Performance Testing

Upon completion of this module, a student will be able to

- work with the high level Android Profiler
- inspect the Java heap and memory allocations with Memory Profiler
- inspect network traffic with Network Profiler
- inspect energy usage with Energy Profiler
- inspect CPU activity and traces with CPU Profiler
- inspect the user interface using layout inspector

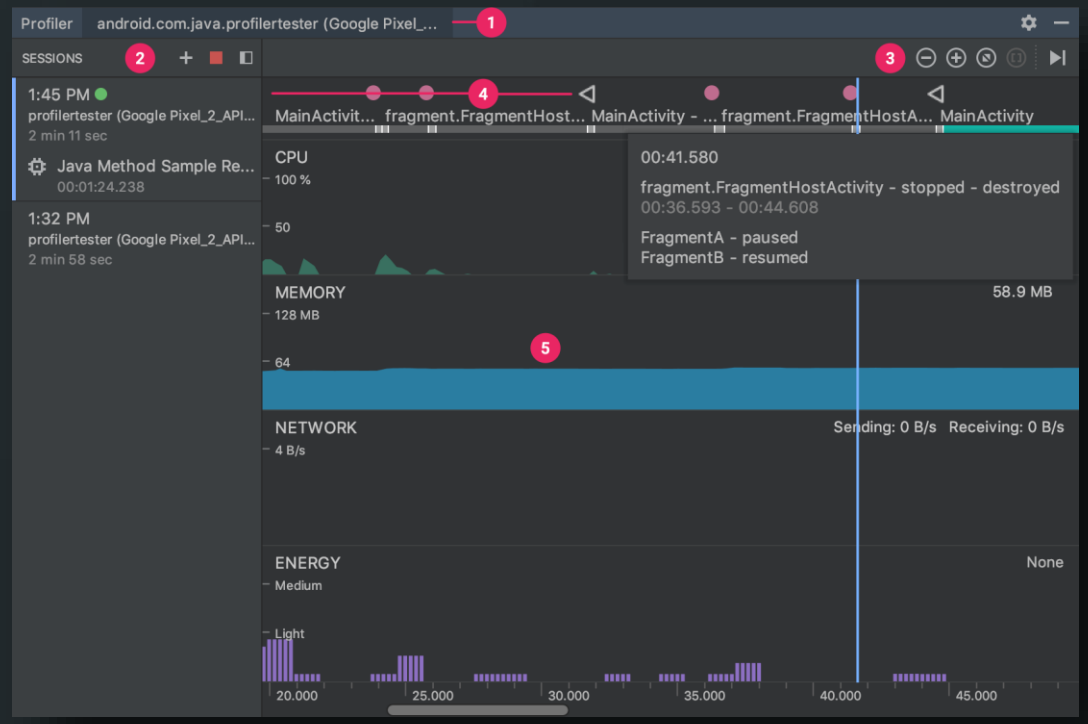
Assignment

- Task
  - Enhance the performance of your custom volume control app.
- Repo
  - https://github.com/LambdaSchool/Android_Performance_CustomView
- Submission
  - Fork on github and submit pull request

A Student Can

work with the high level Android Profiler

Android Profiler



1. Android Profiler shows the process and device currently being profiled.

2. In the **Sessions** pane, choose which session to view, or start a new profiling session.

3. Use the zoom buttons to control how much of the timeline to view, or use the **Attach to live** button to jump to the real-time updates.

4. The event timeline shows events related to user input, including keyboard activity, volume control changes, and screen rotations.

5. The shared timeline view, which includes graphs for CPU, memory, network, and energy usage.

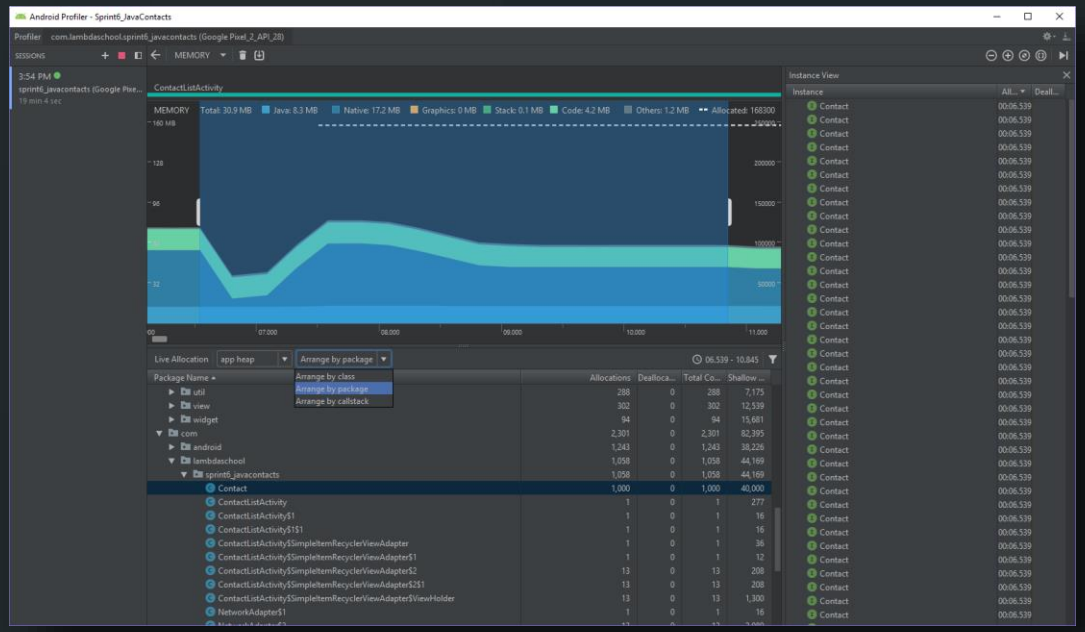
A Student Can

inspect the Java heap and memory allocations with Memory Profiler

Memory Categories

- Java
- Native
- Graphics
- Stack
- Code
- Others
- Allocated

Memory Profiler



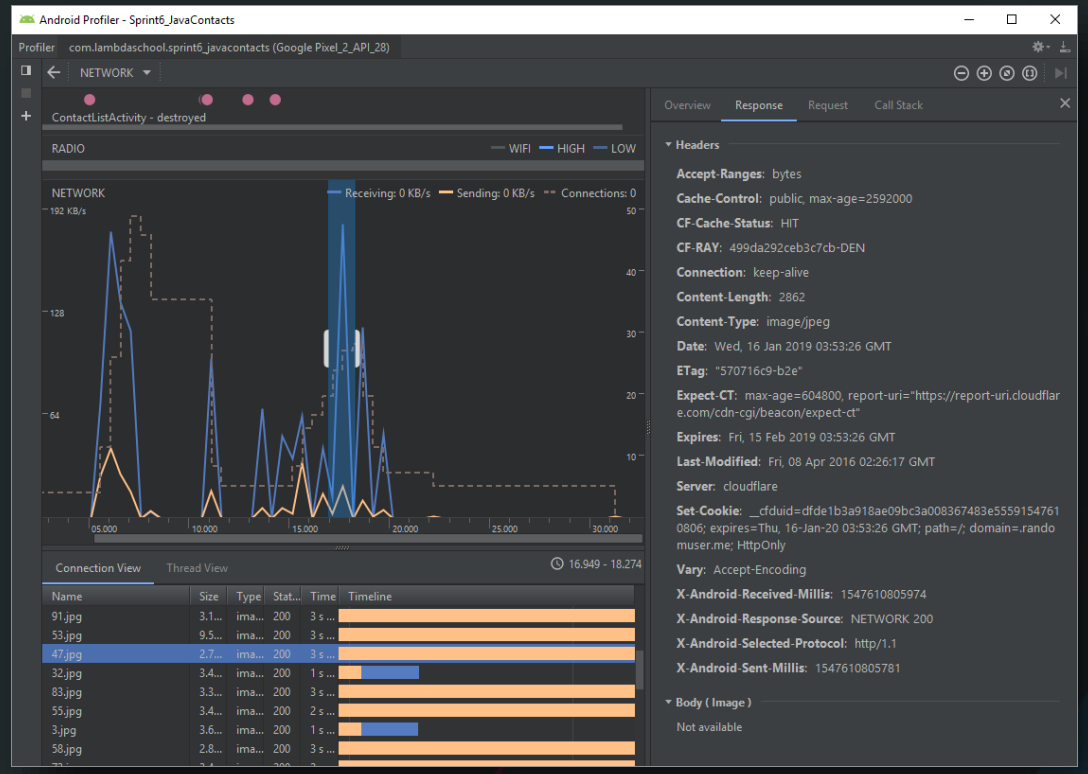
- Observe memory allocations in real time
- Can click a type to see all the objects allocated for that type
- Click the object to see more information about it

A Student Can

inspect network traffic with Network Profiler

Network Profiler

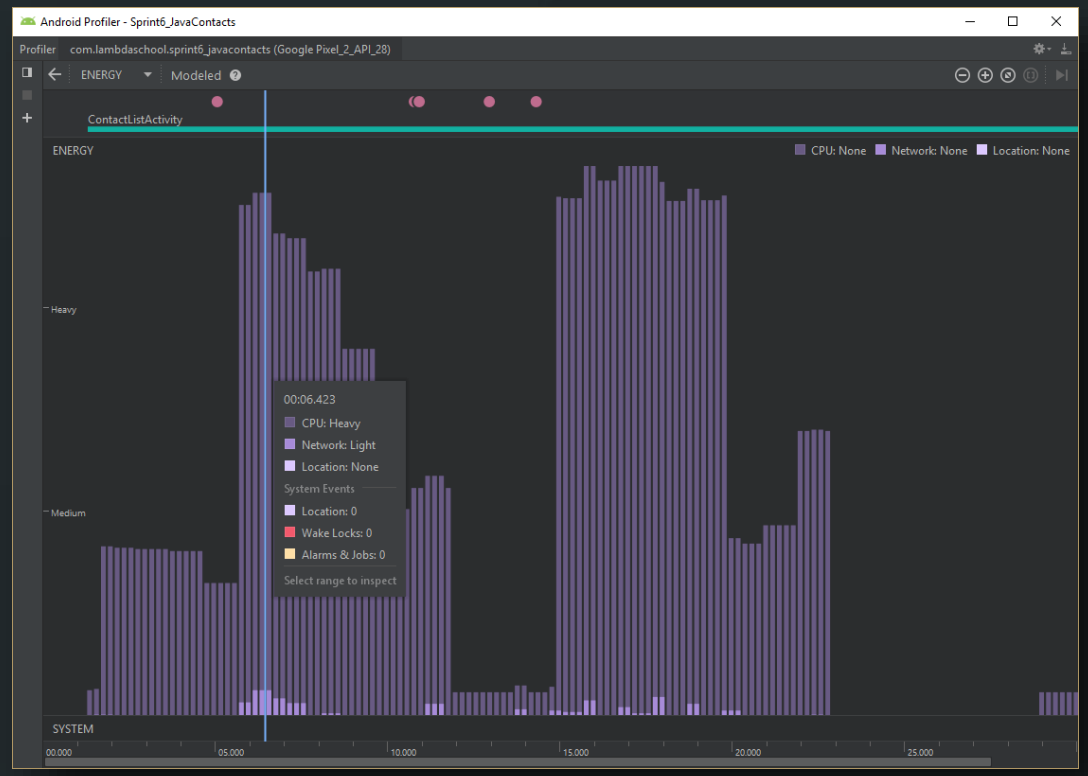
- Network transactions in real time
- See each transaction in time slice
- Select for details



A Student Can

inspect energy usage with Energy Profiler

Energy Profile



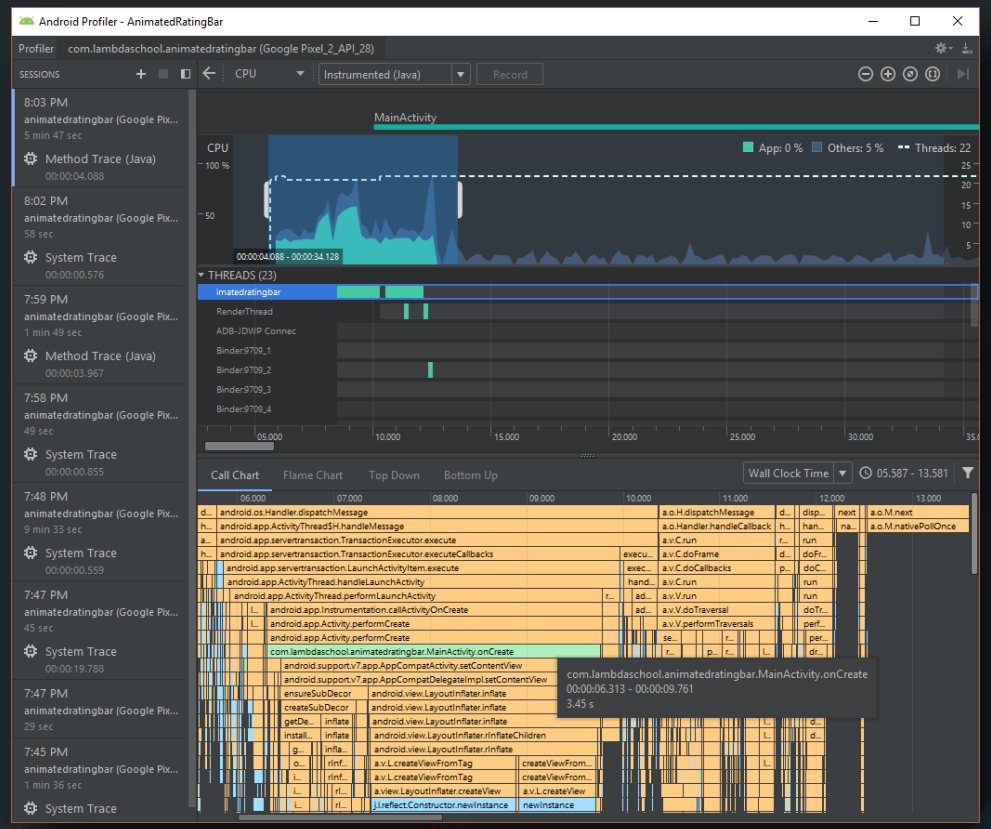
- Estimate energy consumption on device
- Quantifies multiple categories

A Student Can

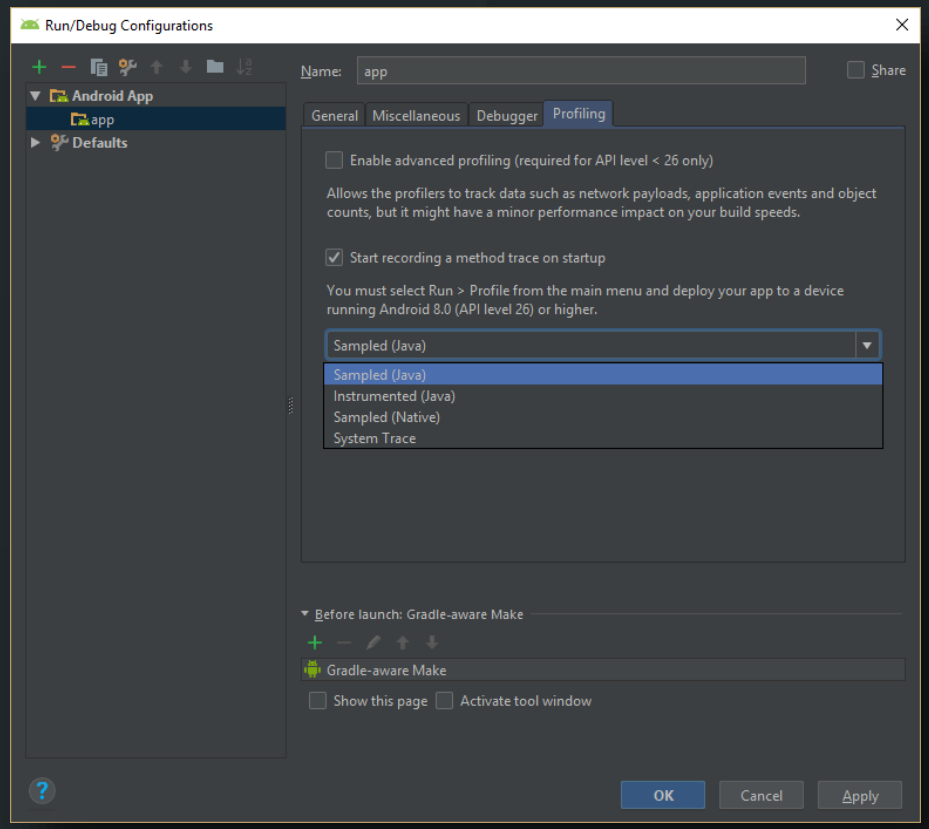
inspect CPU activity and traces with CPU Profiler

CPU Profiler

- Must record time period
- Detailed information on call stack during any point
- Your methods are in green
- Can see if things are taking longer than they should



CPU Profile Recording

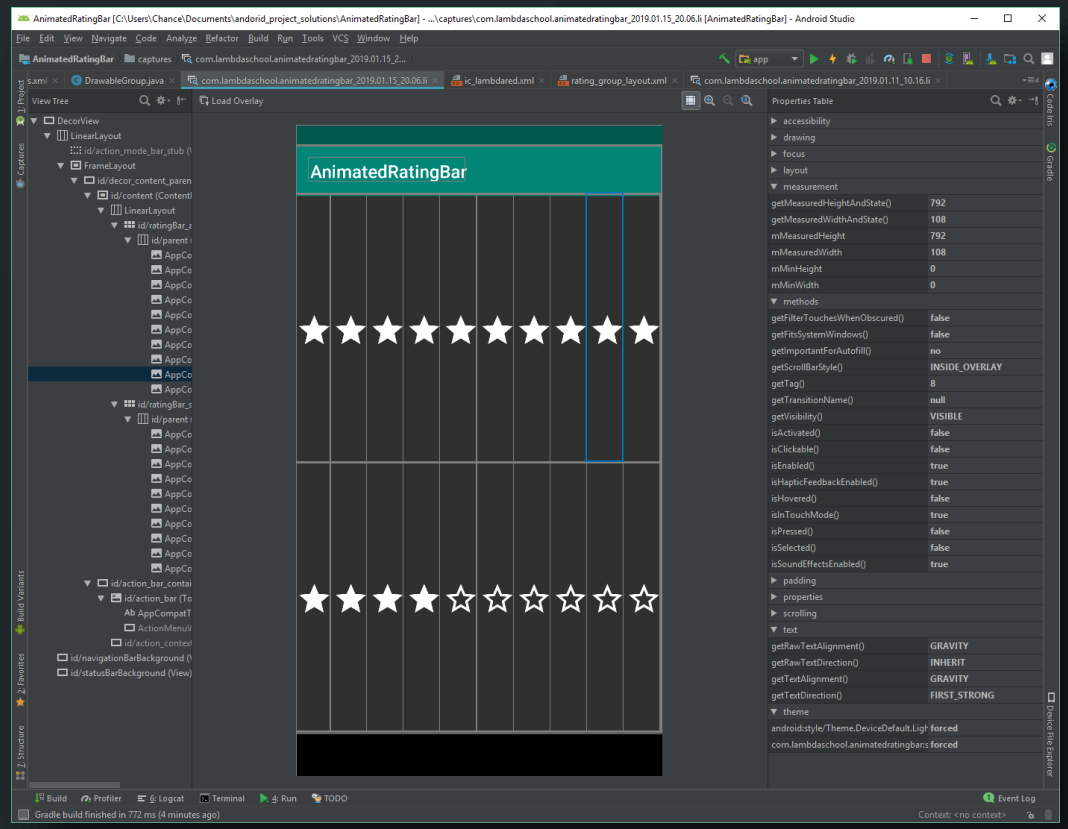


- Sampled (lightweight, coarse view)
- Instrumented (heavyweight, detailed view)
- Native Sampled (C++ testing)
- System Trace (Mark code for instrumented testing)

A Student Can

inspect the user interface using layout inspector

Layout Inspector



- View current layout as visible in test device
- Select components for more detailed view