricさん。フランスからPivotal社にリモート勤務)

```
$ deck2pdf --profile=impressjs index.html
```

これで1つのPDFファイルにしてくれます

ひえー

まとめ

- mdpress
 - mdpress-genarator
 - gulp-gh-pages
 - deck2pdf
- ・・・ で、MarkdownでGitHubな
プレゼンテーションライフを実現しましょう
ご清聴ありがとうございました