Android Packaging Scripts for continued deployments

That title is a mouthful. When working with an Android project, it could be that your assets (.PNGs, .OGGs, .INIs) are coming from another place in another project file structure. The same situation for your source files.

So for changes to your project, you'll need to periodically copy all your assets and sources manually into the Android file structure for building. This gets very tiring very quickly.

I have created a couple of scripts that help me clear out old files and copy in new files from my source projects. Saves time and helps to get test .APKs across to the device a lot quicker.

This is my current solution structure for my Windows, MacOSX and Android projects:

- MyGame
  - bin
    - mac
    - windows
    - *.ini
  - build
    - android
      - assets
      - bin
      - jni
    - mac
    - windows
    - ←— scripts will go here
    - premake4.lua
  - data
  - include
  - lib
  - src

Please note above, The MacOSX and Windows build projects both share ini files at MyGame/bin and share the source files at: MyGame/src. However, the Android build doesn't share with the others. Files are copied into place like this:

```
MyGame/src/*                =>    MyGame/build/android/jni/
MyGame/bin/*.ini            =>    MyGame/build/android/assets/orx.ini
MyGame/data/* (all folders)     =>    MyGame/build/android/assets/
```

This is the “clean” script (**androidclean.bat**), which takes care of clearing stale files out of the android folders:

```
REM -- Clear existing files from folders first.

cd android
IF NOT EXIST .\assets GOTO NEXT
```

```
cd assets
del /Q *.png
del /Q *.ogg
del /Q *.ini
cd ..

:NEXT
cd jni
del /Q *.cpp
del /Q *.h
cd ..

:QUIT
cd ..
REM pause
```

This script is handy for cleaning up your source tree before commiting to a repository (if you use one).

Second, the “packaging” script (**androidpackage.bat**) makes use of the **androidclean.bat** script above. It then continues on to copy all the nescessary files in place for you:

```
REM ---- android packager will output
REM ---- prepackaged files to build into
REM ---- build/android.
REM ---- Import build/android into eclipse.


REM -- Clear expect files from folders first.

call androidclean.bat

REM -- List and copy data to the assets folder
cd android
IF NOT EXIST assets mkdir assets

cd ..
cd ..
cd data
cd
REM pause
for /D %%f in (.\*) do copy "%%f\*" "..\build\android\assets\"
REM pause

REM -- Concat all located ini files into the assets folder
cd ..
cd bin
copy *.ini ..\build\android\assets\orxd.ini

REM -- Copy source files to the jni folder
```

```
cd ..
cd src
for /D %%f in (.\*) do copy "%%f\*.cpp" "..\build\android\jni\"
for /D %%f in (.\*) do copy "%%f\*.h" "..\build\android\jni\"

pause
```

For the script to work, the scripts must reside at: MyGame/build. That way all the paths will work correctly. If your project structure is arranged differently, you can alter the scripts to suit.

Hopefully these files will help you to perform continuous deployments from your main working project to your Android project by running one script.

From:
https://www.orx-project.org/wiki/ - **Orx Learning**

Permanent link:
https://www.orx-project.org/wiki/en/tutorials/android/android_project_packaging

Last update: **2025/09/30 17:26 (14 months ago)**