CS 892 Overview & Logistics

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
- Setting up the Java & Android IDE on Android Studio
- Accessing Android & Java source code
Course
Philosophy
Course Philosophy

- Effective techniques & practices for designing & programming concurrent 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 a lot of hands-on software development & testing
Summary of the Course Contents
Summary of Course Contents

• Coverage of basic & advanced Java concurrency mechanisms

Including Java 8 programming language features & concurrency frameworks
Summary of Course Contents

- Coverage of basic & advanced Java concurrency mechanisms
- Coverage of Android bound services & inter-process communication (IPC) frameworks

```java
interface IDownload {
    String getImage(in Uri uri);
}
```
Summary of Course Contents

• Coverage of basic & advanced Java concurrency mechanisms
• Coverage of Android bound services & inter-process communication (IPC) frameworks
• Mobile & Web communication & content providers
Summary of Course Contents

• Coverage of basic & advanced Java concurrency mechanisms
• Coverage of Android bound services & inter-process communication (IPC) frameworks
• Mobile & Web communication & content providers
• Patterns/frameworks for concurrent & networked programming
Summary of Course Contents

• Coverage of basic & advanced Java concurrency mechanisms
• Coverage of Android bound services & inter-process communication (IPC) frameworks
• Mobile & Web communication & content providers
• Patterns/frameworks for concurrent & networked programming
• We assume you know (or can quickly learn) Android, Java, & Git

See www.coursera.org/specializations/android-app-development
Structure of the Lecture Material
This course has four main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java &amp; Android Threading</td>
<td>• Coverage of basic &amp; advanced Java &amp; Android threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads, Runnables, &amp; Executor framework</td>
</tr>
<tr>
<td></td>
<td>• Android HaMeR &amp; AsyncTask frameworks</td>
</tr>
<tr>
<td></td>
<td>• Java 8 concurrency mechanisms</td>
</tr>
</tbody>
</table>
Structure of the Lecture Material

- This course has four main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>- Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>- Java Threads, Runnables, &amp; Executor framework</td>
</tr>
<tr>
<td></td>
<td>- Android HaMeR &amp; AsyncTask frameworks</td>
</tr>
<tr>
<td></td>
<td>- Java 8 concurrency mechanisms</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>- Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>- Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>- Myriad synchronizer classes in java.util.concurrent</td>
</tr>
</tbody>
</table>
### Structure of the Lecture Material

- This course has four main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads, Runnables, &amp; Executor framework</td>
</tr>
<tr>
<td></td>
<td>• Android HaMeR &amp; AsyncTask frameworks</td>
</tr>
<tr>
<td></td>
<td>• Java 8 concurrency mechanisms</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
<tr>
<td>Mobile ↔ Web Communication</td>
<td>• Android Bound Services &amp; AIDL, Android Content Providers</td>
</tr>
</tbody>
</table>
This course has four main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Threading</td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads, Runnable, &amp; Executor framework</td>
</tr>
<tr>
<td></td>
<td>• Android HaMeR &amp; AsyncTask frameworks</td>
</tr>
<tr>
<td></td>
<td>• Java 8 concurrency mechanisms</td>
</tr>
<tr>
<td>Java Synchronization</td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
<tr>
<td>Mobile ⇒ Web Communication</td>
<td>• Android Bound Services &amp; AIDL, Android Content Providers</td>
</tr>
<tr>
<td>Software Patterns</td>
<td>• Concurrency &amp; communication patterns</td>
</tr>
</tbody>
</table>
Structure of the Lecture Material

- This course has four main modules

<table>
<thead>
<tr>
<th>Section</th>
<th>Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Java Threading</strong></td>
<td>• Coverage of basic &amp; advanced Java threading mechanisms, e.g.</td>
</tr>
<tr>
<td></td>
<td>• Java Threads, Runnables, &amp; Executor framework</td>
</tr>
<tr>
<td></td>
<td>• Android HaMeR &amp; AsyncTask frameworks</td>
</tr>
<tr>
<td></td>
<td>• Java 8 concurrency mechanisms</td>
</tr>
<tr>
<td><strong>Java Synchronization</strong></td>
<td>• Coverage of basic &amp; advanced Java synchronization mechanisms, e.g.,</td>
</tr>
<tr>
<td></td>
<td>• Build-in monitor objects</td>
</tr>
<tr>
<td></td>
<td>• Myriad synchronizer classes in java.util.concurrent</td>
</tr>
<tr>
<td><strong>Mobile ⇨ Web Communication</strong></td>
<td>• Android Bound Services &amp; AIDL, Android Content Providers</td>
</tr>
<tr>
<td><strong>Software Patterns</strong></td>
<td>• Concurrency &amp; communication patterns</td>
</tr>
</tbody>
</table>

We will bounce around a bit when covering these topics
Structure of the Lecture Material

• This course has four main modules

• Each module is composed of lessons
Structure of the Lecture Material

• This course has four main modules
  • Each module is composed of lessons
  • Each lesson is composed of parts
Structure of the Lecture Material

• This course has four 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/cs892#lectures](http://www.dre.vanderbilt.edu/~schmidt/cs892#lectures)
Structure of the Lecture Material

- This course has four 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

We will interactively cover discussion questions at the end of each part
Structure of the Lecture Material

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

All quizzes (and the final) are “closed book.”
Structure of the Lecture Material

- There will be a bi-weekly quizzes on material covered in the lectures
- 1st quiz will be next Wednesday
Structure of the Lecture Material

- There will be a bi-weekly quizzes on material covered in the lectures
  - 1st quiz will be next Wednesday
  - We’ll hand back & review quizzes at the start of the next class

One of the benefits of a smaller class ;-)}
Structure of the Lecture Material

- There will be a bi-weekly quizzes on material covered in the lectures
  - 1st quiz will be next Wednesday
  - We’ll hand back & review quizzes at the start of the next class

I recommend that you study for quizzes by reviewing slides & watching screencasts available at [www.dre.vanderbilt.edu/~schmidt/cs892#lectures](http://www.dre.vanderbilt.edu/~schmidt/cs892#lectures)
Structure of the Lecture Material

• There will be a cumulative final exam that covers all the lectures

The final exam will be held 2pm to 5pm, Thursday, May 4th in this room
Overview of the Assignments & Assessments
Overview of Assignments & Assessments

- Programming assignments should be written in Java 8 using Android Studio.

You can use any IDE, but your final submission must build/run with Android Studio 2.2.3 & Android 7 (API 24).
Overview of Assignments & Assessments

- Programming assignments should be written in Java 8 using Android Studio
- Please install the Java 8 runtime environment (JRE)

See github.com/douglascairnschmidt/CS892/wiki/InstallingSoftware
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 7.0 Nougat (API level 24)

See en.wikipedia.org/wiki/Android_Nougat
Overview of Assignments & Assessments

- All source code for assignments & examples available at GitHub

Go to GitHub at [github.com/douglascraigschmidt/CS892](https://github.com/douglascraigschmidt/CS892)
Overview of Assignments & Assessments

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

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.

Get a subscription

GitLab Community Edition
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 multiple times

“If you don’t like change, you’re going to like irrelevance even less.”
Assignments will provide a range of experience with Java 8 & Android concurrent/communication programs.

See [github.com/douglasraigschmidt/CS892/tree/master/assignments](https://github.com/douglasraigschmidt/CS892/tree/master/assignments)
Overview of Assignments & Assessments

• Assignments will provide a range of experience with Java 8 & Android concurrent/communication programs
  • Implement Java concurrent resource manager & associated Android app, e.g.
    • Java Threads, Runnables, & Executor frameworks
    • Android HaMeR & AsyncTask frameworks
    • Java 8 Streams & CompletableFutures
    • Java Semaphores, built-in monitor objects, & ConcurrentHashMaps
    • Android Bound Services & AIDL

The topics covered by the assignments may change during the semester
Overview of Assignments & Assessments

- Assignment assessments will be done via combination of reviews by course staff
Overview of Assignments & Assessments

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

See github.com/douglasraigschmidt/CS892/wiki/CS-892-FAQ
• Assignment assessments will be done via combination of 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
Overview of Assignments & Assessments

• Assignment assessments will be done via combination of 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
  • Work must be your own
  • This goes for quizzes & programming assignments

www.vanderbilt.edu/student_handbook/the-honor-system#statement-of-the-honor-code
### Overview of Assignments & Assessments

- **Assessment criteria**

<table>
<thead>
<tr>
<th>Assessment Category</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution correctness</td>
<td>40%</td>
</tr>
<tr>
<td>Structure (e.g., modularization, information hiding, etc.)</td>
<td>30%</td>
</tr>
<tr>
<td>Insightful programming (e.g., developing reusable class components, etc.)</td>
<td>10%</td>
</tr>
<tr>
<td>Consistent style (e.g., capitalization, indenting, etc.)</td>
<td>10%</td>
</tr>
<tr>
<td>Appropriate commenting style</td>
<td>10%</td>
</tr>
</tbody>
</table>

See [www.dre.vanderbilt.edu/~schmidt/cs892/assignments.html](http://www.dre.vanderbilt.edu/~schmidt/cs892/assignments.html)
Overview of Assignments & Assessments

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

- 40% Quizzes
- 40% Programming projects
- 10% Final exam
- 10% Participation

Participation is roughly 5% attendance & 5% in-class involvement in discussions
Overview of Assignments & Assessments

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

Participation is roughly 5% attendance & 5% in-class involvement in discussions

Don’t expect to get an A in this class if you do not participate!!!!
Setting Up the Android & Java IDE on Android Studio
Installing Eclipse Java/Android Developer Tools

- See developer.android.com/sdk
Installing Eclipse Java/Android Developer Tools

• Installation steps
Installing Eclipse Java/Android Developer Tools

• Installation steps
  • Download & install the Java Standard Edition JDK & JRE 8

Java SE Development Kit 8 Downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, applets, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

See also:
- Java Developer Newsletter (tick the checkbox under Subscription Center > Oracle Technology News)
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine
- JDK MD5 Checksum

Looking for JDK 8 on ARM?
JDK 8 for ARM downloads have moved to the JDK 8 for ARM download page.

Java SE Development Kit 8u25

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

- Accept License Agreement
- Decline License Agreement

<table>
<thead>
<tr>
<th>Product / File Description</th>
<th>File Size</th>
<th>Download</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux x86</td>
<td>135.24 MB</td>
<td>jdk-8u25-linux-i586.rpm</td>
</tr>
<tr>
<td>Linux x86</td>
<td>154.88 MB</td>
<td>jdk-8u25-linux-i586.tar.gz</td>
</tr>
<tr>
<td>Linux x64</td>
<td>135.6 MB</td>
<td>jdk-8u25-linux-x64.rpm</td>
</tr>
</tbody>
</table>

www.oracle.com/technetwork/java/javase/downloads
Installing Eclipse Java/Android Developer Tools

- Installation steps
  - Download & install the Java Standard Edition JDK & JRE 8
  - Download & install Android Studio

[Link to Android Studio website: developer.android.com/sdk]
Add Components to the SDK

- Launch the Android Studio SDK Manager
- Select “Nougat” version of Android (7.0, API 24)

See developer.android.com/studio/intro/update.html
Add Components to the SDK

- Launch the Android Studio Virtual Device Manager
- Create an Android API 24 emulator

[Image: Your Virtual Devices window]

[Link: developer.android.com/tools/devices/managing-avds.html]
Intel HAXM Virtualization Driver

- **Requirements**
  - Intel virtualization extensions (VT, VT-x, vmx)
  - AMD virtualization extensions (AMD-v, SVM) [only supported on Linux]
  - Download an x86 emulator image

- **Windows & Mac OSX**
  - `<sdk>/extras/intel/Hardware_Accelerated_Execution_Manager/IntelHAXM.exe/dmg`

- **Linux**
  - Install KVM & pass “-enable-kvm” flag to emulator when starting

[developer.android.com/tools/devices/emulator.html#acceleration](developer.android.com/tools/devices/emulator.html#acceleration)
Accessing Java & Android Source Code
Accessing Java & Android Source Code

• Android source code is available for browsing android.googlesource.com

android Git repositories

To clone one of these repositories, install git, and run:

```
git clone https://android.googlesource.com/name
```

<table>
<thead>
<tr>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessories/manifest</td>
</tr>
<tr>
<td>device/asus/deb</td>
</tr>
<tr>
<td>device/asus/flo</td>
</tr>
<tr>
<td>device/asus/flo-kernel</td>
</tr>
<tr>
<td>device/asus/grouper</td>
</tr>
<tr>
<td>device/asus/tilapia</td>
</tr>
<tr>
<td>device/common</td>
</tr>
<tr>
<td>device/generic/armv7-a</td>
</tr>
<tr>
<td>device/generic/armv7-a-neon</td>
</tr>
<tr>
<td>device/generic/art</td>
</tr>
<tr>
<td>device/generic/common</td>
</tr>
<tr>
<td>device/generic/goldfish</td>
</tr>
<tr>
<td>device/generic/mini-emulator-armv7-a-neon</td>
</tr>
<tr>
<td>device/generic/mini-emulator-mips</td>
</tr>
<tr>
<td>device/generic/mini-emulator-x86</td>
</tr>
</tbody>
</table>
• Android source code is available
  • For browsing android.googlesource.com
  • For downloading source.android.com

The Android Source Code

Android is an open-source software stack created for a wide array of devices with different form factors. The primary purposes of Android are to create an open software platform available for carriers, OEMs, and developers to make their innovative ideas a reality and to introduce a successful, real-world product that improves the mobile experience for users. We also wanted to make sure there was no central point of failure, where one industry player could restrict or control the innovations of any other. The result is a full, production-quality consumer product with source code open for customization and porting.

Governance Philosophy

Android was originated by a group of companies known as the Open Handset Alliance, led by Google. Today, many companies – both original members of the OHA and others – have invested heavily in Android. These companies have allocated significant engineering resources to improve Android and bring Android devices to market.

The companies that have invested in Android have done so on its merits because we believe an open platform is necessary. Android is intentionally and explicitly an open-source – as opposed to a free software – effort; a group of organizations with shared needs has pooled resources to collaborate on a single implementation of a shared product. The Android philosophy is pragmatic; first and foremost. The objective is a shared product that each contributor can tailor and customize.

Uncontrolled customization can, of course, lead to incompatible implementations. To prevent this, the Android Open Source Project also maintains the Android Compatibility Program, which spells out what it means to be "Android compatible" and what is required of device builders to achieve that status. Anyone can (and will) use the Android source code for any purpose, and we welcome all legitimate uses. However, in order to take part in the shared ecosystem of applications we are building around Android, device builders must participate in the Android Compatibility Program.

The Android Open Source Project is led by Google, who maintains and further develops Android. Although Android consists of multiple subprojects, this is strictly a project management technique. We view and manage Android as a single, holistic software product, not a "distribution", specification, or collection of replaceable parts. Our intent is that device builders port Android to a device; they don't implement a specification or curate a distribution.
Accessing Java & Android Source Code

- Java 8 source code is available
- For Browsing: grepcode.com/file/repository.grepcode.com/java/root/jdk/openjdk/8-b132/java
Accessing Java & Android Source Code

- Java 8 source code is available
  - For Browsing grepcode.com/file/repository.
    grepcode.com/java/root/jdk/openjdk/8-b132/java
  - For downloading jdk8.java.net/download.html
Summary
Summary

- You will get out of this course what you put into it
Summary

• You will get out of this course what you put into it
• Be prepared to work hard

HARD WORK

“Human Felicity is produc'd not so much by great
Pieces of good Fortune that seldom happen, as by little
Advantages that occur every Day” - Benjamin Franklin
Summary

- You will get out of this course what you put into it
- Be prepared to work hard
- Do not miss deadlines...
Summary

- You will get out of this course what you put into it
- Be prepared to work hard
- Do not miss deadlines…
- Participate in discussions in class & on piazza

See piazza.com/vanderbilt/spring2017/cs892/home
Summary

• You will get out of this course what you put into it
• Be prepared to work hard
• Do *not* miss deadlines…
• Participate in discussions in class & on piazza
• No laptops/phones in class unless explicitly allowed

10 POINTS FROM GRYFFINDOR

Failure to comply with this rule will cost you participation points
You will get out of this course what you put into it
Be prepared to work hard
Do not miss deadlines...
Participate in discussions in class & on piazza
No laptops/phones in class unless explicitly allowed
Avail yourself of available resources
Summary

• You will get out of this course what you put into it
• Be prepared to work hard
• Do not miss deadlines…
• Participate in discussions in class & on piazza
• No laptops/phones in class unless explicitly allowed
• Avail yourself of available resources

Please resist the urge to email me directly unless it’s a confidential matter or you’d like to schedule a meeting!
Summary

- You will get out of this course what you put into it
- Be prepared to work hard
- Do not miss deadlines...
- Participate in discussions in class & on piazza
- No laptops/phones in class unless explicitly allowed
- Avail yourself of available resources
- There are abundant opportunities!

Summary

• If there’s an emergency, pay attention to the escape route!

• See engineering.vanderbilt.edu/about/evacuationplans.php

See video at www.youtube.com/watch?v=zXpFznXb_vI starting ~20 seconds