

AsciiDoc ユーザーマニュアル

K0.1, 2019/08/24

目次

1. 配布パッケージについて.....	1
2. AsciiDocについて	2
3. AsciiDocを使う事にしようと思った背景	3
4. 環境構築手順	5
4.1. Chocolateyをインストール.....	5
4.2. Chocolateyのリポジトリから各種パッケージをインストール.....	6
4.3. AsciiDoc関連ツールをインストール.....	9
4.4. VScodeの拡張機能をインストール	10
5. 使い方.....	11
5.1. VScodeで実際にAsciiDocを書いてみる.....	11
5.2. AsciiDocで書いた文章をGitで管理する.....	14

1. 配布パッケージについて

中身は以下の通り

```
AsciiDocPackage/  
└ template/                                // 文章のテンプレート一式  
  └ csv/                                    // CSVファイルを格納  
  └ fonts/                                  // フォントファイルを格納  
  └ images/                                // イメージファイルを格納  
  └ style/                                  // スタイルファイルを格納  
  └ make_pdf_test.bat                      // AsciiDoc→PDF変換用スクリプト  
  └ test.adoc                              // AsciiDocドキュメントのテストサンプル  
  └ test2.adoc                             // AsciiDocドキュメントのテストサンプル2  
└ tool/                                    // インストール用バッチファイル  
  └ ①ChocolateyInstall.bat  
  └ ②ChocolateyPackageInstall.bat  
  └ ③AsciiDocToolInstall.bat  
  └ ④VScodeExtensionInstall.bat  
└ AsciiDocUserManual.pdf                  // 手順書
```

2. AsciiDocについて

AsciiDocとは？

- Markdownのような軽量マークアップ言語の一つ
- プレーンテキストで体裁が整った文章が書ける
- 可読性が高くMarkdownよりも表現力が高い

3. AsciiDocを使う事にしようと思った背景

従来のOffice(ExcelやWord)での仕様書作成の問題点

- 不適切なバージョン管理
 - 日付ファイル名
 - 命名規則がない、同じ日付で謎のコピー/編集者名前付き
 - 最新であることの確証は無い
 - バックアップ
 - ファイルサーバーのOriginalとローカルPC/別サーバーのBackUp
 - サーバーが死んで取り出せず終了
- 不適切な履歴管理
 - 更新履歴シート
 - 複数の変更点をまとめて修正、ざっくりした履歴コメント、変更の経緯を残しにくい
 - そもそも書かない(誰が編集したのか、追加したのか、不明)
 - 変更箇所は文字色変更
 - 次に修正する人が消さないといけない
 - 文字色が尽きる/見づらい
- 不適切な品質管理
 - レビューなしで保存/発行
 - 担当者のみぞ知る、標準仕様ですら何が変更入っているか知らない
- 本質的ではない作業
 - 罫線、文字色、フォント種別
 - いちいちキーボードからマウスに持ちかえてリボンから選択
 - 選択肢が多くて無駄に選ぶ
 - 図形の整列
 - 図形の差し込みでフォーマットが崩れる、サイズを変えるのも一苦労
 - 目次、章番号
 - デザインの選択からナンバリングまで諸々手動でやることが多い
- 生産性
 - とにかく重い
 - Microsoft Wordは動作を停止しました…
 - ファイルサイズの肥大化で開けない/スクロールできない

仕様書作成をAsciiDocで行うことのメリット

- 不適切なバージョン管理、履歴管理、品質管理
 - Git管理による恩恵を最大限得られる

- 仕様書を一元管理できる
- Originalへは影響を与えず(常にリリース可能な状態に保たれる)ローカルでコミットして編集できる
- 必然とローカル環境にも複製されるので分散開発しやすく障害に強い
- どこを変更したかも全て記録されており、過去の履歴を簡単に参照できる
- AsciiDocがテキストベースなので変更箇所の差分管理も容易
- プルリクエストによりメンバに周知とレビューを兼ねられる

- **本質的ではない作業、生産性**

- **軽量マークアップ言語としてのAsciiDocの利点が最大限発揮される**
 - 全てテキストベースで簡単に作業が行える(かつ難しくない!)
 - AsciiDoc自体の記法が少ないことで良い意味で制限がかかり、編集能力の担当者差が出にくく、かつ編集時間も短縮化
 - シーケンス図などをPlantUMLでテキストベースで書いて埋め込むことも可能
 - 外部ファイルのインクルードもテキストベースで書いて読み込み可能
 - 展開用にHTML化やPDF化なども容易にできる
 - テキストベースなので軽い!

4. 環境構築手順

4.1. Chocolateyをインストール

以下のバッチファイルをダブルクリックで実行する

①ChocolateyInstall.bat

- ユーザーアカウント制御の許可のポップアップが出るので **はい** をクリック
- コマンドプロンプトが表示されて処理が進むので自動的に閉じたら完了

4.2. Chocolateyのリポジトリから各種パッケージをインストール

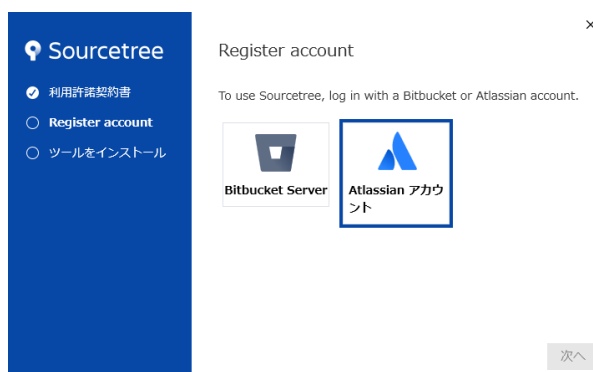
以下のバッチファイルをダブルクリックで実行する

②ChocolateyPackageInstall.bat

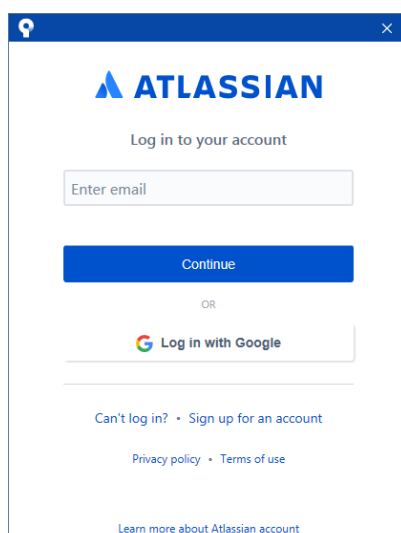
- ユーザーアカウント制御の許可のポップアップが出るので **はい** をクリック
- コマンドプロンプトが表示されて処理が進むのでしばらく待つ
- 下記画面が表示されたら **ライセンスに同意します** にチェックを入れて **次へ** をクリック



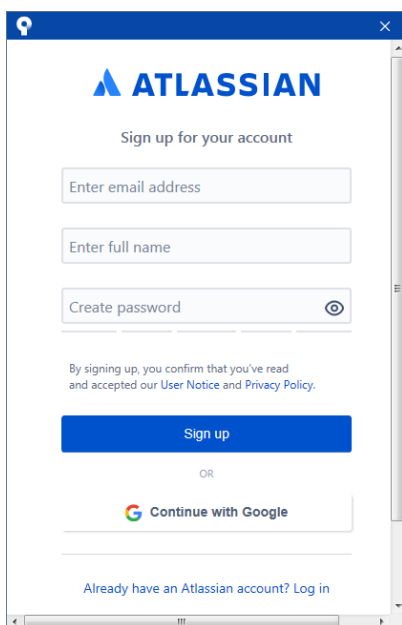
- **Atlassianアカウント** を選択して **次へ** をクリック



- **Enter email** にメールアドレスを入力して **Continue** をクリック



- 続けて、**Enter full name** にニックネーム、**Create password** にパスワードを求められるので入力して、**Sign up** をクリック（※既にアカウントを持っている場合は通常のサインインを行う）

A screenshot of the Atlassian sign-up form. The form is titled "Sign up for your account" and includes three input fields: "Enter email address", "Enter full name", and "Create password" with a toggle for password visibility. Below the fields is a checkbox for terms and conditions, followed by a blue "Sign up" button. An "OR" separator is present, followed by a "Continue with Google" button. At the bottom, there is a link for "Already have an Atlassian account? Log in".

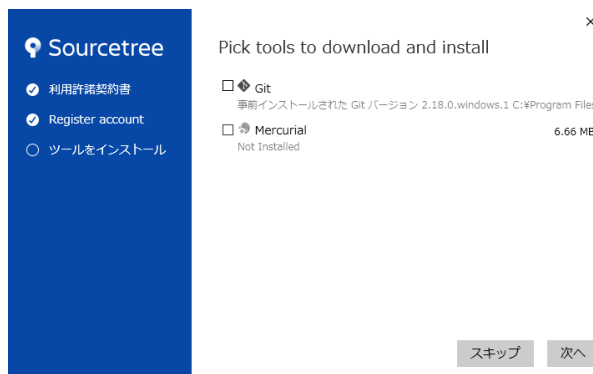
- reCAPTCHAの画像認証の指示に従って選択を行い、**確認** をクリック



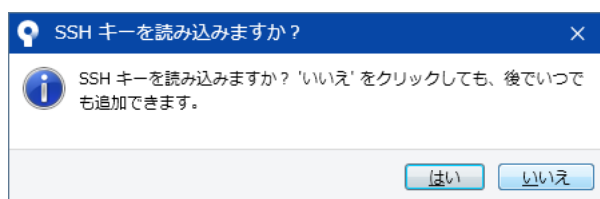
- 認証に成功すれば登録完了画面に遷移するので、**次へ** をクリック



- ツールのインストール画面に遷移するので **Git** にだけチェックを入れて、**次へ** をクリック (※既にGitをインストール済みの場合はそのまま **次へ** をクリック)



- **SSHキーを読み込みますか？** が表示されたら **いいえ** をクリック



- Sourcetreeが自動的に立ち上がったのが確認できたらアプリを一旦閉じる



- コマンドプロンプトの画面内にて、Atlassianアカウントを作成完了したか聞かれるので、完了していたら **y**、完了していなければ **n** を入力して **Enter** を押す
y の場合：コマンドプロンプトが表示されて処理が進むので自動的に閉じたら完了
n の場合：再度、**Enter** を押すとコマンドプロンプトが閉じる

4.3. AsciiDoc関連ツールをインストール

以下のバッチファイルをダブルクリックで実行する

③AsciiDocToolInstall.bat

- コマンドプロンプトが表示されて処理が進むので自動的に閉じたら完了

4.4. VScodeの拡張機能をインストール

以下のバッチファイルをダブルクリックで実行する

```
④VScodeExtensionInstall.bat
```

- コマンドプロンプトが表示されて処理が進むので自動的に閉じたら完了

5. 使い方

5.1. VScodeで実際にAsciiDocを書いてみる

ここでは、テストサンプルでプレビューやPDFへの変換を行い、正しく環境構築ができたことを確認します
また、テストサンプルの内容はAsciiDocの文法紹介も兼ねているので参考にしてください

5.1.1. 作業ディレクトリを作成する

配布パッケージ内のtemplateフォルダー一式をローカルPCの任意の場所にコピーして使います
このフォルダー一式が文章のテンプレートとなります

5.1.2. VScode を起動する

AsciiDocで書くためのテキストエディタとして使用します
Windowsのスタートメニューから **Visual Studio Code** (以下、VScodeとする)を検索して起動します

5.1.3. テストサンプルを開く

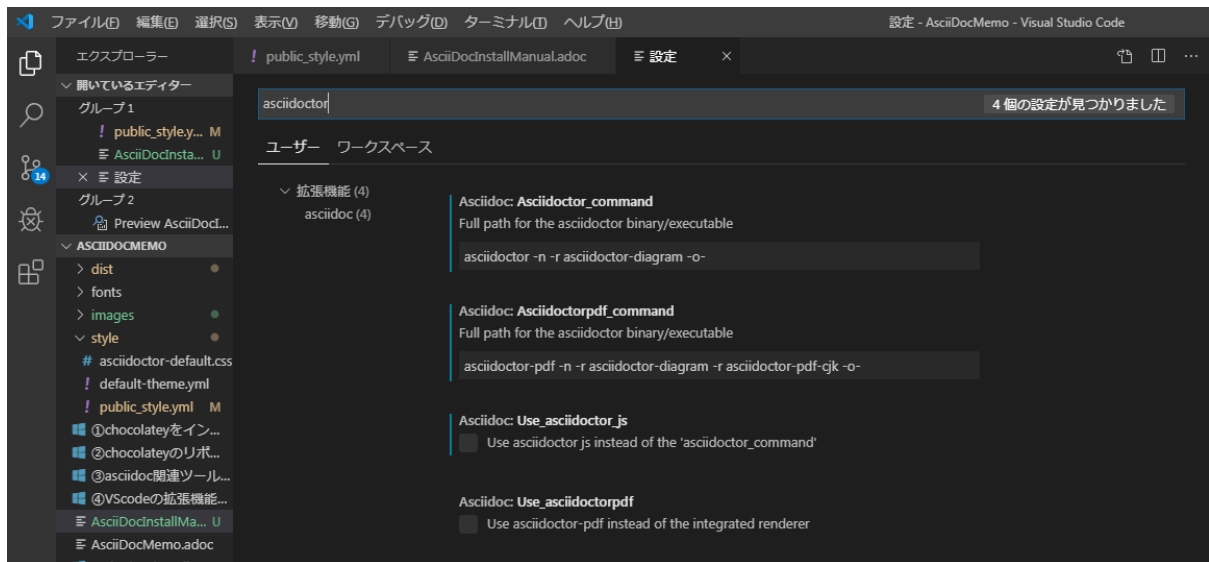
[ファイル]→[ファイルを開く]から **template** フォルダ内の **test.adoc** ファイルを開きます

5.1.4. テストサンプルをプレビューする

asciidoctorの設定を変更する

VScode上でプレビュー表示を行うための設定を行います
[ファイル]→[基本設定]→[設定]から **asciidoctor** を検索し、以下の設定を行います

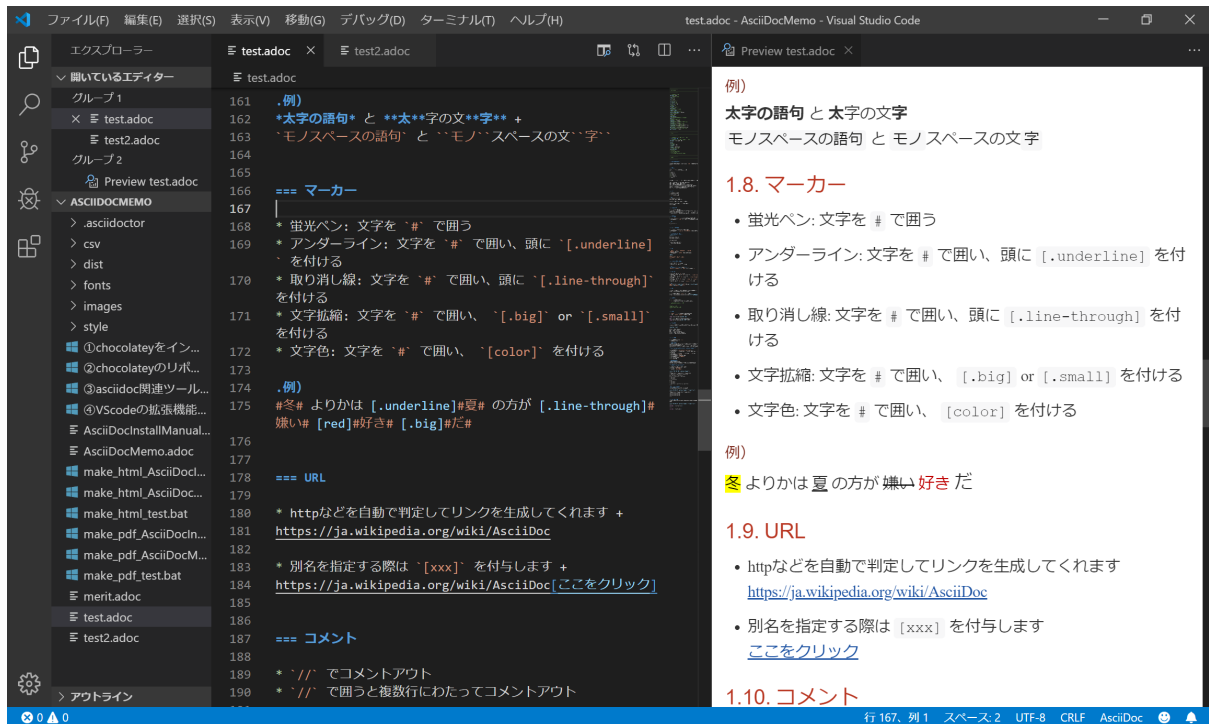
```
asciidoctor_command : asciidoctor -n -r asciidoctor-diagram -o-  
asciidoctorpdf_command : asciidoctor-pdf -n -r asciidoctor-diagram -r  
asciidoctor-pdf-cjk -o-  
use_asciidoctor_js : false(チェックを外す)
```



プレビューを行う

ショートカット **Ctrl+K → V** で画面右側にプレビューが表示されます

参考までにテストサンプルのプレビュー結果を以下に示します



5.1.5. テストサンプルをPDFに変換する

以下のバッチファイルをダブルクリックで実行する

```
make_pdf_test.bat
```

- コマンドプロンプトが表示されて処理が進むので自動的に閉じたら完了
- **test.pdf** が作成されます



必要に応じてバッチファイル内のファイル名を修正して使ってください
-o 変換後ファイル名.pdf 変換前ファイル名.adoc

5.2. AsciiDocで書いた文章をGitで管理する

以上で終わりです!