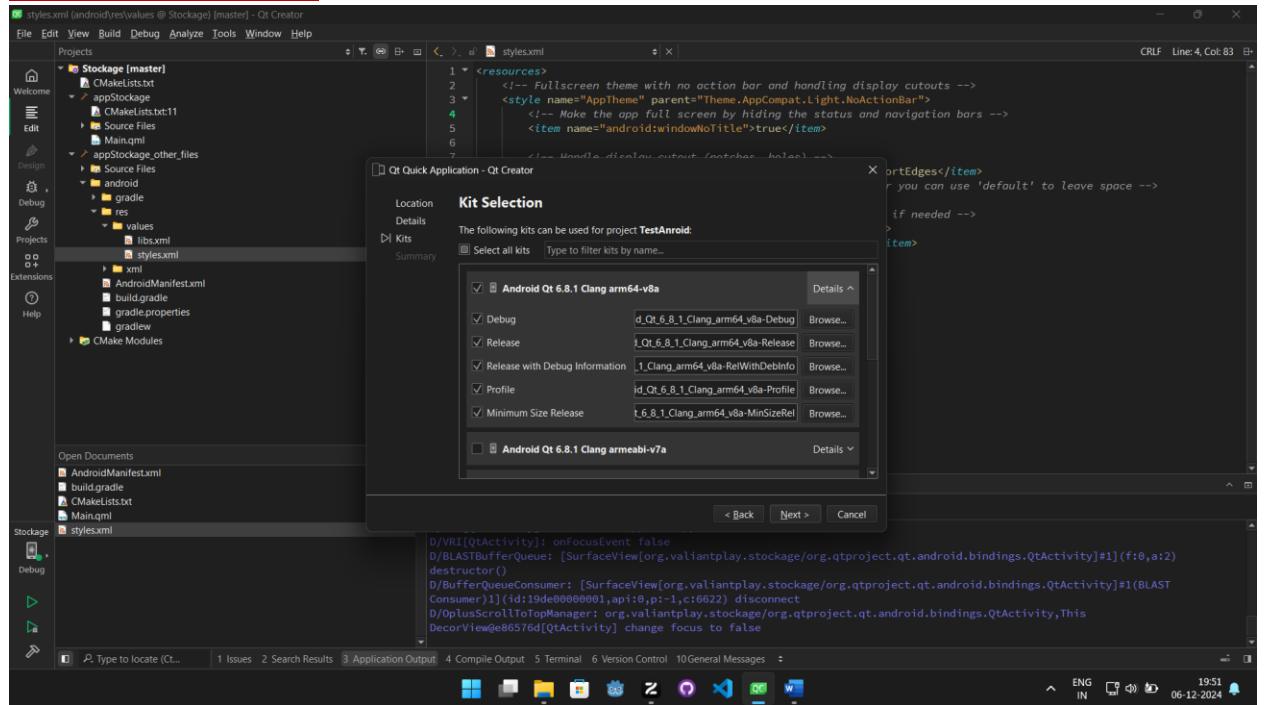
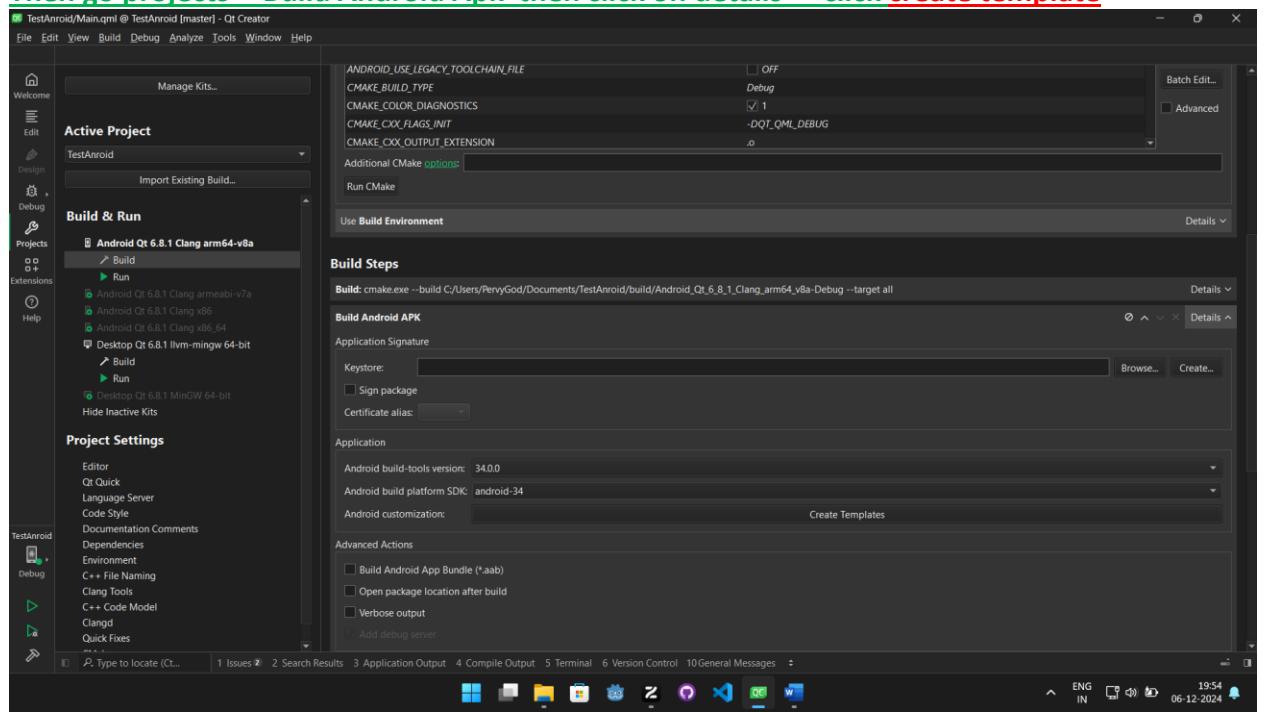


# Steps to Build Application for Android using Qt QML

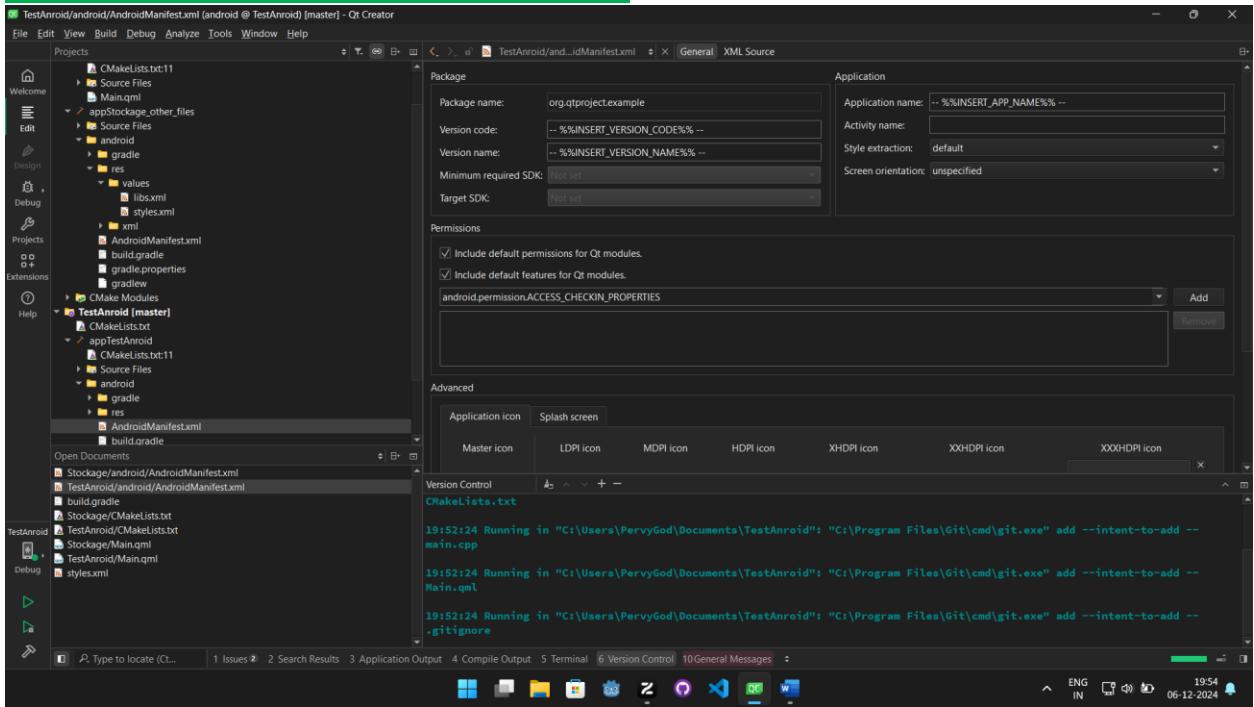
## ● Create Project



## ● Then go projects ->Build Android Apk then click on details -> click create template



- **Edit this according to your requirements**



## QML

```

import QtQuick
import QtQuick.Controls
ApplicationWindow {
    width: 640
    height: 480
    visible: true
    Rectangle
    {
        anchors
        {
            left:parent.left
            right:parent.right
            top:parent.top
        }
        height:parent.height*0.1
        color:"black"
        Text
        {
            text:"Made with C++ and QML"
            anchors.centerIn:parent
            color:"white"
            font.pixelSize:16
        }
    }
}

```

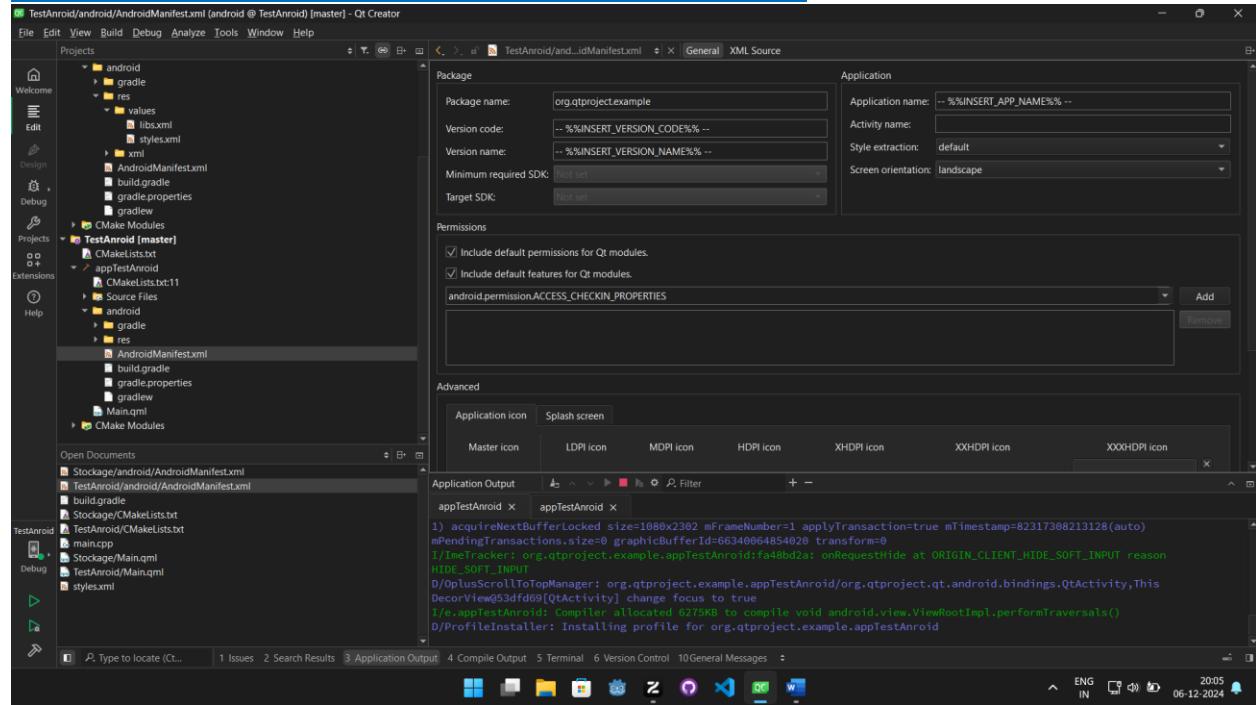
}

20:00 19

48.0 KB/S LTE 5G 100%

Made with C++ and QML

As u can see if I even set the screen orientation to landscape it's still running in portrait mode  
That's because our manifest.xml is not linked to our app yet



in order to use this AndroidManifest.xml in your project u have to edit CMake.txt

**CMAKE CODE :**

```
set_property(TARGET app${PROJECT_NAME} APPEND PROPERTY
    QT_ANDROID_PACKAGE_SOURCE_DIR ${CMAKE_CURRENT_SOURCE_DIR}/android
)

qt_finalize_executable(app${PROJECT_NAME})
```

***NOTE: don't add these lines to top of ur cmake file append these lines to end of ur cmake***

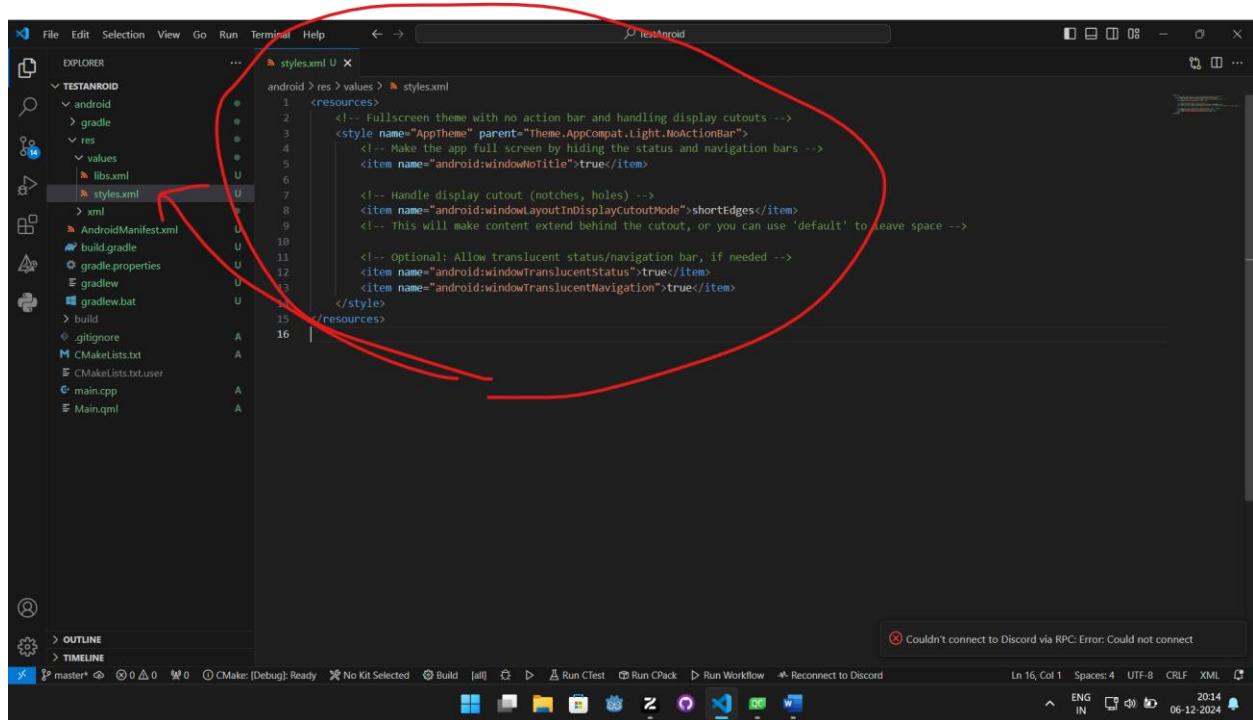
***After that rebuild***

***As you can see now our app is in landscape orientation***



## Customizing android theme:

[Create styles.xml in ur project\\_name/android/res/values/](#)



**Add this code to ur styles.xml or customize ur theme according to ur need**  
**styles.xml:**

```
<resources>
    <!-- Fullscreen theme with no action bar and handling display cutouts -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
        <!-- Make the app full screen by hiding the status and navigation bars -->
        <item name="android:windowNoTitle">true</item>
        <item name="android:windowFullscreen">true</item>
        <!-- Handle display cutout (notches, holes) -->
        <item name="android:windowLayoutInDisplayCutoutMode">shortEdges</item>
        <!-- This will make content extend behind the cutout, or you can use 'default' to leave space -->

        <!-- Optional: Allow translucent status/navigation bar, if needed -->
        <item name="android:windowTranslucentStatus">true</item>
        <item name="android:windowTranslucentNavigation">true</item>
        <!-- Customize your theme here. -->
        <item name="colorPrimary">#000000</item>
        <item name="colorPrimaryDark">#000000</item>
        <item name="colorAccent">#0000FF</item>
        <item name="android:statusBarColor">#000000</item>
    </style>
</resources>
```

**Now if you try to run you will face this gradle error**

```
> Task :stripDebugDebugSymbols

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':processDebugResources'.
> A failure occurred while executing com.android.build.gradle.internal.res.LinkApplicationAndroidResourcesTask$TaskAction
  > Android resource linking failed
    ERROR: AAPT: error: resource style/Theme.AppCompat.Light.NoActionBar (aka org.qtproject.example.appTestAndroid:style/
Theme.AppCompat.Light.NoActionBar) not found.
    error: failed linking references.

* Try:
> Run with --stacktrace option to get the stack trace.
> Run with --info or --debug option to get more log output.
> Run with --scan to get full insights.
> Get more help at https://help.gradle.org.

BUILD FAILED in 8s
28 actionable tasks: 9 executed, 19 up-to-date
Building the android package failed!
-- For more information run this command with --verbose
```

To solve this you have to edit gradle file

Like this

4 mavenCentral()
5 }
6
7 dependencies {
8 classpath 'com.android.tools.build:gradle:8.6.0'
9 }
10 }
11
12 repositories {
13 google()
14 mavenCentral()
15 }
16
17 apply plugin: qtGradlePluginType
18
19 dependencies {
20 implementation fileTree(dir: 'libs', include: ['\*.jar', '\*.aar'])
21 //noinspection GradleDependency
22 implementation 'androidx.core:core:1.13.1'
23 implementation 'androidx.appcompat:appcompat:1.2.0' // or a more recent version
24
25 }
26
27 android {
28 \*\*\*\*
29 \* The following variables:
30 \* - androidBuildToolsVersion,
31 \* - androidCompileSdkVersion
32 \* - qtAndroidDir - holds the path to qt android files
33 \* - needed to build any Qt application
34 \* - on Android.
35 \* - qtGradlePluginType - whether to build an app or a library
36 \*
37 \* are defined in gradle.properties file. This file is

We have to add dependencies to our build.gradle

implementation 'androidx.appcompat:appcompat:1.2.0' // or a more recent version

you can comment this

15 qt\_add\_qml\_module(appTestAndroid
16 URI TestAndroid
17 VERSION 1.0
18 QML\_FILES
19 Main.qml
20 #RESOURCES android/AndroidManifest.xml android/build.gradle android/res/values/libs.xml android/re
21 )
22
23 # Qt for iOS sets MACOSX\_BUNDLE\_GUI\_IDENTIFIER automatically since Qt 6.1.
24 # If you are developing for iOS or macOS you should consider setting an
25 # explicit, fixed bundle identifier manually though.
26 set\_target\_properties(appTestAndroid PROPERTIES
27 # MACOSX\_BUNDLE\_GUI\_IDENTIFIER com.example.appTestAndroid

And after this to use styles.xml we just create in our application we have edit manifest.xml file

android:theme="@style/AppTheme" in your <application>

The screenshot shows the Qt Creator IDE interface. The left sidebar displays the project structure under 'Projects'. It includes a 'Stockage [master]' folder and a 'TestAndroid [master]' folder, which contains files like CMakeLists.txt, appTestAndroid, appTestAndroid\_other\_files, Source Files, android, gradle, res (with values, libs.xml, styles.xml), xml (with AndroidManifest.xml), build.gradle, gradle.properties, and gradlew. Below these are CMake Modules. The right pane shows the content of 'AndroidManifest.xml' as an XML source. The code is as follows:

```
<?xml version="1.0"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="org.qtproject.qt.android.bindings.QtApplication">
    <!-- %&INSERT_PERMISSIONS -->
    <!-- %&INSERT_FEATURES -->
    <supports-screens android:anyDensity="true" android:largeScreens="true" android:resizeable="true" android:smallScreens="true" />
    <application android:name="org.qtproject.qt.android.bindings.QtApplication" android:label="Qt Application">
        <activity android:theme="@style/AppTheme" android:name="org.qtproject.qt.android.bindings.QtActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <meta-data android:name="android.app.lib_name" android:value="-->" />
            <meta-data android:name="android.app.arguments" android:value="-->" />
        </activity>
        <provider android:name="androidx.core.content.FileProvider" android:authorities="org.qtproject.qt.android.bindings.QtFileProvider" android:exported="false">
            <meta-data android:name="android.support.FILE_PROVIDER_PATHS" android:resource="@xml/provider_paths" />
        </provider>
    </application>
</manifest>
```

Build and test it

That's it 😊

Best regards

Vishal Ahirwar