

SlerでもSphinxを使いた い！ 総括

2014/10/26 SphinxCon JP 2014

@kk_Ataka

自己紹介

- Twitter: @kk_Ataka
- GitHub: gosyujin
- 普段いるところ
 - Kawasaki.rb
 - Jekyllrb-ja(主催)
- Python力
 - 文法もまだあやしい



発表の趣旨

- **Sler**でも**Sphinx**を使いたいので職場で奮闘してみた備忘録
 - 利用事例
 - 拡張の紹介
- **Kawasaki.rb**というイベントで話した内容の総集編
- できしたこと、できなかったこと含めて駆け足で共有します
- 主觀が多め

今日話さないこと

- **Sphinx**とはなんぞや
- **reST**記法とはなんぞや
- **Sphinx**のインストール方法とか使い方とか

アジェンダ

1. 競合ツールとの比較
2. 導入のためのあれこれ、または導入後の課題
3. 実際に導入してみた感想

競合ツールとの比較

競合ツールとの比較！

- 導入するためには上の人を説得するための政治が必要...
- 競合ツールと比較してよさ気と思ったことを伝えていく
 - 1. Office(Word, Excel)
 - 2. Wiki, Markdown
 - 3. Sphinx
- 好きな言葉 「適材適所」

比較1 Office

Office 長所

- SI界のスタンダード
- WYSIWYGな操作
 - きめ細かいデザインが可能
 - 図やフローの挿入が容易
- 誰のPCにも入っていて、誰でも使える

!?

Office 短所

- あらゆるもののがOfficeで作成され、至る所にちらかる
- 日付バージョン管理...(主観)
- 恐怖の「議事録_20140505_2(最新)(xx修正).xls」
- 検索性が非常に悪い
 - 違うシートとか、吹出しどとか 非表示とか...探せない...

Office 短所

- diffが取るのがメンドくさい
 - 取れないとは言ってない、最近は取れるっぽい
- ミリ単位のレイアウト修正を強いられる
 - リストとかすぐ壊れる
 - 内容を集中して書かせて！
- (職人芸が発揮されるほど)重い

番外



Officeのいいところ...カット！

Officeのいいところ

- Officeでしかできないことも、ある
- 過去資料でいいところあげてます！
- ようは適材適所でよろしくお願ひします

比較2 Wiki, Markdown と
Sphinx

Wiki, Markdown と Sphinx の長所

Officeで短所として挙げた問題は解消できている！...と思う

Wiki, Markdown と Sphinx の長所

- › あらゆるものがOfficeで作成され、至る所にちらかる 問題
 - プレーンテキストで作成される
 - Officeよりは探しやすいんじゃないかと思うのだが...
 - ブラウザ、エディタの検索とか、Wiki内検索とか

Wiki, Markdown と Sphinx の長所

- > diffが取るのがめんどくさい 問題
- Markdownはプレーンテキストなので簡単
 - バージョン管理もしやすい
- Wikiもだいたい差分表示機能あり
- diff取りやすい

Wiki, Markdown と Sphinx の長所

- › ミリ単位のレイアウト修正 問題
- 出力先(html+css、pdfなど)である程度統一できる

Wiki, Markdown と Sphinx の短所

Officeでは特に意識していなかったことを考慮する必要あり

Wiki, Markdown と Sphinx の短所

- 記法を覚える必要がある
 - 未経験者にちょい敷居が高い...
- 「特定部分のみ」のレイアウト修正
 - `css`などに独自の処理を入れなければならない
- 図やフローの挿入はタグで挿入
 - 直感的にいじれない(現物をD&D...)

Wiki, Markdown の短所

- 検索性はあまりよくない
 - しっかり作れば...
 - それでもOffice + 共有サーバコンボよりは...
- 重い
 - 体感としては Office > Wiki, Markdown > Sphinx
 - サーバ性能とか同時アクセス数によるけど

Sphinx だけの長所

- 体系的なドキュメントの骨組みを簡単に整えられる 強力な機能がある
- この辺をサクッとよろしくやってくれているのが `doctree...` である気がする(まだ未調査)
- Wikiとかでこれを整備するのはちょっとしんどい

Sphinx だけの長所

- 検索性はよいと思う
 - 体系的にまとまるため
- 軽い
 - Outputが静的ファイル
 - htmlをWebサーバに置けば静的ファイルを取ってくるのと変わらない

ここまでまとめ

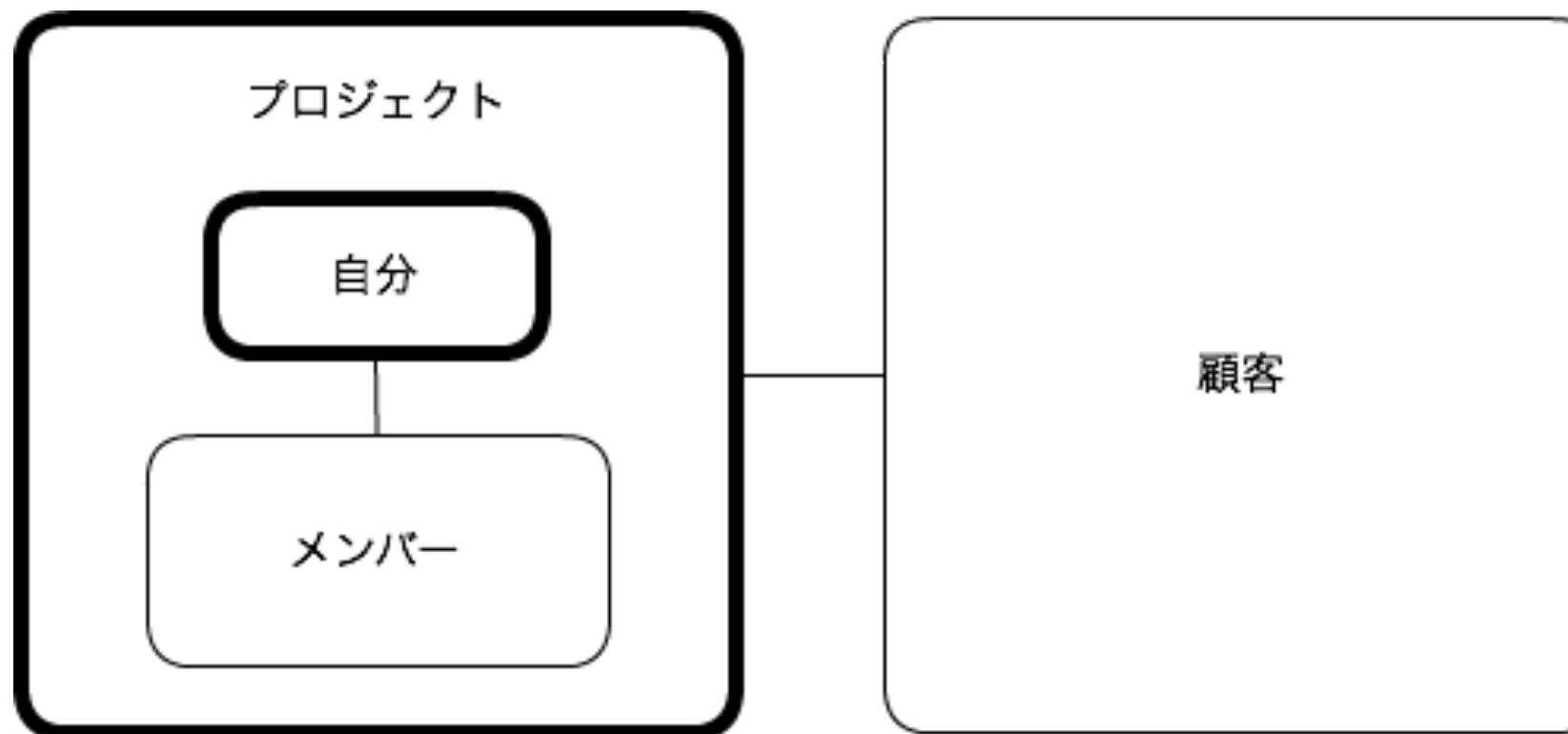
- 慣れ親しんだOfficeから脱却したい、管理しやすい形式でドキュメント作成に挑戦してみよう
- ならば Wiki, Markdown か Sphinx だ！
- TipsとかならWiki, Markdownでもいいけど、ドキュメントなのである程度体系的に管理したい
- 体系的に管理するのが得意な Sphinx だ！

結論: 一回Sphinxチャレンジしてみましょう！

導入のためのあれこれ、
または導入後の課題

導入するための壁

1. 対プロジェクトメンバー(PM)に対して(布教)
2. 対顧客に対して(納品)



壁1. 対PM

対PM 登場人物

- 自分
 - Sphinxを導入したい人、基本的になんでもやる
- メンバー
 - 導入したSphinxを使ってほしい人

対PM 「自分」の仕事

- 今回はreSTで進めることの明確な宣言とサポート
 - 一番大事
 - これができないと負の成果物が生成される...

対PM 「自分」の仕事

- メンバーが「rstファイルを開いてドキュメント作成」に注力できる環境を作る
 - 1. `sphinx-quickstart`で下準備
 - 2. ドキュメント自身のアウトライン作成
 - 3. `doctree`の作成
- などなど

対PM 「自分」の仕事

- ビルド環境、デプロイ環境などもお膳立て
- ビルドはJenkinsなどで拾う
- デプロイはApacheにhtmlファイル配備とか(お手軽)

対PM 「メンバー」 の仕事

- reST記法を覚えてもらう
- Markdown -> reST -> Outputという裏技もある
(Pandoc経由)
- バージョン管理ツールとかの話は...割愛

対PM 課題

- ローカルPCでのプレビュー
 - メンバーのPCにSphinxを入れてもらうのは厳しい...
 - そうすると、確認できるのがサーバにプッシュした時のみになってしまう
 - ローカルで簡単にreSTレビューできないだろうか...
 - GitHubとか使えばある程度できるんだけど

対PM 課題

- プロジェクト(会社)の風土にあわせたカスタマイズが必要かも
- こういうレイアウトがいい
- こういうヘッダフッタがほしい
- こういう改訂履歴ページがほしい

対PM 課題

- 今回は「変更履歴」出力プラグイン作ってみた
 - `sphinxcontrib-releasenotes` プラグイン
- `.. releasenotes::` ディレクティブを追加したところにGitのコミットログと差分を貼り付け
- Git使えない人用

```
.. realeasenotes::  
:lang: ja  
:sur: Sato  
:app: Tanaka  
"source/index.rst" line 45 of 45 --100%-- col 4
```

4c43e09 2014-06-13

ドキュメントのシンタックスワーニング修正

M source/local/test/unit_feature_test.rst

Ataka Sato Tanaka

[show diff]

4b689b2 2014-06-13

20140613打ち合わせ議事録追加

A source/local/minutes/20140613.rst

Ataka Sato Tanaka

[show diff]

4c43e09 2014-06-13

ドキュメントのシンタックスワーニング修正

M source/local/test/unit_feature_test.rst

Ataka Sato Tanaka

[show diff]

```
diff --git a/source/local/test/unit_feature_test.rst b/source/local/test/unit_feature_test.rst
index 31566ab..bed9e2a 100644
--- a/source/local/test/unit_feature_test.rst
+++ b/source/local/test/unit_feature_test.rst
@@ -1,6 +1,6 @@
-----
+
+=====
```

単体・機能テスト仕様書

```
-----
+
.. contents:: 目次
:local:
```

4b689b2 2014-06-13

20140613打ち合わせ議事録追加

A source/local/minutes/20140613.rst

Ataka Sato Tanaka

[show diff]

壁2. 対顧客

対顧客 登場人物

- プロジェクト
 - **Sphinx**でドキュメント納品する側
- 顧客(社内の人 or 社外の人)
 - ドキュメントを納品される側
 - 歴史的経緯から**Office**で納品される事が多い
 - 例外は**Javadoc**とか？

対顧客 「プロジェクト」側の仕事

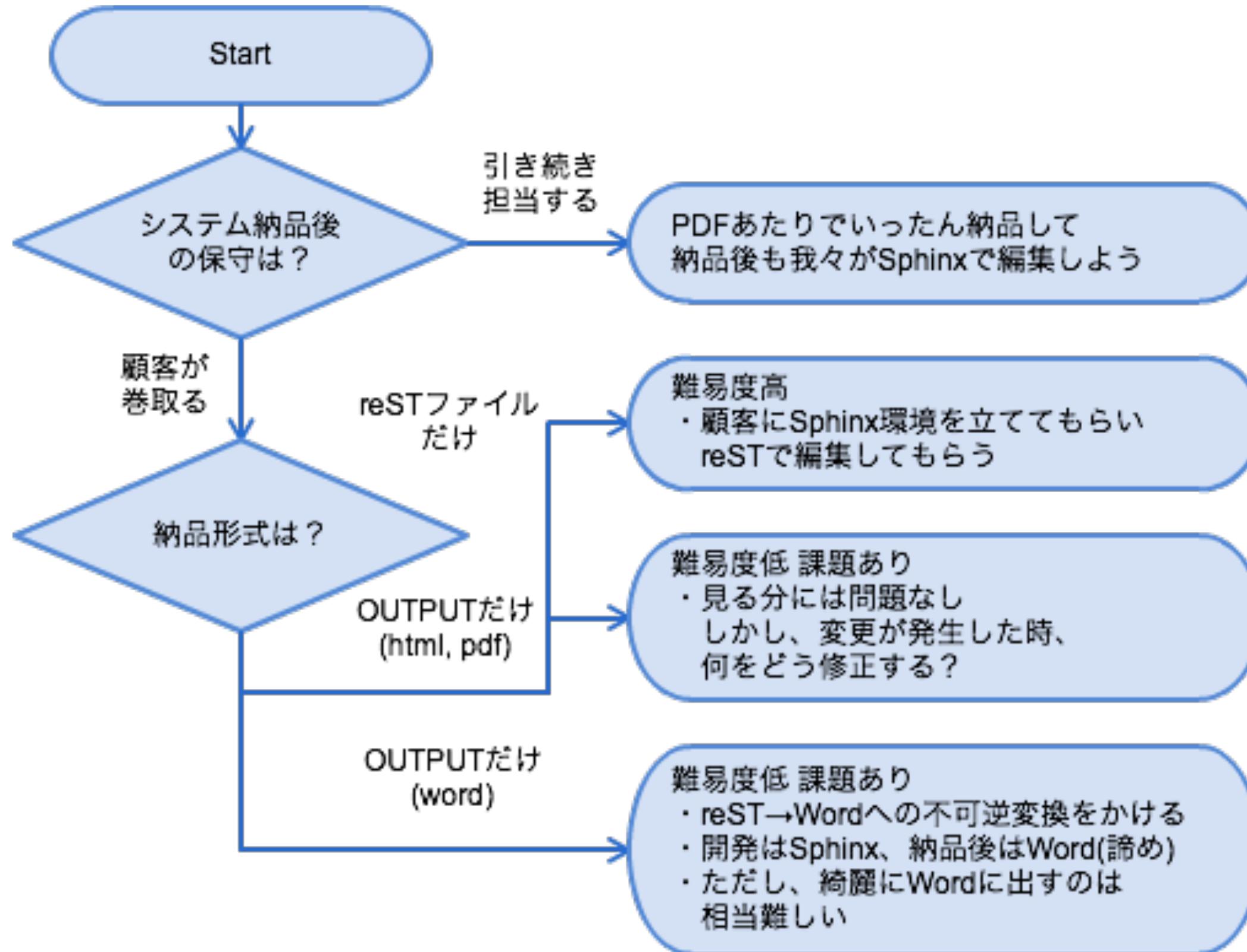
- Sphinxで作成するドキュメントに関して「今回はOfficeじゃない形式で設計書とか書きますよ」の合意を得とく

対顧客 「顧客」 側の仕事

→ 特になし？心構えくらい？

対顧客 課題

- 納品形式 最重要(次ページ参照)
 - 納品後 「誰が」 保守するのか
 - 顧客が巻取ってしまう場合、どう保守すればよいのか
 - 要解決事項、誰かどうやってるか教えてください...
 - これが解決できないと、Sphinxは納品対象外ドキュメントにしか適用できない...



実際に導入してみた感想

環境

- 3ヶ月位のほぼ1人プロジェクトで「社内資料」をSphinx !
- 社内資料なので、納品関係の課題はスルー
- 環境揃えたい放題
 - Git + Sphinx + Jenkins etc
- 途中で1人サポートに入ってもらった

結果

- **Git**でバージョン管理、テキストなので差分管理も簡単！
- エディタが軽いので作成が快適！
- **Output**からお目当ての章が見やすい！探しやすい！
- **Output**は、意外と営業の人を受けが良かった！
 - 「え？ なにこれ？ なんてツール？ あとで教えて」

結論

- 競合ツールとの比較
 - ドキュメントを書くスピードは早いよ！
- 導入のためのあれこれ、または導入後の課題
 - 「**Sphinx**で書いていく！」空気を作るのが難しい
 - 個人的には、「仲間(賛同者)」がいてくれると助かる

結論

- メンバーにreSTを書いてもらうのが難しい
 - メリットの周知、慣例からの脱却までが長い
- 納品物にするにはちょっと課題が多いかも
 - 今回解決の糸口は見つからず...