Learning Objectives in this Lesson

• Understand the course topics & logistics
• Course philosophy
• Course contents
• Structure of the lecture material
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)
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())`

**Completable Futures**
- `/page\ = 8`
- `supplyAsync`
  - `(getStartPage())`
- `/imgNum\ = /page\ 8`
- `.thenApplyAsync`
  - `(countImages(page))`
  - `.thenApply(List::size)`
- `/imgNum\ = /page\ 8`
- `.thenComposeAsync`
  - `(crawlHyperLinks(page))`

**Reactive Streams**
- `observeOn()`
- `subscribeOn()`
- `observeOn()`

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](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/douglasraignschmidt/CS253/wiki/CS-253-FAQ
Structure of the Lecture Material
### 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](http://www.dre.vanderbilt.edu/~schmidt/cs253#lectures)
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, August 31st

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, August 31st
  - 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, August 31st
  - 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 *may* be held 2 to 5pm, Friday, December 16th via Brightspace
CS 253: Parallel Functional Programming with Java, Android, & Spring WebFlux: Overview (Part 1)