How to Keep the Device Screen On in Android?

In Android it's seen that screen timeout will be set for 30 seconds or it is set by the user manually in system settings, to avoid battery drain. But there are cases where applications like **stopwatch**, **document reader applications**, **games**, etc, need the screen to be always awake. In this article its been demonstrated, how to keep the device screen awake.

Steps for Keeping the Device Screen On

Step 1: Create a New Project

To create a new project in Android Studio

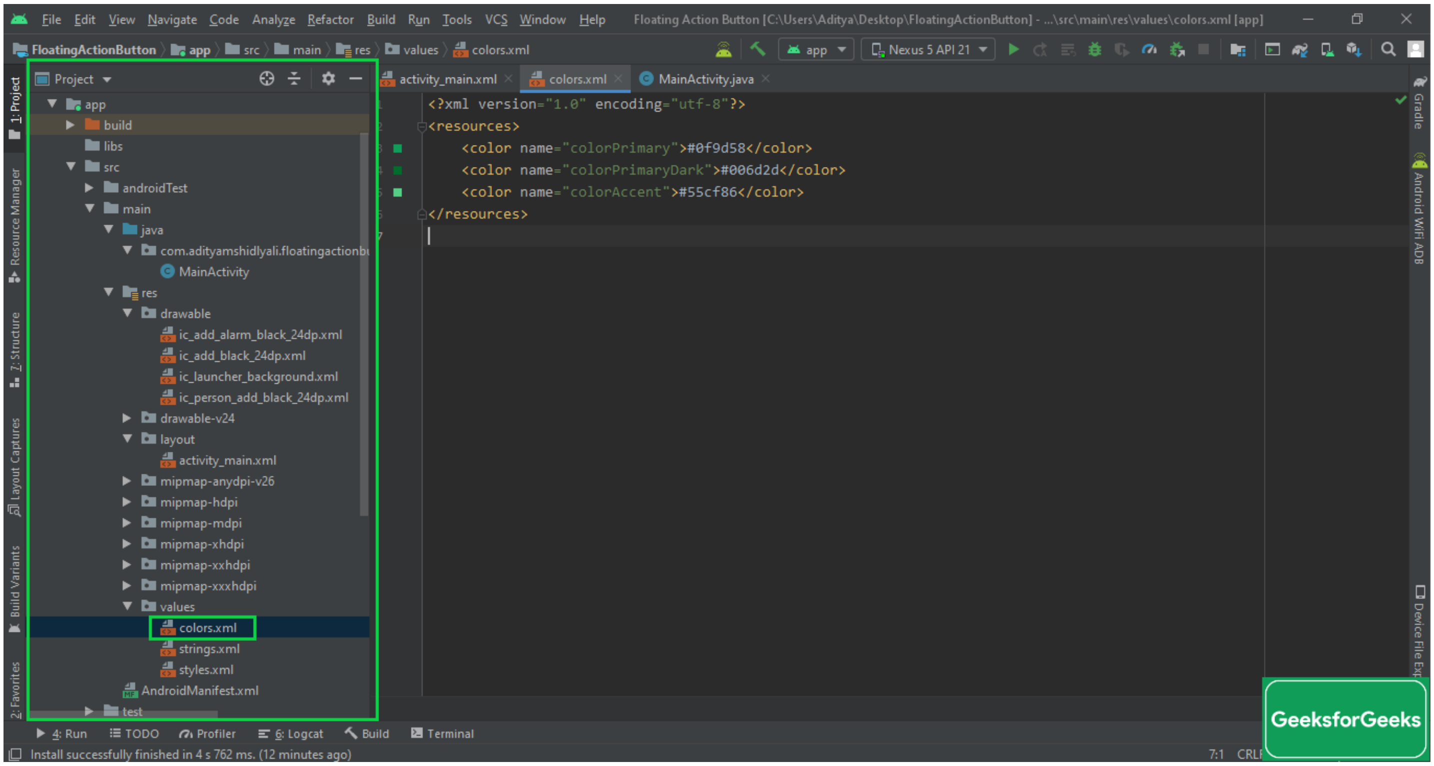
Note that select Java as the programming language.

Step 2: Change the color combination of the base theme of the application

To change the base application theme colors Goto **app -> res -> values -> colors.xml**, and invoke the following color combination.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#0f9d58</color>
    <color name="colorPrimaryDark">#006d2d</color>
    <color name="colorAccent">#55cf86</color>
</resources>
```

- Refer the following image if one has not got the **colors.xml** file:



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#0f9d58</color>
    <color name="colorPrimaryDark">#006d2d</color>
    <color name="colorAccent">#55cf86</color>
</resources>
```

Install successfully finished in 4 s 762 ms. (12 minutes ago)

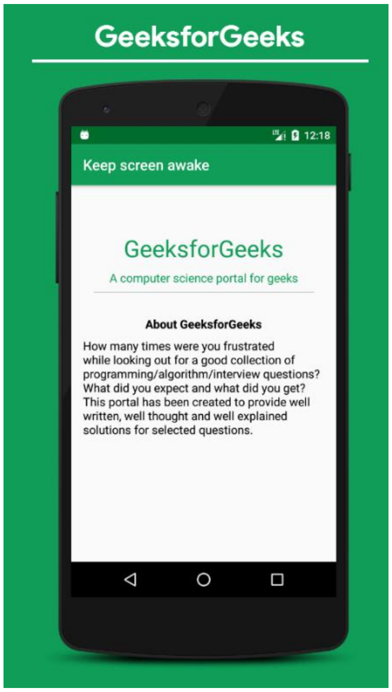
Step 3: Working with the activity_main.xml file

In the **activity_main.xml** file add TextViews to make an app like the document reading application.

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

    <!--This layout contains some simple text views-->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:fontFamily="sans-serif"
        android:gravity="center"
        android:text="GeeksforGeeks"
        android:textColor="@color/colorPrimary"
        android:textSize="32sp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:fontFamily="sans-serif"
        android:gravity="center"
        android:text="A Computer Science portal for geeks"
        android:textColor="@color/colorPrimary"
        android:textSize="16sp" />

    <View
        android:layout_width="300dp"
        android:layout_height="1dp"
        android:layout_gravity="center"
        android:layout_marginTop="8dp"
        android:background="@android:color/darker_gray" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="32dp"
        android:text="About GeeksforGeeks"
        android:textColor="@android:color/black"
        android:textSize="16sp"
        android:textStyle="bold" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:text="How many times were you frustrated while looking out for a good
collection of programming/algorithm/interview questions? What did you expect and what did
```

```
you get? This portal has been created to provide well written, well thought and well
explained solutions for selected questions."
        android:textColor="@android:color/black"
        android:textSize="16sp" />

</LinearLayout>
```

The following output UI is produced:



GeeksforGeeks

A computer science portal for geeks

About GeeksforGeeks

How many times were you frustrated while looking out for a good collection of programming/algorithm/interview questions? What did you expect and what did you get? This portal has been created to provide well written, well thought and well explained solutions for selected questions.

Step 4: Working on keep the device screen awake

There are two methods to implement the screen always awake.

Method 1: Invoking the “keepScreenOn” as true

One can keep the device screen awake by invoking the following attribute in the root view of the application.

android:keepScreenOn = “true”

You can have a look at the following **activity_main.xml** code:

```
<?xml version="1.0" encoding="utf-8"?>

<!--one needs to focus on the keepScreenOn
    in the root view of the application-->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

    android:keepScreenOn="true"
    android:orientation="vertical"
    tools:context=".MainActivity"
    tools:ignore="HardcodedText">

    <!--This layout contains some simple text views-->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="64dp"
        android:fontFamily="sans-serif"
        android:gravity="center"
        android:text="GeeksforGeeks"
        android:textColor="@color/colorPrimary"
        android:textSize="32sp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:fontFamily="sans-serif"
        android:gravity="center"
        android:text="A Computer Science portal for geeks"
        android:textColor="@color/colorPrimary"
        android:textSize="16sp" />

    <View
        android:layout_width="300dp"
        android:layout_height="1dp"
        android:layout_gravity="center"
        android:layout_marginTop="8dp"
        android:background="@android:color/darker_gray" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="32dp"
        android:text="About GeeksforGeeks"
        android:textColor="@android:color/black"
        android:textSize="16sp"
        android:textStyle="bold" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="16dp"
        android:text="How many times were you frustrated while looking out for a good
collection of programming/algorithm/interview questions? What did you expect and what did
you get? This portal has been created to provide well written, well thought and well
explained solutions for selected questions."
        android:textColor="@android:color/black"
        android:textSize="16sp" />

</LinearLayout>
```

Method 2: Keep screen on programmatically

Now you can remove the attribute **android:keepScreenOn="true"** from the activity_main.xml file and the rest the code remains the same and invoke the following code in **MainActivity.java file.**

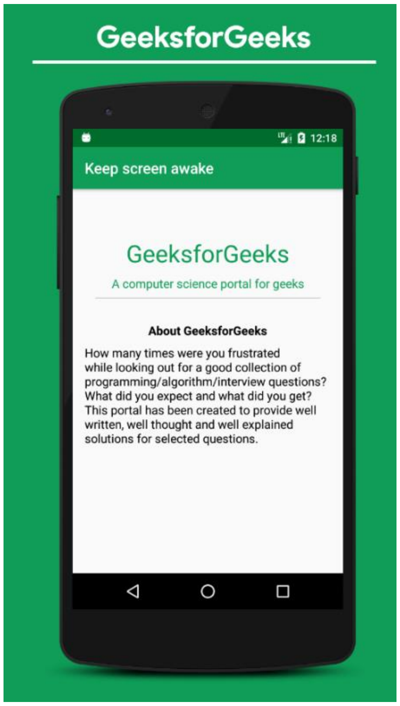
getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);

The complete code is given below.

```
import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.WindowManager;

public class MainActivity extends AppCompatActivity {

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

        // setting up the flag programmatically so that the
        // device screen should be always on
        getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
    }
}
```

Output:

The output is produced as the following image (it is recommended to test this application in a physical android device so that you can see the result whether the app screen is awake or not) and both method's output will remain the same:



About GeeksforGeeks

How many times were you frustrated while looking out for a good collection of programming/algorithm/interview questions? What did you expect and what did you get? This portal has been created to provide well written, well thought and well explained solutions for selected questions.

Which method is recommended?

Both the methods are the same and one can use whichever it feels better, but **implementing this programmatically is recommended** because in complex android applications, developers set the many flags in a particular activity and it becomes easy to get those all flags and manually disable and manage them.