CS 891: Scalable Microservices: Overview (Part 2)

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

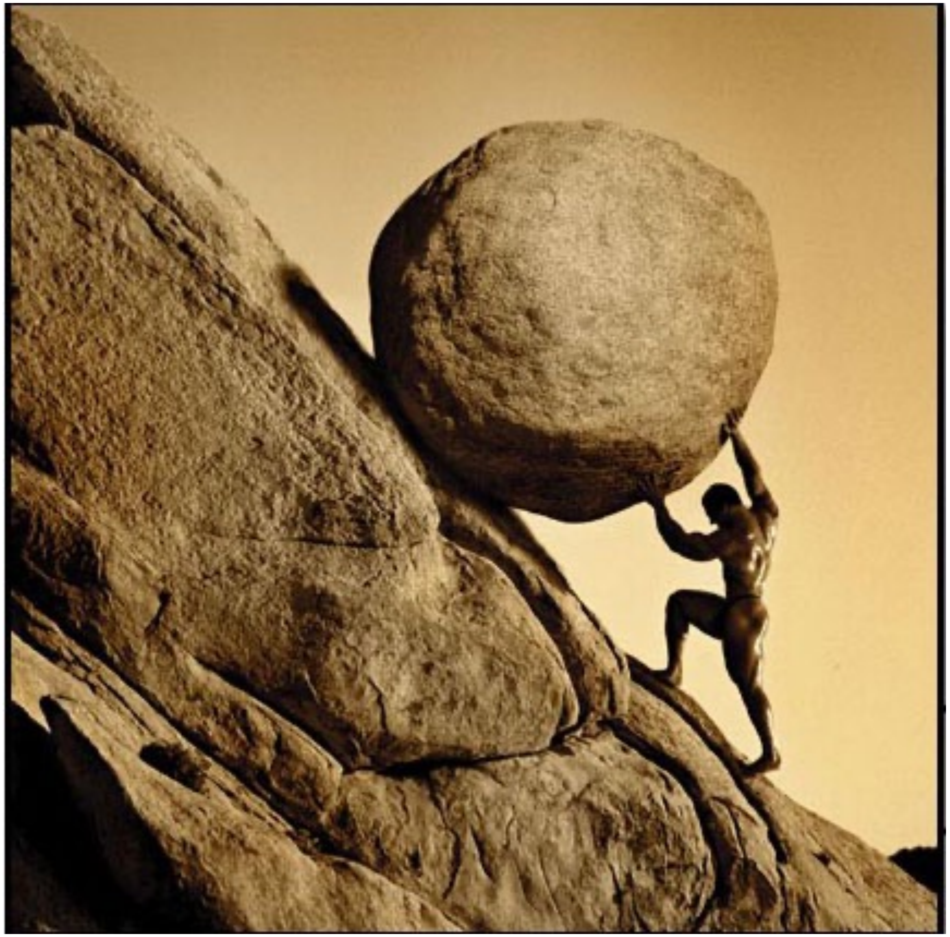
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Lesson

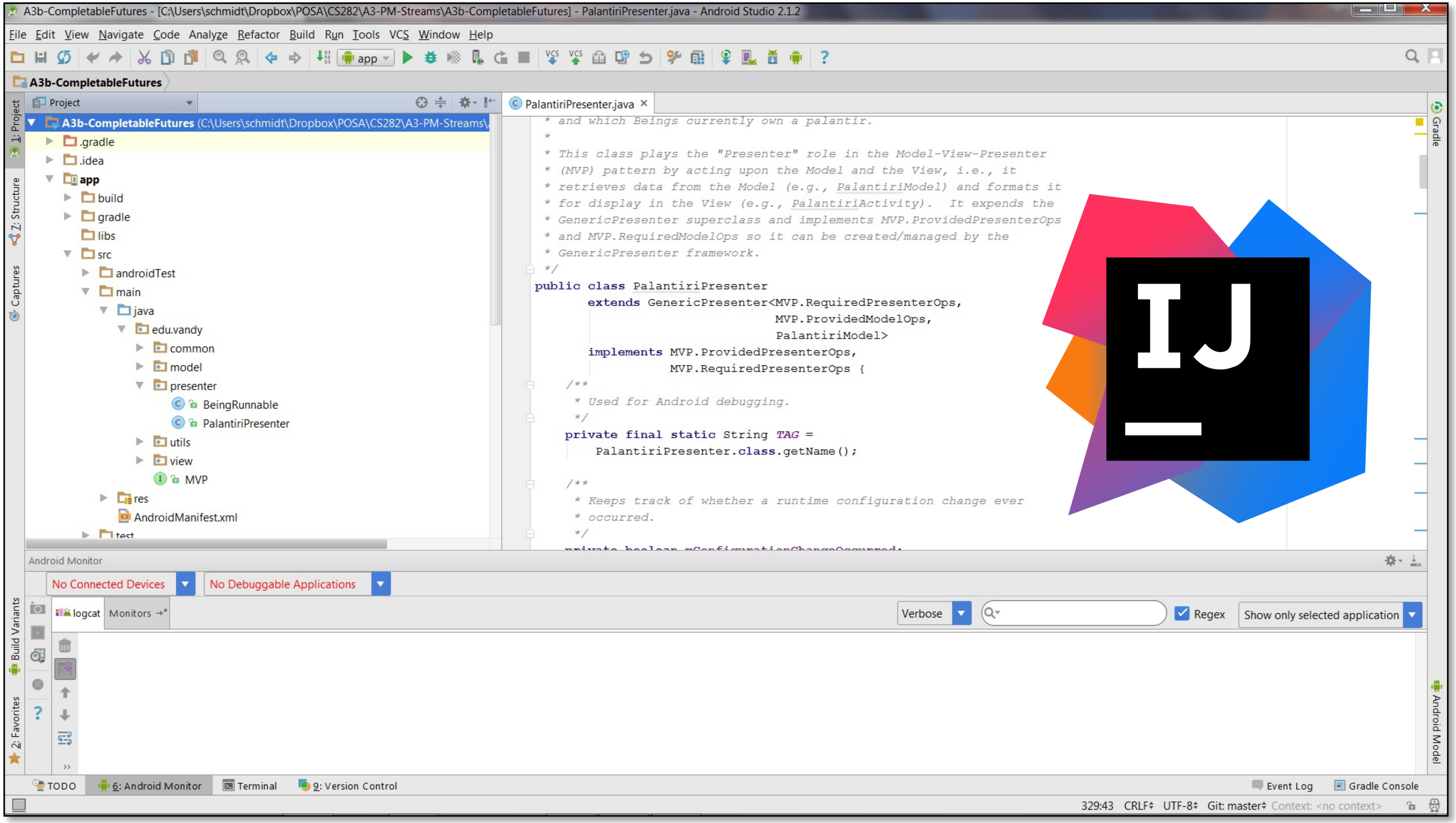
- Understand the course topics & logistics
  - Course philosophy
  - Course contents
  - Structure of the lecture material
  - Overview of the assignments & assessments



Overview of Assignments & Assessments

Overview of Assignments & Assessments

- Programming assignments are written in modern Java using IntelliJ



```java
 * and which Beings currently own a palantir.
 *
 * This class plays the "Presenter" role in the Model-View-Presenter
 * (MVP) pattern by acting upon the Model and the View, i.e., it
 * retrieves data from the Model (e.g., PalantiriModel) and formats it
 * for display in the View (e.g., PalantiriActivity).  It expends the
 * GenericPresenter superclass and implements MVP.ProvidedPresenterOps
 * and MVP.RequiredModelOps so it can be created/managed by the
 * GenericPresenter framework.
 */
public class PalantiriPresenter
       extends GenericPresenter<MVP.RequiredPresenterOps,
                                MVP.ProvidedModelOps,
                                PalantiriModel>
       implements MVP.ProvidedPresenterOps,
                  MVP.RequiredPresenterOps {
    /**
     * Used for Android debugging.
     */
    private final static String TAG =
        PalantiriPresenter.class.getName();

    /**
     * Keeps track of whether a runtime configuration change ever
     * occurred.
     */
    private boolean mConfigurationChangeOccurred;
```

You can use any IDE, but your final submission *must* build & run with the latest IntelliJ & Java 19

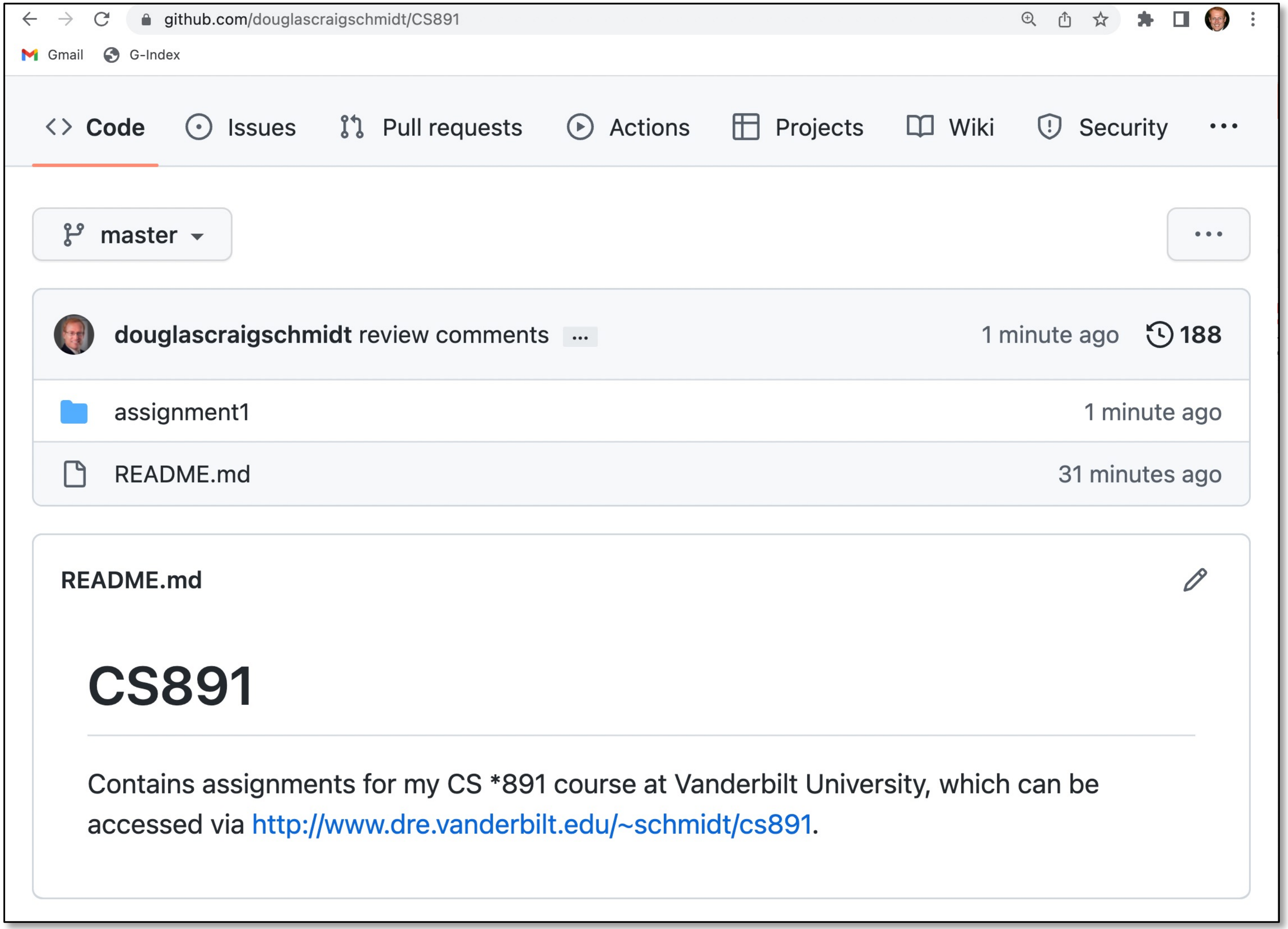
Overview of Assignments & Assessments

- Programming assignments are written in modern Java using IntelliJ
  - The Java 19 runtime environment (JRE) can be downloaded via IntelliJ

See github.com/douglascraigschmidt/CS891/wiki/Installing-Software

Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub



douglascraigschmidt review comments ... 1 minute ago 188

assignment1 1 minute ago

README.md 31 minutes ago

README.md

CS891

Contains assignments for my CS *891 course at Vanderbilt University, which can be accessed via http://www.dre.vanderbilt.edu/~schmidt/cs891.

Go to GitHub at github.com/douglascraigschmidt/CS891

Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub
  - You will need to learn how to use GitLab et al.

Open source software to collaborate on code

GitLab offers git repository management, code reviews, issue tracking, activity feeds and wikis. Enterprises install GitLab on-premise and connect it with LDAP and Active Directory servers for secure authentication and authorization. A single GitLab server can handle more than 25,000 users but it is also possible to create a high availability setup with multiple active servers.

Do you want more from your GitLab installation? A subscription bundles the Enterprise Edition with support from the GitLab team. The Enterprise Edition allows you to sync LDAP groups, control pushes via git hooks, integrate better with Jenkins and Jira, and to run MySQL and forward logs when using our Omnibus package. Our service engineers will help you keep your server running smoothly.

We'll discuss how to setup GitLab shortly

Overview of Assignments & Assessments

- All source code for assignments & exam
  - You will need to learn how to use GitLab et al.
  - Be prepared to update your repositories occasionally

"If you don't like change, you're going to like irrelevance even less."

Overview of Assignments & Assessments

- Assignments will provide a range of experience with modern Java concurrent & parallel microservices

Go to GitHub at github.com/douglascraigschmidt/CS891

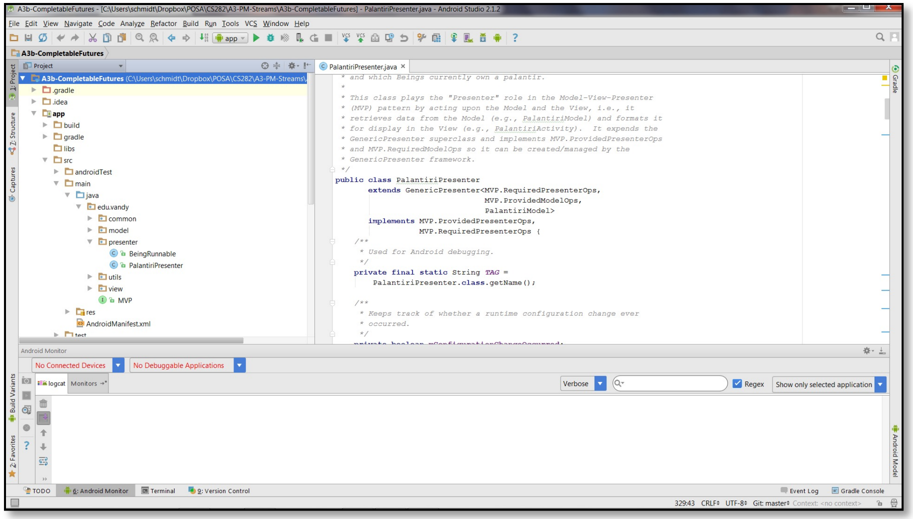
Overview of Assignments & Assessments

- Assignments will provide a range of experience with modern Java concurrent & parallel microservices
  - Implement a microservice-based movie recommendation system on Spring using modern Java features, e.g.
    - Java lambda expressions, method references, & functional interfaces
    - Java sequential streams
    - Java structured concurrency
    - Java reactive streams
    - Spring WebMVC & WebFlux

The topics covered by the assignments may change during the semester

Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff



A3b-CompletableFutures - [C:\Users\schmidt\Dropbox\POSA\CS282\A3-PM-Streams\A3b-CompletableFutures] - PalantiriPresenter.java - Android Studio 2.1.2
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
app
A3b-CompletableFutures
Project
A3b-CompletableFutures (C:\Users\schmidt\Dropbox\POSA\CS282\A3-PM-Streams\
.gradle
.idea
app
build
gradle
libs
src
androidTest
main
java
edu.vandy
common
model
presenter
BeingRunnable
PalantiriPresenter
utils
view
MVP
res
AndroidManifest.xml
test
PalantiriPresenter.java
Android Monitor
No Connected Devices
No Debuggable Applications
logcat
Monitors
Verbose
Regex
Show only selected application
1: Project
7: Structure
Captures
Build Variants
2: Favorites
Gradle
Android Model
TODO
6: Android Monitor
Terminal
9: Version Control
Event Log
Gradle Console
329:43 CRLF UTF-8 Git: master Context: <no context>

```java
 * and which Beings currently own a palantir.
 *
 * This class plays the "Presenter" role in the Model-View-Presenter
 * (MVP) pattern by acting upon the Model and the View, i.e., it
 * retrieves data from the Model (e.g., PalantiriModel) and formats it
 * for display in the View (e.g., PalantiriActivity).  It expends the
 * GenericPresenter superclass and implements MVP.ProvidedPresenterOps
 * and MVP.RequiredModelOps so it can be created/managed by the
 * GenericPresenter framework.
 */
public class PalantiriPresenter
       extends GenericPresenter<MVP.RequiredPresenterOps,
                                MVP.ProvidedModelOps,
                                PalantiriModel>
       implements MVP.ProvidedPresenterOps,
                  MVP.RequiredPresenterOps {
    /**
     * Used for Android debugging.
     */
    private final static String TAG =
        PalantiriPresenter.class.getName();

    /**
     * Keeps track of whether a runtime configuration change ever
     * occurred.
     */
    private boolean mConfigurationChangeOccurred;
```

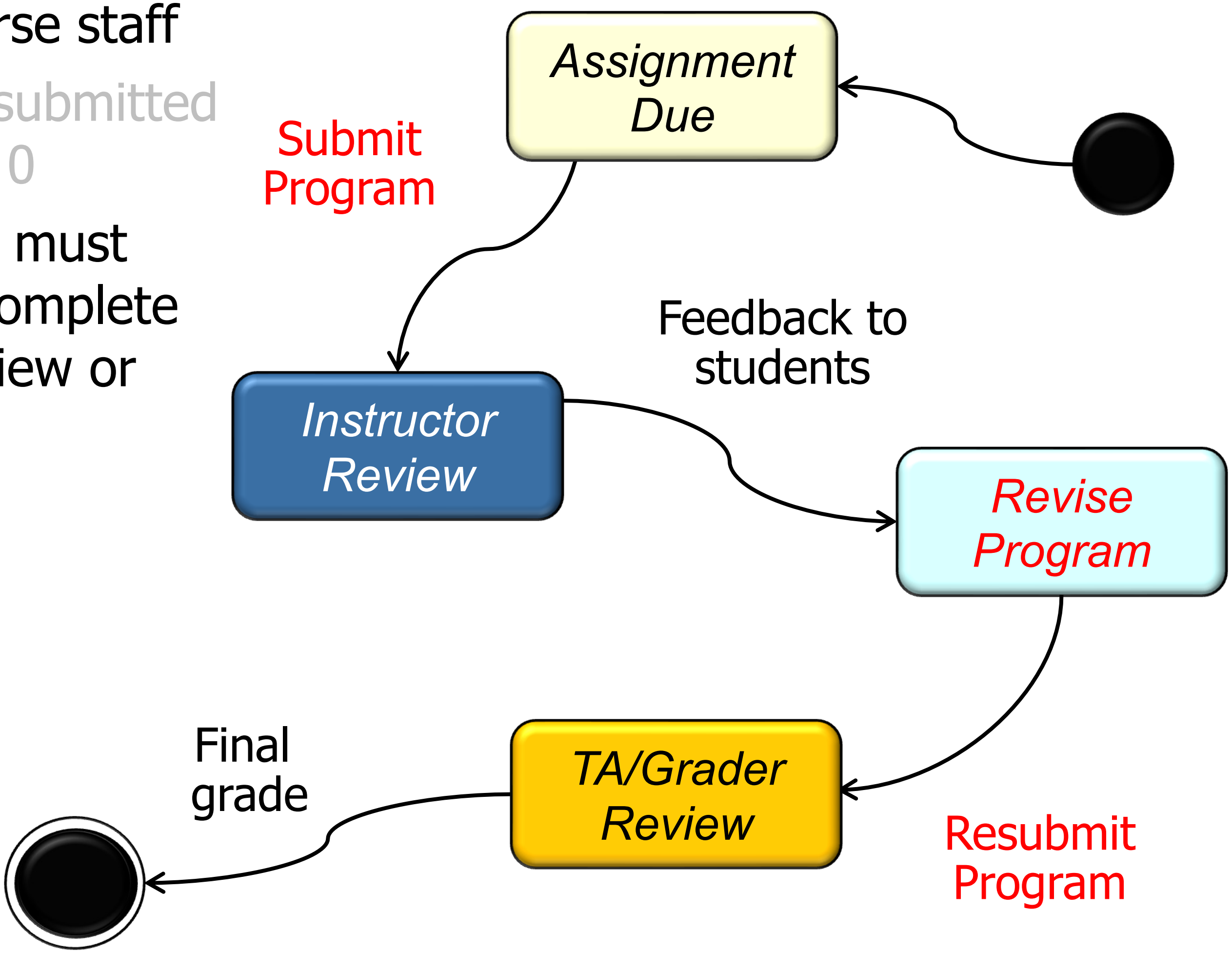
Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff
  - Assignments *must* be submitted on time or you'll get a 0

See github.com/douglascraigschmidt/CS891/wiki/CS-891-FAQ

Overview of Assignments & Assessments

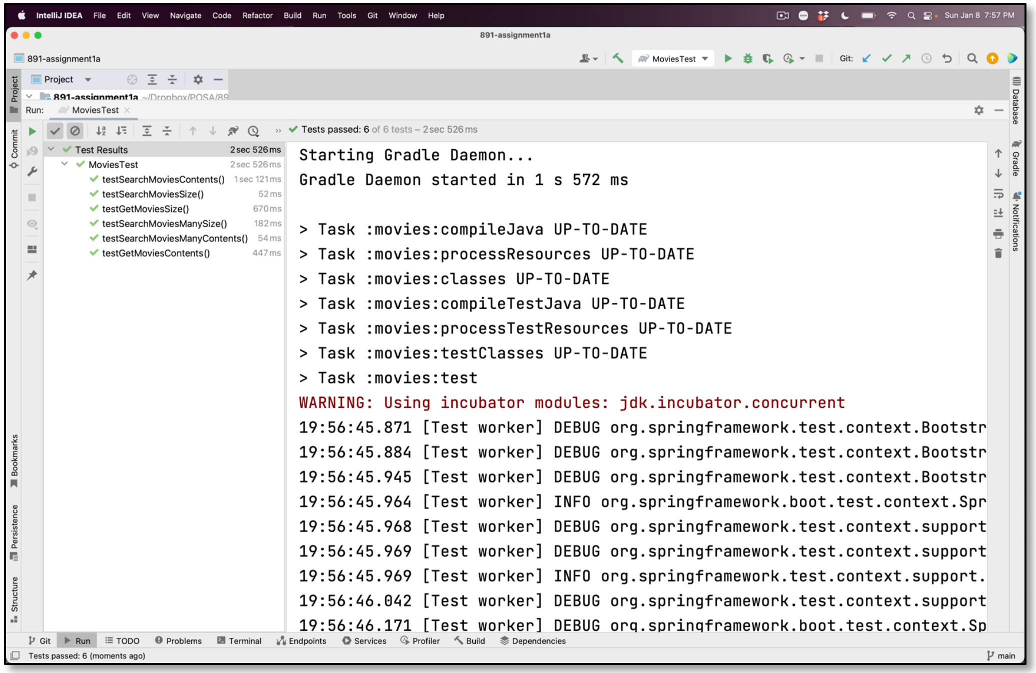
- Assignment assessments will be done via reviews by course staff
  - Assignments *must* be submitted on time or you'll get a 0
  - Your initial submission must compile & be largely complete or you won't get a review or a final grade



See www.dre.vanderbilt.edu/~schmidt/cs891/assignments.html

Overview of Assignments & Assessments

- Assignment assessments will be done via reviews by course staff
  - Assignments *must* be submitted on time or you'll get a 0
  - Your initial submission must compile & be largely complete or you won't get a review or a final grade
    - You *must* also run the regression tests & push a screenshot of the results to GitLab



✓ Tests passed: 6 of 6 tests – 2 sec 526 ms

✓ Test Results 2 sec 526 ms
✓ MoviesTest 2 sec 526 ms
✓ testSearchMoviesContents() 1 sec 121 ms
✓ testSearchMoviesSize() 52 ms
✓ testGetMoviesSize() 670 ms
✓ testSearchMoviesManySize() 182 ms
✓ testSearchMoviesManyContents() 54 ms
✓ testGetMoviesContents() 447 ms

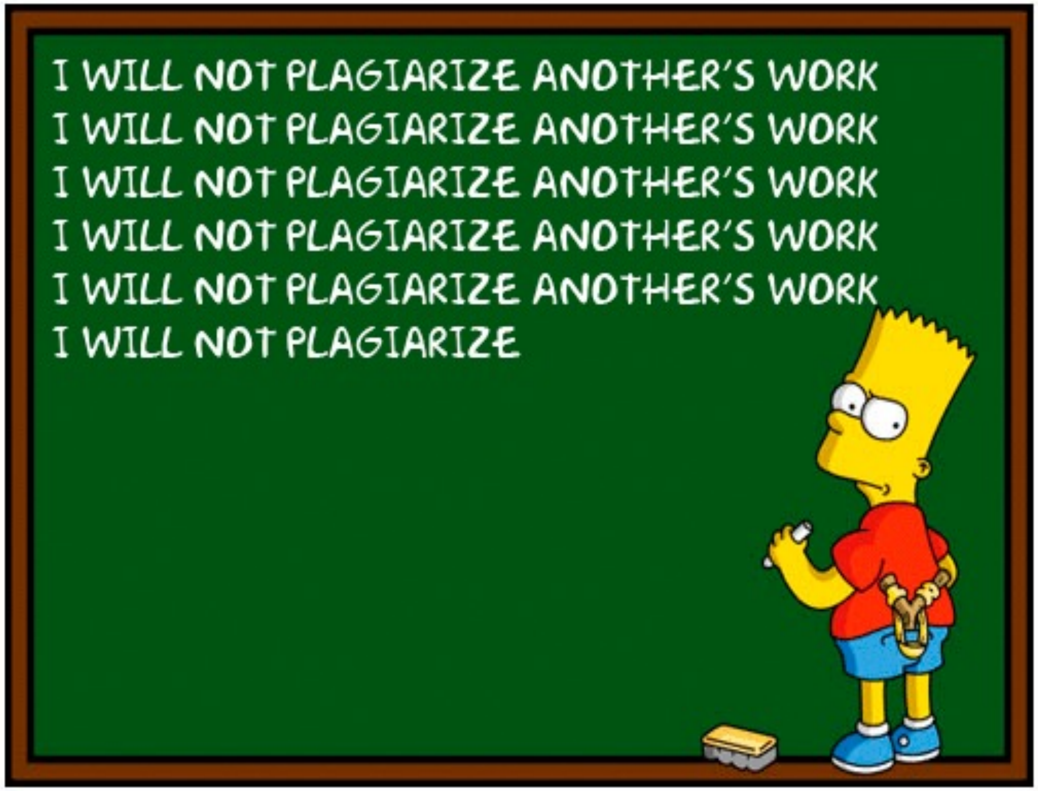
```
Starting Gradle Daemon...
Gradle Daemon started in 1 s 572 ms

> Task :movies:compileJava UP-TO-DATE
> Task :movies:processResources UP-TO-DATE
> Task :movies:classes UP-TO-DATE
> Task :movies:compileTestJava UP-TO-DATE
> Task :movies:processTestResources UP-TO-DATE
> Task :movies:testClasses UP-TO-DATE
> Task :movies:test
WARNING: Using incubator modules: jdk.incubator.concurrent
19:56:45.871 [Test worker] DEBUG org.springframework.test.context.Bootstr
19:56:45.884 [Test worker] DEBUG org.springframework.test.context.Bootstr
19:56:45.945 [Test worker] DEBUG org.springframework.test.context.Bootstr
19:56:45.964 [Test worker] INFO org.springframework.boot.test.context.Spr
19:56:45.968 [Test worker] DEBUG org.springframework.test.context.support
19:56:45.969 [Test worker] DEBUG org.springframework.test.context.support
19:56:45.969 [Test worker] INFO org.springframework.test.context.support.
19:56:46.042 [Test worker] DEBUG org.springframework.test.context.support
19:56:46.171 [Test worker] DEBUG org.springframework.boot.test.context.Sp
```

Tests passed: 6 (moments ago)

See www.dre.vanderbilt.edu/~schmidt/cs891/assignments.html

Overview of Assignments & Assessments

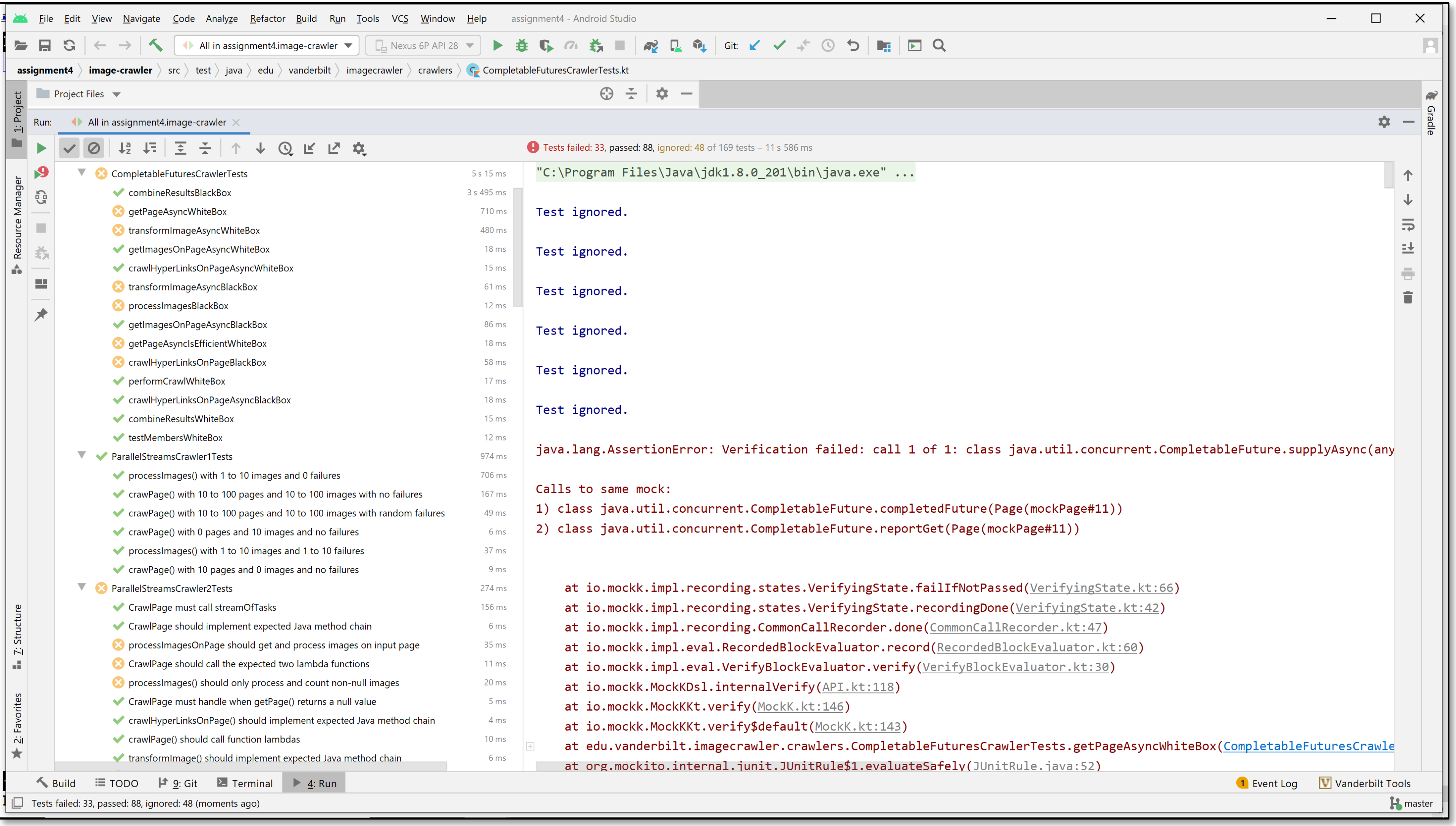
- Assignment assessments will be done via reviews by course staff
  - Assignments *must* be submitted on time or you'll get a 0
  - Your initial submission must compile & be largely complete or you won't get a review or a final grade
  - Work *must* be your own
    - This applies for quizzes & programming assignments


I WILL NOT PLAGIARIZE ANOTHER'S WORK
I WILL NOT PLAGIARIZE ANOTHER'S WORK
I WILL NOT PLAGIARIZE ANOTHER'S WORK
I WILL NOT PLAGIARIZE ANOTHER'S WORK
I WILL NOT PLAGIARIZE ANOTHER'S WORK
I WILL NOT PLAGIARIZE

www.vanderbilt.edu/student_handbook/the-honor-system#statement-of-the-honor-code

Overview of Assignments & Assessments

- The bulk of your grade is based on the results of the automated unit tests


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
assignment4 - Android Studio
All in assignment4.image-crawler
Nexus 6P API 28
Git:
assignment4 〉 image-crawler 〉 src 〉 test 〉 java 〉 edu 〉 vanderbilt 〉 imagecrawler 〉 crawlers 〉 CompletableFuturesCrawlerTests.kt
Project Files
Run: All in assignment4.image-crawler
1: Project
Resource Manager
7: Structure
2: Favorites
Gradle
Build
TODO
9: Git
Terminal
4: Run
Event Log
Vanderbilt Tools
master

Tests failed: 33, passed: 88, ignored: 48 of 169 tests – 11 s 586 ms

```
CompletableFuturesCrawlerTests                                        5 s 15 ms
    combineResultsBlackBox                                             3 s 495 ms
    getPageAsyncWhiteBox                                               710 ms
    transformImageAsyncWhiteBox                                        480 ms
    getImagesOnPageAsyncWhiteBox                                       18 ms
    crawlHyperLinksOnPageAsyncWhiteBox                                 15 ms
    transformImageAsyncBlackBox                                        61 ms
    processImagesBlackBox                                              12 ms
    getImagesOnPageAsyncBlackBox                                       86 ms
    getPageAsyncIsEfficientWhiteBox                                    18 ms
    crawlHyperLinksOnPageBlackBox                                      58 ms
    performCrawlWhiteBox                                               17 ms
    crawlHyperLinksOnPageAsyncBlackBox                                 18 ms
    combineResultsWhiteBox                                             15 ms
    testMembersWhiteBox                                                12 ms
ParallelStreamsCrawler1Tests                                           974 ms
    processImages() with 1 to 10 images and 0 failures                 706 ms
    crawPage() with 10 to 100 pages and 10 to 100 images with no failures   167 ms
    crawPage() with 10 to 100 pages and 10 to 100 images with random failures   49 ms
    crawPage() with 0 pages and 10 images and no failures              6 ms
    processImages() with 1 to 10 images and 1 to 10 failures           37 ms
    crawPage() with 10 pages and 0 images and no failures              9 ms
ParallelStreamsCrawler2Tests                                           274 ms
    CrawlPage must call streamOfTasks                                  156 ms
    CrawlPage should implement expected Java method chain              6 ms
    processImagesOnPage should get and process images on input page    35 ms
    CrawlPage should call the expected two lambda functions            11 ms
    processImages() should only process and count non-null images      20 ms
    CrawlPage must handle when getPage() returns a null value          5 ms
    crawlHyperLinksOnPage() should implement expected Java method chain   4 ms
    crawlPage() should call function lambdas                           10 ms
    transformImage() should implement expected Java method chain       6 ms
```

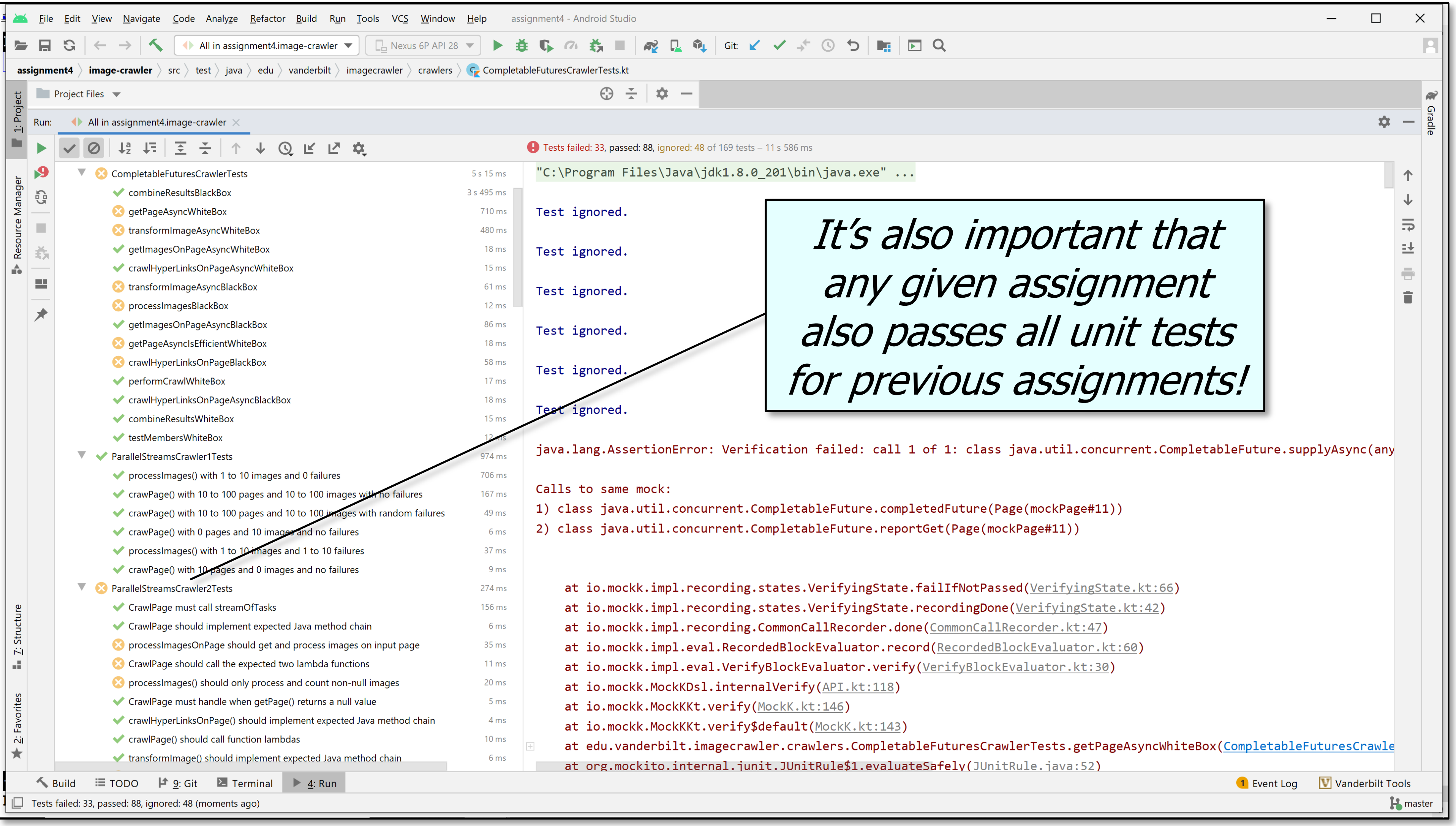
```
"C:\Program Files\Java\jdk1.8.0_201\bin\java.exe" ...

Test ignored.

Test ignored.

Test ignored.

Test ignored.

Test ignored.

Test ignored.

java.lang.AssertionError: Verification failed: call 1 of 1: class java.util.concurrent.CompletableFuture.supplyAsync(any

Calls to same mock:
1) class java.util.concurrent.CompletableFuture.completedFuture(Page(mockPage#11))
2) class java.util.concurrent.CompletableFuture.reportGet(Page(mockPage#11))


    at io.mockk.impl.recording.states.VerifyingState.failIfNotPassed(VerifyingState.kt:66)
    at io.mockk.impl.recording.states.VerifyingState.recordingDone(VerifyingState.kt:42)
    at io.mockk.impl.recording.CommonCallRecorder.done(CommonCallRecorder.kt:47)
    at io.mockk.impl.eval.RecordedBlockEvaluator.record(RecordedBlockEvaluator.kt:60)
    at io.mockk.impl.eval.VerifyBlockEvaluator.verify(VerifyBlockEvaluator.kt:30)
    at io.mockk.MockKDsl.internalVerify(API.kt:118)
    at io.mockk.MockKKt.verify(MockK.kt:146)
    at io.mockk.MockKKt.verify$default(MockK.kt:143)
    at edu.vanderbilt.imagecrawler.crawlers.CompletableFuturesCrawlerTests.getPageAsyncWhiteBox(CompletableFuturesCrawle
    at org.mockito.internal.junit.JUnitRule$1.evaluateSafely(JUnitRule.java:52)
```

Tests failed: 33, passed: 88, ignored: 48 (moments ago)

See www.dre.vanderbilt.edu/~schmidt/cs891/assignments.html

Overview of Assignments & Assessments

- The bulk of your grade is based on the results of the automated unit tests


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
assignment4 - Android Studio
All in assignment4.image-crawler
Nexus 6P API 28
Git:
assignment4 〉 image-crawler 〉 src 〉 test 〉 java 〉 edu 〉 vanderbilt 〉 imagecrawler 〉 crawlers 〉 CompletableFuturesCrawlerTests.kt
1: Project
Resource Manager
Project Files
Run: All in assignment4.image-crawler
Gradle
7: Structure
2: Favorites
Build
TODO
9: Git
Terminal
4: Run
Event Log
Vanderbilt Tools
master

Tests failed: 33, passed: 88, ignored: 48 of 169 tests – 11 s 586 ms

```
CompletableFuturesCrawlerTests                                          5 s 15 ms
    combineResultsBlackBox                                              3 s 495 ms
    getPageAsyncWhiteBox                                                710 ms
    transformImageAsyncWhiteBox                                         480 ms
    getImagesOnPageAsyncWhiteBox                                        18 ms
    crawlHyperLinksOnPageAsyncWhiteBox                                  15 ms
    transformImageAsyncBlackBox                                         61 ms
    processImagesBlackBox                                               12 ms
    getImagesOnPageAsyncBlackBox                                        86 ms
    getPageAsyncIsEfficientWhiteBox                                     18 ms
    crawlHyperLinksOnPageBlackBox                                       58 ms
    performCrawlWhiteBox                                                17 ms
    crawlHyperLinksOnPageAsyncBlackBox                                  18 ms
    combineResultsWhiteBox                                              15 ms
    testMembersWhiteBox                                                 12 ms
ParallelStreamsCrawler1Tests                                            974 ms
    processImages() with 1 to 10 images and 0 failures                  706 ms
    crawPage() with 10 to 100 pages and 10 to 100 images with no failures   167 ms
    crawPage() with 10 to 100 pages and 10 to 100 images with random failures   49 ms
    crawPage() with 0 pages and 10 images and no failures               6 ms
    processImages() with 1 to 10 images and 1 to 10 failures            37 ms
    crawPage() with 10 pages and 0 images and no failures               9 ms
ParallelStreamsCrawler2Tests                                            274 ms
    CrawlPage must call streamOfTasks                                   156 ms
    CrawlPage should implement expected Java method chain               6 ms
    processImagesOnPage should get and process images on input page     35 ms
    CrawlPage should call the expected two lambda functions             11 ms
    processImages() should only process and count non-null images       20 ms
    CrawlPage must handle when getPage() returns a null value           5 ms
    crawlHyperLinksOnPage() should implement expected Java method chain 4 ms
    crawlPage() should call function lambdas                            10 ms
    transformImage() should implement expected Java method chain        6 ms
```

```
"C:\Program Files\Java\jdk1.8.0_201\bin\java.exe" ...

Test ignored.

Test ignored.

Test ignored.

Test ignored.

Test ignored.

Test ignored.

java.lang.AssertionError: Verification failed: call 1 of 1: class java.util.concurrent.CompletableFuture.supplyAsync(any

Calls to same mock:
1) class java.util.concurrent.CompletableFuture.completedFuture(Page(mockPage#11))
2) class java.util.concurrent.CompletableFuture.reportGet(Page(mockPage#11))


    at io.mockk.impl.recording.states.VerifyingState.failIfNotPassed(VerifyingState.kt:66)
    at io.mockk.impl.recording.states.VerifyingState.recordingDone(VerifyingState.kt:42)
    at io.mockk.impl.recording.CommonCallRecorder.done(CommonCallRecorder.kt:47)
    at io.mockk.impl.eval.RecordedBlockEvaluator.record(RecordedBlockEvaluator.kt:60)
    at io.mockk.impl.eval.VerifyBlockEvaluator.verify(VerifyBlockEvaluator.kt:30)
    at io.mockk.MockKDsl.internalVerify(API.kt:118)
    at io.mockk.MockKKt.verify(MockK.kt:146)
    at io.mockk.MockKKt.verify$default(MockK.kt:143)
    at edu.vanderbilt.imagecrawler.crawlers.CompletableFuturesCrawlerTests.getPageAsyncWhiteBox(CompletableFuturesCrawle
    at org.mockito.internal.junit.JUnitRule$1.evaluateSafely(JUnitRule.java:52)
```

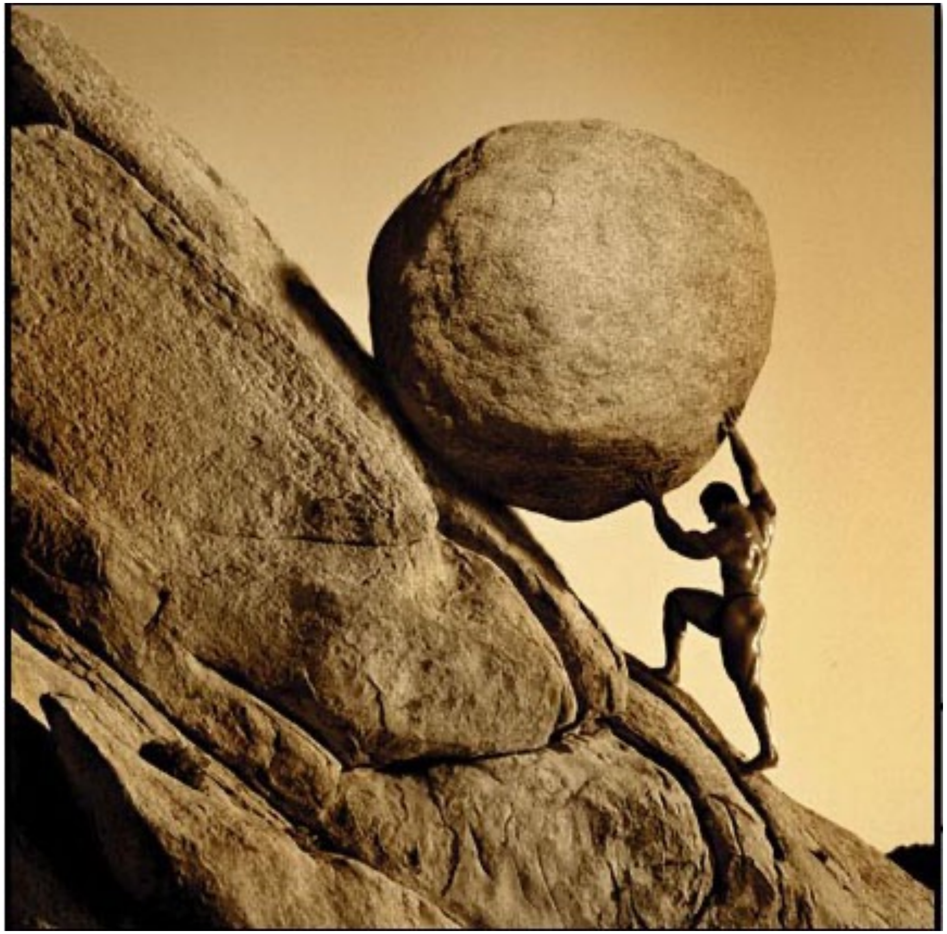
Tests failed: 33, passed: 88, ignored: 48 (moments ago)

It's also important that any given assignment also passes all unit tests for previous assignments!

See item #16 at github.com/douglascraigschmidt/CS891/wiki/CS-891-FAQ

Overview of Assignments & Assessments

- The relative weighting of each portion of the course is:
  - 45% Quizzes
  - 40% Programming projects
  - 10% Final exam
  - 05% Participation



These weightings may change, depending on various factors

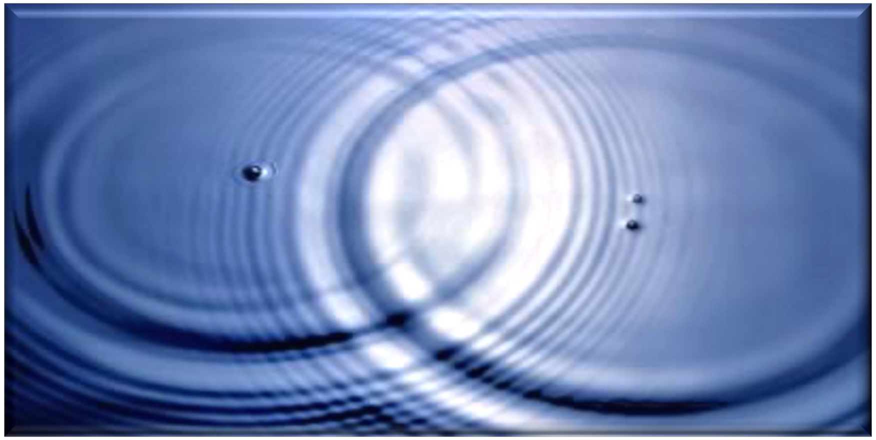
Overview of Assignments & Assessments

- The relative weighting of each portion of the course is:
  - 45% Quizzes
  - 40% Programming projects
  - 10% Final exam
  - 05% Participation
    - Participation includes attendance, involvement, & “following directions”

Overview of Assignments & Assessments

- The relative weighting of each portion of the course is:
  - 45% Quizzes
  - 40% Programming projects
  - 10% Final exam
  - 05% Participation
    - Participation includes attendance, involvement, & "following directions"

Attendance also affects other aspects of your quiz & assignment grades



See www.dre.vanderbilt.edu/~schmidt/cs891/work-summary.html#quizzes & www.dre.vanderbilt.edu/~schmidt/cs891/assignments.html

Overview of Assignments & Assessments

- The relative weighting of each portion of the course is:
  - 45% Quizzes
  - 40% Programming projects
  - 10% Final exam
  - 05% Participation
    - Participation includes attendance, involvement, & “following directions”

Don’t expect to get an A in this class if you do not actively participate!!!!

CS 891: Scalable Microservices: Overview (Part 2)