Build & Deploy iOS TestFlight

with Azure DevOps Pipeline

Why?

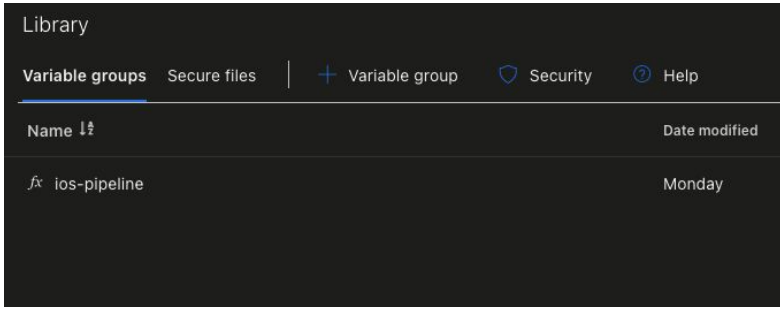
- iOS app development
- Apple upholds a strict review process
- Automation = Saves time!

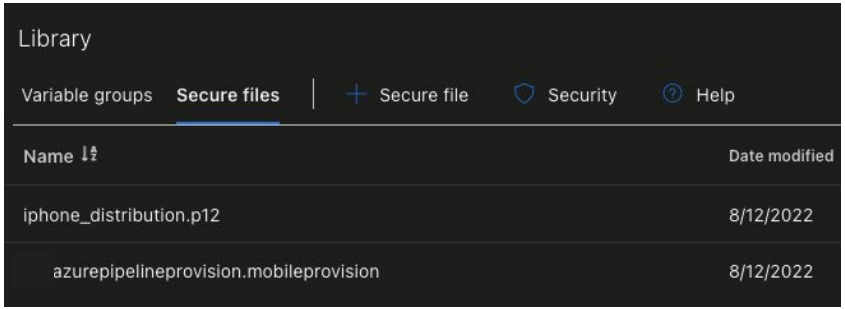
The Tasks

- InstallAppleCertificate@2
- InstallAppleProvisioningProfile@1
- CocoaPods@0 (Not needed if you don’t use Pods for your iOS application)
- Xcode@5
- CopyFiles@2
- PublishBuildArtifacts@1
- AppStoreRelease@1

Variable Groups & Secure Files

- Credentials, variables in variable groups.
  - Repository → Pipelines → Library → Variable Group
- Files will be in Secure files.
  - Repository → Pipelines → Library → Secure files.
- (Don’t forget to add pipeline permission on both)





First Line of Code

```
variables:
  - group: ios-pipeline
```

Base Case

- No trigger for my case

- Compile with the latest version of mac agent available.
  - Could be different depending on the project itself.

```
variables:
  - group: ios-pipeline

trigger:
  - none # master # release-branch

pool:
  vmImage: "macos-latest" #'macos-10.14'

steps:
…
```

First Task - Install Apple Certificate

- Create certificate in Apple Store Connect
  - Download certificate
  - Sign certificate with KeyChain
  - Add password and download it as p12 format
- Add the corresponding name of the file and the password here.

```
- task: InstallAppleCertificate@2
  inputs:
    certSecureFile: '$(p12FileName)'
    certPwd: '$(p12Password)'
    keychain: 'temp'
    deleteCert: true
```

Second Task - Install Apple Provision Profile

The provision profile is located at Apple Developer:

- If it doesn't exist, create one!
- Download, and add it in secure files.

```
- task: InstallAppleProvisioningProfile@1
  displayName: "Install Provisioning Profile"
  inputs:
    provisioningProfileLocation: 'secureFiles'
    provProfileSecureFile:
'$(provisioningProfile)'
    removeProfile: true
```

Third Task (Optional) - CocoaPods

- Setting `forceRepoUpdate` will force running 'pod repo update' before install.
- Don’t add this task if you don’t use Pods in your project

```
- task: CocoaPods@0
  inputs:
    workingDirectory: 'app/MyProject'
    forceRepoUpdate: false
```

CocoaPods - Adding code to Podfile

```
post_install do |installer|
    installer.pods_project.build_configurations.each do |config|
        config.build_settings['CODE_SIGNING_REQUIRED'] = "NO"
        config.build_settings['CODE_SIGNING_ALLOWED'] = "NO"
    end
end
```

New variables

```
variables:
  - group: ios-pipeline
  - name: configuration # <--
    value: 'Release'
  - name: sdk # <--
    value: 'iphoneos'
```

The Tasks

- ✅ Variable Groups & Secure Files
- ✅ InstallAppleCertificate@2
- ✅ InstallAppleProvisioningProfile@1
- ✅ CocoaPods@0 (Not needed if you don’t use Pods for your iOS application)
- Xcode@5
- CopyFiles@2
- PublishBuildArtifacts@1
- AppStoreRelease@1

Fourth Task - Build Project

- The xcodeVersion can be set to default or to a specific version depending on your project

- '$(APPLE_CERTIFICATE_SIGNING_IDENTITY) and '$(APPLE_PROV_PROFILE_UUID)' are predefined variables.
  - Smart enough to understand.

```
- task: Xcode@5
  inputs:
    actions: "clean build"
    scheme: "MIS"
    sdk: '$(sdk)'
    configuration: '$(configuration)'
    xcWorkspacePath: "**/MIS.xcworkspace"
    xcodeVersion: 'default'
    packageApp: true
    # Signing
    signingOption: 'manual'
    signingIdentity: '$(APPLE_CERTIFICATE_SIGNING_IDENTITY)'
    provisioningProfileUuid: '$(APPLE_PROV_PROFILE_UUID)'
```

Fifth Task - Copy & Publish Artifact

1. Copy the ipa from the build sources directory ➡ the artifact directories.

2. Publish the ipa so it can be available to download by clicking on the Artifacts button

```
- task: CopyFiles@2
  inputs:
    contents: "**/*.ipa"
    targetFolder: "$(Build.ArtifactStagingDirectory)"
    overwrite: true

- task: PublishBuildArtifacts@1
  inputs:
    pathtoPublish:
"$(Build.ArtifactStagingDirectory)/output/$(sdk)/$(configuration)"
    artifactName: "drop"
    publishLocation: "Container"
```

Sixth Task - Release to App Store

```yaml
- task: AppStoreRelease@1
  displayName: 'Publish to the App Store TestFlight track'
  inputs:
    authType: "UserAndPass" # "Service Connection" "App Store Connect API Key"
    username: "$(appStoreUsername)"
    password: "$(appStorePassword)"
    isTwoFactorAuth: true
    appSpecificPassword: "$(appPassword)"
    releaseTrack: "TestFlight"
    appIdentifier: "se.ctk.mis"
    appType: "iOS"
    #distributedToExternalTesters: true
    #externalTestersGroups: 'XXX Testers'
    ipaPath: "$(Build.ArtifactStagingDirectory)/**/*.ipa"
    shouldSkipWaitingForProcessing: true
    appSpecificId: "$(appStoreSpecificAppleId)"
    teamId: "$(teamID)"
    teamName: "$(teamName)"
```

The Tasks

- ✅ Variable Groups & Secure Files
- ✅ InstallAppleCertificate@2
- ✅ InstallAppleProvisioningProfile@1
- ✅ CocoaPods@0 (Not needed if you don’t use Pods for your iOS application)
- ✅ Xcode@5
- ✅ CopyFiles@2
- ✅ PublishBuildArtifacts@1
- ✅ AppStoreRelease@1

Questions?


a non-toxic senior who supports, guides and teaches me
Thank you
I will never forget this
Me
I would fight for you