YAMAPアプリのCI環境のおはなし (iOS版)

2019-12-11

YAMAP 藤木

YAMAP

ざっくりとしたアジェンダ

自己紹介

弊社iOSアプリチームの...

- ビルド・テスト・リリース環境紹介
- 開発フロー紹介
- Circle CIの使い方紹介

自己紹介

名前: 藤木

YAMAP社員歴: 半年くらい

登山歴: 半年くらい

主な趣味: スキューバダイビング (4年ちょっと)

CI歴: Jenkinsおじさんを数年、Circle CIはYAMAP入ってから

ビルド・テスト・リリース環境

YAMAP

ビルド・テスト・リリース環境

タスク管理: **Backlog**

レポジトリ: **GitHub**

ビルドツール: **fastlane**

ビルド環境: **CircleCI**

リリース・テスト: **TestFlight**

ビルド・テスト・リリース環境 (2)

以下のようなコマンドで、ビルド・自動テスト・TestFlightへの配信を行う。
これは個々の開発者のマシン上で実行できる。CIではこれを決められたタイミングでパラメーターを変えながら実行するだけ。

```
$ bundle exec fastlane scheme:"prod" \
    test_and_upload skip_test:false skip_upload:false
```

審査提出は以下のコマンドで実行。これは手動で行う。

```
$ bundle exec fastlane submit build_number:x.y.z1.z2
```

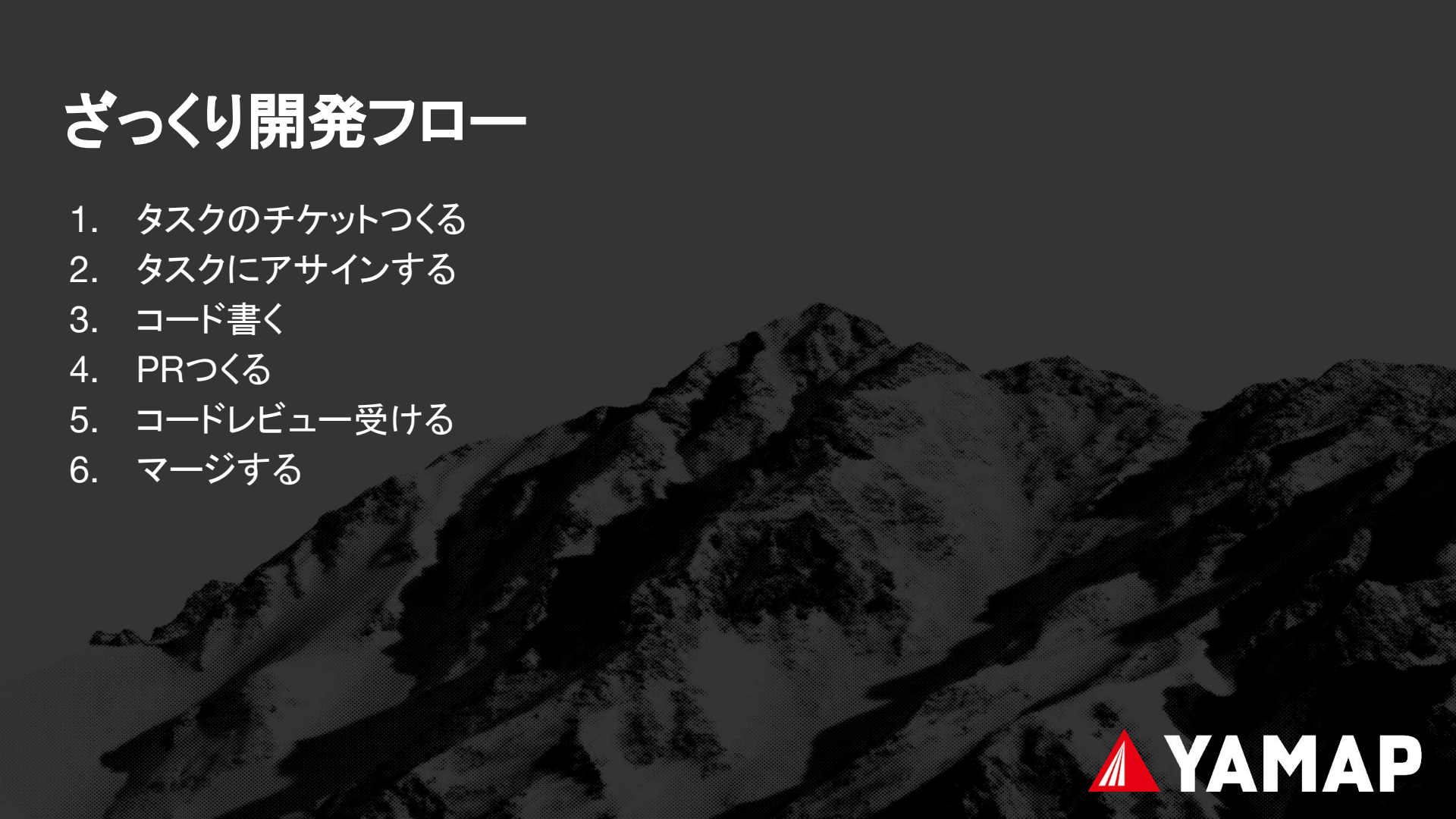
YAMAP

開発フロー

YAMAP

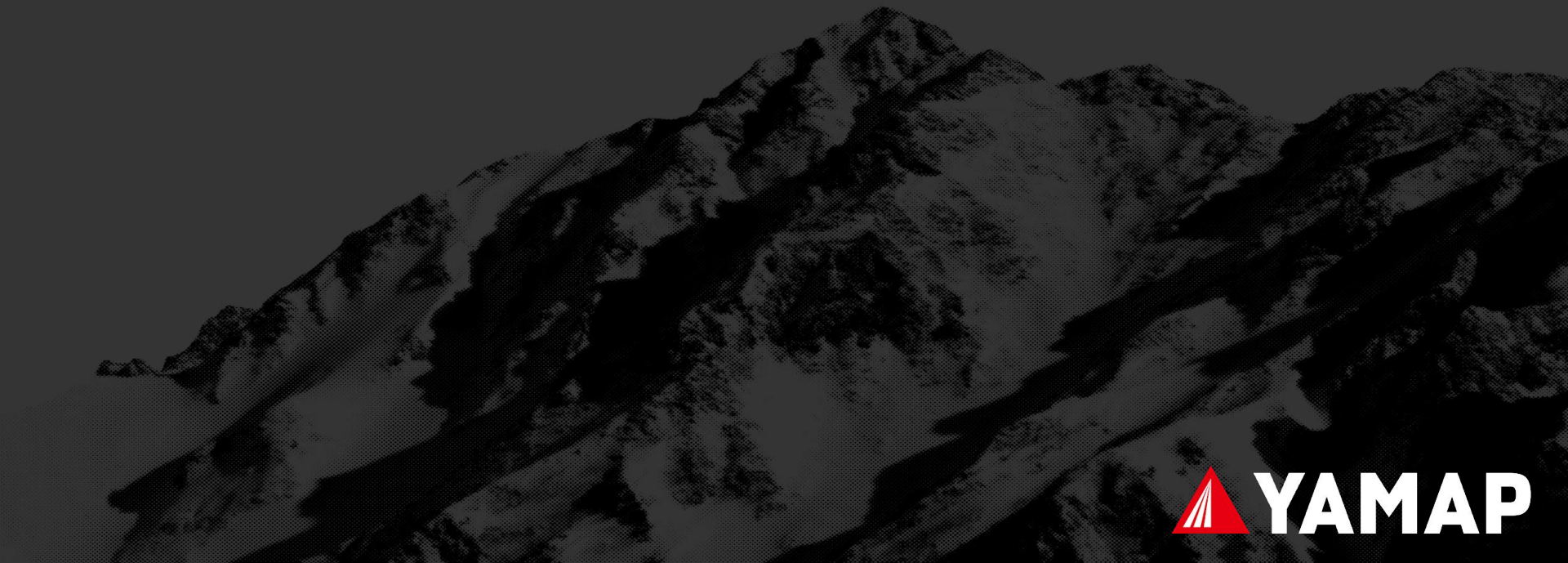
ざっくり開発フロー

1. タスクのチケットつくる
2. タスクにアサインする
3. コード書く
4. PRつくる
5. コードレビュー受ける
6. マージする

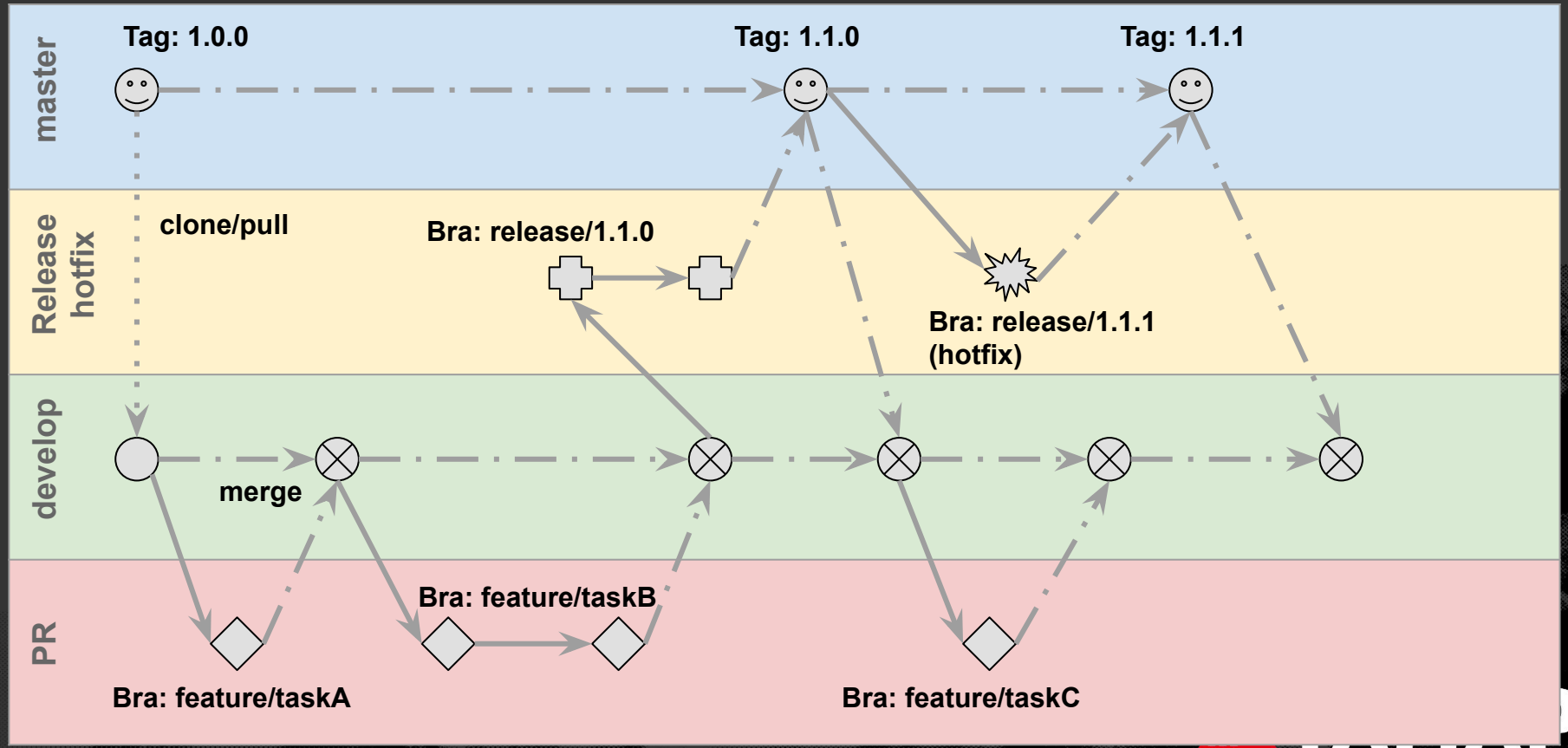


iOSアプリチームのGitブランチ運用

Git-Flow + GitHub Flowがベース。



iOSアプリチームのGitブランチ運用




CI
YAMAP

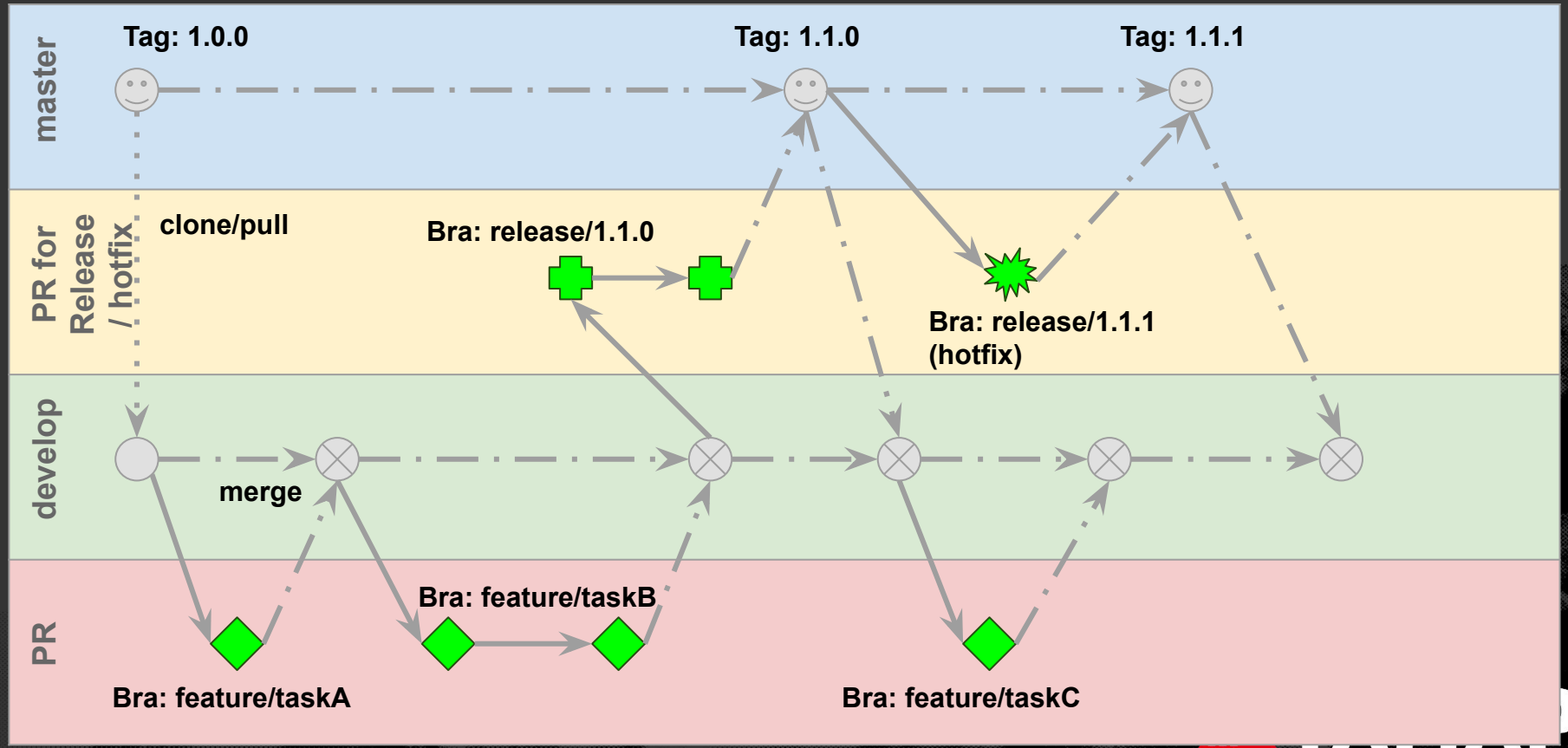
こころがけてること

個々の開発者のマシン上で実行できる(している)ものを
CIで動かす

=> リリースビルドのビルド手順の変更・デバッグが容易になる
=> 必要以上にCI設定が複雑にならない
=> Circle CIで障害起きたり、何らかの形でCIが死んでも、人力でリリースが可能になる

YAMAP

CIのビルドがはしるところ



ビルド戦略 (一部)

Workflow	ブランチ	スキーマ	自動テスト	TestFlightに配信	バージョン書式
PR	feature/*	prod	ON	OFF	x.y.z
	feature-testflight/*	prod	ON	ON	998.$PR_NUM
Release Hotfix	release/*	dev	ON	ON	x.y.z
		prod	ON	ON	x.y.z

YAMAP

ビルド戦略ちょっとほそく

どの場合も、PRを作成した時にビルドを走らせる。

基本的にブランチ名でどの戦略を使うか(行を選択するか)を決める。

PRのブランチからビルドしたやつと、Releaseブランチからビルドしたやつはマーケティングバージョン(CFBundleShortVersionString)を変える。

YAMAP


.circleci/config.yml
YAMAP

Jobの仕様説明 (擬似コード)

```
# ビルドしてテストしてTestFlightに上げる便利なjob
# パラメーターで動作を切り替えられる
test_and_upload(
    scheme="prod",              # ビルドコンフィグ
    skip_test=false,            # テスト実行するか否か
    skip_upload=true,           # TestFlightに上げるか否か
    marketing_version=null,     # CFBundleShortVersionString 上書きする場合
    build_number=null,          # CFBundleVersion 上書きしたい場合
)
```

YAMAP

Jobのパラメタライズの実現

宣言

```
jobs:
  test_and_upload:
    macos:
      xcode: "11.1.0"

    parameters:
      scheme:
        type: enum
        default: "prod"
        enum: ["dev", "prod"]

      skip_upload:
        type: boolean
        default: true
...
```

呼び出し

```
workflows:
  yamap_release:

    jobs:
      - test_and_upload:
          name: "YAMAP Release"
          scheme: "prod"
          skip_upload: false
          filters:
            branches:
              only:
                - /^release\/\d+\.\d+\.\d+/
```

YAMAP

Jobのパラメタライズ (2)

API 2.1 から追加されたらしい **parameters** key を使う。
いろんなところで使える。詳しくは Circle CI のドキュメントを参照。

- **Reusing Config - Authoring Parameterized Jobs**
  https://circleci.com/docs/2.0/reusing-config/#authoring-parameterized-jobs

DRYの実現 (before)

```
workflows:
  yamap_release:
    # releaseブランチでは
    # prod / dev 両方ビルド
    jobs:
      - test_and_upload:
          name: "YAMAP Release"
          scheme: "yamap"
          skip_upload: false
          filters:
            branches:
              only:
                -
/^release\/\d+\.\d+\.\d+/
```

```
      - test_and_upload:
          name: "YAMAP (develop)
Release"
          scheme: "yamap (develop)"
          skip_upload: false
          skip_test: true
          filters:
            branches:
              only:
                -
/^release\/\d+\.\d+\.\d+/
```

YAMAP

DRYの実現 (after)

```
workflows:
  yamap_release:
    jobs:
      - test_and_upload:
&YAMAP_RELEASE_COMMON
          name: "YAMAP Release"
          scheme: "yamap"
          skip_upload: false
          filters:
            branches:
              only:
                -
/^release\/\d+\.\d+\.\d+/
```

```
      - test_and_upload:
          <<: *YAMAP_RELEASE_COMMON
          name: "YAMAP (develop)
Release"
          scheme: "yamap (develop)"
          skip_test: true
```

YAMAP

DRYの実現 (2)

config.yml 上では定義したオブジェクトの再利用(Anchors/Aliases YAMLの仕様)や、
オブジェクト同士のマージが可能なのでそれを使う。

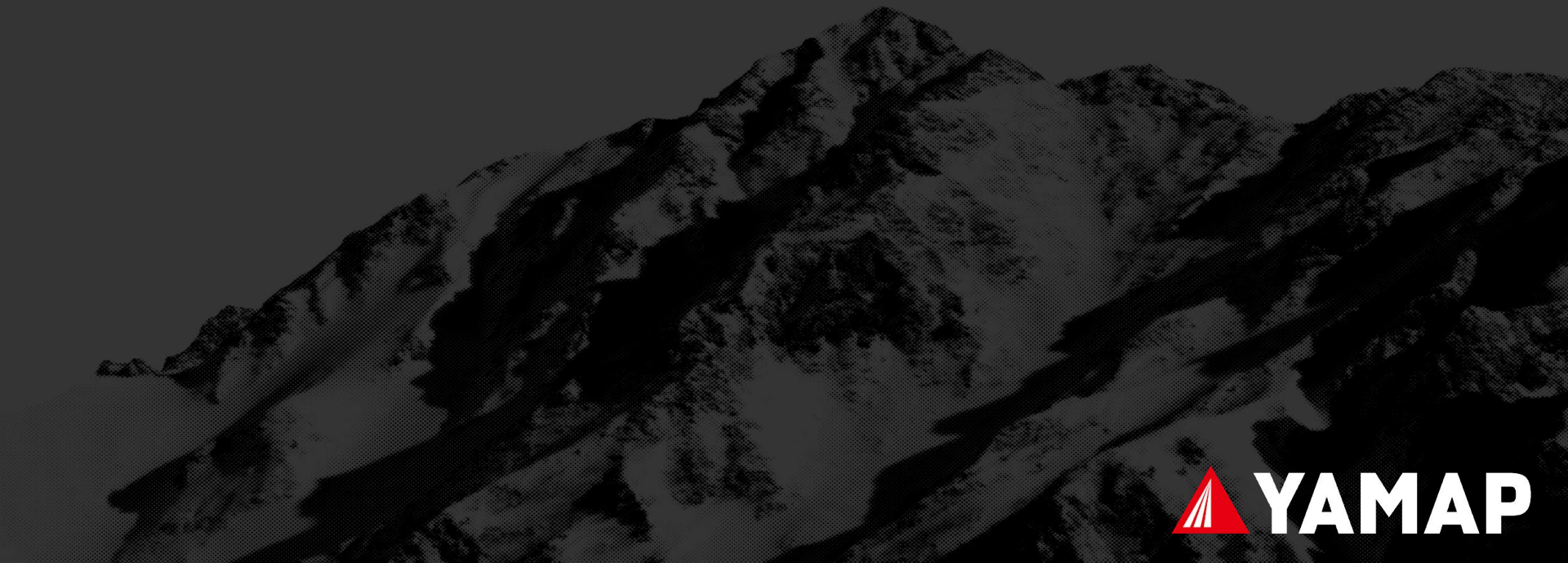
- **Writing YAML - Anchors and Aliases**
  https://circleci.com/docs/2.0/writing-yaml/#anchors-and-aliases

- **Writing YAML - Merging Maps**
  https://circleci.com/docs/2.0/writing-yaml/#merging-maps

YAMAP

.circleci/config.yml 公開

https://tkysfjk.github.io/files/2019-12-11-circleci-meetup-fukuoka-3/config.yaml



今のつらいところ

YAMAP

Linux Plan | macOS Plan

Billing Period	Credits Used	Usage		
Nov 24 - Dec 24	218377		87%	4,365/5,000 minutes
Oct 24 - Nov 24	220424		88%	4,406/5,000 minutes
Sep 24 - Oct 24	264165		106%	5,279/5,000 minutes
Aug 24 - Sep 24	298622		119%	5,968/5,000 minutes
Jul 24 - Aug 24	170569		68%	3,409/5,000 minutes
Jun 24 - Jul 24	131484		53%	2,628/5,000 minutes
May 24 - Jun 24	161780		65%	3,233/5,000 minutes
Apr 24 - May 24	69795		28%	1,395/5,000 minutes
Mar 24 - Apr 24	59379		24%	1,186/5,000 minutes
Feb 24 - Mar 24	44548		18%	890/5,000 minutes
Jan 24 - Feb 24	60287		24%	1,203/5,000 minutes
Dec 24 - Jan 24	28923		12%	578/5,000 minutes

めっちゃ時間くう

ビルド・テストに約10分

TestFlightへのアップロードに約30分 (CPUはほぼアイドリング)

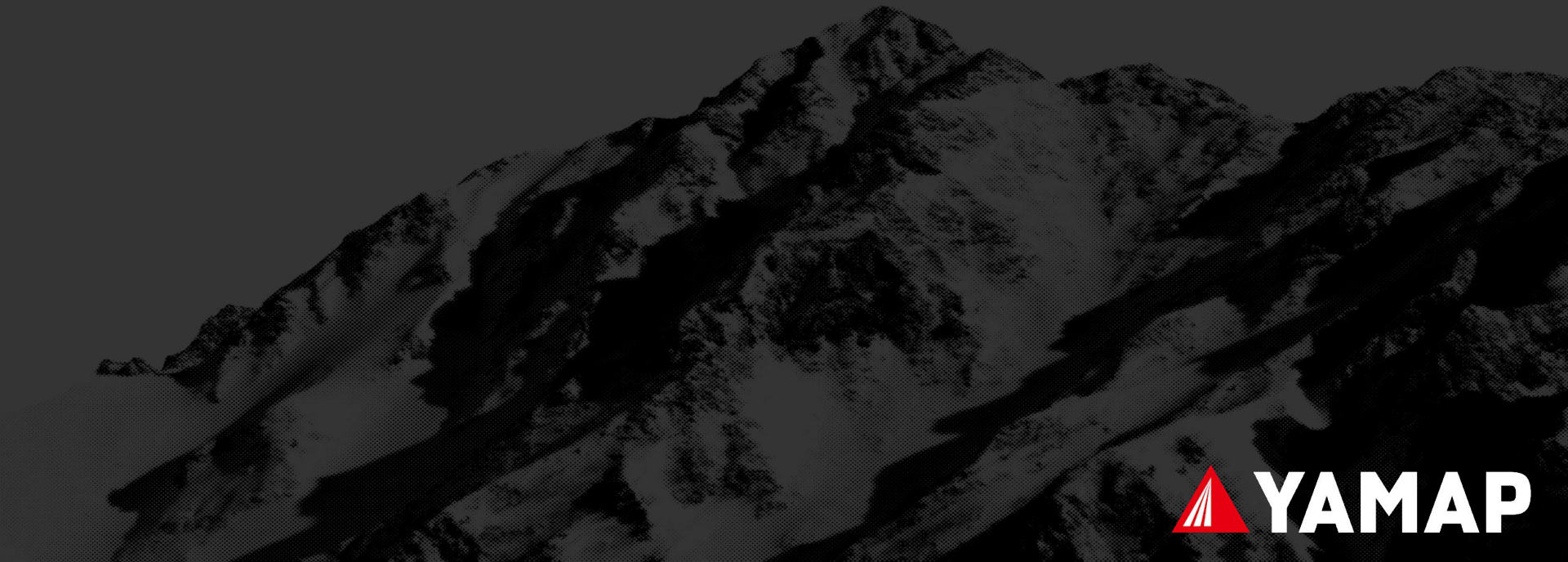
TestFlightアプリからダウンロード可能になるまでに1時間から1日以上

ただ、TestFlightの使い勝手が良いので我慢してる状態。(バイナリ配布のやりやすさ、ユーザー管理のやりやすさ、審査提出のやりやすさなど...)

YAMAP

Circle CI これがほしい

CPU Usageに基づく割引 (ほぼほぼアイドル中は課金対象じゃないとか)



YAMAP


YAMAP

```
▸ Compiling ...
❌  fatal error: error in backend: Unexpected end of file
❌  clang: error: clang frontend command failed with exit
code 70 (use -v to see invocation)
```

YAMAP

```
Downloading CocoaPods master repo from cocoapods-specs S3
bucket...
Uncompressing CocoaPods master repo...
mas
ter/.git/objects/pack/pack-14ae493bfd83338a3ccfb96bfbd84523a
b805ed7.pack: gzip decompression failed
tar: Error exit delayed from previous errors.
Download from S3 failed, cleaning up and falling back to
standard checkout...
Exited with code 1
CircleCI received exit code 1
```

YAMAP

実行タイミングによってはエラーが出て止まる

- 負荷が高いのかファイルが読めなくてビルドが止まる(exit code 70)
- 公式 S3 から取ってきたCocoapodsのレポジトリが腐ってて止まる

=> **いまのところワークフロー再実行しかない**


おしまい
YAMAP