More Android
How hard was week 9 code review assignment?

A) Easy
B) Moderate
C) Challenging
D) Unreasonable
How long did week 9 assignment take?

A) Less than 2 hours
B) 2 to 4 hours
C) 4 to 6 hours
D) 6 to 8 hours
E) More than 8 hours
Unix

- wc -- “word count”
- grep – “search files for particular strings”
- find – “identify files with matching names”
Testing

- Two kinds of tests for Android Projects:
  - Normal non-UI tests (‘test’ directory)
    - Just use Junit as normal
  - User Interface tests (‘androidTest’ directory)
    - Use Espresso
Which is not an Android logging level

A) ERROR
B) DEBUG
C) WTF
D) VERBOSE
E) All of the above are valid
Logging

- Dump messages to the log; see with Android Monitor

- Log messages have:
  - Priority (ERROR, WARN, INFO, DEBUG, VERBOSE)
  - Tag
  - Message

- Usage: (use logd shortcut)

```java
private final static String TAG = ClassName.class.getSimpleName();
Log.d(TAG, "functionName: your message here");
```
What is going to happen?

A) Success
B) Fail – Networking on main thread exception
C) Fail – Didn’t initialize networking library
D) Fail – Don’t have permission to access network
E) Fail – Can’t translate URL to IP address
Android Permission Model

- Users don’t want apps to violate their privacy
- Users grant apps permission to do things
  - Access the network, camera, calendar, phone book, etc.
  - Historically, these have been granted at install time
  - All or nothing model

- Starting in Marshmallow, incremental permission model
  - Request “mandatory” permissions at install time
  - Request other permissions as needed (for clarity)
  - Allow users to revoke permissions
    - App must check permissions before doing controlled things.
Threads

- When you write code, you tell the machine what to do
  - One thing at a time.
- Hard/bad to interleave multiple things
  - E.g., a user interface with long latency tasks
Threads, cont.

- Computer programs are made up of threads
- Each thread:
  - Performs a series of tasks
  - In the order specified by the code
  - One at a time
- Hard/bad to interleave multiple things on a single thread
  - E.g., a user interface with long latency tasks
- Solution: use multiple threads; dedicate a thread to the UI
  - So it is always responsive
  - Do slow stuff on other threads
  - Have to handle communication/synchronization between threads