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

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
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
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 & resilient by 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

- `filter(not(this::urlCached))`
- `map(this::downloadImage)`
- `flatMap(this::applyFilters)`
- `collect(toList())`

### Reactive Streams

- `observeOn( )`
- `subscribeOn( )`
- `map( )`
- `subscribeOn( )`
- `observeOn( )`

### Completable Futures

- `/page\ = 8
  supplyAsync
  (getStartPage())`
- `/imgNum\ = /page\ 8
  .thenApplyAsync
  (countImages(page))
  .thenApply(List::size)`
- `/imgNum\ = /page\ 8
  .thenComposeAsync
  (crawlHyperLinks
  (page))`
- `/imgNum\ .thenCombine(/imgNum\
  (imgNum, imgNum) ->
  Integer::sum)`

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](http://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
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/douglascraigschmidt/CS253/wiki/CS-253-FAQ
Structure of the Lecture Material
This course has three main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Parallelism</td>
<td>• Coverage of modern Java parallelism frameworks, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java sequential &amp; parallel streams</td>
</tr>
<tr>
<td></td>
<td>• Java completable futures</td>
</tr>
<tr>
<td></td>
<td>• Reactive streams (e.g., RxJava &amp; Project Reactor)</td>
</tr>
</tbody>
</table>
### Structure of the Lecture Material

- This course has three main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Parallelism</td>
<td>• Coverage of modern Java parallelism frameworks, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java sequential &amp; parallel streams</td>
</tr>
<tr>
<td></td>
<td>• Java completable futures</td>
</tr>
<tr>
<td></td>
<td>• Reactive streams (e.g., RxJava &amp; Project Reactor)</td>
</tr>
<tr>
<td>Mobile ↔ Web Communication</td>
<td>• Spring WebMVC &amp; WebFlux</td>
</tr>
</tbody>
</table>
### Structure of the Lecture Material

- This course has three main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Parallelism</td>
<td>• Coverage of modern Java parallelism frameworks, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java sequential &amp; parallel streams</td>
</tr>
<tr>
<td></td>
<td>• Java completable futures</td>
</tr>
<tr>
<td></td>
<td>• Reactive streams (e.g., RxJava &amp; Project Reactor)</td>
</tr>
<tr>
<td>Mobile ⇔ Web Communication</td>
<td>• Spring WebMVC &amp; WebFlux</td>
</tr>
<tr>
<td>Software Patterns</td>
<td>• Parallel programming &amp; communication patterns</td>
</tr>
</tbody>
</table>
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
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
- 1st quiz will be on Wednesday, September 1st

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
  • 1st quiz will be on Wednesday, September 1st
  • 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
  - 1st quiz will be on Wednesday, September 1st
  - 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 18th 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/douglasschmidt/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
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
Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub

Go to GitHub at [github.com/douglascraigschmidt/CS253](https://github.com/douglascraigschmidt/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.
Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub
- 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/douglascraigslistschmidt/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

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

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

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

See item #16 at github.com/douglascraigschmidt/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”
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”

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”

Don’t expect to get an A in this class if you do not actively participate!!!!
End of Part 1