COMP90018
Mobile Computing Systems Programming

Week 1: Introduction, Android Studio and Git

Mohammed Atiq Mohammed Mashaq Shaikh

>>>>>> **ATIQ** <<<<<<

shaikhm@student.unimelb.edu.au

THE UNIVERSITY OF MELBOURNE

Outline

1. Introduction
2. Android Studio
3. Git

1. Introduction

1. Introduction – Android

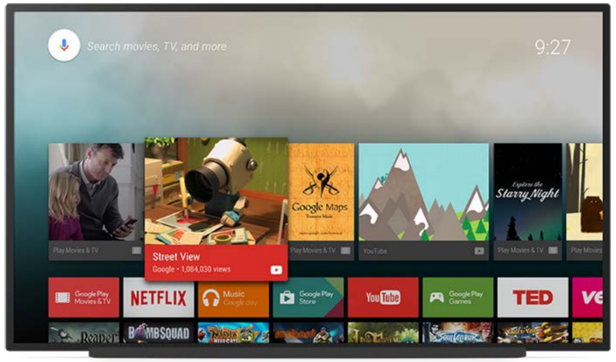
- A mobile operating system developed by **Google**;
- Based on a modified version of the **Linux kernel**;
- Currently version Android 9 (**Pie** – API level 28);
- Beta version Android 10 (Q – API level 29).

Introduction – Android Platforms

Phone

Watch

Tablet



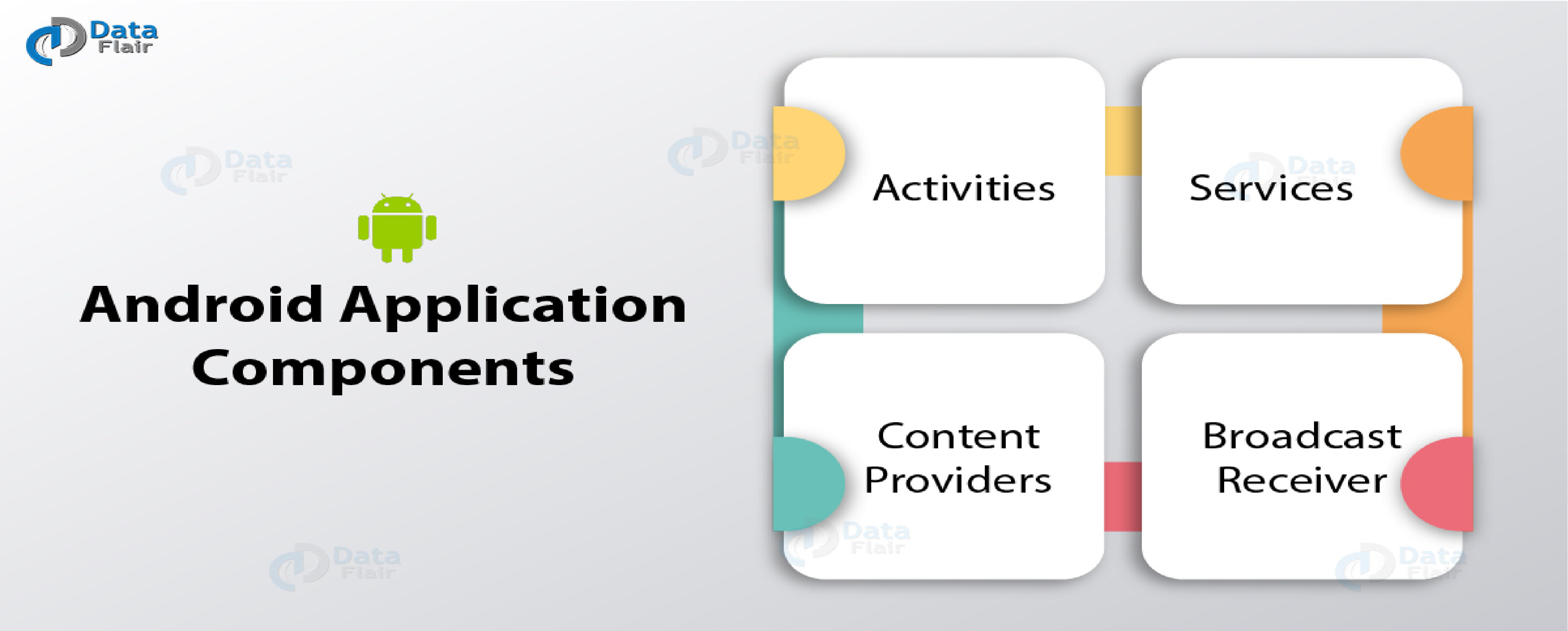
TV

Introduction – Android Applications

- Android apps can be written *using **Kotlin, Java, and C++***;

- The Android SDK tools compile your code along with any data and resource files into an **APK** *(Android package)* with an .apk suffix.

Introduction – Android Applications



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

THE UNIVERSITY OF
MELBOURNE

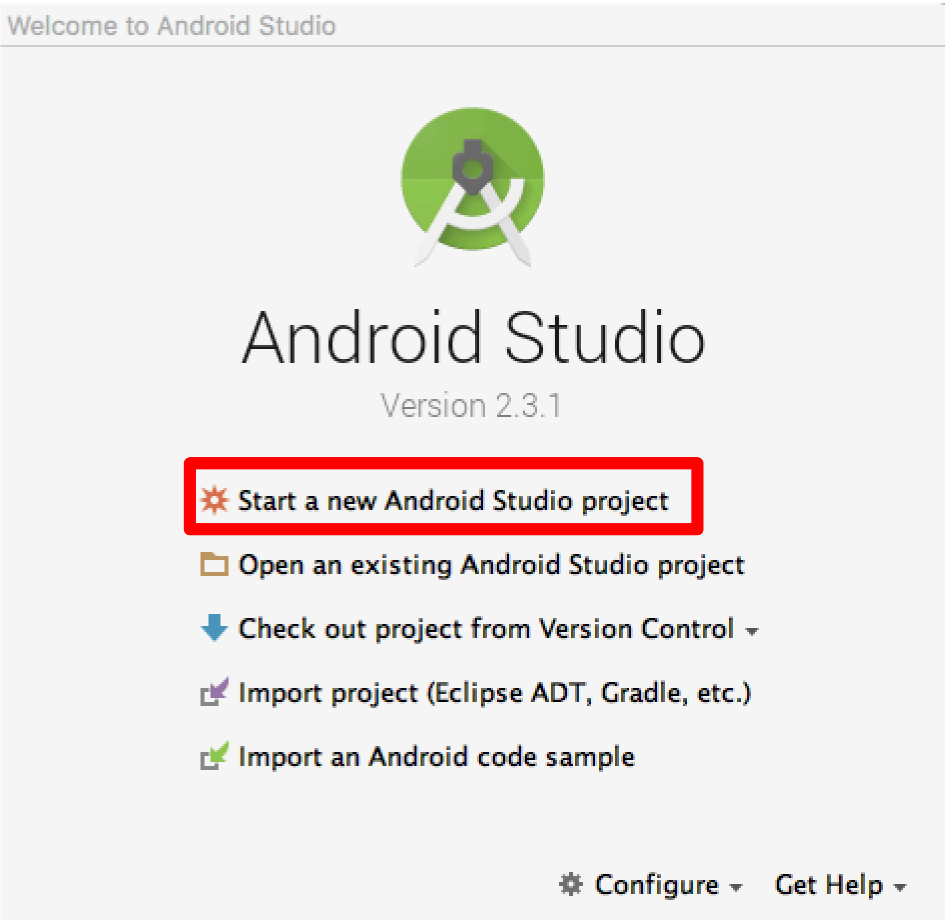
2. Android Studio

2. Android Studio

Android Studio is the official IDE for Google's Android, designed specifically for Android development.

It is available for download on ***Windows, macOS and Linux*** *based operating systems.* *https://developer.android.com/studio*

First Example Demonstration

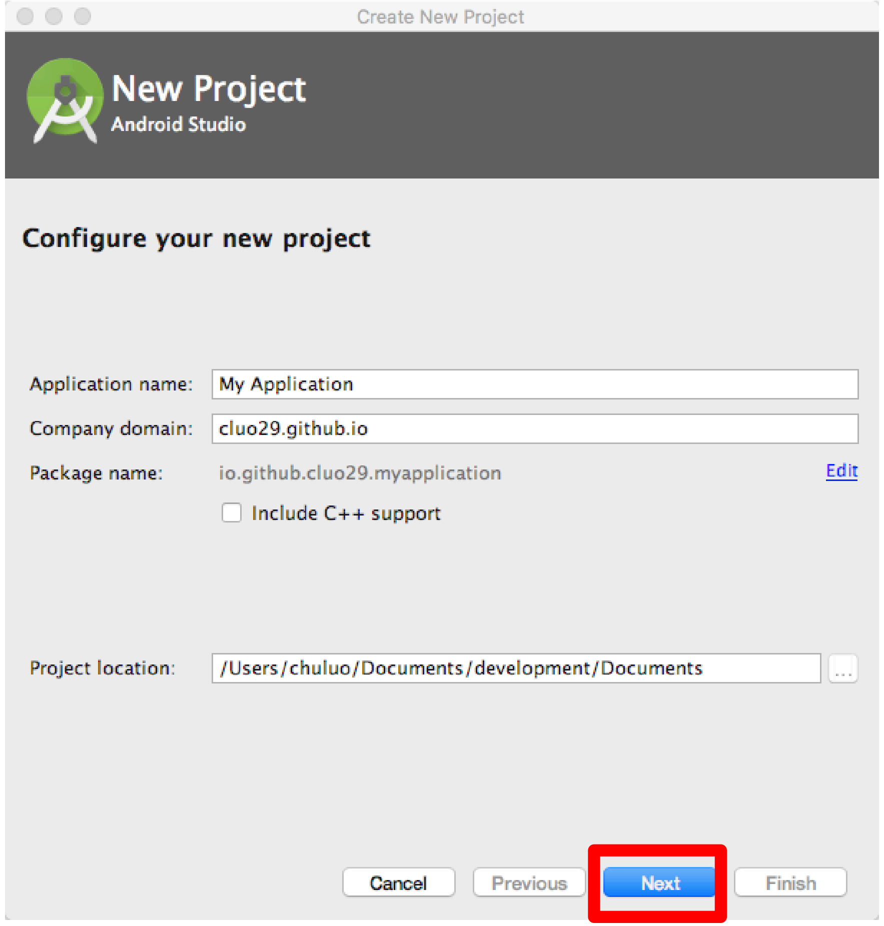


Welcome to Android Studio

Android Studio

Version 2.3.1

Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Import project (Eclipse ADT, Gradle, etc.)
Import an Android code sample
Configure
Get Help



Create New Project

New Project

Android Studio

Configure your new project

Application name: My Application

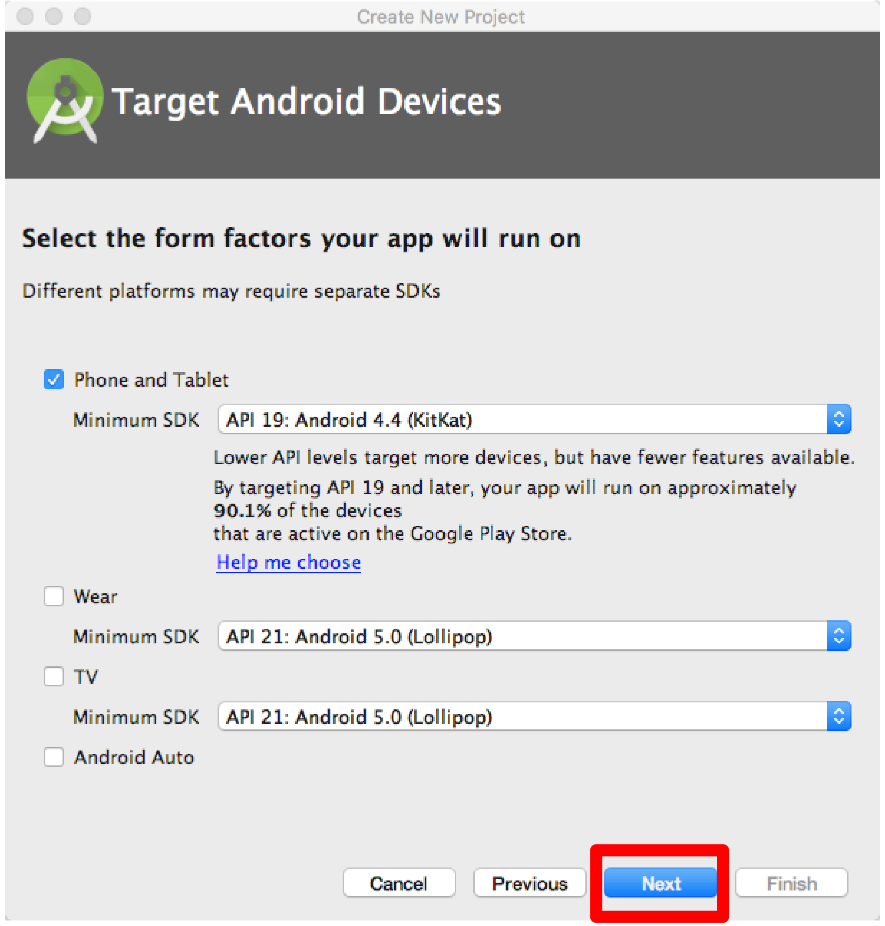
Company domain: cluo29.github.io

Package name: io.github.cluo29.myapplication

☐ Include C++ support

Project location: /Users/chuluo/Documents/development/Documents

First Example Demonstration



Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 19: Android 4.4 (KitKat)

Lower API levels target more devices, but have fewer features available.

By targeting API 19 and later, your app will run on approximately **90.1%** of the devices that are active on the Google Play Store.

Help me choose

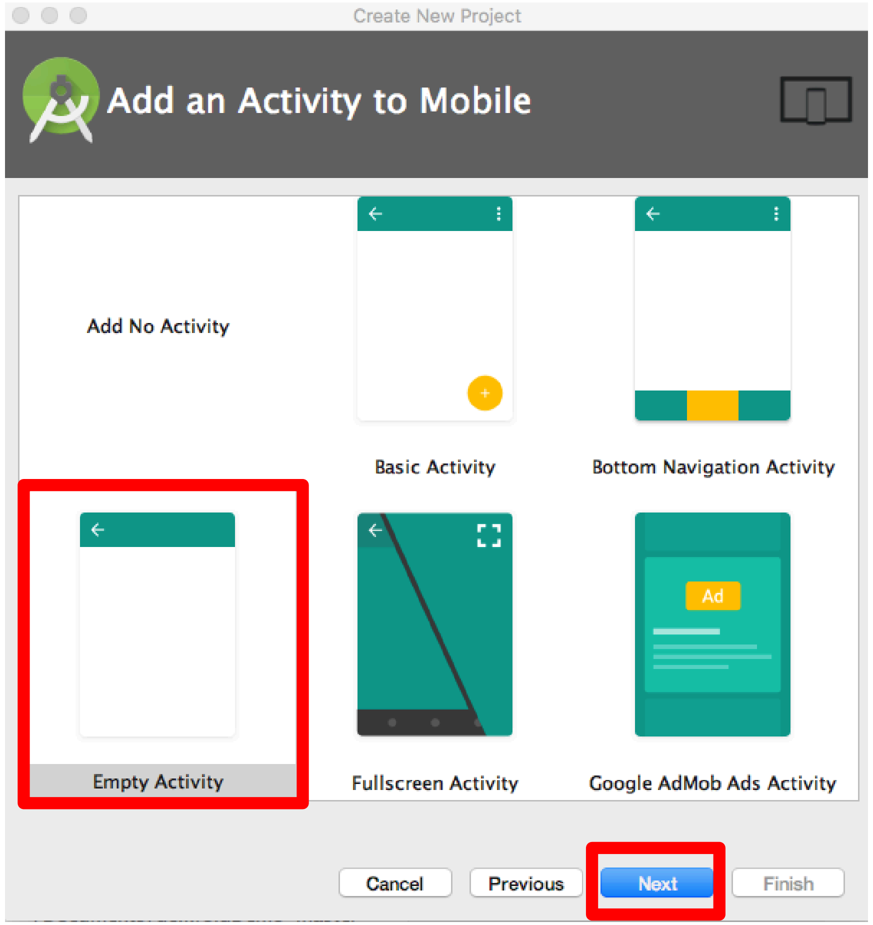
□ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

□ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

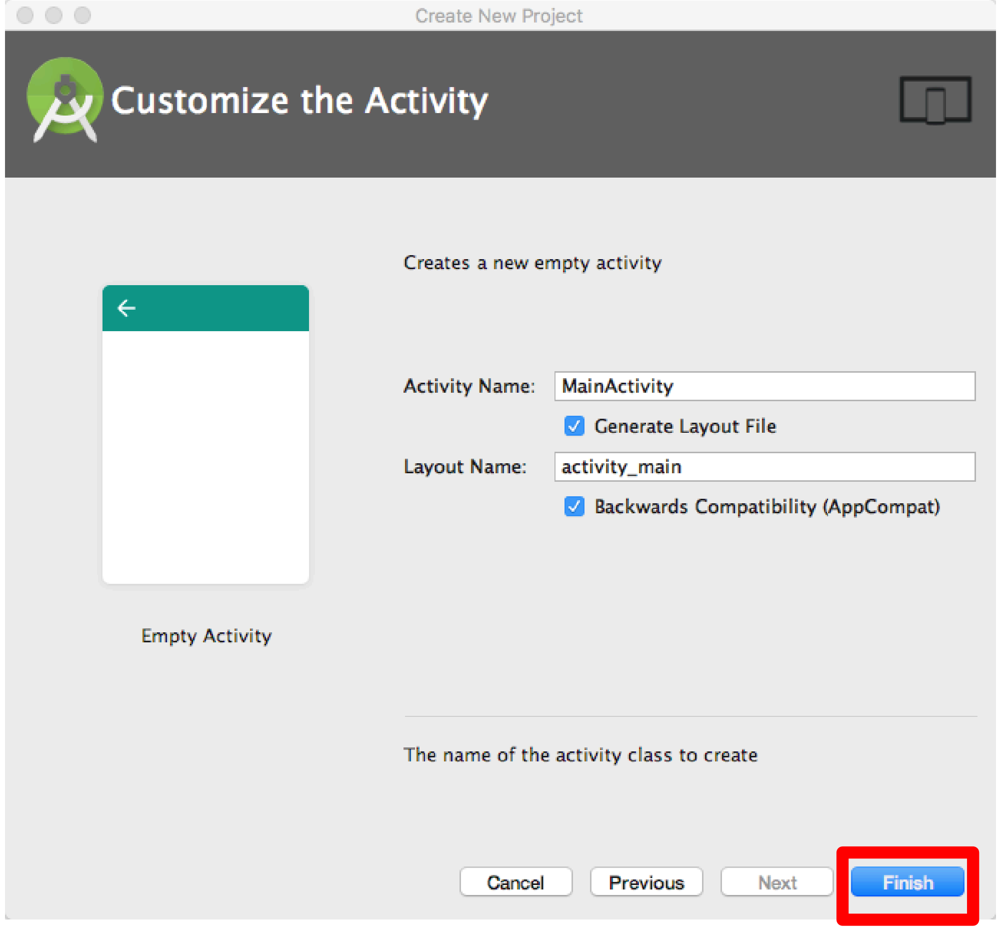
□ Android Auto



Create New Project

Add an Activity to Mobile

First Example Demonstration



Create New Project

Customize the Activity

Creates a new empty activity

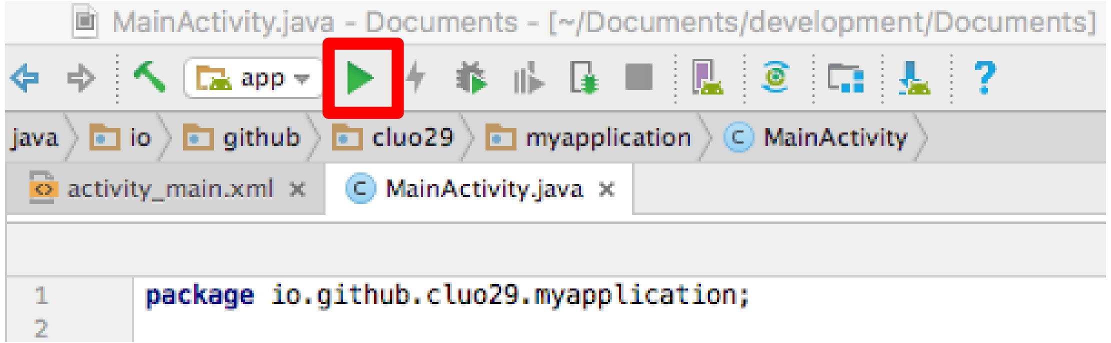
Activity Name: MainActivity

☑ Generate Layout File

Layout Name: activity_main

☑ Backwards Compatibility (AppCompat)

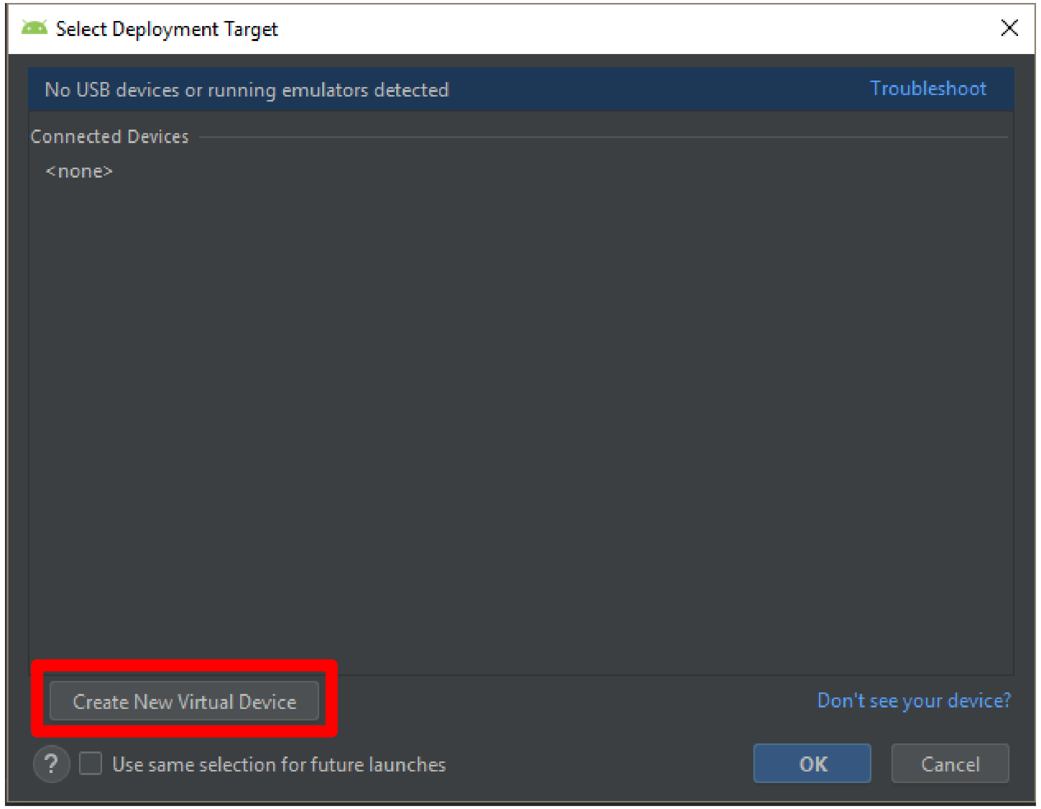
The name of the activity class to create



MainActivity.java - Documents - [~/Documents/development/Documents]

```
package io.github.cluo29.myapplication;
```

First Example Demonstration – Virtual Machine (VM)



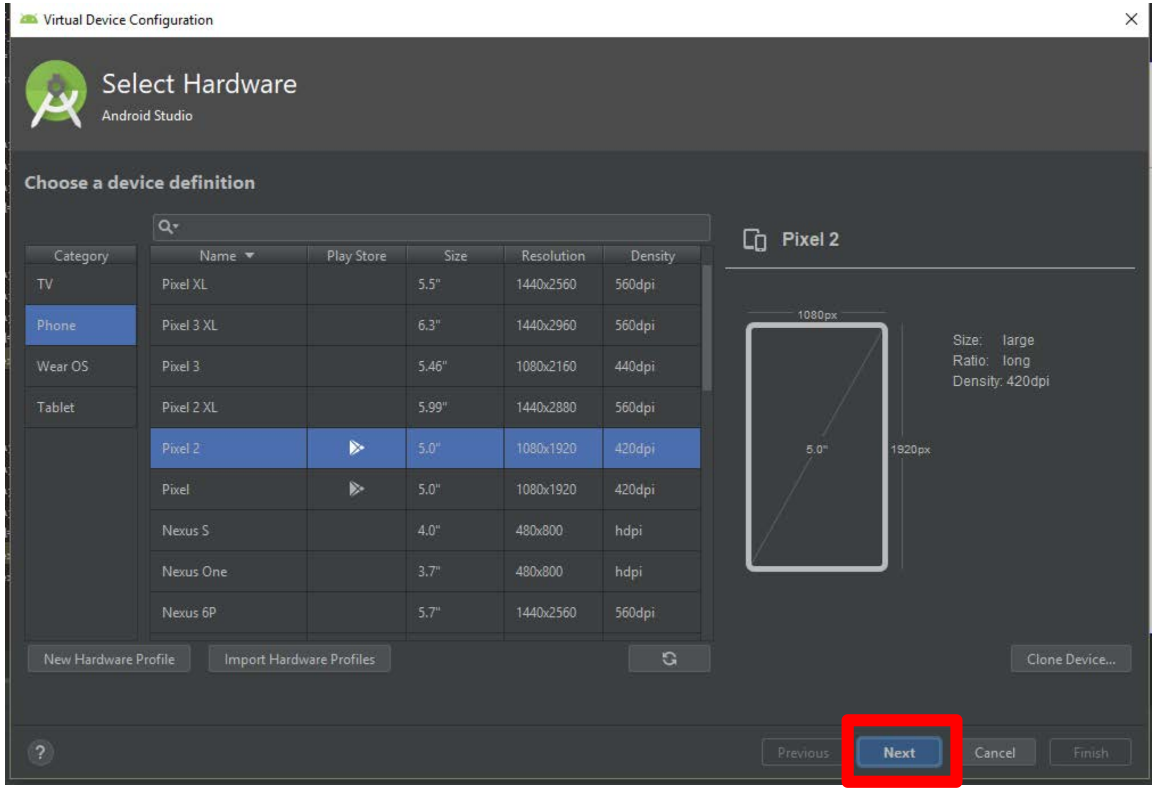
Select Deployment Target

No USB devices or running emulators detected

Connected Devices

<none>

☐ Use same selection for future launches



Virtual Device Configuration

Select Hardware

Android Studio

Choose a device definition

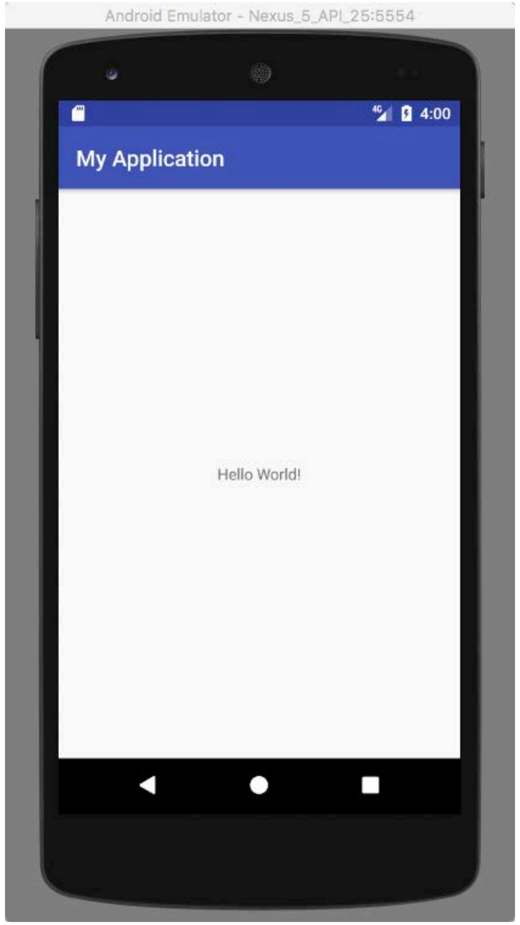
Category
TV
Phone
Wear OS
Tablet

Name	Play Store	Size	Resolution	Density
Pixel XL		5.5"	1440x2560	560dpi
Pixel 3 XL		6.3"	1440x2960	560dpi
Pixel 3		5.46"	1080x2160	440dpi
Pixel 2 XL		5.99"	1440x2880	560dpi
Pixel 2	▶	5.0"	1080x1920	420dpi
Pixel	▶	5.0"	1080x1920	420dpi
Nexus S		4.0"	480x800	hdpi
Nexus One		3.7"	480x800	hdpi
Nexus 6P		5.7"	1440x2560	560dpi

Pixel 2

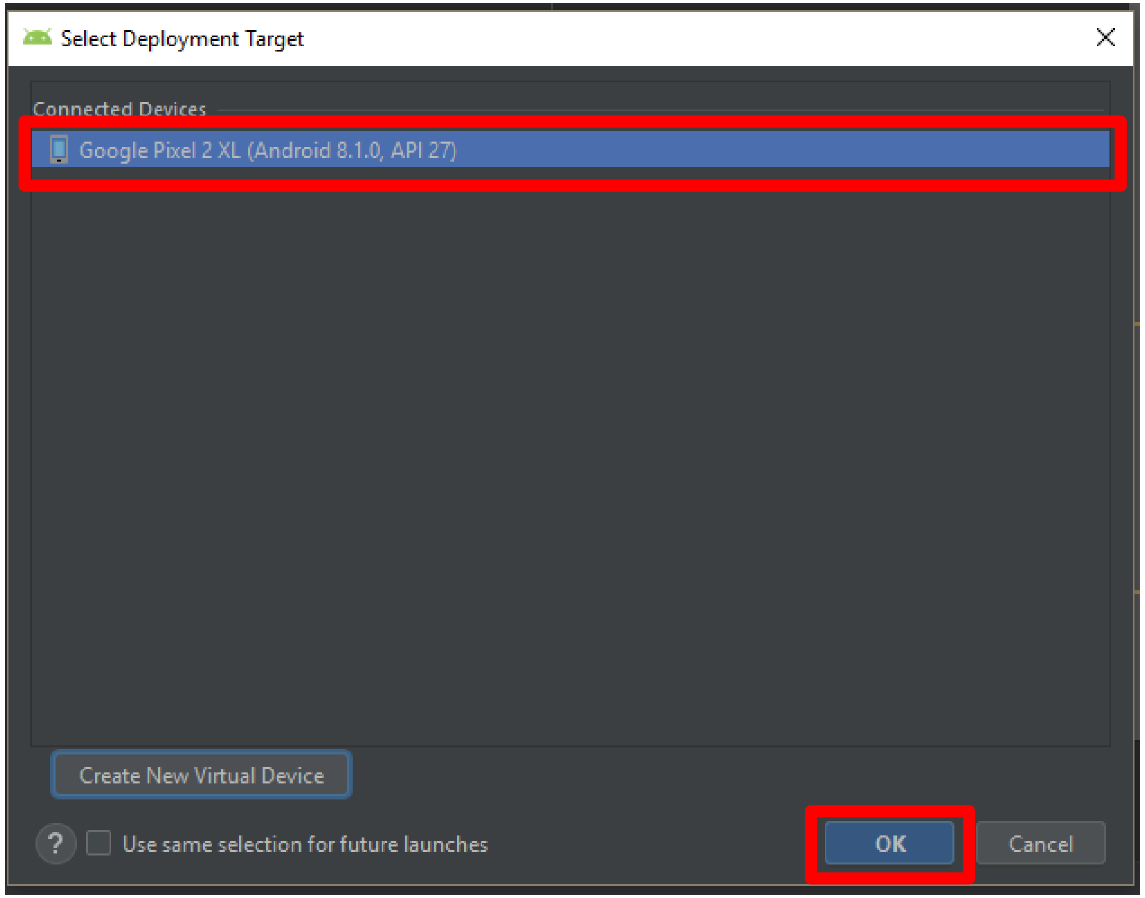
Size: large
Ratio: long
Density: 420dpi

First Example Demonstration



First Application for Android Studio.

First Example Demonstration – Android Phone

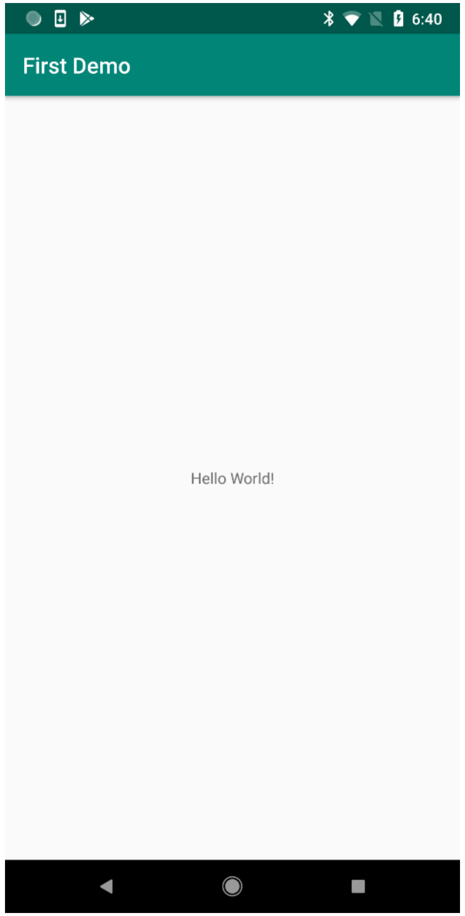


Select Deployment Target

Connected Devices

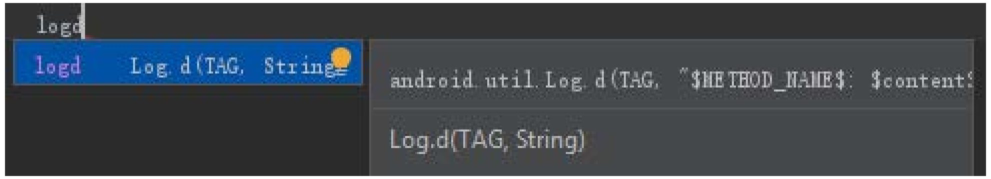
Google Pixel 2 XL (Android 8.1.0, API 27)

Create New Virtual Device
?
□ Use same selection for future launches
OK
Cancel



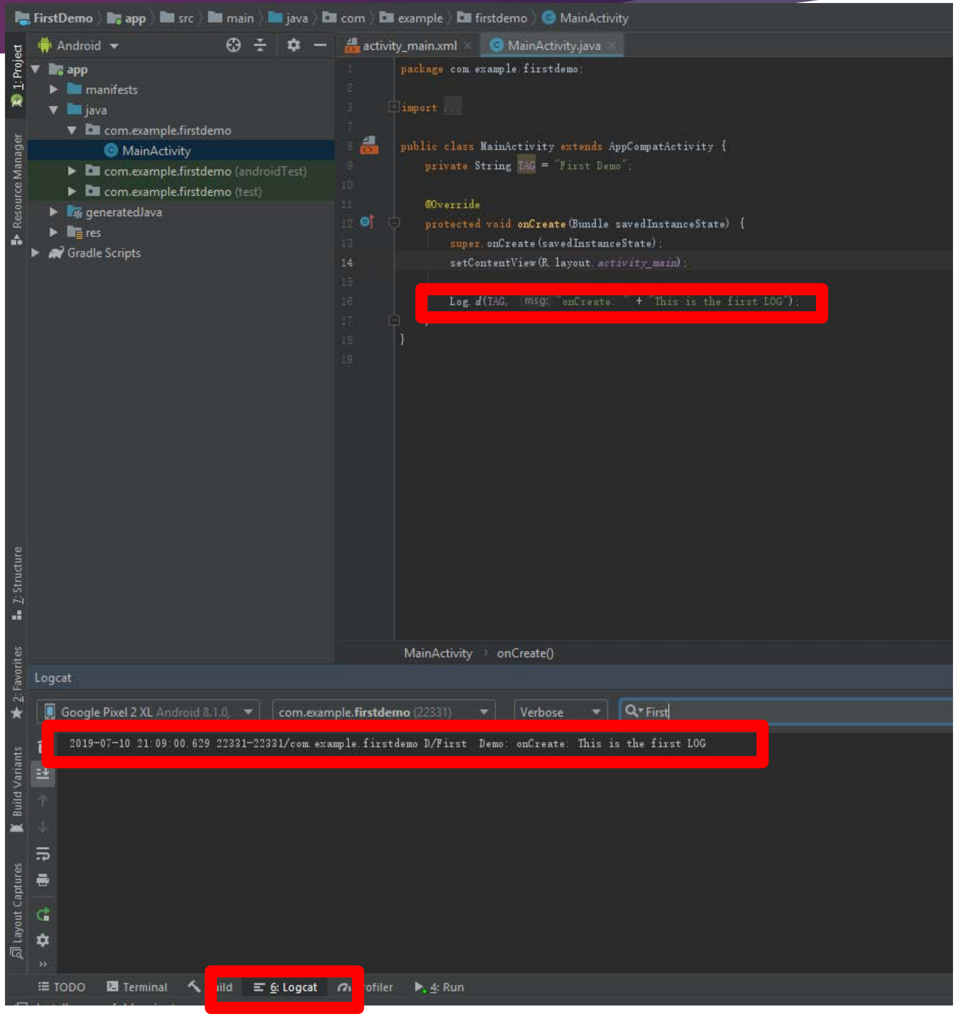
Hello World!

Log in Android Studio



```
logd
logd      Log.d(TAG, String
android.util.Log.d(TAG, "$METHOD_NAME$: $content$
Log.d(TAG, String)
```

Activity(Java)



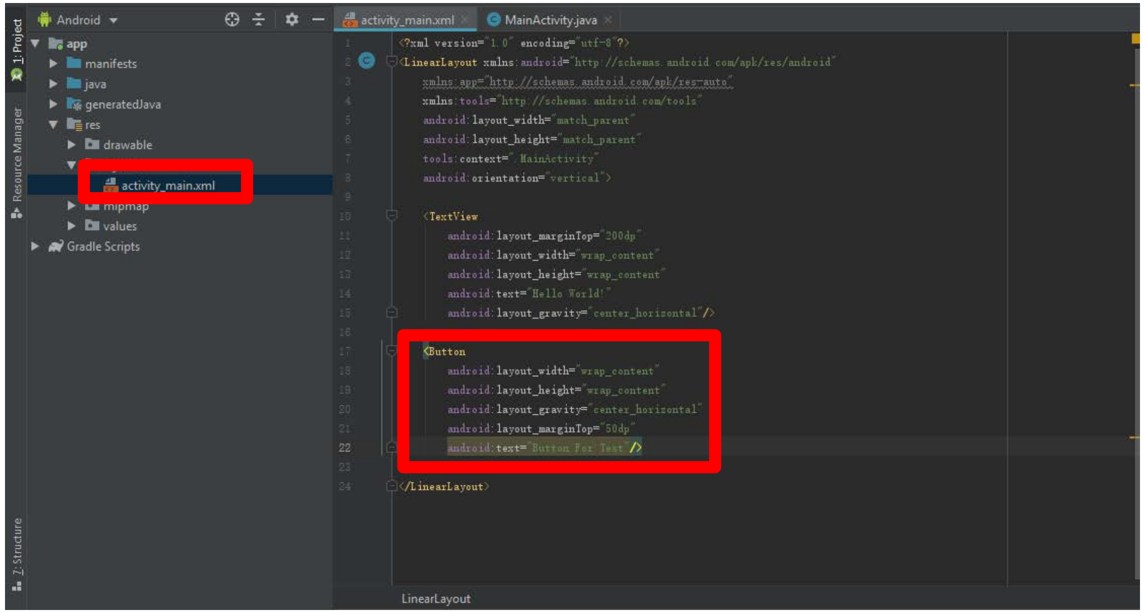
```
package com.example.firstdemo;

import ...

public class MainActivity extends AppCompatActivity {
    private String TAG = "First Demo";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Log.d(TAG, msg: "onCreate: " + "This is the first LOG");
    }
}
```

```
2019-07-10 21:09:00.629 22331-22331/com.example.firstdemo D/First Demo: onCreate: This is the first LOG
```

Android Studio

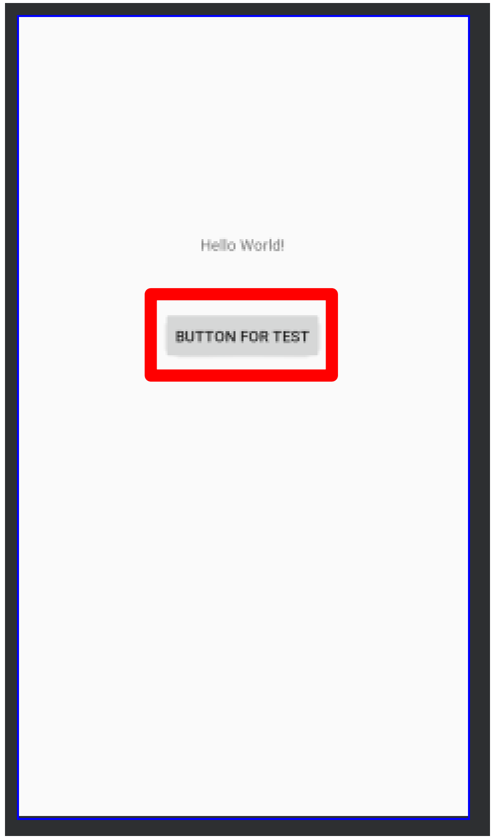
- Add Button and Bind Onclick Listener

Add a button at the First Activity

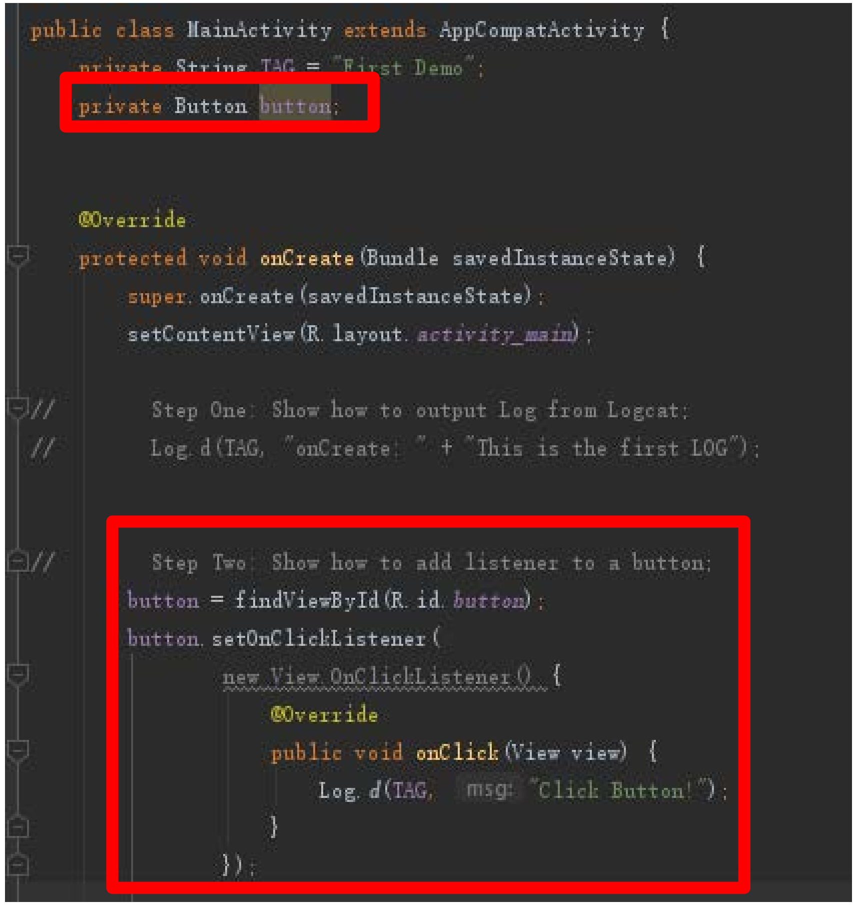


```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:orientation="vertical">

    <TextView
        android:layout_marginTop="200dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        android:layout_gravity="center_horizontal"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="50dp"
        android:text="Button For Test"/>

</LinearLayout>
```

Activity(xml)

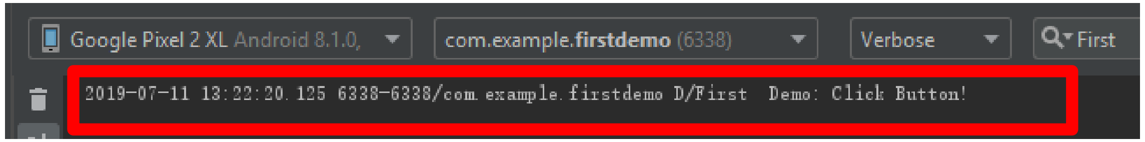


Binding Event to the Button – Native Method



```java
public class MainActivity extends AppCompatActivity {
    private String TAG = "First Demo";
    private Button button;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

//          Step One: Show how to output Log from Logcat;
//          Log.d(TAG, "onCreate: " + "This is the first LOG");

//          Step Two: Show how to add listener to a button;
        button = findViewById(R.id.button);
        button.setOnClickListener(
                new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        Log.d(TAG, msg: "Click Button!");
                    }
                });
```

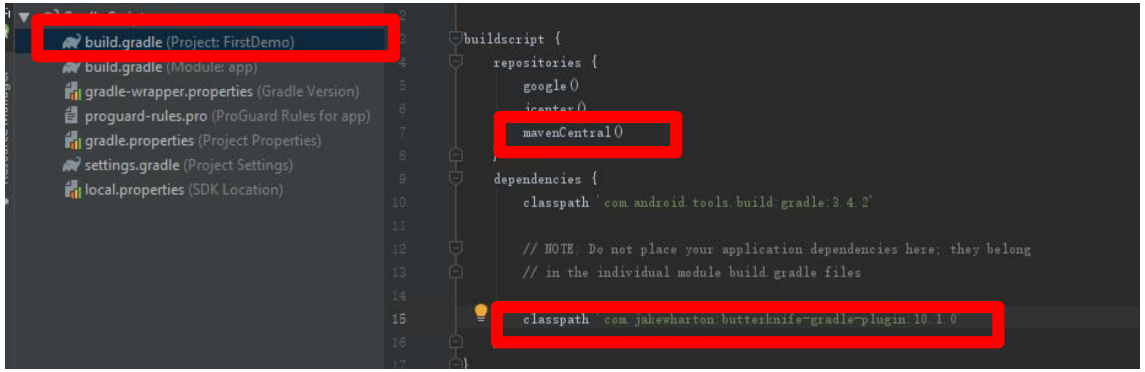
Activity (Java)



Google Pixel 2 XL Android 8.1.0, | com.example.firstdemo (6338) | Verbose | First

```
2019-07-11 13:22:20.125 6338-6338/com.example.firstdemo D/First  Demo: Click Button!
```

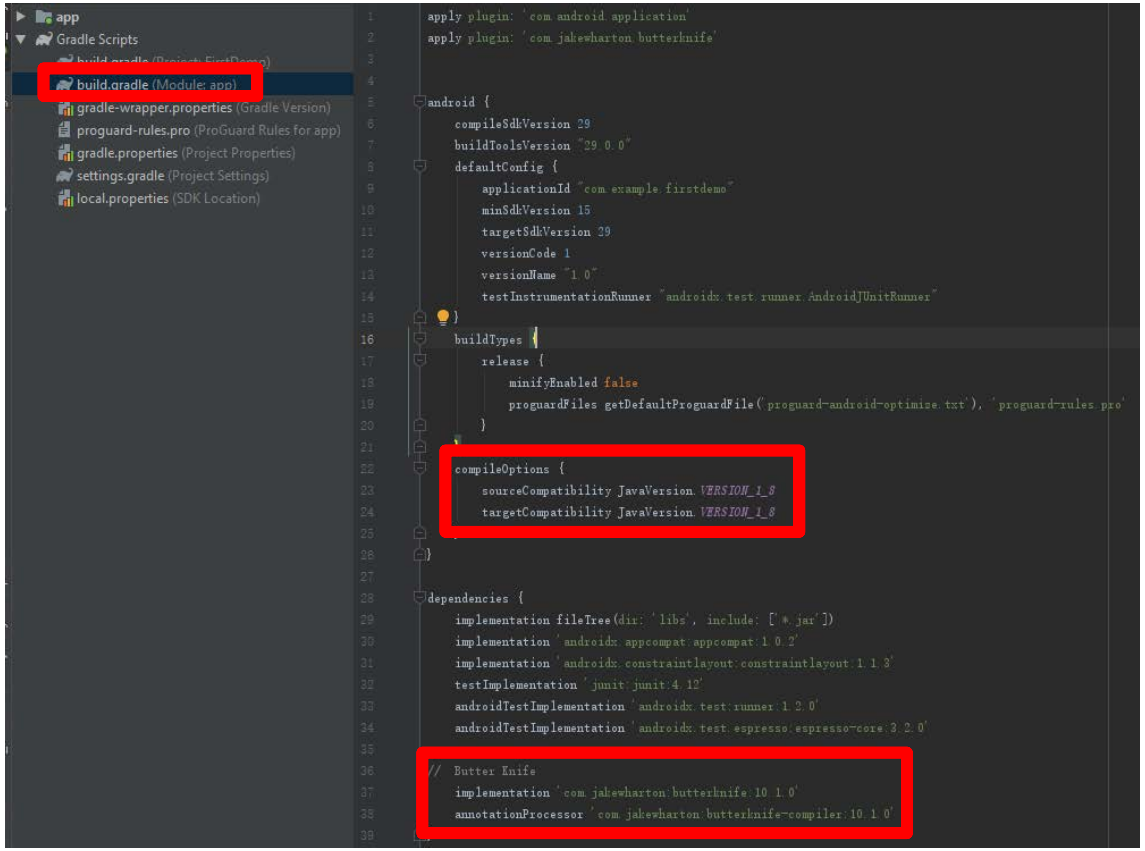
Binding Event to the Button – Butter Knife


build.gradle (Project: FirstDemo)
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)

```
buildscript {
    repositories {
        google()
        jcenter()
        mavenCentral()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.4.2'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files

        classpath 'com.jakewharton:butterknife-gradle-plugin:10.1.0'
    }
}
```

build.gradle (project level)

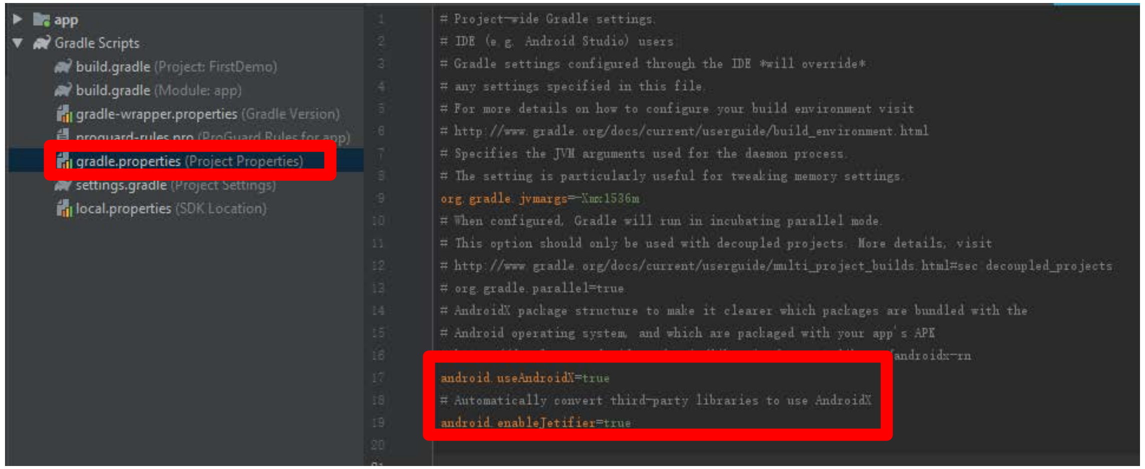
https://github.com/JakeWharton/butterknife
https://jakewharton.github.io/butterknife/


app
Gradle Scripts
build.gradle (Project: FirstDemo)
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)

```
apply plugin: 'com.android.application'
apply plugin: 'com.jakewharton.butterknife'


android {
    compileSdkVersion 29
    buildToolsVersion "29.0.0"
    defaultConfig {
        applicationId "com.example.firstdemo"
        minSdkVersion 15
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'androidx.appcompat:appcompat:1.0.2'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test:runner:1.2.0'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'

//  Butter Knife
    implementation 'com.jakewharton:butterknife:10.1.0'
    annotationProcessor 'com.jakewharton:butterknife-compiler:10.1.0'
```

build.gradle (app level)

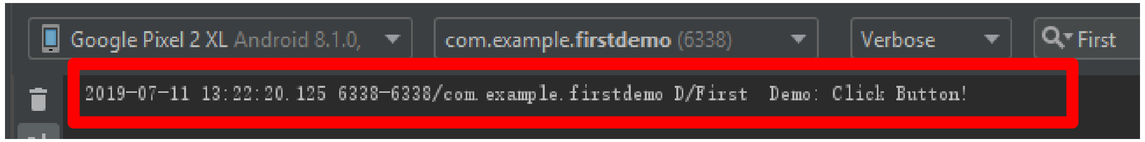
Binding Event to the Button – Butter Knife


app
Gradle Scripts
build.gradle (Project: FirstDemo)
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)

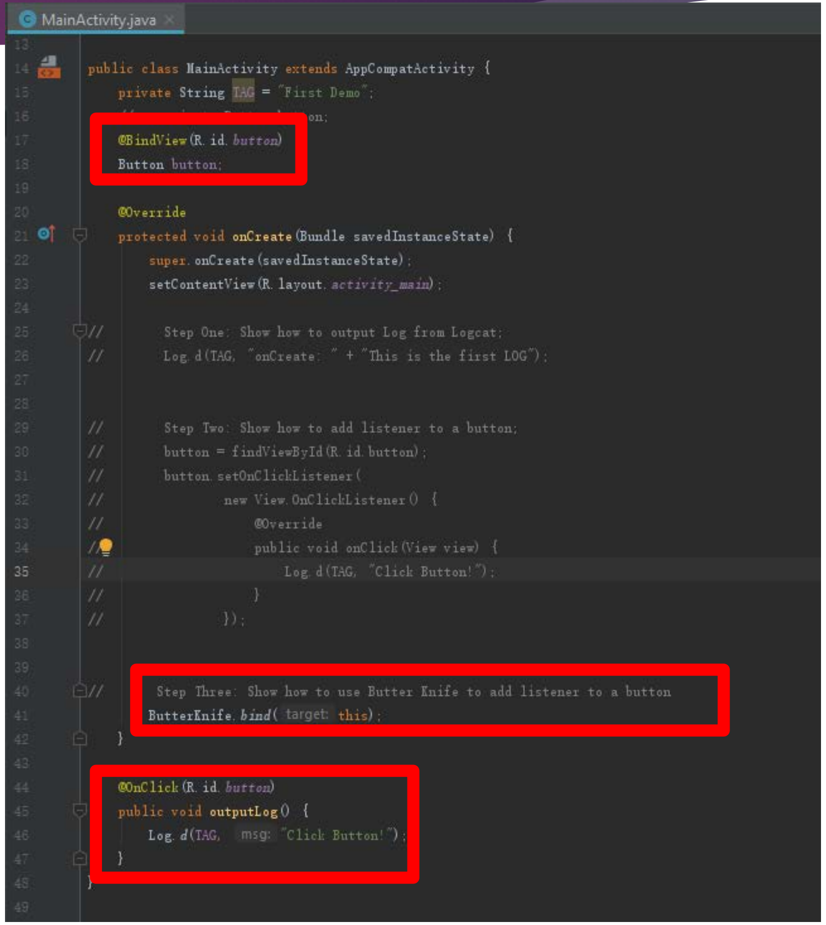
```
# Project-wide Gradle settings.
# IDE (e.g. Android Studio) users:
# Gradle settings configured through the IDE *will override*
# any settings specified in this file.
# For more details on how to configure your build environment visit
# http://www.gradle.org/docs/current/userguide/build_environment.html
# Specifies the JVM arguments used for the daemon process.
# The setting is particularly useful for tweaking memory settings.
org.gradle.jvmargs=-Xmx1536m
# When configured, Gradle will run in incubating parallel mode.
# This option should only be used with decoupled projects. More details, visit
# http://www.gradle.org/docs/current/userguide/multi_project_builds.html#sec:decoupled_projects
# org.gradle.parallel=true
# AndroidX package structure to make it clearer which packages are bundled with the
# Android operating system, and which are packaged with your app's APK
[illegible]androidx-rn
android.useAndroidX=true
# Automatically convert third-party libraries to use AndroidX
android.enableJetifier=true
```

gradle.properties

Result


Google Pixel 2 XL Android 8.1.0,
com.example.firstdemo (6338)
Verbose
First

```
2019-07-11 13:22:20.125 6338-6338/com.example.firstdemo D/First Demo: Click Button!
```



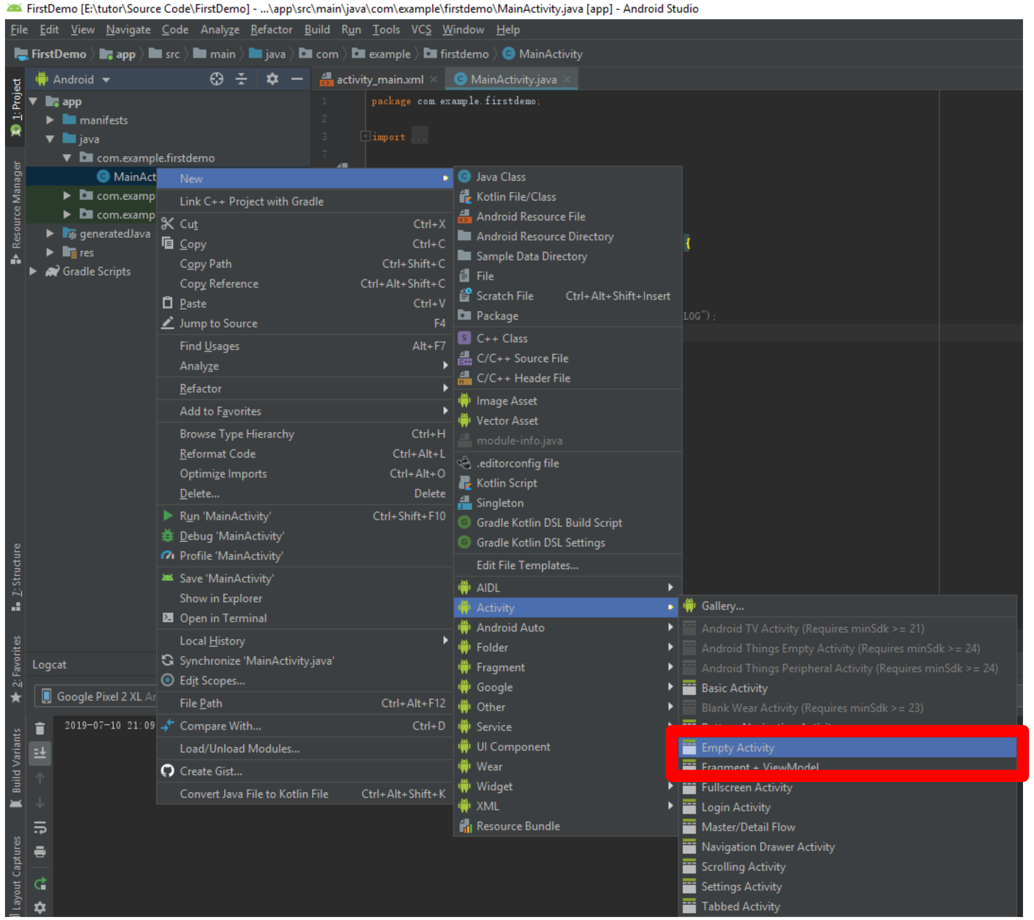
```java

public class MainActivity extends AppCompatActivity {
    private String TAG = "First Demo";
    [illegible]on;
    @BindView(R.id.button)
    Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

//         Step One: Show how to output Log from Logcat;
//         Log.d(TAG, "onCreate: " + "This is the first LOG");


//         Step Two: Show how to add listener to a button;
//         button = findViewById(R.id.button);
//         button.setOnClickListener(
//                 new View.OnClickListener() {
//                     @Override
//                     public void onClick(View view) {
//                         Log.d(TAG, "Click Button!");
//                     }
//                 });


//        Step Three: Show how to use Butter Knife to add listener to a button
        ButterKnife.bind( target: this);
    }

    @OnClick(R.id.button)
    public void outputLog() {
        Log.d(TAG, msg: "Click Button!");
    }
}

```

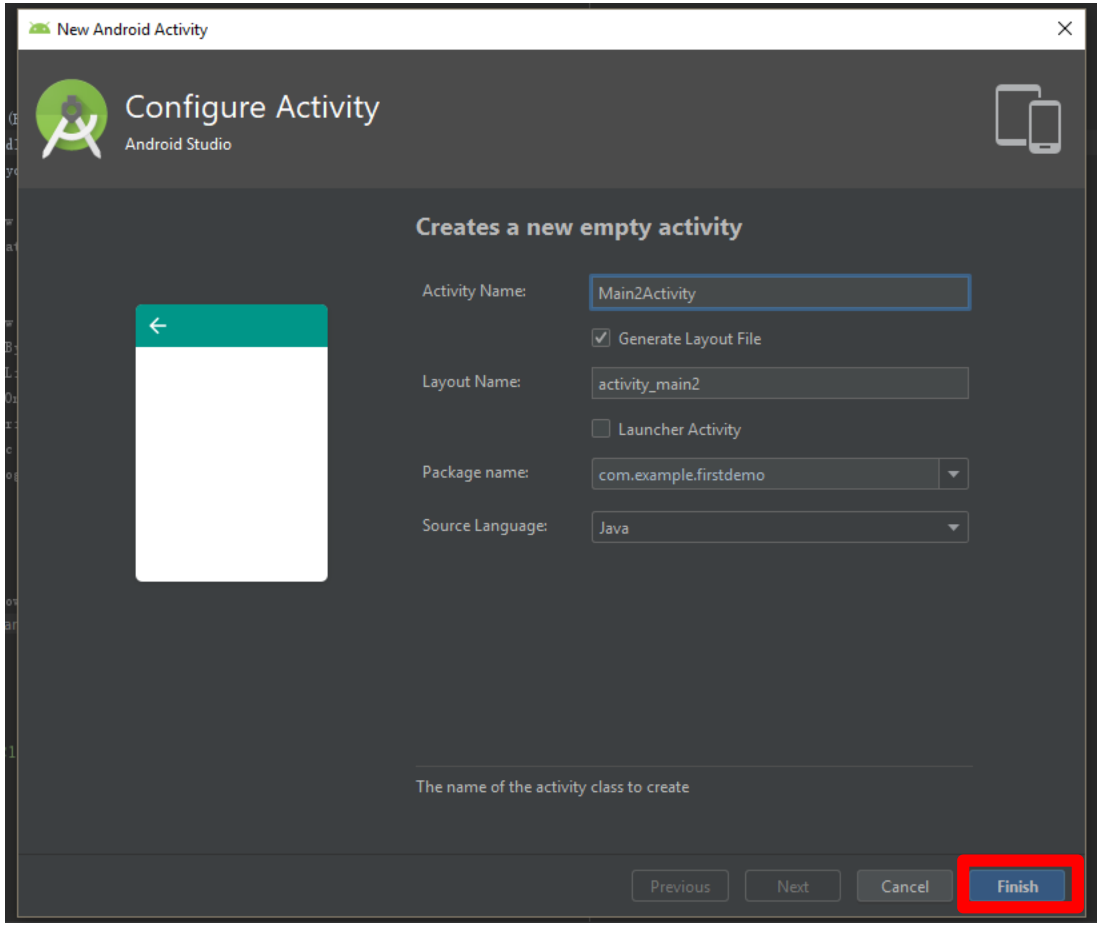
Activity (Java)

Android Studio

- Create Second Activity

Create the Second Activity





Creates a new empty activity

Activity Name: Main2Activity

☑ Generate Layout File

Layout Name: activity_main2

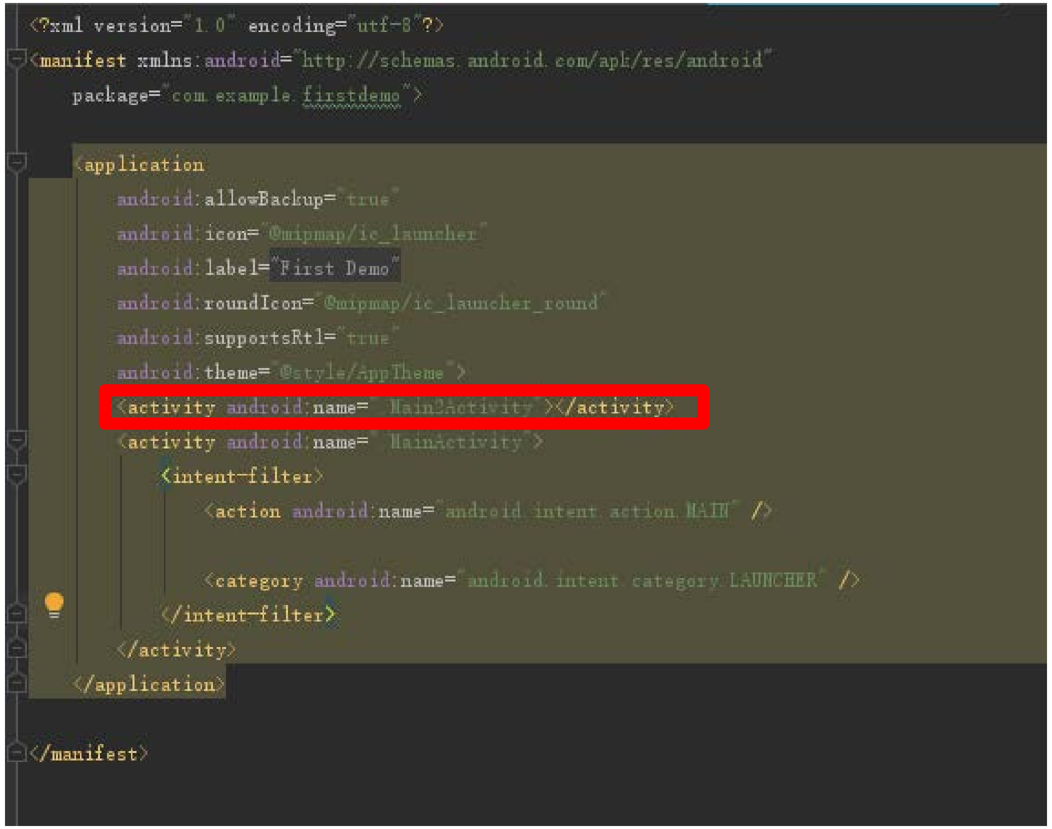
□ Launcher Activity

Package name: com.example.firstdemo

Source Language: Java

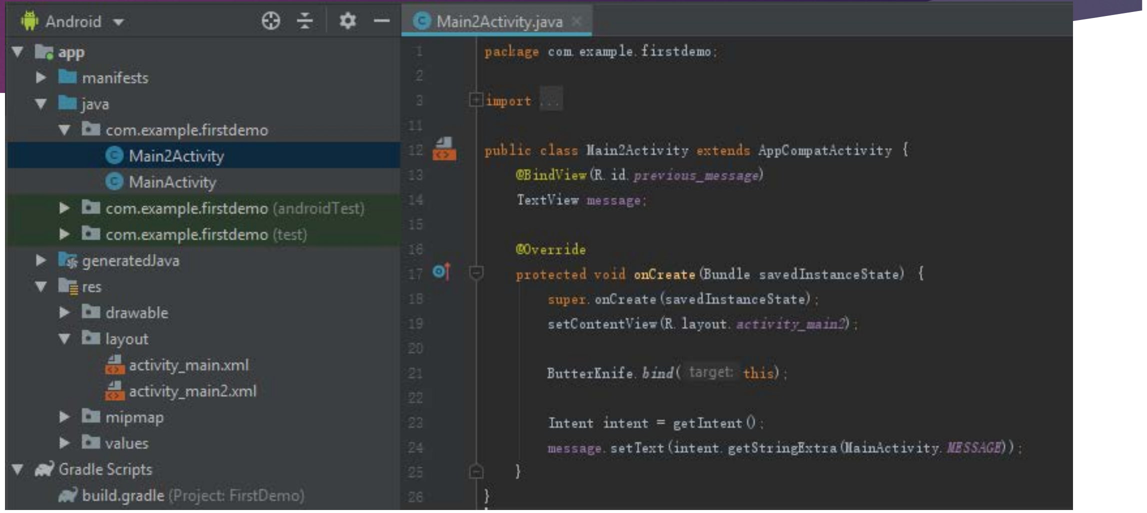
The name of the activity class to create

Create the Second Activity – Generated Scripts



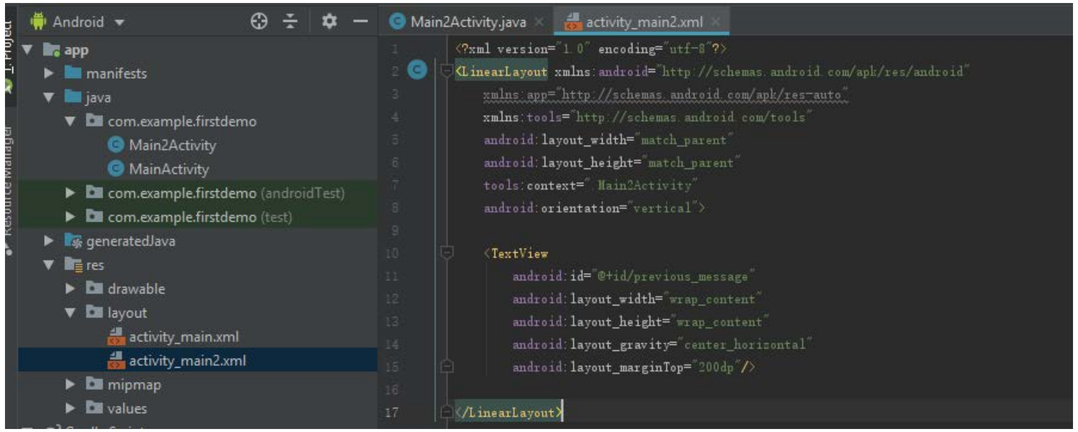
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.firstdemo">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="First Demo"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".Main2Activity"></activity>
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

AndroidManifest.xml



```java
package com.example.firstdemo;

import ...

public class Main2Activity extends AppCompatActivity {
    @BindView(R.id.previous_message)
    TextView message;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        ButterKnife.bind( target: this);

        Intent intent = getIntent();
        message.setText(intent.getStringExtra(MainActivity.MESSAGE));
    }
}
```

Activity 2 (Java)



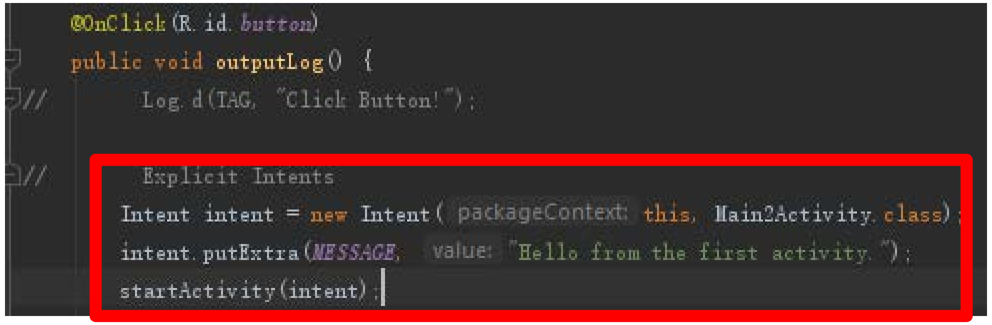
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Main2Activity"
    android:orientation="vertical">

    <TextView
        android:id="@+id/previous_message"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="200dp"/>

</LinearLayout>
```

Activity 2 (xml)

Android Studio

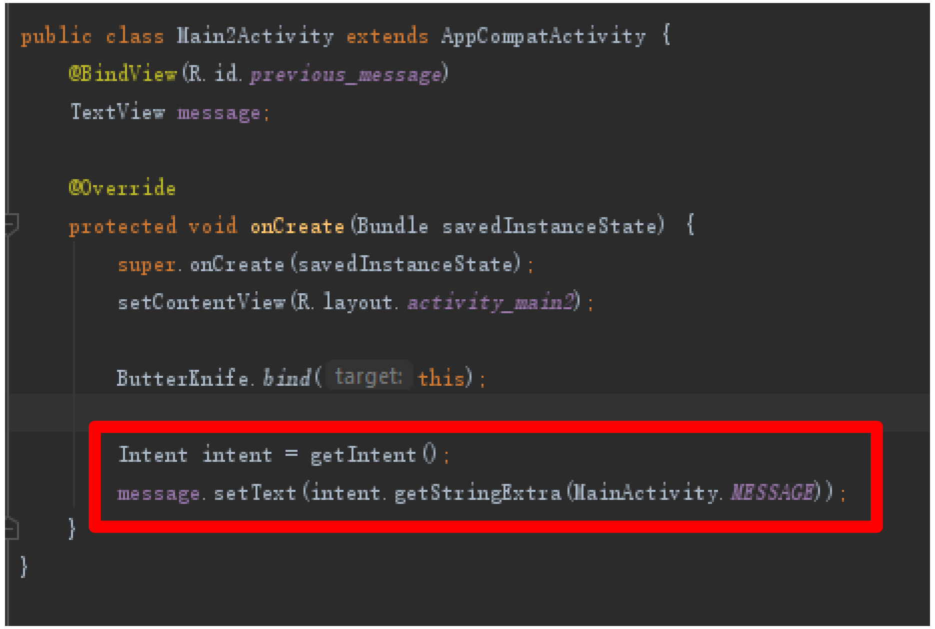
- Communicate between First Activity and Second Activity

Explicit Intents



```java
    @OnClick(R.id.button)
    public void outputLog() {
//      Log.d(TAG, "Click Button!");

//      Explicit Intents
        Intent intent = new Intent( packageContext: this, Main2Activity.class);
        intent.putExtra(MESSAGE,  value: "Hello from the first activity.");
        startActivity(intent);
```

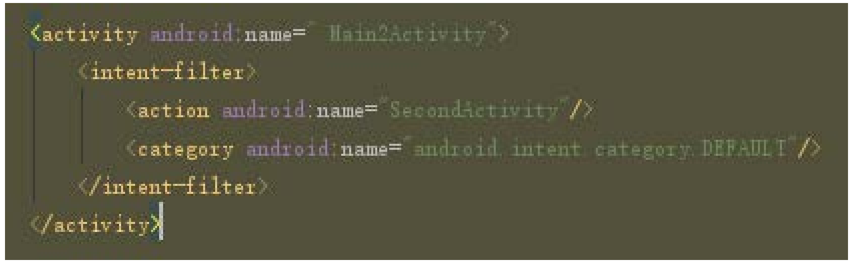
Activity 1 (Java)



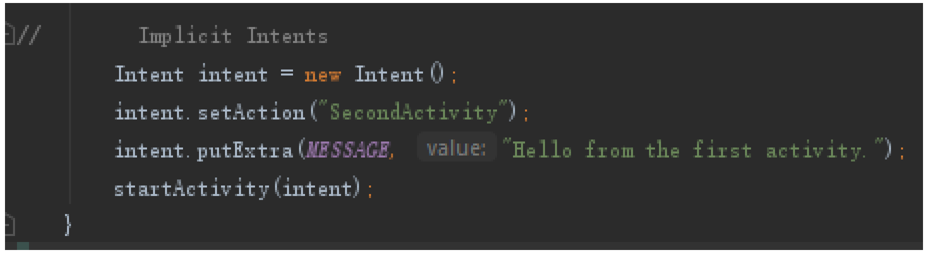
```java
public class Main2Activity extends AppCompatActivity {
    @BindView(R.id.previous_message)
    TextView message;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main2);

        ButterKnife.bind( target: this);

        Intent intent = getIntent();
        message.setText(intent.getStringExtra(MainActivity.MESSAGE));
    }
}
```

Activity 2 (Java)

Implicit Intents



```
<activity android:name=".Main2Activity">
    <intent-filter>
        <action android:name="SecondActivity"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
</activity>
```

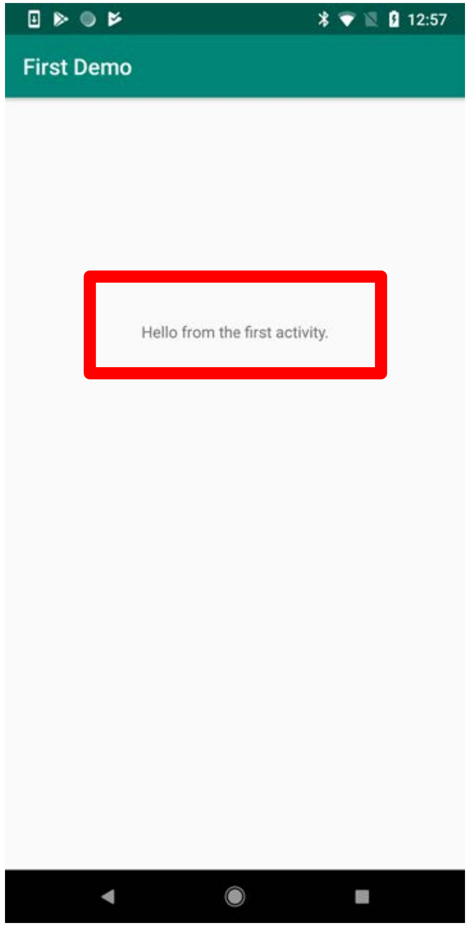


```
//      Implicit Intents
        Intent intent = new Intent();
        intent.setAction("SecondActivity");
        intent.putExtra(MESSAGE, value: "Hello from the first activity.");
        startActivity(intent);
    }
```

AnroidManifest.xml

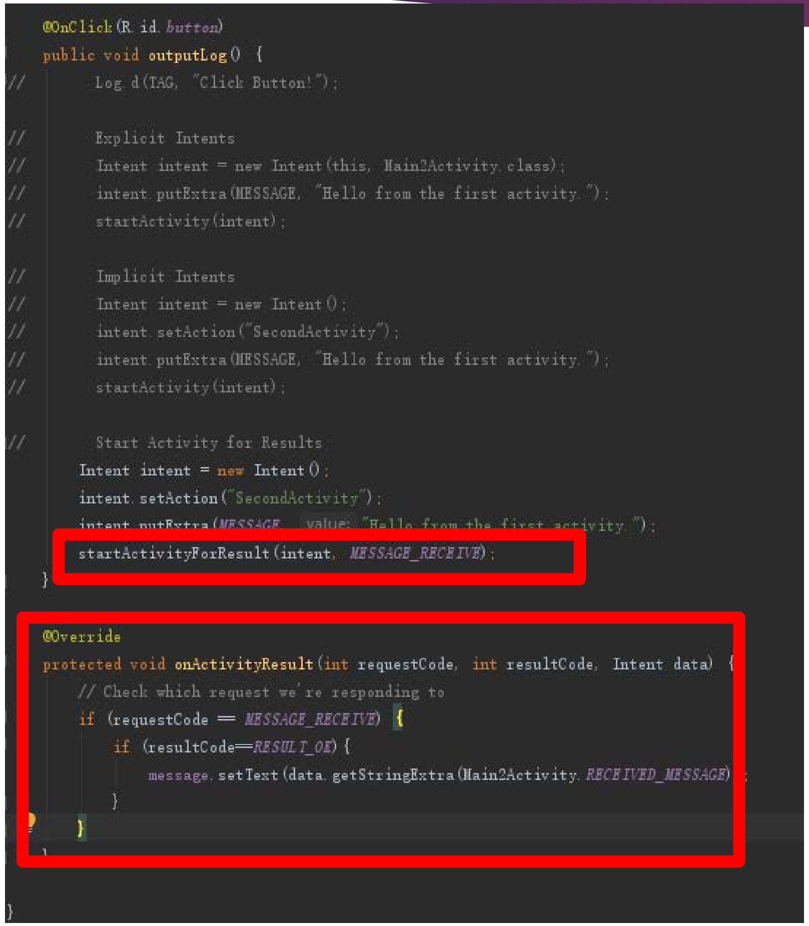
Activity 1 (Java)

Message from First Activity



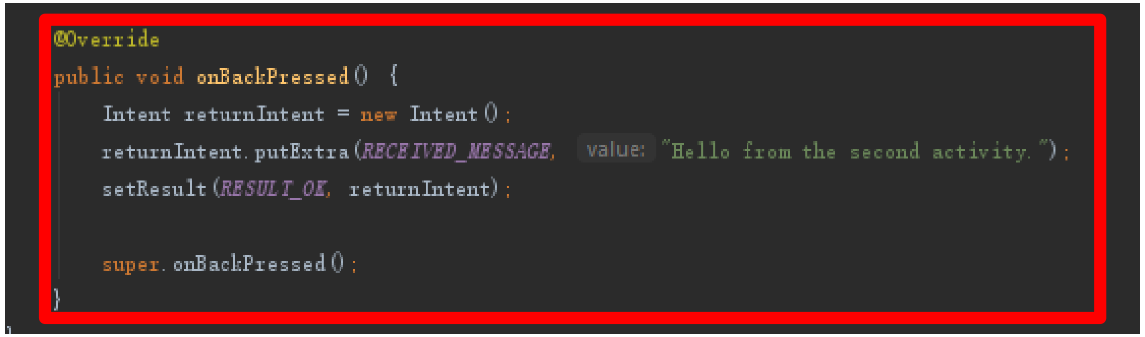
Hello from the first activity.

Collect Results from Called Activities



```
    @OnClick(R.id.button)
    public void outputLog() {
//      Log.d(TAG, "Click Button!");

//      Explicit Intents
//      Intent intent = new Intent(this, Main2Activity.class);
//      intent.putExtra(MESSAGE, "Hello from the first activity.");
//      startActivity(intent);

//      Implicit Intents
//      Intent intent = new Intent();
//      intent.setAction("SecondActivity");
//      intent.putExtra(MESSAGE, "Hello from the first activity.");
//      startActivity(intent);

//      Start Activity for Results
        Intent intent = new Intent();
        intent.setAction("SecondActivity");
        intent.putExtra(MESSAGE, value: "Hello from the first activity.");
        startActivityForResult(intent, MESSAGE_RECEIVE);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // Check which request we're responding to
        if (requestCode == MESSAGE_RECEIVE) {
            if (resultCode==RESULT_OK) {
                message.setText(data.getStringExtra(Main2Activity.RECEIVED_MESSAGE));
            }
        }
    }

}
```

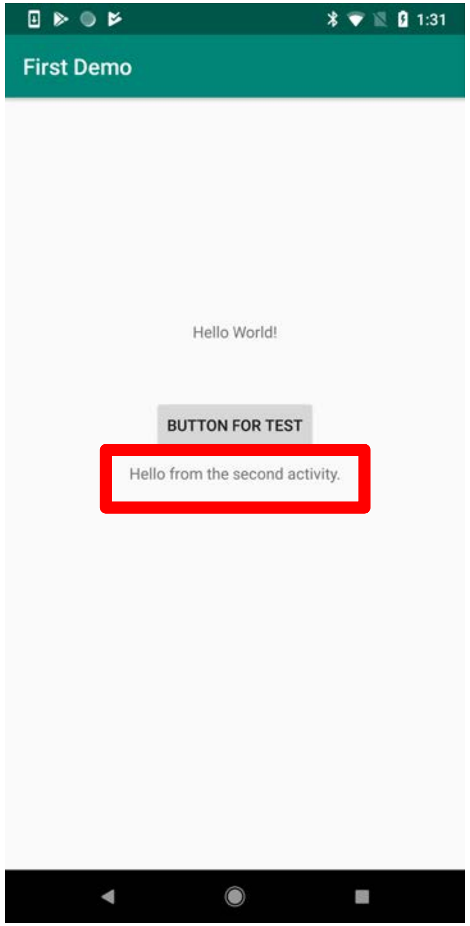
Activity 1 (Java)



```
@Override
public void onBackPressed() {
    Intent returnIntent = new Intent();
    returnIntent.putExtra(RECEIVED_MESSAGE, value: "Hello from the second activity.");
    setResult(RESULT_OK, returnIntent);

    super.onBackPressed();
}
```

Activity 2 (Java)

Message from First Activity



First Demo

Hello World!

BUTTON FOR TEST

Hello from the second activity.

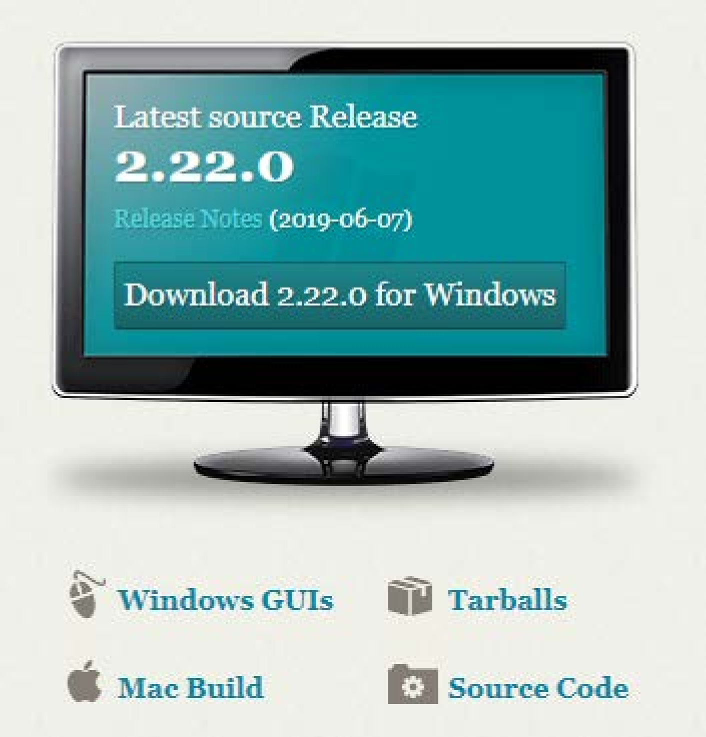
3. Git

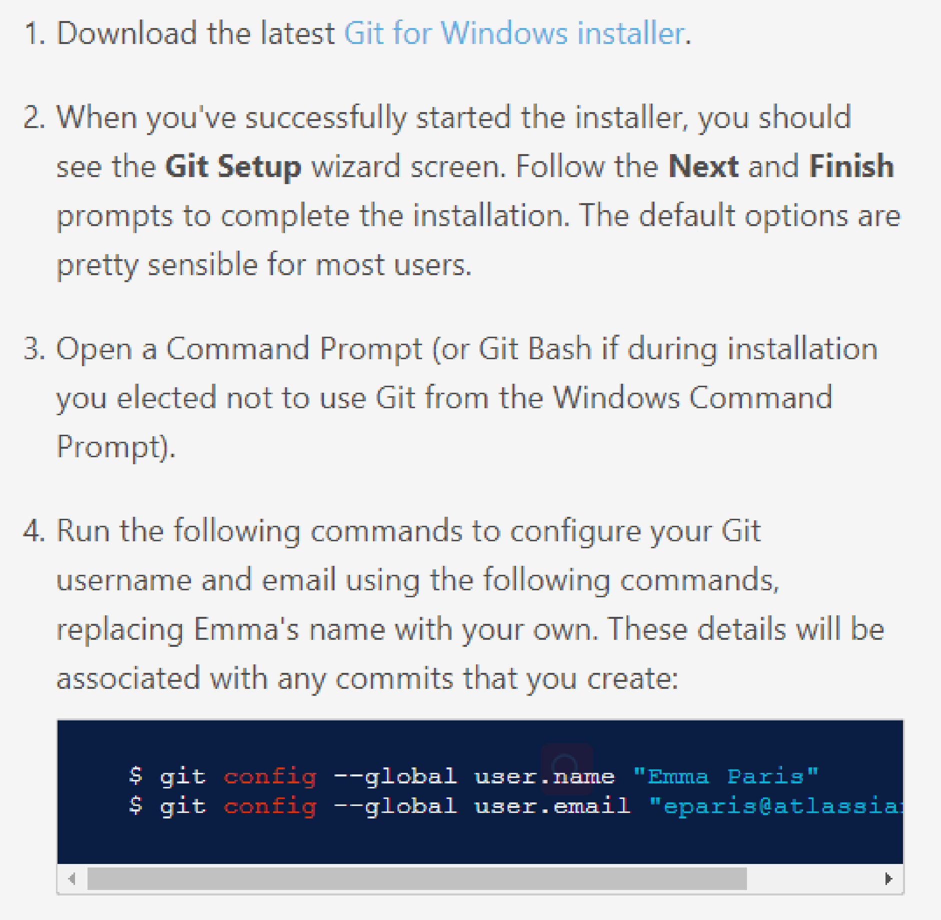
Git

Git is a free and open source distributed ***version control system*** designed to handle everything from small to very large projects with speed and efficiency.

https://git-scm.com/

Git Installation



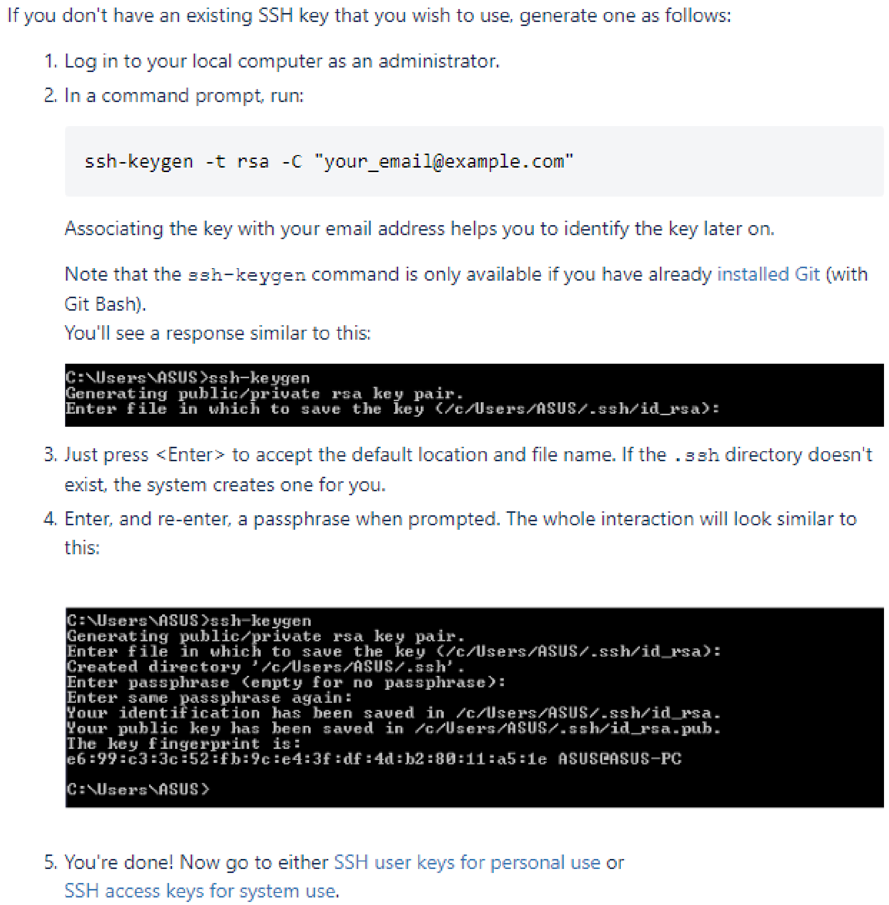


1. Download the latest Git for Windows installer.
2. When you've successfully started the installer, you should see the **Git Setup** wizard screen. Follow the **Next** and **Finish** prompts to complete the installation. The default options are pretty sensible for most users.
3. Open a Command Prompt (or Git Bash if during installation you elected not to use Git from the Windows Command Prompt).
4. Run the following commands to configure your Git username and email using the following commands, replacing Emma's name with your own. These details will be associated with any commits that you create:

```
$ git config --global user.name "Emma Paris"
$ git config --global user.email "eparis@atlassia
```

https://www.atlassian.com/git/tutorials/install-git

SSH Keys



If you don't have an existing SSH key that you wish to use, generate one as follows:

1. Log in to your local computer as an administrator.
2. In a command prompt, run:

```
ssh-keygen -t rsa -C "your_email@example.com"
```

Associating the key with your email address helps you to identify the key later on.

Note that the `ssh-keygen` command is only available if you have already installed Git (with Git Bash).
You'll see a response similar to this:

```
C:\Users\ASUS>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ASUS/.ssh/id_rsa):
```

3. Just press <Enter> to accept the default location and file name. If the `.ssh` directory doesn't exist, the system creates one for you.
4. Enter, and re-enter, a passphrase when prompted. The whole interaction will look similar to this:

```
C:\Users\ASUS>ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ASUS/.ssh/id_rsa):
Created directory '/c/Users/ASUS/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/ASUS/.ssh/id_rsa.
Your public key has been saved in /c/Users/ASUS/.ssh/id_rsa.pub.
The key fingerprint is:
e6:99:c3:3c:52:fb:9c:e4:3f:df:4d:b2:80:11:a5:1e ASUS@ASUS-PC

C:\Users\ASUS>
```

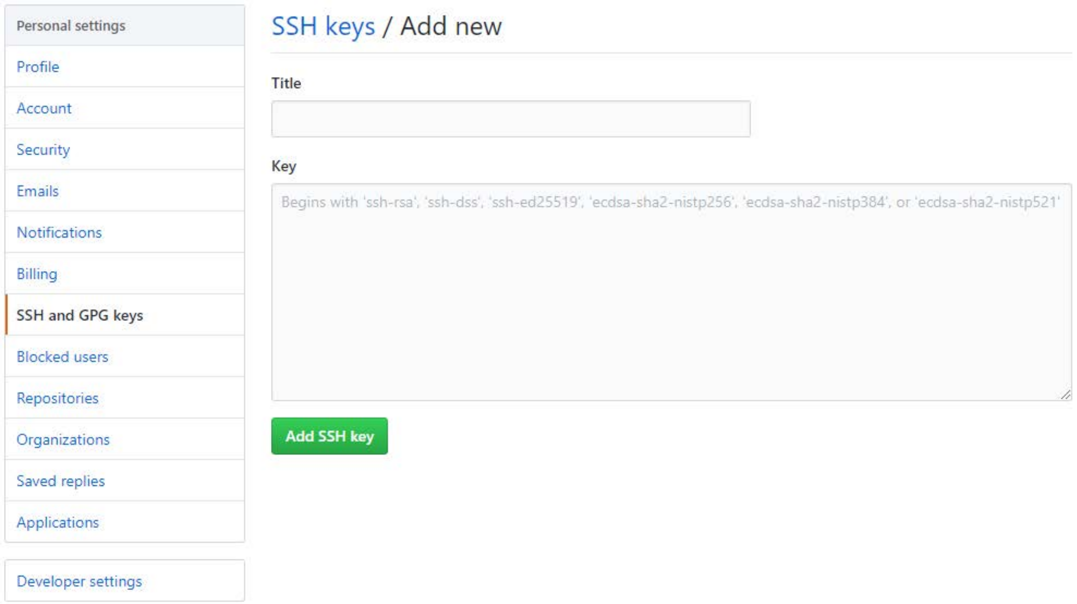
5. You're done! Now go to either SSH user keys for personal use or SSH access keys for system use.

SSH keys can be used to establish a secure connection with Server for:

- when you are performing Git operations from your local machine
- when another system or process needs access to repositories in Server (for example your build server)

https://confluence.atlassian.com/bitbucketserver/creating-ssh-keys-776639788.html

Add SSH Keys to Github and Bitbucket



Personal settings

- Profile
- Account
- Security
- Emails
- Notifications
- Billing
- SSH and GPG keys
- Blocked users
- Repositories
- Organizations
- Saved replies
- Applications

Developer settings

SSH keys / Add new

Title

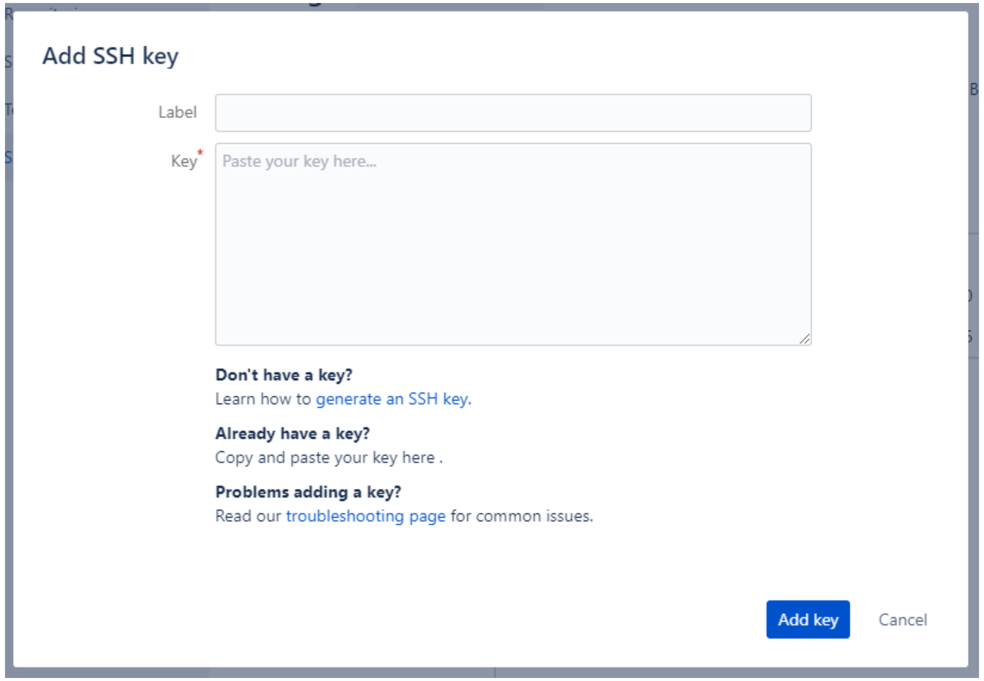
Key

Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

Add SSH key

Github

https://help.github.com/en/enterprise/2.15/user/articles/adding-a-new-ssh-key-to-your-github-account



Add SSH key

Label

Key*

Paste your key here...

Don't have a key?
Learn how to generate an SSH key.

Already have a key?
Copy and paste your key here .

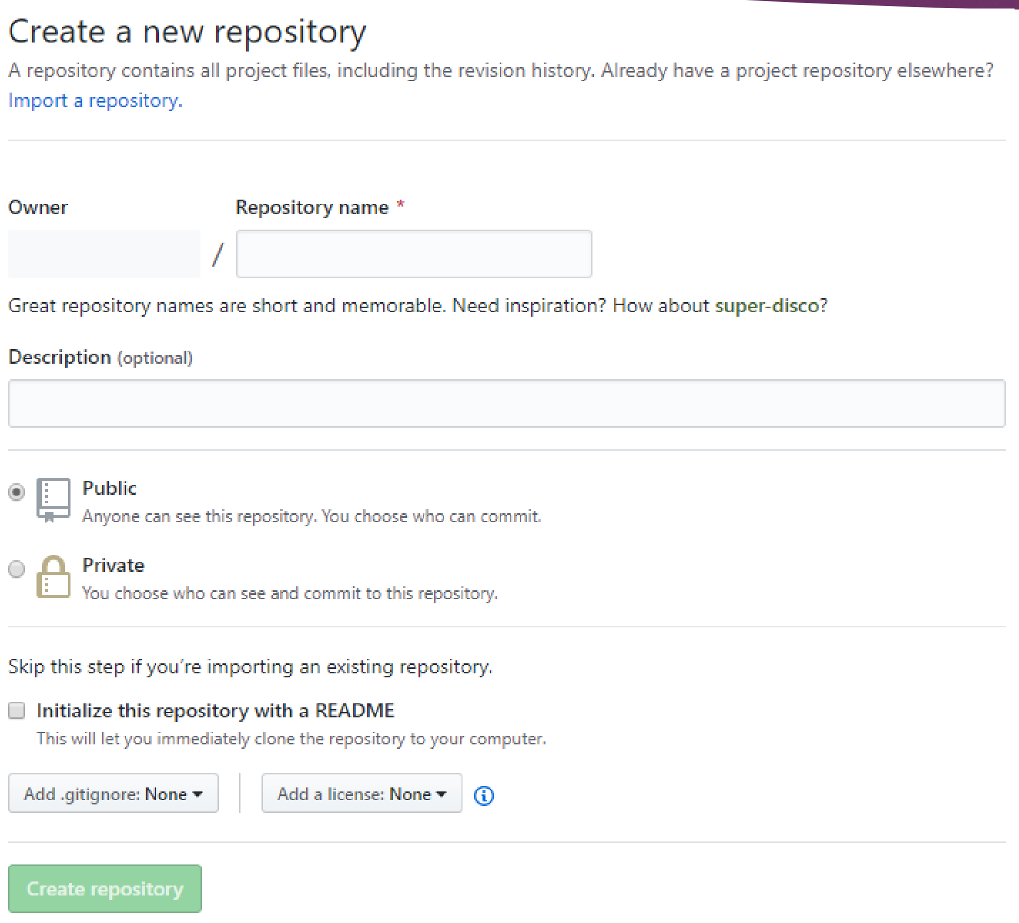
Problems adding a key?
Read our troubleshooting page for common issues.

Add key Cancel

Bitbucket

https://confluence.atlassian.com/bitbucket/set-up-an-ssh-key-728138079.html

Git Init – Github



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner / Repository name *

Great repository names are short and memorable. Need inspiration? How about super-disco?

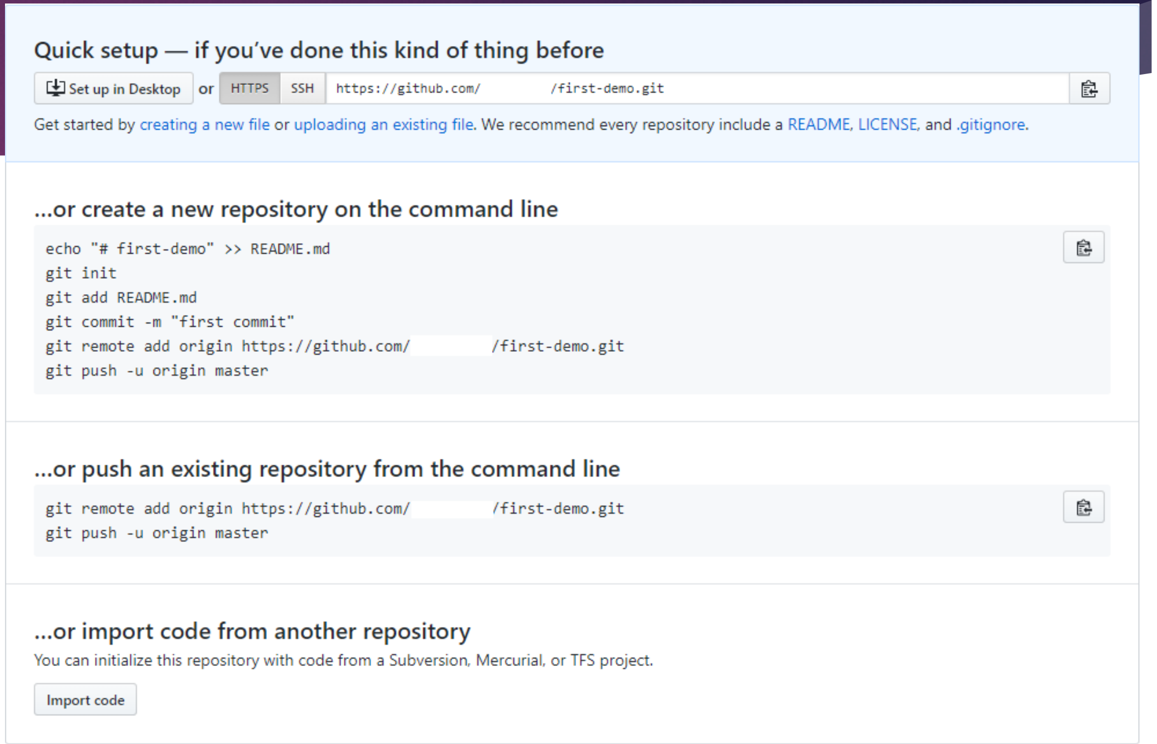
Description (optional)

● Public
Anyone can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

□ Initialize this repository with a README
This will let you immediately clone the repository to your computer.



Quick setup — if you've done this kind of thing before

`https://github.com/          /first-demo.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# first-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/          /first-demo.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/          /first-demo.git
git push -u origin master
```

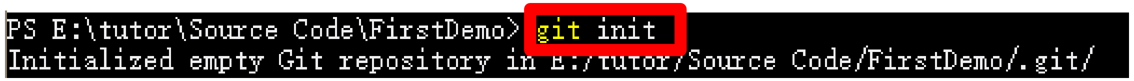
...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

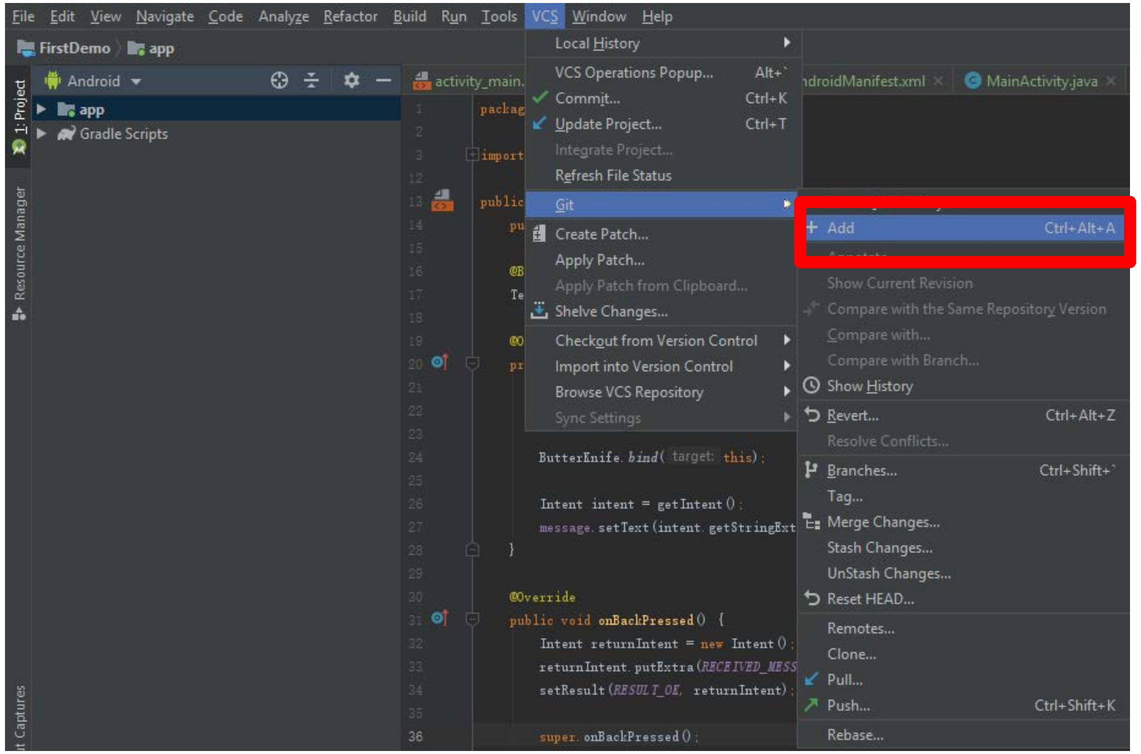
Git push to remote repository (Command line):

```
echo "# first-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/          /first-demo.git
git push -u origin master
```

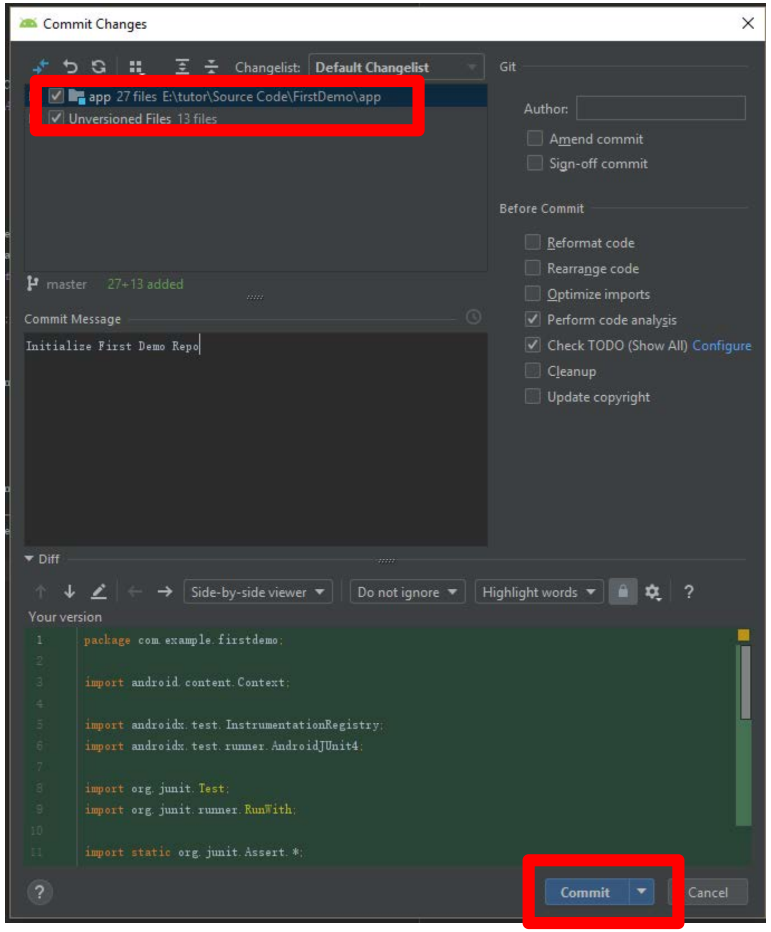
Git Push to Remote Repository (Android Studio)



```
PS E:\tutor\Source Code\FirstDemo> git init
Initialized empty Git repository in E:/tutor/Source Code/FirstDemo/.git/
```



```
packag
import
public
    pu
    @B
    Te
    @O
    pr
        ButterKnife.bind( target: this);

        Intent intent = getIntent();
        message.setText(intent.getStringExt
    }

    @Override
    public void onBackPressed() {
        Intent returnIntent = new Intent();
        returnIntent.putExtra(RECEIVED_MESS
        setResult(RESULT_OK, returnIntent);

        super.onBackPressed();
```



Commit Changes

Changelist: Default Changelist

☑ app 27 files E:\tutor\Source Code\FirstDemo\app

☑ Unversioned Files 13 files

master 27+13 added

Commit Message

Initialize First Demo Repo

Git

Author:

□ Amend commit

□ Sign-off commit

Before Commit

□ Reformat code

□ Rearrange code

□ Optimize imports

☑ Perform code analysis

☑ Check TODO (Show All) Configure

□ Cleanup

□ Update copyright

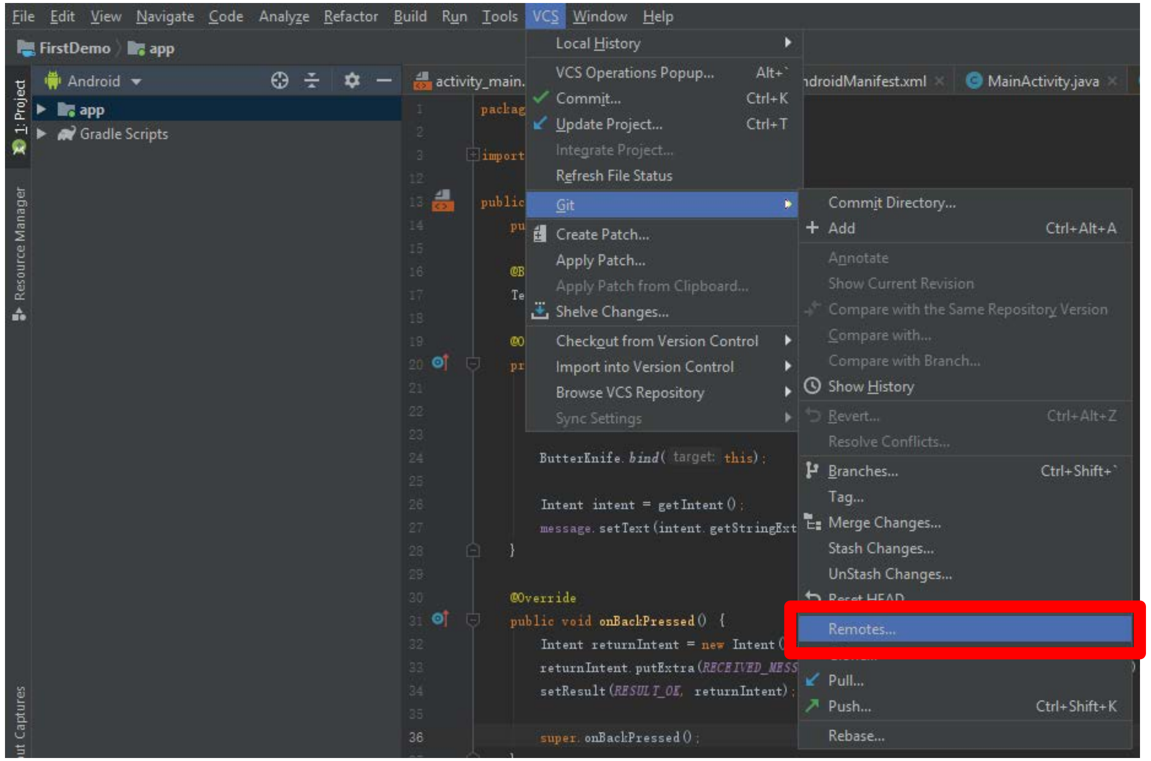
Diff

Side-by-side viewer | Do not ignore | Highlight words

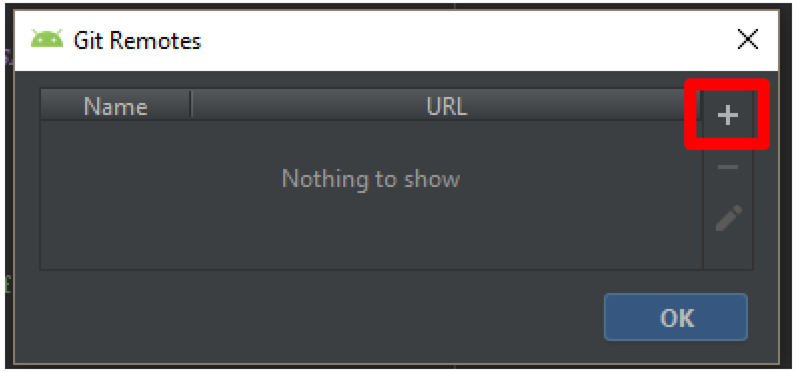
Your version

```
package com.example.firstdemo;

import android.content.Context;

import androidx.test.InstrumentationRegistry;
import androidx.test.runner.AndroidJUnit4;

import org.junit.Test;
import org.junit.runner.RunWith;

import static org.junit.Assert.*;
```

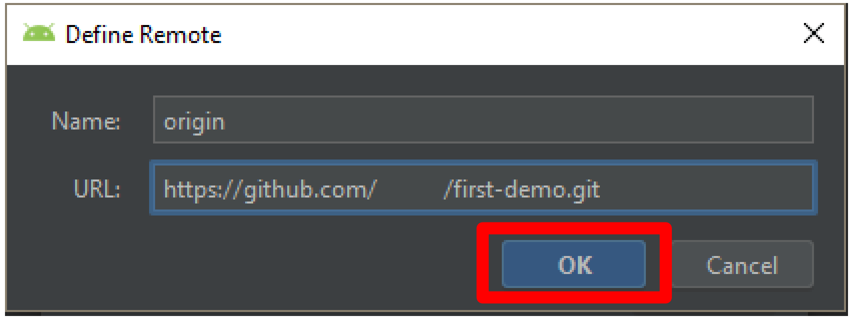
Git Push to Remote Repository (Android Studio)



```
packag
import
public
    pu
    @B
    Te
    @O
    pr
        ButterKnife.bind( target: this);

        Intent intent = getIntent();
        message.setText(intent.getStringExt
    }

    @Override
    public void onBackPressed() {
        Intent returnIntent = new Intent(
        returnIntent.putExtra(RECEIVED_MESS
        setResult(RESULT_OK, returnIntent);

        super.onBackPressed();
```



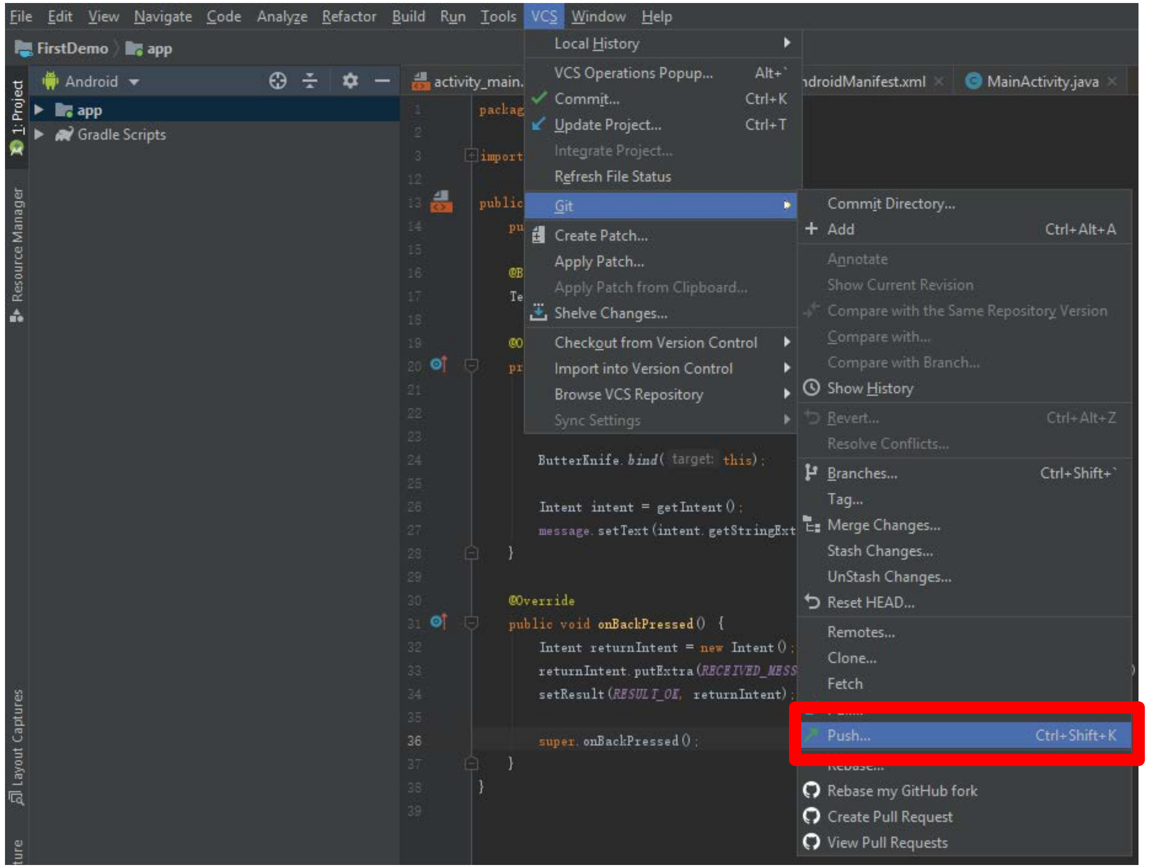
Git Push to Remote Repository (Android Studio)



Define Remote

Name: origin

URL: https://github.com/   /first-demo.git



```
ButterKnife.bind( target: this);

Intent intent = getIntent();
message.setText(intent.getStringExt
}

@Override
public void onBackPressed() {
    Intent returnIntent = new Intent();
    returnIntent.putExtra(RECEIVED_MESS
    setResult(RESULT_OK, returnIntent);

    super.onBackPressed();
}
}
```

Git Tutorial

- ***Git***:

- https://git-scm.com/docs/gittutorial

- ***Bitbucket:***

- https://www.atlassian.com/git/tutorials/setting-up-a-repository

- ***Github:***

- https://try.github.io/