**Manifest.xml additions**:

To show application in VicoVR list:

<category android:name="com.nuitrack.intent.category.VICOVR" />

Features and permissions:

<uses-feature android:name="android.hardware.sensor.accelerometer" android:required="true"/>

<uses-feature android:name="android.hardware.sensor.gyroscope" android:required="true"/>

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.BLUETOOTH" />

<uses-permission android:name="android.permission.BLUETOOTH\_ADMIN" />

**Prefabs description:**

**NuitrackScripts (preferably put it in some kind of init scene)**:

*NuitrackManager*:

Creates nuitrack modules and updates data from them. Has events for each update:

public static event nuitrack.DepthSensor.OnUpdate onDepthUpdate;

public static event nuitrack.UserTracker.OnUpdate onUserTrackerUpdate;

public static event nuitrack.SkeletonTracker.OnSkeletonUpdate onSkeletonTrackerUpdate;

public static event nuitrack.HandTracker.OnUpdate onHandsTrackerUpdate;

*CurrentUserTracker*:

Tracks ID and skeleton data of current user, logic may be modified to track several users and other modules data (depending on application needs).

*IMURotation*:

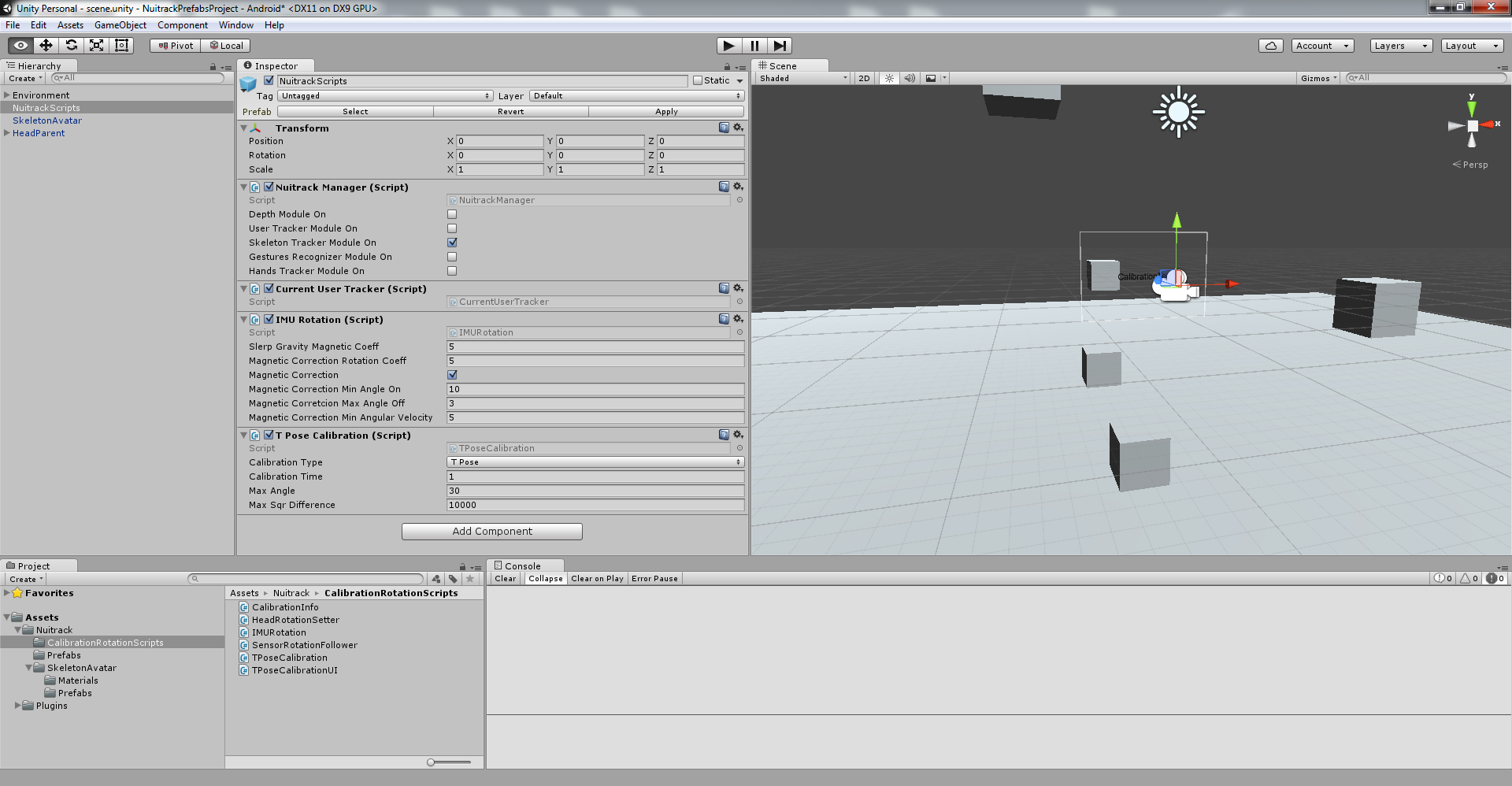
Calculates orientation of device using IMU data (may be removed if used with other plugins that return orientation of device (GoogleVR, GearVR, etc.)

*TPoseCalibration*:

Example calibration script (player should stay in T-pose for some time to calibrate). Allows to get sensor pitch angle (assuming that player stands vertically and vector from neck to torso joints is alligned with gravity) and recenter cameras.

*CalibrationInfo*:

Saves calibration data (sensor pitch angle). May be modified to save other data (sensor - floor offset for example, if application assumes that sensor should be able to see user’s feet).



**HeadParent (should be replaced with other head prefabs If used with VR plugins (GoogleVR, GearVr, etc.))**:

*HeadRotationSetter*:

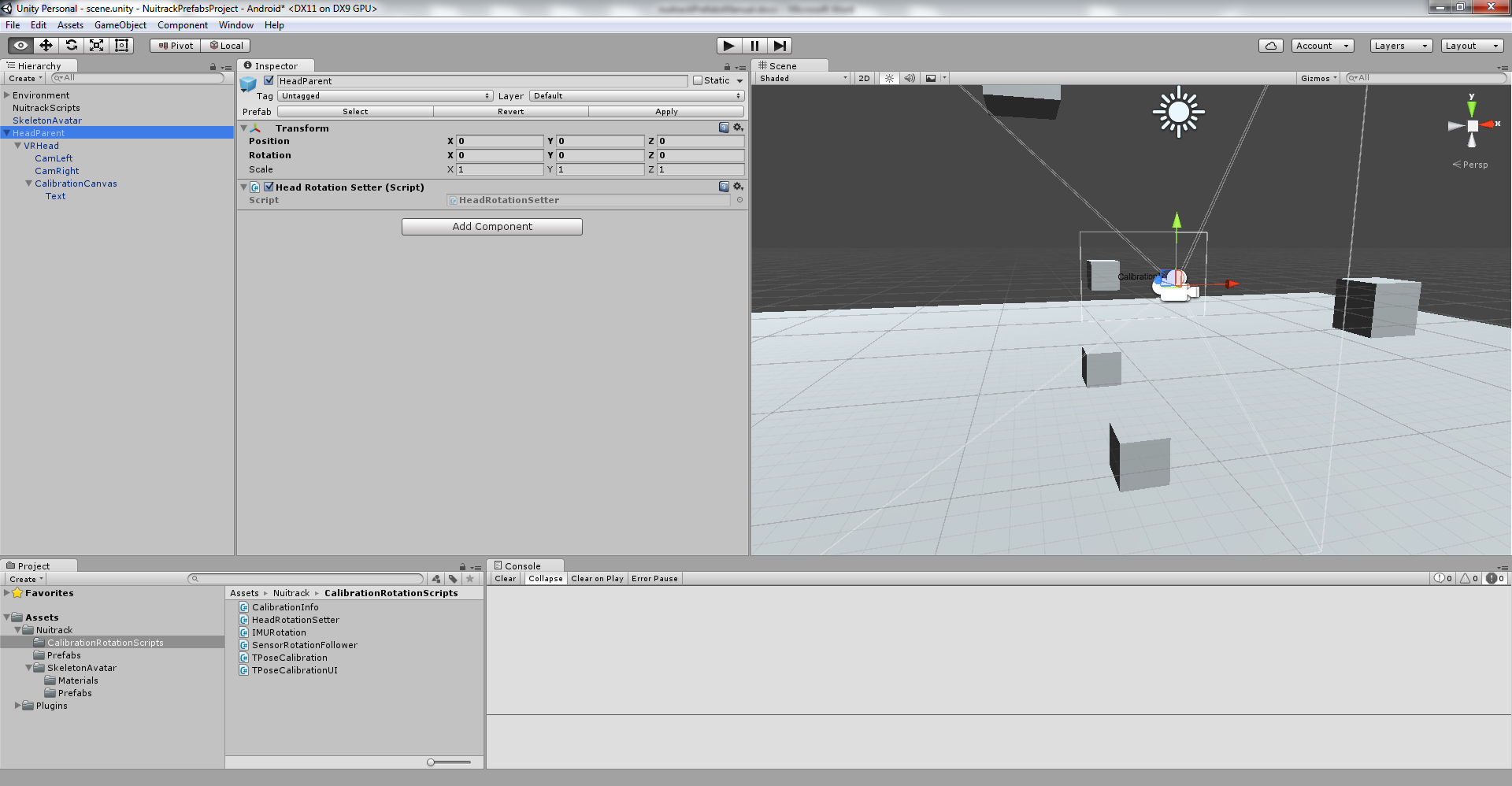
Resets IMURotation.rotation on calibration success event.

*IMURotationFollower*:

Updates transform’s local rotation every frame with IMURotation.Rotation value (rotates head).

*TPoseCalibrationUI*:

Example of visual feedback for calibration process.



**SkeletonAvatar**:

*SkeletonAvatar*:

Visualization of skelton data for current user.

