

Ionic Workshop 17

Apple App Store und Google Play Store

Heute schauen wir die einzelnen Schritte an, wie wir unsere App in die jeweiligen Stores hochladen können. Für Apple und Google wirst du einen Developer Account brauchen. Beginnen wir mit iOS, für Android ist der Prozess einiges einfacher und befindet sich gegen Schluss dieses Dokuments. Es gibt jedoch ein paar Schritte, welche für beide Plattformen gelten, daher macht es Sinn gleich hier mit dem Lesen zu starten. Wir starten mit dem Erstellen aller Zertifikate für iOS und Android. Ebenfalls schauen wir Push Notifications an, was heute schon fast Standard ist für jede App.

iOS: App ID

Jede iOS App braucht eine ID, sprich eine App ID. Diese muss bei Apple zuerst registriert werden. Im Developer Center, unter Certificates, Identifiers & Profiles > Identifiers > App IDs, kannst du eine neue App ID erstellen.

Unter Explicit App ID musst du die ID angeben, welche du im config.xml deines Ionic Projektes angegeben hast. Sie wird üblicherweise in der **Reverse-Domain-Name** Schreibweise geschrieben, also z.B. ch.romanrast.rhyschwimme

```
app.module.ts      config.xml x    app.component.ts    overview.txt
1   <?xml version='1.0' encoding='utf-8'?>
2   <widget id="ch.romanrast.rhyschwimme" version="0.0.1" xmlns='
3       <name>Rhyschwimme</name>
```

The screenshot shows the Apple Developer Center interface. On the left, there's a sidebar with navigation links: 'Certificates, Identifiers & Profiles' (selected), 'iOS, tvOS, watchOS' dropdown, 'Certificates' (All, Pending, Development, Production, APNs Auth Key), 'Identifiers' (selected, App IDs, Pass Type IDs, Website Push IDs, iCloud Containers, App Groups, Merchant IDs), 'Devices' (All, Apple TV, Apple Watch, iPad, iPhone, iPod Touch), and 'Provisioning Profiles' (All, Development). The main content area is titled 'Register iOS App IDs' and 'ID Registering an App ID'. It contains fields for 'Name' (with placeholder 'You cannot use special characters such as @, &, *, ',") and 'Value' (with placeholder '(Team ID)'). Below these, there's a section for 'App ID Prefix' and 'App ID Suffix'. A note at the bottom says 'Explicit App ID' is selected, with a note: 'If you plan to incorporate app services such as Game Center, In-App Purchase, Data Protection, and iCloud, or want a provisioning profile unique to a single app, you must'. The URL in the browser bar is https://developer.apple.com/account/ios/identifier/bundle/create.

Ionic Workshop 17

App ID Suffix

Explicit App ID
If you plan to incorporate app services such as Game Center, In-App Purchase, Data Protection, and iCloud, or want a provisioning profile unique to a single app, you must register an explicit App ID for your app.

To create an explicit App ID, enter a unique string in the Bundle ID field. This string should match the Bundle ID of your app.

Bundle ID: **ch.romanrast.rhyschwimme**
We recommend using a reverse-domain name style string (i.e., com.domainname.appname). It cannot contain an asterisk (*).

Wildcard App ID
This allows you to use a single App ID to match multiple apps. To create a wildcard App ID, enter an asterisk (*) as the last digit in the Bundle ID field.

Bundle ID:
Example: com.domainname.*

App Services
Select the services you would like to enable in your app. You can edit your choices after this App ID has been registered.

Enable Services:

- App Groups
- Apple Pay
- Associated Domains
- Data Protection
 - Complete Protection
 - Protected Unless Open
 - Protected Until First User Authentication
- Game Center
- HealthKit
- HomeKit
- iCloud
 - Compatible with Xcode 5
 - Include CloudKit support (requires Xcode 6)
- In-App Purchase
- Inter-App Audio
- Network Extensions
- Personal VPN
- Push Notifications
- SiriKit
- Wallet
- Wireless Accessory Configuration

Selektiere dann **Push Notifications**, wenn du Push Notifications in deiner App verwenden möchtest. Danach weiter und registrieren.

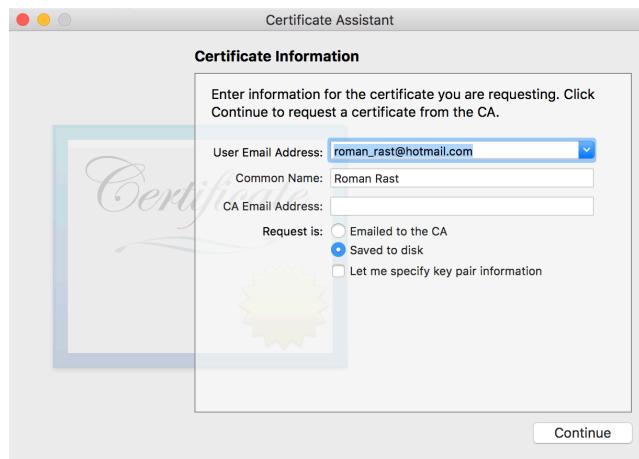
iOS: Certificate

Als nächstes brauchen wir ein **Certificate** für unsere App. Dazu müssen wir zuerst ein **certificate signing request** File erstellen. Öffne dazu auf deinem Mac die Schlüsselbundverwaltung (Keychain Access).

Geh dann unter Keychain Access > Certificate Assistant > Request a Certificate From a Certificate Authority.



Ionic Workshop 17

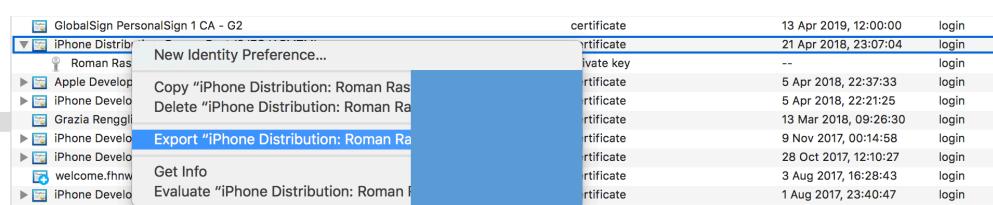


Das Feld CA Email Address solltest du leer lassen und wähle **Save to disk aus**. Danach erhältst du das Signing Request File xxx.certSigningRequest.

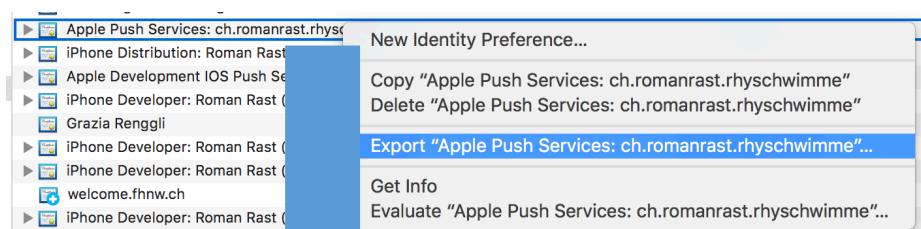
Nun können wir das Certificate erstellen. Hierzu gibt es zwei Varianten, ein Development und ein Production Certificate. Mit der Development Variante haben wir die Möglichkeit zu debuggen und Infos in der Konsole zu bekommen, wir können aber nur auf max. 100 Devices die App installieren. Das Production Certificate kann jedoch für eine unlimitierte Anzahl Devices verwendet werden, jedoch fehlt dann der Debugmodus. Das Erstellen beider Zertifikate ist ziemlich ähnlich.

Gehe dazu unter Certificates, Identifiers & Profiles > Certificates > Development oder Production. Hier kannst du zwischen **iOS App Development (dev) und App Store and Ad Hoc (prod)** wählen, sowie auch die beiden Push Certificates Versionen für dev und prod (du musst jeweils für Push diesen Ablauf nochmals durchmachen, da nur Radiobuttons zur Option stehen). Klick dich durch und du wirst nach dem CSR File gefragt, welches wir vorhin als xxx.certSigningRequest erstellt haben. Lade es hoch und danach kannst du das **.cer** File runterladen ;)

So, als nächstes muss das **.cer** File in ein **.p12** File umgewandelt werden. Dazu kannst du auf das .cer File doppelklicken und es wird in die Schlüsselbundverwaltung kopiert, oder du kannst es einfach in die Schlüsselbundverwaltung unter **login** hereinziehen. Danach mit Rechtsklick > Export „iPhone Distribution:...“ das .p12 File herunterladen.



Gib ein Passwort an und speichere es und für Push dasselbe.

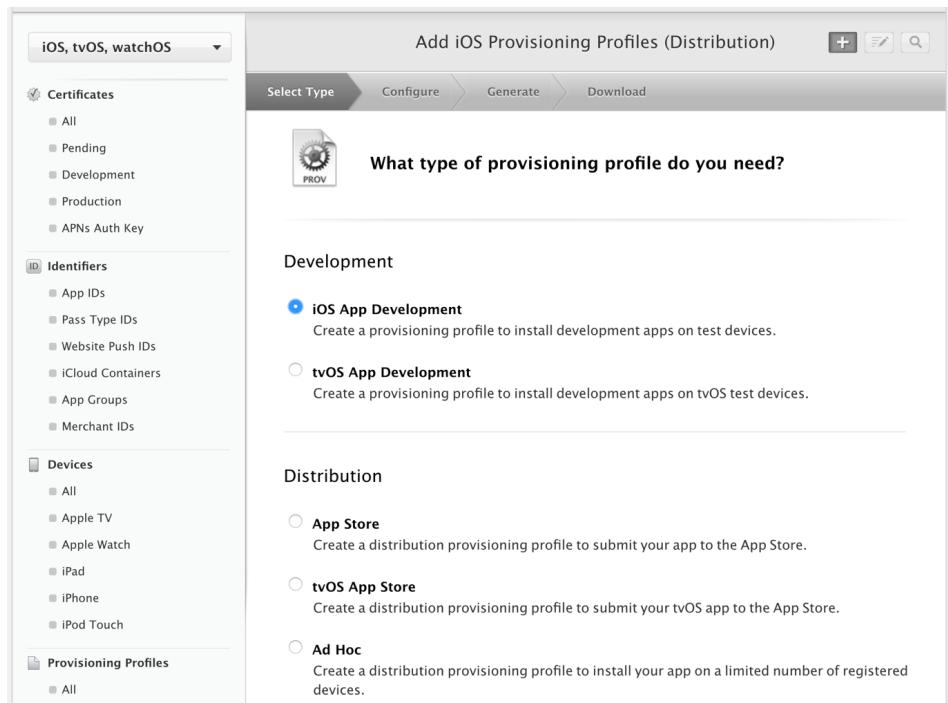


Ionic Workshop 17

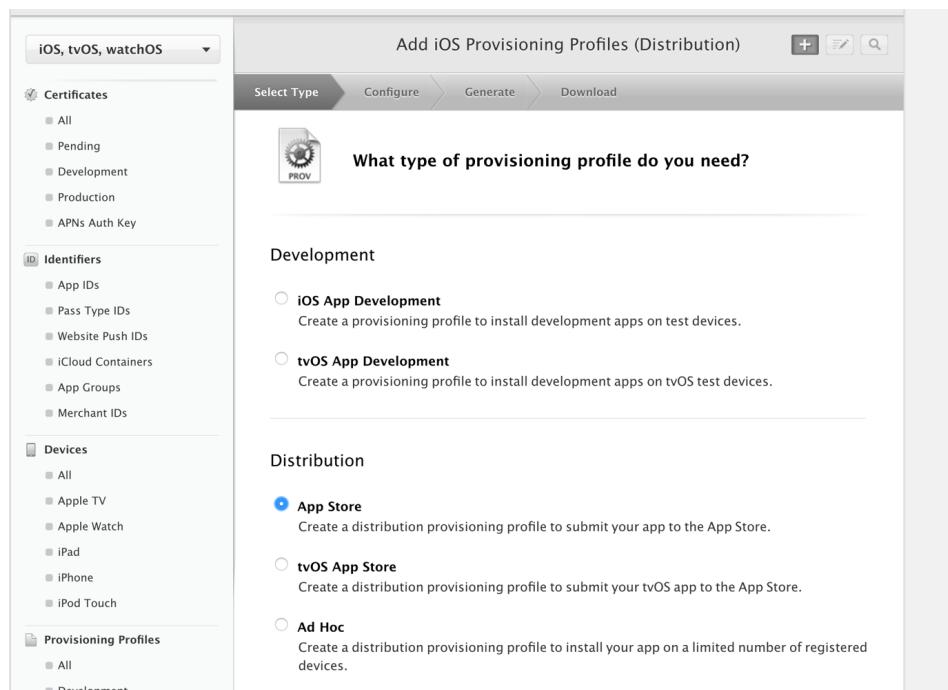
iOS: Provisioning Profiles

Damit die App installiert werden kann, brauchen wir ein Provisioning Profile.

Unter Certificates, Identifiers & Profiles > Provisioning Profiles > Development oder Distribution wählen. Danach auf Continue und die App ID auswählen, welche zur ID in deinem Ionic Projekt im config.xml steht. Klicke dich wie auf den folgenden Screens durch und lade das xxx.mobileprovision File herunter.



Oder Distribution für App Store



Ionic Workshop 17

The screenshot shows the "Certificates, Identifiers & Profiles" section of the Apple Developer Portal. The left sidebar lists categories: Certificates (All, Pending, Development, Production, APNs Auth Key), Identifiers (App IDs, Pass Type IDs, Website Push IDs, iCloud Containers, App Groups, Merchant IDs), Devices (All, Apple TV, Apple Watch, iPad, iPhone, iPod Touch), and Provisioning Profiles (...). The main area is titled "Add iOS Provisioning Profiles (Distribution)" and shows a progress bar: Select Type → Configure → Generate → Download. A sub-section titled "Select App ID." displays a list of identifiers. One identifier, "Rhyschwimme (.ch.romanrast.rhyschwimme)", is selected and highlighted in blue. A note below explains the use of App IDs for Game Center, In-App Purchase, and Push Notifications.

If you plan to use services such as Game Center, In-App Purchase, and Push Notifications, or want a Bundle ID unique to a single app, use an explicit App ID. If you want to create one provisioning profile for multiple apps or don't need a specific Bundle ID, select a wildcard App ID. Wildcard App IDs use an asterisk (*) as the last digit in the Bundle ID field. Please note that iOS App IDs and Mac App IDs cannot be used interchangeably.

App ID: Rhyschwimme (.ch.romanrast.rhyschwimme)

Ionic Workshop 17

The image consists of two side-by-side screenshots of the Apple Developer Provisioning Portal.

Screenshot 1: Select certificates.

This screen shows the "Select Type" step of the provisioning profile creation process. The "Distribution" type is selected. A list of available certificates is shown, with one certificate being selected:

- Roman Rast (iOS Distribution) Jun 08, 2017
- Roman Rast (iOS Distribution) Apr 21, 2018

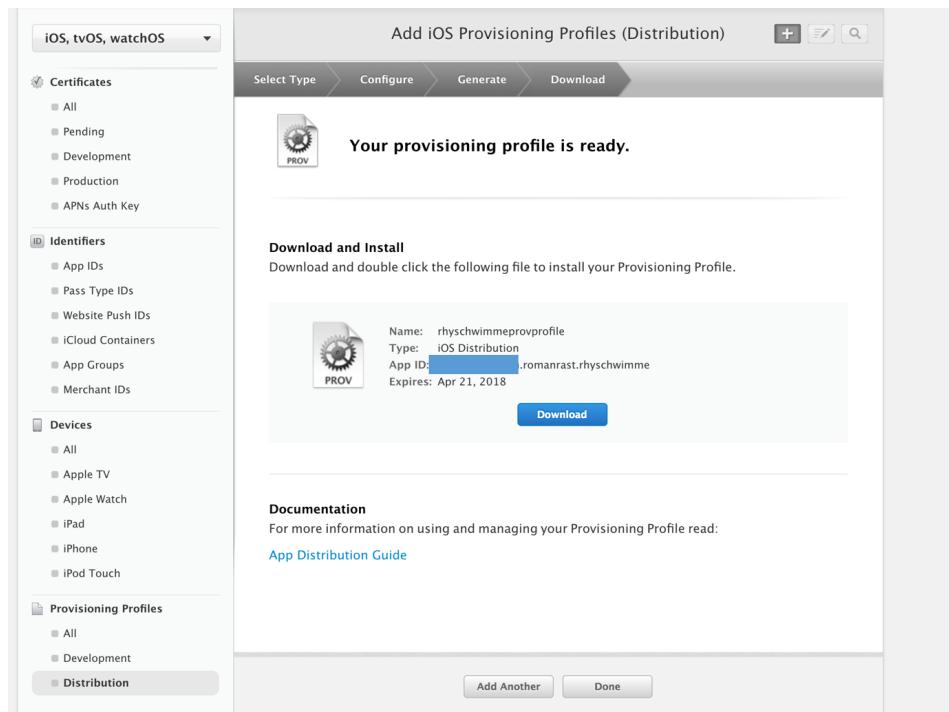
Screenshot 2: Name this profile and generate.

This screen shows the "Configure" step. The profile name is set to "rhyschwimmeprofile". Other configuration details are as follows:

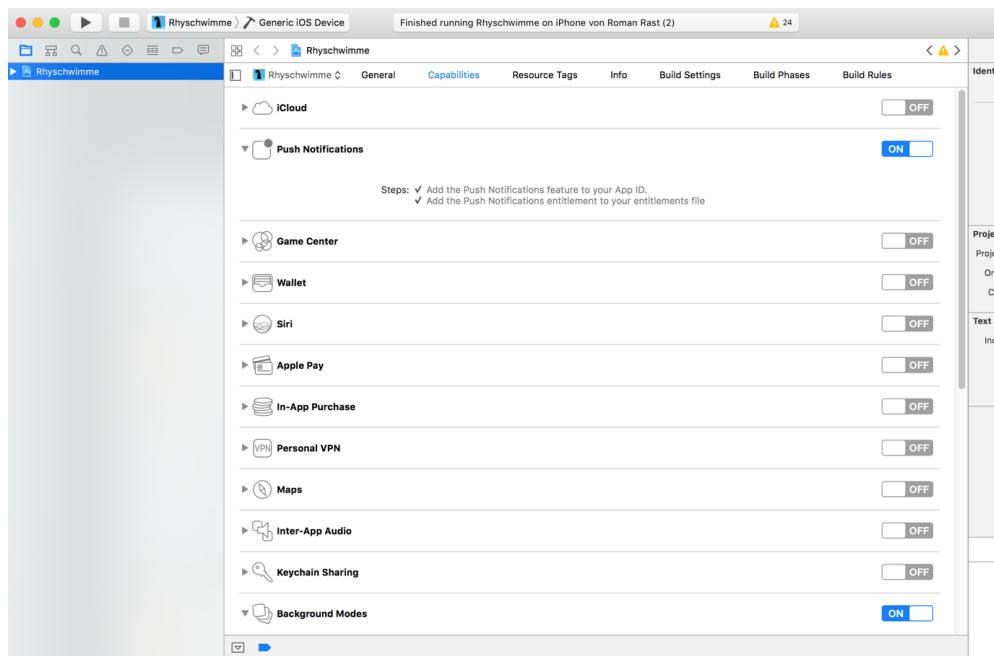
- Type: iOS Distribution
- App ID: Rhyschwimme (ch.romanrast.rhyschwimme)
- Certificates: 1 Included

Buttons at the bottom include "Cancel", "Back", and "Continue".

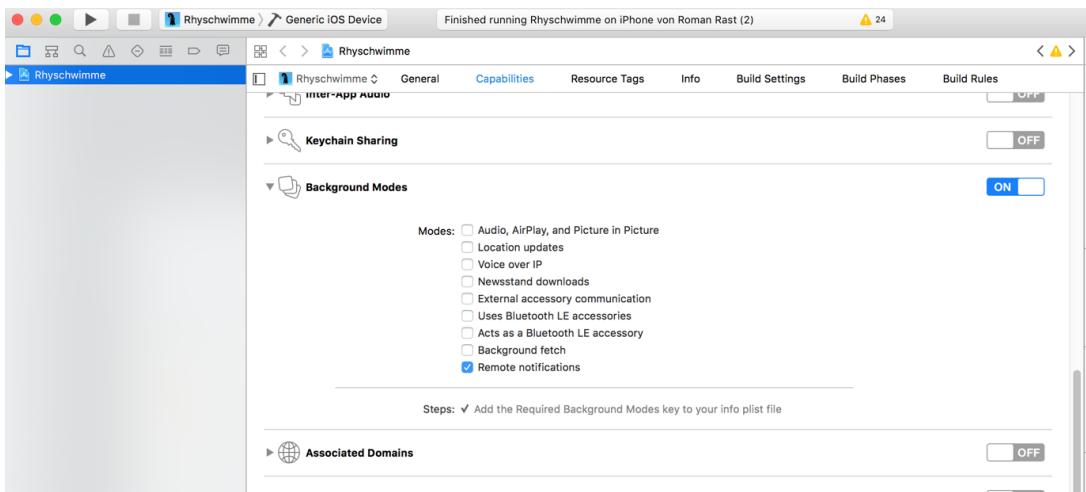
Ionic Workshop 17



Jetzt haben wir alle Zertifikate, die wir benötigen. Als nächster Schritt für Push Notifications, müssen wir in Xcode die Push Notification Funktion freischalten. Unter dem Capabilities Tab > Push Notifications auf ON setzen und auch Background Modes auf ON setzen, sowie Remote notifications markieren.



Ionic Workshop 17



Ionic bietet einen Service, damit wir bequem Push Notifications versenden und organisieren können <https://apps.ionic.io>. Falls du noch nicht registriert bist, wäre jetzt eine gute Möglichkeit dies zu tun!

iOS & Android: Upload deiner App zu ionic.io

Als nächstes können wir unser Ionic Projekt hochladen. In der Konsole, auf dem Root deines Ionic Projektes führe folgenden Befehl aus:

```
ionic upload
```

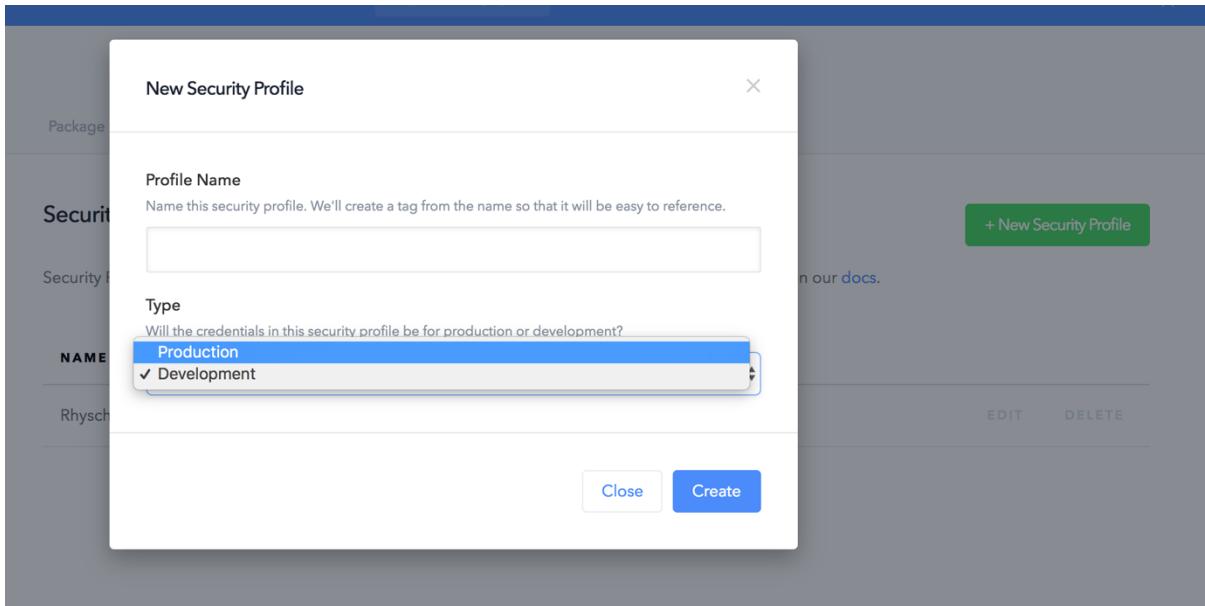
Gib dort deine Login Daten an und die App wird in dein Dashboard hochgeladen.

Nun können wir die Zertifikate einbinden um von Ionic.io Push Notifications zu versenden.

Auf dem Dashboard > Deine App > Settings > Certificates > new Security Profile klicken > Name des Profils angeben und wählen ob Dev oder Prod.

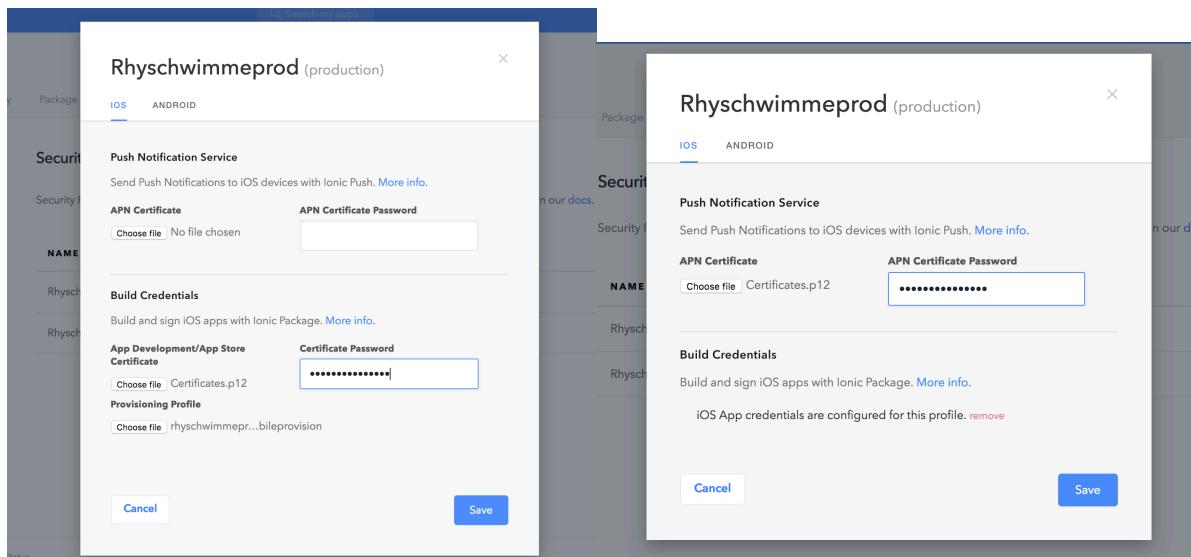
NAME	TAG	EDIT	DELETE
Rhyschwimme	rhyschwimme		

Ionic Workshop 17



Danach auf EDIT klicken. Unter iOS > Build Credentials > App Development > Choose File dein xxx.p12 Zertifikat hochladen und das zugehörige Passwort, welches du gesetzt hast, eingeben. Weiter noch das Provisioning Profile xxx.mobileprovision File auswählen und hochladen.

Als nächstes kannst du dein Push Certificate ebenfalls hier hochladen. Unter APN Certificate > Choose File dein erstelltes xxx.p12 Push Zertifikat hochladen und das zugehörige Passwort angeben.



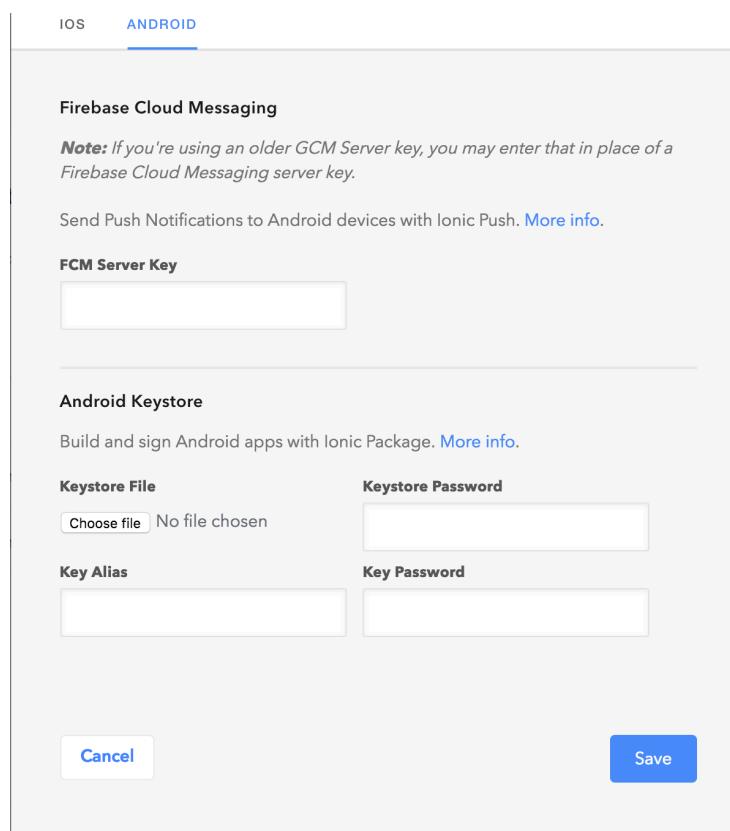
Ionic Workshop 17

Android: Android Keystore

Für Android brauchen wir einen Android Keystore um Apps zu signieren. Diesen kannst du in der Konsole im Root deiner App mit folgendem Befehl erstellen (ersetze dabei MY-RELEASE-KEY und MY_ALIAS_NAME mit z.B. dem Namen deiner App -> rhyschwimme-release-key). Du musst noch zum Keystore und Key ein Passwort angeben. Merke die Passwörter und den Alias, wir werden diese später noch brauchen.

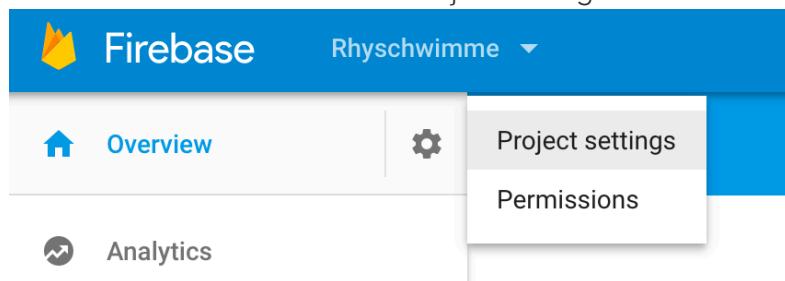
```
keytool -genkey -v -keystore MY-RELEASE-KEY.keystore -alias MY_ALIAS_NAME -keyalg RSA -keysize 2048 -validity 10000
```

Im Root Ordner deines Projektes erscheint nun eine xxx.keystore Datei. Diese Datei können wir auf <https://apps.ionic.io> unter deinem App > Settings > Certificates > EDIT > Android > Keystore File hochladen. Gib noch die Passwörter für Keystore und Key an sowie den Key Alias vom vorhergehenden Schritt.



Nun fehlt uns noch der FCM (Firebase Cloud Messaging) Server Key und ID. Diese kannst du auf Firebase erstellen <https://console.firebaseio.google.com/>. Erstelle hier ein neues Projekt mit dem Namen deiner App.

Klicke dann auf das Zahnrad neben Overview > Project Settings



Ionic Workshop 17

Hier findest du einen Tab mit dem Titel CLOUD MESSAGING und darin den FCM Server key und Sender ID. Den FCM Server key kannst du nun auf apps.ionic.io in deiner App > Settings > Certificates > EDIT > Android > FCM Server Key eingeben.

Ionic App iOS und Android: Install Push Plugin

Bald haben wir es! Als nächstes müssen wir in der App das Push Plugin implementieren, die Verknüpfung zu Ionic.io herstellen und die nötigen Einstellungen vornehmen.

Öffne die Konsole im Root deiner App und gebe folgende Befehle ein um Ionic Cloud (für die Verbindung zu ionic.io) und das Push Plugin zu installieren. Bei **SENDER_ID** verwende deine Sender ID von Firebase zu deiner App.

```
npm install @ionic/cloud-angular --save
cordova plugin add phonegap-plugin-push --variable SENDER_ID=12341234 --save
```

Öffne dein Ionic Projekt im Code Editor und öffne src/app/app.module.ts. Ergänze den Code, achte dabei darauf, die app_id deiner Ionic App in ionic.io zu verwenden. Ersetze auch die Sender ID mit der ID aus deinem Firebase Projekt.

```
import { CloudSettings, CloudModule } from '@ionic/cloud-angular';

const cloudSettings: CloudSettings = {
  'core': {
    'app_id': 'abcde123'
  },
  'push': {
    'sender_id': '1234567890',
    'pluginConfig': {
      'ios': {
        'badge': true,
        'sound': true
      },
      'android': {
        'iconColor': '#ffffff'
      }
    }
  }
};
```

...

```
imports: [
  IonicModule.forRoot(MyApp),
  CloudModule.forRoot(cloudSettings),
  BrowserModule,
  HttpModule
],
```

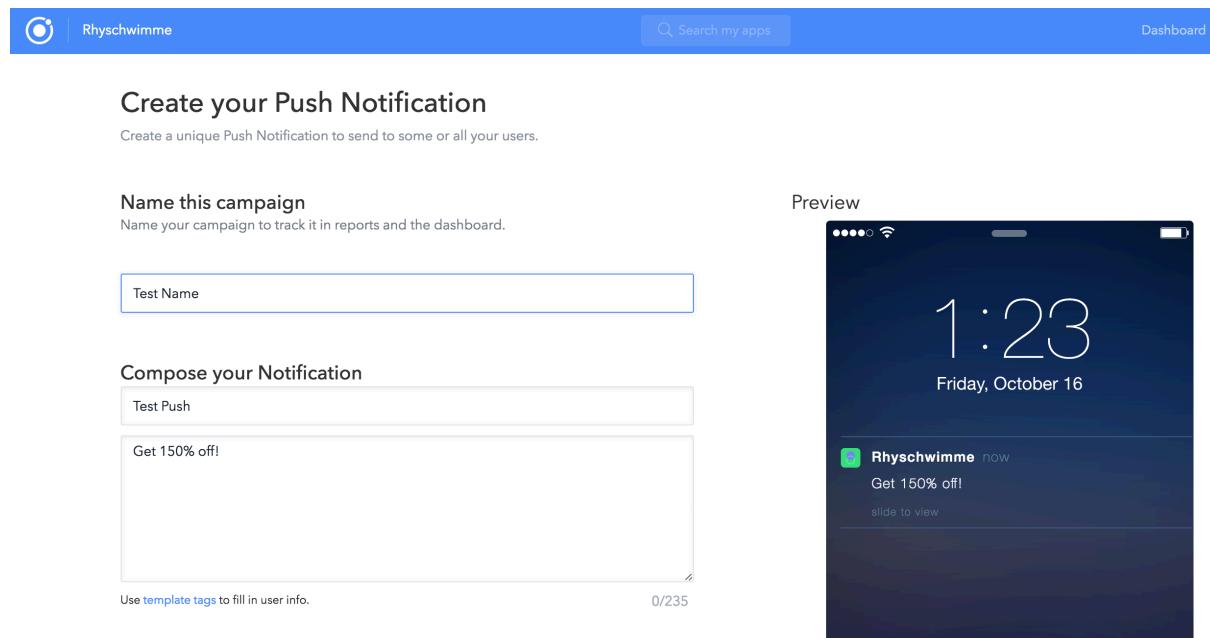
In src/app/app.component.ts können wir auf Push Notifications reagieren und registrieren.

```
import { Push, PushToken } from '@ionic/cloud-angular';
```

Ionic Workshop 17

```
constructor(public platform: Platform, public statusBar: StatusBar, public splashScreen: SplashScreen, public push: Push) {...  
...  
  initializeApp() {  
    this.platform.ready().then(() => {  
      ...  
      //pushnotification  
      this.push.register().then((t: PushToken) => {  
        return this.push.saveToken(t);  
      }).then((t: PushToken) => {  
        console.log('Token saved:', t.token);  
      });  
  
      //pushnotification  
      this.push.rx.notification()  
      .subscribe((msg) => {  
        console.log('I received awesome push: ' + msg);  
      });  
  
    });  
  }  
}
```

Jetzt ist die App bereit Push Notifications zu empfangen! Gehe dazu auf <https://apps.ionic.io/> und unter Push kannst du jetzt neue Push Notifications an die App senden!



The screenshot shows the Ionic Push Notifications dashboard. At the top, there's a navigation bar with a user icon, the name "Rhyschwimme", a search bar, and a "Dashboard" button. Below the navigation, the main area has a title "Create your Push Notification" and a subtitle "Create a unique Push Notification to send to some or all your users." There are two main sections: "Name this campaign" and "Compose your Notification". In the "Name this campaign" section, there's a text input field with "Test Name" typed into it. In the "Compose your Notification" section, there's a text input field with "Test Push" and another larger text area containing "Get 150% off!". Below these fields, there's a note "Use [template tags](#) to fill in user info." and a character count "0/235". To the right of these sections, there's a "Preview" section showing a smartphone screen with a dark blue background. The phone displays the time "1:23" and the date "Friday, October 16". A notification is shown from "Rhyschwimme" with the message "Get 150% off!" and a "slide to view" button.

DONE!

Ionic Workshop 17

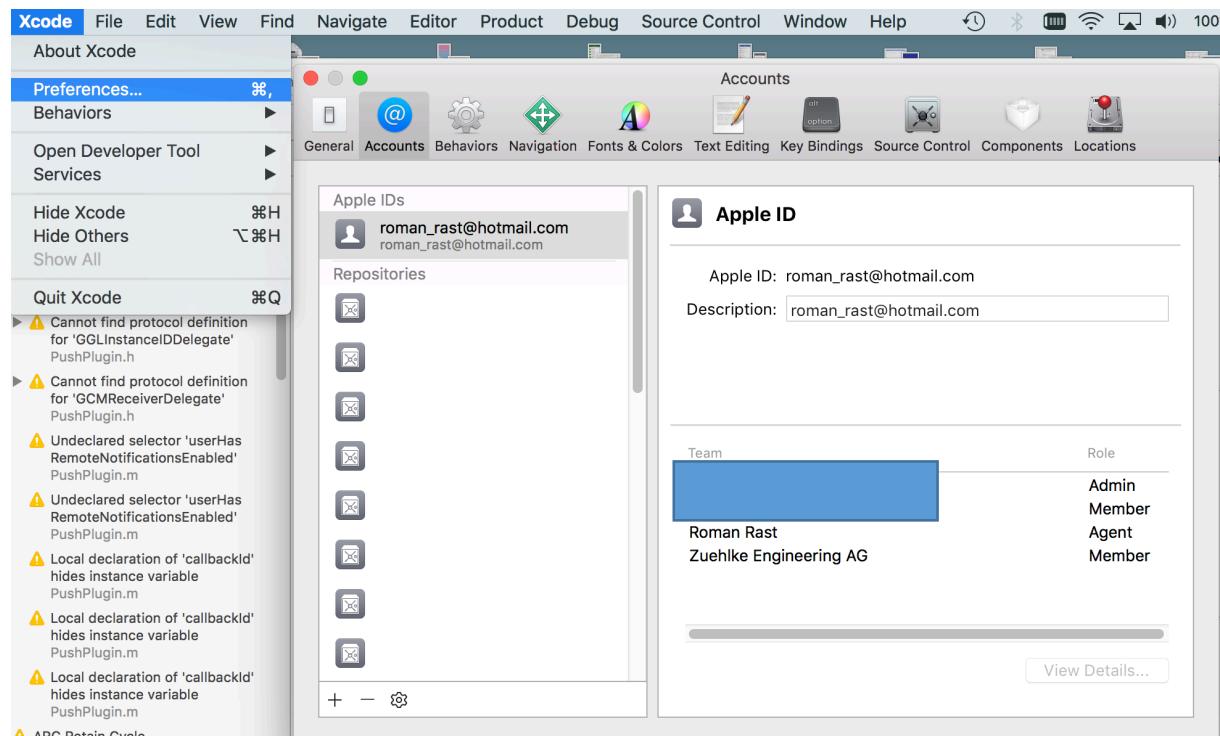
iOS und Android: Upload in die Stores

Um die App in die beiden Stores zu bringen, müssen wir einen Release Build der App erstellen. Als erstes entfernen wir das Console Plugin, welches wir für den Release nicht brauchen. Gib in der Konsole auf dem Root deines Projektes folgendes ein:

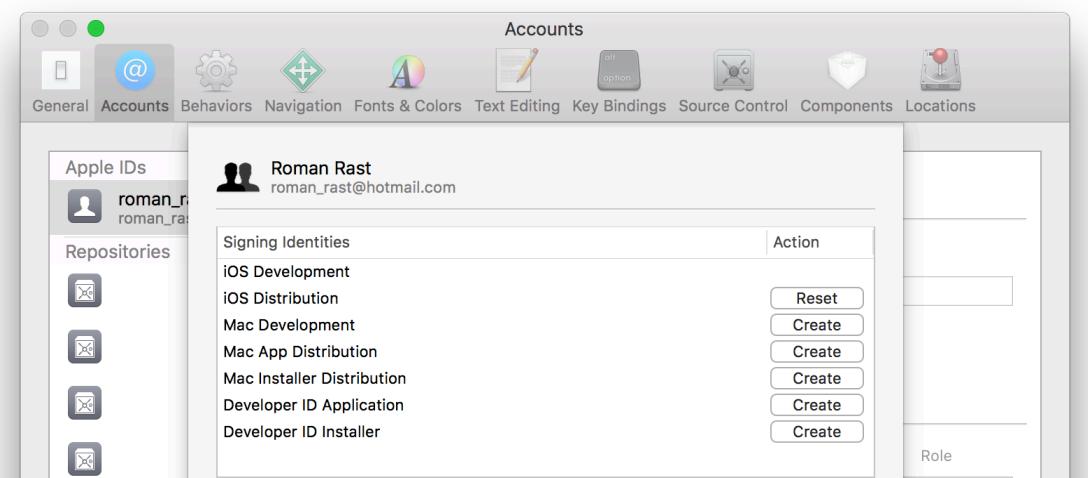
```
cordova plugin rm cordova-plugin-console
```

iOS: Release für App Store

Öffne Xcode und gehe unter Xcode > Preferences > Deine Apple ID > Dein Team > View Details



Danach unter Signing Identities klicke bei iOS Distribution auf Create



Ionic Workshop 17

Als nächster Schritt müssen wir in iTunes Connect einen Platz bzw. das Store Listing für unsere App anlegen. Hier werden wir auch den Build unserer App hochladen sowie die Screenshots und die Beschreibung der App.

Gehe dazu auf <https://itunesconnect.apple.com/login>

> Meine Apps > Klicke auf das Plus > Neue App

iTunes Connect Meine Apps ▾

Rhyschwimme Basel	Sticky Sticker	Hello Cat
Neue App	Neue Mac-App	Neues App-Bundle
iOS 1.0 Warten auf Prüfung..	iOS 1.0 Bereit zum Verkauf	iOS 1.2 Bereit zum Verkauf

Wähle iOS bei den Plattformen an und fülle die Angaben aus. Name -> Wie der App Name im Store angezeigt werden soll. Bundle-ID ist dieselbe, welche du in deinem Xcode Projekt findest. SKU ist eine eindeutige ID, setze hier einen Namen, dieser wird nicht im App Store angezeigt.

Neue App

Plattformen ?
 iOS tvOS

Name ?
FHNW APP

Primärsprache ?
Englisch (USA)

Bundle-ID ?
testapp - ch.romanrast.testappp

SKU ?
MY FHNW APP

Abbrechen Erstellen

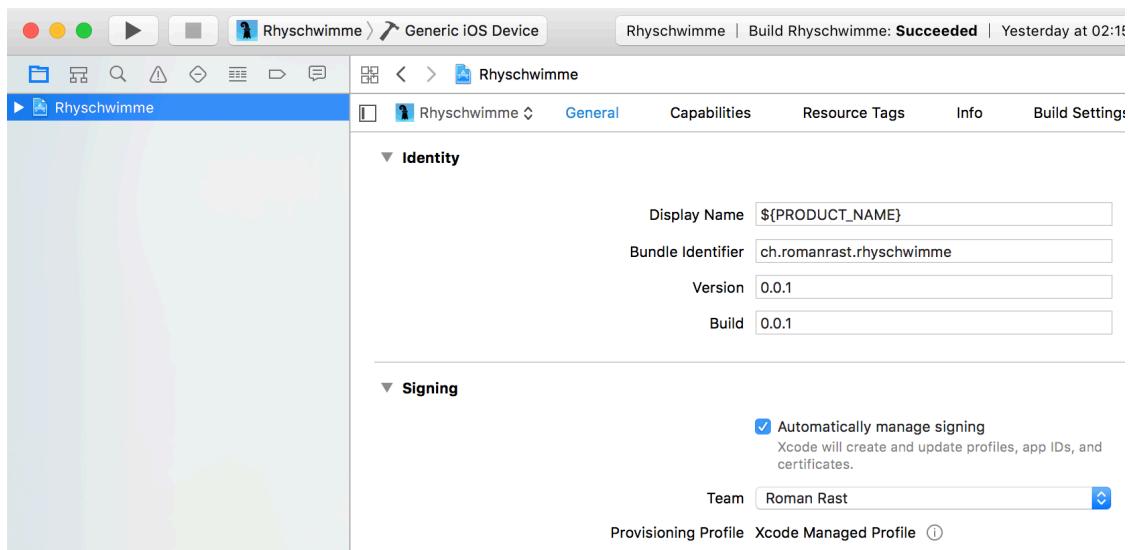
Nun solltest du auf den App-Informationen Screen kommen. Hier kannst du vorerst alle Informationen angeben.

Ionic Workshop 17

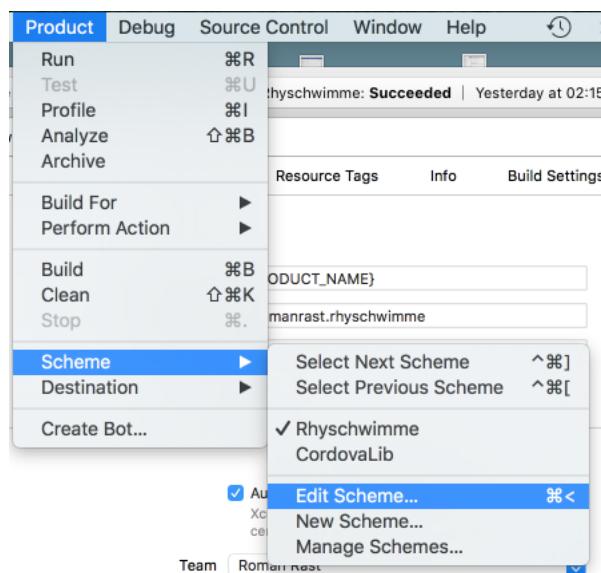
Als nächstes können wir jetzt den Build erstellen und in iTunes Connect hochladen. Öffne dazu dein Terminal im Root deiner App und gib folgendes ein:

```
ionic build ios --release
```

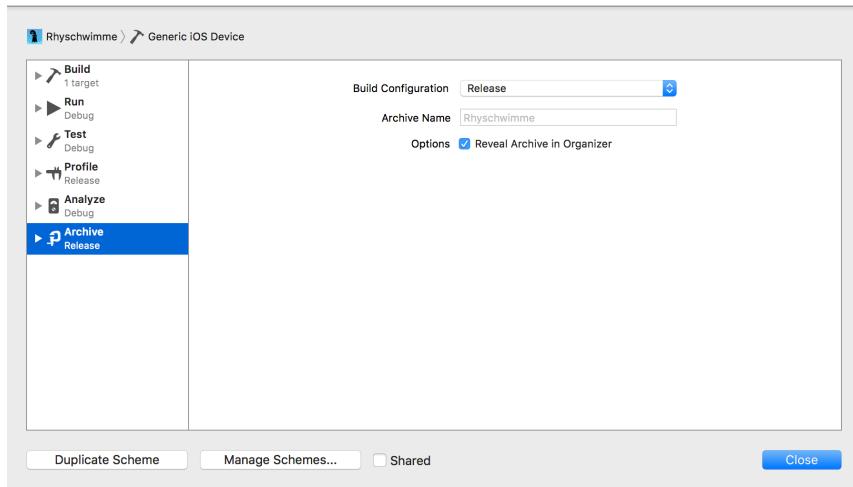
Der Build sollte mit BUILD SUCCEEDED durchlaufen. Öffne nun dein xxx.xcodeproj deiner App unter Deine App > platforms > ios
Vergewissere dich, dass alle Angaben unter General korrekt sind und Bundle Identifier mit der Bundle-ID in iTunes Connect übereinstimmen. Falls nicht, kannst du diese im config.xml deines Ionic Projektes noch anpassen und nochmals builden.



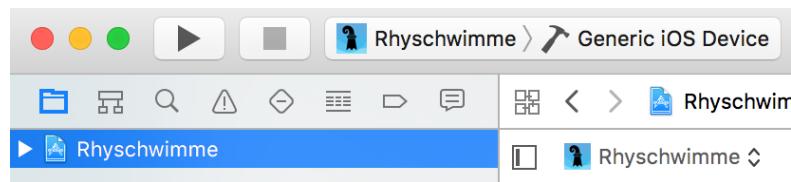
Unter Product > Scheme > Edit Scheme > Archive setze die Build Configuration auf Release



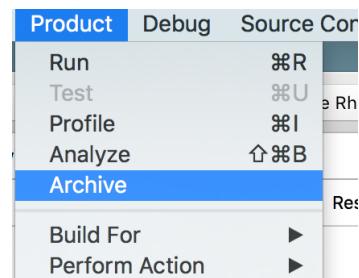
Ionic Workshop 17



Jetzt können wir das Archive erstellen. Schliesse das Fenster und setze im Hauptfenster die Auswahl des Devices auf **Generic iOS Device**.



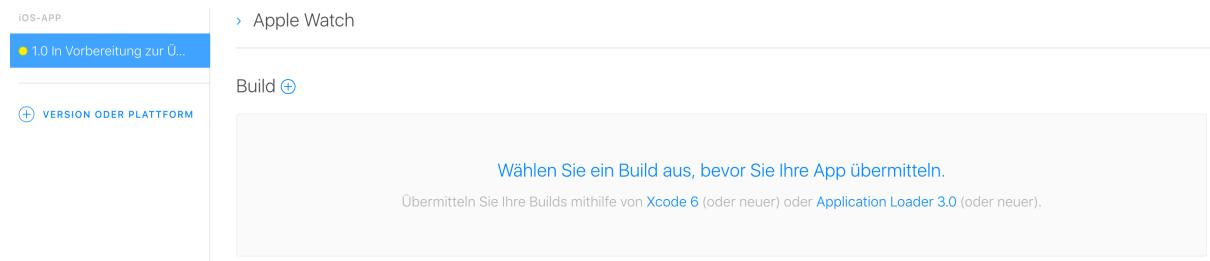
Danach auf Product > Archive > Wähle deine App aus > Upload to App Store



A screenshot of the Xcode Archives tab. The table lists two archives: 'Rhyschwimme' (created on 24 Apr 2017, 11:35) and another 'Rhyschwimme' (created on 23 Apr 2017, 00:20). The first archive is selected. To the right, under 'Archive Information', it shows the selected archive ('Rhyschwimme') and its creation date (24 Apr 2017, 11:35). A large blue button at the bottom right says 'Upload to App Store...'.

Ionic Workshop 17

Nun wieder zurück auf iTunes Connect, kannst du die restlichen Infos angeben wie Screenshots, App Icon, Search Keywords etc. Ebenfalls wirst du jetzt den Build auswählen können.



The screenshot shows the 'Build' section of an iOS app's page in iTunes Connect. On the left, there's a sidebar with 'iOS-APP' and a status bar indicating '1.0 In Vorbereitung zur ...'. Below that is a button labeled '+ VERSION ODER PLATTFORM'. The main area has a header 'Apple Watch' with a right-pointing arrow. Underneath is a 'Build +' button. A large callout box contains the text: 'Wählen Sie ein Build aus, bevor Sie Ihre App übermitteln.' followed by 'Übermitteln Sie Ihre Builds mithilfe von Xcode 6 (oder neuer) oder Application Loader 3.0 (oder neuer.)'.

Für Screenshots ist es am einfachsten den iOS Simulator als iPhone 7Plus (5.5") und als iPad Pro (12.9") zu starten. Mit cmd+s kannst du Screenshots erstellen. Diese werden auf deinem Desktop abgelegt. Die restlichen Screens für kleinere Devices werden runter gerechnet anhand der beiden Größen.

Nun gilt es nur noch abzuwarten, bis die App veröffentlicht ist. Üblicherweise geht die Prüfung folgende Schritte durch: Prepare For Review > Waiting For Review > In Review > Ready for Sale. Unter <http://appreviewtimes.com/> kannst du die durchschnittliche Dauer des Reviewprozesses anschauen (meistens im Bereich von zwei Tagen).

Ionic Workshop 17

Android: Release für Google Play Store

Für Android ist der Prozess einiges einfacher. In der Konsole im Root der App gib folgendes ein:

```
cordova build --release android
```

Dies generiert das unsignierte .apk File unter platforms/android/build/outputs/apk/android-release-unsigned.apk. Wir können nun dieses apk signieren mit:

```
jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore android-release-unsigned.apk alias_name
```

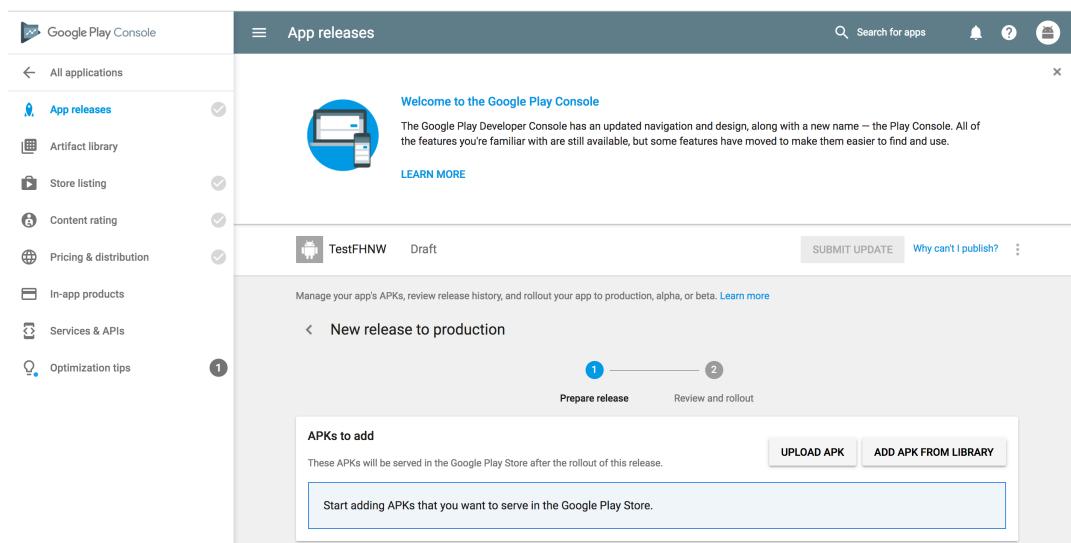
Ersetze dabei **My-RELEASE-KEY.keystore** mit dem Keystorenamen, welchen du schon gesetzt hast und gib das Passwort dazu ein.

Als nächstes können wir das apk optimieren mit dem zipalign Tool. Das zipalign Tool sollte sich auf Mac unter ~/Library/Android/sdk/build-tools/VERSION/zipalign befinden. Falls nicht, schau in Android Studio nach, ob die nötigen SDKs installiert sind > Tools > Android > SDK Manager > Launch Standalone SDK Manager > Android SDK Build-tools die nötigen Versionen installieren. Danach zipalign in den richtigen Pfad kopieren, z.B. /usr/local/Cellar/android-sdk/24.1.2/build-tools/21.1.2/zipalign.

Nun kannst du folgenden Befehl ausführen:

```
zipalign -v 4 android-release-unsigned.apk Rhyschwimme.apk
```

Jetzt können wir auf der Google Play Console <https://play.google.com/apps> unsere App hochladen > Create Application > App releases > Manage Production > Upload APK.



Die restlichen Infos ausfüllen und deine App ist in wenigen Stunden im Store!

