1. git flow でのソースコード管理

git-flow は、gitの開発手法で、それをまとめたツール。blanch の役割とブランチの管理について取り決めている。

基本的には、develop ブランチがローカルでの開発用ブランチ。feature ブランチを作成し、自身のソースコードを追加。feature ブランチを develop ブランチにマージ。というようなブランチの使い方を行う。

master ブランチへの develop ブランチのマージは、release コマンドによりマージを実施する。

hotfix ブランチは、master のバグ修正を行うブランチ。hotfix コマンドを使うと、自身の develop ブランチが master ブランチと同じ状態になり、feature ブランチを切って作業する。完了したら、hotfix ブランチを master ブランチにマージする。

1.1. 参考サイト

git-flow cheatsheet

2. git-flow 使い方

2.1. 使い始め

```
git clone
git init
```

のようなgitローカルリポジトリを作成した後、git-flow として初期化する。初期化中に、ブランチの名前とリリース版の版数のプレフィックスを設定する。

```
git flow init
```

2.2. ソースコードの作成

future ブランチで作業する。feature fisnish とすると develop ブランチにマージされる。

```
git flow feature start [feature_branch_name]
git flow feature finish [feature_branch_name]
```

同じ、develop ブランチで開発しているメンバーに対して、開発分をリモートへアップしたり、リモートから最新の修正分を取り込みながら、開発を進めていく場合は、feature publish 、 feature pull を実施しながら開発を進める。

```
git flow feature publish [feature_branch_name]
git flow feature pull [feature_branch_name]
```

2.3. リリース準備開始

```
git flow release start [release_branch_name]
git flow release finish [release_branch_name]
```

リリースブランチを作成後に修正をする場合、以下を実施。

```
git flow release publish [release_branch_name]
git flow release pull [release_branch_name]
```

2.4. 緊急時の対応

```
git flow hotfix start [version]
git flow hotfix finish [version]
```

3. EoF

EoF