Markdownのゆるい話

sylph01 / Ryo Kajiwara

Twitter: @s01

http://s01.ninja/

2014.11.23

資料はここ(予定)

http://s01.ninja/md_yurui.pdf

中の人について

sylph01 / 梶原 龍

Twitter: @s01

オーケストラでファゴット吹いてます

(技術的にはただの人)



コンセプト

「社会人」は忙しい。 なので仕事の本質に集中する必要がある。

プラットフォームは本質ではない

文書はMicrosoft Wordからのみ得られるものではない 同じ本質的機能を実現するより使い慣れた ツールがある

作業環境も本質ではない

それhogehogeでしかできないって本当ですか

Markdown をはまるう

What is Markdown

John Gruber氏によって考案された構造化 テキストフォーマット 可読性を保ちながら機械処理もしやすい

こんな感じ

Markdownのゆるい話

sylph01 / Ryo Kajiwara

Twitter: @s01

http://s01.ninja/

コンセプト

「社会人」は忙しい。 なので仕事の本質に集中する必要がある。

プラットフォームは本質ではない

文書はMicrosoft Wordからのみ得られるものではない 同じ本質的機能を実現するより使い慣れたツールがある

Why Markdown

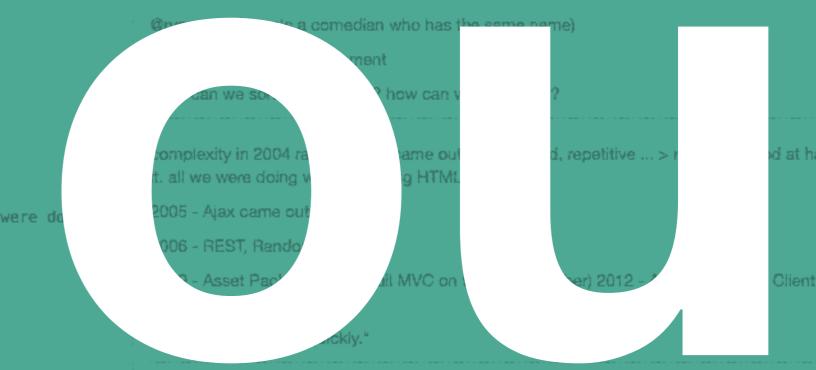
プレーンテキスト Gitに入れてdiffが取れる エディタの縛りを受けない 機械処理しやすいフォーマット 対応ツールが多い デザインとコンテンツの分離

対応ソーノ





Isomorphic App Development with Ruby and Volt



What's the solution?

nates/Auth in Client turbo-links? (...just kidding)

That es 軽量なMac向けMarkdown Large Ruby (pal - Say to JavaScript compiler

Volt lets you share MVC -> same code runs on client and server!

iPad版もある

ic ("similar in form") code between the client and server

Ill MVC on the client (Ember)

set loading in Client (Bower)

somorphic framework in Ruby uby to JavaScript compiler

set Packing

perception - compiling to JS adds complexity

hard to build large JavaScript apps

Stockholm Syndrome WTF's

Min+Deflate -> Angular < Opal+Volt < Ember



what can we so rearrange? how can we simplify it?

complexing 上記量はMac向けMarkdown Line is your producing HTML

GitHub Flavored Markdown寄りのレンダ

2006 - REST, Random JS

2009 - Asset Pasking

2011 - Full MVC on the client (Ember)

2012 - Asset loading in Client (Bower)

2013 - Routes/Auth in Client

"Well, that escalated quickly."

What's the solution?

turbo-links? (...just kidding)

2009 - Asset Packing 2011 - Full MVC on the client (Ember) 2012

Just came out unorganized, repetitive.

- Asset loading in Client (Bower) 2013 - Routes/Auth in Client

"Well, that escalated quickly."

What's the solution?

turbo-links? (...just kidding)

isomorphic ("similar in form") sharing code between the client and server

Volt - Isomorphic framework in Ruby Opal - Ruby to JavaScript



Isomorphic App Development wit

it can rearrange' mplexi ails has jus doing was p TML ndling it. al

2005 - Ajax came out (w/Gmail)

2006 - REST, Random JS

wolt.md Preview

モダンテキストエディタの最先鋒ももちろ

ん対応

Well, that escalated quickly.

What's the solution?

turbo-links? (...just kidding)

最も正しいGitHub Flavored Markdownの

レンダリング

Volt - Isomorphic framework in Ruby Opal - Ruby to JavaScript compil

Volt lets you share MVC -> same code runs on client and



今プレゼンしてるやつ

日本語対応がまだちょっと微妙

数式サポートもある (例: $e^{i\pi}=-1$)

\$30, 数式サポートで+\$10



HTML, PDF, LaTeX, EPUB, docx(!)など、Markdownなどから大抵のドキュメントフォーマットを得られるコマンドラインツール



No more databases, comment

Markdown (or Textile), Liquid,

GitHub Pages O TL SSS Strick sites come out ready for deployment

posts, and custom layouts are all first-class citizens here.

Markdownから静的サイトを得るte your blos →

Get up and running in seconds.

- ~ : gem install jekyll
- jekyll new my-awesome-site
- \sim $^{\circ}$ cd my \sim awesome \sim site
- ~/my-awesome-site 🦠 jekyll serve



応用例: Webサイトをまるっとバージョン管



サブモジュール

~~Gitに入れる

ーサーバー側でpost-update hookでJekyll

を走らせてサイト更新

コミット: 3c974a4454a187176b00ec78c39b323f67ff3047

[3c974a4]

親: <u>879a4d75b5</u>

作者: Ryo Kajiwara <kfe-fecn6.prussian@s01.info>

日時: 2014年11月10日 12:54:34 JST ラベル: origin/master origin/HEAD ?

email: your-email@domain.com

description: > # this means to ignore newlines until "baseurl:"

Web site of sylph01.

- baseurl: "/" # the subpath of your site, e.g. /blog/

- url: "http://s01.info" # the base hostname & protocol for your site

baseurl: "" # the subpath of your site, e.g. /blog/

url: "http://s01.info/" # the base hostname & protocol for your site

url: "http://s01.info/" # the base hostname & protocol for your site
twitter_username: s01
github_username: sylph01

Build settings
markdown: kramdown

Site settings
title: section 01

■ D master ②クリー

Atlaccian

応用例2: ドキュメントレビュー、ドキュメントCI

- ードキュメントをGitでバージョン管理
- GitHub上で内容をレビュー
- ーコミットに対して最終結果をPandoc等で 自動生成
- ー最後にデザイン等を調整

応用例3: どこでもスライド作成

- 同様にGitでバージョン管理
- ーレポジトリの内容をもとにDeckset / Pandocでスライド生成(前者は未検証)
- ーサーバーに自動で更新
- ーテキストエディタだけでスライドが作れる!

テキストエディタと Gitだけで

どんな環境でも仕事ができる



細かい話: フォーマットの違い

方言がたくさんある

例:

ヘッダの前後に空行が必要か必要でない か、

Ordered Listの記法(1. 2. vs +)、

ヘッダの記法(=== vs #)

改行の処理(そのまま vs 末尾に空白x2)

細かい話: フォーマットの違い

何を重視するかの差

GitHub Flavored MarkdownはGitHubに入るような文章を書きやすくしている(コードの文法指定など)

Day OneのMarkdownは書きやすさを重視 ヘッダの下線や数字での箇条書きは読みや すさを重視

個人的な考え

改行のためにTrailing Whitespace入れたく ない

ヘッダの前後の空行とかも要らない
Ordered Listも要素数増えるときに数字増
やすのが厄介

対策

CommonMark/GitHub Flavored Markdown をベースに書きやすいようサブセットを選択したパーサ/コンバータを作成(予定)



Marko own C 美ししい ドキュメントライフ

ありがとうございま した