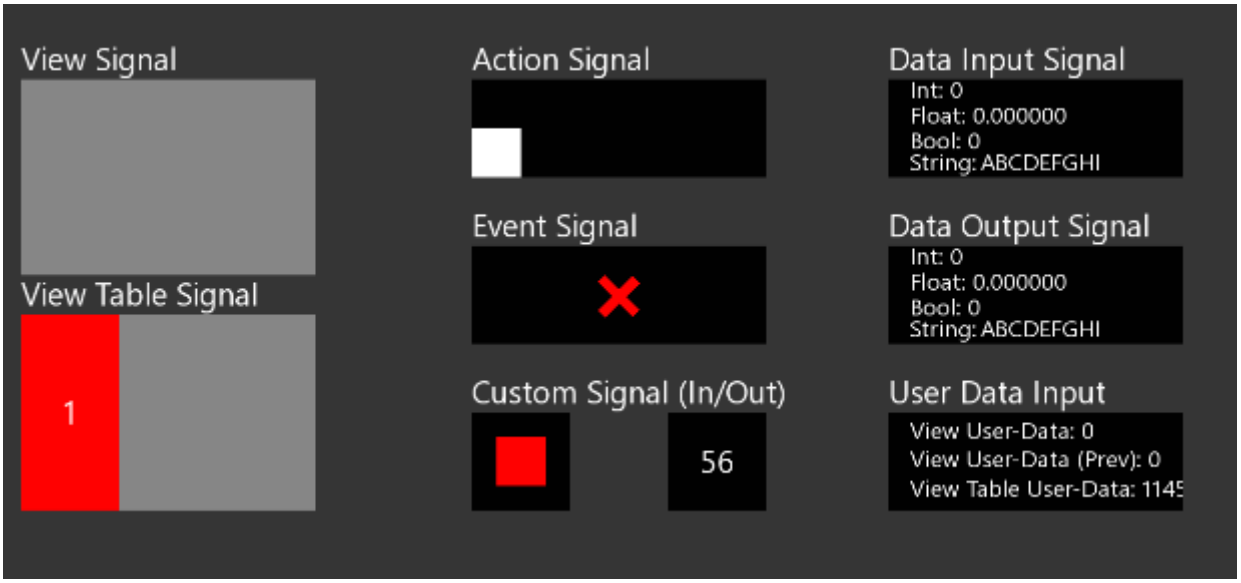
MLC

System Requirements

This document mandates adherence to the Android Application Development Environment Setup Guide, as a prerequisite for system compliance.

By following these steps, you should have a consistent and well-configured development environment for your Android application on Windows 10/11. Remember to restart your system after making changes related to CMake in Android Studio and after updating Environment Variables.

<Signals>



Project Structure

NativeViewLibrary is using the Android Gradle Plugin Version 3.5.3 with Gradle Version 5.4.1. The Application is tested with ndk-26 that can be installed from the document in chapter Configuration and System Requirements. The supported API range is 24-28. The supported ABIs are arm64-v8a, x86_64, x86. The MLC Android Application is NOT using the NativeView library. The build process is different than the Android Applications that use the NativeView library.

Building and Running the Android Application

To be able to build and run the Android Application, the following steps must be done.

1. CGI Studio

These steps are required to prepare the asset from the solution to use in the CMake and Android Studio steps.

1. Subst [windows-commands/subst] the root of CGI Studio.
   - Open a Command Prompt in the root of CGI Studio directory e.g., “...\cgistudio” and type: “subst
     x: .” and click enter. The CGI root is now available under the subst "x" drive in Windows Explorer.
2. Run CGI Studio by double clicking on “X:\bin\SceneComposer\SceneComposer_Android_OpenGLES30.bat”,
3. Open the following solution for MLC <Signals/Cluster> from X:\cgi_studio_matlab_connector\apps\sample\<Signals/Cluster>\src\SCSolution\Android\Solution.scs”.
4. Save this solution as:
   1. AssetLibCff.bin and then copy it into the /sdcard/ directory of the target/emulator. For the emulator this can be done in the "Device Explorer" of the emulator in Android Studio.
   2. AssetLibCff_Android_Target.bin and then copy it to “X:\cgi_studio_matlab_connector\apps\sample\<Signals/Cluster>\src\Asset”. This one is needed to build the ".so" library required by the Android Project.

2. CMake

These steps are to prepare the ".so" library required by the Android application. Make sure the steps of "Alternative CMake Approach" have been done prior to the next step. Note: Currently the ninja build can only succeed when the [cmake | CMAKE_BUILD_TYPE = Release] in CMake during configuration.

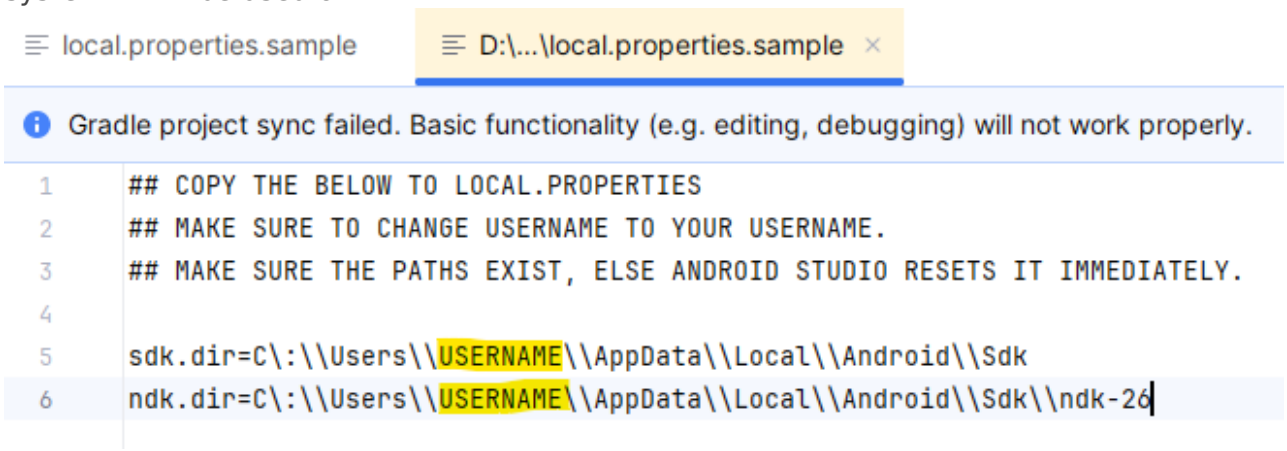
1. Open CMake and choose "X:\cgi_studio_matlab_connector\apps\sample\Signals\src\Application" as source code and e.g. X:/build for where to build the binaries. Click Configure.
2. Specify "Ninja" as the generator for this project and choose "Specify toolchain file for cross-compiling". Click Next.
3. Choose "X:\cgi_studio_devices\src\Android\ToolchainFiles\android.toolchain-ndk26-generic.cmake". Click Finish.
4. The first "error" will ask to set the "ANDROID_ABI", choose the ABI required. This should match the target/emulator CPU Architecture. Click Configure.
5. The next "error" will ask to set the ANDROID_PLATFORM. This should match the target/emulator Android version. Click Configure until there are no red lines anymore and then click on Generate.
6. Open CMD and navigate to "X:\build" by typing first "x:" and clicking enter, and then "cd X:\build" and clicking enter.
7. Type "ninja" and click enter for the build to start.

8. After the build is done, copy "X:\build\libApplicationLib.so" to the directory matching the ABI previously selected e.g. "X:\cgi_studio_apps\References\Android\SampleApp\app\native-libs\x86_64\libApplicationLib.so".

3. Android Studio

These steps are to prepare the Android Studio to build and run the Android application.

1. Open the Android Application in Android Studio from “X:\cgi_studio_apps\References\Android\SampleApp”.
2. Open the “local.properties.sample” file and change the “USERNAME” in the SDK and NDK path to match the system it will be used on.



3. Copy the contents of the "local.properties.sample" to “local.properties”. Note that if the SDK or NDK paths do not exist, Android Studio will reset the “local.properties” again after trying to Sync Project with Gradle Files.
4. Make sure the JDK is set to 11 in [File | Settings | Build, Execution, Deployment | Build Tools | Gradle]

   like described in chapter Java Development Kit (JDK).
5. Choose the supported ['arm64-v8a', 'x86_64', 'x86'] ABI "mAbi" and set the application name "mApplicationName" in “build.gradle (Project: Module :app)”.

build.gradle

Gradle project sync failed. Basic functionality (e.g. editing, debugging) will not work properly.

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

def mAbi :String = 'x86_64' // choose from: 'armó4-v8a', 'x86_ó4', 'x86'
def mPackageName :String = "at.cgistudio.sampleapp"
def mApplicationName :String = "Name" // choose the application name
def mMinApi :Integer = 24
def mMaxApi :Integer = 28
def mVersionCode :Integer = 6
def mVersionName :String = "1.6"
def mFlavorDimension :String = "split"

android {
```

6. Run the Application by selecting the target/emulator and clicking on “run app”.
7. The first time the Application starts, it will request permissions. Click accept and restart the application. The solution should be running.

Troubleshooting

- If encountering any issues during the setup, refer to the official documentation for Android Studio,
  CMake, and NDK for troubleshooting steps.
- If you encounter any issues linked to outdated source code, reach out to the Android developers for
  assistance in implementing a resolution.
- Check for updates regularly and keep your development environment components up to date.

Platform Tests

- <Signals> Emulator Medium Phone, x86_64, Android 9, API 28 (Without Play/APIs)

Revision #3
Created 2024-02-21 22:38:57 UTC by Kanai Tomoaki
Updated 2024-05-28 07:30:32 UTC by Kanai Tomoaki