

# **CS 253: Parallel Functional Programming w/ Java & Android: Overview & Logistics (Part 1)**

**Douglas C. Schmidt**

**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**

**[www.dre.vanderbilt.edu/~schmidt](http://www.dre.vanderbilt.edu/~schmidt)**



**Professor of Computer Science**

**Institute for Software  
Integrated Systems**

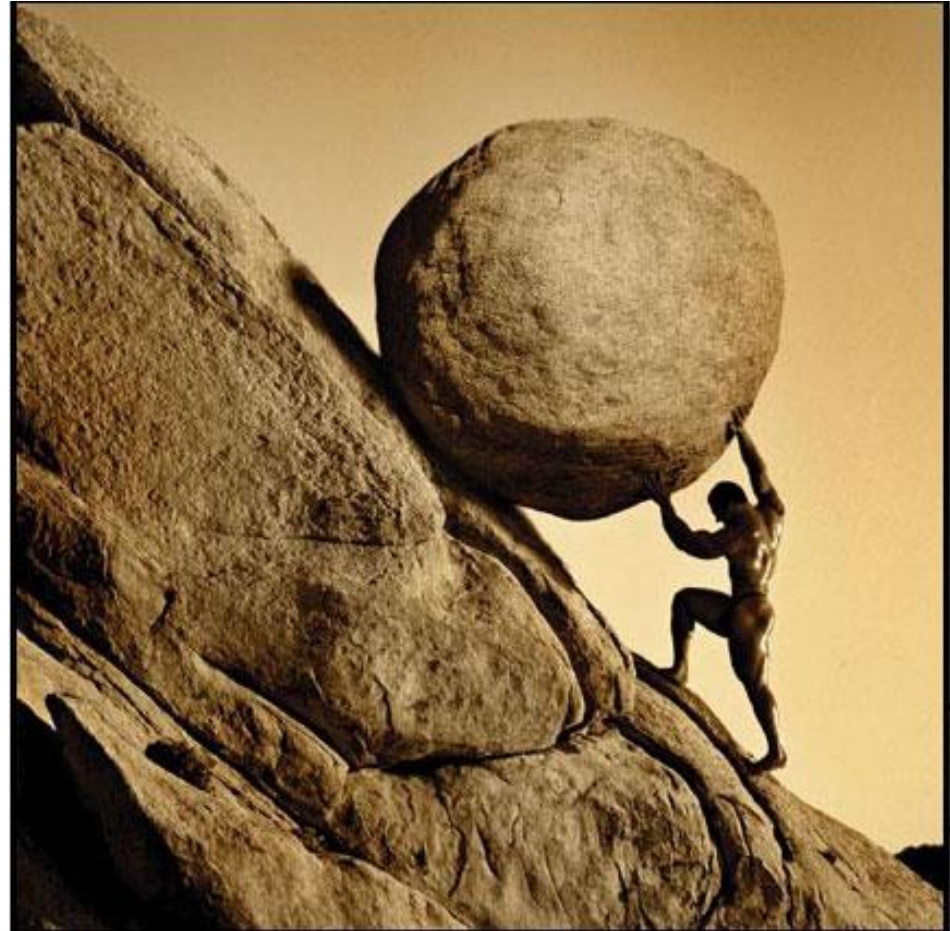
**Vanderbilt University  
Nashville, Tennessee, USA**



# Learning Objectives in this Lesson

---

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

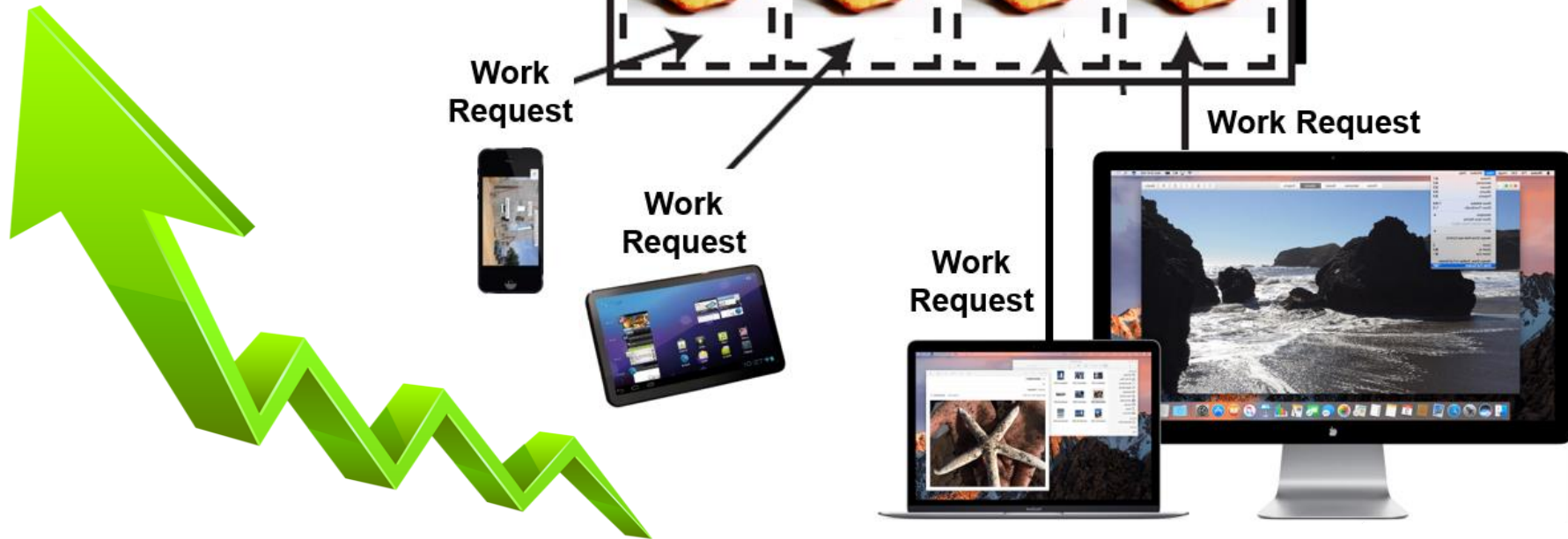


---

# Course Philosophy

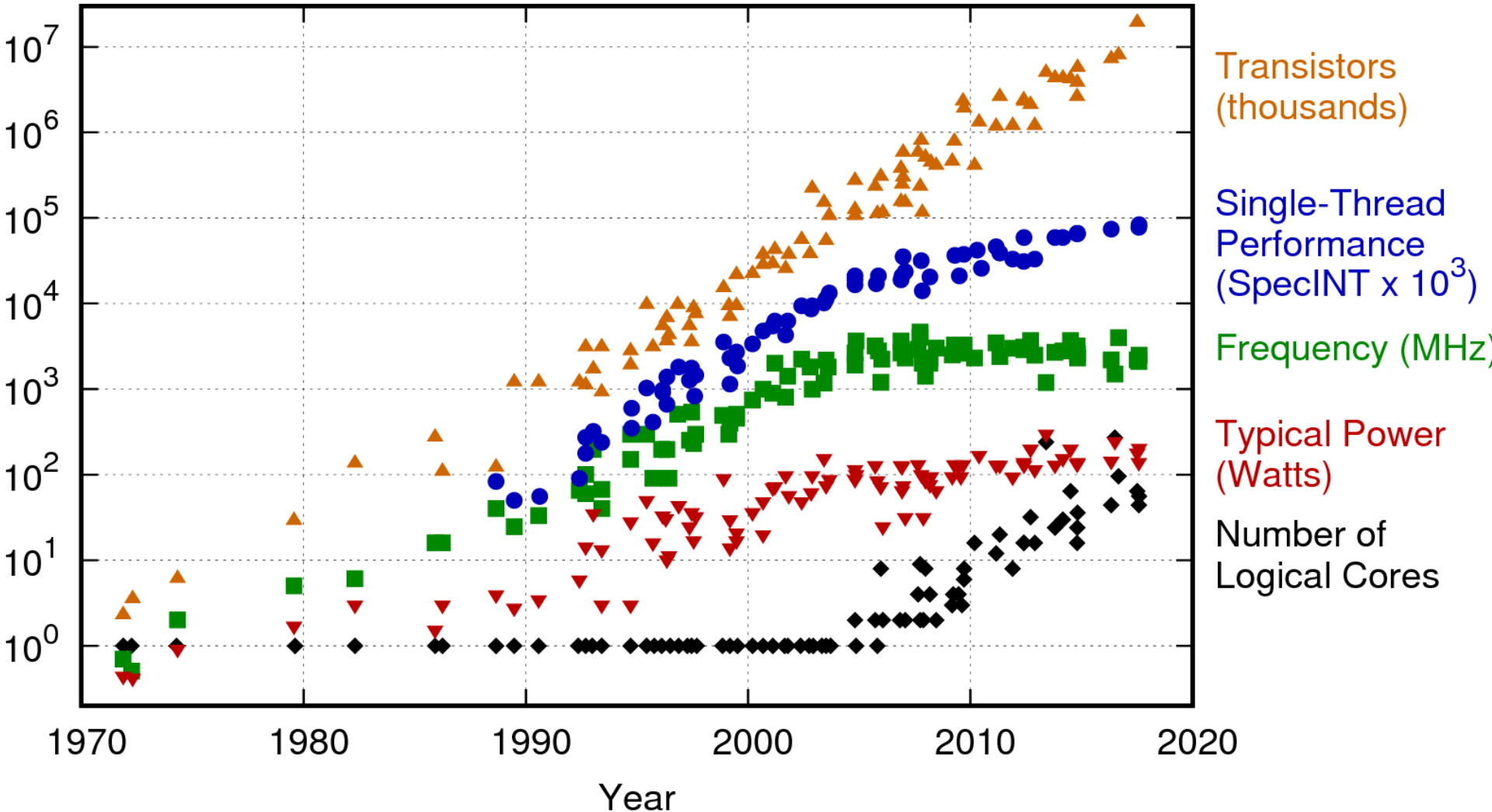
# Course Philosophy

- There's a growing need for software developers who know how to write parallel programs for a range of computing platforms
- e.g., mobile devices, laptops, desktops, & cloud environments



# Course Philosophy

- Demand is driven by software/hardware infrastructure advances

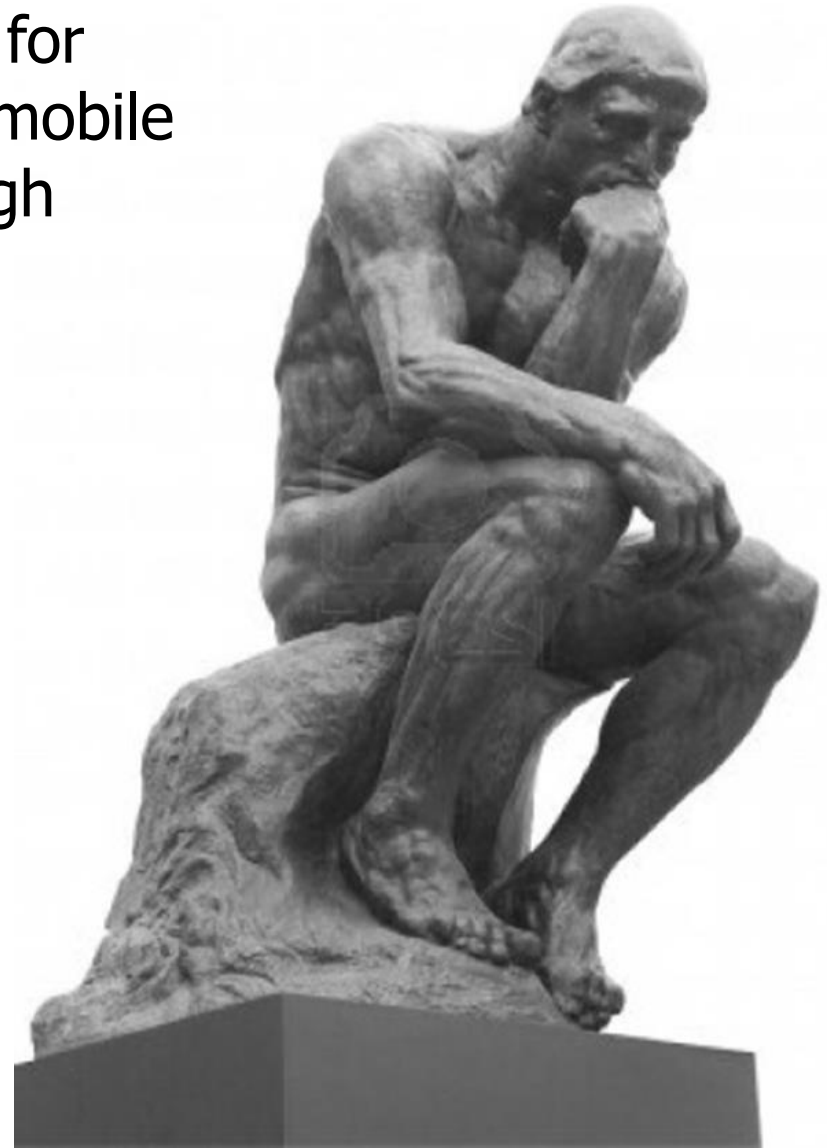


See [www.gotw.ca/publications/concurrency-ddj.htm](http://www.gotw.ca/publications/concurrency-ddj.htm)

# Course Philosophy

---

- Effective techniques & practices for developing parallel programs & mobile apps are *not* best learned through generalities & platitudes



---

“Sitting & thinking” is not sufficient...



# Course Philosophy

---

- Instead, it's better to see *by example* how these programs can be made
  - *easier* to write & read,
  - *easier* to maintain & modify,
  - *more* efficient & resilientby applying time-proven software patterns & object-oriented & functional design & programming techniques



---

This course involves lots of hands-on software development & testing!

---

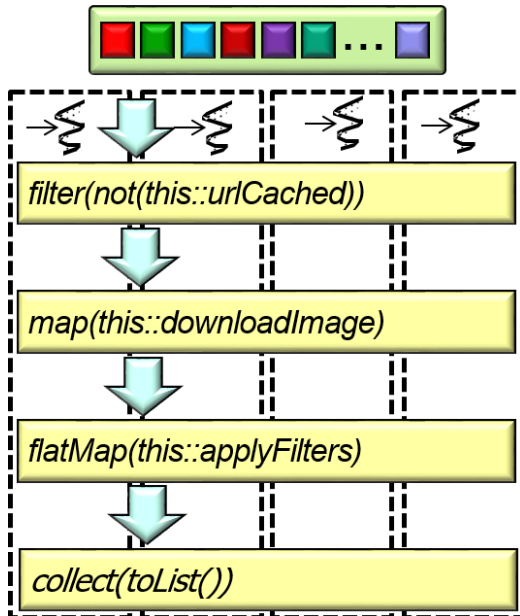
# Summary of the Course Contents



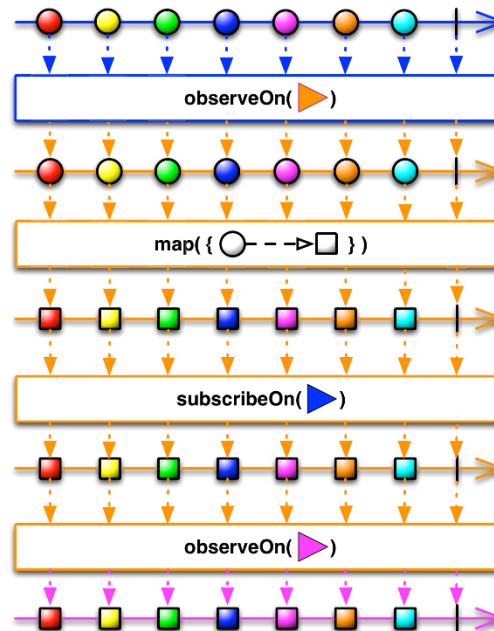
# Summary of Course Contents

- Key Java parallelism frameworks

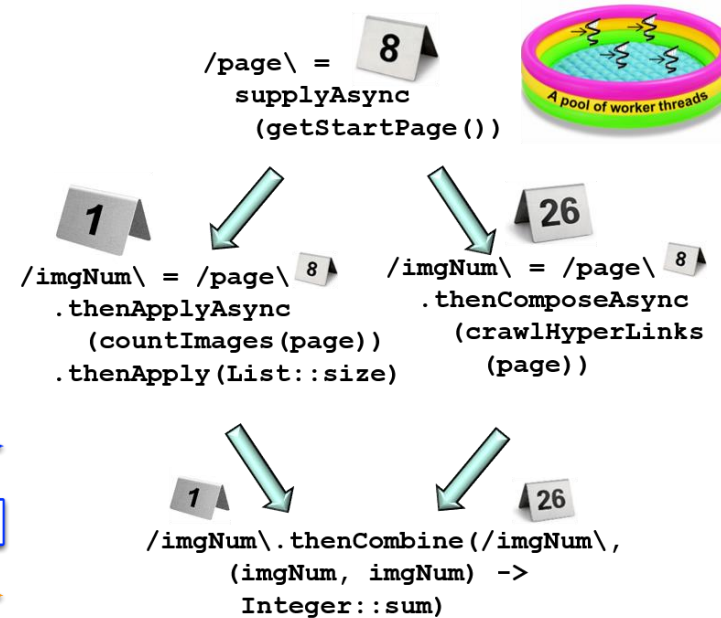
## Parallel Streams



## Reactive Streams



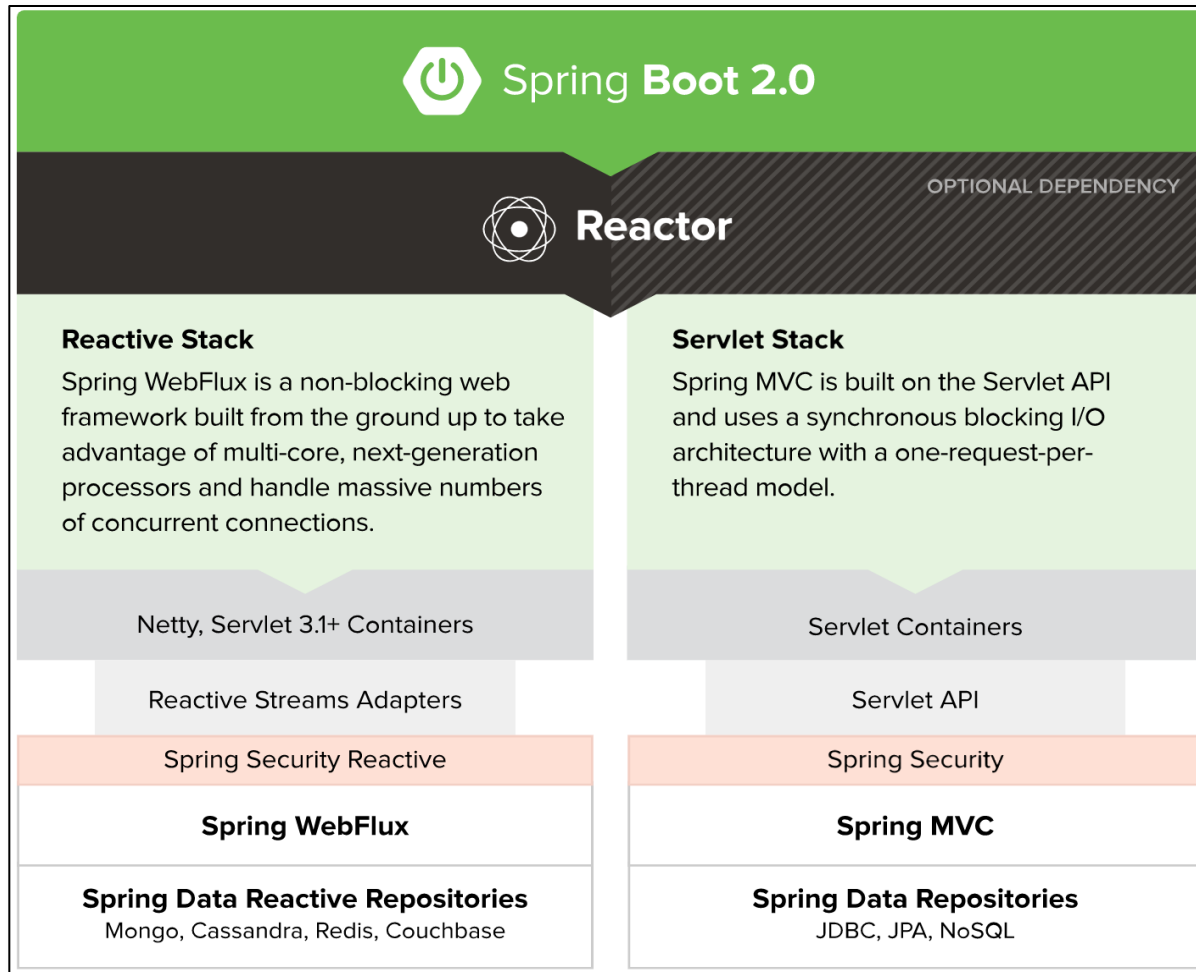
## Completable Futures



Assumes knowledge of Java object-oriented & functional language features

# Summary of Course Contents

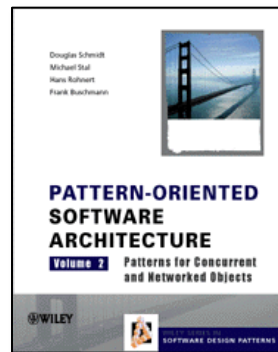
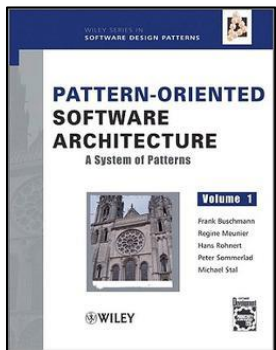
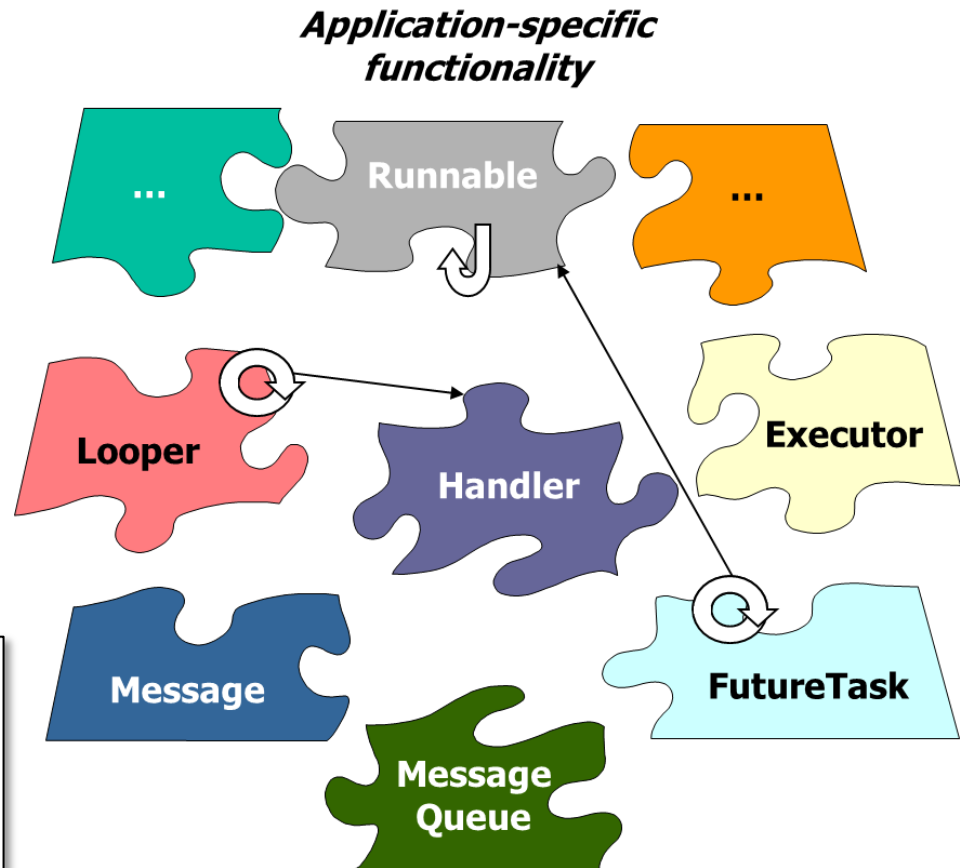
- Key Java parallelism frameworks
- Modern web programming platforms



See [spring.io/projects/spring-boot](https://spring.io/projects/spring-boot)

# Summary of Course Contents

- Key Java parallelism frameworks
- Modern web programming platforms
- Patterns for parallel programming



See [www.dre.Vanderbilt.edu/~Schmidt/POSA](http://www.dre.Vanderbilt.edu/~Schmidt/POSA)

# Summary of Course Contents

---

- Key Java parallelism frameworks
- Modern web programming platforms
- Patterns for parallel programming
- We assume you know (or can quickly learn) modern Java, Android, & Git



---

See item #12 at [github.com/douglasraigschmidt/CS253/wiki/CS-253-FAQ](https://github.com/douglasraigschmidt/CS253/wiki/CS-253-FAQ)

---

# Structure of the Lecture Material

# Structure of the Lecture Material

---

- This course has three main modules

Section	Topics
Java Parallelism	<ul style="list-style-type: none"><li>• Coverage of modern Java parallelism frameworks, e.g.<ul style="list-style-type: none"><li>• Java sequential &amp; parallel streams</li><li>• Java completable futures</li><li>• Reactive streams (e.g., RxJava &amp; Project Reactor)</li></ul></li></ul>

# Structure of the Lecture Material

---

- This course has three main modules

Section	Topics
Java Parallelism	<ul style="list-style-type: none"><li>• Coverage of modern Java parallelism frameworks, e.g.<ul style="list-style-type: none"><li>• Java sequential &amp; parallel streams</li><li>• Java completable futures</li><li>• Reactive streams (e.g., RxJava &amp; Project Reactor)</li></ul></li></ul>
Mobile ⇔ Web Communication	<ul style="list-style-type: none"><li>• Spring WebMVC &amp; WebFlux</li></ul>



# Structure of the Lecture Material

---

- This course has three main modules

Section	Topics
Java Parallelism	<ul style="list-style-type: none"><li>• Coverage of modern Java parallelism frameworks, e.g.<ul style="list-style-type: none"><li>• Java sequential &amp; parallel streams</li><li>• Java completable futures</li><li>• Reactive streams (e.g., RxJava &amp; Project Reactor)</li></ul></li></ul>
Mobile $\leftrightarrow$ Web Communication	<ul style="list-style-type: none"><li>• Spring WebMVC &amp; WebFlux</li></ul>
Software Patterns	<ul style="list-style-type: none"><li>• Parallel programming &amp; communication patterns</li></ul>

# Structure of the Lecture Material

---

- This course has three main modules
  - Each module is composed of lessons



# Structure of the Lecture Material

---

- This course has three main modules
  - Each module is composed of lessons
  - Each lesson is composed of parts



# Structure of the Lecture Material

- This course has three main modules
  - Each module is composed of lessons
  - Each lesson is composed of parts
  - Each part is a single lecture



Screencasts of each lesson “part” & PDF versions of the slides will be uploaded to [www.dre.vanderbilt.edu/~schmidt/cs253#lectures](http://www.dre.vanderbilt.edu/~schmidt/cs253#lectures)

# Structure of the Lecture Material

---

- This course has three main modules
  - Each module is composed of lessons
  - Each lesson is composed of parts
  - Each part is a single lecture
    - Each part is composed of segments





# Structure of the Lecture Material

---

- There will be bi-weekly quizzes on material covered in the lectures



# Structure of the Lecture Material

---

- There will be bi-weekly quizzes on material covered in the lectures
- 1<sup>st</sup> quiz will be on Wednesday, September 1<sup>st</sup>



---

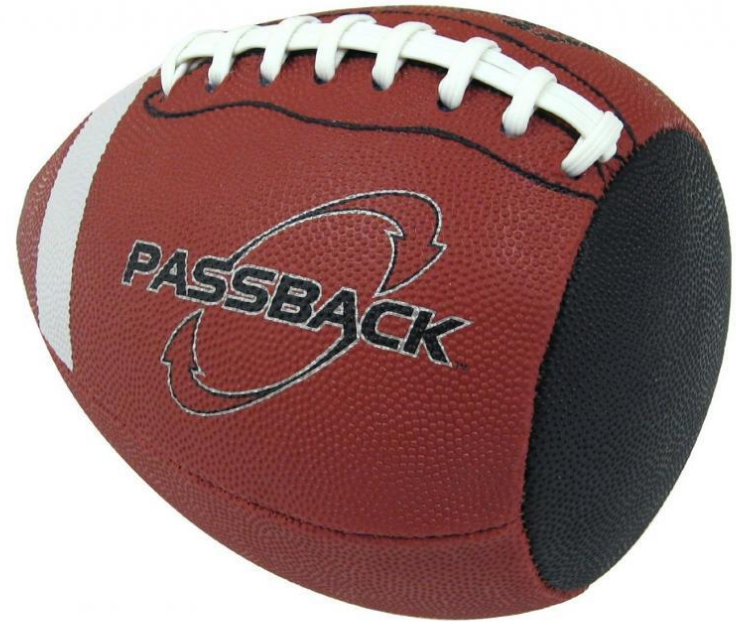
All quizzes are “closed book/note/Internet” & are given on Brightspace



# Structure of the Lecture Material

---

- There will be bi-weekly quizzes on material covered in the lectures
  - 1<sup>st</sup> quiz will be on Wednesday, September 1<sup>st</sup>
- We strive to hand back & review quizzes at the start of next class



---

One of the benefits of a smaller class ;-)

# Structure of the Lecture Material

---

- There will be bi-weekly quizzes on material covered in the lectures
  - 1<sup>st</sup> quiz will be on Wednesday, September 1<sup>st</sup>
- We strive to hand back & review quizzes at the start of next class



I recommend that you study for quizzes by reviewing slides & watching screencasts available at [www.dre.vanderbilt.edu/~schmidt/cs253#lectures](http://www.dre.vanderbilt.edu/~schmidt/cs253#lectures)

# Structure of the Lecture Material

---

- There *may* be a cumulative final exam that covers all the lectures
- The focus will be on the last week(s) of the semester



---

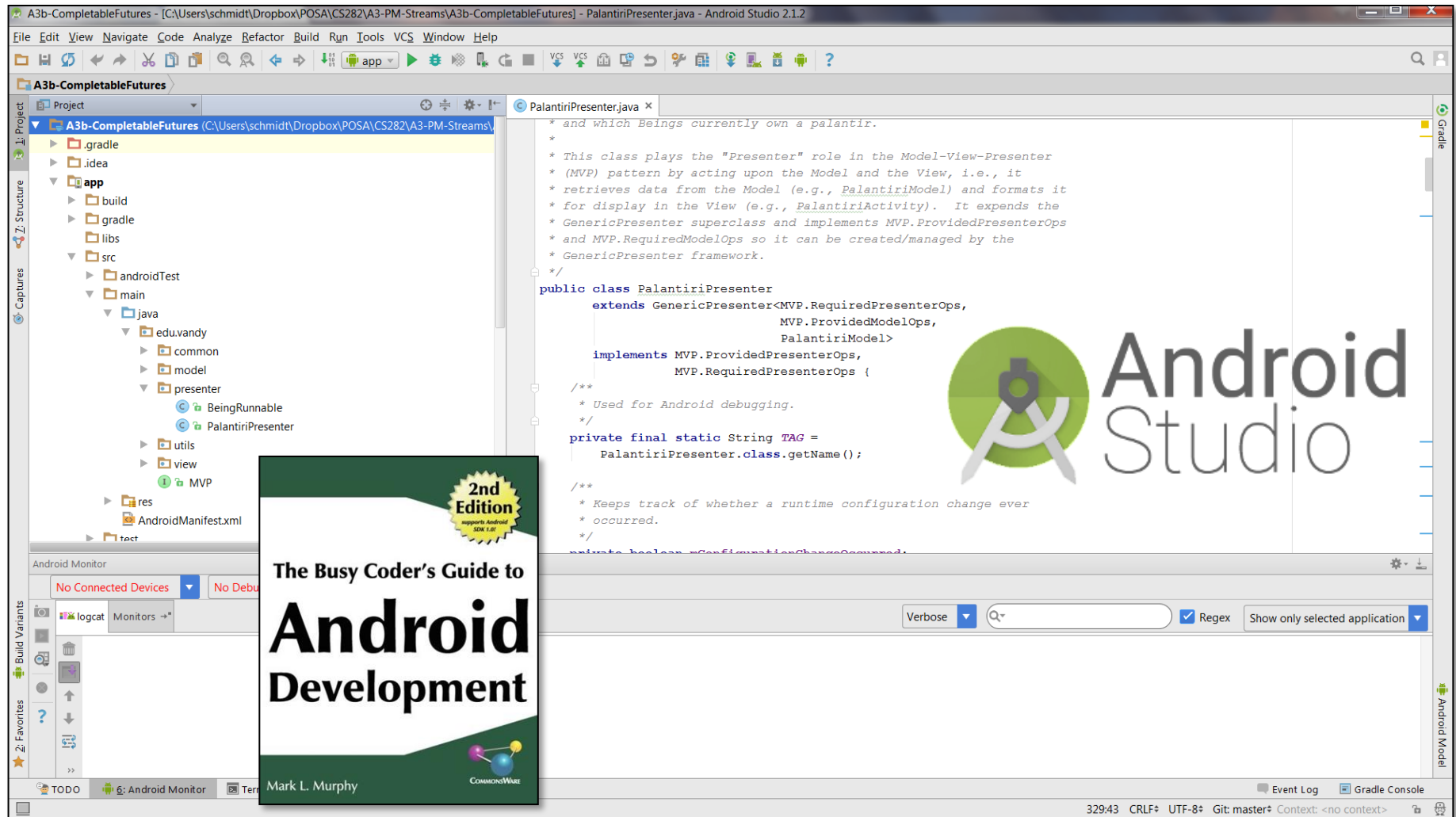
The final exam is 7 to 9pm, Saturday, December 18<sup>th</sup> via Brightspace

---

# Overview of Assignments & Assessments

# Overview of Assignments & Assessments

- Programming assignments are written in modern Java using Android Studio



You can use any IDE, but your final submission *must* build & run with the latest Android Studio & Android 11 (API 30)

# Overview of Assignments & Assessments

---

- Programming assignments are written in modern Java using Android Studio
- The Java runtime environment (JRE) is pre-installed with Android



---

See [github.com/douglasraigschmidt/CS253/wiki/Installing-Software](https://github.com/douglasraigschmidt/CS253/wiki/Installing-Software)

# Overview of Assignments & Assessments

---

- Android programming assignments must be submitted using Android Studio

- A wizard for creating new apps
- A visual editor for creating GUIs
- An editor for manipulating Android XML descriptors needed for your app
- An emulator for testing your apps on your PC
- A debugger for finding errors in the emulator or on a device



---

See [developer.android.com/sdk](https://developer.android.com/sdk)



# Overview of Assignments & Assessments

---

- Android programming assignments must be submitted using Android Studio
  - Please install Android 11 (API level 30)



---

See [en.wikipedia.org/wiki/Android\\_11](https://en.wikipedia.org/wiki/Android_11)

# Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub

The screenshot shows the GitHub repository page for `douglasraigschmidt / CS253`. The repository is on the `master` branch, has 1 branch, and 0 tags. The commit history shows two commits: `assignment1a` (updates, 2 hours ago) and `README.md` (Initial commit, yesterday). The `README.md` file is displayed, containing the title `CS253` and a description: "Contains examples and assignments for my CS 253 course at Vanderbilt University, which can be accessed via <http://www.dre.vanderbilt.edu/~schmidt/cs253>".

`douglasraigschmidt / CS253`

<> Code   Issues   Pull requests   Actions   Projects   Wiki   Security   Insights   Settings

master   1 branch   0 tags   Go to file   Add file   Code

`douglasraigschmidt` updates   1617d1a 2 hours ago   2 commits

assignment1a	updates	2 hours ago
README.md	Initial commit	yesterday

README.md

## CS253

Contains examples and assignments for my CS 253 course at Vanderbilt University, which can be accessed via <http://www.dre.vanderbilt.edu/~schmidt/cs253>

Go to GitHub at [github.com/douglasraigschmidt/CS253](https://github.com/douglasraigschmidt/CS253)

# Overview of Assignments & Assessments

---

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

A screenshot of the GitLab landing page. The background is a solid purple color. In the top left corner is the GitLab logo (a cat face) and the text "GitLab". In the top right corner is a white hamburger menu icon. The main heading is "Open source software to collaborate on code" in a large, white, sans-serif font. Below this is a paragraph of white text describing GitLab's features: "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." Below this paragraph is another paragraph of white text: "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." At the bottom of the page are two buttons: a light blue button with the text "GitLab Community Edition" and a green button with the text "Get a subscription".

GitLab

## 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.

GitLab Community Edition

Get a subscription

# 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 Java 8 & Android parallel programs

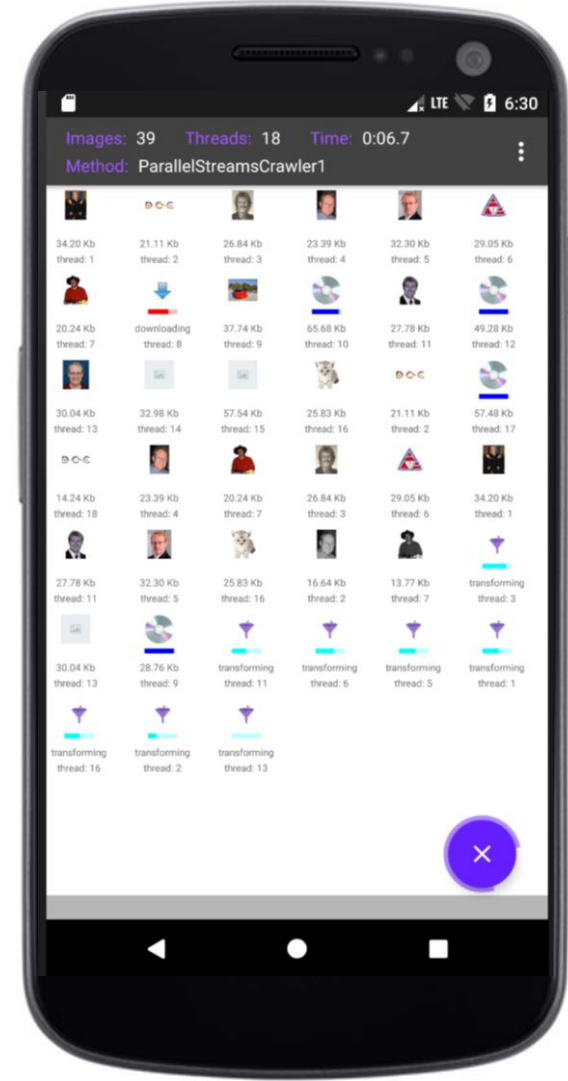


---

Go to GitHub at [github.com/douglasraigschmidt/CS253](https://github.com/douglasraigschmidt/CS253)

# Overview of Assignments & Assessments

- Assignments will provide a range of experience with Java 8 & Android parallel programs
  - Implement an image crawler app on Android & Spring using modern Java features, e.g.
    - Java lambda expressions, method references, & functional interfaces
    - Java sequential & parallel streams
    - Java completable futures
    - Java reactive streams
    - Spring WebSvc & 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



# 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



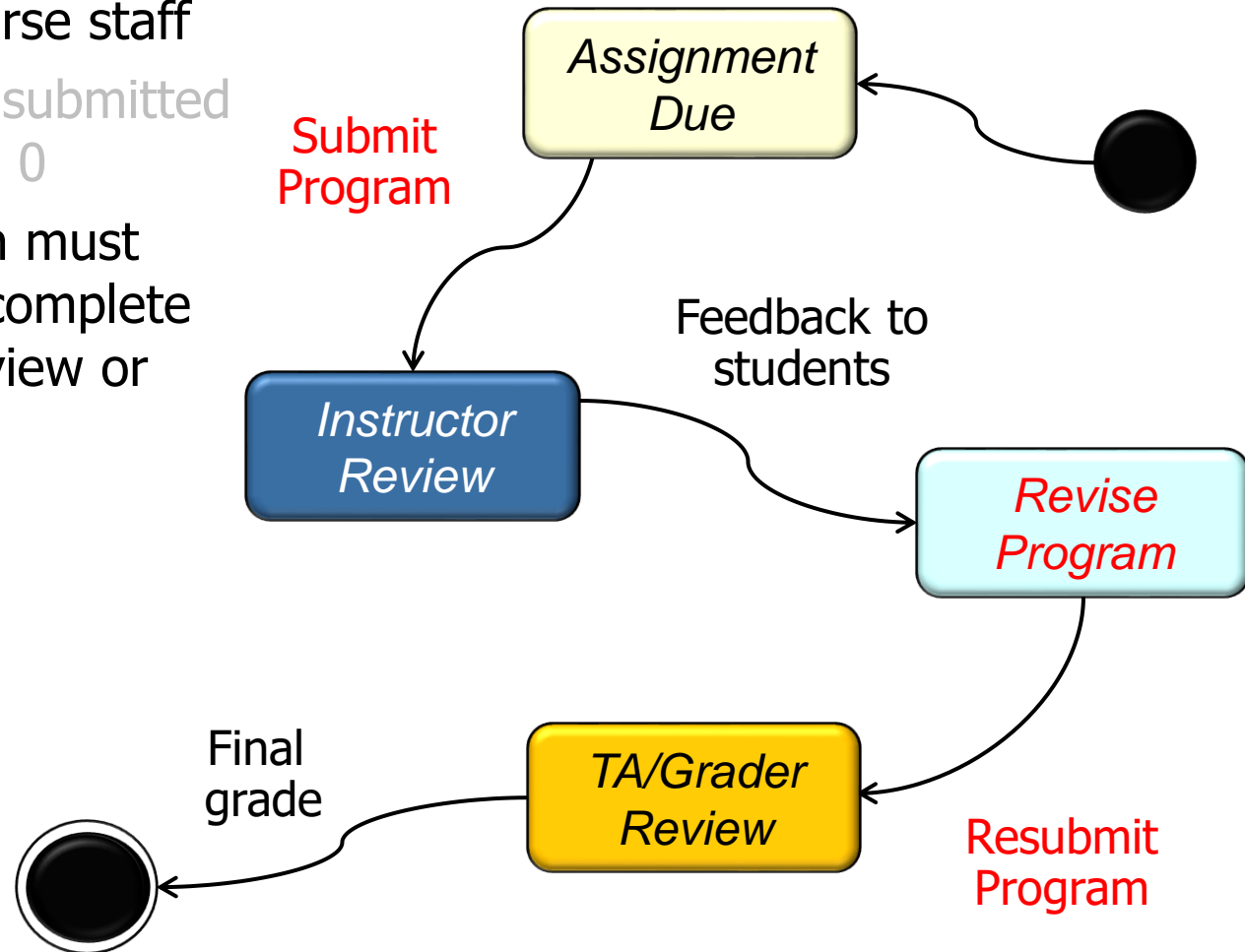
---

See [github.com/douglasraigschmidt/CS253/wiki/CS-253-FAQ](https://github.com/douglasraigschmidt/CS253/wiki/CS-253-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



# 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 will not receive a grade for assignments if you do not attend class regularly

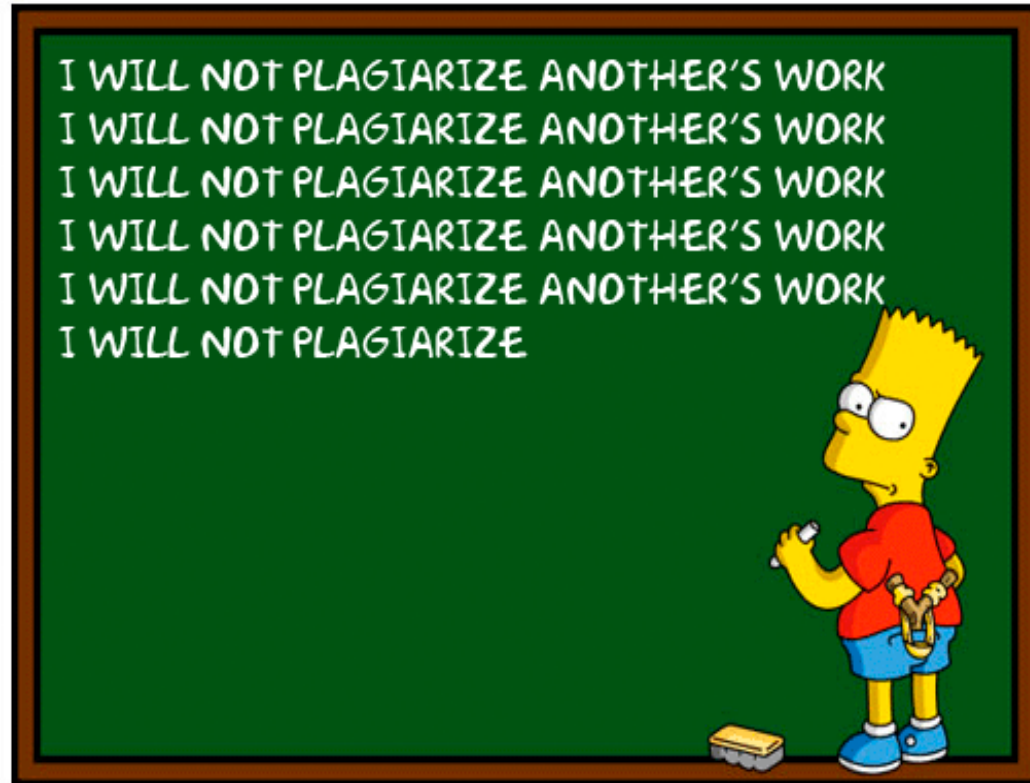


---

See [www.dre.vanderbilt.edu/~schmidt/cs253/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs253/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 will not receive a grade for assignments if you do not attend class regularly
  - Work *must* be your own
    - This applies for quizzes & programming assignments



# Overview of Assignments & Assessments

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

The screenshot displays the Android Studio interface for a project named 'assignment4'. The 'Run' tab is active, showing the execution of unit tests for the 'image-crawler' module. The 'Test Results' panel on the right shows the output of the tests, including a list of failed tests and their corresponding error messages.

**Test Results Summary:**

- CompletetablesCrawlerTests:** 33 tests failed, 88 passed, 48 ignored. Total time: 11 s 586 ms.
- ParallelStreamsCrawler1Tests:** 1 test failed, 167 passed, 0 ignored. Total time: 706 ms.
- ParallelStreamsCrawler2Tests:** 1 test failed, 156 passed, 0 ignored. Total time: 274 ms.

**Failed Test Details:**

- Test ignored.** (Multiple instances)
- 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))
- Stack Trace:**
  - 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.CompletetablesCrawlerTests.getPageAsyncWhiteBox(CompletetablesCrawlerTests.kt:143)
  - at org.mockito.internal.junit.JUnitRule\$1.evaluateSafely(JUnitRule.java:52)

See [www.dre.vanderbilt.edu/~schmidt/cs253/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs253/assignments.html)

# Overview of Assignments & Assessments

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

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

*It's important that your current assignment also passes all the unit tests for previous assignments!*

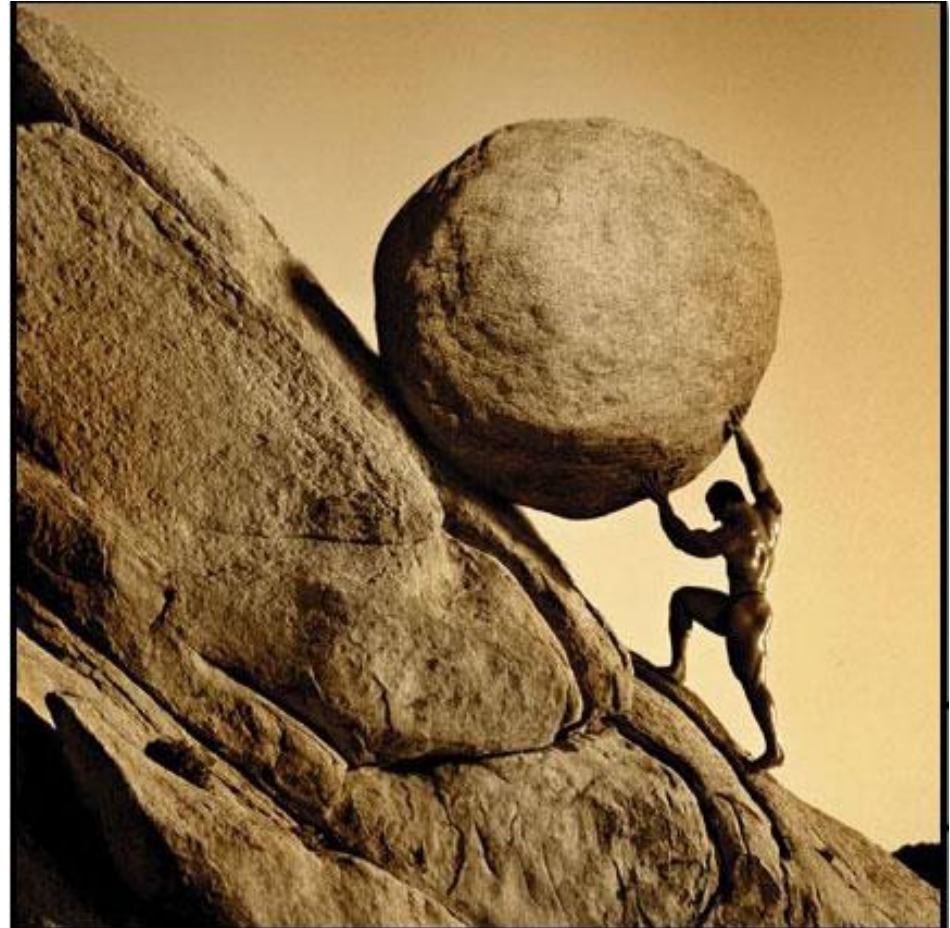
java.lang.AssertionError: Verification failed: call 1 of 1: class java.util.concurrent.CompletableFuture.supplyAsync(any  
Calls to same mock:  
1) class java.util.concurrent.CompletableFuture.completedFuture(Page(mockPage#11))  
2) class java.util.concurrent.CompletableFuture.reportGet(Page(mockPage#11))  
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(CompletableFuturesCrawlerTests.kt:143)  
at org.mockito.internal.junit.JUnitRule\$1.evaluateSafely(JUnitRule.java:52)

See item #16 at [github.com/douglasraigschmidt/CS253/wiki/CS-253-FAQ](https://github.com/douglasraigschmidt/CS253/wiki/CS-253-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”

**IMPORTANT**

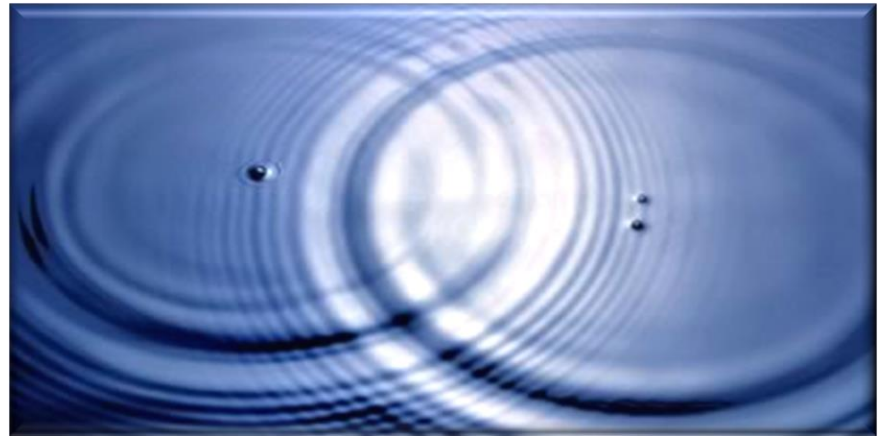


# Overview of Assignments & Assessments

- 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*

**IMPORTANT**



See [www.dre.vanderbilt.edu/~schmidt/cs253/work-summary.html#quizzes](http://www.dre.vanderbilt.edu/~schmidt/cs253/work-summary.html#quizzes)  
& [www.dre.vanderbilt.edu/~schmidt/cs253/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs253/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”

**IMPORTANT**



---

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

---

End of Part 1