AND220

Publishing an Android application

Download class materials from
university.xamarin.com

Microsoft

Xamarin University

Objectives

1. Configure you app for publishing
2. Create a Google Play developer account
3. Create an app package
4. Submit your application for publishing

Configure you app for publishing

Xamarin University

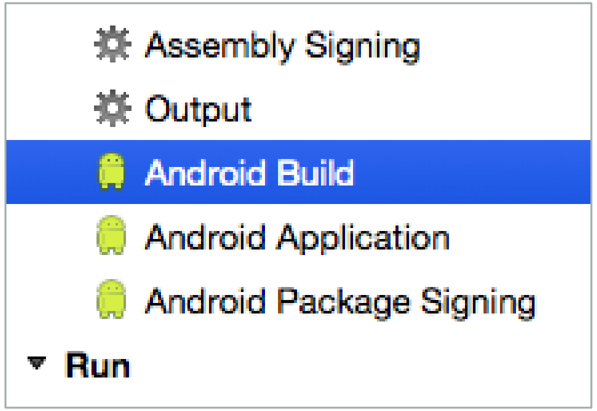
Tasks

1. Optimize the release build settings
2. Verify the app details
3. Set a version number

Setting Android app properties

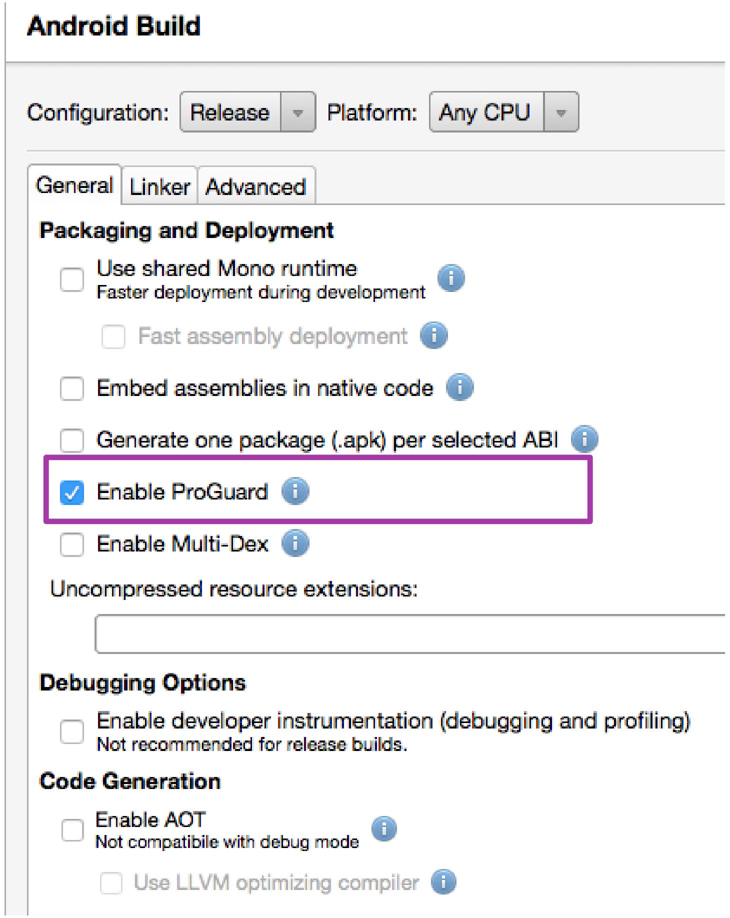
- Android has several unique optimizations for:
  - Packaging
  - Code generation
  - Device compatibility

- All available from the project properties: **Android Build**



Packaging properties

- **ProGuard** is an Android SDK tool that will shrink the Java byte-code by removing unused methods
- This can be helpful to shrink your application if you rely on native 3rd party components such as Google Play
- Only supported for **Release** builds



Android Build

Configuration: Release Platform: Any CPU

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [ ] Generate one package (.apk) per selected ABI
- [✓] Enable ProGuard
- [ ] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

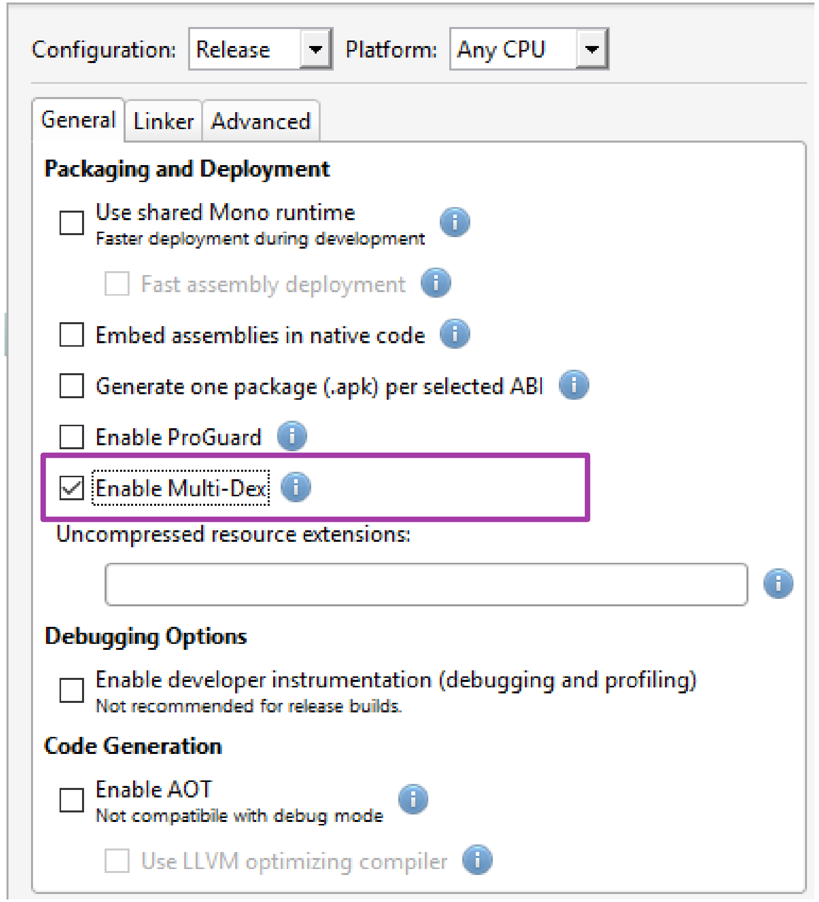
Code Generation

- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

Xamarin University

Packaging properties

- Dalvik executable file (**.dex**) format is limited to 65k Java method references internally
- ProGuard dramatically lifts this restriction by removing unused methods, but apps could still hit limit if they use a lot of components
- If you hit this limit, turn on Multi-Dex to break into multiple files



Configuration: Release Platform: Any CPU

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [ ] Generate one package (.apk) per selected ABI
- [ ] Enable ProGuard
- [✓] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

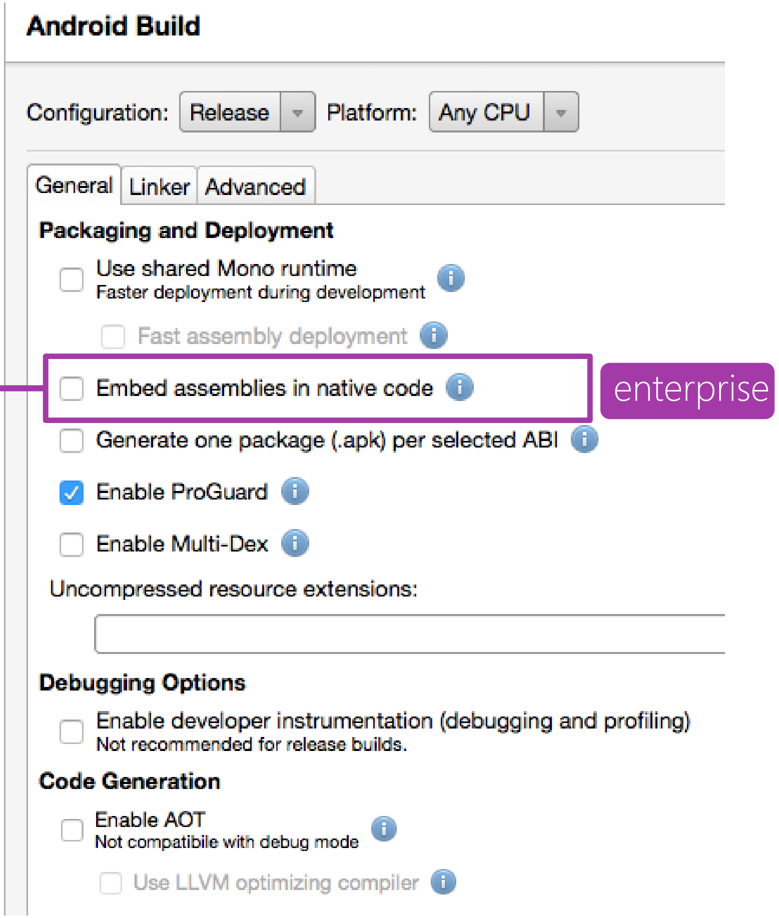
Code Generation

- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

Packaging properties

❖ Some features require higher license

Can package your IL-based assemblies into a native Android library, makes them (slightly) harder to disassemble


Android Build

Configuration: Release Platform: Any CPU

General | Linker | Advanced

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [ ] Generate one package (.apk) per selected ABI
- [x] Enable ProGuard
- [ ] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

Code Generation

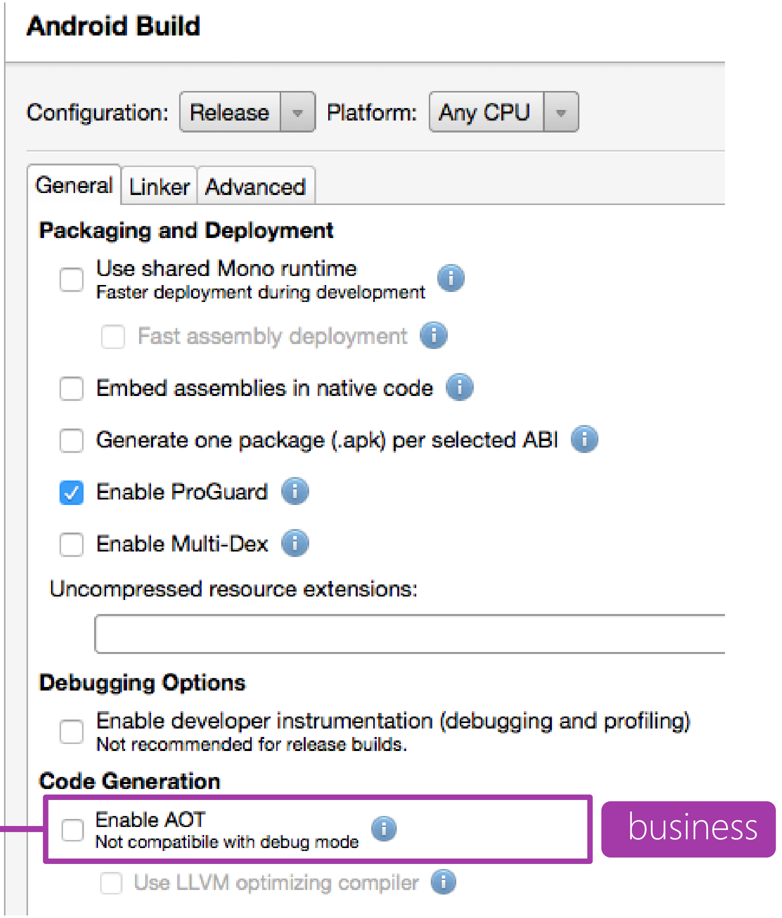
- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

enterprise

Packaging properties

- Some features require higher license

Xamarin.Android supports a pre-JIT code generation option (**AOT**) – this is a <u>preview</u> feature



Android Build

Configuration: Release Platform: Any CPU

General | Linker | Advanced

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [ ] Generate one package (.apk) per selected ABI
- [✓] Enable ProGuard
- [ ] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

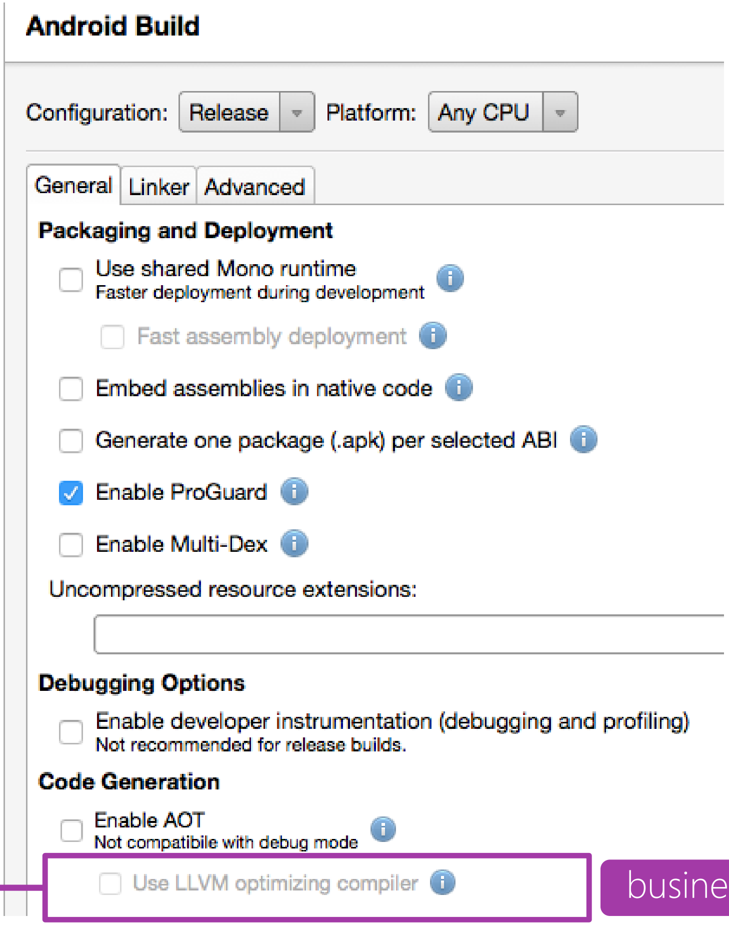
- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

Code Generation

- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

Packaging properties

❖ Some features require higher license



Android Build

Configuration: Release Platform: Any CPU

General | Linker | Advanced

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [ ] Generate one package (.apk) per selected ABI
- [✓] Enable ProGuard
- [ ] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

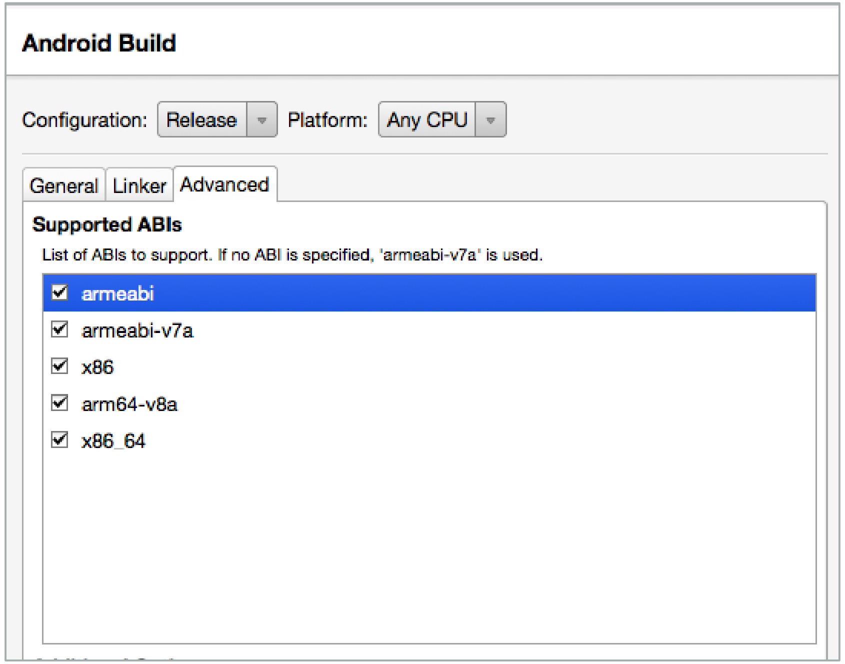
Code Generation

- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

business

LLVM compiler produces smaller, faster executable code – requires AOT be enabled

Select Android target platforms



Android Build

Configuration: Release Platform: Any CPU

Supported ABIs

List of ABIs to support. If no ABI is specified, 'armeabi-v7a' is used.

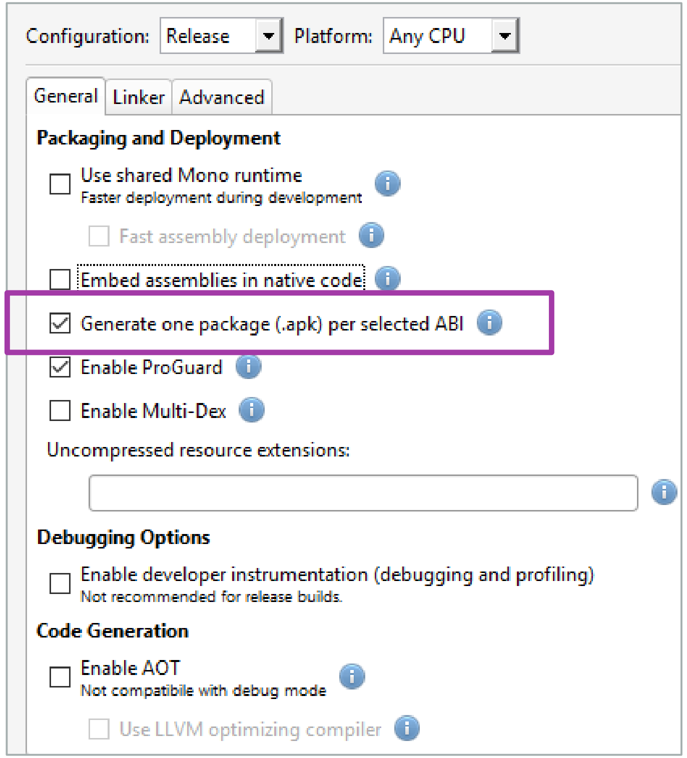
- [✓] armeabi
- [✓] armeabi-v7a
- [✓] x86
- [✓] arm64-v8a
- [✓] x86_64

- Android devices utilize a variety of different CPU architectures
- Must include the Application Binary Interface (ABI) layer for each CPU architecture your app can support
- Each ABI adds to the size of the final application

Xamarin University

Generating one APK package per ABI

- To reduce the app package but still support a variety of CPUs, you can tell Visual Studio for Mac to generate a different app package (APK) for each ABI
- Default behavior is to generate a single (larger) package with all ABIs packaged together


Configuration: Release Platform: Any CPU

General | Linker | Advanced

Packaging and Deployment

- [ ] Use shared Mono runtime
  Faster deployment during development
  - [ ] Fast assembly deployment
- [ ] Embed assemblies in native code
- [✓] Generate one package (.apk) per selected ABI
- [✓] Enable ProGuard
- [ ] Enable Multi-Dex

Uncompressed resource extensions:

Debugging Options

- [ ] Enable developer instrumentation (debugging and profiling)
  Not recommended for release builds.

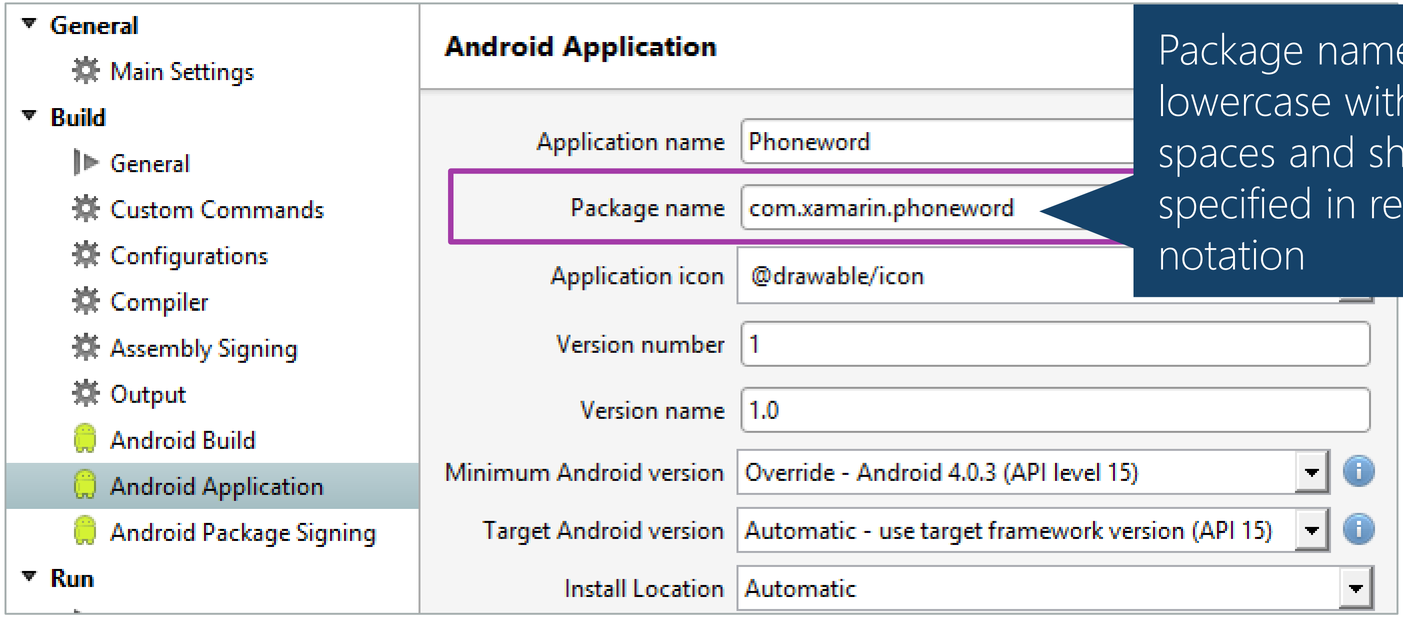
Code Generation

- [ ] Enable AOT
  Not compatibile with debug mode
  - [ ] Use LLVM optimizing compiler

Xamarin University

Identifying the application

❖ Make sure the package name is unique and identifies the publisher



- General
  - Main Settings
- Build
  - General
  - Custom Commands
  - Configurations
  - Compiler
  - Assembly Signing
  - Output
  - Android Build
  - Android Application
  - Android Package Signing
- Run

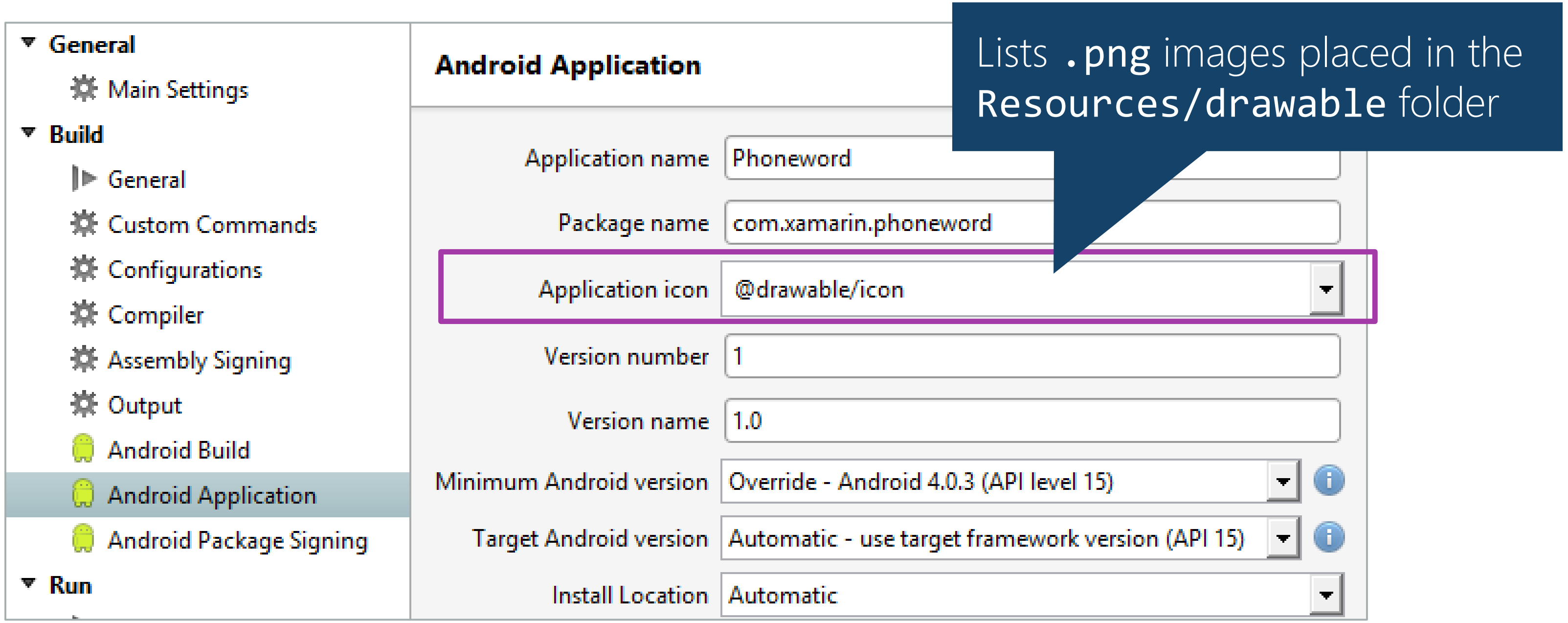
Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Package name must be lowercase with no spaces and should be specified in reverse-DNS notation

Specify the Application Icon

- Make sure to provide a high-quality icon in multiple resolutions



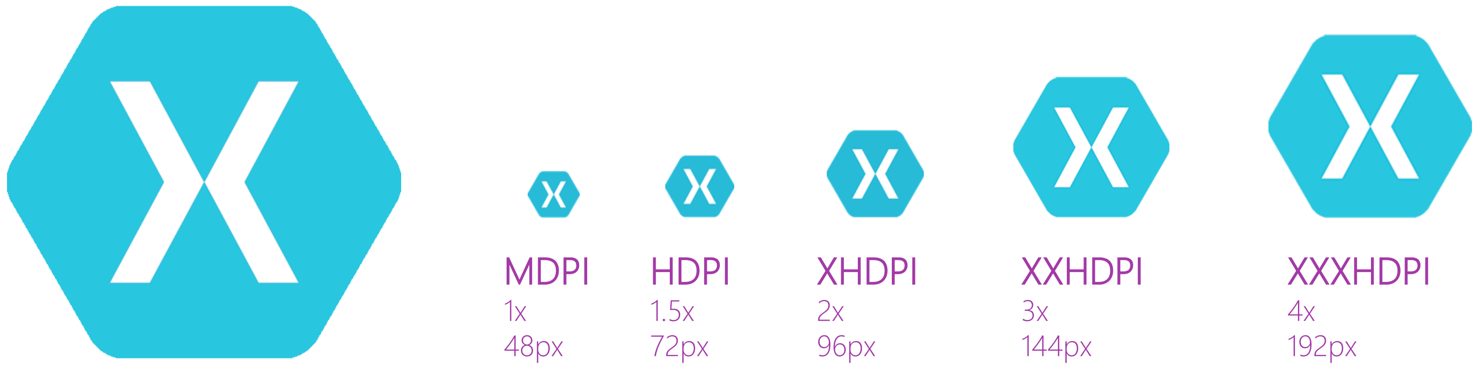
Lists `.png` images placed in the `Resources/drawable` folder

Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Image and Icon Sizes

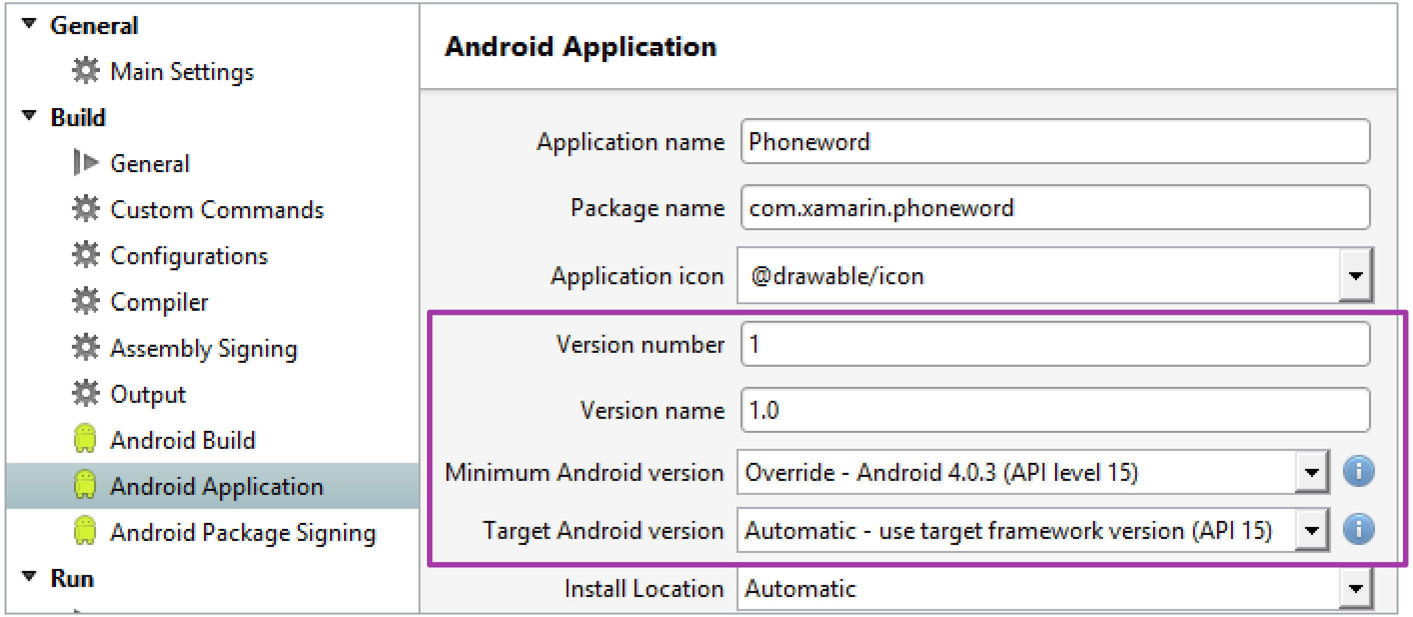
- Ensure images and icons are provided in multiple resolutions in your **drawables** folder – use the naming qualifiers and Android will select the proper image at runtime; 3rd party tools can help create multiple versions



Xamarin University

Dealing with versioning

❖ Several versioning boxes in the properties to deal with both app version and supported Android versions



- General
  - Main Settings
- Build
  - General
  - Custom Commands
  - Configurations
  - Compiler
  - Assembly Signing
  - Output
  - Android Build
  - Android Application
  - Android Package Signing
- Run

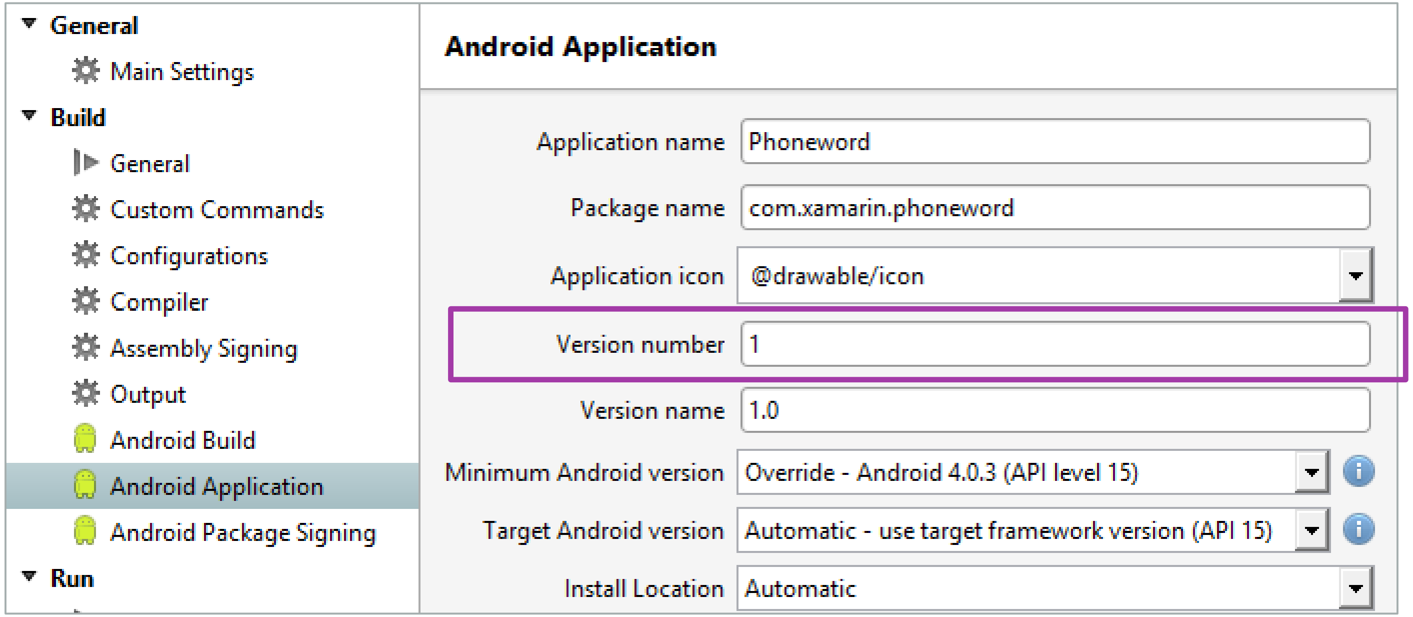
Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Xamarin University

What is the Version number?

- App versioning utilizes the numeric version to determine when an update should applied – increment this value on every release



- General
  - Main Settings
- Build
  - General
  - Custom Commands
  - Configurations
  - Compiler
  - Assembly Signing
  - Output
  - Android Build
  - Android Application
  - Android Package Signing
- Run

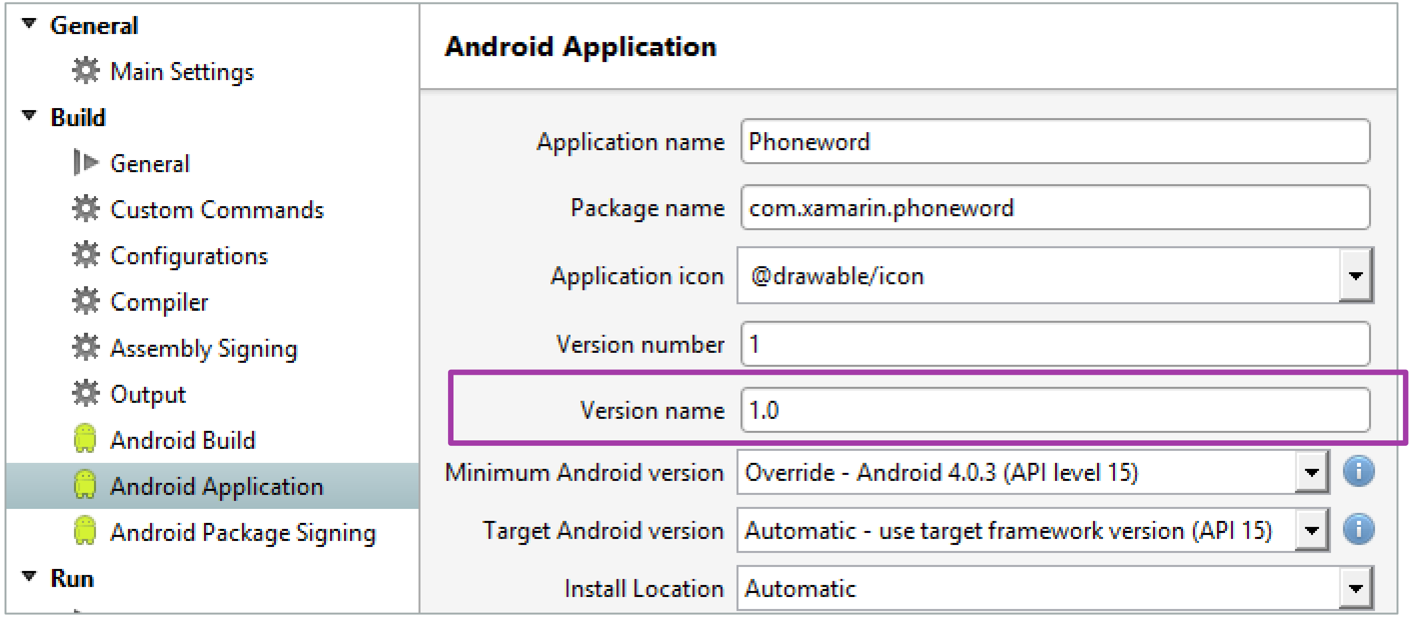
Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Xamarin University

What is the Version name?

- Version name is the alphanumeric text that is displayed to the user to identify the version, should use the **Major.Minor.Revision** format



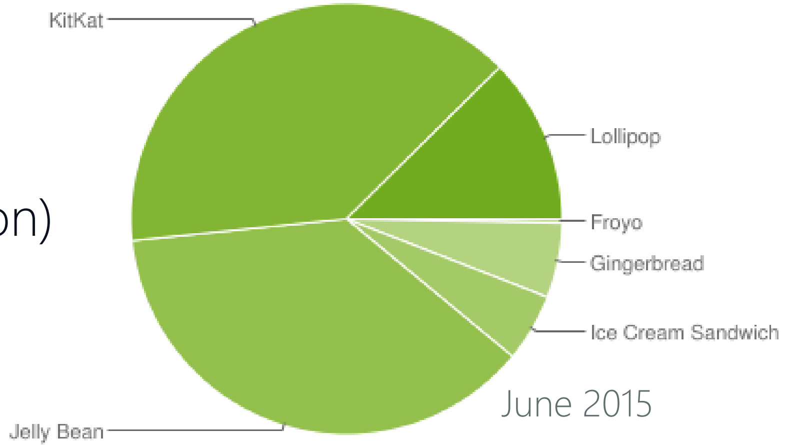
Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Xamarin University

Reminder: Android API levels

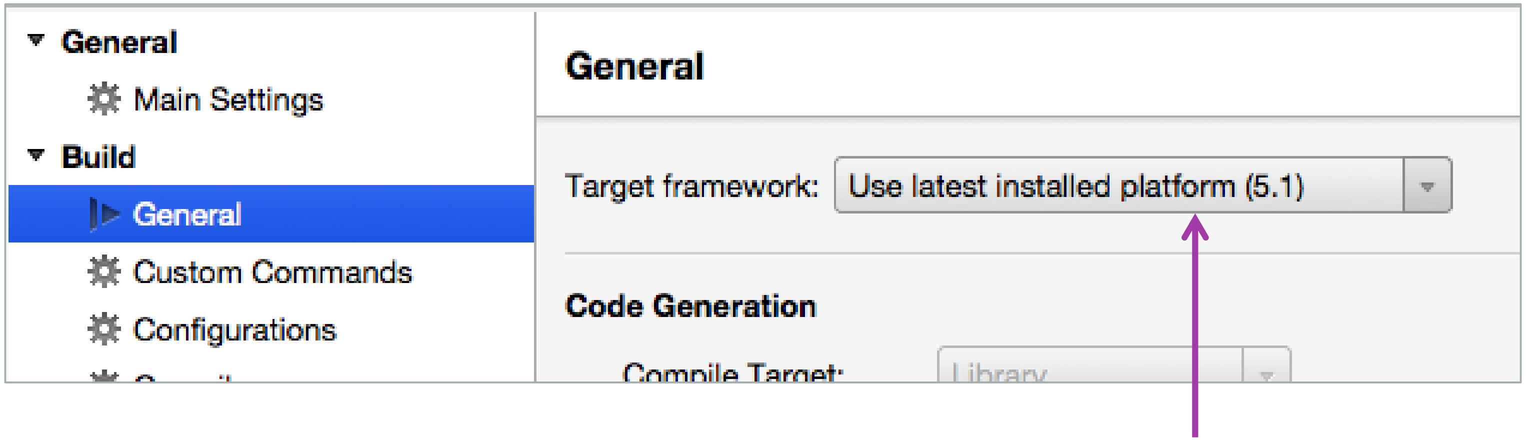
- Three project level settings control the version of Android that your app:
  - **Builds against** (Target Framework)
  - **Supports** (Min. Android Version)
  - **Runs best on** (Target Android Version)



June 2015

Target Framework

- Target Framework setting controls which libraries the *compiler* uses – this decides the features you can use in your code

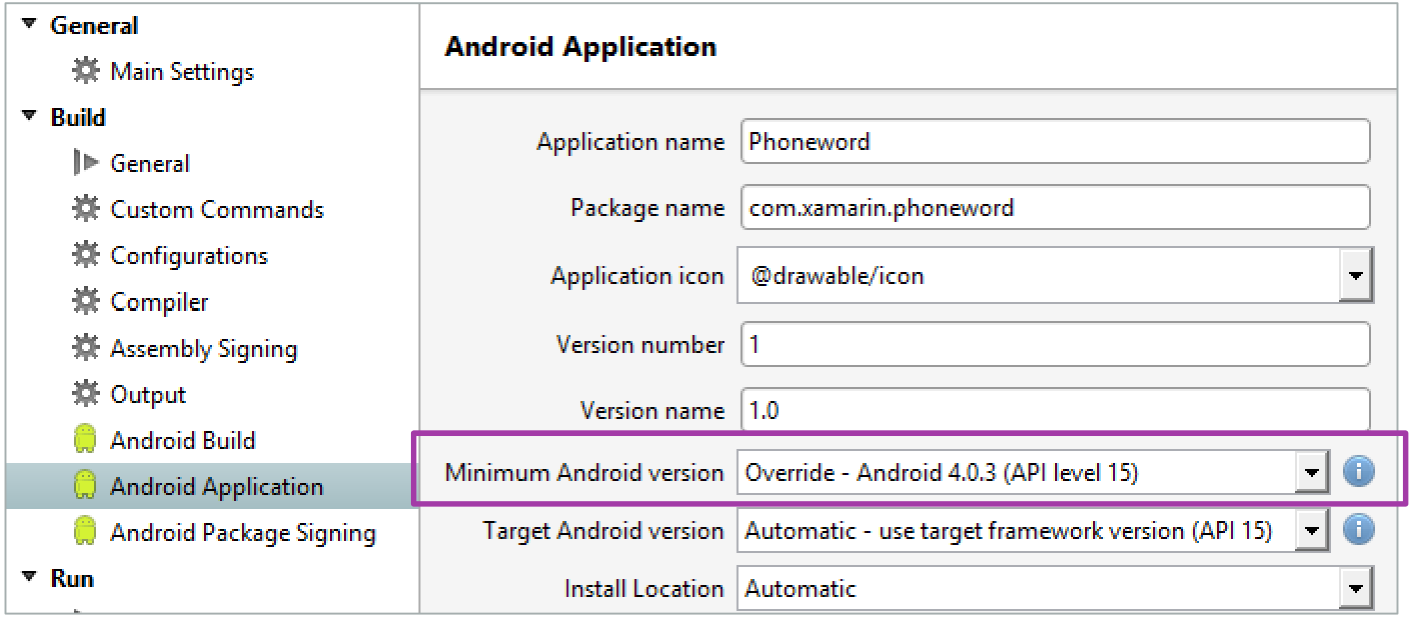


Target framework: Use latest installed platform (5.1)

Should be set to the latest released framework

Minimum Android version

- Set your minimum version to be the lowest possible version of Android you want your app to run on – this is used at *runtime*



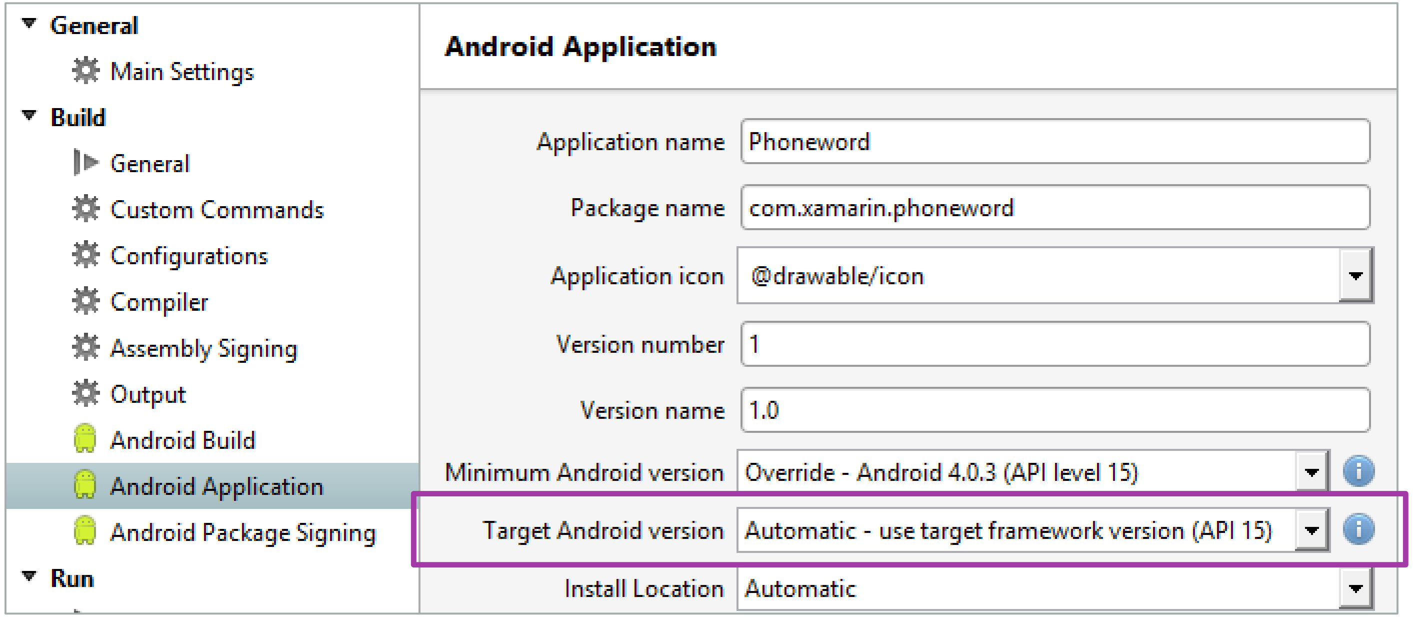
Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Xamarin University

Target Android version

❖ Set the target version to be the Android version your app is intended to run on – this is used at *runtime*



Android Application

Application name	Phoneword
Package name	com.xamarin.phoneword
Application icon	@drawable/icon
Version number	1
Version name	1.0
Minimum Android version	Override - Android 4.0.3 (API level 15)
Target Android version	Automatic - use target framework version (API 15)
Install Location	Automatic

Summary

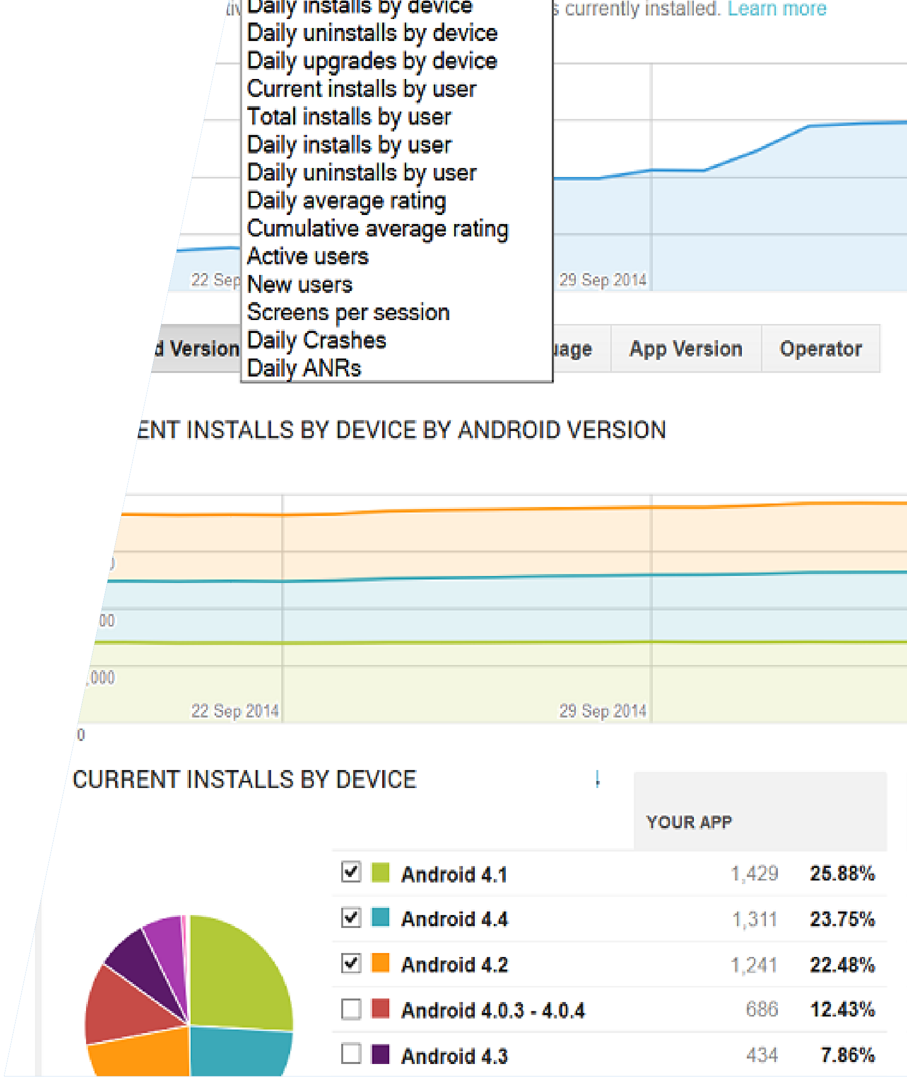
1. Optimize the release build settings
2. Verify the app details
3. Set a version number

Create a Google Play developer account

Xamarin University

Tasks

1. Sign up for a Play account
2. Use the Developer Console



s currently installed. Learn more

Daily installs by device
Daily uninstalls by device
Daily upgrades by device
Current installs by user
Total installs by user
Daily installs by user
Daily uninstalls by user
Daily average rating
Cumulative average rating
Active users
New users
Screens per session
Daily Crashes
Daily ANRs

ENT INSTALLS BY DEVICE BY ANDROID VERSION

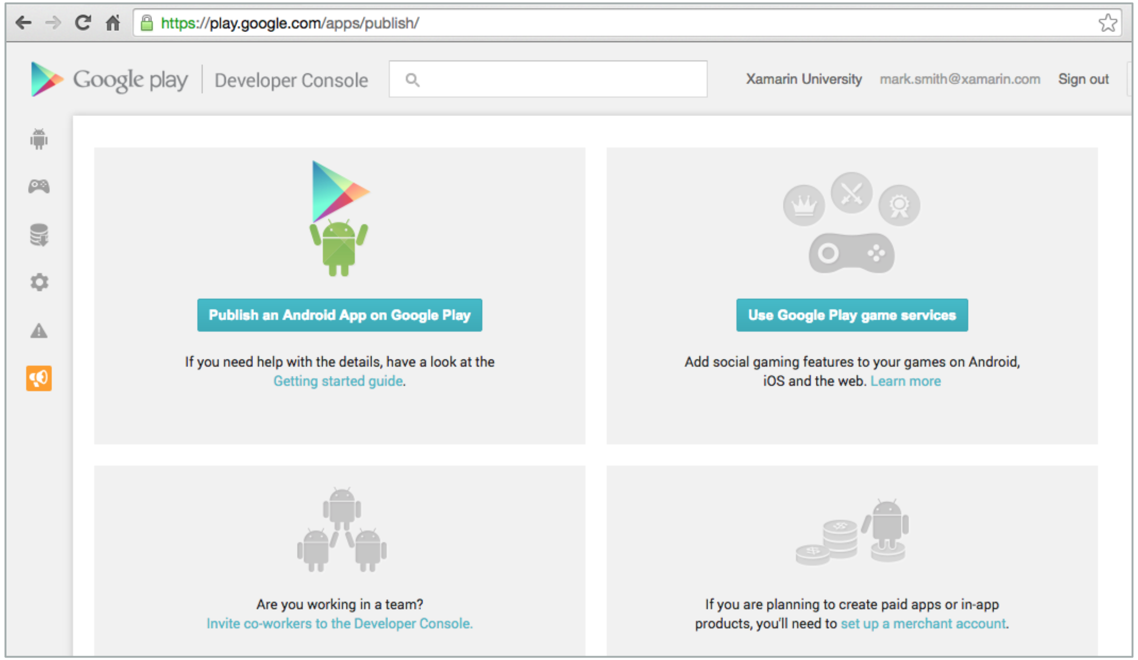
CURRENT INSTALLS BY DEVICE

		YOUR APP	
✓	🟩 Android 4.1	1,429	25.88%
✓	🟦 Android 4.4	1,311	23.75%
✓	🟧 Android 4.2	1,241	22.48%
	🟥 Android 4.0.3 - 4.0.4	686	12.43%
	🟪 Android 4.3	434	7.86%

Xamarin University

Google Play Developer Console

- Developer Console is the home for app publishing on Google Play



https://play.google.com/apps/publish/

Google play | Developer Console

Xamarin University mark.smith@xamarin.com Sign out

Publish an Android App on Google Play

If you need help with the details, have a look at the Getting started guide.

Use Google Play game services

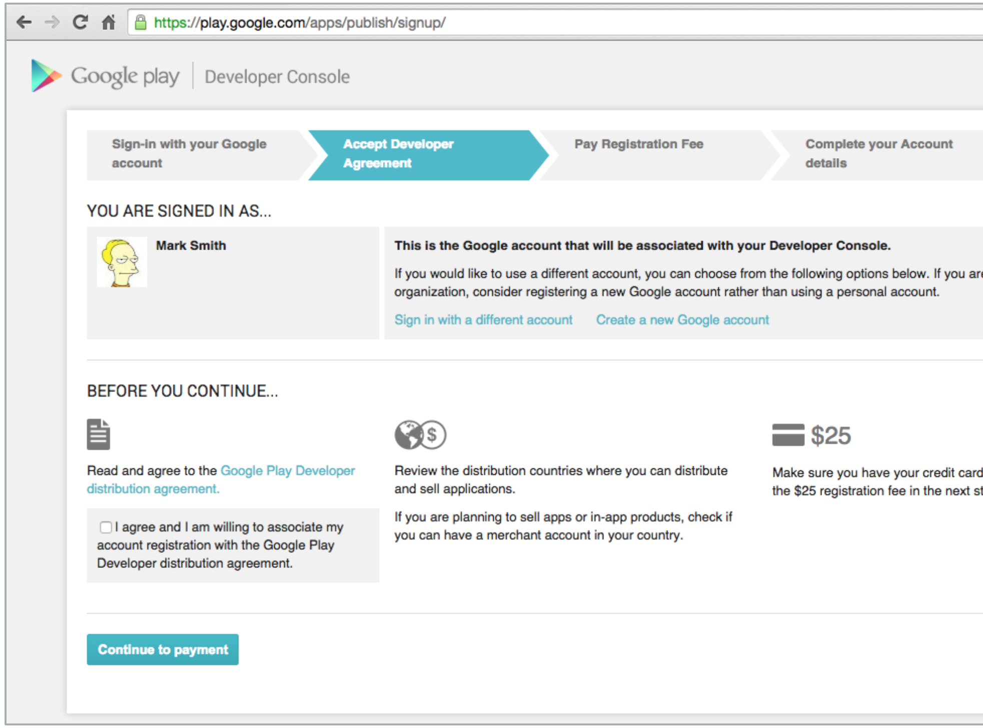
Add social gaming features to your games on Android, iOS and the web. Learn more

Are you working in a team? Invite co-workers to the Developer Console.

If you are planning to create paid apps or in-app products, you'll need to set up a merchant account.

Registering for a publisher account

- Must register with Google Play Developer Console using a Google account (can create a new one)
- Enter basic information about identity and accept the licensing agreement for your region
- Pay $25 USD one-time fee


https://play.google.com/apps/publish/signup/
Google play | Developer Console
Sign-in with your Google account
Accept Developer Agreement
Pay Registration Fee
Complete your Account details
Continue to payment

YOU ARE SIGNED IN AS...

Mark Smith

This is the Google account that will be associated with your Developer Console.

If you would like to use a different account, you can choose from the following options below. If you are
organization, consider registering a new Google account rather than using a personal account.

Sign in with a different account   Create a new Google account

BEFORE YOU CONTINUE...

Read and agree to the Google Play Developer distribution agreement.

☐ I agree and I am willing to associate my account registration with the Google Play Developer distribution agreement.

Review the distribution countries where you can distribute and sell applications.

If you are planning to sell apps or in-app products, check if you can have a merchant account in your country.

$25

Make sure you have your credit card
the $25 registration fee in the next st

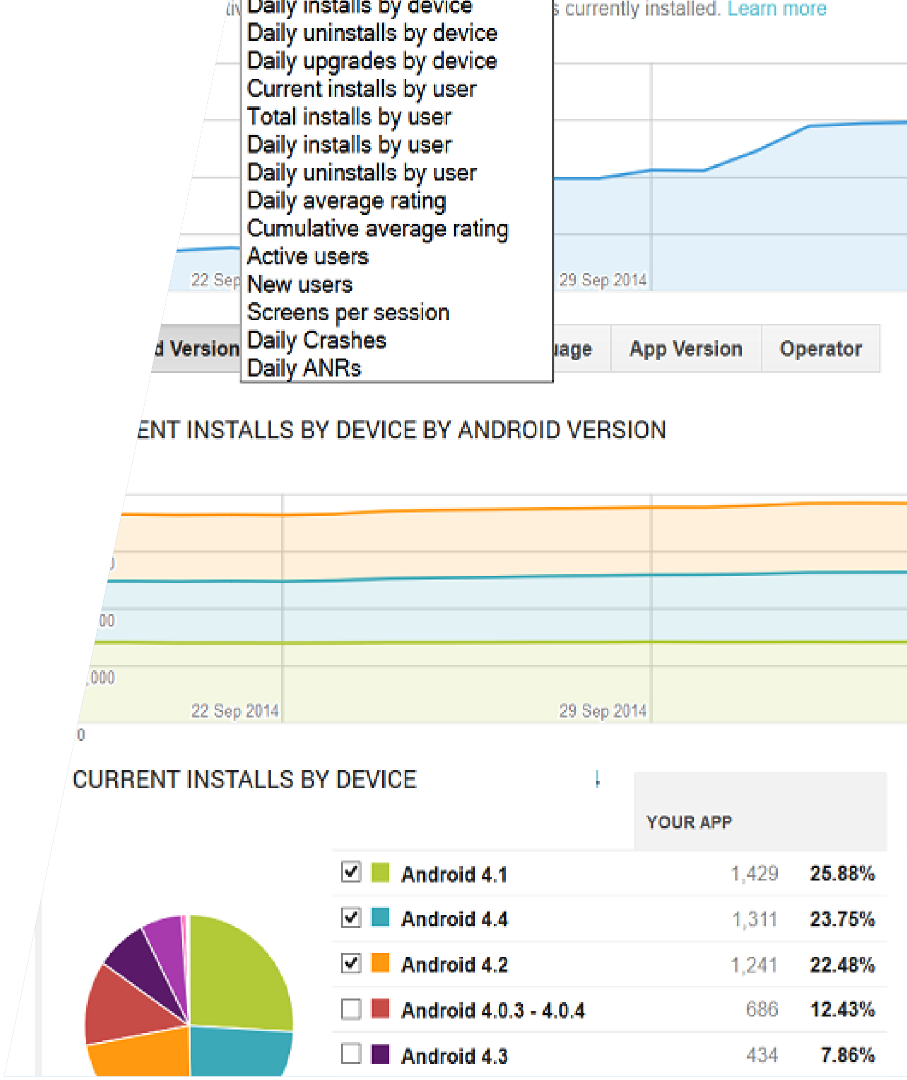
Getting paid

- To sell apps, you must signup for a Google Payment Merchant account
  - Payments issued in your local currency via bank transfer
  - Pays just after month end with no minimum (currently)
  - 70/30 split

- Reports > Financial Reports > Setup a merchant account

Summary

1. Sign up for a Play account
2. Use the Developer Console



s currently installed. Learn more

Daily installs by device
Daily uninstalls by device
Daily upgrades by device
Current installs by user
Total installs by user
Daily installs by user
Daily uninstalls by user
Daily average rating
Cumulative average rating
Active users
New users
Screens per session
Daily Crashes
Daily ANRs

ENT INSTALLS BY DEVICE BY ANDROID VERSION

CURRENT INSTALLS BY DEVICE

			YOUR APP	
✓	🟩	Android 4.1	1,429	25.88%
✓	🟦	Android 4.4	1,311	23.75%
✓	🟧	Android 4.2	1,241	22.48%
☐	🟥	Android 4.0.3 - 4.0.4	686	12.43%
☐	🟪	Android 4.3	434	7.86%

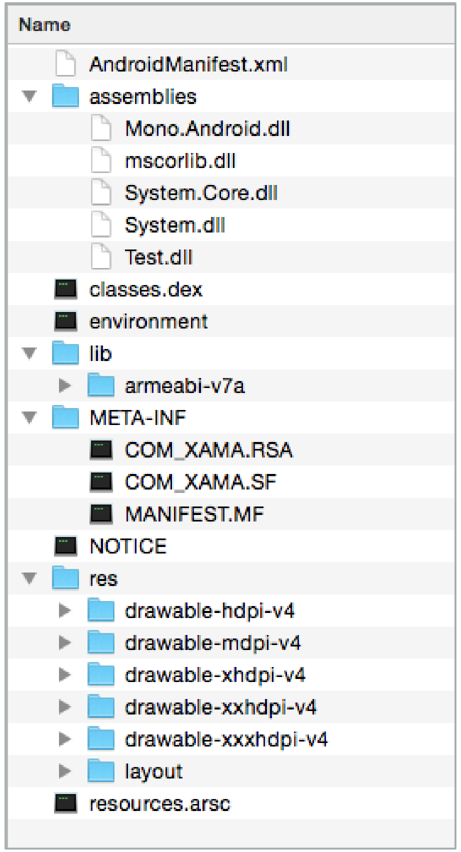
Create an app package

Tasks

1. Create an APK using Visual Studio
2. Sign your application using a keystore

Reminder: what is an APK?

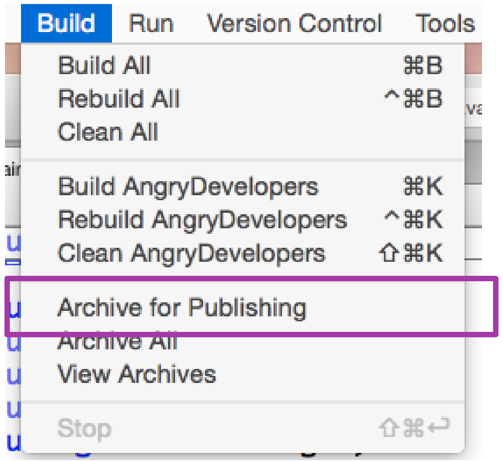
- Android applications are packaged as a zip file with the extension **.apk** (Android Package)
- Contains your IL code, runtime, framework assemblies, native .dex files, manifest, resources, ABIs and any components the app uses
- APKs can be downloaded and installed directly onto devices (more on this later)

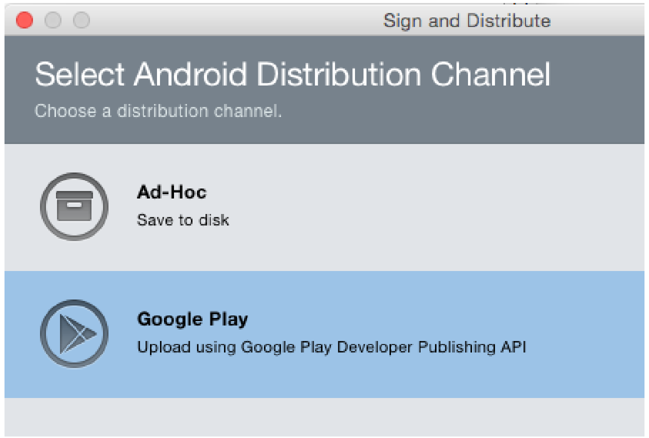


```
Name
    AndroidManifest.xml
▼   assemblies
        Mono.Android.dll
        mscorlib.dll
        System.Core.dll
        System.dll
        Test.dll
    classes.dex
    environment
▼   lib
    ▶   armeabi-v7a
▼   META-INF
        COM_XAMA.RSA
        COM_XAMA.SF
        MANIFEST.MF
    NOTICE
▼   res
    ▶   drawable-hdpi-v4
    ▶   drawable-mdpi-v4
    ▶   drawable-xhdpi-v4
    ▶   drawable-xxhdpi-v4
    ▶   drawable-xxxhdpi-v4
    ▶   layout
    resources.arsc
```

Creating the APK [Mac]

- **Build > Archive for Publishing** then **Sign and Distribute** will create the APK and either save it to your local machine, or upload it to Google





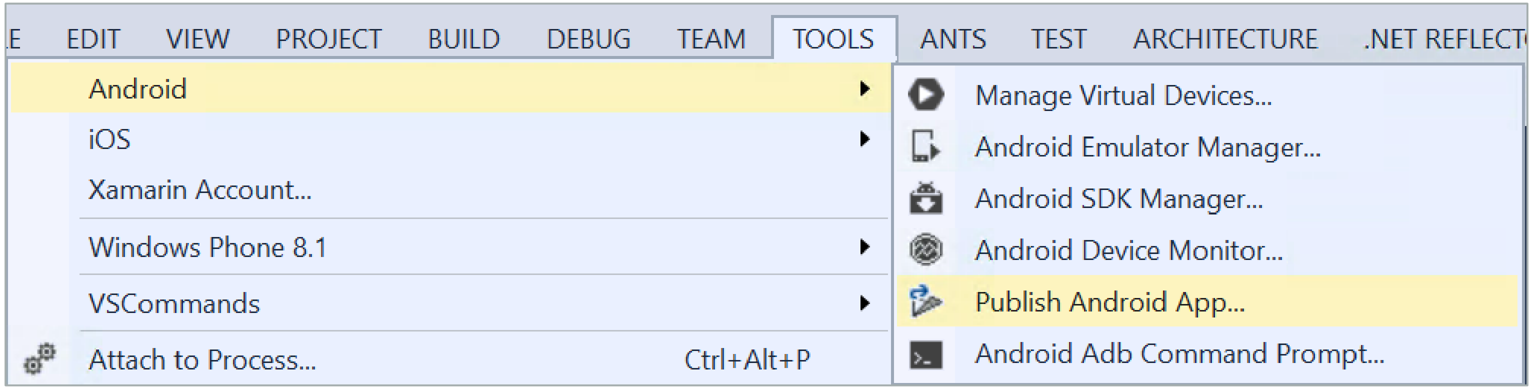
Sign and Distribute
Select Android Distribution Channel
Choose a distribution channel.
Ad-Hoc
Save to disk
Google Play
Upload using Google Play Developer Publishing API

Note: when the APK is built for release, the system will include the selected ABIs for the target Android version; you must have the correct Android SDK components installed!

Xamarin University

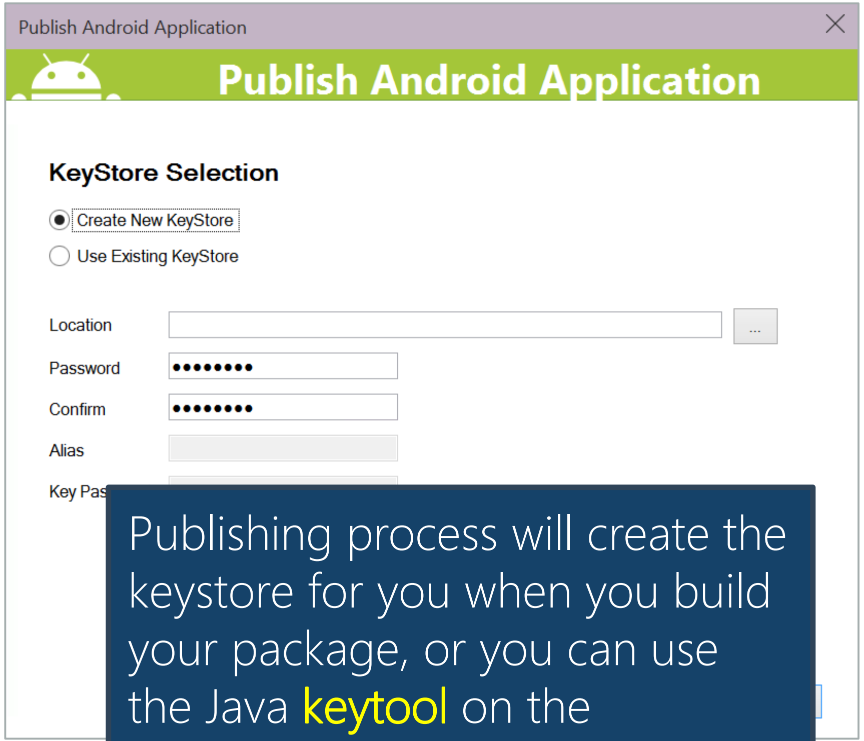
Creating the APK [Windows]

- Can create the APK in release-build configuration with **Tools > Android > Publish Android App**, this just saves the file to your local machine


E EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS ANTS TEST ARCHITECTURE .NET REFLECT
Android
iOS
Xamarin Account...
Windows Phone 8.1
VSCommands
Attach to Process... Ctrl+Alt+P
Manage Virtual Devices...
Android Emulator Manager...
Android SDK Manager...
Android Device Monitor...
Publish Android App...
Android Adb Command Prompt...

What is a Keystore?

- The **keystore** is a certificate key pair used to sign your application and identify you as the publisher
- Typically will use the same release keystore for all your apps
- Should save off the release keystore, signing key is considered part of the app identity



Publish Android Application

KeyStore Selection

(●) Create New KeyStore
( ) Use Existing KeyStore

Location	
Password	••••••••
Confirm	••••••••
Alias	
Key Pas	

Publishing process will create the keystore for you when you build your package, or you can use the Java keytool on the command line

Summary

1. Create an APK using Visual Studio
2. Sign your application using a keystore

Submit your application for publishing

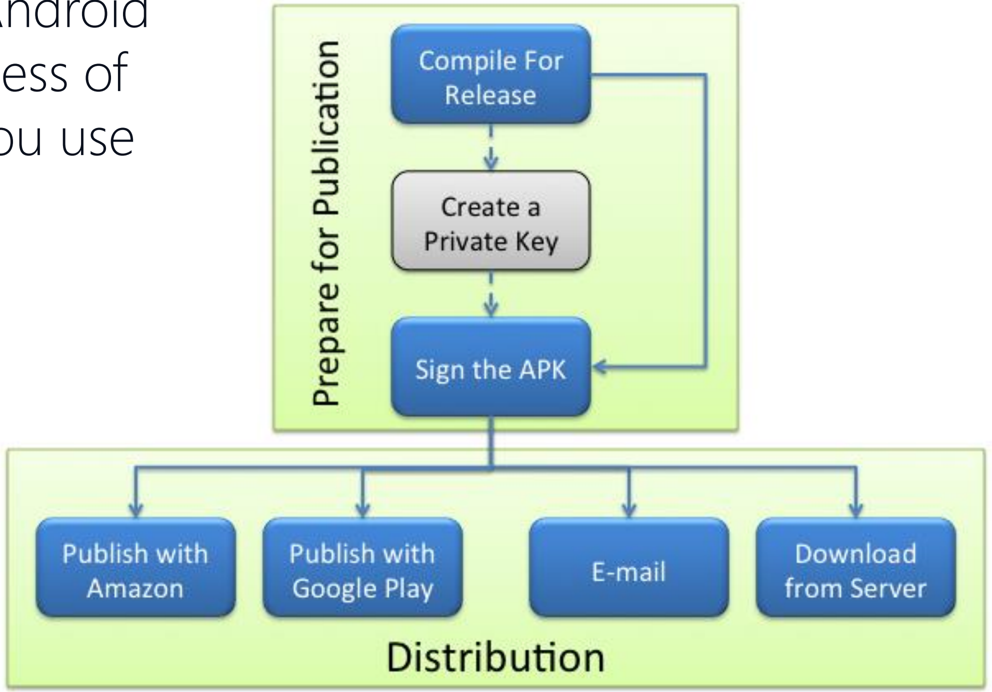
Xamarin University

Tasks

1. Side-load an application
2. Submit your application to the Google Play Store
3. Submit your application to the Amazon app store

Publishing an Android app

- The steps used to publish an Android app is always the same regardless of the distribution method that you use



Android Deployment styles

- Android is quite flexible in terms of distributing applications (APK)

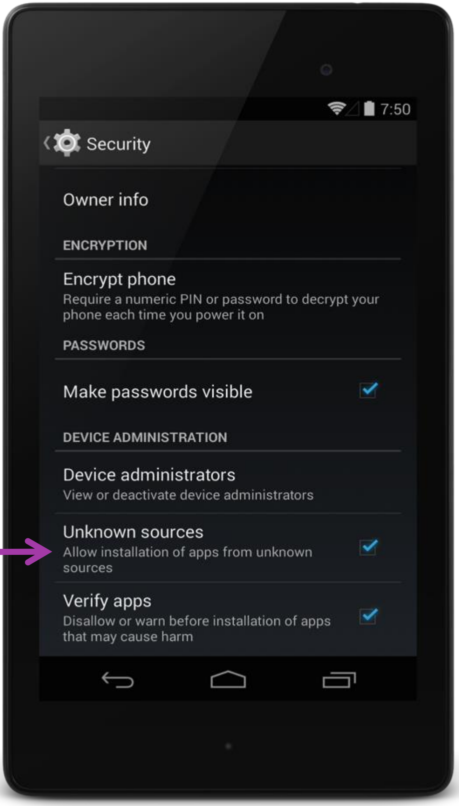
Side-loading

Store publishing

Side-loading

- The signed APK can be installed onto any Android device using the device browser, one of several GUI tools or **adb**

The device must allow "unknown publishers" in Settings > Security



7:50

Security

Owner info

ENCRYPTION

Encrypt phone
Require a numeric PIN or password to decrypt your phone each time you power it on

PASSWORDS

Make passwords visible ✓

DEVICE ADMINISTRATION

Device administrators
View or deactivate device administrators

Unknown sources ✓
Allow installation of apps from unknown sources

Verify apps ✓
Disallow or warn before installation of apps that may cause harm

Store publishing

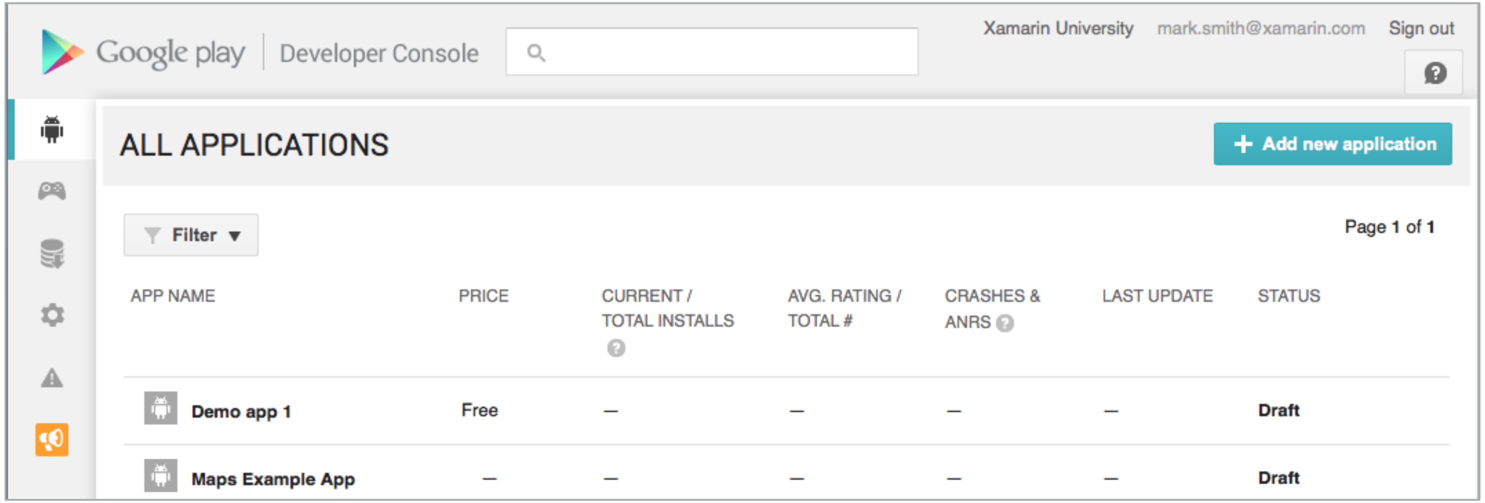
- Android has many stores you can publish your apps to, consider publishing to at least Google Play and Amazon for the broadest distribution and look at other online venues for more specialized audiences

Google play

amazon apps Available at amazon

Google Play

- Google Play store is the official distribution host from Google



Google play | Developer Console

Xamarin University mark.smith@xamarin.com Sign out

ALL APPLICATIONS

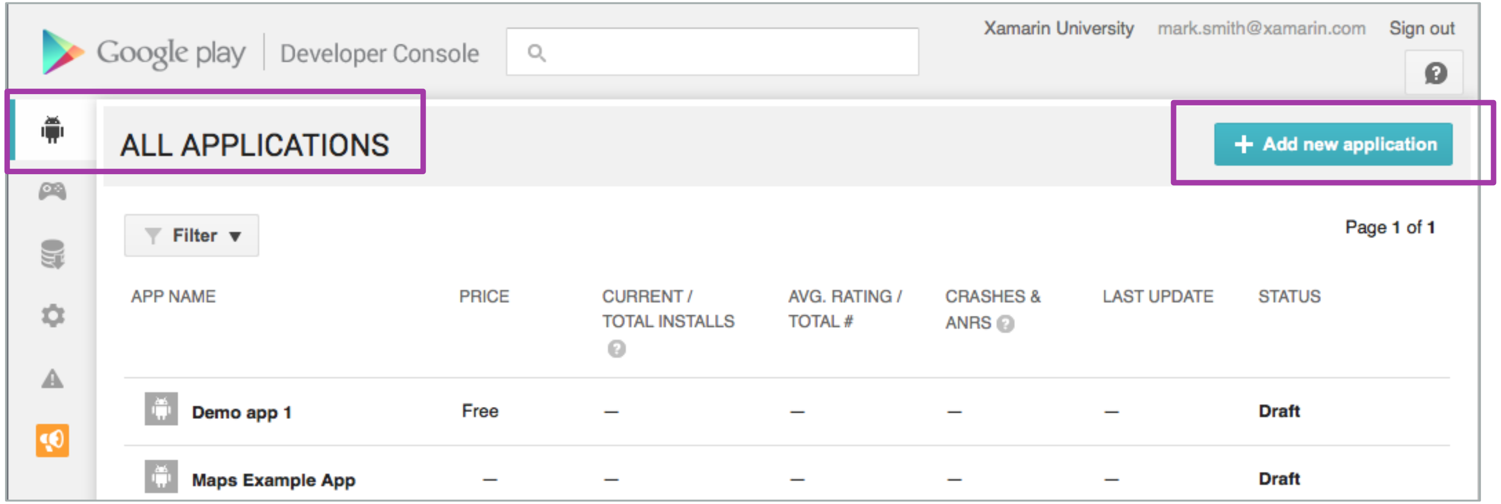
APP NAME	PRICE	CURRENT / TOTAL INSTALLS	AVG. RATING / TOTAL #	CRASHES & ANRS	LAST UPDATE	STATUS
Demo app 1	Free	—	—	—	—	Draft
Maps Example App	—	—	—	—	—	Draft

play.google.com/apps/publish/

Xamarin University

Creating a new app in Google Play

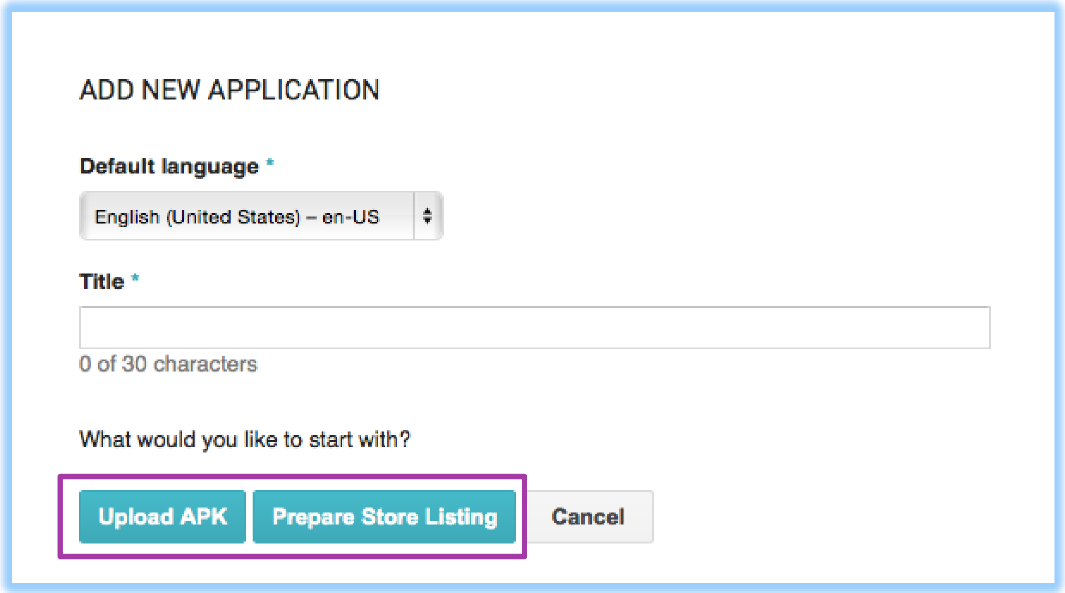
❖ Simple process to add a new app into the store


Google play | Developer Console
Xamarin University mark.smith@xamarin.com Sign out
ALL APPLICATIONS
+ Add new application
Filter ▼
Page 1 of 1

APP NAME	PRICE	CURRENT / TOTAL INSTALLS	AVG. RATING / TOTAL #	CRASHES & ANRS	LAST UPDATE	STATUS
Demo app 1	Free	—	—	—	—	Draft
Maps Example App	—	—	—	—	—	Draft

Creating a new app in Google Play

- Simple process to add a new app into the store



ADD NEW APPLICATION

Default language *

English (United States) – en-US

Title *

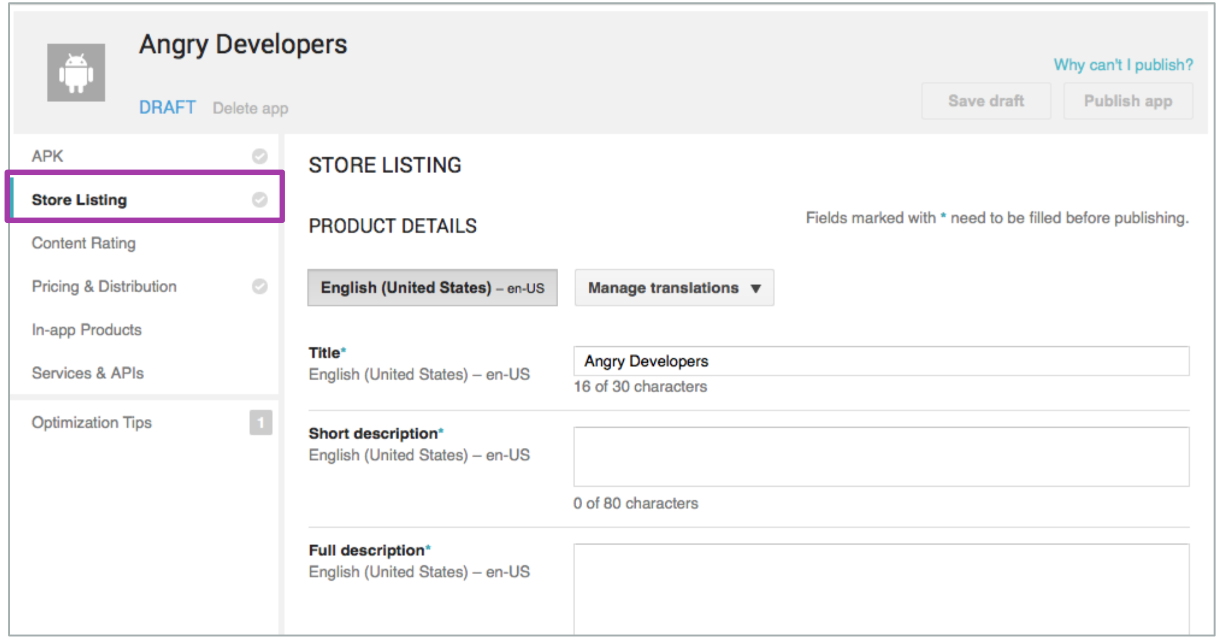
0 of 30 characters

What would you like to start with?

Xamarin University

Fill in the application info

- Title
- Description
- Price
- Phone Screenshots
- Tablet Screenshots
- High-Res Icon
- Category Information
- Contact Details
- Rating Information
- Available Countries
- …



STORE LISTING

PRODUCT DETAILS

Fields marked with * need to be filled before publishing.

English (United States) – en-US | Manage translations ▼

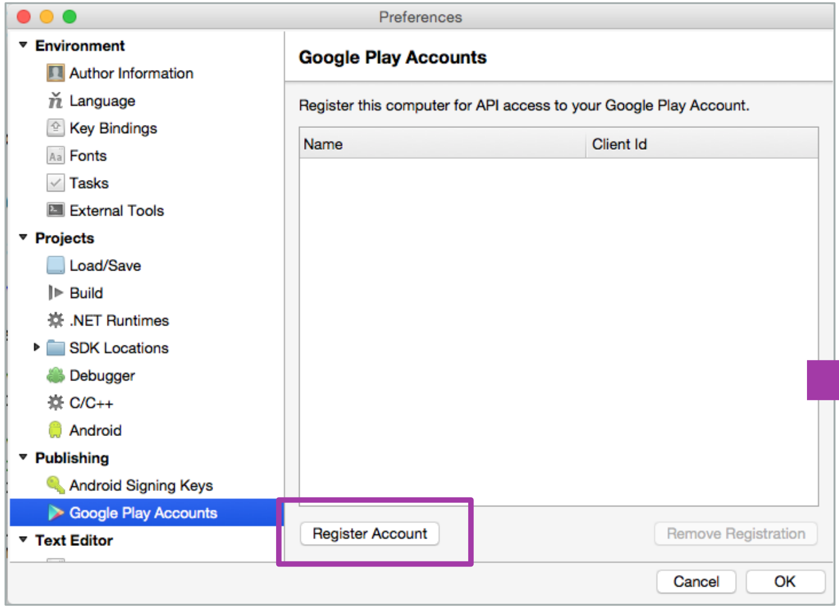
Title* English (United States) – en-US: Angry Developers — 16 of 30 characters

Short description* English (United States) – en-US: — 0 of 80 characters

Full description* English (United States) – en-US:

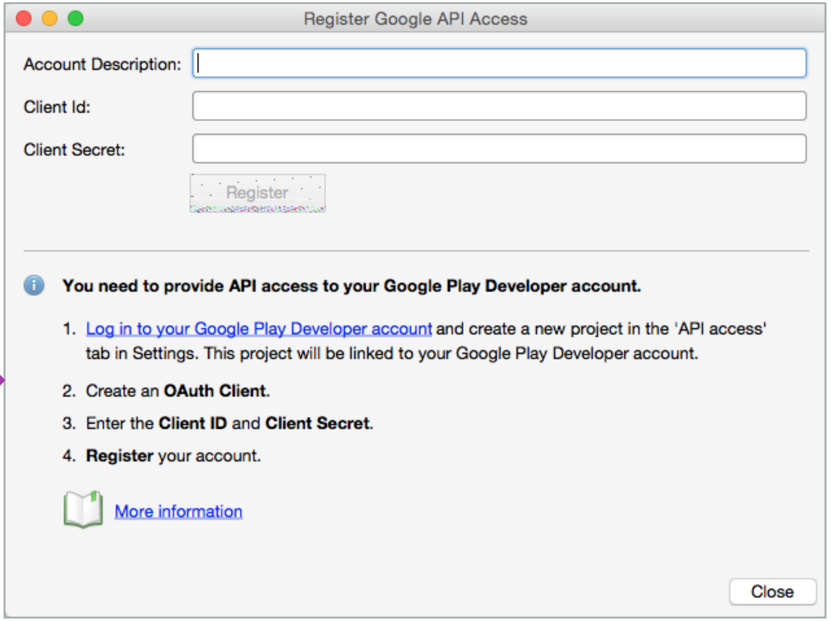
Registering Google dev account [Mac]

- Visual Studio for Mac has built-in support to publish to the Google Play Store



Google Play Accounts

Register this computer for API access to your Google Play Account.



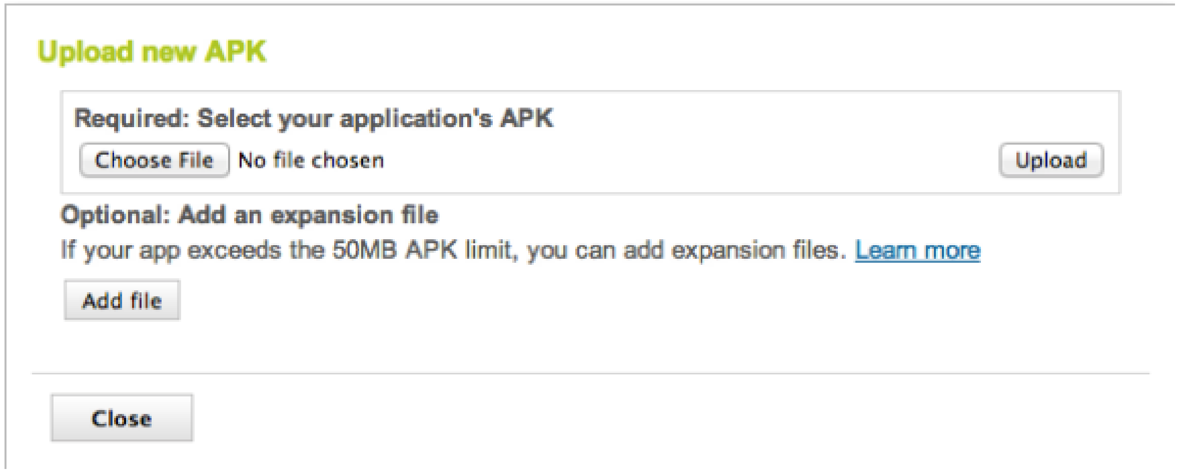
You need to provide API access to your Google Play Developer account.

1. Log in to your Google Play Developer account and create a new project in the 'API access' tab in Settings. This project will be linked to your Google Play Developer account.
2. Create an **OAuth Client.**
3. Enter the **Client ID** and **Client Secret.**
4. **Register** your account.

Xamarin University

Uploading the APK manually

- Click the Upload Application button on your application's developer console page to prompt you to upload the APK manually



Upload new APK

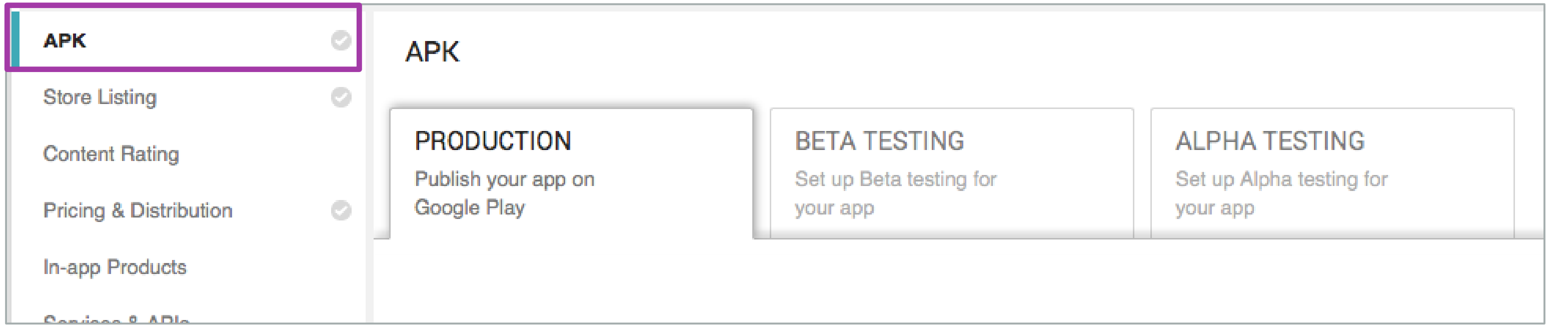
Required: Select your application's APK

Optional: Add an expansion file

If your app exceeds the 50MB APK limit, you can add expansion files. Learn more

Alpha and Beta Testing

- Can invite alpha and beta testers to privately test your app



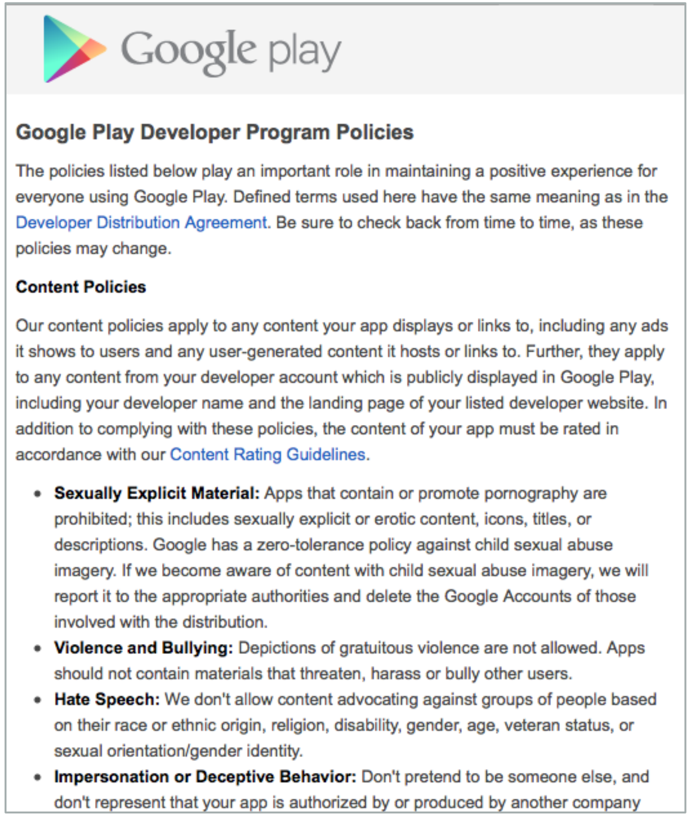
PRODUCTION
Publish your app on Google Play

BETA TESTING
Set up Beta testing for your app

ALPHA TESTING
Set up Alpha testing for your app

Review process

- Once you publish the application on the Google Play Store, your app will enter a review process – similar to the Apple AppStore
- Reviewers look for violations of the published Developer program policies
- Can check publishing status on the app page in the developer console



Google play

Google Play Developer Program Policies

The policies listed below play an important role in maintaining a positive experience for everyone using Google Play. Defined terms used here have the same meaning as in the Developer Distribution Agreement. Be sure to check back from time to time, as these policies may change.

Content Policies

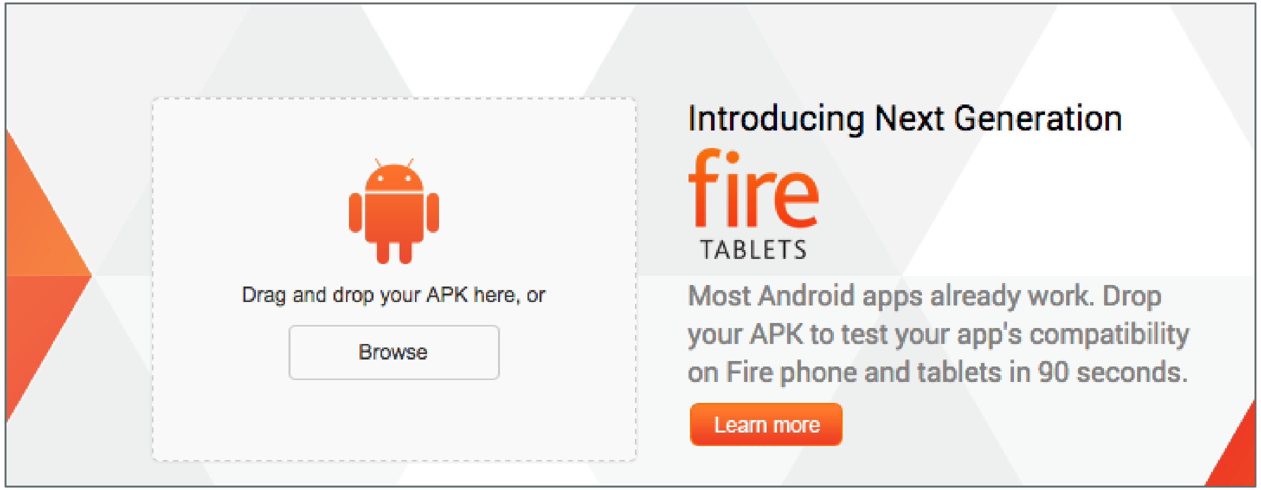
Our content policies apply to any content your app displays or links to, including any ads it shows to users and any user-generated content it hosts or links to. Further, they apply to any content from your developer account which is publicly displayed in Google Play, including your developer name and the landing page of your listed developer website. In addition to complying with these policies, the content of your app must be rated in accordance with our Content Rating Guidelines.

- **Sexually Explicit Material:** Apps that contain or promote pornography are prohibited; this includes sexually explicit or erotic content, icons, titles, or descriptions. Google has a zero-tolerance policy against child sexual abuse imagery. If we become aware of content with child sexual abuse imagery, we will report it to the appropriate authorities and delete the Google Accounts of those involved with the distribution.
- **Violence and Bullying:** Depictions of gratuitous violence are not allowed. Apps should not contain materials that threaten, harass or bully other users.
- **Hate Speech:** We don't allow content advocating against groups of people based on their race or ethnic origin, religion, disability, gender, age, veteran status, or sexual orientation/gender identity.
- **Impersonation or Deceptive Behavior:** Don't pretend to be someone else, and don't represent that your app is authorized by or produced by another company

Xamarin University

Amazon Appstore for Android

- Amazon has a store intended for their Fire devices, your app must run properly on Fire to be accepted into their store


Drag and drop your APK here, or

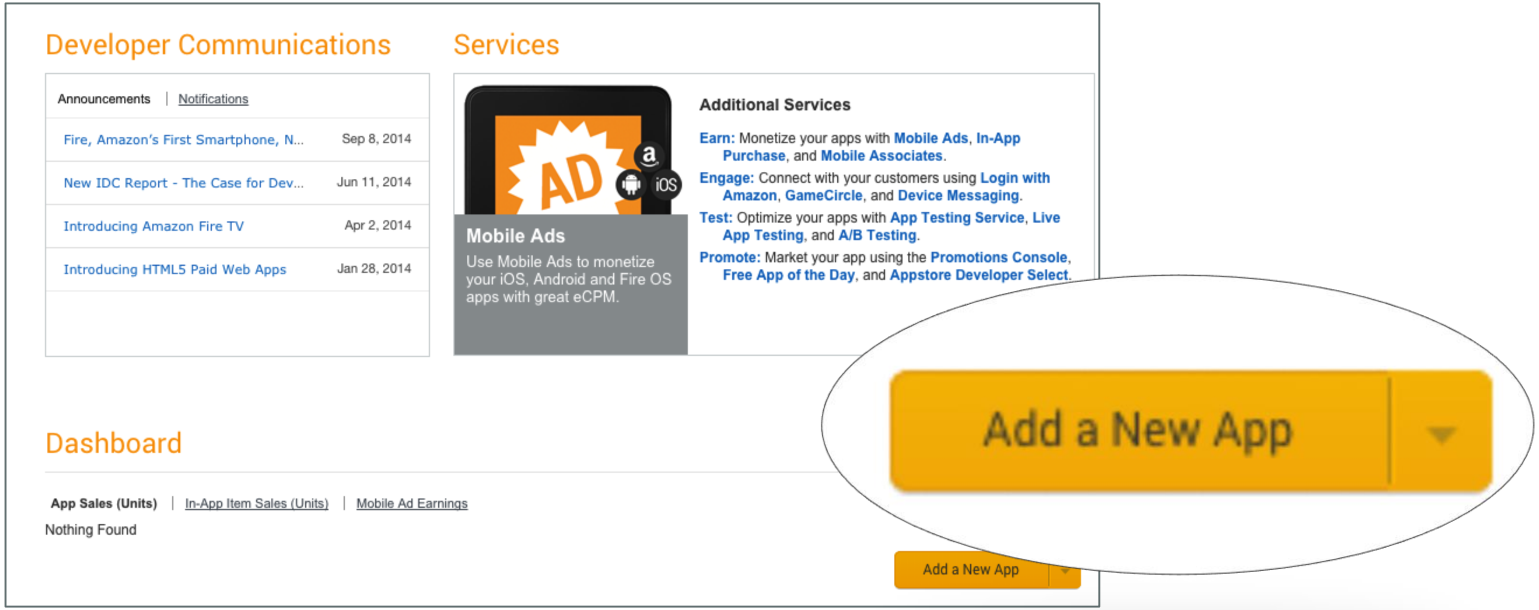
Introducing Next Generation fire TABLETS

Most Android apps already work. Drop your APK to test your app's compatibility on Fire phone and tablets in 90 seconds.

developer.amazon.com/appsandservices/

Adding a new application

❖ Once you have a registered (free) account, you can use the developer console to add a new application



Developer Communications

Announcements | Notifications

Fire, Amazon's First Smartphone, N...	Sep 8, 2014
New IDC Report - The Case for Dev...	Jun 11, 2014
Introducing Amazon Fire TV	Apr 2, 2014
Introducing HTML5 Paid Web Apps	Jan 28, 2014

Services

Mobile Ads

Use Mobile Ads to monetize your iOS, Android and Fire OS apps with great eCPM.

Additional Services

Earn: Monetize your apps with Mobile Ads, In-App Purchase, and Mobile Associates.

Engage: Connect with your customers using Login with Amazon, GameCircle, and Device Messaging.

Test: Optimize your apps with App Testing Service, Live App Testing, and A/B Testing.

Promote: Market your app using the Promotions Console, Free App of the Day, and Appstore Developer Select.

Dashboard

App Sales (Units) | In-App Item Sales (Units) | Mobile Ad Earnings

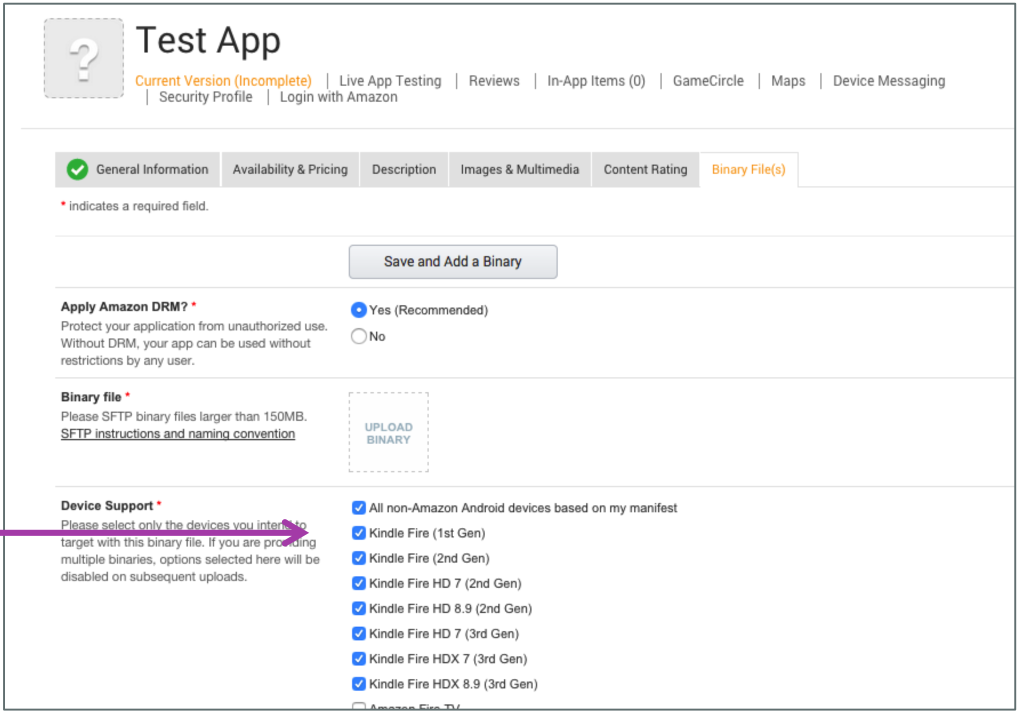
Nothing Found

Xamarin University

Uploading your app

- Entered information is almost identical to what is used in the Google Play Store, but tailored to Amazon's ecosystem

Select the Fire devices your app is compatible with



Test App

Current Version (Incomplete) | Live App Testing | Reviews | In-App Items (0) | GameCircle | Maps | Device Messaging | Security Profile | Login with Amazon

✓ General Information | Availability & Pricing | Description | Images & Multimedia | Content Rating | Binary File(s)

* indicates a required field.

Save and Add a Binary

Apply Amazon DRM? *
Protect your application from unauthorized use. Without DRM, your app can be used without restrictions by any user.
[✓ Yes (Recommended)] No

Binary file *
Please SFTP binary files larger than 150MB.
SFTP instructions and naming convention
UPLOAD BINARY

Device Support *
Please select only the devices you intend to target with this binary file. If you are providing multiple binaries, options selected here will be disabled on subsequent uploads.

- ✓ All non-Amazon Android devices based on my manifest
- ✓ Kindle Fire (1st Gen)
- ✓ Kindle Fire (2nd Gen)
- ✓ Kindle Fire HD 7 (2nd Gen)
- ✓ Kindle Fire HD 8.9 (2nd Gen)
- ✓ Kindle Fire HD 7 (3rd Gen)
- ✓ Kindle Fire HDX 7 (3rd Gen)
- ✓ Kindle Fire HDX 8.9 (3rd Gen)
- ☐ Amazon Fire TV

Summary

1. Publish an application
2. Side-load an application for testing

Thank You!

Please complete the class survey in your profile:
university.xamarin.com/profile

Microsoft