## CodeMagic journey

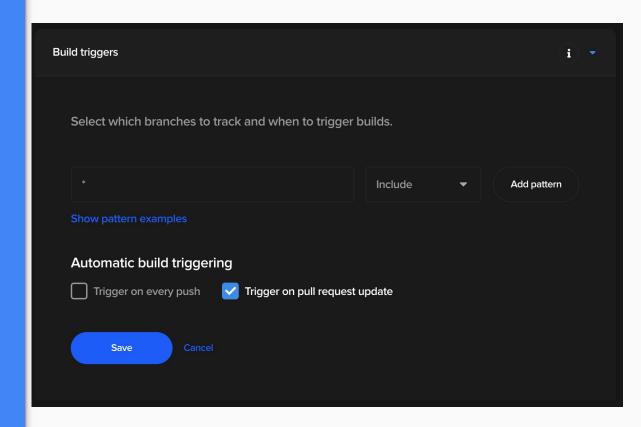
Suesi Tran

## DISCLAIMER

- Using GitHub account
- No setup, no Flutter starter guide

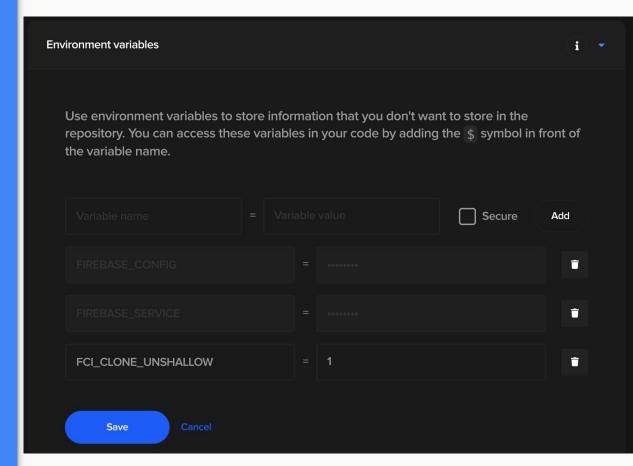
## Build trigger

- Default trigger setting
- NOTE: automatic build trigger will only run TEST



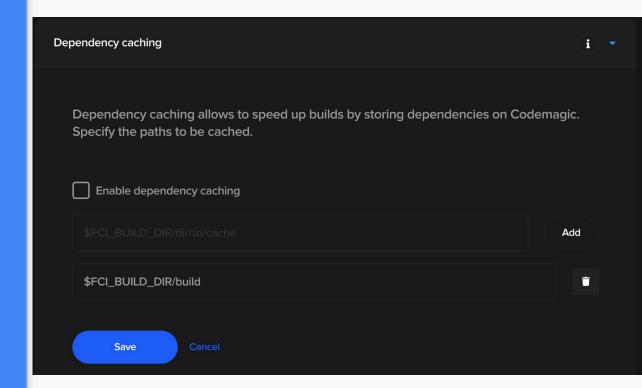
# Environment variables

- For private info that is not committed to Git
- FCI\_CLONE\_UNSHALLOW: useful for automated buildNumber using
  - git rev-list --count



## Dependency Caching

- Default setting



### Post clone script

- Use your Environment variable here
- Need to start with #!/bin/sh
- This script is run immediately after git clone success

#### Post-clone script

Run script after the clone phase has finished

#!/bin/sh

echo \$FIREBASE\_CONFIG > ./lib/firebase\_config.dart echo \$FIREBASE\_SERVICE > ./android/app/google-services.json

#### Pre-test script

Run script before the test phase has started

#!/bin/sh

set -e # exit on first failed command

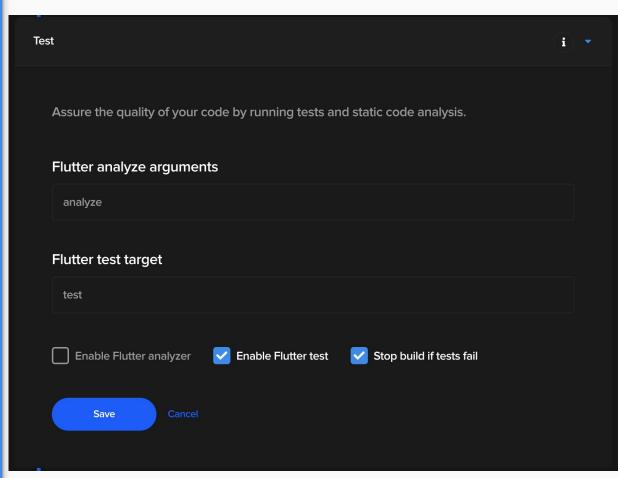
set -x # print all executed commands to the log

Save

Cance

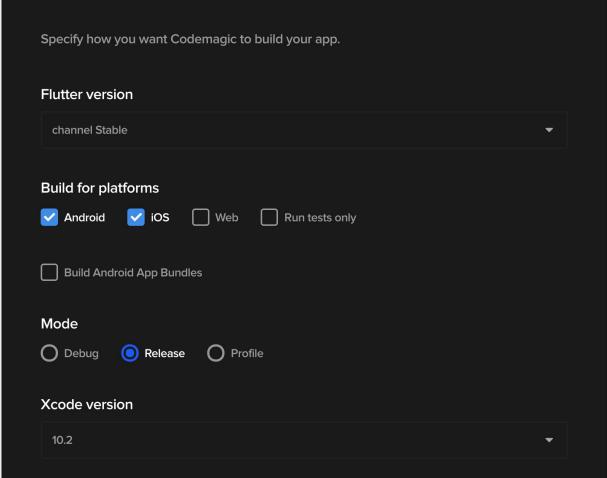
#### Test

- Run flutter test
- Recommend to use flutter analyze here
- Tips: use --flutter-repo to analyze your flutter repo packages



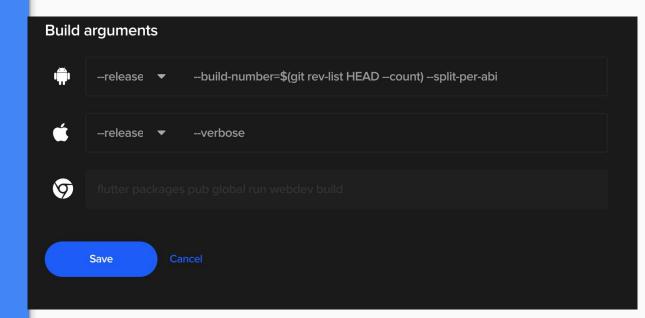
### Build (part 1)

- Flutter channel
- Platform
- Mode
  - Need Code Sign for Release Mode
- Xcode version



# Build - Build arguments

- You can still change your build mode here.
- FCI\_CLONE\_UNSHALLOW is used in this step if you use the command 'git rev-list HEAD --count'
- Need Code Signing for release mode



## Code Signing -Setup Android (1)

- Keystore
- Use key.properties for release

#### **Steps**

- Create key.properties file
- Get signing details from your key.properties file
- Configure release buildType

To Be Continued...

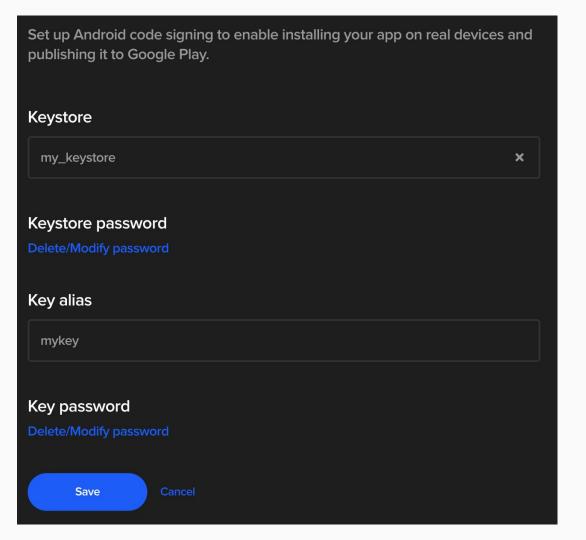
```
keyPassword=<password from previous step>
keyAlias=key
storeFile=<location of the key store file, such as /Users/<user name>/key.jks>
   def keystoreProperties = new Properties()
   def keystorePropertiesFile = rootProject.file('key.properties')
   if (keystorePropertiesFile.exists()) {
       keystoreProperties.load(new FileInputStream(keystorePropertiesFile))
    signingConfigs {
        release {
            keyAlias keystoreProperties['keyAlias']
            keyPassword keystoreProperties['keyPassword']
            storeFile file(keystoreProperties['storeFile'])
            storePassword keystoreProperties['storePassword']
    buildTypes {
        release {
            signingConfig signingConfigs.release
```

storePassword=<password from previous step>

## Code Signing -Setup Android (2)

#### Easy:

- upload your keystore
- Enter keystore password
- Enter your alias
- Give alias password



### Code Signing - Setup iOS (1)

#### Easy way:

- Use CodeMagic automatic signing
- Success rate: 0 for me

#### My Way

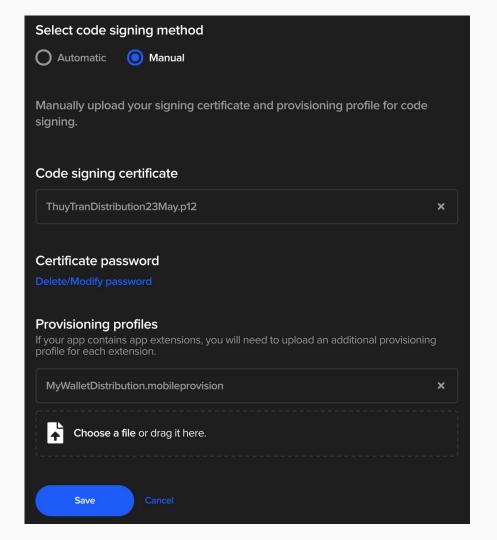
- Upload distribution key in p12 format
- Upload provisioning profile

To Be Continued...

## Code Signing -Setup iOS (2)

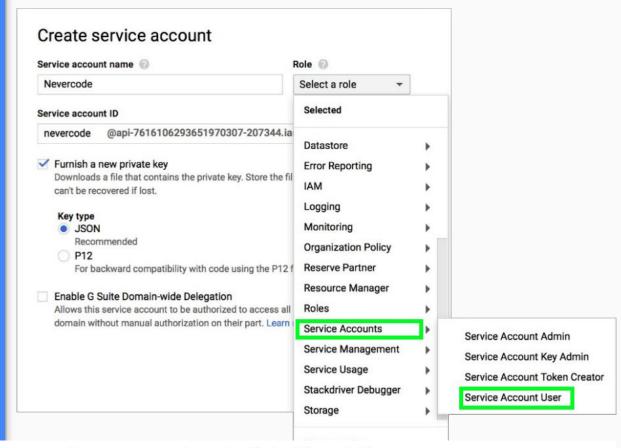
#### **Easy - but tricky:**

- Choose the correct distribution key and provisioning profile
- Generate P12 key for your distribution key
- Give your password



#### Publish

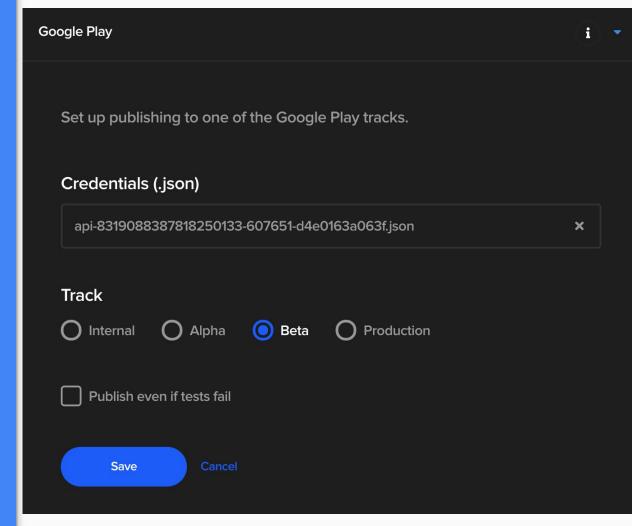
- Android (1)
- Setup Service Account in Google Play to allow third party upload
- After all steps are done, download the json file and keep it to use later.



- 7. Find the new service account and click Grant Access.
- 8. Select **Release manager** from the Role dropdown and click **Add user**.

#### Publish

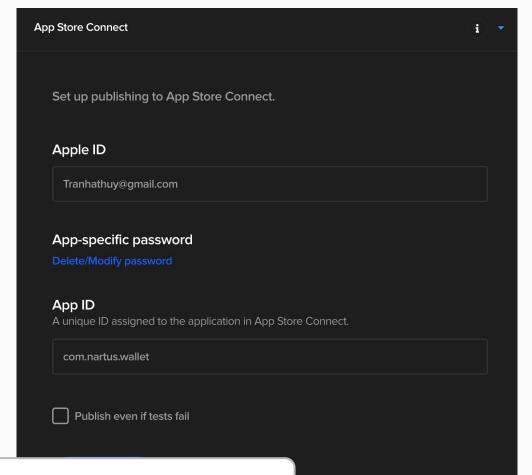
- Android (2)
- Upload Service Account json file
- Select channel to publish to



#### Publish - iOS

#### **Easy - but tricky:**

- Enter your AppleID and password
- App-specific password is tricky part



https://appleid.apple.com/account/manage

## Good

But not great



Q&A

Any Questions?



Suesi Tran An Fluttering Ando-chan