

# App Store Connect API



# What does is it?

App Store Connect API - it is an API that allows you to customize and automates your workflows.

This standards-based REST API lets you automate tasks across developer tools, such as App Store Connect, Xcode, and Certificates, Identifiers & Profiles, to give you greater flexibility and increased efficiency in your workflows.

You can use it for development, beta testing, managing app metadata, reporting, and more.

# What does it allow to realise ?



## **App Management**

Create new versions of your app, set up pre-orders, manage phased releases for version updates, and submit your app to App Review.



## **Provisioning**

Streamline app development by managing bundle IDs, signing certificates, development devices, and provisioning profiles.



## **App Metadata**

Build and maintain your App Store product page by uploading and managing assets, such as your app description, screenshots, and app previews.



## **Reporting Sales and Trends**

Download reports to view the number of first-time downloads, sales, proceeds, pre-orders, subscriptions activity, and more for your apps on all Apple platforms.



## **Pricing and Availability**

Set your app's pricing and territory availability. Access reference information, such as a list of available App Store territories, app price tiers, and proceeds for currencies that the App Store supports.



## **TestFlight**

Manage beta testing by automating tester management and build distribution. You can add and remove testers, manage tester groups, assign builds to testers, submit builds for beta app review, and more.



## **Power and Performance**

Download power and performance metrics and diagnostics logs to monitor app performance indicators, such as launch time, hang rate, disk writes, memory use, and battery life. This information is also available in Xcode.



## **Users and Roles**

Integrate your Apple Developer Program team membership with your organization's internal directory to automate tasks associated with team management. For example, you can automatically revoke access to App Store Connect when a user leaves your organization.

```

19
20 "servers" : [ {
21   "url" : "https://api.appstoreconnect.apple.com/"
22 } ],
23 "paths" : {
24   "/v1/ageRatingDeclarations/{id}" : {
25     "patch" : {
26       "tags" : [ "AgeRatingDeclarations" ],
27       "operationId" : "ageRatingDeclarations-update_instance",
28       "requestBody" : {
29         "description" : "AgeRatingDeclaration representation",
30         "content" : {
31           "application/json" : {
32             "schema" : {
33               "$ref" : "#/components/schemas/AgeRatingDeclarationUpdateRequest"
34             }
35           }
36         },
37         "required" : true
38       },
39       "responses" : {
40         "400" : {
41           "description" : "Parameter error(s)",
42           "content" : {
43             "application/json" : {
44               "schema" : {
45                 "$ref" : "#/components/schemas/ErrorResponse"
46               }
47             }
48           }
49         },
50         "403" : {
51           "description" : "Forbidden error",
52           "content" : {
53             "application/json" : {
54               "schema" : {
55                 "$ref" : "#/components/schemas/ErrorResponse"
56               }
57             }
58           }
59         },
60         "404" : {
61           "description" : "Not found error",
62           "content" : {
63             "application/json" : {

```

```
// Create a new App Store version
```

**POST** /v1/appStoreVersions

> HTTP/1.1 201 CREATED

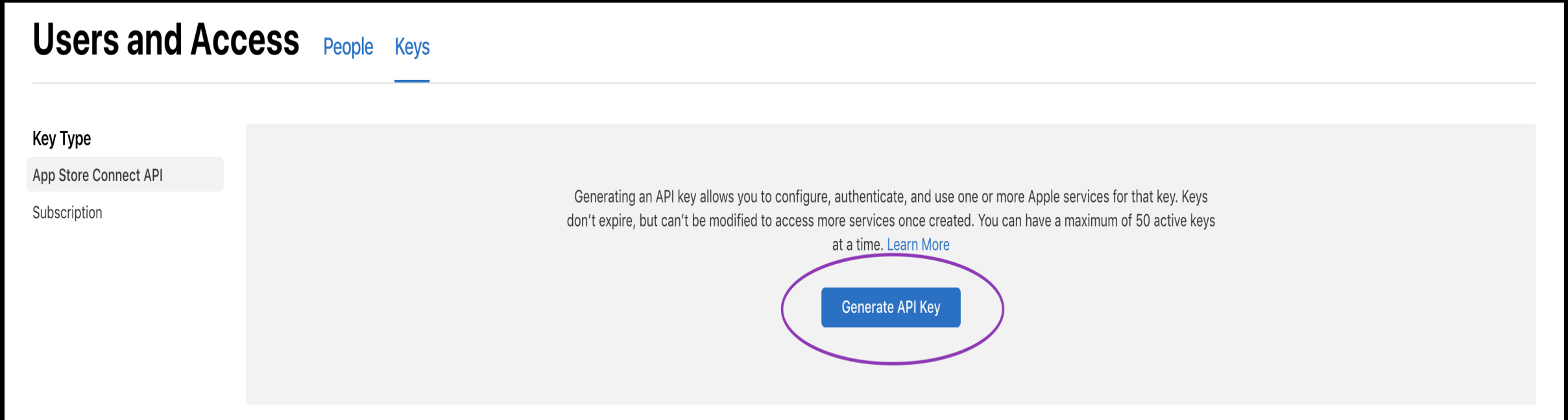
```

{
  "data": {
    "type": "appStoreVersions",
    "id": "ef374ee6-edaf-416c-9522-75cf2008300a",
    "attributes": {
      "platform": "IOS",
      "versionString": "1.1"
    },
    "relationships": {
      // ...
    }
  }
}

```

# How setup App Store API Connect?

## 1) Create App Store API key



**\*For creating API Key you must have the role of Account Holder or Admin with corresponding permission**

Users and Access

People

Keys

Shared Secret

Key Type

App Store Connect API

Subscription

Generating an API key allows you to configure, authenticate, and use one or more Apple services for that key. Keys don't expire, but can't be modified to access more services once created. You can have a maximum of 50 active keys at a time. [Learn More](#)

Issuer ID ?

Copy

Active (3) +

Edit

NAME	GENERATED BY	KEY ID	LAST USED	ACCESS
Test API Key	Deep Shah	<div></div> <a href="#">Copy Key ID</a>		Developer

[Download API Key](#)



## 2) Generate JWT Token

JWT Token must follow next guidelines:

- Issuer ID: The ID found on the top of App Store Connect
- Private Key ID: The ID associated with Private Key on App Store Connect
- Expiration Time: 20 min maximum, the token cannot be valid more than 20 min so that we have to make sure that, we will create new token before it expires.
- Audience: This is constant with API version value usually “applestoreconnect-v1”
- Algorithm: This is JWT algorithm required to generate token e.g ES256

```
require "base64"
require "jwt"

ISSUER_ID = "YOUR_ISSUER_ID"
KEY_ID = "YOUR_PRIVATE_KEY_ID"

private_key =
  OpenSSL::PKey.read(File.read(path_to_your_private_key/AuthKey_#
    {KEY_ID}.p8))

token = JWT.encode(
  {
    iss: ISSUER_ID,
    exp: Time.now.to_i + 20 * 60,
    aud: "appstoreconnect-v1"
  },
  private_key,
  "ES256",
  header_fields={
    kid: KEY_ID }
)

puts token
```

The JWT is almost available in all languages including Swift but it would be quicker to generate it using dynamic or interpreted languages like Ruby or Python. Here is a Ruby script that has been used in WWDC demo.

We can save this file as jwt.rb somewhere. You need to have JWT ruby gem installed for this script to run properly. Just replace the value of ISSUER\_ID and KEY\_ID and you will be good to go. This script can be run using

```
$ ruby jwt.rb
```

This will return a long token that we can use to access an App Store Connect API, we also need to create another token if we want to continue using API after 20 minutes.



### 3) Include the JWT in the Request's Authorization Header

Once you have a complete and signed token, provide the token in the request's authorization header as a bearer token.

The following example shows a curl command using a bearer token. Replace the text "[signed token]" with the value of the signed token itself.

```
curl -v -H 'Authorization: Bearer [signed token]'  
"https://api.appstoreconnect.apple.com/v1/apps"
```

Example in php curl



Where is should be launched?

---

What you need to do:

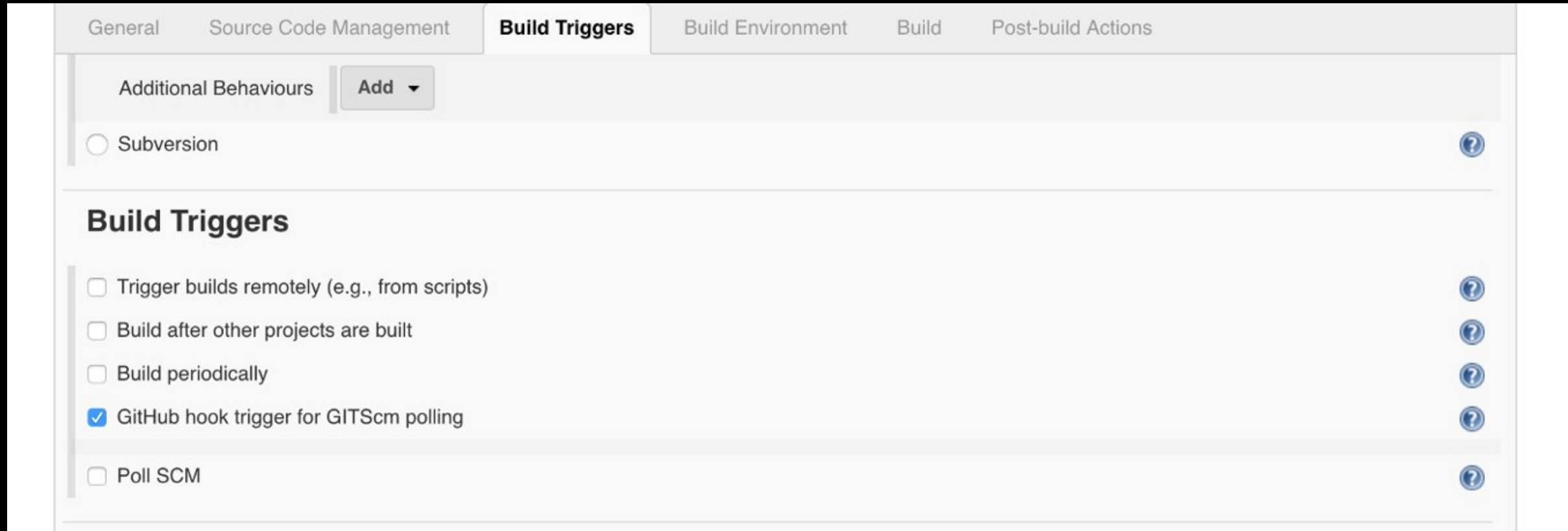
- 1) Setup Jenkins, Fastlane and Git repository
- 2) Connect Jenkins with Fastlane and Git repository

### 3) Creating Jenkins job which will execute Fastlane's actions after push in master branch

The screenshot shows the Jenkins configuration interface for a job named "MakeIPA". The "Source Code Management" tab is selected, showing the following configuration:

- Source Code Management:**
  - ☐ None
  - ☒ Git
- Repositories:**
  - Repository URL: `git@github.com:litoarias/jenkins_fastlane.git`
  - Credentials: `- none -` with an "Add" button.
  - Buttons: "Advanced...", "Add Repository"
- Branches to build:**
  - Branch Specifier (blank for 'any'): `*/master`
  - Buttons: "Add Branch" (with a red "X" icon)
- Repository browser:** `(Auto)`
- Additional Behaviours:** "Add" button
- Buttons:** "Save", "Apply"

## 4) Choose build trigger



The screenshot shows the Jenkins configuration interface with the 'Build Triggers' tab selected. The top navigation bar includes 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. Below the tabs, there is a section for 'Additional Behaviours' with an 'Add' button. The main content area is titled 'Build Triggers' and contains a list of checkboxes for different build triggers. The 'GitHub hook trigger for GITScm polling' option is selected with a blue checkmark. Each option has a help icon (a question mark in a circle) to its right.

General Source Code Management **Build Triggers** Build Environment Build Post-build Actions

Additional Behaviours Add ▾

☐ Subversion ?

### Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?

## 5) Execute command to deploy build on fastlane

General

Source Code Management

Build Triggers

Build Environment

**Build**

Post-build Actions

### Build

Execute shell

X

?

Command

```
fastlane generate_ipa_develop
```

[See the list of available environment variables](#)

Advanced...

Add build step ▾

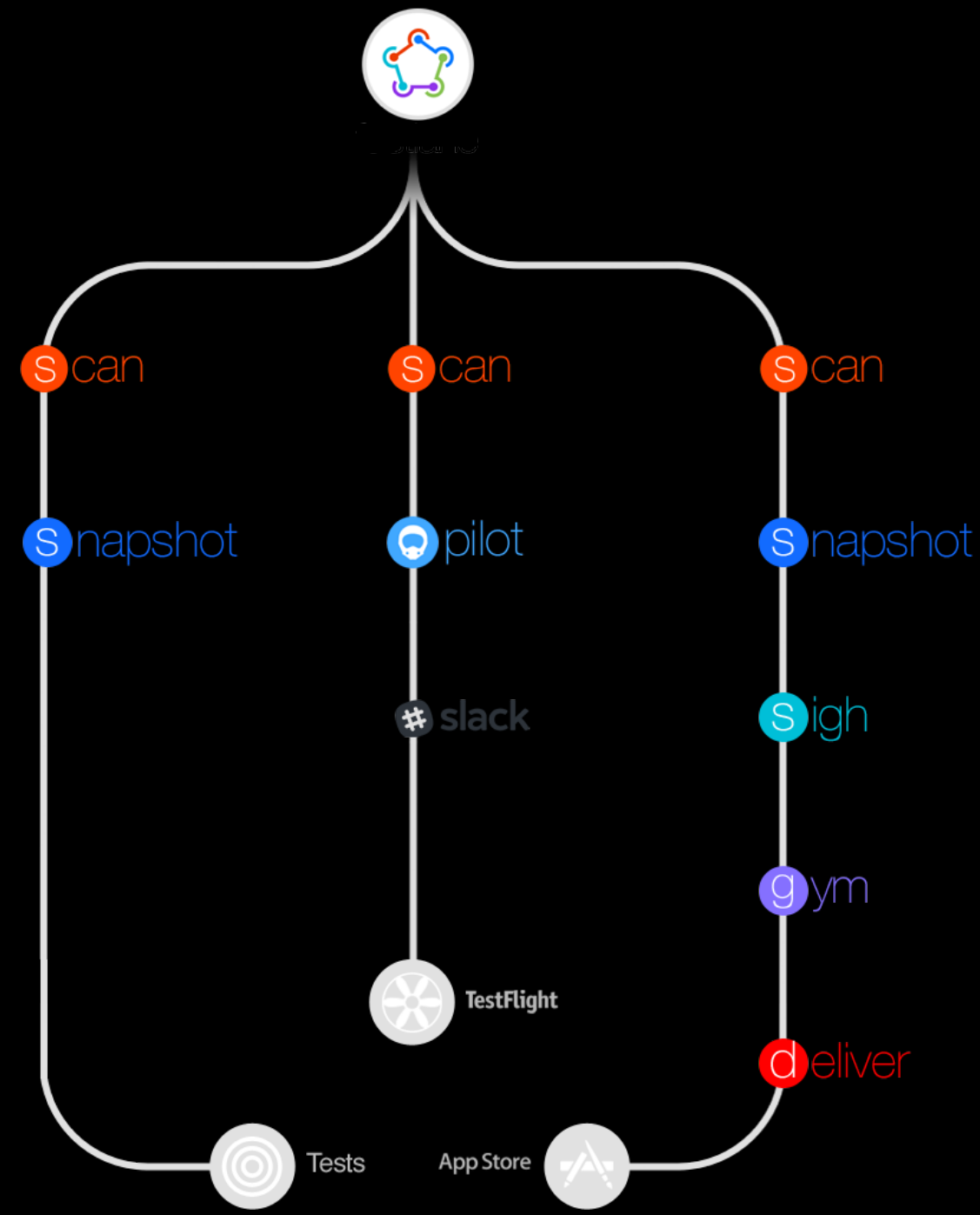
3) Depends of your needs in Apple Store Connect API you can choose Fastlane function for realise your automatisaton tasks

**Supported Actions/Tools**

The App Store Connect API has not been integrated into all tools and actions yet.

Below are the statuses of each tool:

Name	Apple ID	API Key
pilot	Yes	Yes
deliver	Yes	Yes
sigh	Yes	Yes
cert	Yes	Yes
match	Yes	Yes
produce	Partial	No
pem	No	No
precheck	Yes (except for IAP)	Yes (except for IAP)





*scan* makes it easy to run tests of your iOS and Mac app on a simulator or connected device.



*snapshot* generates localized iOS and tvOS screenshots for different device types and languages for the App Store and can be uploaded using (*deliver*).

\*this action execute all necessary simulator and creates a snapshots. For creating them you should setup UITest Target and execute corresponding command and you will get html file with all screenshots.





**Pilot** makes it easier to manage your app on Apple's TestFlight. You can:

- Upload & distribute builds
- Add & remove testers
- Retrieve information about testers & devices
- Import/export all available testers



*sigh* can create, renew, download and repair provisioning profiles (with one command). It supports App Store, Ad Hoc, Development and Enterprise profiles and supports nice features, like auto-adding all test devices.



*gym* builds and packages iOS apps for you. It takes care of all the heavy lifting and makes it super easy to generate a signed ipa or app file 💪



*deliver* uploads screenshots, metadata and binaries to App Store Connect. Use *deliver* to submit your app for App Store review.

# Fastlane installation guide

1) Install bundler using this command:

```
$ gem install bundler
```


2) Execute next command in project folder for creating Gemfile

```
$ bundle init
```



### 3) Edit Gemfile as showing on the picture

 Gemfile ×

Users > user > Desktop > Messenger > Messenger >  Gemfile

```
1  source "https://rubygems.org"
2
3  gem "fastlane"
4
5
6
```

4) Install Fastlane library using bundle by executing the command in project folder:

```
% bundle install
```

So, we use bundler only for installing Fastlane in our project directory. It's the recommended method.

5) Execute next command in project folder for initialise folder with fast lane files

```
% fastlane init
```

After that fast lane folder with two files will be create

▼	Folder	Messenger	Yesterday, 17:34	--
▼	Folder	fastlane	Today, 23:23	--
	File	Appfile	Yesterday, 22:11	138 bytes
	File	Fastfile	Yesterday, 22:11	611 bytes
	File	Gemfile	8 October 2020, 21:02	48 bytes
	File	Gemfile.lock	8 October 2020, 22:43	5 KB

## 6) Edit App file



The screenshot shows a code editor with three tabs: 'Gemfile', 'Appfile', and 'Fastfile'. The 'Appfile' tab is active. The breadcrumb path is 'Users > user > Desktop > Messenger > Messenger > fastlane > Appfile'. The code in the Appfile is as follows:

```
1 | app_identifier "com.chisw.Messenger1" # The bundle identifier of your app
2   apple_id "kyrylo.triskalo@chisw.com" # Your Apple email address
3
```

\*The Appfile stores useful information that are used across all *fastlane* tools like your *Apple ID* or the application *Bundle Identifier*, to deploy your lanes faster and tailored on your project needs.



## 7) Edit Fastfile

```
desc "Create app on Dev Portal and Apple Connect"
lane :create_app do
  # authenticate using a cookie-based web session works as a default connection

  # authenticate using a app store connect api key
  app_store_connect_api_key(
    key_id: "46XMU4Z473",
    issuer_id: "04a13ec8-b0d9-4cfd-ba94-4af120582ae9",
    key_filepath: "/Users/kirilltriskalo/Desktop/Messenger/Messenger/fastlane/AuthKey_46XMU4Z473.p8",
  )
  produce(
    app_name: 'Messenger2'
  )
end

platform :ios do
```

Lane **create\_app** – function which contain advance authorization using API key not regular authorization via web session.

Action **produce** creates new iOS apps on both the Apple Developer Portal and App Store Connect with the minimum required information.

```

platform :ios do
  desc "Sync signing"
  lane :signing do

  end

  desc "Build"
  lane :build do
    signing
  end

  desc "Release"
  lane :release do
    build
  end
end

```

**platforms ios** it's a body for building app where we can write actions like **match**.

**match** responsible for creating all required certificates & provisioning profiles and stores them in a separate git repository, Google Cloud, or Amazon S3

example of using **match**:

```

5
6 platform :ios do
7   desc "Sync signing"
8   lane :signing do
9     sync_code_signing # match
10
11     mapping = Actions.lane_context[
12       SharedValues::MATCH_PROVISIONING_PROFILE_MAPPING
13     ]
14     update_code_signing_settings(
15       profile_name: mapping[ENV['MATCH_APP_IDENTIFIER']]
16     )
17   end

```

## 8) Execute command **fastlane ios build** in project folder

```
MacBook-Pro-Kirill-2:Messenger kirilltriskalo$ fastlane ios build
[✓] 🚀
[22:12:25]: fastlane detected a Gemfile in the current directory
[22:12:25]: However, it seems like you didn't use `bundle exec`
[22:12:25]: To launch fastlane faster, please use
[22:12:25]:
[22:12:25]: $ bundle exec fastlane ios build
[22:12:25]:
[22:12:25]: Get started using a Gemfile for fastlane https://docs.fastlane.tools/getting-started/i
os/setup/#use-a-gemfile
[22:12:28]: Driving the lane 'ios build' 🚀
[22:12:28]: -----
[22:12:28]: --- Step: Switch to ios signing lane ---
[22:12:28]: -----
[22:12:28]: Cruising over to lane 'ios signing' 🚗
[22:12:28]: Cruising back to lane 'ios build' 🚗

+-----+-----+-----+-----+
|                fastlane summary                |
+-----+-----+-----+-----+
| Step | Action                               | Time (in s) |
+-----+-----+-----+-----+
| 1     | Switch to ios signing lane          | 0            |
+-----+-----+-----+-----+

[22:12:28]: fastlane.tools finished successfully 🎉
MacBook-Pro-Kirill-2:Messenger kirilltriskalo$
```

## 9) Execute `fastlane create_app` in project folder

```
[MacBook-Pro-Kirill-2:Messenger kirilltriskalo$ fastlane create_app
[✓] 🚀
[22:12:35]: fastlane detected a Gemfile in the current directory
[22:12:35]: However, it seems like you didn't use `bundle exec`
[22:12:35]: To launch fastlane faster, please use
[22:12:35]:
[22:12:35]: $ bundle exec fastlane create_app
[22:12:35]:
[22:12:35]: Get started using a Gemfile for fastlane https://docs.fastlane.tools/getting-started/i
os/setup/#use-a-gemfile
[22:12:37]: Driving the lane 'create_app' 🚀
[22:12:37]: -----
[22:12:37]: --- Step: app_store_connect_api_key ---
[22:12:37]: -----
[22:12:37]: -----
[22:12:37]: --- Step: produce ---
[22:12:37]: -----

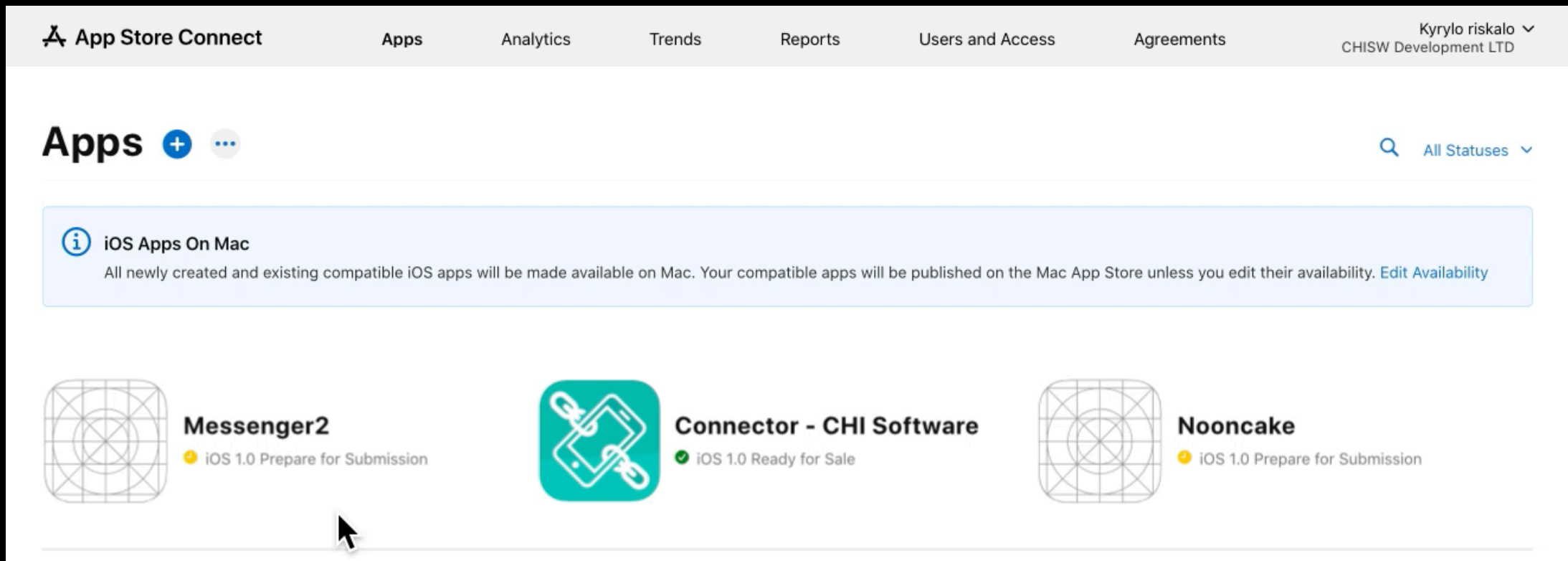
+-----+-----+
|          Summary for produce 2.162.0          |
+-----+-----+
| app_name      | Messenger2      |
| username      | kyrylo.triskalo@chisw.com |
| app_identifier | com.chisw.Messenger1 |
| sku           | 1602357157      |
| platform      | ios             |
| language      | English         |
| skip_itc      | false           |
| skip_devcenter | false           |
+-----+-----+
```

```
[22:12:43]: Creating new app 'Messenger2' on App Store Connect
[22:12:43]: Sending language name is deprecated. 'English' has been mapped to 'en-US'.
[22:12:43]: Please enter one of available languages: ["ar-SA", "ca", "cs", "da", "de-DE", "el", "en-AU", "en-CA", "en-GB", "en-US", "es-ES", "es-MX", "fi", "fr-CA", "fr-FR", "he", "hi", "hr", "hu", "id", "it", "ja", "ko", "ms", "nl-NL", "no", "pl", "pt-BR", "pt-PT", "ro", "ru", "sk", "sv", "th", "tr", "uk", "vi", "zh-Hans", "zh-Hant"]
[22:12:51]: Ensuring version number
[22:12:51]: Successfully created new app 'Messenger2' on App Store Connect with ID 1535343571
```

```
+-----+-----+-----+
|                fastlane summary                |
+-----+-----+-----+
| Step | Action                      | Time (in s) |
+-----+-----+-----+
| 1     | app_store_connect_api_key   | 0            |
| 2     | produce                     | 14           |
+-----+-----+-----+
```

```
[22:12:51]: fastlane.tools finished successfully 🎉
MacBook-Pro-Kirill-2:Messenger kirilltriskalo$
```

We can see that all command was executed successfully, and our application was created on App Connect



Our application was uploaded to Apple Connect and was created bundle id "com.chisw.Messenger1" without any manual setup.

**THANK YOU  
FOR ATTENTIONS**