REU & NRT Android App Development Tutorial

George Mason University

Instructor:
Dr. Kevin Moran

Tutorial will start in:

12:01

The tutorial will begin soon

REU & NRT Android App Development Tutorial

George Mason University

Instructor:
Dr. Kevin Moran

Tutorial will start in:

12:01

The tutorial will begin soon

REU & NRT Tutorial Series

Summer 2021

George Mason University

Dr. Kevin Moran

Android Application Development Tutorial

Welcome to the Tutorial!

Tutorial Overview

1. *A Brief Introduction to Android* - Getting Oriented
2. *Android App Fundamentals* - Knowing Your Building Blocks
3. *Our First Android App* - A Quick Walkthrough
4. *10 Minute Break -* Lecture Over
5. *Group Coding Sessions* - Implementing some Android Features!

A Few Quick Notes

- This tutorial assumes no prior experience with Android, but does assume some general programming knowledge
- Android development is a HUGE topic, entire courses are dedicated to it! (Check out GMU's CS-477 course)
  - Today I will be giving an introduction and providing some instructions related to popular features.
- The Android developer documentation is typically excellent, and there are lots of other tutorials and documentation that are a quick Google Search away!

https://sagelab.io/android-dev-tutorial/

Introductions

Instructor: Kevin Moran

Education: Ph.D. from William & Mary - 2018

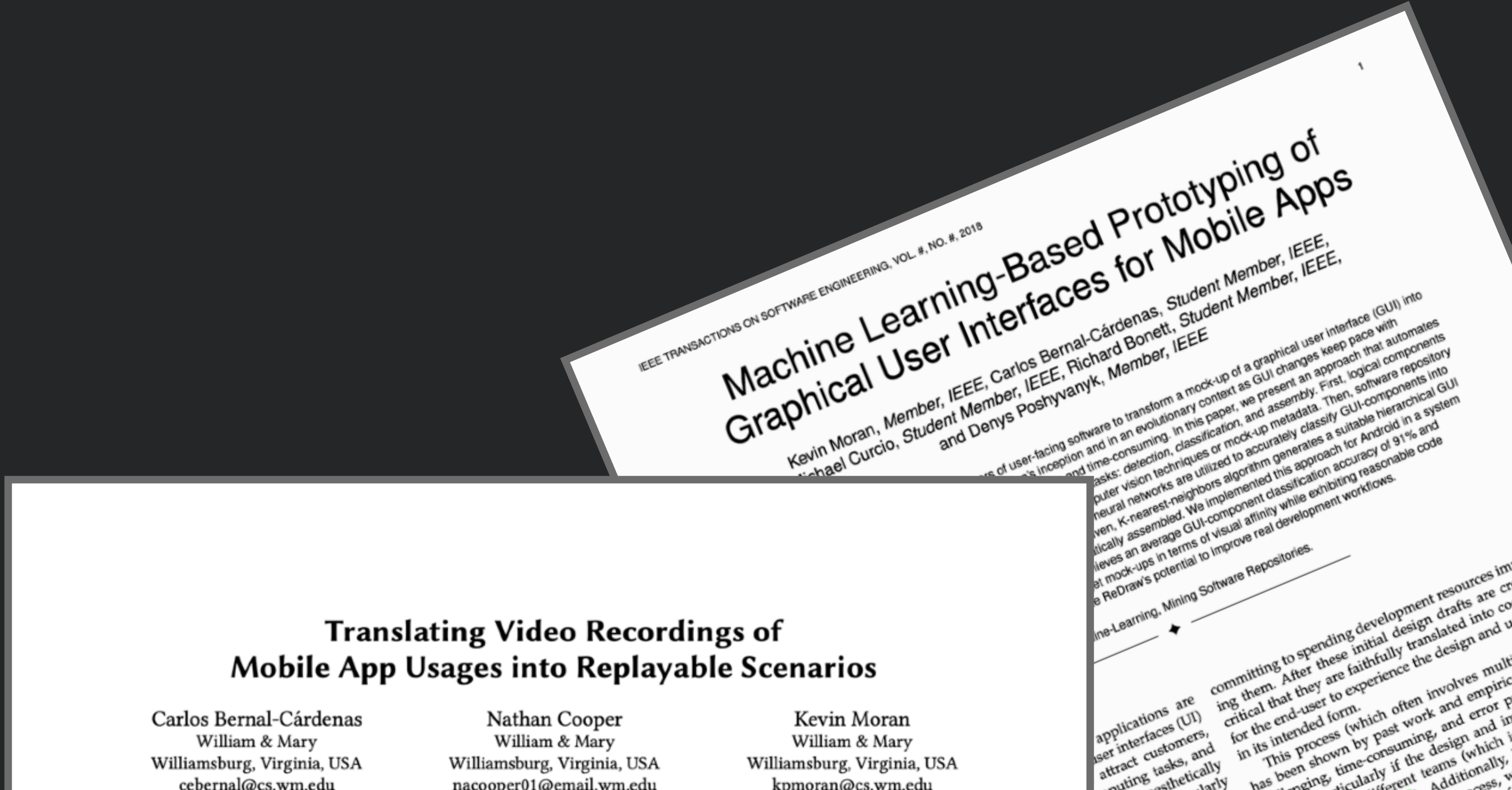
Research Interests: Software Engineering , Mobile App Development, UI Analysis, Machine Learning

Introductions

Instructor: Kevin Moran

Education: Ph.D. from William & Mary - 2018

Research Interests: Software Engineering , Mobile App Development, UI Analysis, Machine Learning



IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. #, NO. #, 2018 1

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

Kevin Moran, *Member, IEEE*, Carlos Bernal-Cárdenas, *Student Member, IEEE*, ichael Curcio, *Student Member, IEEE*, Richard Bonett, *Student Member, IEEE*, and Denys Poshyvanyk, *Member, IEEE*

rs of user-facing software to transform a mock-up of a graphical user interface (GUI) into 's inception and in an evolutionary context as GUI changes keep pace with nd time-consuming. In this paper, we present an approach that automates asks: *detection, classification,* and *assembly*. First, logical components puter vision techniques or mock-up metadata. Then, software repository neural networks are utilized to accurately *classify* GUI-components into ven, K-nearest-neighbors algorithm generates a suitable hierarchical GUI tically *assembled*. We implemented this approach for Android in a system ieves an average GUI-component classification accuracy of 91% and et mock-ups in terms of visual affinity while exhibiting reasonable code e ReDraw's potential to improve real development workflows.

ine-Learning, Mining Software Repositories.

applications are ser interfaces (UI) attract customers, uting tasks, and esthetically larly

committing to spending development resources in ing them. After these initial design drafts are cr critical that they are faithfully translated into co for the end-user to experience the design and u in its intended form.

This process (which often involves mult has been shown by past work and empiric llenging, time-consuming, and error p rticularly if the design and in ifferent teams (which Additionally, ess,

Translating Video Recordings of Mobile App Usages into Replayable Scenarios

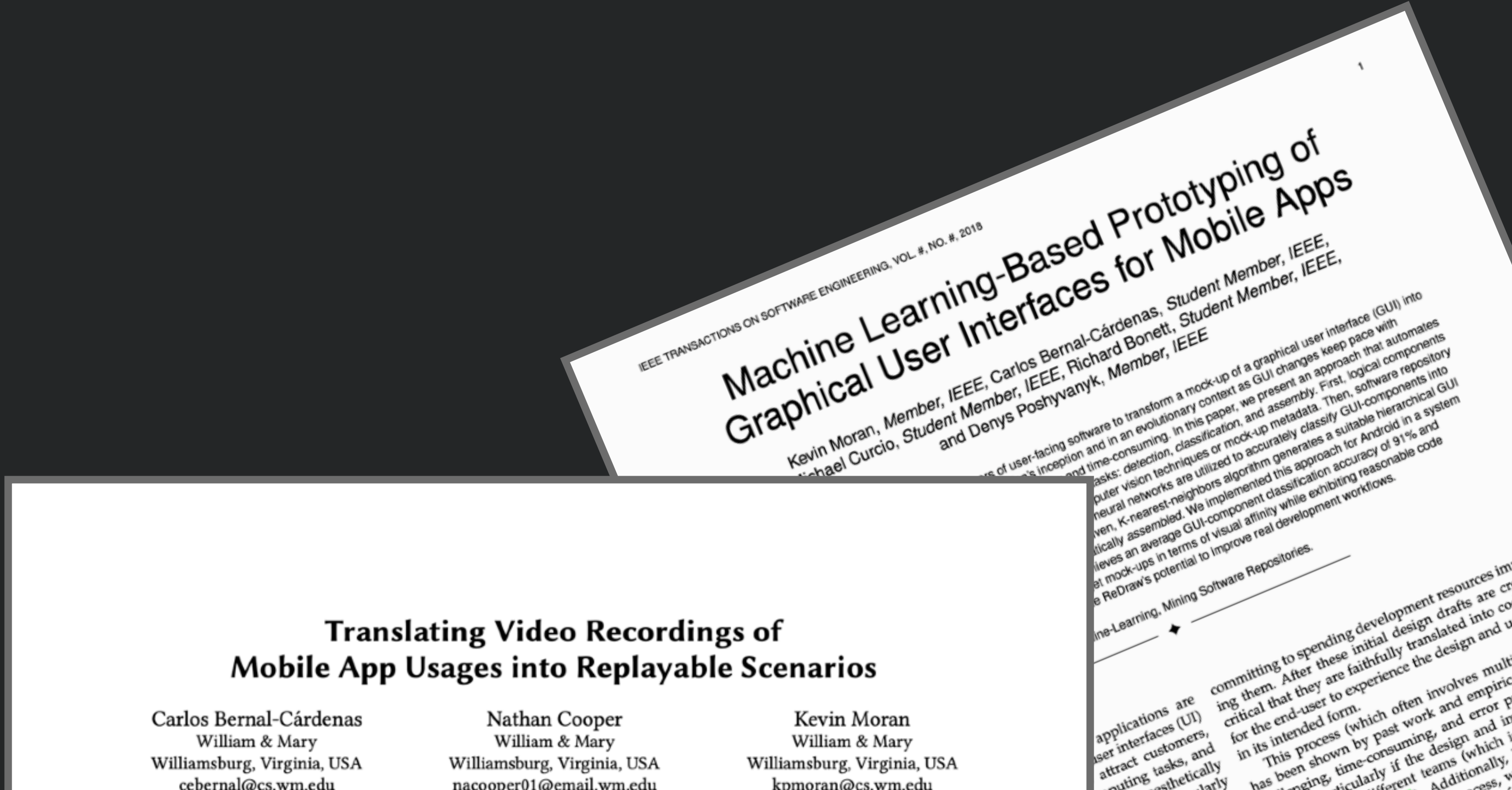
Carlos Bernal-Cárdenas	Nathan Cooper	Kevin Moran
William & Mary	William & Mary	William & Mary
Williamsburg, Virginia, USA	Williamsburg, Virginia, USA	Williamsburg, Virginia, USA
cebernal@cs.wm.edu	nacooper01@email.wm.edu	kpmoran@cs.wm.edu

Introductions

Instructor: Kevin Moran

Education: Ph.D. from William & Mary - 2018

Research Interests: Software Engineering , Mobile App Development, UI Analysis, Machine Learning



IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. #, NO. #, 2018

1

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

Kevin Moran, *Member, IEEE*, Carlos Bernal-Cárdenas, *Student Member, IEEE*, ichael Curcio, *Student Member, IEEE*, Richard Bonett, *Student Member, IEEE*, and Denys Poshyvanyk, *Member, IEEE*

rs of user-facing software to transform a mock-up of a graphical user interface (GUI) into 's inception and in an evolutionary context as GUI changes keep pace with nd time-consuming. In this paper, we present an approach that automates asks: *detection*, *classification*, and *assembly*. First, logical components puter vision techniques or mock-up metadata. Then, software repository neural networks are utilized to accurately *classify* GUI-components into ven, K-nearest-neighbors algorithm generates a suitable hierarchical GUI tically *assembled*. We implemented this approach for Android in a system lieves an average GUI-component classification accuracy of 91% and et mock-ups in terms of visual affinity while exhibiting reasonable code e ReDraw's potential to improve real development workflows.

ine-Learning, Mining Software Repositories.

✦

applications are ser interfaces (UI) attract customers, puting tasks, and esthetically larly

committing to spending development resources im ing them. After these initial design drafts are cr critical that they are faithfully translated into co for the end-user to experience the design and u in its intended form.

This process (which often involves mult has been shown by past work and empiric llenging, time-consuming, and error p ticularly if the design and im fferent teams (which Additionally, cess,

Translating Video Recordings of Mobile App Usages into Replayable Scenarios

Carlos Bernal-Cárdenas
William & Mary
Williamsburg, Virginia, USA
cebernal@cs.wm.edu

Nathan Cooper
William & Mary
Williamsburg, Virginia, USA
nacooper01@email.wm.edu

Kevin Moran
William & Mary
Williamsburg, Virginia, USA
kpmoran@cs.wm.edu

A Brief Introduction to Android

What is Android?

- Mobile OS maintained by Google
- Runs on a plethora of hardware devices: phones, tablets, watches, TVs, cars, refrigerators
- Based on **Linux** (kernel) and uses **Java** and **Kotlin** as dev languages
- The #1 overall OS in the world!
- Code is released as Open Source under the AOSP
- Easier to customize, license, pirate, etc. than iOS

Why Android?

- Android is ***open source***, ***highly modifiable***, and tends to ***fit well*** into a variety of research topics.

- Android is the most popular OS in the world with over 3 Billion active devices!

History of Android

- Started with ex-Apple veteran Andy Rubin and a small company called Danger

credit: androidcentral.com

History of Android

- Started with ex-Apple veteran Andy Rubin and a small company called Danger

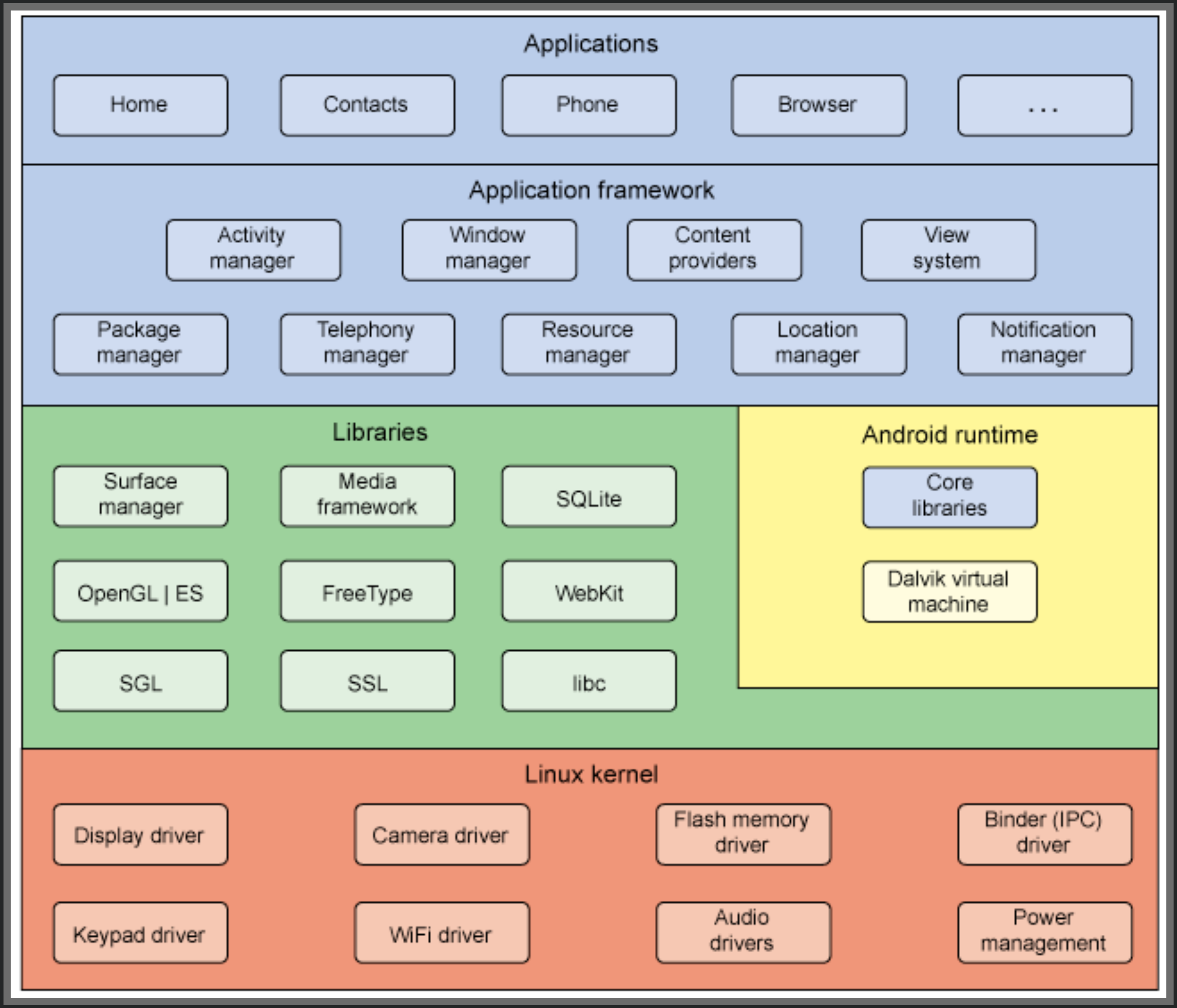
credit: androidcentral.com

History of Android

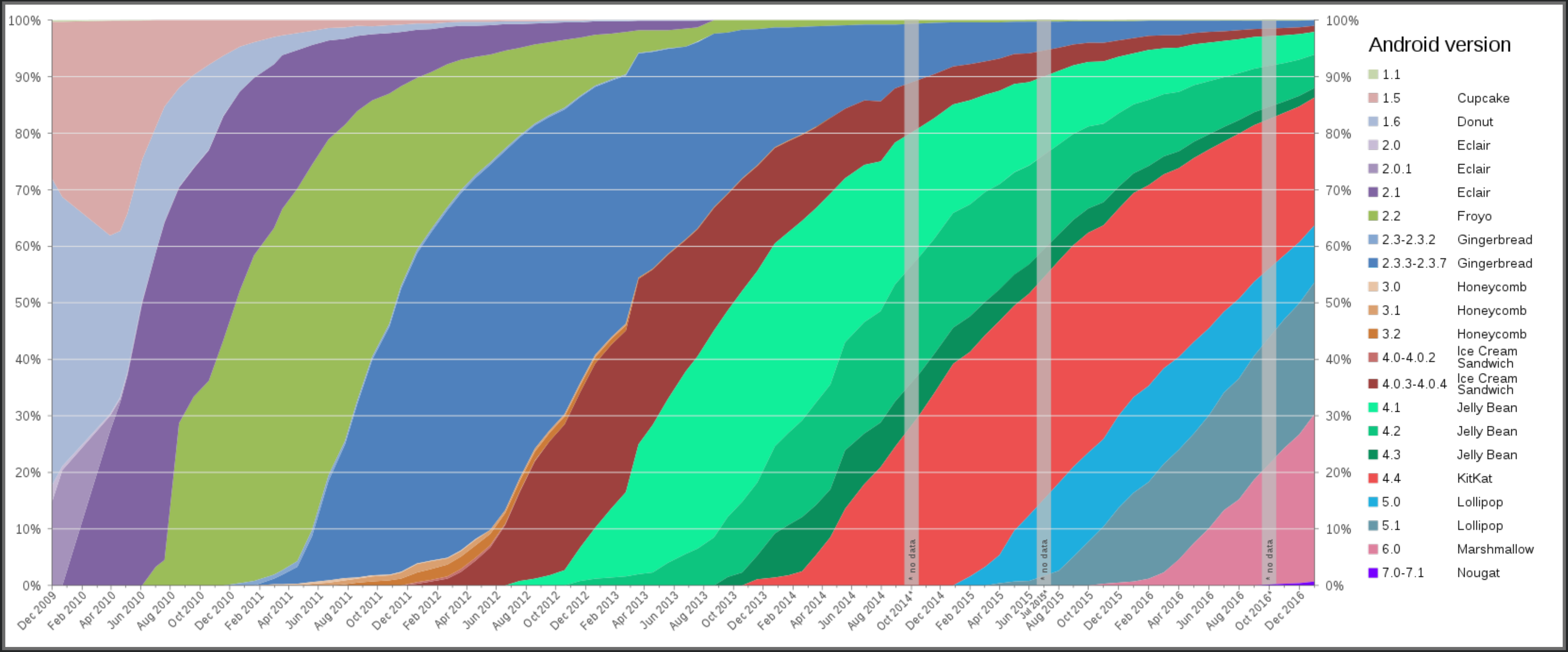
- Rubin was ousted from Danger and started Android Inc.
- This was a company focused on making standalone mobile software for phones
- In 2005, Android Inc. was acquired by Google
- In 2008, the T-Mobile was G1 was released

Android Architecture

- Android OS provides libraries and APIs for many system features, such as notifications, camera, audio, phone, etc.
- Android code is compiled into a special bytecode format called *Dalvik*



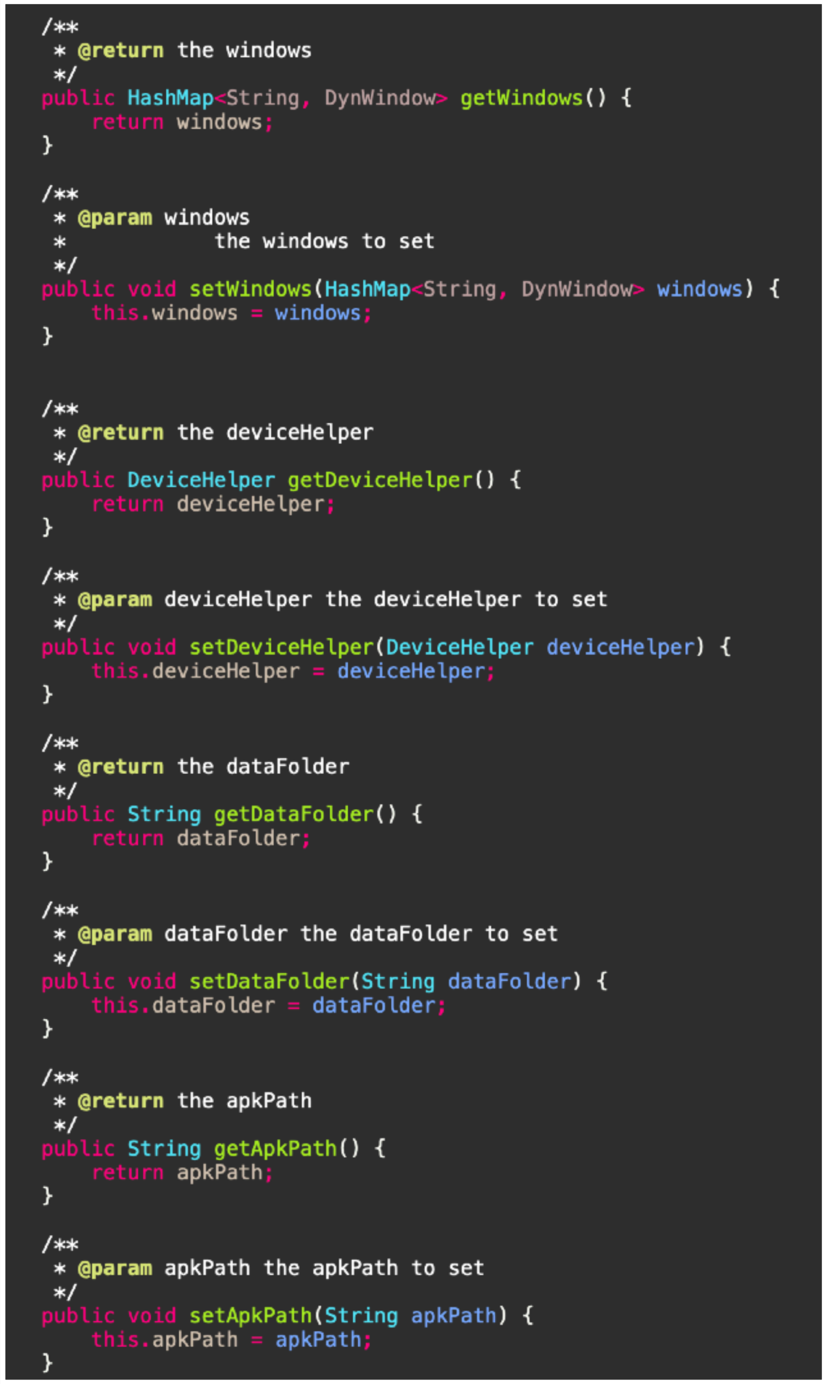
Android Version Distribution



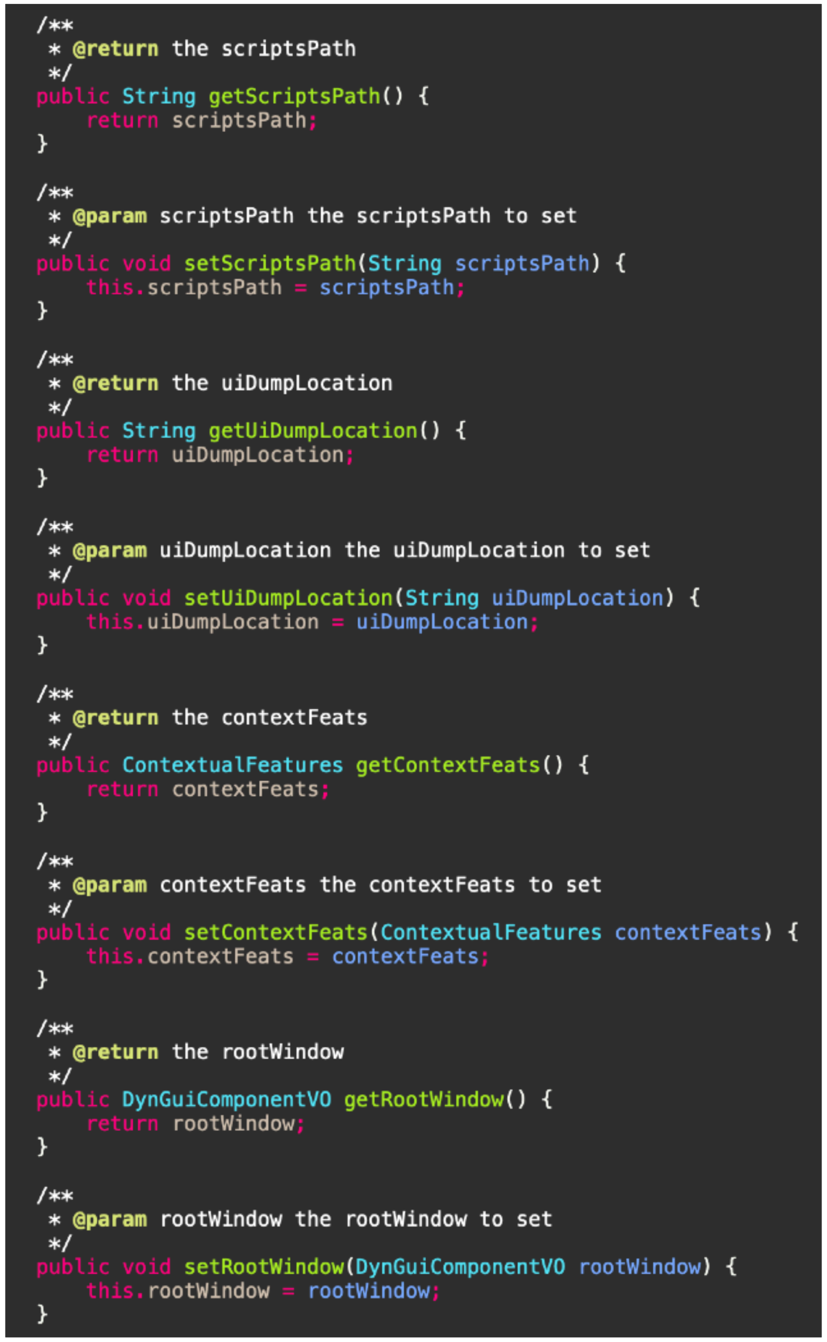
The Kotlin Language

- Kotlin was developed by *Jet Brains* as an alternative to Java.
- Java is an extremely verbose language, and this was slowing down development time for JetBrains, whose IDE, IntelliJ, was written primarily in Java.
- Kotlin was unveiled in July 2011, in 2017, at Google I/O Google announced first-class support for Kotlin on Android

Java's Verbosity…



```java
/**
 * @return the windows
 */
public HashMap<String, DynWindow> getWindows() {
    return windows;
}

/**
 * @param windows
 *            the windows to set
 */
public void setWindows(HashMap<String, DynWindow> windows) {
    this.windows = windows;
}


/**
 * @return the deviceHelper
 */
public DeviceHelper getDeviceHelper() {
    return deviceHelper;
}

/**
 * @param deviceHelper the deviceHelper to set
 */
public void setDeviceHelper(DeviceHelper deviceHelper) {
    this.deviceHelper = deviceHelper;
}

/**
 * @return the dataFolder
 */
public String getDataFolder() {
    return dataFolder;
}

/**
 * @param dataFolder the dataFolder to set
 */
public void setDataFolder(String dataFolder) {
    this.dataFolder = dataFolder;
}

/**
 * @return the apkPath
 */
public String getApkPath() {
    return apkPath;
}

/**
 * @param apkPath the apkPath to set
 */
public void setApkPath(String apkPath) {
    this.apkPath = apkPath;
}
```



```java
/**
 * @return the scriptsPath
 */
public String getScriptsPath() {
    return scriptsPath;
}

/**
 * @param scriptsPath the scriptsPath to set
 */
public void setScriptsPath(String scriptsPath) {
    this.scriptsPath = scriptsPath;
}

/**
 * @return the uiDumpLocation
 */
public String getUiDumpLocation() {
    return uiDumpLocation;
}

/**
 * @param uiDumpLocation the uiDumpLocation to set
 */
public void setUiDumpLocation(String uiDumpLocation) {
    this.uiDumpLocation = uiDumpLocation;
}

/**
 * @return the contextFeats
 */
public ContextualFeatures getContextFeats() {
    return contextFeats;
}

/**
 * @param contextFeats the contextFeats to set
 */
public void setContextFeats(ContextualFeatures contextFeats) {
    this.contextFeats = contextFeats;
}

/**
 * @return the rootWindow
 */
public DynGuiComponentVO getRootWindow() {
    return rootWindow;
}

/**
 * @param rootWindow the rootWindow to set
 */
public void setRootWindow(DynGuiComponentVO rootWindow) {
    this.rootWindow = rootWindow;
}
```

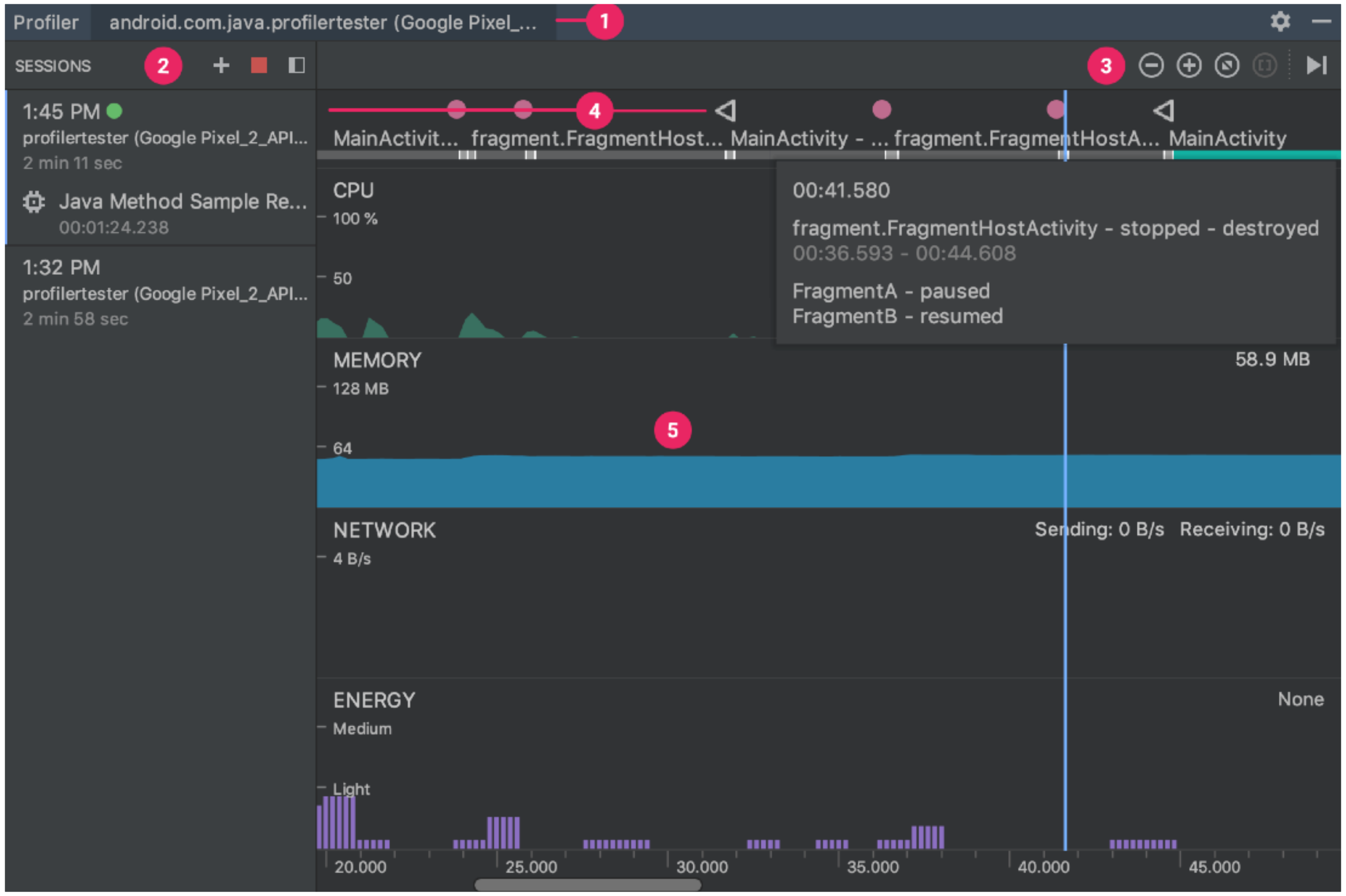
The Kotlin Language

Kotlin

- Kotlin runs on the JVM and is fully interoperable with Java code.
- Semicolons are optional!
- Is far less verbose than Java

Android Developer Tools

- ***Android Profiler:*** Provides real-time data to help you understand how your app uses CPU, memory, network, and battery resources.



Profiler android.com.java.profilertester (Google Pixel_...

SESSIONS

1:45 PM
profilertester (Google Pixel_2_API...
2 min 11 sec

Java Method Sample Re...
00:01:24.238

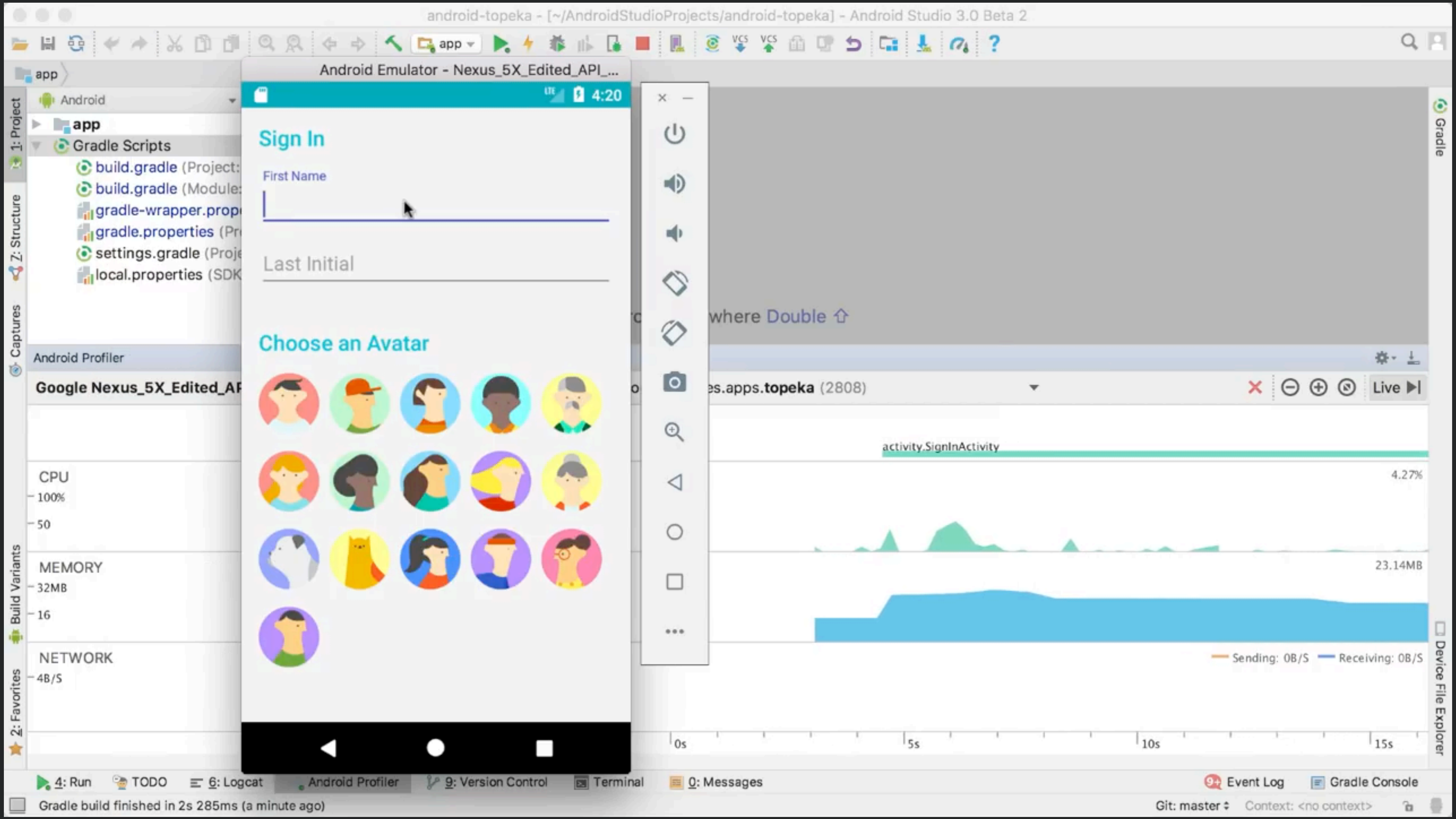
1:32 PM
profilertester (Google Pixel_2_API...
2 min 58 sec

MainActivit... fragment.FragmentHost... MainActivity - ... fragment.FragmentHostA... MainActivity

00:41.580
fragment.FragmentHostActivity - stopped - destroyed
00:36.593 - 00:44.608
FragmentA - paused
FragmentB - resumed

https://developer.android.com/studio/profile/android-profiler

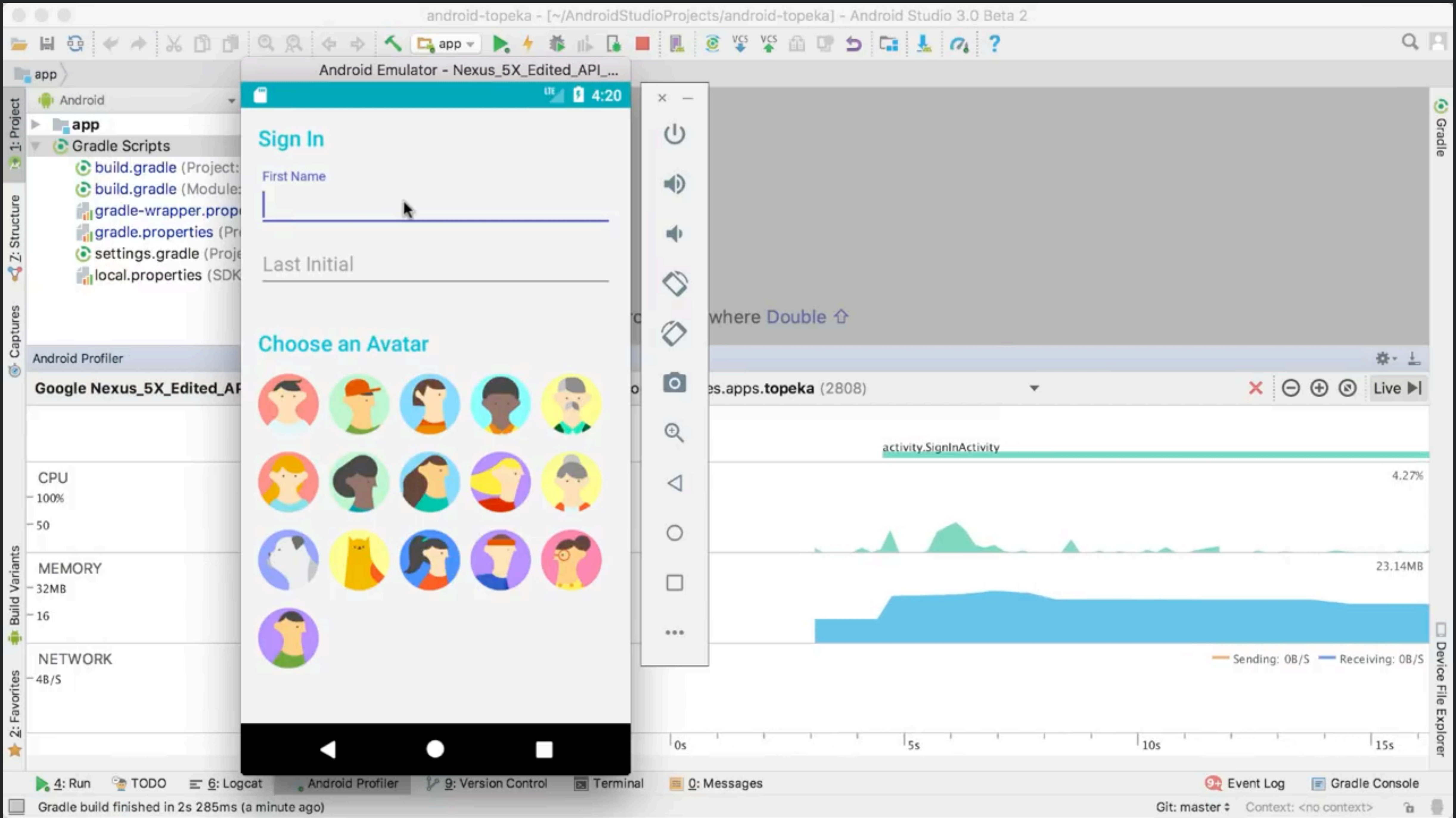
Android Developer Tools



Gradle build finished in 2s 285ms (a minute ago)

https://developer.android.com/studio/profile/android-profiler

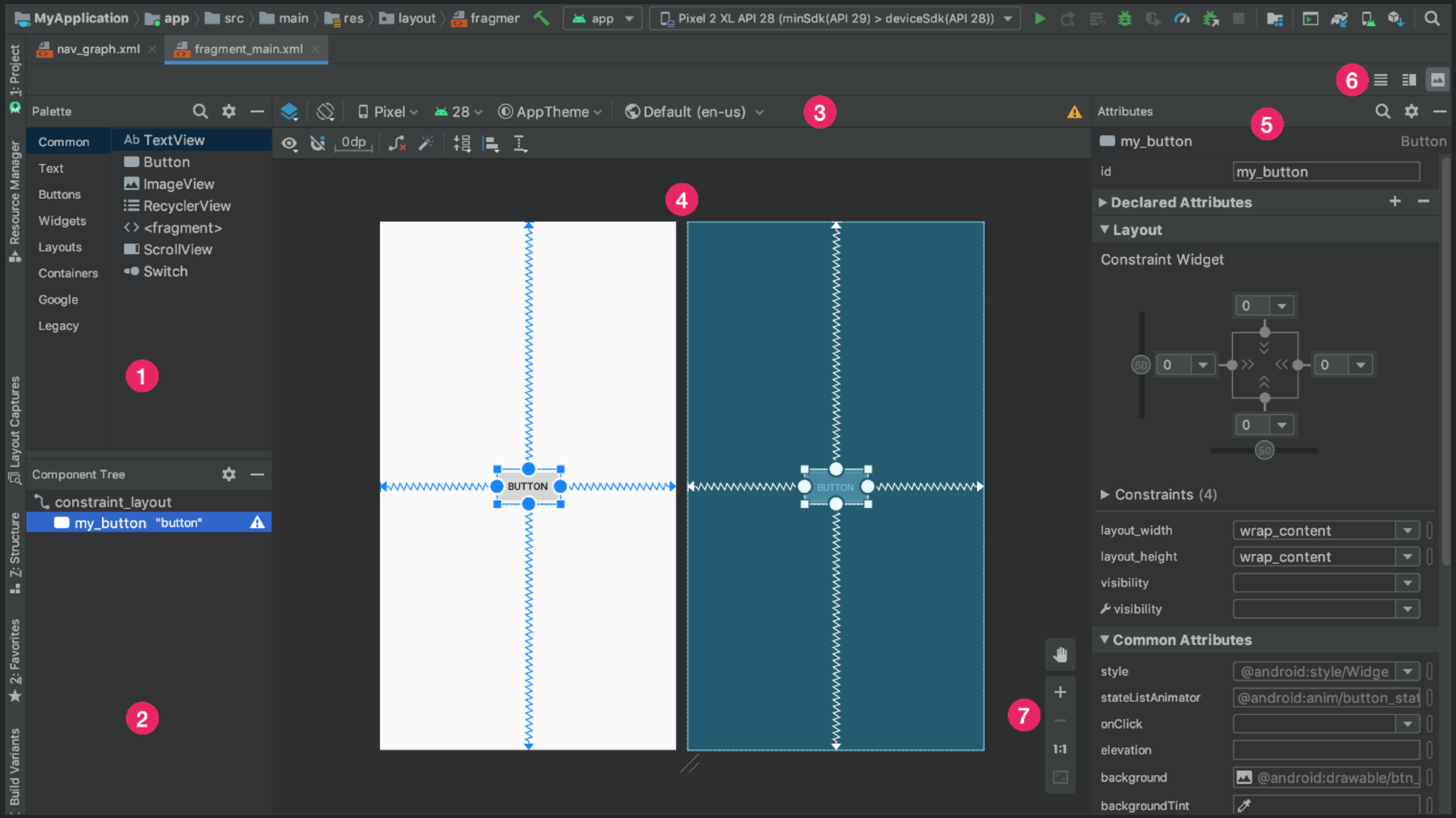
Android Developer Tools



https://developer.android.com/studio/profile/android-profiler

Android Developer Tools

- ***Android Layout Editor:*** Allows you to quickly build UI Layouts by dragging elements into a visual design editor instead of writing layout XML by hand.



id: my_button

layout_width: wrap_content

layout_height: wrap_content

style: @android:style/Widge

stateListAnimator: @android:anim/button_stat

background: @android:drawable/btn_

Android UI Dev Research

UI/UX Design Team

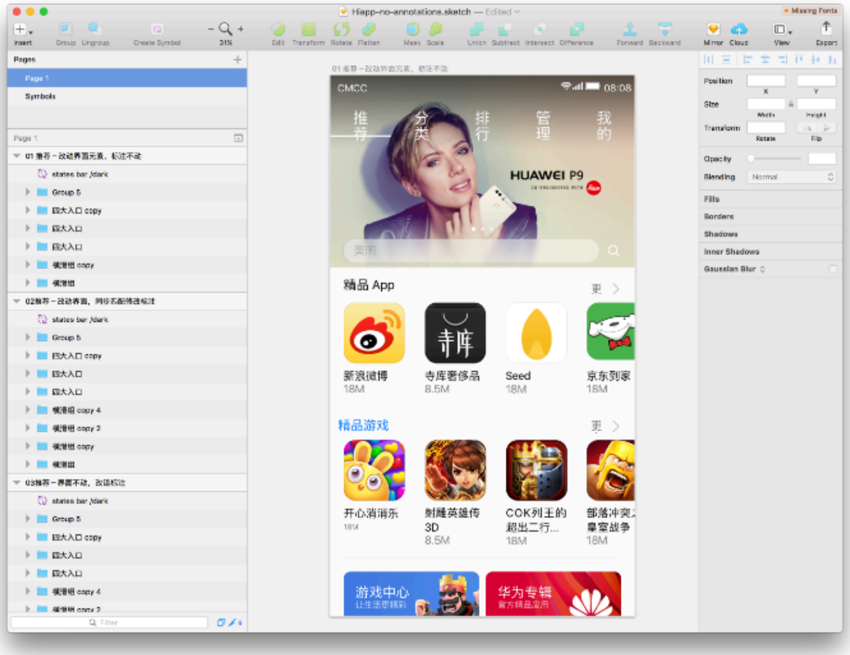
Development Team

Android UI Dev Research

UI/UX Design Team

Development Team

Android UI Dev Research

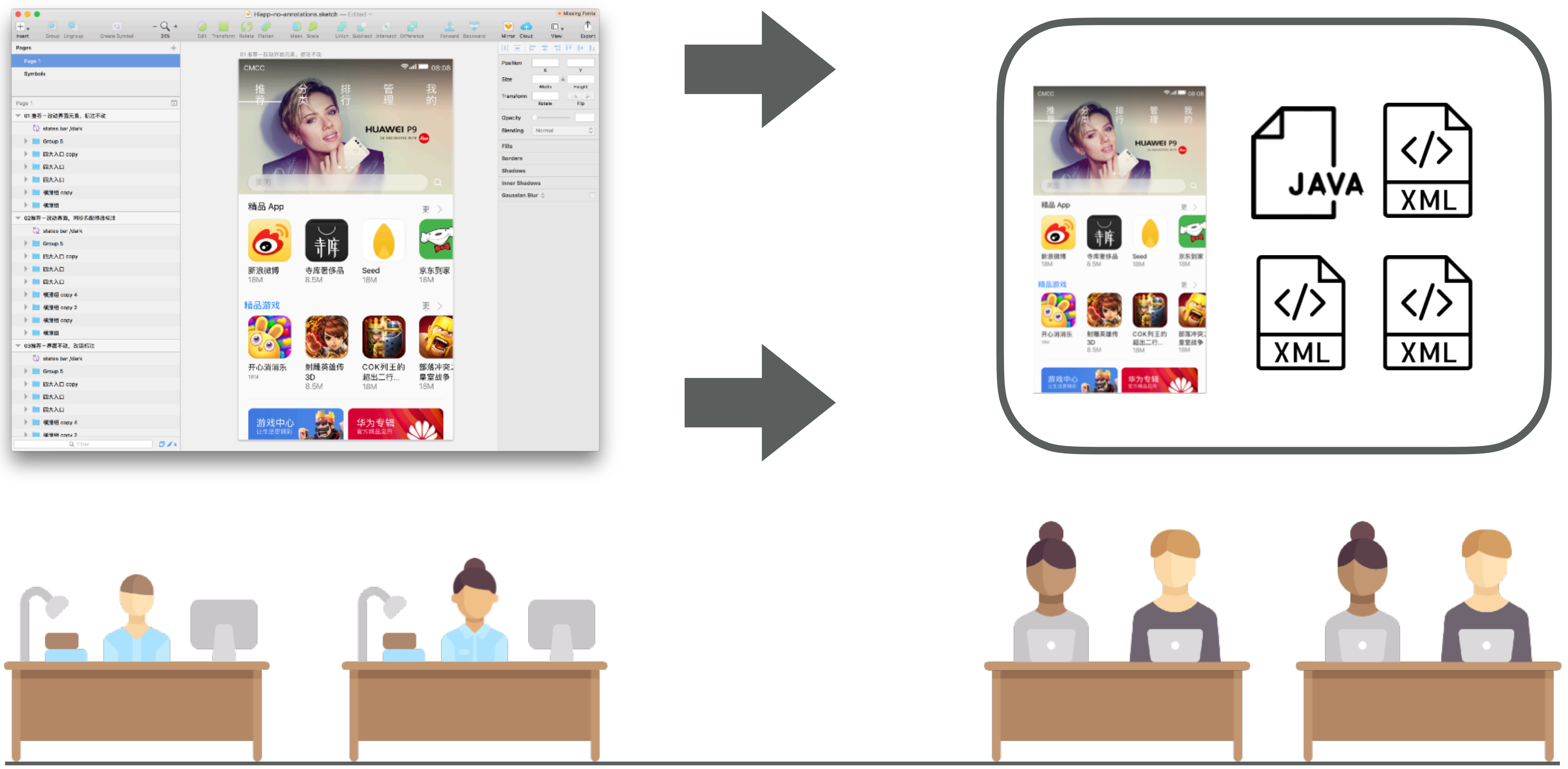


UI/UX Design Team

Development Team

Android UI Dev Research

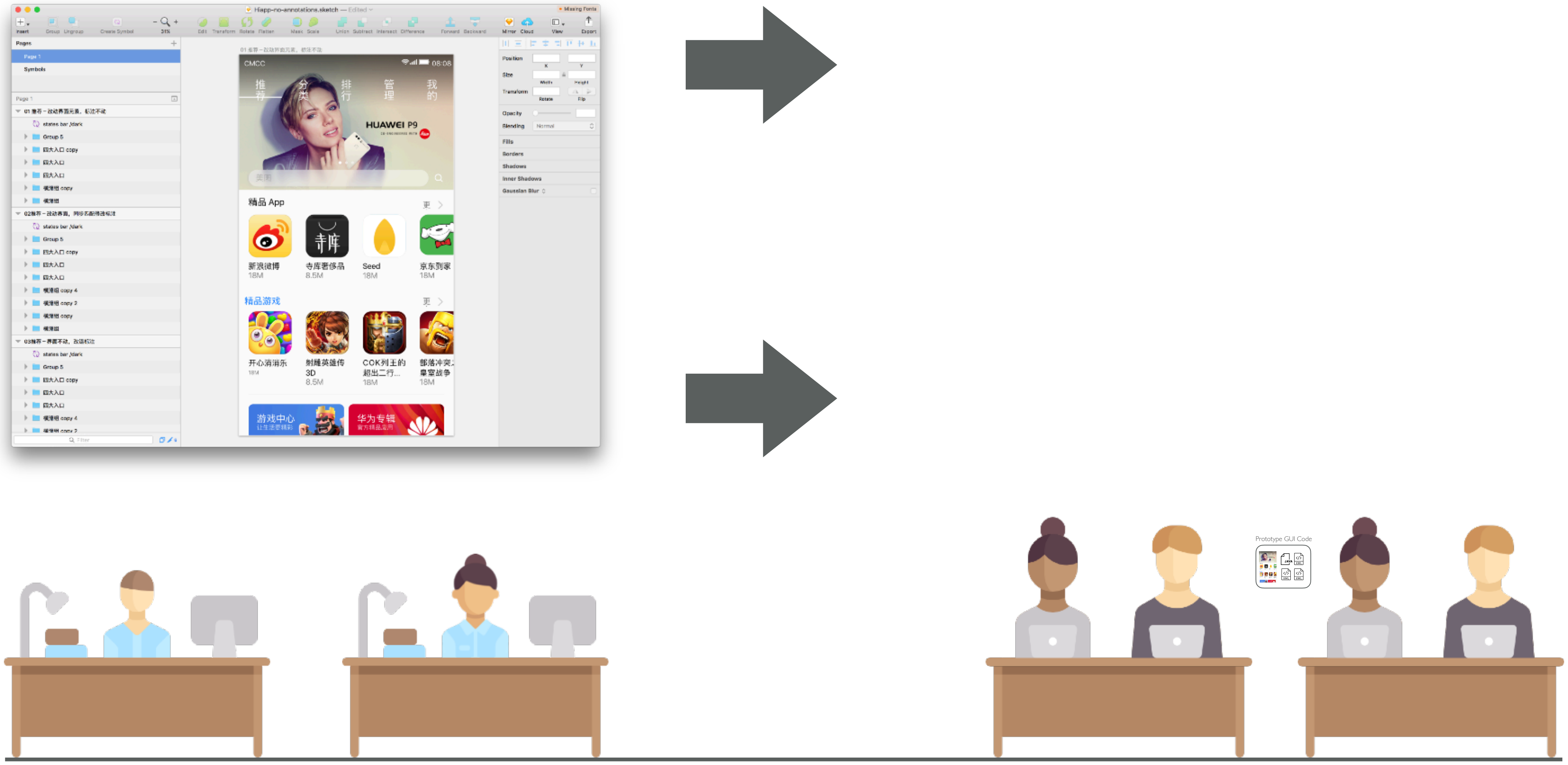
Prototype GUI Code



UI/UX Design Team

Development Team

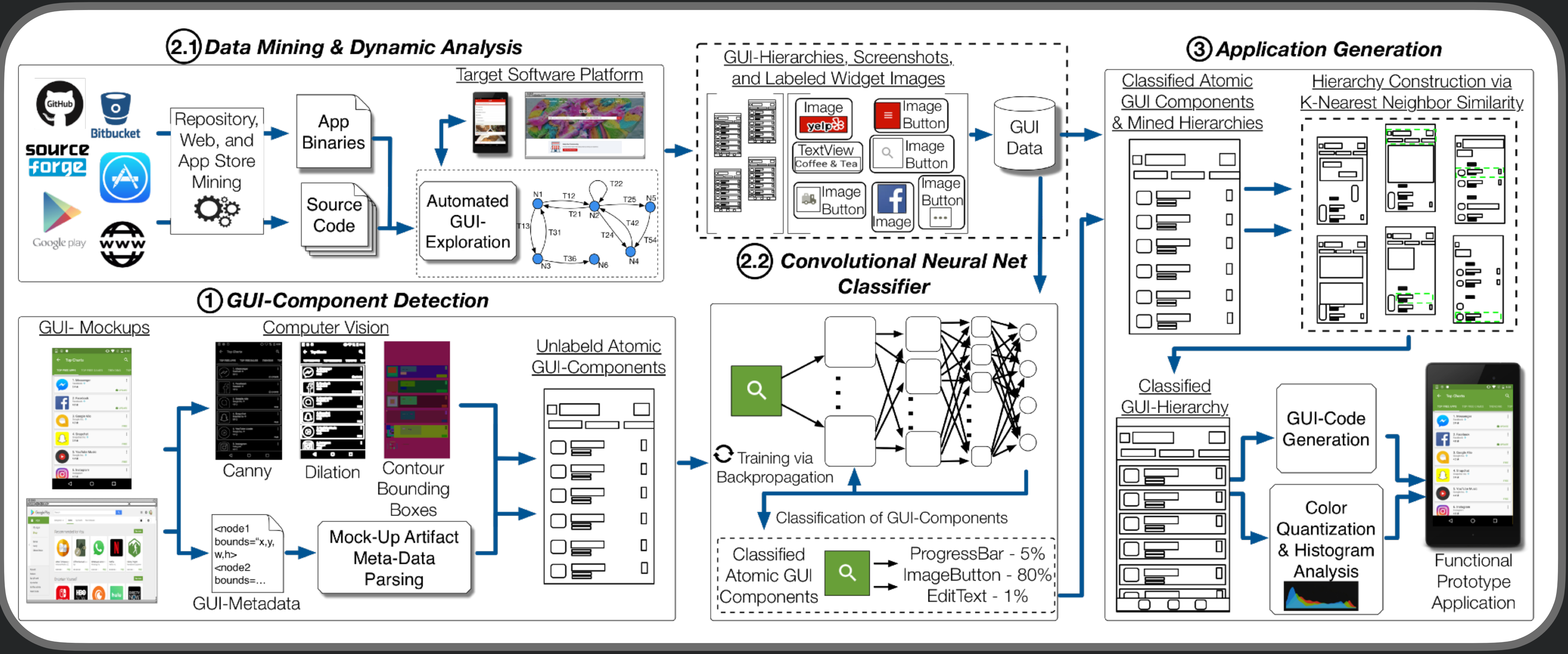
Android UI Dev Research



UI/UX Design Team

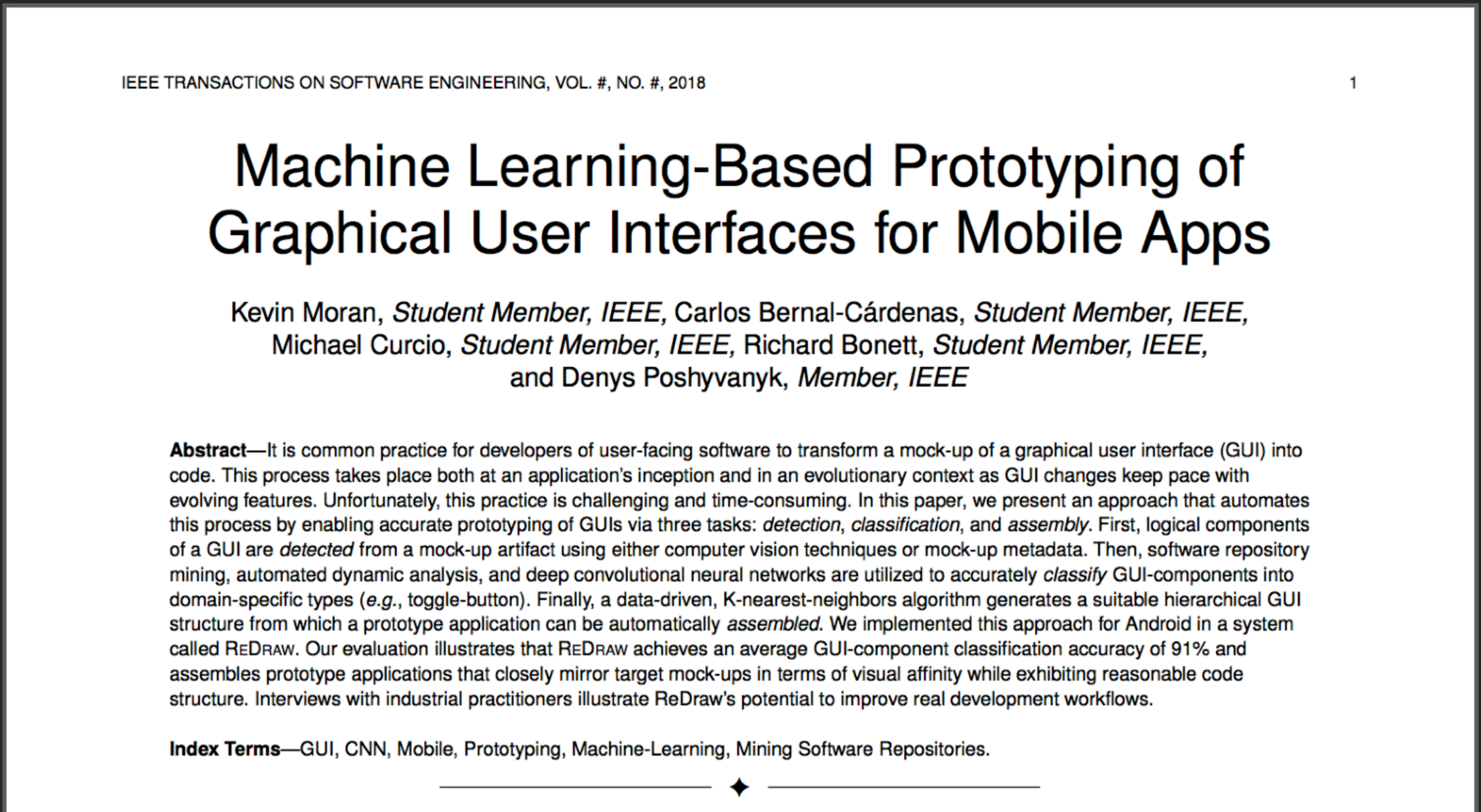
Development Team

Android UI Dev Research


2.1 Data Mining & Dynamic Analysis
Target Software Platform
GitHub
Bitbucket
source forge
Google play
www
Repository, Web, and App Store Mining
App Binaries
Source Code
Automated GUI-Exploration
N1
T12
T22
T25
N5
T21
N2
T13
T31
T42
T24
T54
T36
N3
N6
N4
GUI-Hierarchies, Screenshots, and Labeled Widget Images
Image
yelp
Image Button
TextView
Coffee & Tea
Image Button
Image Button
Image
Image Button
GUI Data
2.2 Convolutional Neural Net Classifier
Training via Backpropagation
Classification of GUI-Components
Classified Atomic GUI Components
ProgressBar - 5%
ImageButton - 80%
EditText - 1%
3 Application Generation
Classified Atomic GUI Components & Mined Hierarchies
Hierarchy Construction via K-Nearest Neighbor Similarity
Classified GUI-Hierarchy
GUI-Code Generation
Color Quantization & Histogram Analysis
Functional Prototype Application
1 GUI-Component Detection
GUI- Mockups
Computer Vision
Canny
Dilation
Contour Bounding Boxes
Unlabeld Atomic GUI-Components
<node1 bounds="x,y, w,h> <node2 bounds=...
GUI-Metadata
Mock-Up Artifact Meta-Data Parsing

Android UI Dev Research

https://www.android-dev-tools.com/redraw



IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. #, NO. #, 2018 1

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

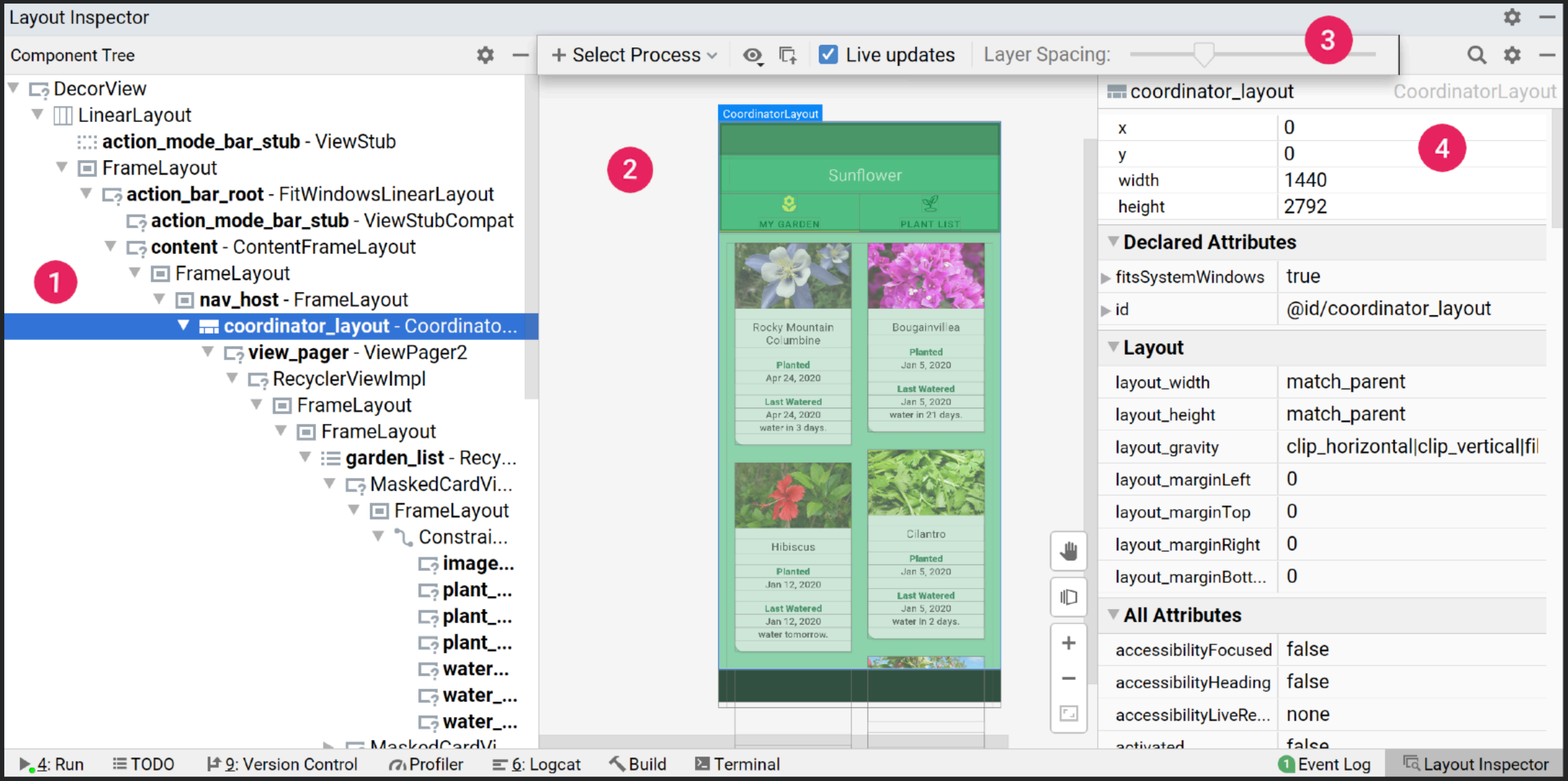
Kevin Moran, *Student Member, IEEE,* Carlos Bernal-Cárdenas, *Student Member, IEEE,*
Michael Curcio, *Student Member, IEEE,* Richard Bonett, *Student Member, IEEE,*
and Denys Poshyvanyk, *Member, IEEE*

Abstract—It is common practice for developers of user-facing software to transform a mock-up of a graphical user interface (GUI) into code. This process takes place both at an application's inception and in an evolutionary context as GUI changes keep pace with evolving features. Unfortunately, this practice is challenging and time-consuming. In this paper, we present an approach that automates this process by enabling accurate prototyping of GUIs via three tasks: *detection*, *classification*, and *assembly*. First, logical components of a GUI are *detected* from a mock-up artifact using either computer vision techniques or mock-up metadata. Then, software repository mining, automated dynamic analysis, and deep convolutional neural networks are utilized to accurately *classify* GUI-components into domain-specific types (*e.g.*, toggle-button). Finally, a data-driven, K-nearest-neighbors algorithm generates a suitable hierarchical GUI structure from which a prototype application can be automatically *assembled*. We implemented this approach for Android in a system called ReDraw. Our evaluation illustrates that ReDraw achieves an average GUI-component classification accuracy of 91% and assembles prototype applications that closely mirror target mock-ups in terms of visual affinity while exhibiting reasonable code structure. Interviews with industrial practitioners illustrate ReDraw's potential to improve real development workflows.

Index Terms—GUI, CNN, Mobile, Prototyping, Machine-Learning, Mining Software Repositories.

Android Developer Tools

- ***Android Layout Inspector:*** Allows you to compare your app with design mockups, and examine runtime details of your UI layout to help with testing and validation.



```
▼ DecorView
  ▼ LinearLayout
      action_mode_bar_stub - ViewStub
    ▼ FrameLayout
      ▼ action_bar_root - FitWindowsLinearLayout
          action_mode_bar_stub - ViewStubCompat
        ▼ content - ContentFrameLayout
          ▼ FrameLayout
            ▼ nav_host - FrameLayout
              ▼ coordinator_layout - Coordinato...
                ▼ view_pager - ViewPager2
                  ▼ RecyclerViewImpl
                    ▼ FrameLayout
                      ▼ FrameLayout
                        ▼ garden_list - Recy...
                          ▼ MaskedCardVi...
                            ▼ FrameLayout
                              ▼ Constrai...
                                  image...
                                  plant_...
                                  plant_...
                                  plant_...
                                  water...
                                  water_...
                                  water_...
                          ▶ MaskedCardVi...
```

coordinator_layout CoordinatorLayout

x	0
y	0
width	1440
height	2792

▼ **Declared Attributes**

▶ fitsSystemWindows	true
▶ id	@id/coordinator_layout

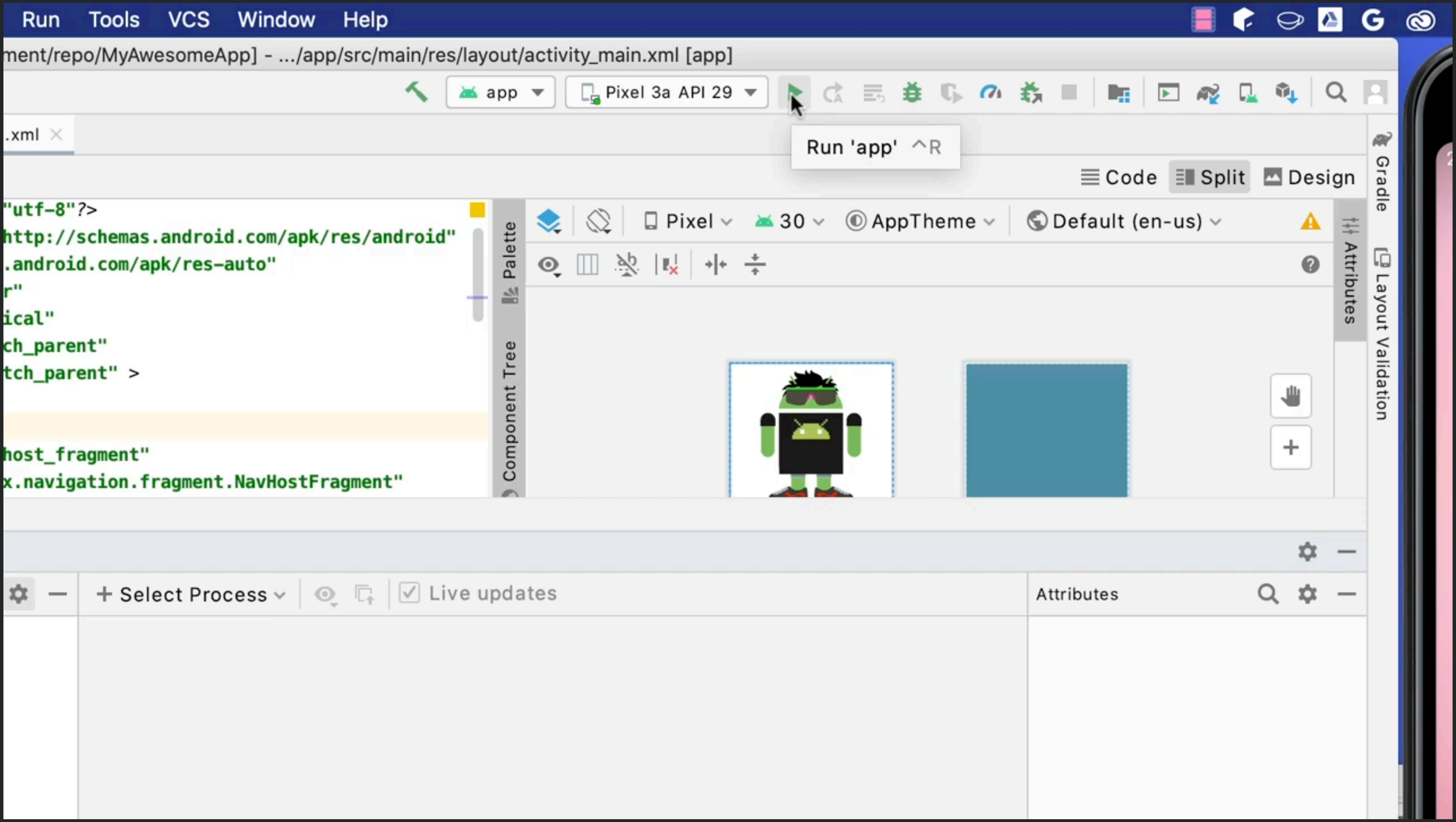
▼ **Layout**

layout_width	match_parent
layout_height	match_parent
layout_gravity	clip_horizontal|clip_vertical|fi
layout_marginLeft	0
layout_marginTop	0
layout_marginRight	0
layout_marginBott...	0

▼ **All Attributes**

accessibilityFocused	false
accessibilityHeading	false
accessibilityLiveRe...	none
activated	false

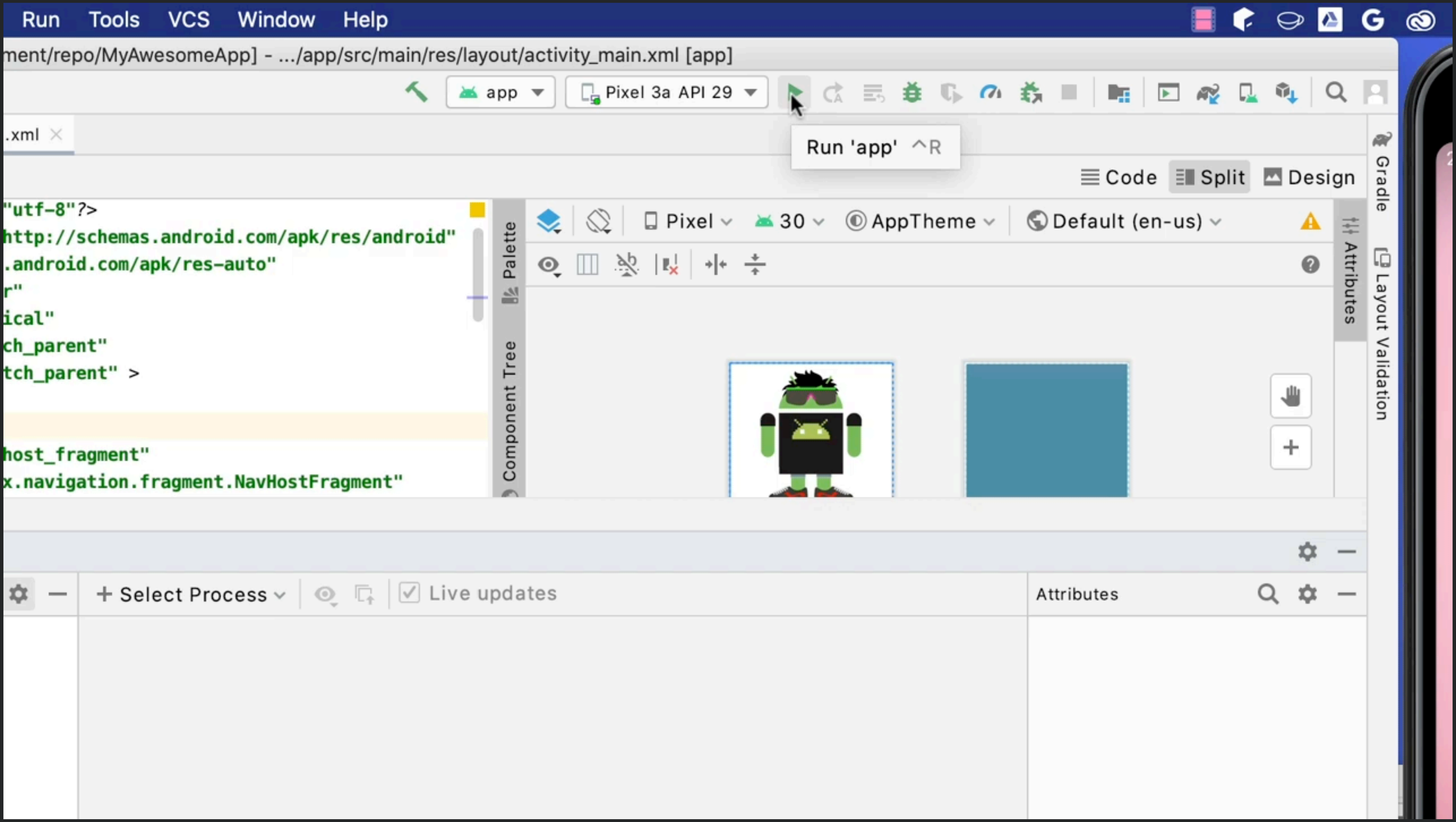
Android Developer Tools



ment/repo/MyAwesomeApp] - .../app/src/main/res/layout/activity_main.xml [app]

```
"utf-8"?>
http://schemas.android.com/apk/res/android"
.android.com/apk/res-auto"
r"
ical"
ch_parent"
tch_parent" >

host_fragment"
x.navigation.fragment.NavHostFragment"
```

Android Developer Tools

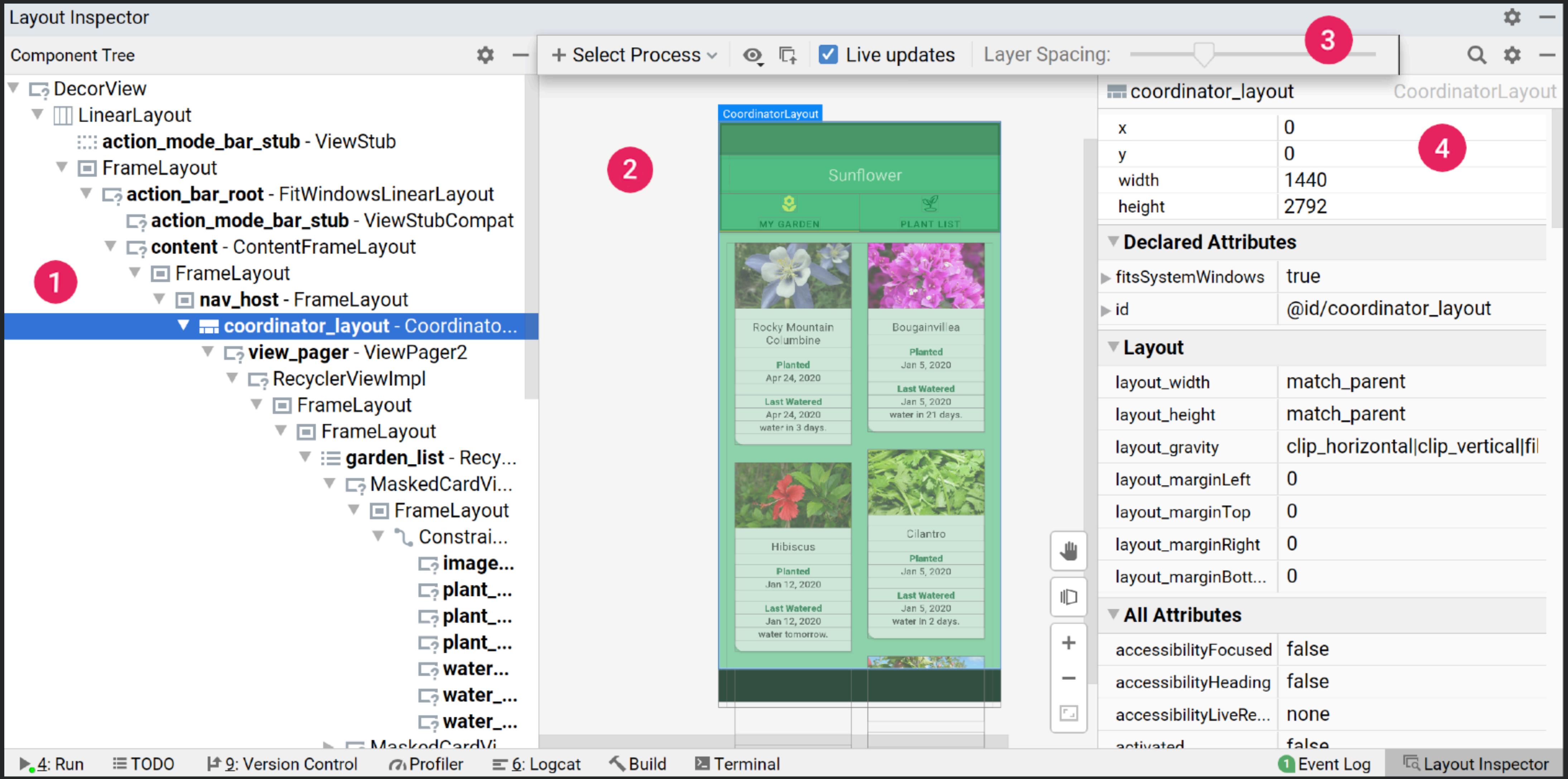


ment/repo/MyAwesomeApp] - .../app/src/main/res/layout/activity_main.xml [app]

```
"utf-8"?>
http://schemas.android.com/apk/res/android"
.android.com/apk/res-auto"
r"
ical"
ch_parent"
tch_parent" >

host_fragment"
x.navigation.fragment.NavHostFragment"
```

Android Developer Tools

- ***Espresso UI Test Cases:*** Allows you to write and run UI test cases to test your application and avoid regressions



Layout Inspector

Component Tree

```
▼ DecorView
  ▼ LinearLayout
      action_mode_bar_stub - ViewStub
    ▼ FrameLayout
      ▼ action_bar_root - FitWindowsLinearLayout
          action_mode_bar_stub - ViewStubCompat
        ▼ content - ContentFrameLayout
          ▼ FrameLayout
            ▼ nav_host - FrameLayout
              ▼ coordinator_layout - Coordinato...
                ▼ view_pager - ViewPager2
                  ▼ RecyclerViewImpl
                    ▼ FrameLayout
                      ▼ FrameLayout
                        ▼ garden_list - Recy...
                          ▼ MaskedCardVi...
                            ▼ FrameLayout
                              ▼ Constrai...
                                  image...
                                  plant_...
                                  plant_...
                                  plant_...
                                  water...
                                  water_...
                                  water_...
                          ▶ MaskedCardVi...
```

+ Select Process ⌄ | ☑ Live updates | Layer Spacing:

coordinator_layout — CoordinatorLayout

x	0
y	0
width	1440
height	2792

▼ Declared Attributes

▶ fitsSystemWindows	true
▶ id	@id/coordinator_layout

▼ Layout

layout_width	match_parent
layout_height	match_parent
layout_gravity	clip_horizontal|clip_vertical|fil
layout_marginLeft	0
layout_marginTop	0
layout_marginRight	0
layout_marginBott...	0

▼ All Attributes

accessibilityFocused	false
accessibilityHeading	false
accessibilityLiveRe...	none
activated	false

▶ 4: Run | ☰ TODO | 9: Version Control | Profiler | ☰ 6: Logcat | Build | Terminal | 1 Event Log | Layout Inspector

Android Testing Research

- Testing Android Apps is hard work. I have worked on several research projects that aim to improve this!

V2S: A Tool for Translating Video Recordi
Mobile App Usages into Replayable

Madeleine Havranek*, Carlos Bernal-Cárdenas*, Nat
Oscar Chaparro*, Denys Poshyvnayk*, Ke
lege of William & Mary (Williamsburg, VA, USA), †Georg
mrhavranek@email.wm.edu, cebernal@cs.wm.e
oscarch@wm.edu, denys@cs.w

rdings are becoming i
ifacts that infor
vever, the
ion f

Automatically Discovering, Reporting and
Reproducing Android Application Crashes

Kevin Moran, Mario Linares-Vásquez, Carlos Bernal-Cárdenas, Christoph
College of William & Mary
{kpmoran, mlinarev, cebernal, cvendom

ue challenges when de-
heir prevailing GUI
uts (e.g., sensor
e introduce
This tool
ion,

the mo
Furtherm
of develop
addition to
specific to A
fragmentation
[48], the event-d
tion, sensor

How do Developers Test Android Applications?

Mario Linares-Vásquez[1], Carlos Bernal-Cárdenas[2], Kevin Moran[2], and Denys Poshyvanyk[2]
[1]Universidad de los Andes, Bogotá, Colombia
[2]College of William & Mary, Williamsburg, VA, USA
m.linaresv@uniandes.edu.co, {cebernal, kpmoran, denys}@cs.wm.edu

—Enabling fully automated testing of mobile applica-
ntly become an important topic of study for both
actitioners. A plethora of tools and approaches
id mobile developers both by augmenting
nd by automating various parts of
rrent approaches for automated
lopers about their benefits,
ing performed manually.
ractitioners – who
to understand
102 open
tices

or tester's perspective are the automate
(AIG) techniques. The high-level goal o
relatively simple: given a mobile applic
generate a series of program *inputs* ac
testing goal. For the vast majority
generated *inputs* are simulated tow
a device, and the *testing goal*
the highest possible code-cove
number of faults (e.g., crashe
However, despite the lar
icated to building AIG t
oaches, recent studies
ically not used i
ial reaso

Android App Fundamentals

Android App Basics

- Each Android application exists in its own security sandbox.
  - Android is essentially a multi-user Linux operating system, where each app is its own “user”
  - Each user has access to certain set of permissions granted by the user
  - Each process is run in its own virtual machine or VM
  - Every app runs in it’s own Linux process
  - Each app has a main “UI Thread” that handles general processing

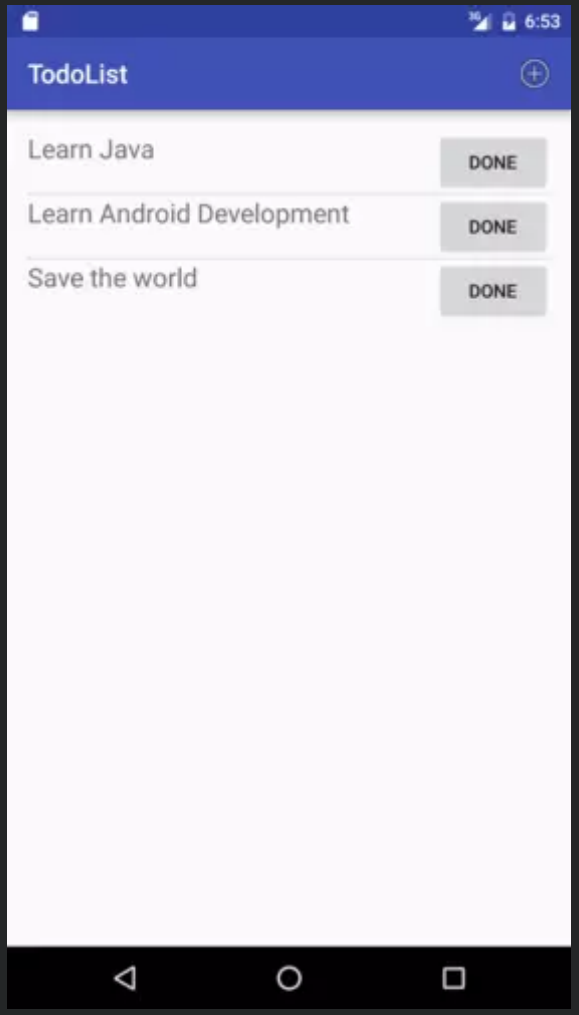
https://developer.android.com/guide/components/fundamentals

Android Application Components

- ***Activities (Fragments):*** Entry point for interacting with a user, representing a single screen with a user interface.
- ***Services:*** General Purpose entry point for keeping an app running in the background.
- ***Broadcast Receivers:*** Allows an app to respond to system-wide broadcast announcements
- ***Content Providers:*** Manages a shared set of app data that you can store in the file system, in a SQLite database, or on the web.

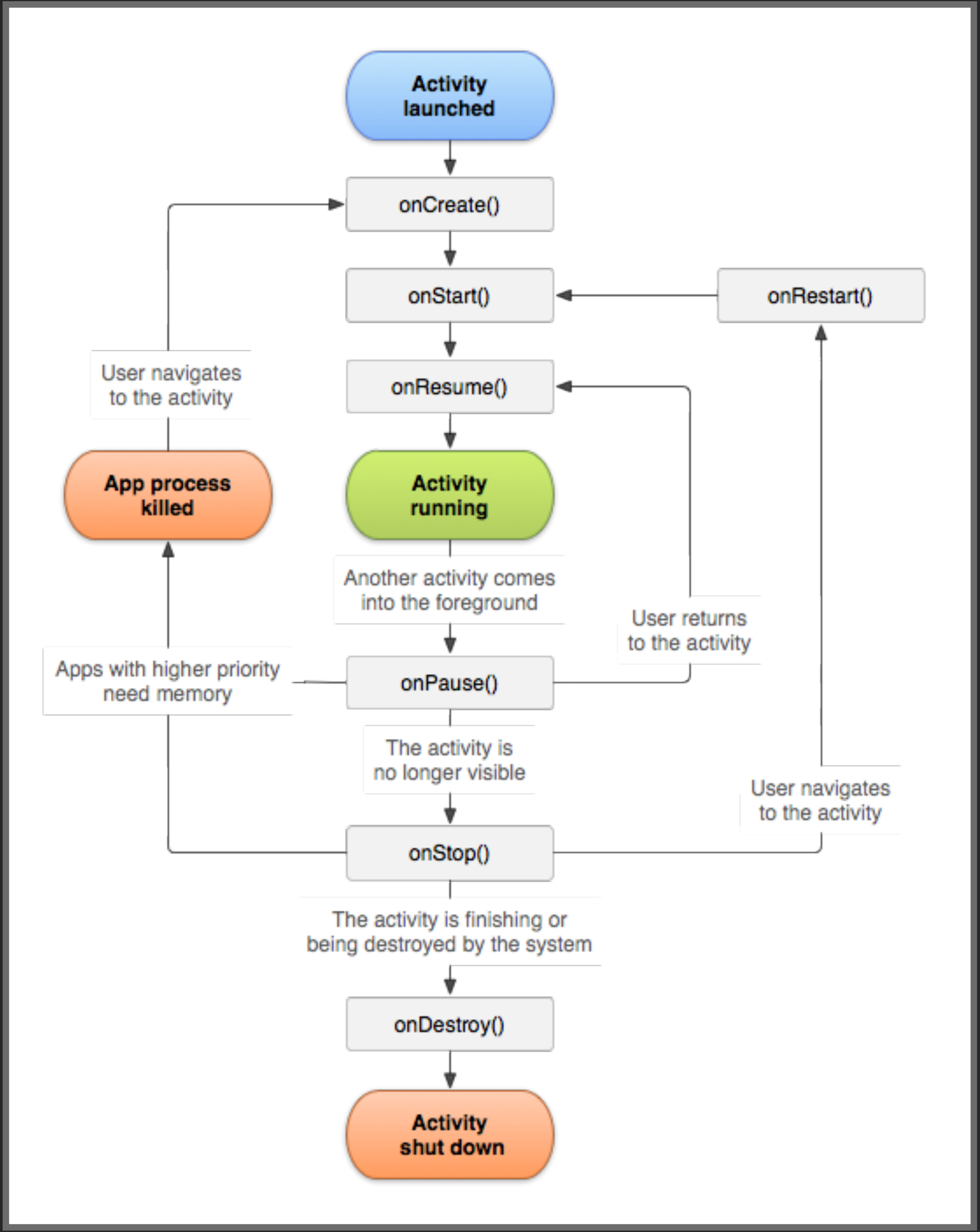
Android Activities

- Represents a single screen with a user interface
- Keeps track of what is currently on the screen
- Helps the app handle when its process gets killed, and allows the user to return to previous states
- Provides a way to implement user flows between various screens



Android Activity Lifecycle

- The Activity lifecycle dictates the various states that an activity could find itself in, and helps developers plan how each case should be handled.
- For instance, what should happen if the app process is killed? or what should happen if user navigates away from the activity and it is paused?

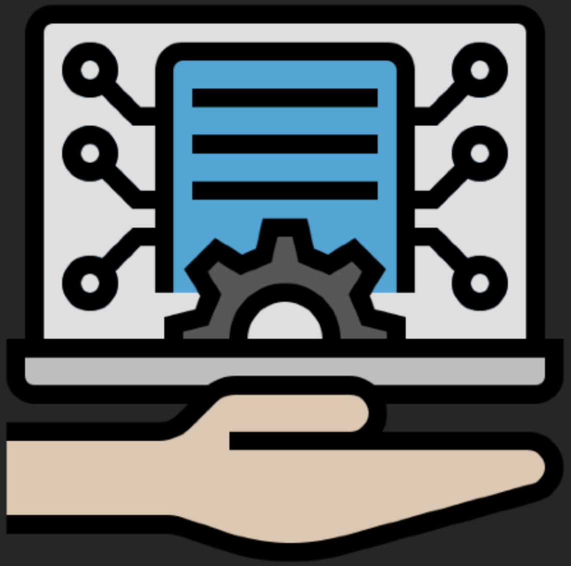


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Android Services

- Provides a mechanism for keeping an app running in the background for different purposes.
- Does not provide a user interface
- Once started, a service can continue running for some time, even after the user switches to another app.
- Example Use Cases:
    - Playing music in the background, handling network transactions, performing file I/O, interacting with other apps

Android Services



- Three Different Service Types:
    - *Foreground:* performs an operation that is noticeable to the user
    - *Background:* performs an operation that is not noticeable by the user
    - *Bound:* Offers a client-service interface that allows other app components to interact with the service

Android Services

- Using a Service vs. Using a Thread:
  - ***Services*** are only meant to be used when work needs to be done when the user is not interacting with your app
  - If you need to do some work outside the main UI thread of your application you should typically use a ***Thread***.

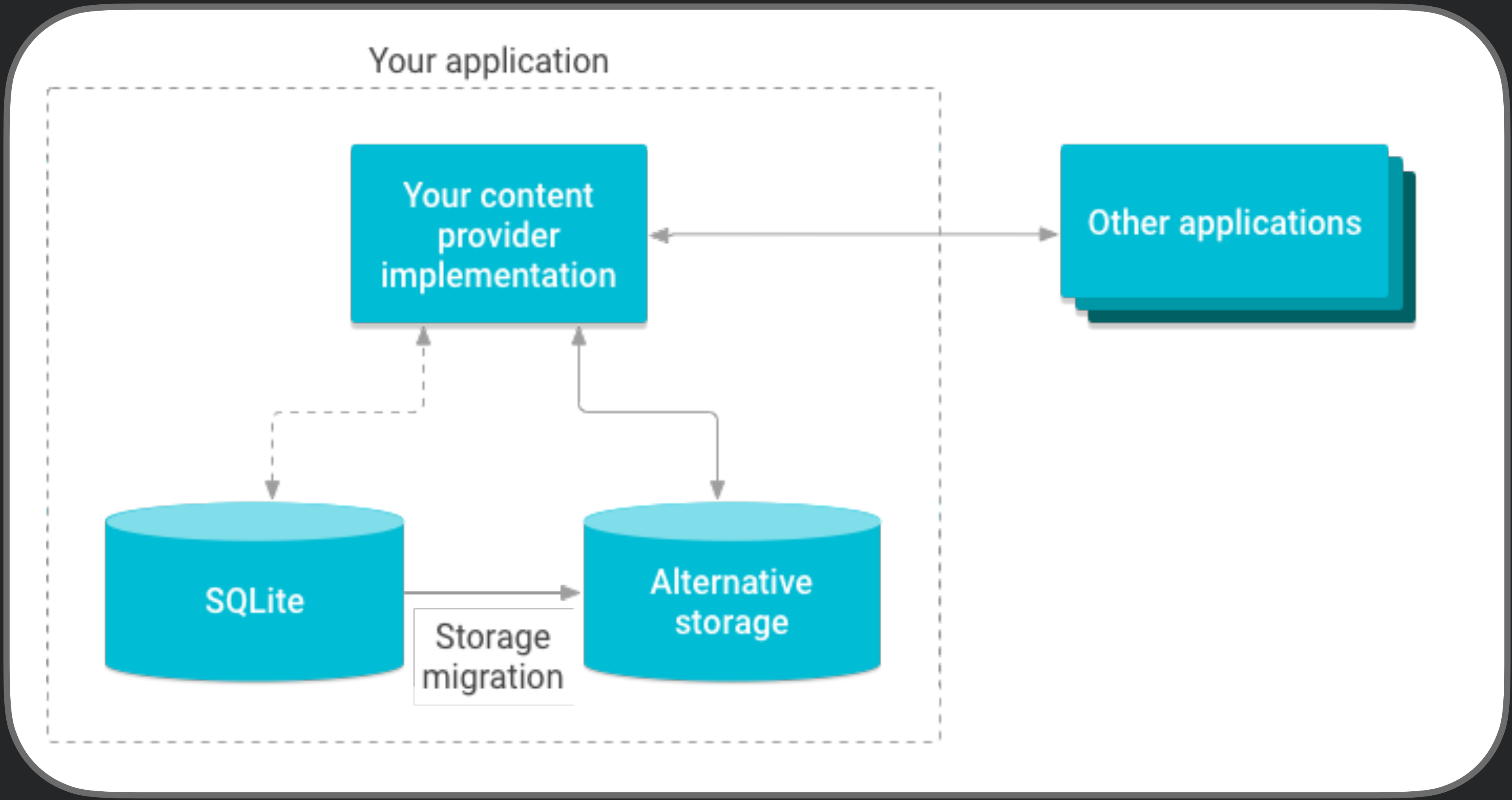
Android Broadcast Receivers

- Enables the system to deliver events to the app outside of a regular user flow allowing the app to respond to system wide broadcast announcements.
- Many broadcasts come from the system: screen turning off, battery getting low, picture was captured, etc.
- Apps can also initiate broadcasts
- No user interface, but broadcasts can create a status bar notification
- Typically used as gateways to other components and they should not do a lot of work

Android Content Providers

- Manages a shared set of app data that you can store in the file system, in a SQLite database, or any other persistent storage solution
- Other apps can query or modify data through the content provider - however the app must have proper permissions
- One example of this is managing a user's contacts
- Data is typically defined by a URI scheme

Android Content Providers



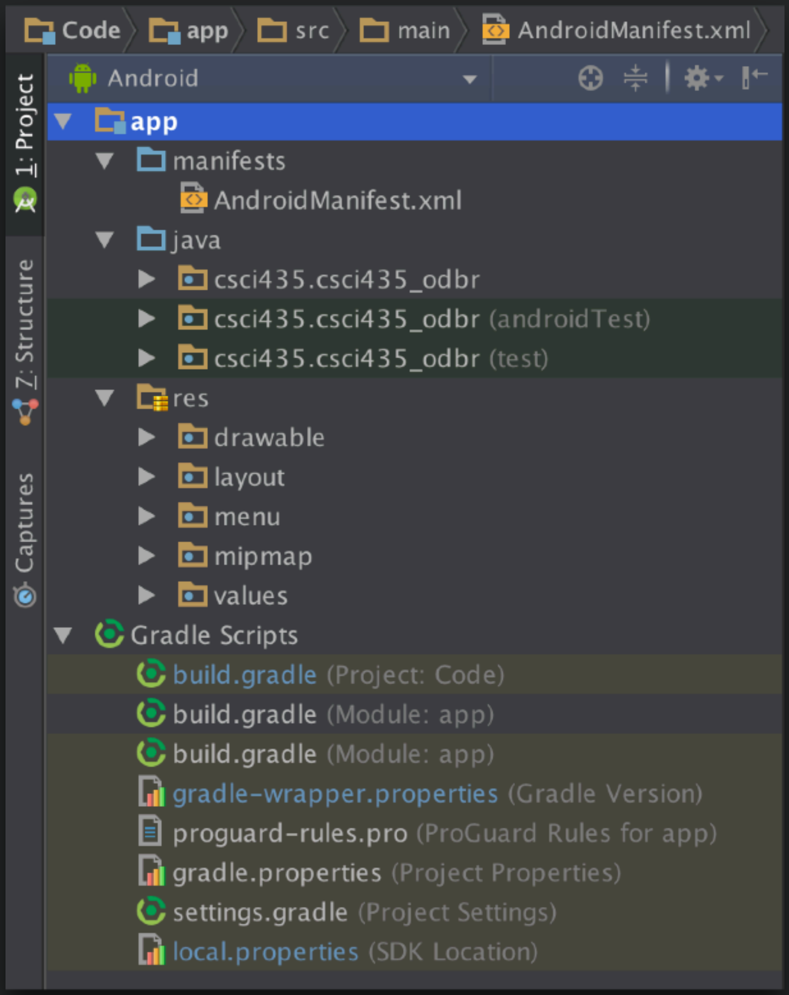
Activating Components

- Activities, Services, and Broadcast Receivers can be activated through asynchronous messages called intents.
- They essentially serve as messengers that request actions from other components
- Created with an `Intent` object, and can be implicit or explicit
- Content Providers are activated by `ContentResolvers` and not intents.

Android Project Structure

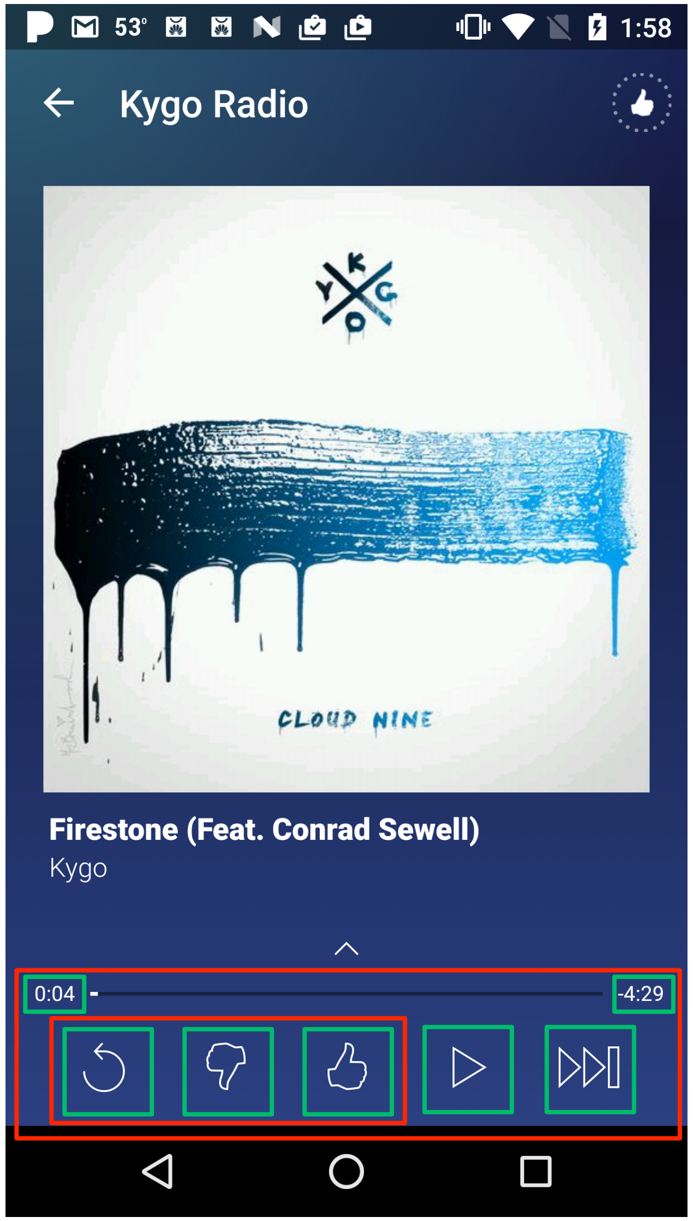
- AndroidManifest.xml
  - Overall project config and settings
- src/java/…
  - Source Code for your Java Classes
- res/…
  - **drawable/** = images
  - **layout/** = descriptions of GUI layout
  - **menu/** = overall app menu options
  - **values/** = constant values and arrays
  - **strings/** = localization data
  - **styles/** = general appearance styling

- Gradle
  - a build/compile management system
  - **build.gradle** = main build config file

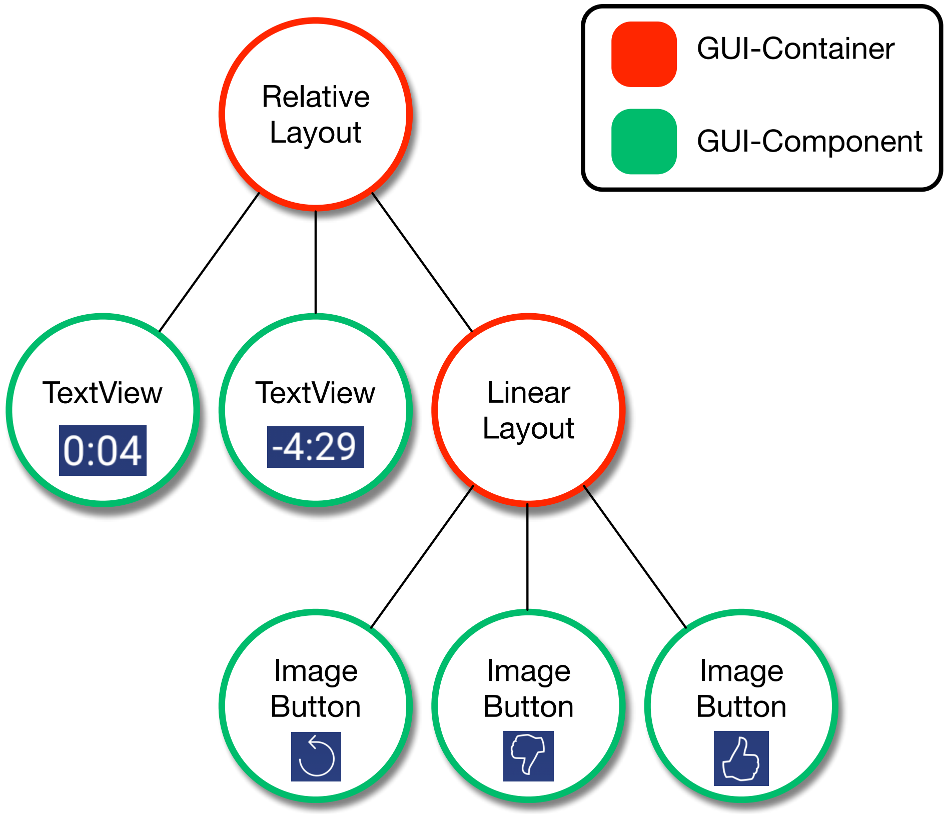


```
▼ app
  ▼ manifests
      AndroidManifest.xml
  ▼ java
    ▶ csci435.csci435_odbr
    ▶ csci435.csci435_odbr (androidTest)
    ▶ csci435.csci435_odbr (test)
  ▼ res
    ▶ drawable
    ▶ layout
    ▶ menu
    ▶ mipmap
    ▶ values
▼ Gradle Scripts
    build.gradle (Project: Code)
    build.gradle (Module: app)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Version)
    proguard-rules.pro (ProGuard Rules for app)
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

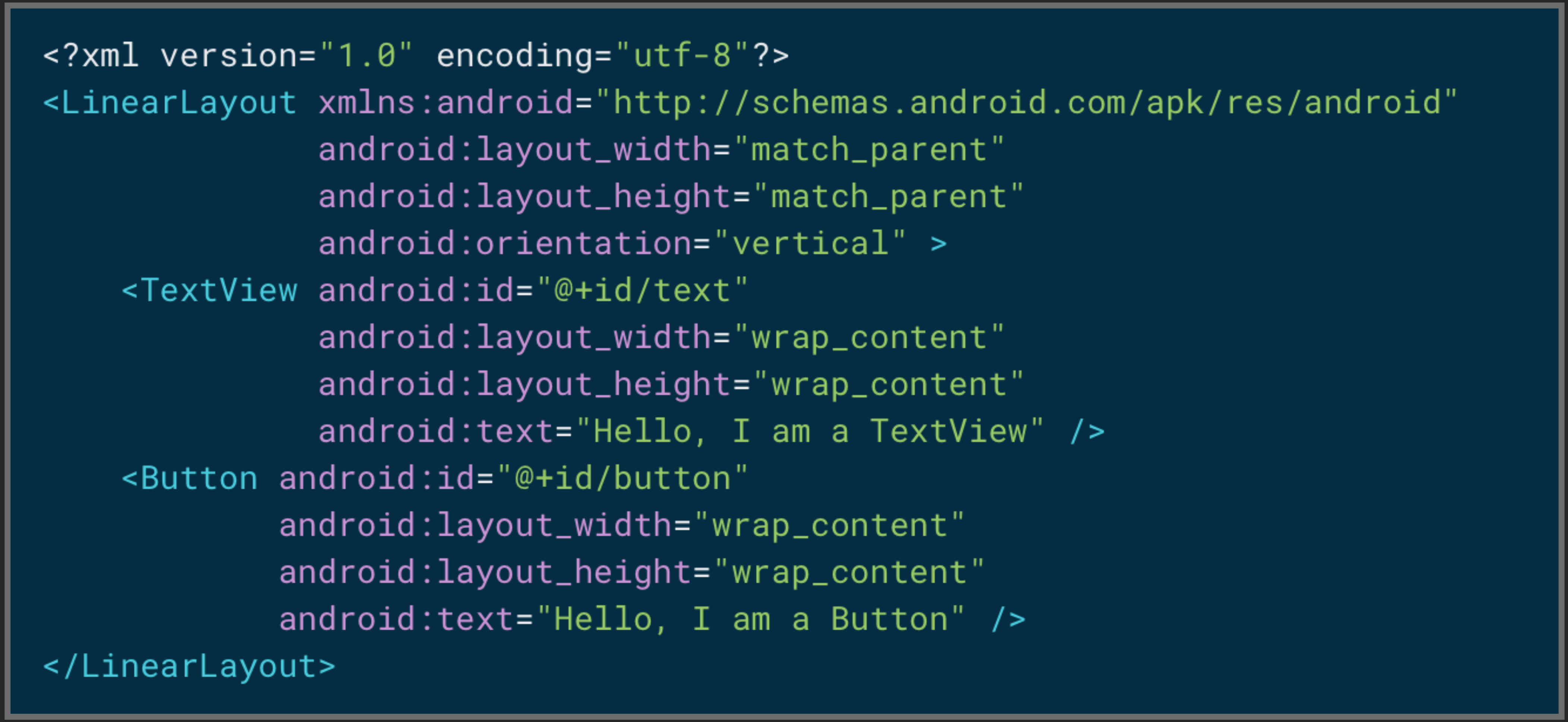
Android User Interface



Partial GUI Hierarchy
for the Pandora Application



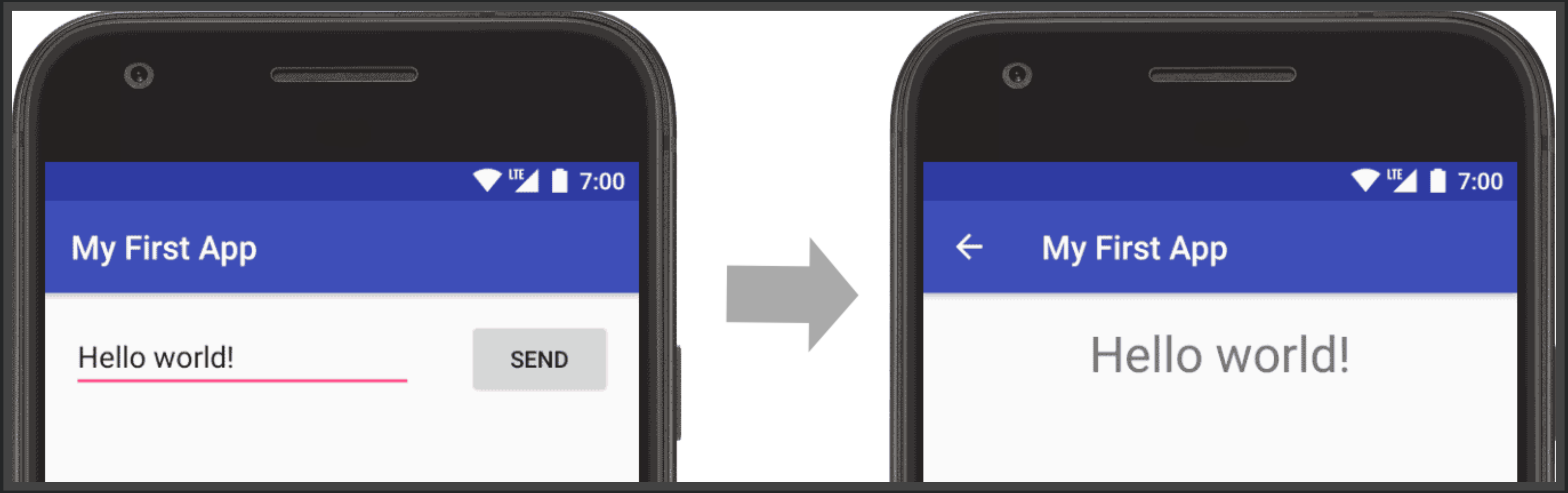
Android User Interface



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
              android:layout_width="match_parent"
              android:layout_height="match_parent"
              android:orientation="vertical" >
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button" />
</LinearLayout>
```

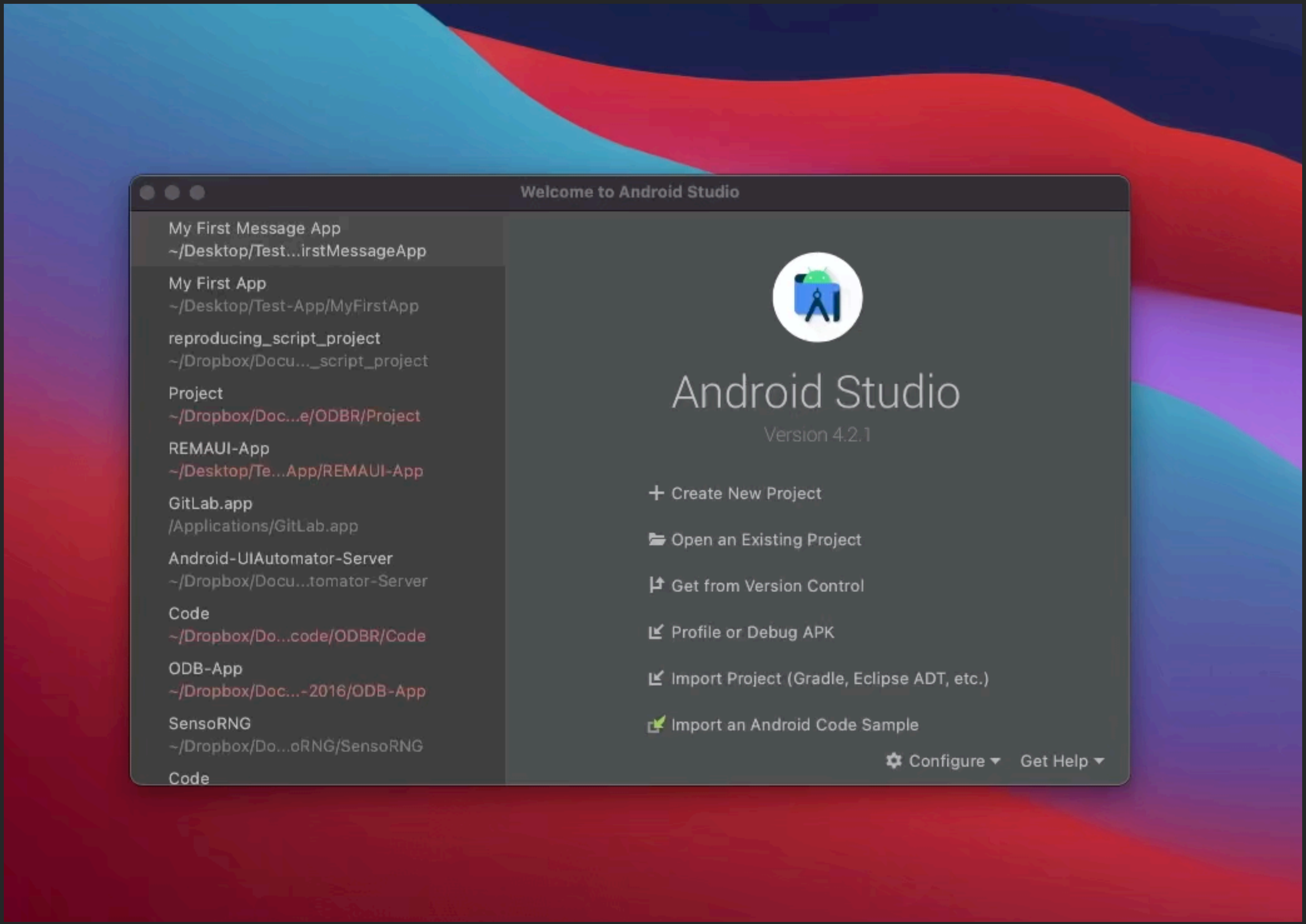
Building an Example Android App

Message Displaying App



https://developer.android.com/training/basics/firstapp

Creating a New Project



Welcome to Android Studio

My First Message App
~/Desktop/Test...irstMessageApp

My First App
~/Desktop/Test-App/MyFirstApp

reproducing_script_project
~/Dropbox/Docu..._script_project

Project
~/Dropbox/Doc...e/ODBR/Project

REMAUI-App
~/Desktop/Te...App/REMAUI-App

GitLab.app
/Applications/GitLab.app

Android-UIAutomator-Server
~/Dropbox/Docu...tomator-Server

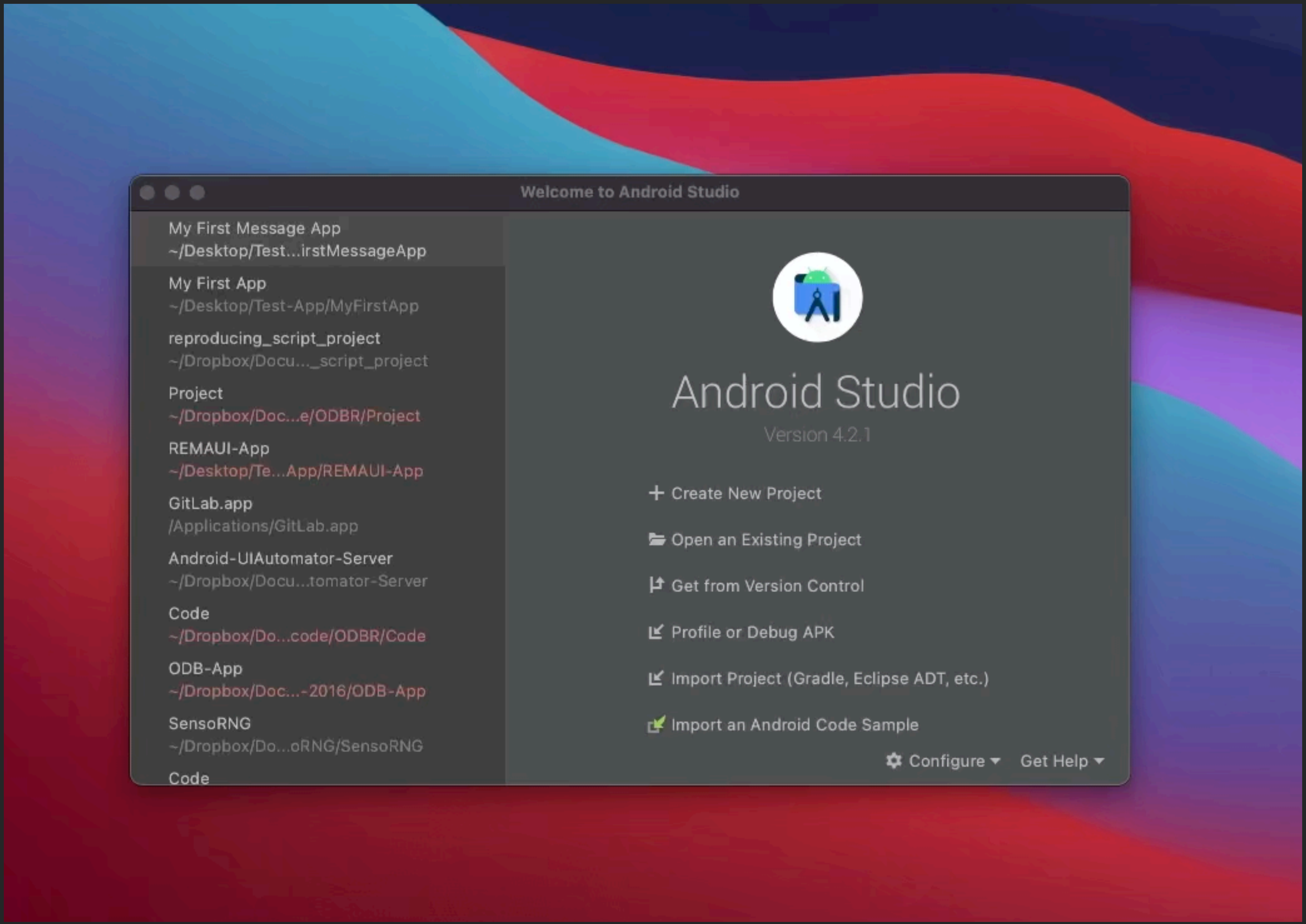
Code
~/Dropbox/Do...code/ODBR/Code

ODB-App
~/Dropbox/Doc...-2016/ODB-App

SensoRNG
~/Dropbox/Do...oRNG/SensoRNG

Code

Creating a New Project



Welcome to Android Studio

My First Message App
~/Desktop/Test...irstMessageApp

My First App
~/Desktop/Test-App/MyFirstApp

reproducing_script_project
~/Dropbox/Docu..._script_project

Project
~/Dropbox/Doc...e/ODBR/Project

REMAUI-App
~/Desktop/Te...App/REMAUI-App

GitLab.app
/Applications/GitLab.app

Android-UIAutomator-Server
~/Dropbox/Docu...tomator-Server

Code
~/Dropbox/Do...code/ODBR/Code

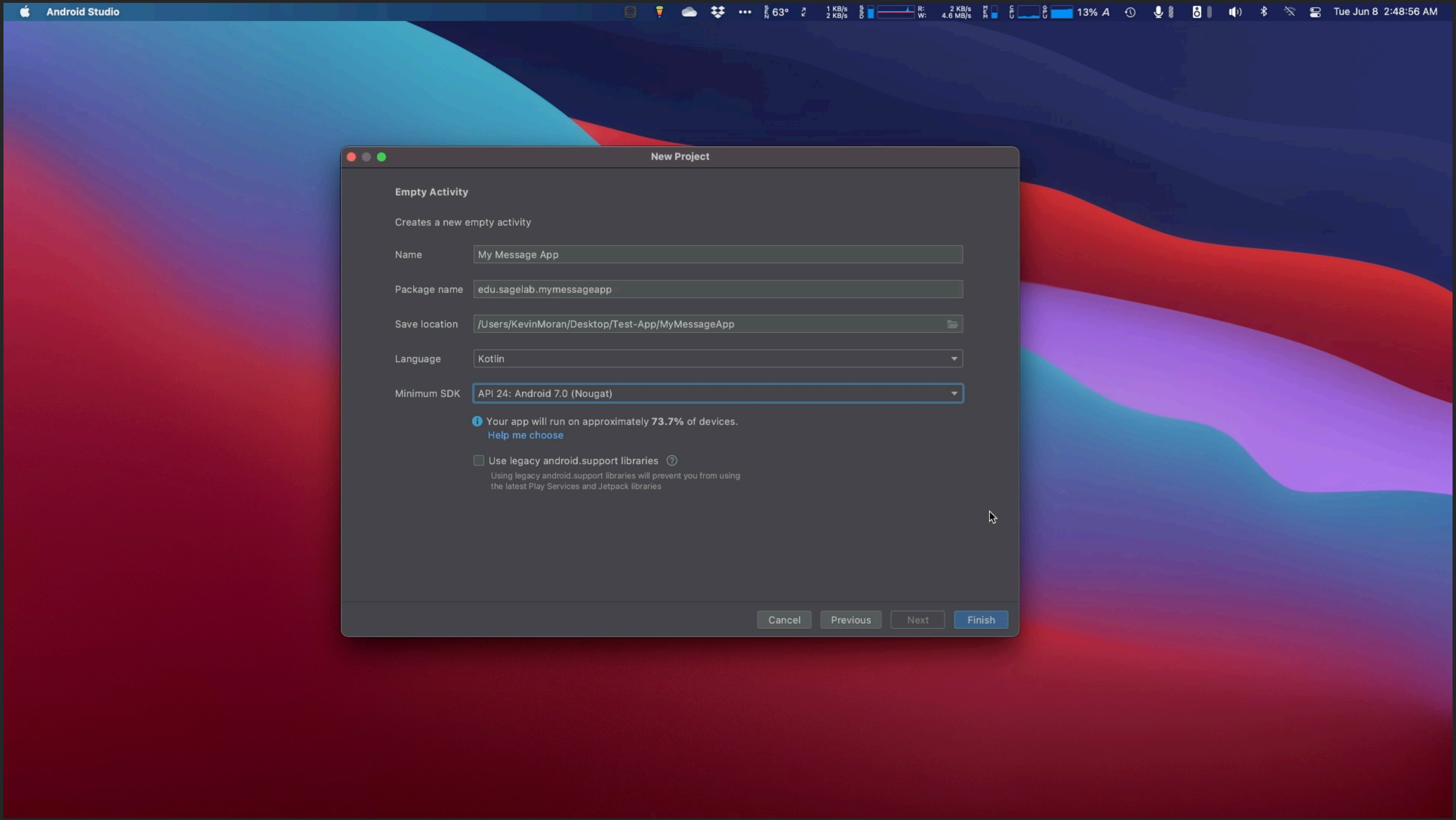
ODB-App
~/Dropbox/Doc...-2016/ODB-App

SensoRNG
~/Dropbox/Do...oRNG/SensoRNG

Code

Android Studio
Version 4.2.1

Running the App



New Project

Empty Activity

Creates a new empty activity

Name: My Message App

Package name: edu.sagelab.mymessageapp

Save location: /Users/KevinMoran/Desktop/Test-App/MyMessageApp

Language: Kotlin

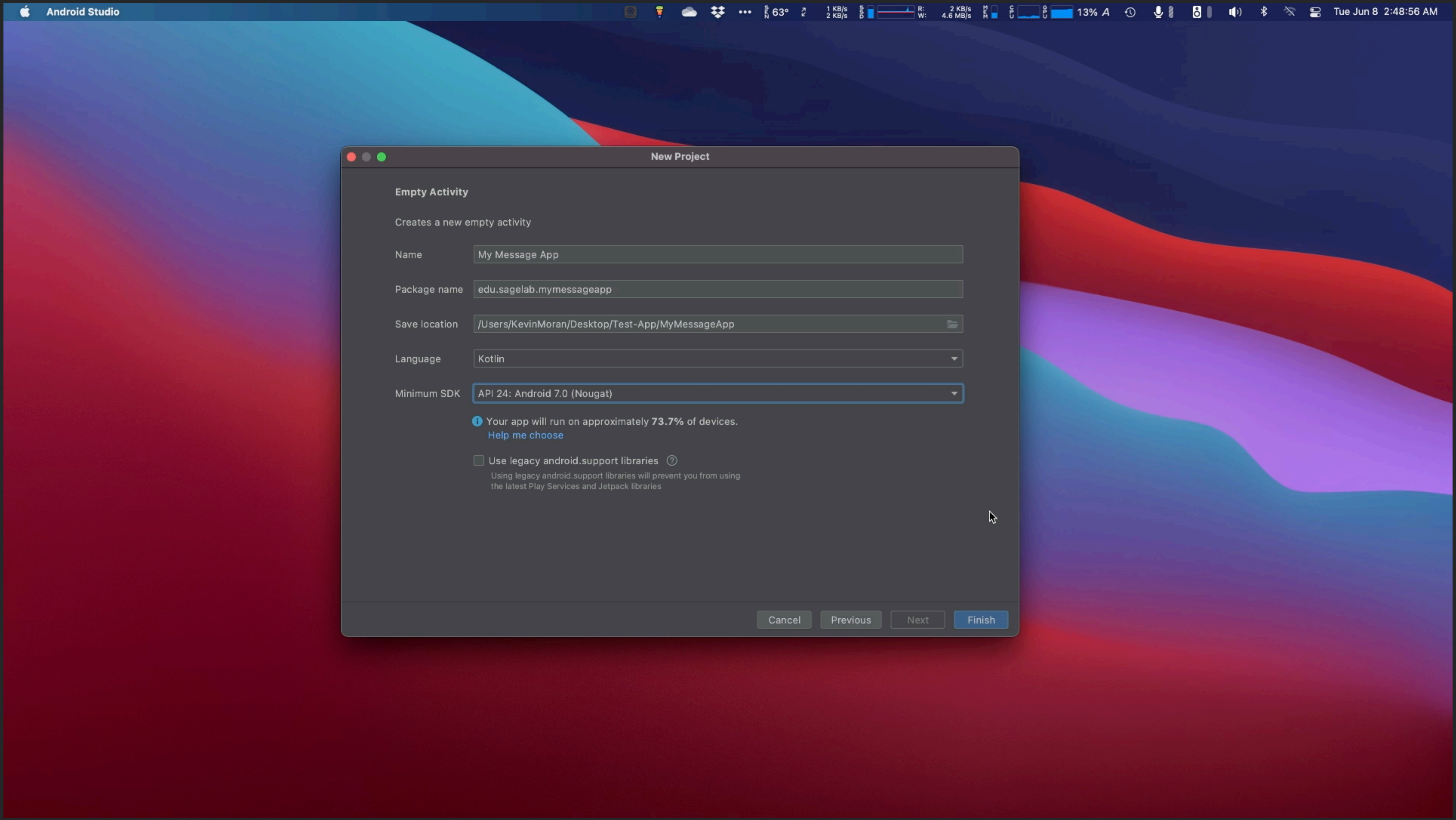
Minimum SDK: API 24: Android 7.0 (Nougat)

Your app will run on approximately **73.7%** of devices.
Help me choose

□ Use legacy android.support libraries

Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Running the App



New Project

Empty Activity

Creates a new empty activity

Name: My Message App

Package name: edu.sagelab.mymessageapp

Save location: /Users/KevinMoran/Desktop/Test-App/MyMessageApp

Language: Kotlin

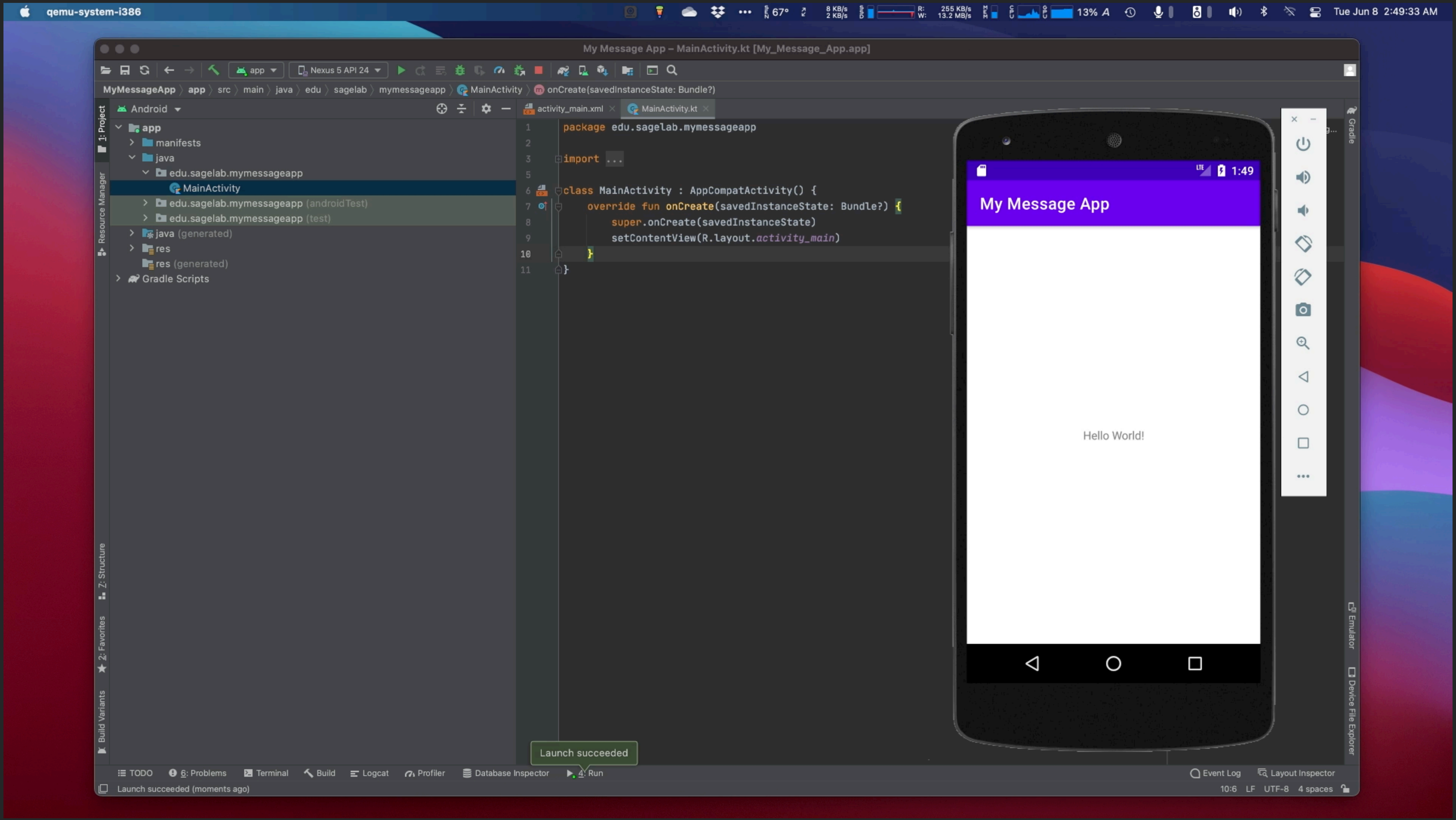
Minimum SDK: API 24: Android 7.0 (Nougat)

Your app will run on approximately **73.7%** of devices.
Help me choose

□ Use legacy android.support libraries

Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Building a Simple User Interface



My Message App – MainActivity.kt [My_Message_App.app]

MyMessageApp 〉 app 〉 src 〉 main 〉 java 〉 edu 〉 sagelab 〉 mymessageapp 〉 MainActivity 〉 onCreate(savedInstanceState: Bundle?)

```
package edu.sagelab.mymessageapp

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

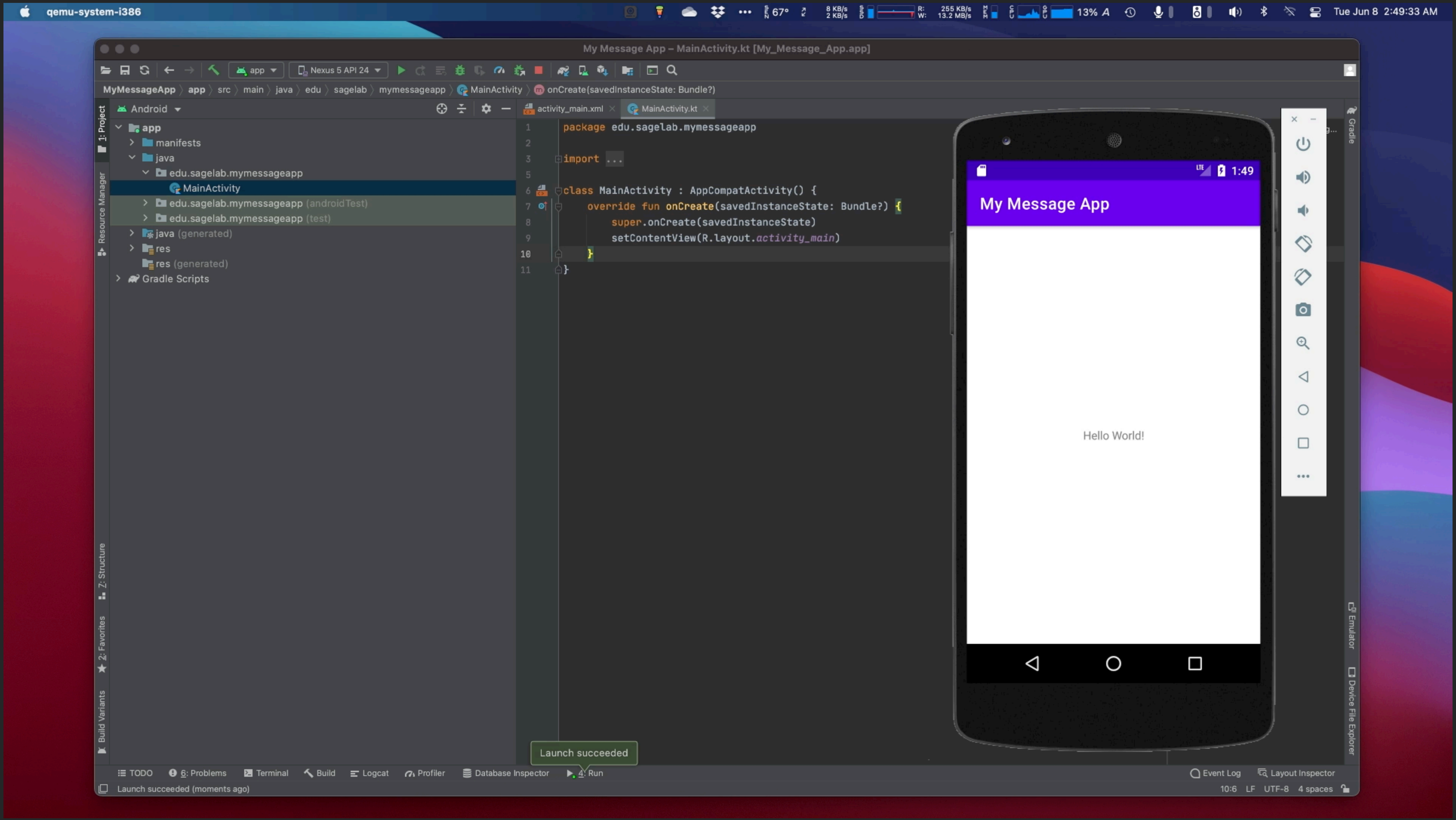
My Message App

Hello World!

Launch succeeded

Launch succeeded (moments ago)

Building a Simple User Interface



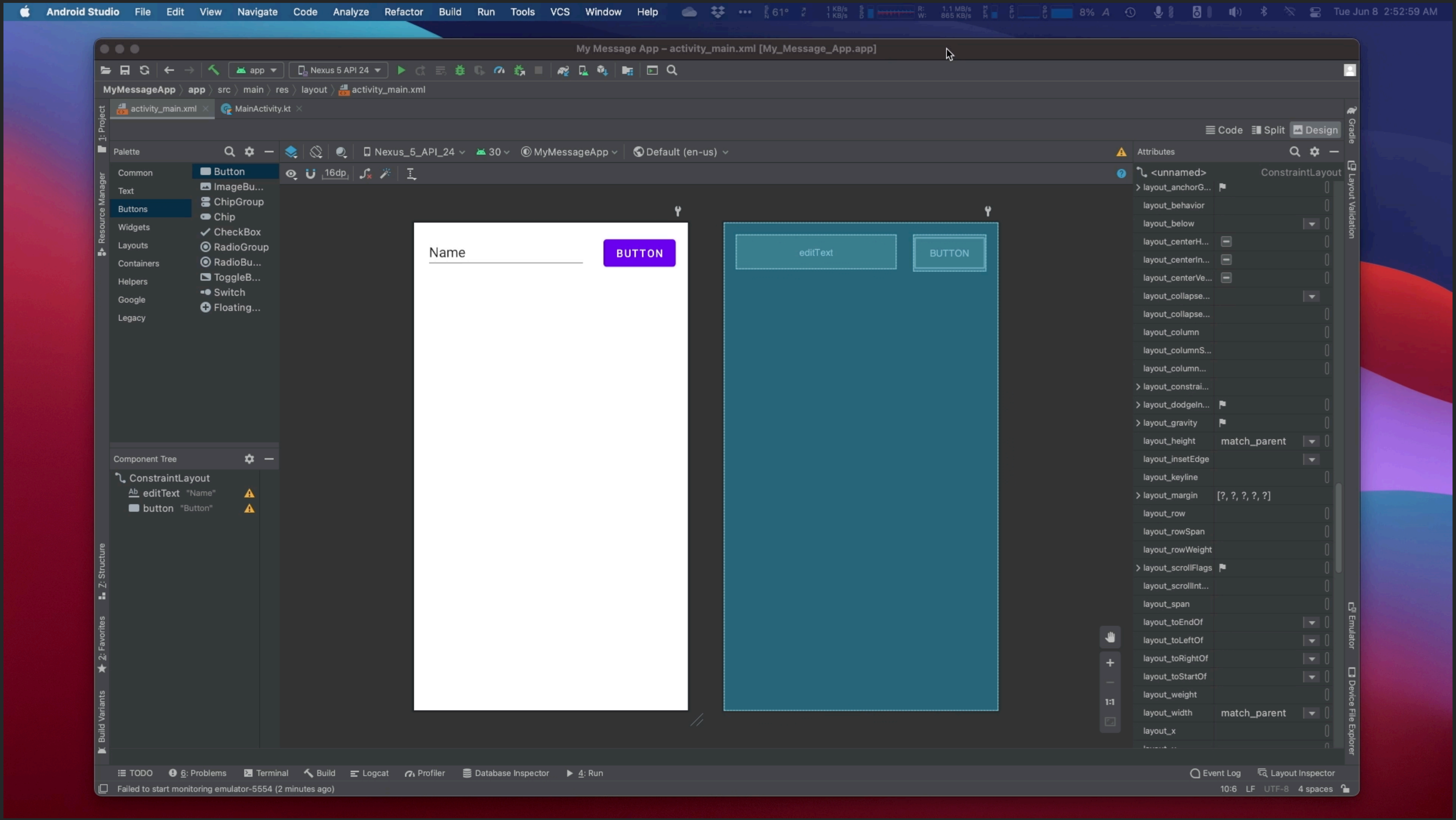
My Message App – MainActivity.kt [My_Message_App.app]

```
package edu.sagelab.mymessageapp

import ...

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Launch succeeded

Launch succeeded (moments ago)

Changing the UI Strings



My Message App – activity_main.xml [My_Message_App.app]

MyMessageApp 〉 app 〉 src 〉 main 〉 res 〉 layout 〉 activity_main.xml

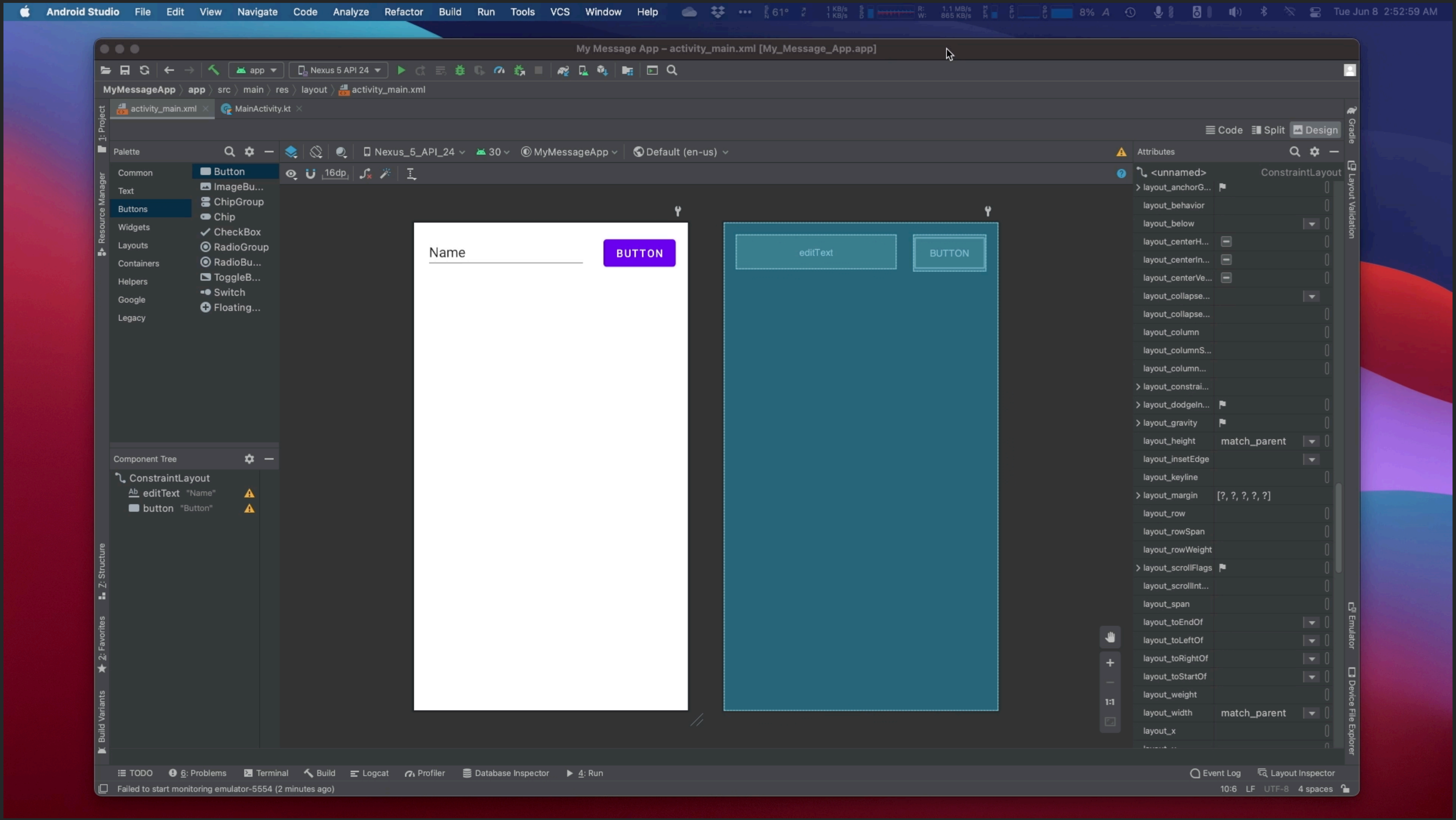
Component Tree
- ConstraintLayout
  - editText "Name"
  - button "Button"

Attributes: <unnamed> ConstraintLayout

layout_height	match_parent
layout_margin	[?, ?, ?, ?, ?]
layout_width	match_parent

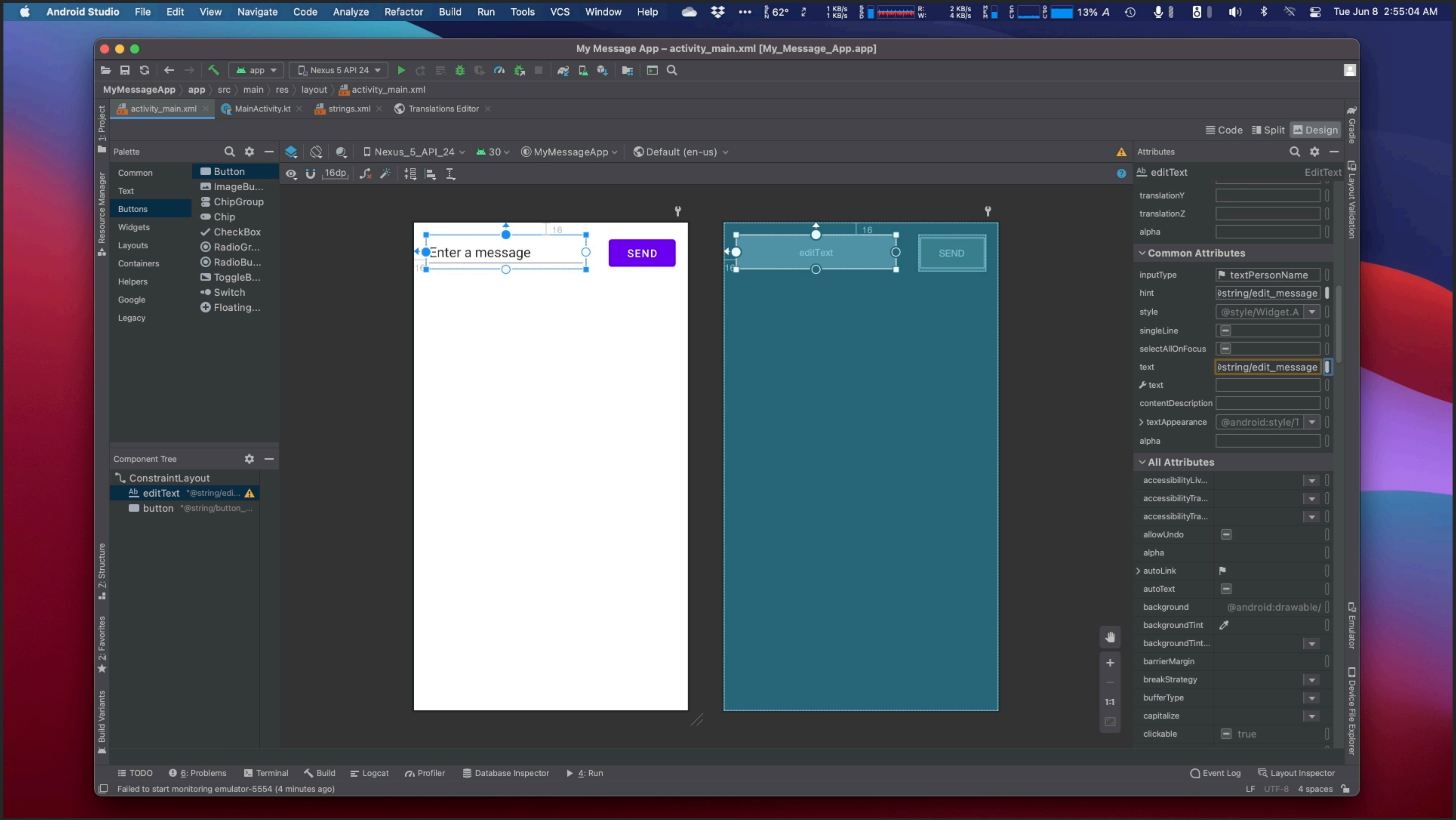
Failed to start monitoring emulator-5554 (2 minutes ago)

Changing the UI Strings



Failed to start monitoring emulator-5554 (2 minutes ago)

Making the Text Box Size Flexible



My Message App – activity_main.xml [My_Message_App.app]

MyMessageApp 〉 app 〉 src 〉 main 〉 res 〉 layout 〉 activity_main.xml

Nexus_5_API_24 · 30 · MyMessageApp · Default (en-us)

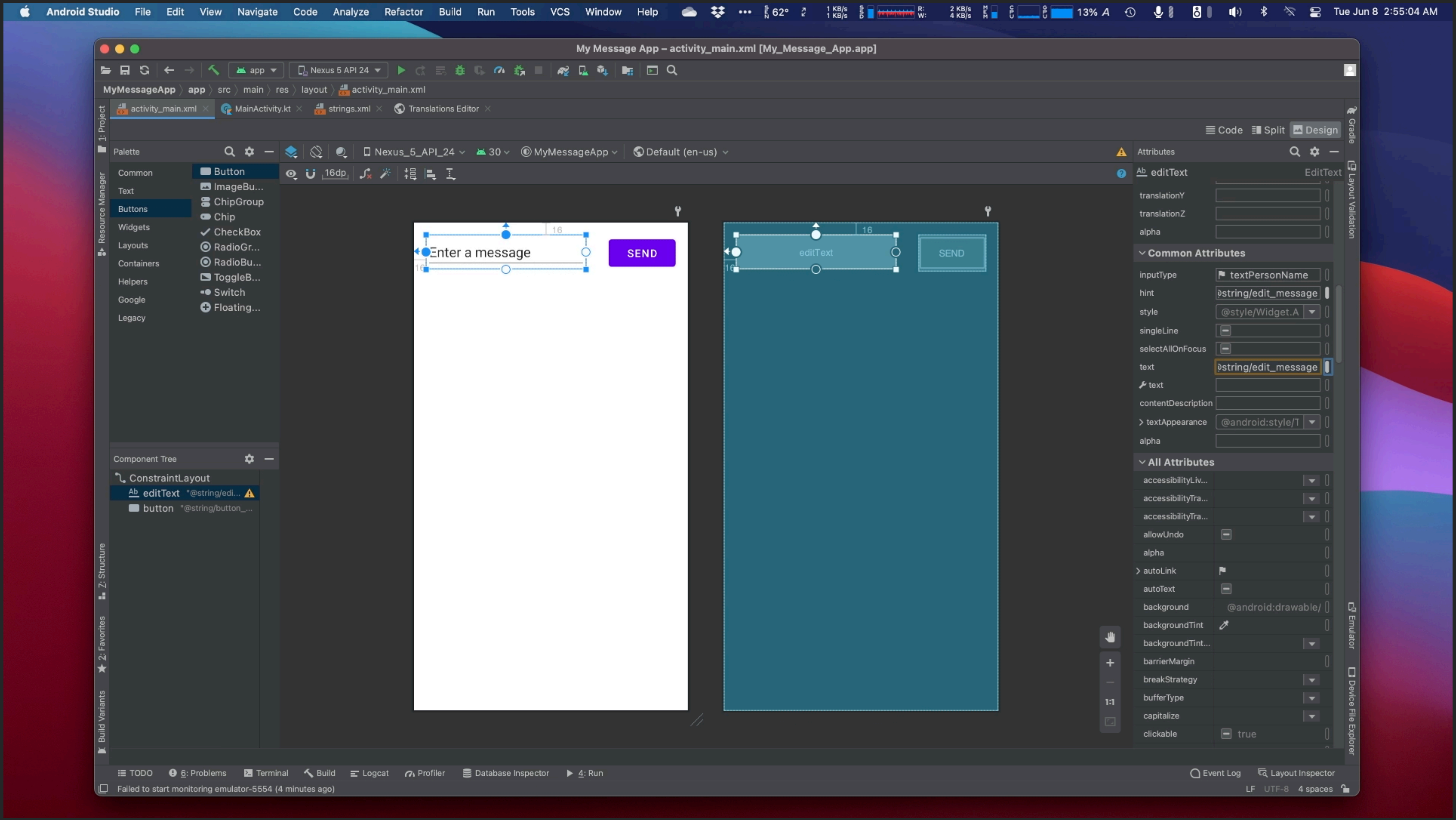
Attributes: editText — EditText

inputType	textPersonName
hint	@string/edit_message
style	@style/Widget.A
text	@string/edit_message
textAppearance	@android:style/T
background	@android:drawable/
clickable	true

Component Tree
- ConstraintLayout
  - editText "@string/edi...
  - button "@string/button_...

Failed to start monitoring emulator-5554 (4 minutes ago)

Making the Text Box Size Flexible



My Message App – activity_main.xml [My_Message_App.app]

MyMessageApp 〉 app 〉 src 〉 main 〉 res 〉 layout 〉 activity_main.xml

Attributes: editText EditText

translationY	
translationZ	
alpha	

Common Attributes

inputType	textPersonName
hint	@string/edit_message
style	@style/Widget.A
singleLine	
selectAllOnFocus	
text	@string/edit_message
text	
contentDescription	
textAppearance	@android:style/T
alpha	

All Attributes

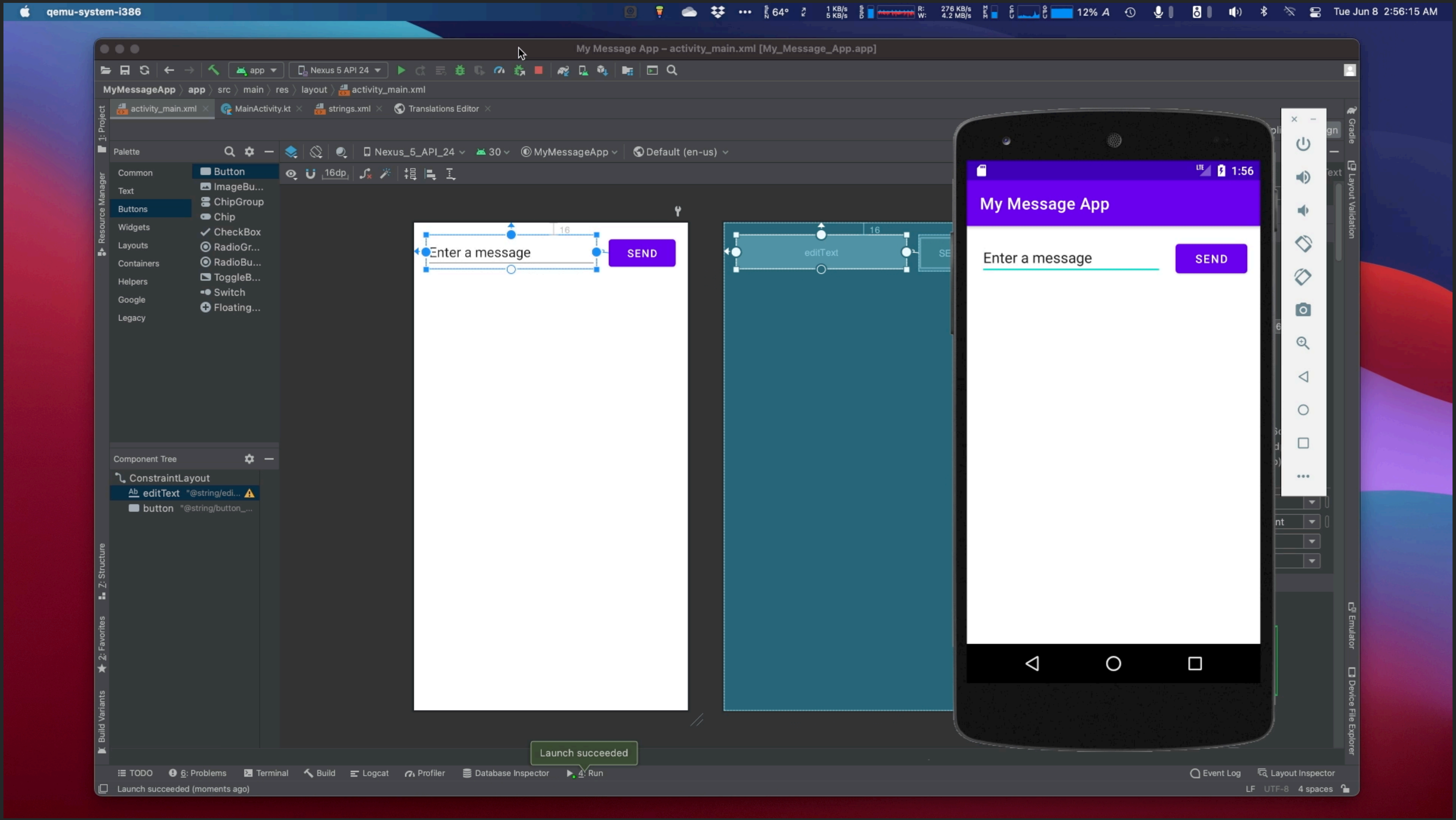
accessibilityLiv...	
accessibilityTra...	
accessibilityTra...	
allowUndo	
alpha	
autoLink	
autoText	
background	@android:drawable/
backgroundTint	
backgroundTint...	
barrierMargin	
breakStrategy	
bufferType	
capitalize	
clickable	true

Component Tree

```
ConstraintLayout
  editText "@string/edi...
  button "@string/button_...
```

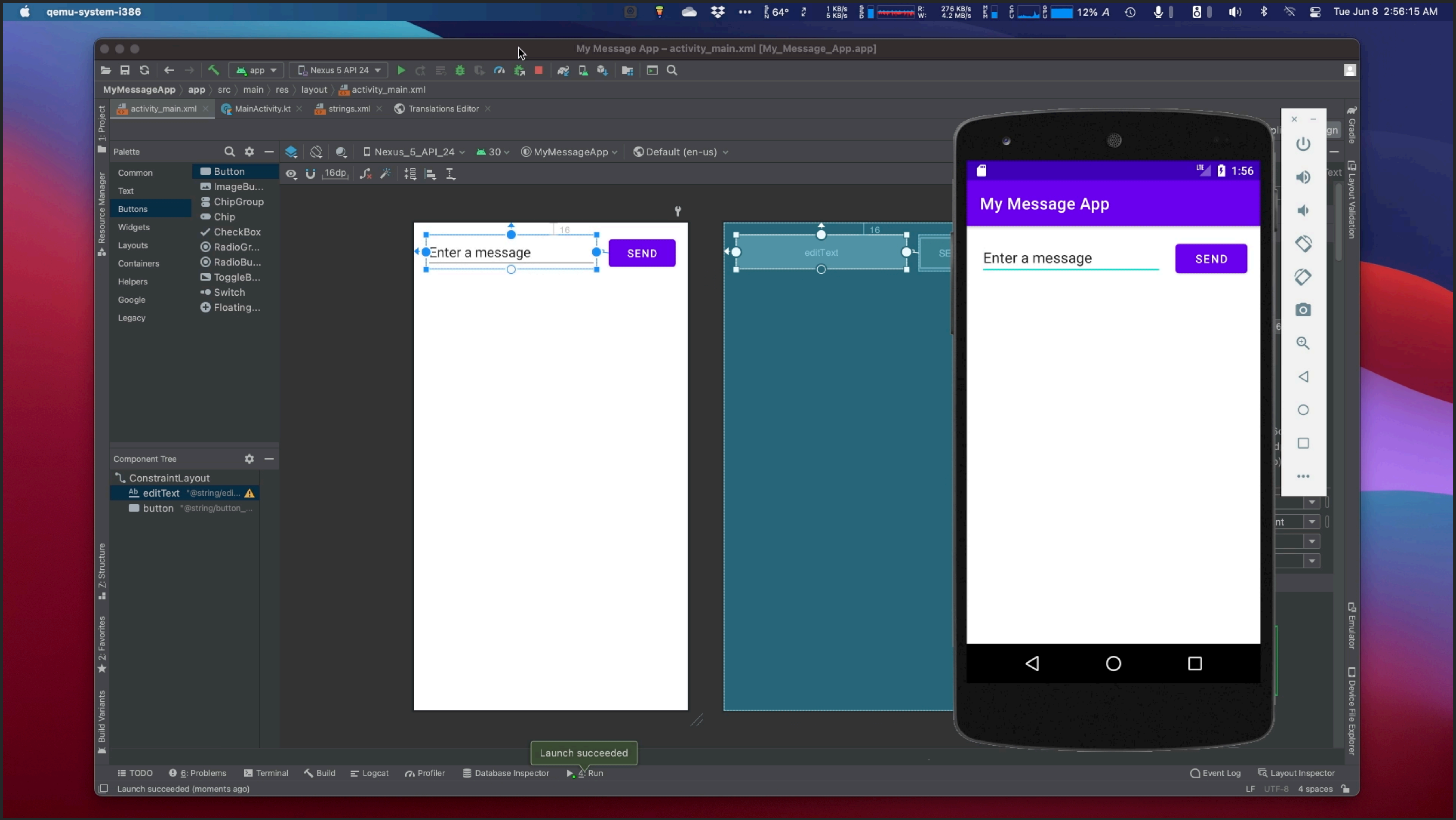
Failed to start monitoring emulator-5554 (4 minutes ago)

Starting Another Activity

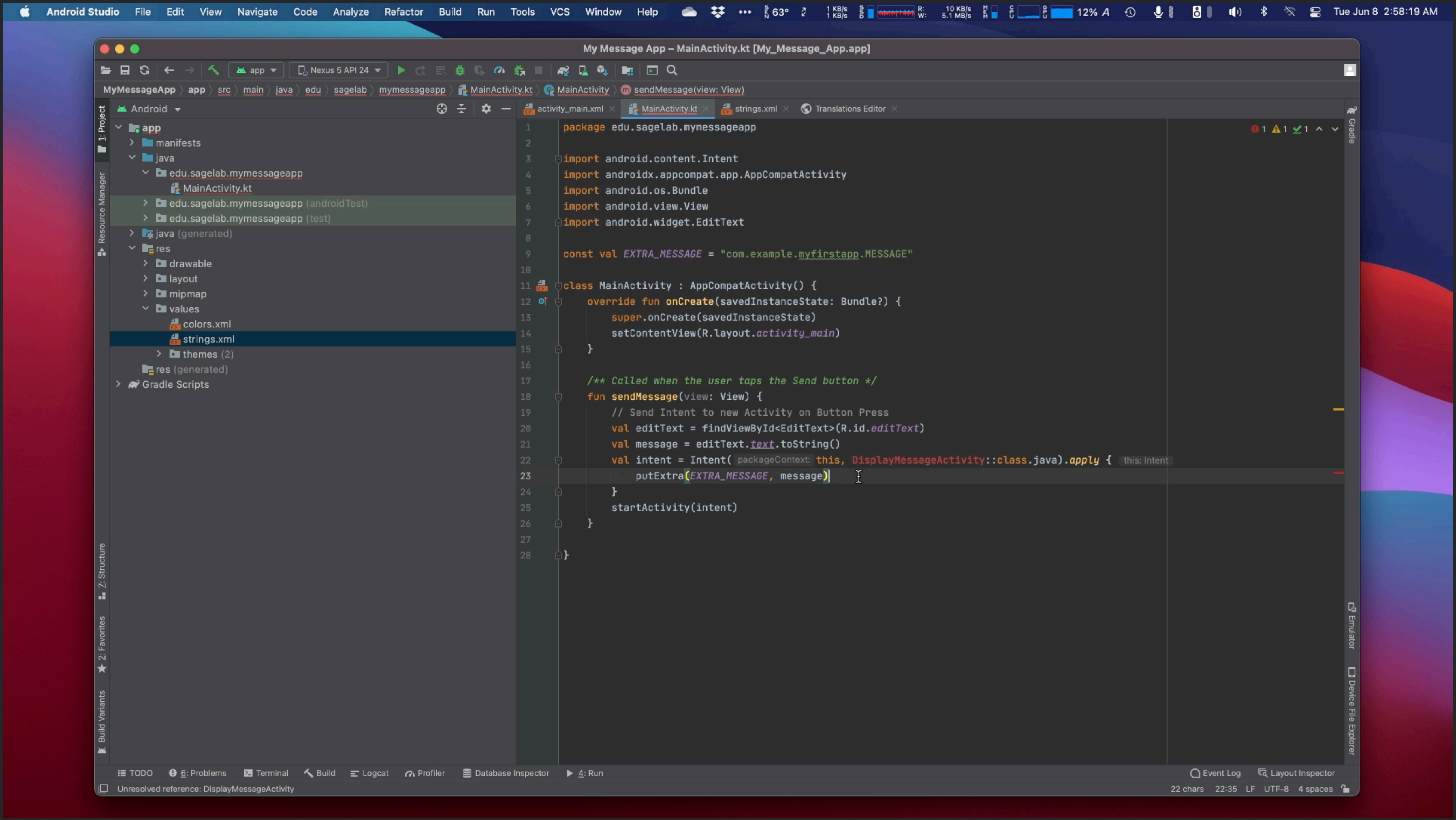


Launch succeeded (moments ago)

Starting Another Activity



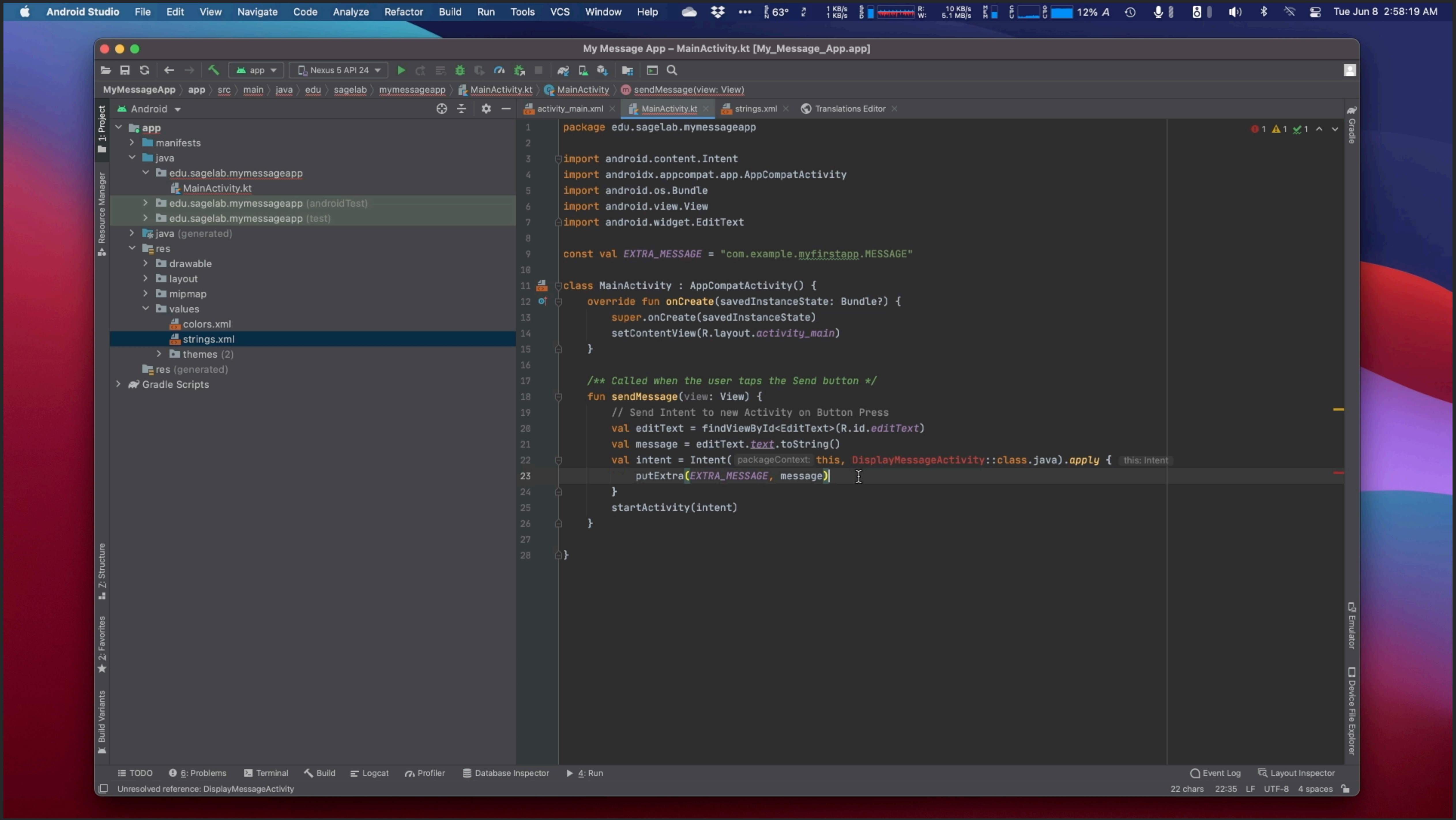
Creating the Second Activity



```kotlin
package edu.sagelab.mymessageapp

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.EditText

const val EXTRA_MESSAGE = "com.example.myfirstapp.MESSAGE"

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    /** Called when the user taps the Send button */
    fun sendMessage(view: View) {
        // Send Intent to new Activity on Button Press
        val editText = findViewById<EditText>(R.id.editText)
        val message = editText.text.toString()
        val intent = Intent( packageContext: this, DisplayMessageActivity::class.java).apply {  this: Intent
            putExtra(EXTRA_MESSAGE, message)
        }
        startActivity(intent)
    }

}
```

Unresolved reference: DisplayMessageActivity

Creating the Second Activity



My Message App – MainActivity.kt [My_Message_App.app]

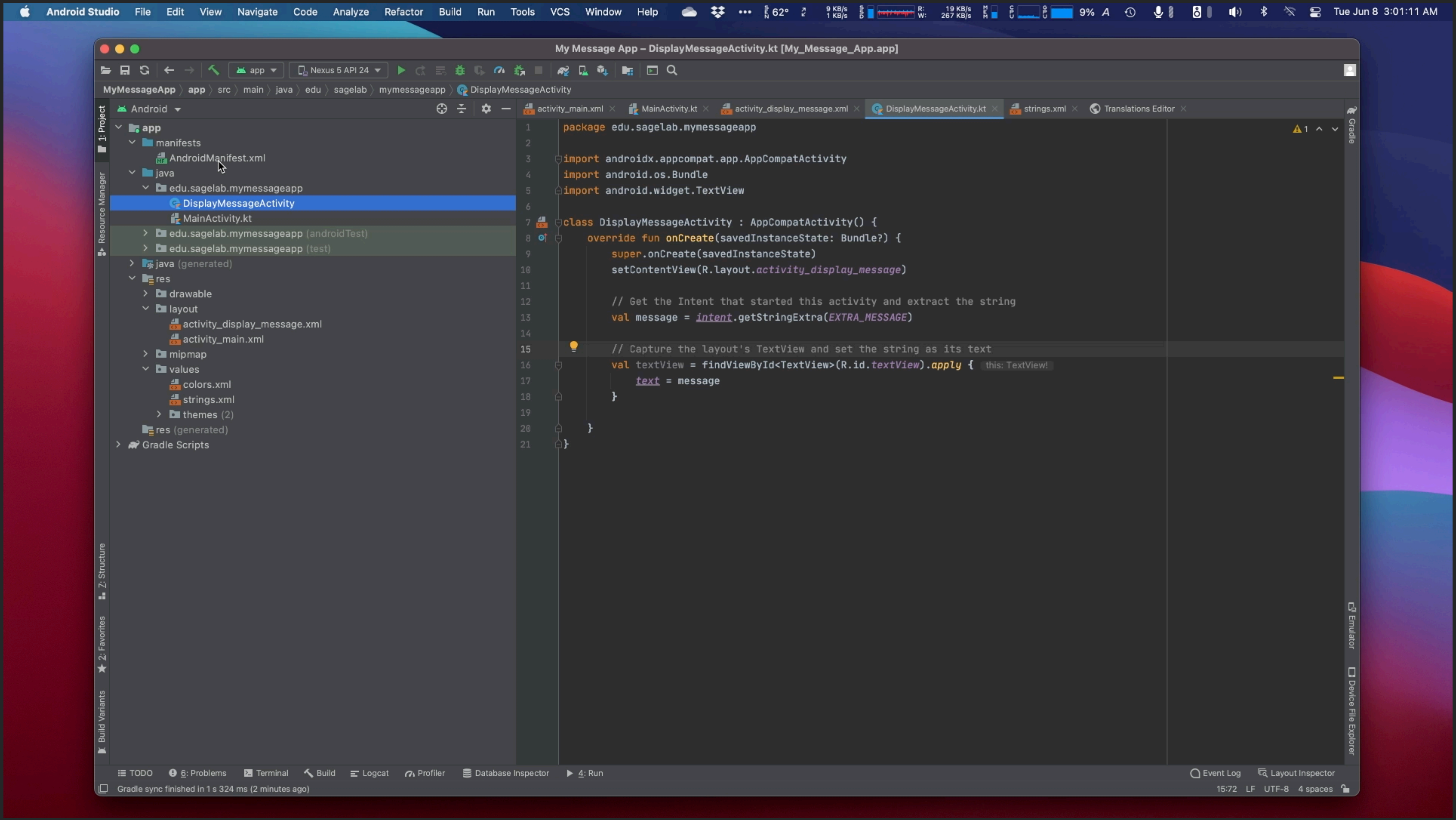
MyMessageApp 〉 app 〉 src 〉 main 〉 java 〉 edu 〉 sagelab 〉 mymessageapp 〉 MainActivity.kt 〉 MainActivity 〉 sendMessage(view: View)

```
app
  manifests
  java
    edu.sagelab.mymessageapp
      MainActivity.kt
    edu.sagelab.mymessageapp (androidTest)
    edu.sagelab.mymessageapp (test)
  java (generated)
  res
    drawable
    layout
    mipmap
    values
      colors.xml
      strings.xml
      themes (2)
  res (generated)
Gradle Scripts
```

```kotlin
package edu.sagelab.mymessageapp

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.EditText

const val EXTRA_MESSAGE = "com.example.myfirstapp.MESSAGE"

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    /** Called when the user taps the Send button */
    fun sendMessage(view: View) {
        // Send Intent to new Activity on Button Press
        val editText = findViewById<EditText>(R.id.editText)
        val message = editText.text.toString()
        val intent = Intent(this, DisplayMessageActivity::class.java).apply {
            putExtra(EXTRA_MESSAGE, message)
        }
        startActivity(intent)
    }

}
```

Unresolved reference: DisplayMessageActivity

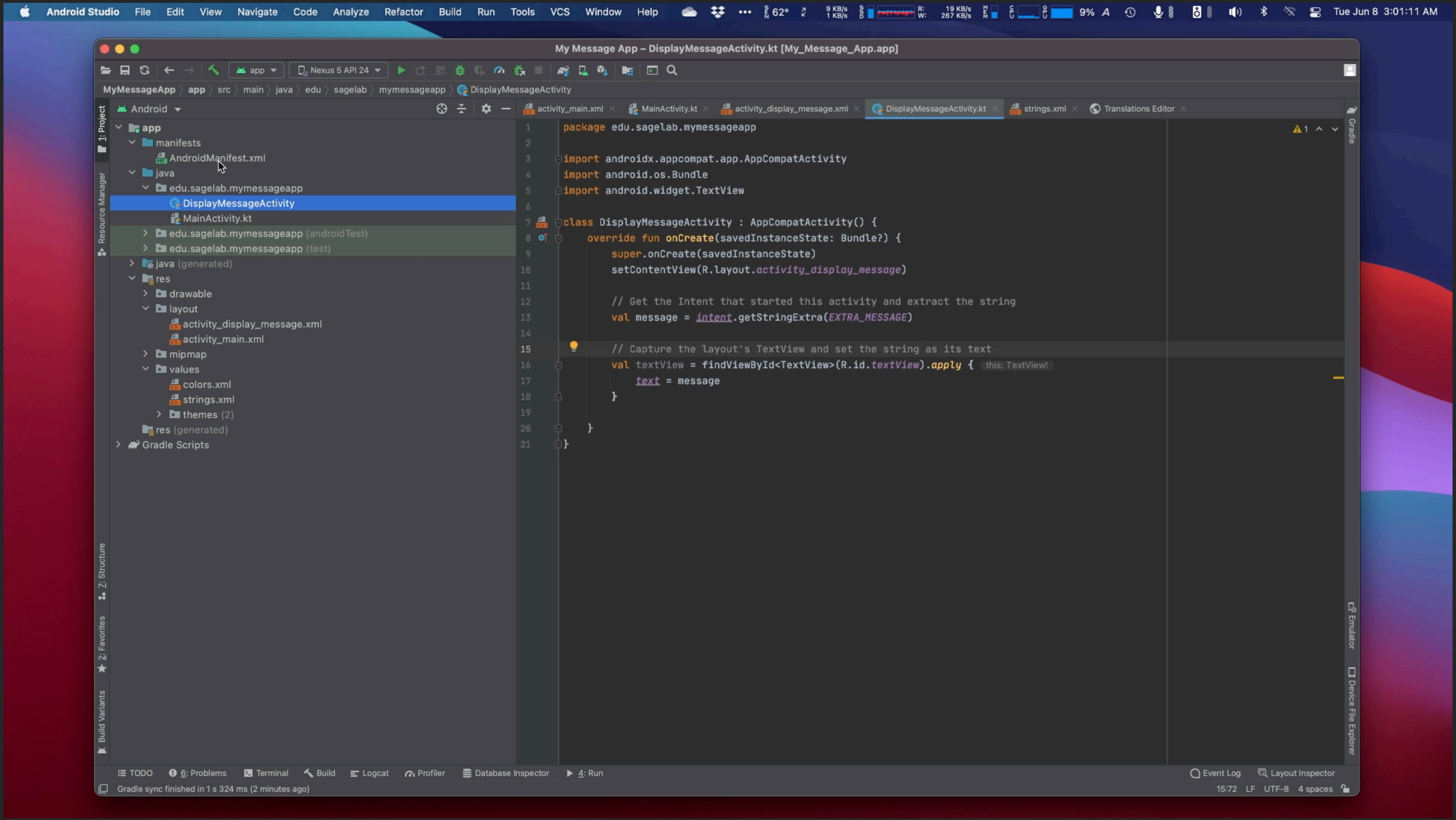
Configuring Resources & Testing!



```kotlin
package edu.sagelab.mymessageapp

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.TextView

class DisplayMessageActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_display_message)

        // Get the Intent that started this activity and extract the string
        val message = intent.getStringExtra(EXTRA_MESSAGE)

        // Capture the layout's TextView and set the string as its text
        val textView = findViewById<TextView>(R.id.textView).apply {  this: TextView!
            text = message
        }

    }
}
```

Gradle sync finished in 1 s 324 ms (2 minutes ago)

Configuring Resources & Testing!



```kotlin
package edu.sagelab.mymessageapp

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.TextView

class DisplayMessageActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_display_message)

        // Get the Intent that started this activity and extract the string
        val message = intent.getStringExtra(EXTRA_MESSAGE)

        // Capture the layout's TextView and set the string as its text
        val textView = findViewById<TextView>(R.id.textView).apply {  this: TextView!
            text = message
        }

    }
}
```

Gradle sync finished in 1 s 324 ms (2 minutes ago)

10 Minute Break

REU & NRT Android App Development Tutorial

George Mason University

Instructor:
Dr. Kevin Moran

Tutorial will start in:

10:00

Say Hi in the Chat!

REU & NRT Android App Development Tutorial

George Mason University

Instructor:
Dr. Kevin Moran

Tutorial will start in:

10:00

Say Hi in the Chat!

Android Dev Tutorial

- Breakout Rooms
- Different options:
  - Work independently, but share information
  - One person shares screen, work collaboratively
- Follow the tutorial on https://sagelab.io/android-dev-tutorial/
- You can refer to the GitHub project if you get stuck!

Acknowledgements

- https://developer.android.com/training/basics/firstapp
- https://developer.android.com