Integrating HOOPS Native Platform w/Android

Virtual Training – December 1st, 2020

Tony Tyrrell – Tony.Tyrrell@techsoft3d.com

TECHSOFT 3D
Dependable. Dynamic. Durable.

Agenda

- Dependencies & Setup
  - What you need to follow along.
- Introduction to the HOOPS Native SDKs
  - Data Visualization
  - Data Conversion/Import.
- Basic Android Overview
  - Explain Android Components Involved
- The Java Native Interface (JNI)
- Use the HOOPS Native SDKs! (Fill in the blanks)
- Help, Questions, etc..

Dependencies

TECH SOFT 3D
Dependable. Dynamic. Durable.

- Android Studio (4.1.1)
  - Native Dev Kit (NDK) v18 – Install in Android Studio
  - CMake 3.10.2 - https://cmake.org/download/
  - Python >2.7
- HOOPS Visualize for Android (2020 Service Pack 2)
  - https://developer.techsoft3d.com/hoops/hps/downloads/latest/
  - https://virtual-training-2020.s3.amazonaws.com/HOOPS_Visualize_2020_SP2_Android.zip
- Skeleton Project:
  - https://virtual-training-2020.s3.amazonaws.com/Visualize_Training.zip
- Android Virtual Device (This demo uses API 28, should be compatible w/ 14+)

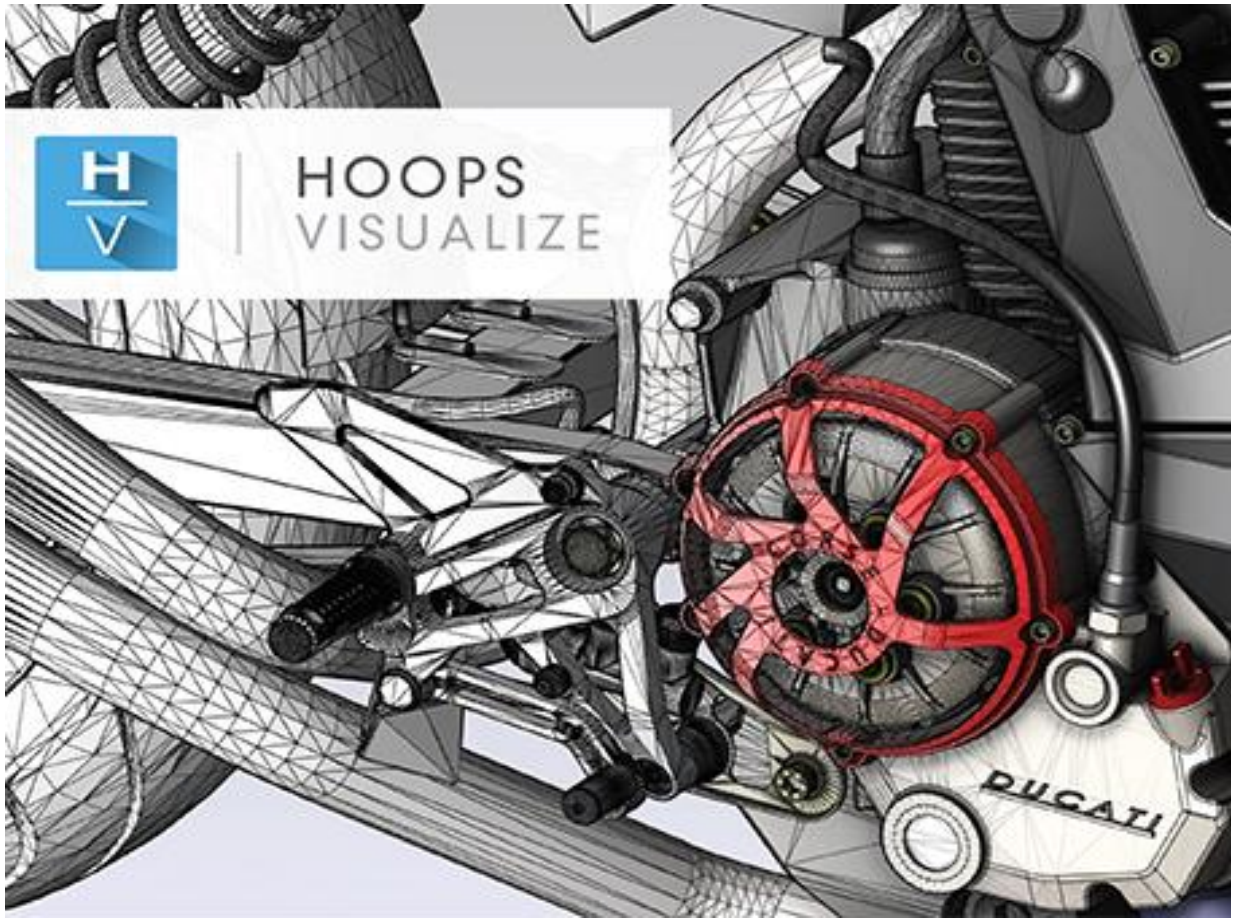
HOOPS Native Platform

HOOPS Visualize – 3D CAD Visualization SDK

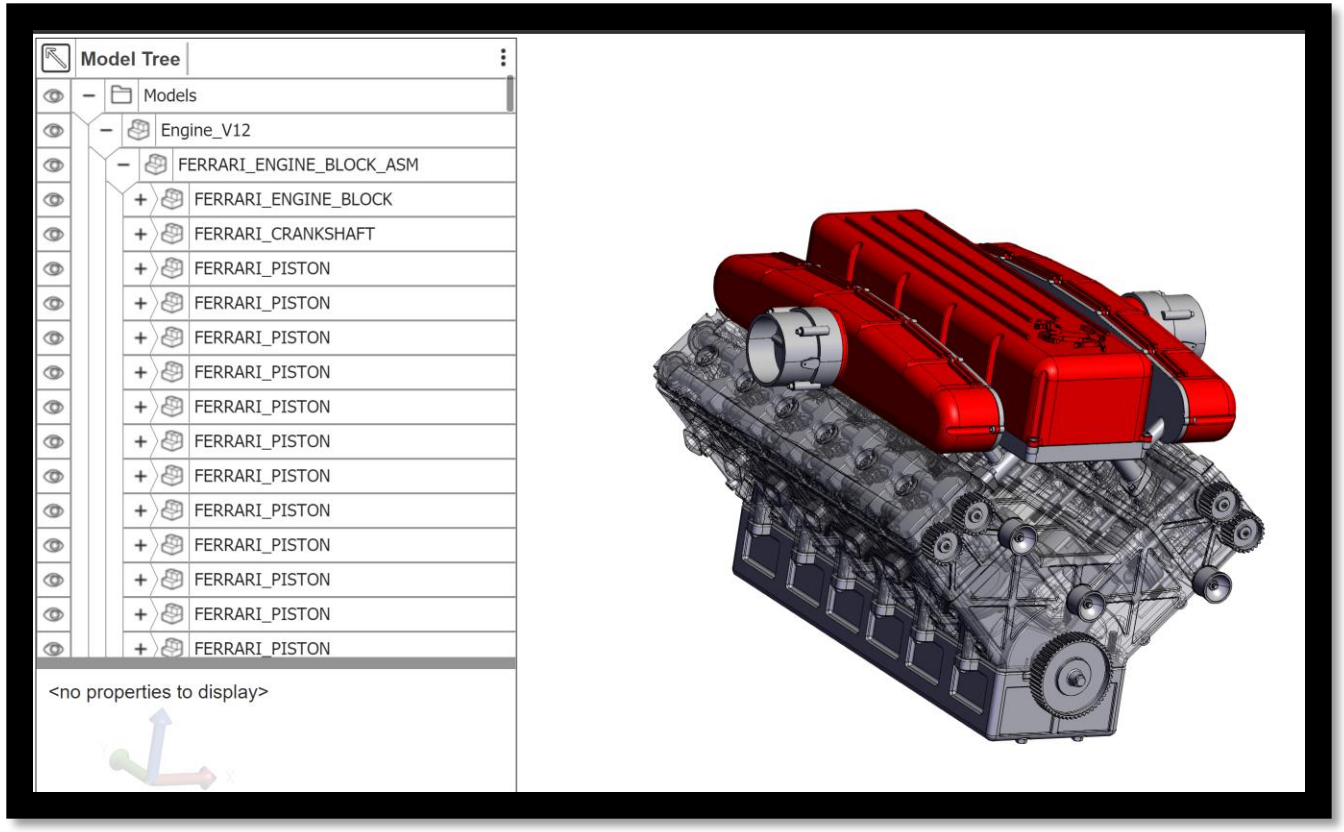
TECH SOFT 3D
Dependable. Dynamic. Durable.

HOOPS Visualize

- High Performance 3D CAD Engineering SDK
- Create/Enhance professional-grade engineering Applications.
- Support for: C, C++, C#
- Support for multiple Graphics Drivers



HOOPS Exchange – CAD Translation SDK



HOOPS Exchange

- Access to 30+ CAD Formats in one Toolkit
- Enrich and empower applications with 3D CAD Access and Translation
- Available on Windows/Linux/Mac & Mobile

TECH SOFT 3D
Dependable. Dynamic. Durable.

Android Native Development

This Photo by Unknown author is licensed under CC BY-SA-NC.

Java Native Interface

TECH SOFT 3D
Dependable. Dynamic. Durable.

- Allow Java interoperability with C/C++, Assembly, etc.
- Programming Interface – not language
- Generate the *Interface Language* Programatically! (swig.org)

Java code using **native** keyword

```
public class Main {
    public native int square(int i);
    public static void main(String[] args) {
        System.loadLibrary("Main");
        System.out.println(new Main().square(2));
    }
}
```

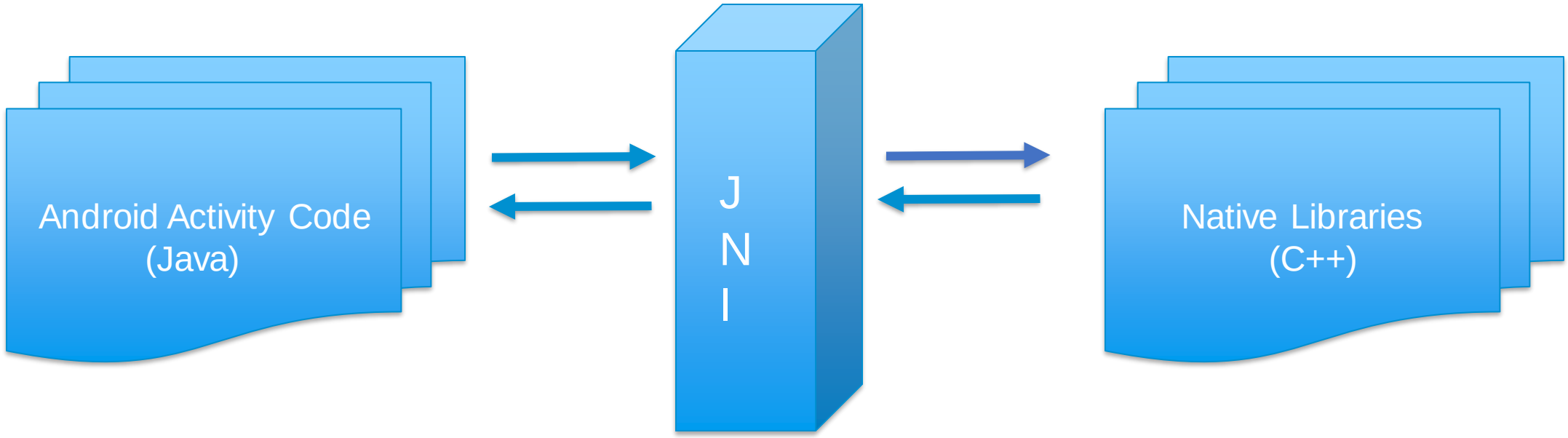
CPP Java Native Interface

```
#include <jni.h>
#include "Main.h"

JNIEXPORT jint JNICALL Java_Main_square(
    JNIEnv *env, jobject obj, jint i) {
  return i * i;
}
```

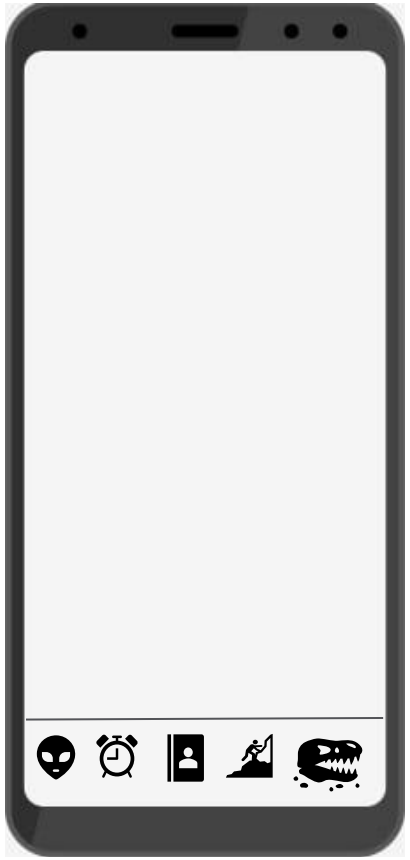
Java Native Interface – The Glue

TECH SOFT 3D
Dependable. Dynamic. Durable.

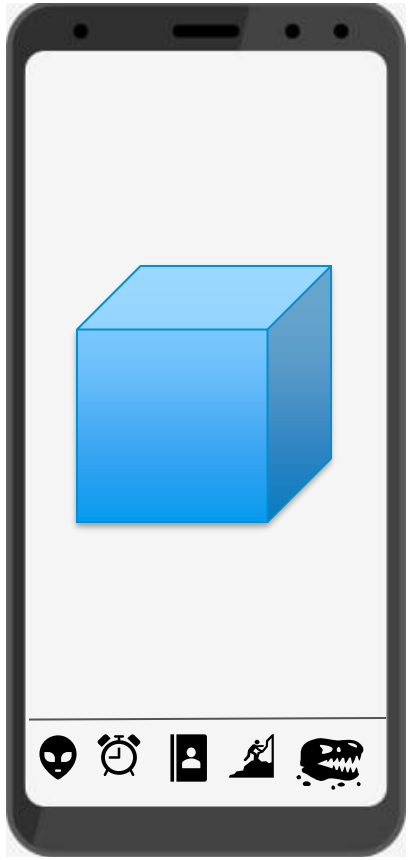


In Summary...

TECH SOFT 3D
Dependable. Dynamic. Durable.



+ JNI(HOOPS VISUALIZE) =



TECH SOFT 3D
Dependable. Dynamic. Durable.

Android Sample Project – Lets code!

Android Application Basics

TECH SOFT 3D
Dependable. Dynamic. Durable.

Activity

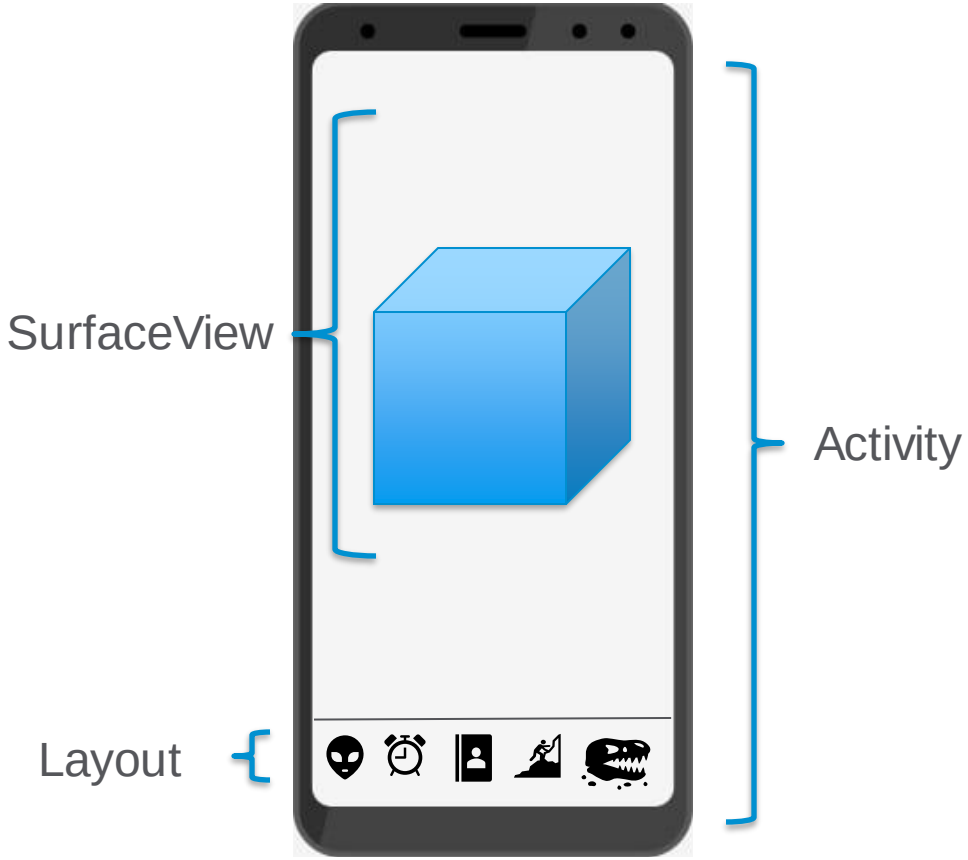
Implement App Logic with Java

Layout

Defines visual structure of User Interface (XML)

SurfaceView (View)

Dedicated drawing surface embedded within UI



.Java Code

TECH SOFT 3D
Dependable. Dynamic. Durable.

AndroidUserMobileSurfaceView.java

- Java Class containing Native method declarations.

- Programatically generated Native Interface, just instantiate it and execute logic!

MobileSurfaceActivity.java:

- Main entrypoint of Application.

- Instantiate Classes to access Native Interface methods.

This Photo by Unknown author is licensed under CC BY-SA.

.CPP Code

MobileSurface (cpp/h)

- Contains methods to setup and interact with Mobile Surface.
- OnTap, onPress, GestureDetection, etc.
- No need to modify contents.

MobileApp (cpp/h)

- Setup event handlers for Errors and Warnings inside of Visualize code.

- **UserMobileSurface.cpp & .h**

  Implements MobileSurface & MobileApp instances.

  Write ***YOUR*** Application logic here.

  Load Files, interact with geometry, etc.

Connect the Dots...

TECH SOFT 3D
Dependable. Dynamic. Durable.

- ***MobileSurfaceActivity.java:***

  Instantiate AndroidUserMobileSurfaceView

  Invoke ***MSurfaceView.onUserCode1();***

  native OnUserCode1() -> AndroidMobileSurfaceViewJNI.cpp (/cpp/JNI/<here>)

  Native Wrapper invokes ***UserMobileSurface.cpp*** methods.

- Wrapper Generation

  SIP.py is invoked as part of build step.

  Targets CPP Source Files – Generates JNI.CPP Wrappers based on specified keywords (***SURFACE/APP_ACTION)***

  Output is Native Libraries (.so) generated in ***../jniLibs***. These are bundled and shipped with APK.

OnUserCode1()

- Toggle shadows on/off
- Use existing function: ***OnModeSimpleShadow(Boolean)***
- Create global Boolean: ***shadowEnabled***

- Toggle boolean: ***shadowEnabled = !shadowEnabled;***
- Invoke function***: onModeSimpleShadow(shadowEnabled)***

*Function is ***Void*** so don't worry about returning anything.

OnUserCode2()

TECH SOFT 3D
Dependable. Dynamic. Durable.

- Cycle the current Render Mode
  *(stored in **currentRenderingMode**)*
- HPS::Rendering::Mode::<Mode>
  Supported modes include:
  - Flat
  - Default
  - Wireframe
  - HiddenLine
  - Gouraund
  - More in Docs..

```
// Tell Visualize to use the new mode.
GetCanvas().GetFrontView()
        .SetRenderingMode(newMode)

GetCanvas.Update()
```

OnUserCode3()

- Implement Cutting Plane Operator.

- Create Boolean for toggle (Global Var)
- If Cutting Disabled – enable and create Operator.
  - Add Operator to Controls.
- If Cutting Enabled – Disable and update View to remove it.

OnUserCode4()

- Your turn!

  Visit the HOOPS Visualize Technical Documentation for countless samples/examples.

Implement a Custom Operator to satisfy your Business Logic needs.

This Photo by Unknown author is licensed under CC BY.

HOOPS Exchange Deep Dive

TECH SOFT 3D
Dependable. Dynamic. Durable.

- Works with Mobiles Apps identically to HOOPS Visualize.
- Ship applications capable of accessing any widely supported CAD Format
- Enhance and empower 3D CAD Applications with CAD Data Query and Traversal

For more detailed examples of Exchange:

Previous Training Session:

https://www.youtube.com/watch?v=GWIWLSZQaX0

Tony.Tyrrell@techsoft3d.com

manage.techsoft3d.com