Building ROS 2 enabled Android apps with C++

Shane Loretz, ROSCon 2022

Android is a trademark of Google LLC.
Overview

- Why Android and ROS?
- Comparison with existing projects
- How to build Apps with only C++?
- What are the downsides?
Why Android + ROS?
Because Android is useful to Robots

Spot Tablet
Boston Dynamics

Pepper
SoftBank Robotics

Astrobee
NASA

Spot image: Web Summit, CC BY 2.0 <https://creativecommons.org/licenses/by/2.0>, via Wikimedia Commons (edited)
Pepper Image: Softbank Robotics Europe, CC BY-SA 4.0 <https://creativecommons.org/licenses/by-sa/4.0>, via Wikimedia Commons (edited)
Because Android is useful to Robots

Android is used by

- Astrobee to enable Guest Science apps
- Pepper to show relevant information to users
- Spot Tablet to control Spot
Comparison with existing projects
What already exists?

## ROS projects usable on Android

<table>
<thead>
<tr>
<th></th>
<th>Purpose</th>
<th>ROS 1 or ROS 2?</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosjava</td>
<td>Client Library</td>
<td>1</td>
</tr>
<tr>
<td>ROS-Mobile-Android</td>
<td>App</td>
<td>1</td>
</tr>
<tr>
<td>ros2_java (aka rcljava)</td>
<td>Client Library</td>
<td>2</td>
</tr>
</tbody>
</table>
What already exists?

## ROS projects usable on Android

<table>
<thead>
<tr>
<th></th>
<th>Purpose</th>
<th>ROS 1 or ROS 2?</th>
<th>Cross-compiles ROS packages?</th>
</tr>
</thead>
<tbody>
<tr>
<td>rosjava</td>
<td>Client Library</td>
<td>1</td>
<td>No</td>
</tr>
<tr>
<td>ROS-Mobile-Android</td>
<td>App</td>
<td>1</td>
<td>No (uses rosjava)</td>
</tr>
<tr>
<td>ros2_java (aka rcljava)</td>
<td>Client Library</td>
<td>2</td>
<td>Yes (up to rcl)</td>
</tr>
<tr>
<td>Sensors for ROS</td>
<td>App</td>
<td>2</td>
<td>Everything (No Java or Kotlin)</td>
</tr>
</tbody>
</table>
What is Sensors for ROS?

It's an Android app that publishes sensor data to ROS 2 topics.
What is Sensors for ROS?

Supported sensors:
- Accelerometer
- Barometer
- Camera(s)
- Gyroscope
- Illuminance
- Magnetometer
What is Sensors for ROS?

- It uses CMake, C++, and rclcpp
- No Java* or Kotlin!

```
$ mkdir build
$ cd build
$ cmake .. -DANDROID_HOME=/path/to/android-sdk/
  ...
$ make -j`nproc`
  ...
```

* We won’t write any Java, but we still have to interact with the JVM.
How to build Apps with only C++?
“The Android NDK is a toolset that lets you implement parts of your app in native code, using languages such as C and C++. For certain types of apps, this can help you reuse code libraries written in those languages.”
What is an Android App?

- It's an .apk
- Zip archive containing:
  - Compiled code
  - Resources
    - App Icon
    - Strings
  - A manifest
Steps to make an App

1. Create a manifest
2. Cross-compile code
3. Convert resources
4. Package it into an .apk
1. Creating the manifest

Create the manifest

- Hand written
- Describes what's in the app
  - Required permissions
  - Activity (ies) in the app
2. Cross-compiling everything

Cross compile everything from source

- Anything that runs on the device
  - ROS 2 Humble
    - rcl_logging_android
  - App specific code
- Not cross-compiling build tool dependencies
2. Cross-compiling everything

1. Get ROS source code with `rosinstall_generator` and `vcstool`
2. Use CMake toolchain provided by NDK
2a. Making a native Activity

An App needs an Activity

● Entry point for an Android App
● Apps have a “main” activity
● An Activity is a Java class
2a. Making a native Activity

ANativeActivity

- C API provided by Android NDK
- Using it requires:
  - Creating shared library
  - Implementing an “on create” function
  - Implementing Activity callbacks
3. Converting resources

Resources

- The App icon
- Localized strings
- Use **aapt2** to convert to Android format
4. Packaging an App

Packaging the .apk

- Create folder structure
- Create a zip archive
- Use `zipalign` on it
- Sign it with `apksigner`
4. Packaging an App

Tools needed to make APKs

- aapt2
- zipalign
- apksigner
What are the downsides?
What are the downsides?

ANativeActivity requires using OpenGL

- Can’t use native GUI widgets
- Can’t use Android XML UI layouts
- Sensors for ROS uses Dear ImGui
What are the downsides?

Some APIs aren’t in the NDK

- Requesting CAMERA permissions
  - Workaround: Call Java APIs using Java Native interface (JNI)
Building ROS 2 enabled Android apps with C++

https://github.com/sloretz/sensors_for_ros
https://github.com/sloretz/rcl_logging_android