githubを準備し，コードをアップするまでの手順

作成日：2019/01/30

0.目次

1. 最初にやること

- はじめてgithubに登録をしたとき.
- 新しいPCでgithubにアクセスするとき.

1.1 ホームディレクトリに移動する

ターミナルを起動するとホームディレクトリにいるはずだが一応下記を実行.

```
$ cd ~
```

1.2 ssh-keyの発行

- githubにアクセスするにはssh-keyが必要となる.
- ssh-keyは公開鍵と秘密鍵のペアになっており, 「公開鍵をgithubに登録」「秘密鍵をPCのローカルに保存」することで通信時に組み合わせがあっているかどうか判断する.
- ターミナル(windowsはgit bash)を開いて以下のコマンドを入力する.

```
$ ssh-keygen
```

実行結果

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/vagrant/.ssh/id_rsa):
```

上の後で`github_rsa`を入力してEnter.

```
Enter passphrase (empty for no passphrase):
```

続き. とりあえず入力せずにEnter.

```
Enter same passphrase again:
```

続き. パスワード入力していないので再度そのままEnter.

```
Your identification has been saved in github_rsa.
Your public key has been saved in github_rsa.pub.
The key fingerprint is:
6f:09:00:22:44:55:66:77:95:89:41:7d:a7:58:1b:92
vagrant@localhost.localdomain
The key's randomart image is:
```

```
+--[ RSA 2048]----+
|   .o. .         |
|     oEo+ .      |
|    . +=+=       |
|     o.+==       |
|    . . S .      |
|       - + .     |
|      .   +      |
|         .       |
|                 |
+-----------------+
```

これでssh-keyを発行できた．発行した内容を確認する．下記のようになっていればOK．．

```
$ ls -la | grep github
```

実行結果.

```
-rw-------  1 vagrant vagrant 1675 11月 20 12:18 2014 github_rsa
-rw-r--r--  1 vagrant vagrant  411 11月 20 12:18 2014 github_rsa.pub
```

1.3 ssh-keyの配置

- ssh-keyは適切な場所に配置しないと動かない.
- 以下のコマンドで動作する場所に配置する.

下記コマンドでフォルダの有無を確認する.

```
$ ls -a
```

`.ssh/`**が存在しない場合のみ**下記コマンドでホームディレクトリに`.ssh`を作成する.

```
$ mkdir -p ~/.ssh
```

ここから共通．続けて以下を実行する．上で作成したディレクトリに作成したキーペアを移動する.

```
$ mv github_rsa* ~/.ssh
```

エラーが出なければOK．引き続き，以下のコマンドでssh-keyを動作するようにする.

```
$ eval $(ssh-agent)
Agent pid 9899
```

作成したキーペアをssh-agentに登録する.

```
$ ssh-add ~/.ssh/github_rsa
Identity added: /home/vagrant/.ssh/github_rsa
```

以上でssh-keyの準備は完了だが，設定ファイルに変更を加える必要がある.

1.4 設定ファイルの書き込み

- 下記のコマンドで設定ファイルに書き込む.
- 2行目の「vi ~....」ではターミナル内でエディタを起動する．画面が変わりますがパニックにならないよう注意！
- このエディタにはインサートモードとコマンドモードの2つがあり，最初はコマンドモードで表示されるが，追記を行うにはインサートモードに変更する必要がある.
- インサートモードにするには「i」キーを押す．画面のどこかに－INSERT－や－挿入－などの文字列が表示される．その状態で「Host github」からの5行を追記する.
- 追記が終わったら「esc(コマンドモードに戻る)」→「:wq(保存して終了のコマンド)」→「enter(実行)」で元の画面に戻る.
- ミスったと感じたら「esc」→「:q!」→「enter」でエディタを終了できるので，もう一度「vi ~.....」を入力してやり直しましょう.

ターミナルで以下を1行ずつ実行.

```
$ touch ~/.ssh/config
$ vi ~/.ssh/config
```

`.ssh/config`ファイルを開くと空なので以下を追記する.

```
+ Host github
+       HostName github.com
+       Identityfile ~/.ssh/github_rsa
+       Port 22
+       User git
```

続いて，設定ファイルの権限を変更する．ターミナルで以下を実行.

```
$ chmod 700 ~/.ssh/config
```

エラーが出なければOK.

1.5 githubへのssh-key登録

githubのサイトにアクセスし，「設定」→「SSH keys」へ進む．「Add SSH key」をクリックして入力画面へ進む．

ターミナルで下記のコマンドを入力し，ssh-keyを表示させる．

```
$ cat ~/.ssh/github_rsa.pub
```

実行結果

```
ssh-rsa ...
...
...
...
...
... localhost@0-mac
```

上のように文字列が表示されたら，「ssh-rsa」から全てコピーし，githubサイトの入力欄に貼り付ける．タイトルはPC名など適当につけてOK．入力したら「Add key」をクリックして終了．

1.6 初期設定

githubのユーザ名とメールアドレスを登録する．ターミナルで下記コマンドを入力し，エンターを押す．(それぞれ自身のアカウントのものを入力する)

```
$ git config --global user.name "taroosg"
$ git config --global user.email "taro.osg@gsmail.com"
```

続いて以下を入力して内容を確認する．

```
$ git config -l
user.name=TaroOhsugi
user.email=taro.Ohsugi@gsmail.com
```

上で入力した内容に間違いなければOK．エラーが出る場合は1.8参照．

1.7 githubとの接続テスト(ターミナル)

ターミナルで書きを実行

```
$ ssh -i ~/.ssh/github_rsa git@github.com
```

実行結果

```
Hi username! You've successfully authenticated, but GitHub does not
provide shell access.
Connection to github.com closed.
```

上記のように表示されればOK.

1.8 「xcrun: error」が発生する場合(macのみ)

この場合，gitコマンドを実行する「Command Line Tools(macOS High Sierra version 10.13) for XCode」がインストールされていないことが原因.
ウインドウが出てきたら指示に従ってインストールし，完了後に再度コマンドを実行すればOK. ウインドウが出てこない場合は以下のコマンドを実行すればインストールできる.

```
$ xcode-select --install
```

2. プロダクトを登録するときにやること

- githubにアップしていない新しいプロダクトを作成したとき.

2.1 リポジトリを作成する

- ブラウザでgithubにアクセスし，新しいリポジトリを作成する.
- 名前はプロダクト名にしておくとわかりやすくて良い．(janken, mamopadなど)
- 作成するとurlが表示されるのでそのままにしておく．SSHのタブを選択しておく(後でurlを使用するため)

2.2 ディレクトリを変更する

ターミナルで下記コマンドを入力する．(enterは押さないこと)
【重要】cdのあとにスペースを入れること

```
$ cd
```

入力したらプロダクトのフォルダをターミナルのウインドウ内にドラッグ&ドロップする． すると，「cd 」の後にパスが表示されるので，間違いなければ「enter」を押す.

2.3 初期化

下記コマンドを入力する.

```
$ git init
```

2.4 リポジトリの登録

コードをアップするリポジトリ(宛先)を登録する．先程ブラウザで作成したリポジトリのurl（`git@github.com/***`）を使用する.

ターミナルで以下を実行する.

```
$ git remote add origin リポジトリのurl
```

登録されているかどうかは以下のコマンドで確認できる.

```
$ git remote -v
```

確認結果

```

```

```
origin  git@github.com:******/******.git (fetch)
origin  git@github.com:******/******.git (push)
```

間違っている場合は以下で登録し直す.

```
$ git remote set-url origin 正しいurl
```

2.5 ファイルをadd

githubにアップするファイルをaddする. アップするには「add」「commit」「push」の3手順が必要.

ターミナルで以下を実行する. addの後にはスペースが入る点に注意！

```
$ git add .
```

2.6 ファイルをcommit

commitはファイルのバージョンを作成するイメージ. コミットを重ねても, 以前のコミットへ状態を戻すことでファイルの内容をもとに戻すことが可能. 以下のコマンドを入力する.

-mのあとの「""」内に適宜コメントを追加する. (変更内容など)

```
$ git commit -m"ここにコメントを書く"
```

2.7 ファイルをpush

pushは実際にファイルをアップロードするイメージ. この段階ではじめてgithubにコードが追加される. 下記コマンドを入力する.

```
$ git push origin master
```

実行結果

```
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 385 bytes | 385.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:******/******.git
   9ae72be..eb700ec  master -> master
```

ブラウザでgithubのページを確認し，ファイルがアップされていれば成功！

3. 登録済みのプロダクトを変更したときにやること

- すでにリポジトリに登録してあるファイルに追記，編集などしたとき

3.1 【超重要】ディレクトリを変更する

- これを忘れると別のリポジトリに登録することになりものすごく面倒なので注意すること！！
- ターミナルで下記コマンドを入力する．cdのあとには必ずスペース入れる．(enterは押さないこと)

```
$ cd
```

入力したらプロダクトのフォルダをドラッグ&ドロップする． すると，「cd」の後にパスが表示されるので，間違いなければ「enter」を押す．

以降は上記「2.5」「2.6」「2.7」と同様．

3.2 ファイルをadd

```
$ git add .
```

3.3 ファイルをcommit

```
$ git commit -m"ここにコメントを書く"
```

3.4 ファイルをpush

```
$ git push origin master
```

実行結果

```
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 385 bytes | 385.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:******/******.git
   9ae72be..eb700ec  master -> master
```

ブラウザでgithubのリポジトリを確認して登録されていればOK！