How to Create Your Own Android Library and Publish it in GitHub?

Last Updated : 04 Jun, 2024

Android libraries are tools for developers, that offers reusable code components and functionalities to streamline app development. Creating and publishing your own Android library on GitHub allows you to share your code with the wider developer community, contribute to open-source projects, and establish your expertise. In this article, we'll walk you through the process of creating and publishing your own Android library on GitHub.

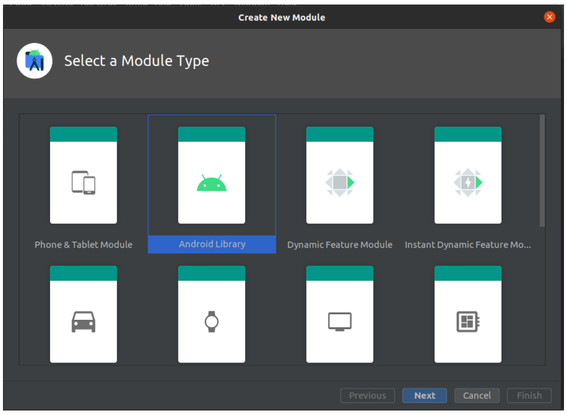
Steps to Create an Android Library

Step 1: Open Android Studio and Create a New Project

To create a new project in Android Studio please refer to How to Create/Start a New Project in Android Studio. Name your application as **BlinkEffectExample** and make sure to select **Kotlin** as the programming language as I'm going to use Kotlin for the source code, but you can follow the same process of **creating & publishing the android library** for Java as well.

Step 2: Now Create a new Module in your project

Go to **file > New > New Module.** Select **Android Library** from the options and click on **Next**.



Select a module type

Name Your Library as **BlinkLibrary** and click **Finish**.

Create a new module

Your module **BlinkLibrary** gets created.

BlinkLibrary is added to the structure

Step 3: Now, it's time to add code to our library. For this go to your module **BlinkLibrary > Java > your package name(like com.learn.blinklibrary)** Right-click go to **New > Kotlin/class file**.

Create a new file in your module and name it as **BlinkEffect** make sure to choose **Object** from the below options.

create a new file

Step 4: Modify BlinkEffect.kt as follows

```
import android.animation.ArgbEvaluator
import android.animation.ObjectAnimator
import android.graphics.Color
import android.view.View
import android.view.animation.Animation

object BlinkEffect {

    fun blink(view: View) {
        // adding the color to be shown
        val animator: ObjectAnimator = ObjectAnimator.ofInt(
            view, "backgroundColor", Color.YELLOW,
            Color.RED, Color.GREEN
        )
        // duration of one color
        animator.duration = 500;
        animator.setEvaluator(ArgbEvaluator())
        // color will be shown in reverse manner
        animator.repeatCount = Animation.REVERSE
        // Repeat up to infinite time
        animator.repeatCount = Animation.INFINITE
        animator.start()
    }
}
```

We are taking view as a parameter to the **blink()** function so that the blink effect can be added to that particular view. Now our library is completed. It's time to publish it, so for this **create a new repository** on your GitHub and push the project to it (if not please create your Github). We will use **JitPack** to publish our library as it makes the process a lot easier in order to publish a library.

Publishing the Android Library in GitHub

Step 1: Create a new GitHub repository.

create a repository – BlinkEffect-library

Copy the **repository's https address**.

copy the repo's address

Step 2: Push your code

Make sure **git** is installed in your system. If Git and GitHub are already set up in the android studio then follow the following steps to push your code.

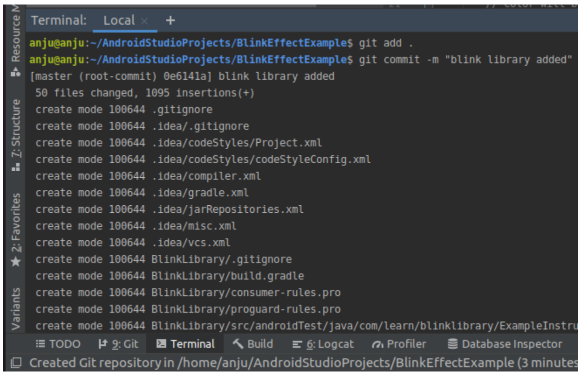
Go to **VCS** (option from menu) **> select Enable version control enable**
A dialogue box opens, select **Git** and click the **OK** button.

You will observe that color of all filename changes to a red color as shown in the below image.

Now, change the file tree structure (left panel) from android to project as shown below.

Right-click the above project name **BlinkEffectExample** then go to **Git > Add.** You will observe the color of all filename changes to green color. Now open the **terminal** of your android studio (find the terminal at bottom of the android studio). Do the following to add, commit and push the code.

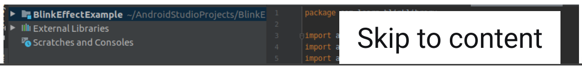
```
-> git add .
-> git commit -m "blink
library added"
```



```
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$ git add .
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$ git commit -m "blink library added"
[master (root-commit) 0e6141a] blink library added
 50 files changed, 1095 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/codeStyles/Project.xml
 create mode 100644 .idea/codeStyles/codeStyleConfig.xml
 create mode 100644 .idea/compiler.xml
 create mode 100644 .idea/gradle.xml
 create mode 100644 .idea/jarRepositories.xml
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/vcs.xml
 create mode 100644 BlinkLibrary/.gitignore
 create mode 100644 BlinkLibrary/build.gradle
 create mode 100644 BlinkLibrary/consumer-rules.pro
 create mode 100644 BlinkLibrary/proguard-rules.pro
 create mode 100644 BlinkLibrary/src/androidTest/java/com/learn/blinklibrary/ExampleInstru
```

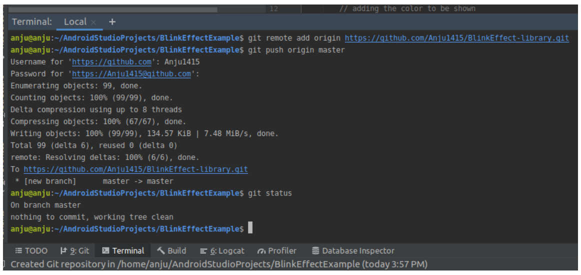
Created Git repository in /home/anju/AndroidStudioProjects/BlinkEffectExample (3 minutes

```
-> git remote add origin "
Paste Your-Repository-
Address"
-> git push origin master
-> Give your GitHub id name &
password
```



Skip to content

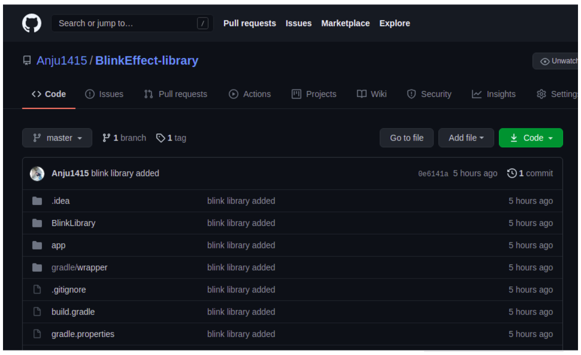
HTML CSS JavaScript TypeScript jQuery AngularJS ReactJS Next.js React Native NodeJS Express.js



```
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$ git remote add origin https://github.com/Anju1415/BlinkEffect-library.git
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$ git push origin master
Username for 'https://github.com': Anju1415
Password for 'https://Anju1415@github.com':
Enumerating objects: 99, done.
Counting objects: 100% (99/99), done.
Delta compression using up to 8 threads
Compressing objects: 100% (67/67), done.
Writing objects: 100% (99/99), 134.57 KiB | 7.48 MiB/s, done.
Total 99 (delta 6), reused 0 (delta 0)
remote: Resolving deltas: 100% (6/6), done.
To https://github.com/Anju1415/BlinkEffect-library.git
 * [new branch]      master -> master
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$ git status
On branch master
nothing to commit, working tree clean
anju@anju:~/AndroidStudioProjects/BlinkEffectExample$
```

Created Git repository in /home/anju/AndroidStudioProjects/BlinkEffectExample (today 3:57 PM)

Your project got **pushed** to the repository!



Anju1415 / BlinkEffect-library

master 1 branch 1 tag

Anju1415 blink library added 0e6141a 5 hours ago 1 commit

.idea	blink library added	5 hours ago
BlinkLibrary	blink library added	5 hours ago
app	blink library added	5 hours ago
gradle/wrapper	blink library added	5 hours ago
.gitignore	blink library added	5 hours ago
build.gradle	blink library added	5 hours ago
gradle.properties	blink library added	5 hours ago

You can also follow another method to push your code (when git and GitHub is not set in the android studio). Open the system's terminal and make sure that your current directory is in “BlinkEffectExample”(folder name of your project) and perform the following commands :

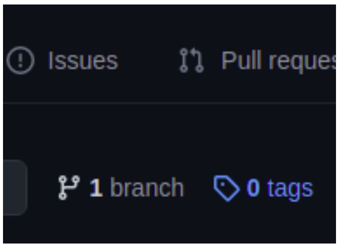
```
-> git init
-> git add
-> git commit -m "blink
library added"
```

Now add your remote origin by performing

```
-> git remote add origin
“Paste Your-Repository-
Address”
-> git remote -v
-> git push origin master
```

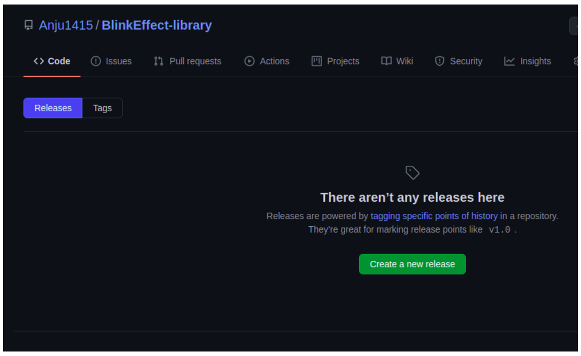
Your application code got pushed to the repository.

Step 3: In your repository, go to tag and select release



click on the tag and choose release

Click on **create a new release**.

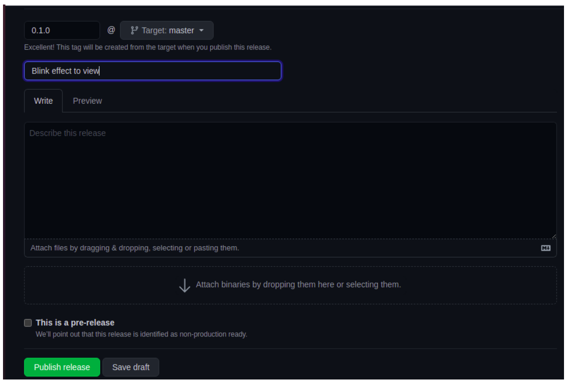


There aren't any releases here

Releases are powered by tagging specific points of history in a repository. They're great for marking release points like `v1.0` .

Now add the release **version** and a

brief introduction about your library.



Excellent! This tag will be created from the target when you publish this release.

Blink effect to view

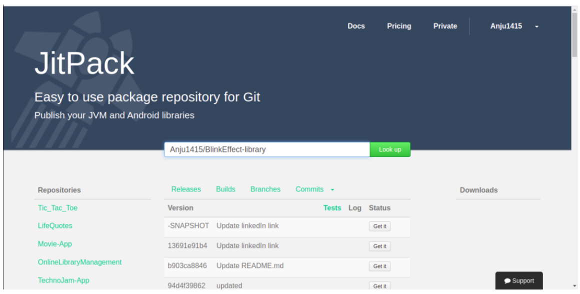
Attach files by dragging & dropping, selecting or pasting them.

Attach binaries by dropping them here or selecting them.

□ **This is a pre-release**
We'll point out that this release is identified as non-production ready.

publish release

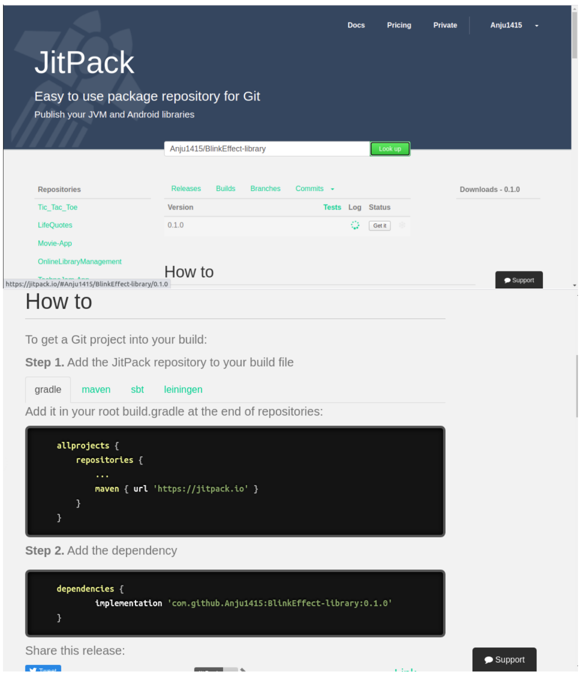
Click on **publish release**. Open a new tab and go to jitpack.io. Insert your **repository address** (in my case it's Anju1415/BlinkEffect-library) and click on **LOOKUP**. Your releases will be listed.



Anju1415/BlinkEffect-library

Version		Tests	Log	Status
-SNAPSHOT	Update linkedIn link			Get it
13691e91b4	Update linkedIn link			Get it
b903ca8846	Update README.md			Get it
94d4f39862	updated			Get it

Now click on **Get it**.



Anju1415/BlinkEffect-library

Version	Tests	Log	Status
0.1.0			Get it

https://jitpack.io/#Anju1415/BlinkEffect-library/0.1.0

How to

To get a Git project into your build:

Step 1. Add the JitPack repository to your build file

Add it in your root build.gradle at the end of repositories:

```
allprojects {
    repositories {
        ...
        maven { url 'https://jitpack.io' }
    }
}
```

Step 2. Add the dependency

```
dependencies {
        implementation 'com.github.Anju1415:BlinkEffect-library:0.1.0'
}
```

Share this release:

Your Android Library is now **published** and is **ready to use**.

Using the Android Library in Your Android App

Step 1: Now you can use this android library in any of your projects. Open any project or create a new one (Here I'm creating a new project to use this library). In your project's **build.gradle (project : applicationName)** add the following line

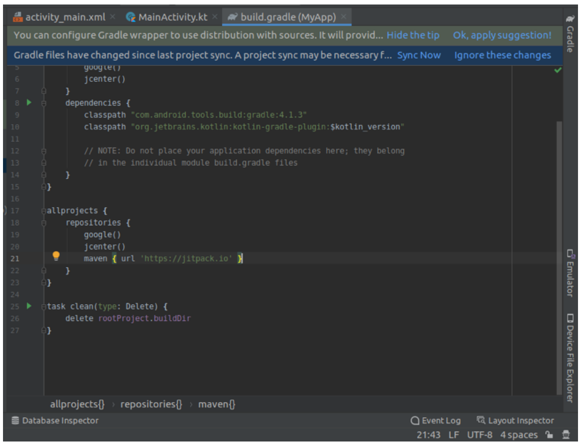
allprojects{

…

maven { url 'https://jitpack.io' }

…

}



You can configure Gradle wrapper to use distribution with sources. It will provid... Hide the tip Ok, apply suggestion!

Gradle files have changed since last project sync. A project sync may be necessary f... Sync Now Ignore these changes

```
        google()
        jcenter()
    }
    dependencies {
        classpath "com.android.tools.build:gradle:4.1.3"
        classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"

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

allprojects {
    repositories {
        google()
        jcenter()
        maven { url 'https://jitpack.io' }
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

and in your app's **build.gradle (module : applicationName.app)** add the dependency.

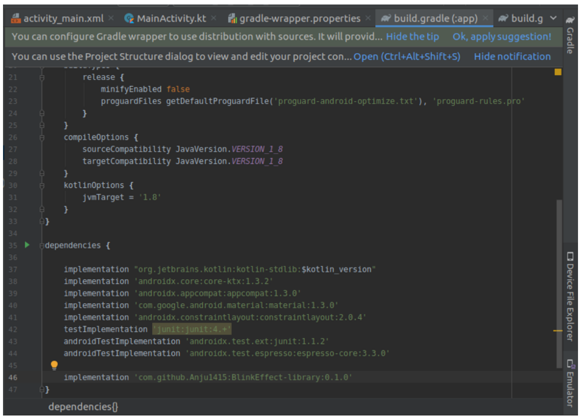
dependencies {

….

implementation 'com.github.Anju1415:BlinkEffect-library:0.1.0'

…

}



activity_main.xml | MainActivity.kt | gradle-wrapper.properties | build.gradle (:app) | build.g

You can configure Gradle wrapper to use distribution with sources. It will provid... Hide the tip   Ok, apply suggestion!

You can use the Project Structure dialog to view and edit your project con... Open (Ctrl+Alt+Shift+S)   Hide notification

```
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}

dependencies {

    implementation "org.jetbrains.kotlin:kotlin-stdlib:$kotlin_version"
    implementation 'androidx.core:core-ktx:1.3.2'
    implementation 'androidx.appcompat:appcompat:1.3.0'
    implementation 'com.google.android.material:material:1.3.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'

    implementation 'com.github.Anju1415:BlinkEffect-library:0.1.0'
}
```

dependencies{}

And now you **can use the library** in your activity.

Step 2: Working with the activity_main.xml file

Navigate to the **app > res > layout > activity_main.xml** and add the below code to that file. Below is the code for the **activity_main.xml** file.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout

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">

    <Button

android:id="@+id/buttonView"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:text="GeeksForGeeks"

android:textAllCaps="false"

app:layout_constraintBottom_toBottomOf="parent"

app:layout_constraintLeft_toLeftOf="parent"

app:layout_constraintRight_toRightOf="parent"

app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Step 3: Working with the MainActivity.kt file

Go to the **MainActivity.kt** file and refer to the following code. Below is the code for the **MainActivity.kt** file. Comments are added inside the code to understand the code in more detail.

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import com.learn.blinklibrary.BlinkEffect

class MainActivity : AppCompatActivity() {

    private lateinit var btn : Button

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

        btn = findViewById(R.id.buttonView)

        btn.setOnClickListener {
            // use of blink-library
            BlinkEffect.blink(btn)
        }
    }
}
```

Output:

00:00 00:06

Source Code on GitHub: https://github.com/Anju1415/BlinkEffect-library

Comment

More info

Advertise with us

Next Article

How to Create a Pull Request on GitHub using Android Studio?

Similar Reads

How to Create Your Own Android Library

Android libraries are tools for developers, that offers reusable code components and functionalities to

5 min read

How to Create a Pull Request on GitHub

Creating a pull request is an important part of collaborating on projects hosted on GitHub. It allows

2 min read

How to Create a New Branch on GitHub

Creating a new branch on GitHub using Android Studio can help your development workflow by

2 min read

How to Add External Library in Android

Android Studio is the official IDE (Integrated Development Environment) for Android app

2 min read

How to Create a New Branch in Git and

Branching in Git is a helpful feature for software developers working on a big team project. It allows

8 min read

How to Create a Tag in a GitHub

Tags are one of the important parts of Git and GitHub. We can see in the case of open source

3 min read

How to Create a Shayari Android App Using

A Shayari app built in Android Studio consists of various Shayaries and categories of it using Firebase

8 min read

How to Create a Custom Intro Slider of an

Intro Slider in many apps is mostly used to educate the users about the app, the features of the app, and

9 min read

How to Use Fast Android Networking

In Android, we know multiple networking libraries like Retrofit, Volley. We use these libraries

4 min read

How to Create a New Branch on Github

Git is an open-source version control system. It means that whenever a developer develops some

2 min read

Corporate & Communications Address:

A-143, 7th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh (201305)

Registered Address:

K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

GET IT ON Google Play

Download on the App Store

Advertise with us

Company

- About Us
- Legal
- Privacy Policy
- In Media
- Contact Us
- Advertise with us
- GFG Corporate Solution
- Placement Training Program
- GeeksforGeeks Community

Languages

- Python
- Java
- C++
- PHP
- GoLang
- SQL
- R Language
- Android Tutorial
- Tutorials Archive

DSA

- Data Structures
- Algorithms
- DSA for Beginners
- Basic DSA Problems
- DSA Roadmap
- Top 100 DSA Interview Problems
- DSA Roadmap by Sandeep Jain
- All Cheat Sheets

Data Science & ML

- Data Science With Python
- Data Science For Beginner
- Machine Learning
- ML Maths
- Data Visualisation
- Pandas
- NumPy
- NLP
- Deep Learning

Web Technologies

- HTML
- CSS
- JavaScript
- TypeScript
- ReactJS
- NextJS
- Bootstrap
- Web Design

Python Tutorial

- Python Programming Examples
- Python Projects
- Python Tkinter
- Web Scraping
- OpenCV Tutorial
- Python Interview Question
- Django

Computer Science

- Operating Systems
- Computer Network
- Database Management System
- Software Engineering
- Digital Logic Design
- Engineering Maths
- Software Development
- Software Testing

DevOps

- Git
- Linux
- AWS
- Docker
- Kubernetes
- Azure
- GCP
- DevOps Roadmap

System Design

- High Level Design
- Low Level Design
- UML Diagrams
- Interview Guide
- Design Patterns
- OOAD
- System Design Bootcamp
- Interview Questions

Inteview Preparation

- Competitive Programming
- Top DS or Algo for CP
- Company-Wise Recruitment Process
- Company-Wise Preparation
- Aptitude Preparation
- Puzzles

School Subjects

- Mathematics
- Physics
- Chemistry
- Biology
- Social Science
- English Grammar
- Commerce
- World GK

GeeksforGeeks Videos

- DSA
- Python
- Java
- C++
- Web Development
- Data Science
- CS Subjects