MOOC Organization, Content, & Strategies

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment

MOOC Organization, Content, & Strategies

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Please watch this lesson carefully!

Summary of the Organization & Contents in this MOOC

Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment

Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment

Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

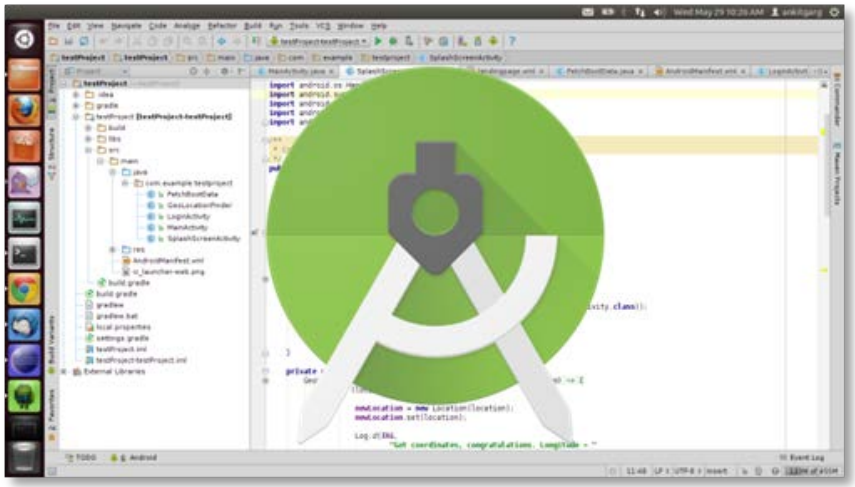
Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment



Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment

Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

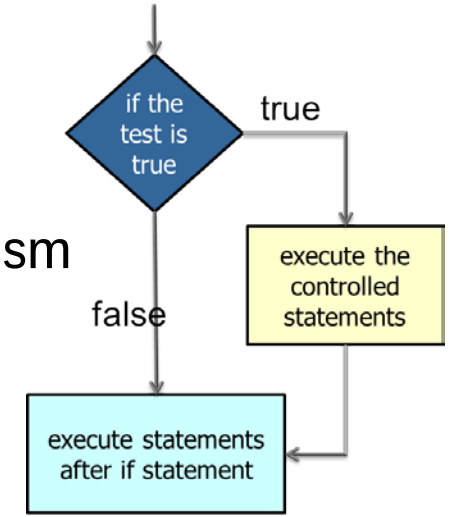
Module 4: Control Flow

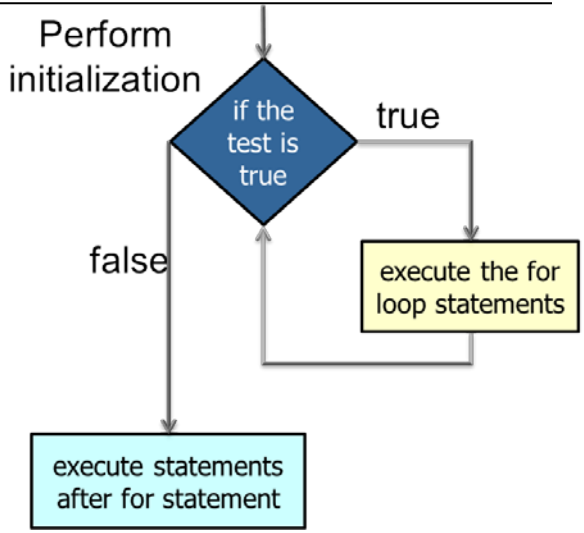
Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment





Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

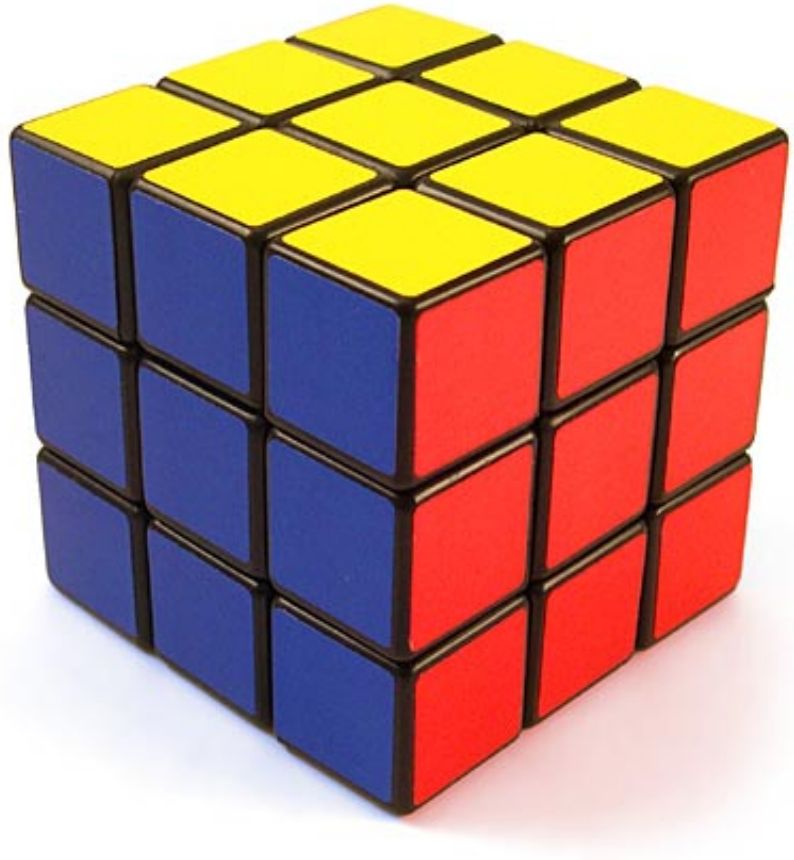
Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment



Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

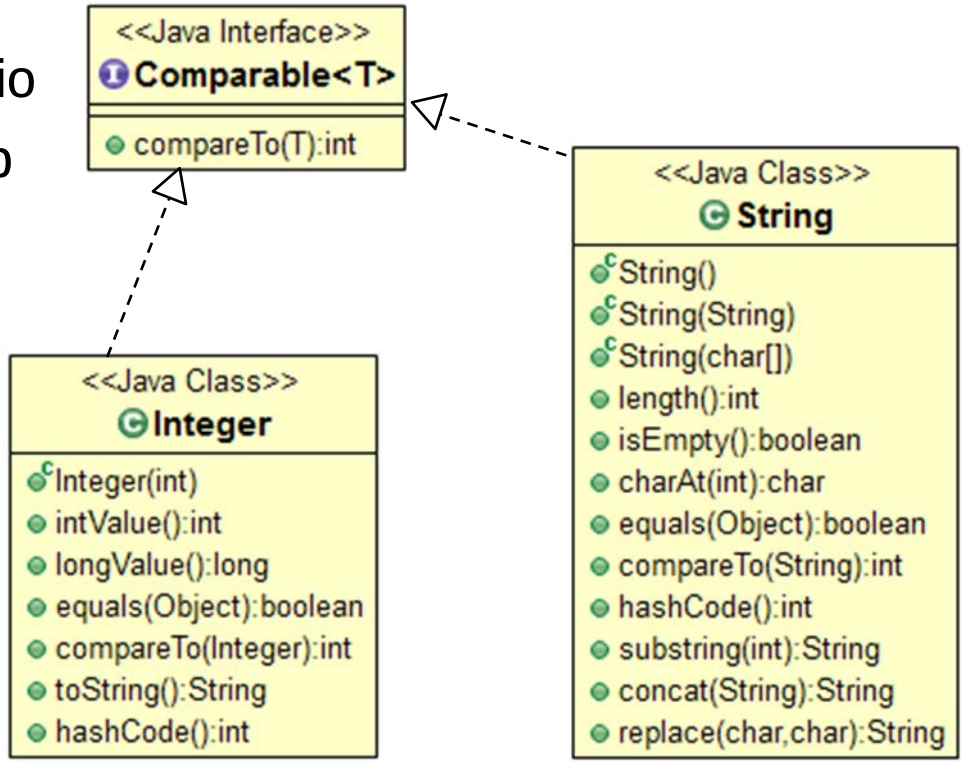
Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment



Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

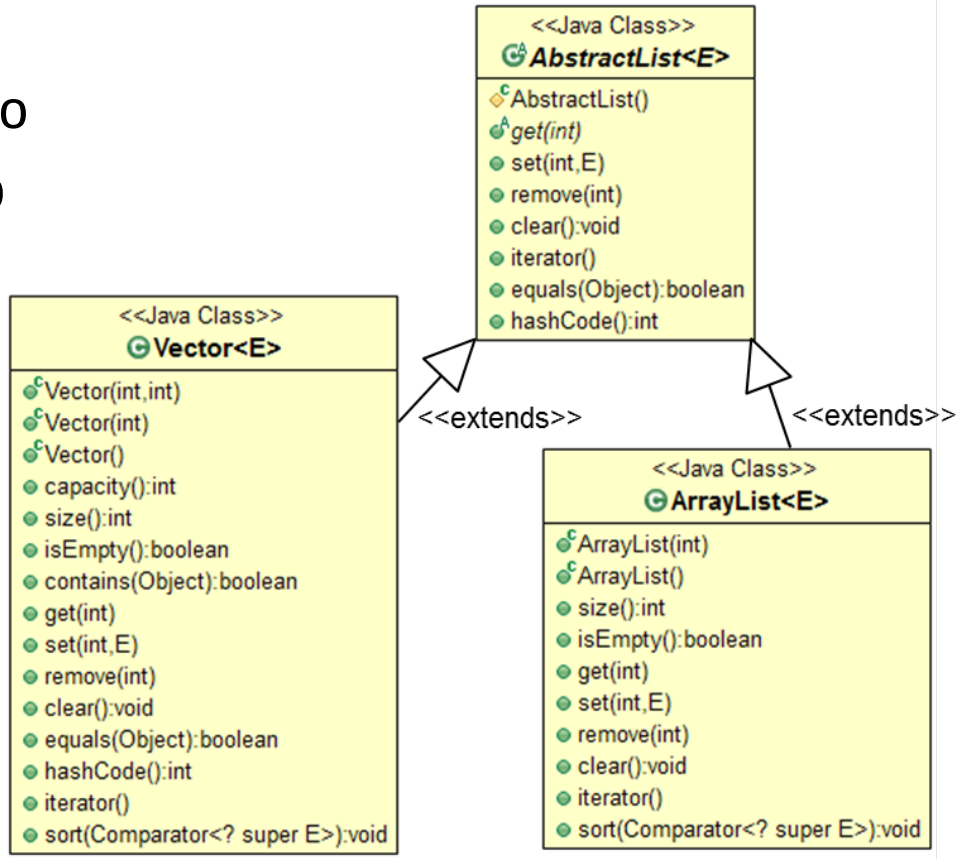
Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment



Summary of the Organization & Contents in this MOOC

Module 1: MOOC Overview

Module 2: Introduction to Android Studio

Module 3: Writing a Simple Android App Using Basic Java Features

Module 4: Control Flow

Module 5: Structured Data

Module 6: Classes & Interfaces

Module 7: Inheritance & Polymorphism

Module 8: Android Calculator App Assignment

Summary of the Organization & Contents in this MOOC

- The modules in this MOOC are structured uniformly

Summary of the Organization & Contents in this MOOC

- The modules in this MOOC are structured uniformly
  - Each Module is composed of Lessons

A lesson covers related topics, such as Java classes, conditional statements, etc.

Summary of the Organization & Contents in this MOOC

- The modules in this MOOC are structured uniformly
  - Each Module is composed of Lessons
  - Each Lesson is presented as a single lecture

Summary of the Organization & Contents in this MOOC

- The modules in this MOOC are structured uniformly
  - Each Module is composed of Lessons
  - Each Lesson is presented as a single lecture
  - Each Lesson is composed of segments

A segment presents an intro, outro, or technical content of a lesson

Summary of the Organization & Contents in this MOOC

- The modules in this MOOC are structured uniformly
  - Each Module is composed of Lessons
  - Each Lesson is presented as a single lecture
  - Each Lesson is composed of segments
  - Quizzes pop up periodically

If you download the lessons & watch them locally the quizzes won't appear!

MOOC Specialization Prerequisites & Expectations

MOOC Specialization Prerequisites & Expectations

- This MOOC Specialization is intended for learners just starting to program...

MOOC Specialization Prerequisites & Expectations

- ... or those switching to Java & Android for the first time

MOOC Specialization Prerequisites & Expectations

- MOOC Specialization topics are based on material we teach at Vanderbilt

CS 279. Software Engineering Project. Students work in teams to specify, design, implement, document, and test a nontrivial software project. The use of CASE (Computer-Assisted Software Engineering) tools is stressed. Prerequisite: CS 278. SPRING. [3]

CS 278. Principles of Software Engineering. The nature of software. The object-oriented paradigm. Software life-cycle models. Requirements, specification, design, implementation, documentation, and testing of software. Object-oriented analysis and design. Software maintenance. Prerequisite: CS 251. FALL. [3]

CS 251. Intermediate Software Design. High quality development and reuse of architectural patterns, design patterns, and software components. Theoretical and practical aspects of developing, documenting, testing, and applying reusable class libraries and object-oriented frameworks using object-oriented and component-based programming languages and tools. Prerequisite: CS 201. FALL, SPRING. [3]

CS 101. Programming and Problem Solving. An intensive introduction to algorithm development and problem solving on the computer. Structured problem definition, top down and modular algorithm design. Running, debugging, and testing programs. Program documentation. FALL, SPRING. [3]

CS 282. Principles of Operating Systems II. Projects involving modification of a current operating system. Lectures on memory management policies, including virtual memory. Protection and sharing of information, including general models for implementation of various degrees of sharing. Resource allocation in general, including deadlock detection and prevention strategies. Introduction to operating system performance measurement, for both efficiency and logical correctness. Two hours lecture and one hour laboratory. Prerequisite: CS 281. SPRING. [3]

CS 281. Principles of Operating Systems I. Resource allocation and control functions of operating systems. Scheduling of processes and processors. Concurrent processes and primitives for their synchronization. Use of parallel processes in designing operating system subsystems. Methods of implementing parallel processes on conventional computers. Virtual memory, paging, protection of shared and non-shared information. Structures of data files in secondary storage. Security issues. Case studies. Prerequisite: CS 231, CS 251. FALL, SPRING. [3]

CS 201. Program Design and Data Structures. Continuation of CS 101. The study of elementary data structures, their associated algorithms and their application in problems; rigorous development of programming techniques and style; design and implementation of programs with multiple modules, using good data structures and good programming style. Prerequisite: CS 101. FALL, SPRING. [3]

See engineering.vanderbilt.edu/eecs

MOOC Specialization Prerequisites & Expectations

- MOOC Specialization topics are based on material we teach at Vanderbilt

CS 279. Software Engineering Project. Students work in teams to specify, design, implement, document, and test a nontrivial software project. The use of CASE (Computer-Assisted Software Engineering) tools is stressed. Prerequisite: CS 27[illegible] SPRING. [3]

CS 278. Principles of Software Engineering. The nature of software. The object-oriented paradigm. Software life-cycle models. Requirements, specification, design, implementation, documentation, and testing of software. Object-oriented analysis and design. Software maintenance. Prerequisite: CS 251. FALL. [3]

CS 251. Intermediate Software Design. High quality development and reuse of architectural patterns, design patterns, and software components. Theoretical and practical aspects of developing, documenting, testing, and applying reusable class libraries and object-oriented frameworks using object-oriented and component-based programming languages and tools. Prerequisite: CS 201. FALL, SPRING. [3]

CS 101. Programming and Problem Solving. An intensive introduction to algorithm development and problem solving on the computer. Structured problem definition, top down and modular algorithm design. Running, debugging, and testing programs. Program documentation. FALL, SPRING. [3]

CS 282. Principles of Operating Systems II. Projects involving modification of a current operating system. Lectures on memory management policies, including virtual memory. Protection and sharing of information, including general models for implementation of various degrees of sharing. Resource allocation in general, including deadlock detection and prevention strategies. Introduction to operating system performance measurement, for both efficiency and logical correctness. Two hours lecture and one hour laboratory. Prerequisite: CS 281. SPRING. [3]

CS 281. Principles of Operating Systems I. Resource allocation and control functions of operating systems. Scheduling of processes and processors. Concurrent processes and primitives for their synchronization. Use of parallel processes in designing operating system subsystems. Methods of implementing parallel processes on conventional computers. Virtual memory, paging, protection of shared and non-shared information. Structures of data files in secondary storage. Security issues. Case studies. Prerequisite: CS 231, CS 251. FALL, SPRING. [3]

CS 201. Program Design and Data Structures. Continuation of CS 101. The study of elementary data structures, their associated algorithms and their application in problems; rigorous development of programming techniques and style; design and implementation of programs with multiple modules, using good data structures and good programming style. Prerequisite: CS 101. FALL, SPRING. [3]

MOOC Specialization Prerequisites & Expectations

- We don’t know what you know or whether you’re prepared or not!

MOOC Specialization Prerequisites & Expectations

- This MOOC Specialization has certain expectations of learners

FAQ

1. **What are the course learning objectives?**

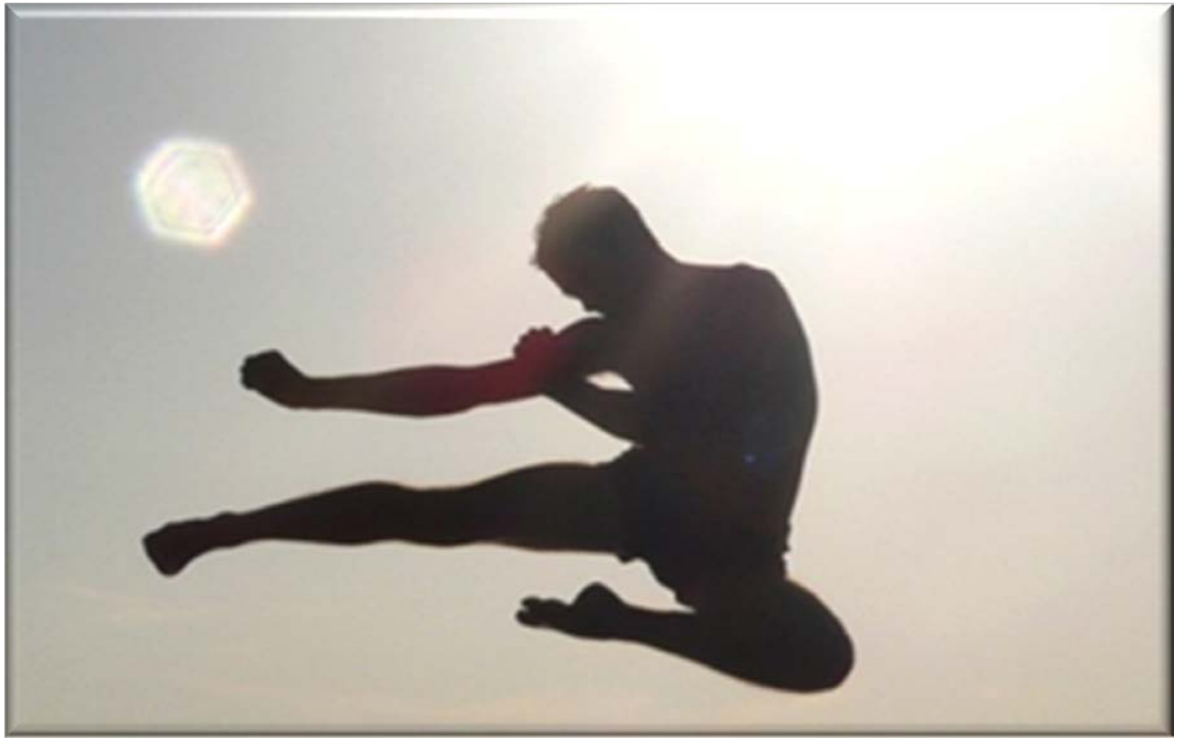
   Upon completing this course, students should be able to:

   - Understand the key object oriented features needed to program Java for Android
   - Know how to install Android Studio and apply it and other tools to develop & debug Android Apps.
   - Know where to find additional sources of information on how to program mobile apps on Android handheld systems.

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

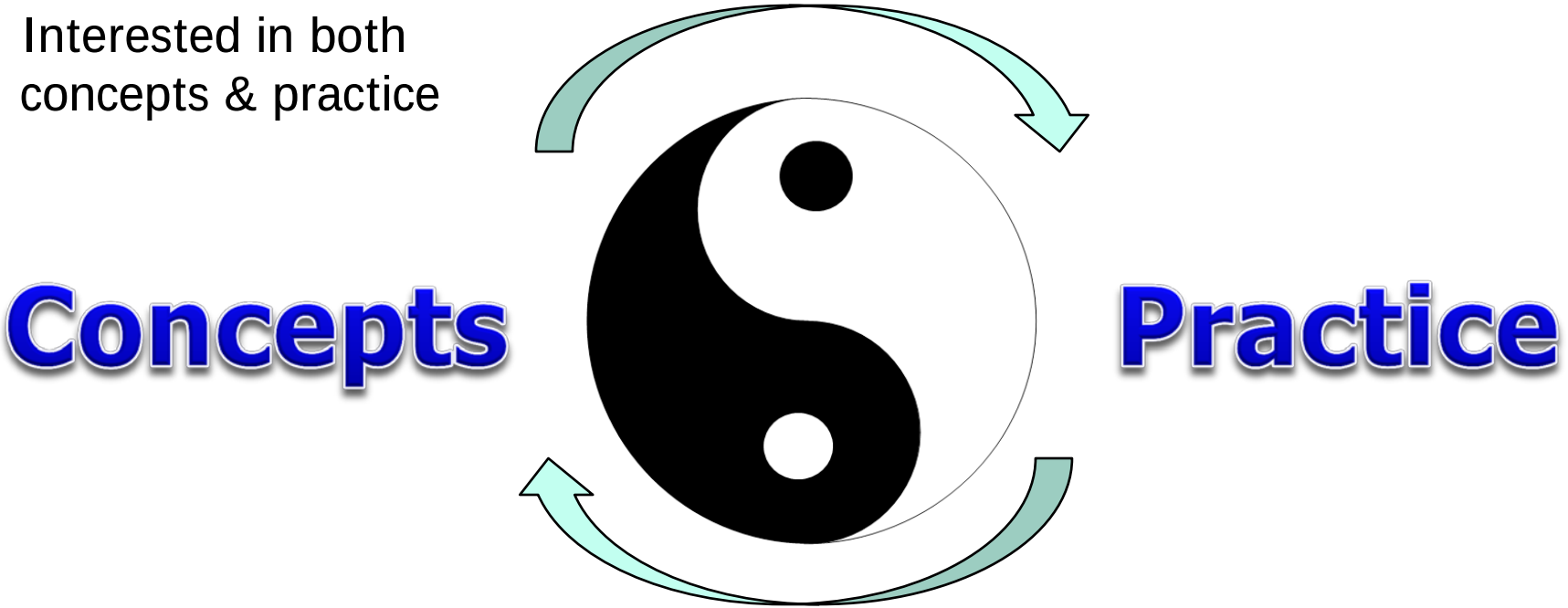
- This MOOC Specialization has certain expectations of learners
  - Self-motivated



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

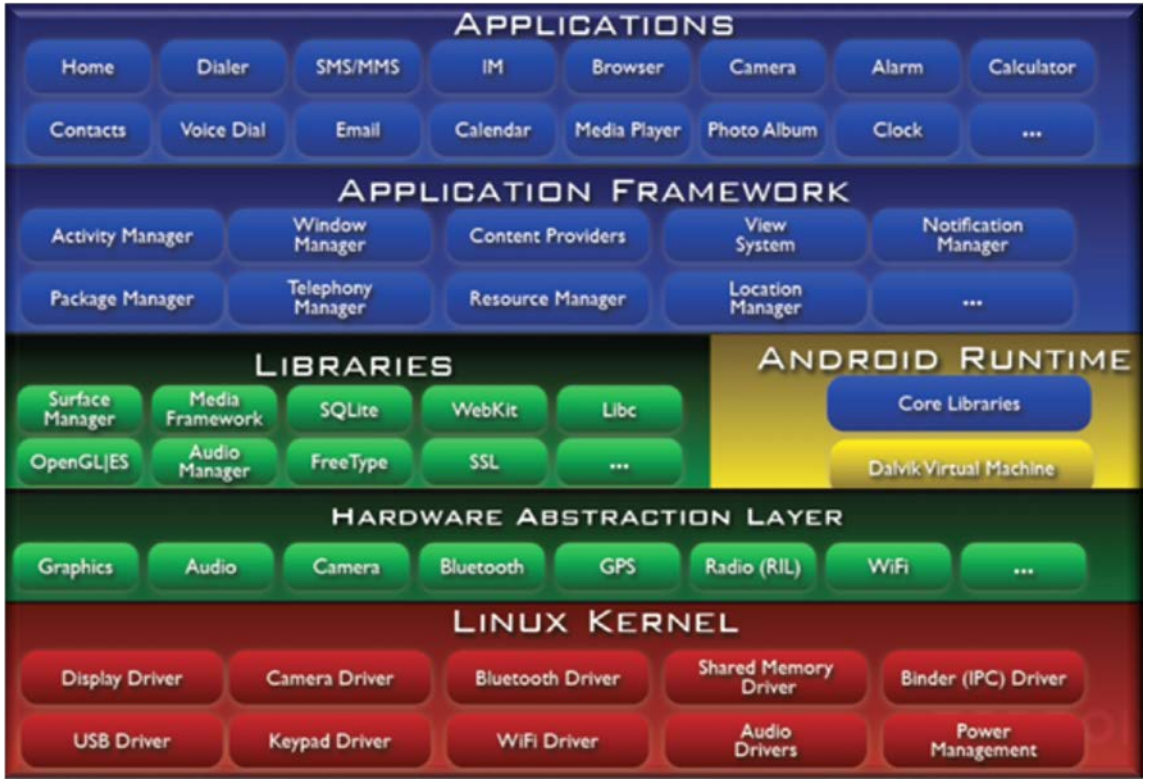
- This MOOC Specialization has certain expectations of learners
  - Self-motivated
  - Interested in both concepts & practice



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

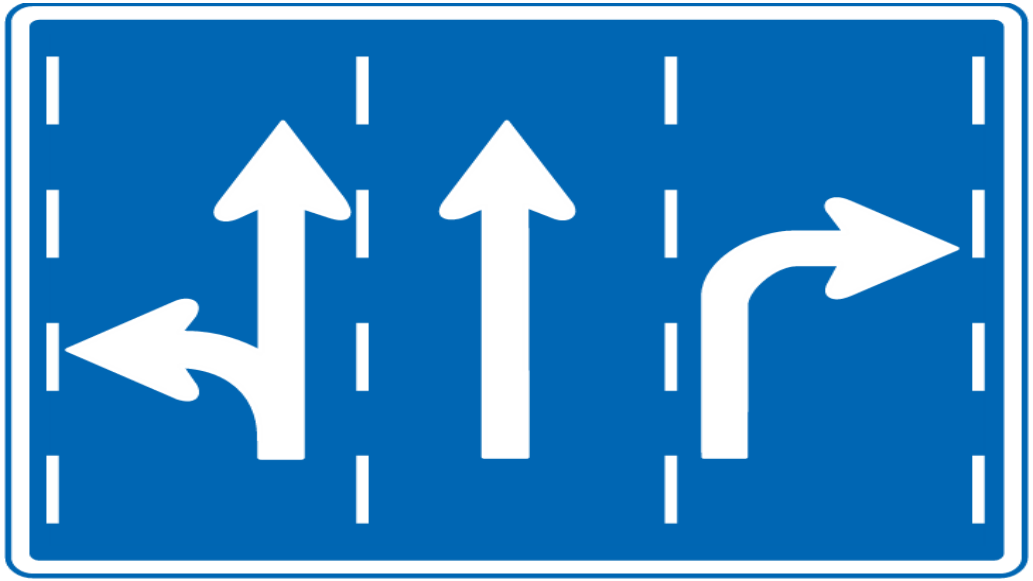
- This MOOC Specialization has certain expectations of learners
  - Self-motivated
  - Interested in both concepts & practice
  - Curious about entire Android software stack



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

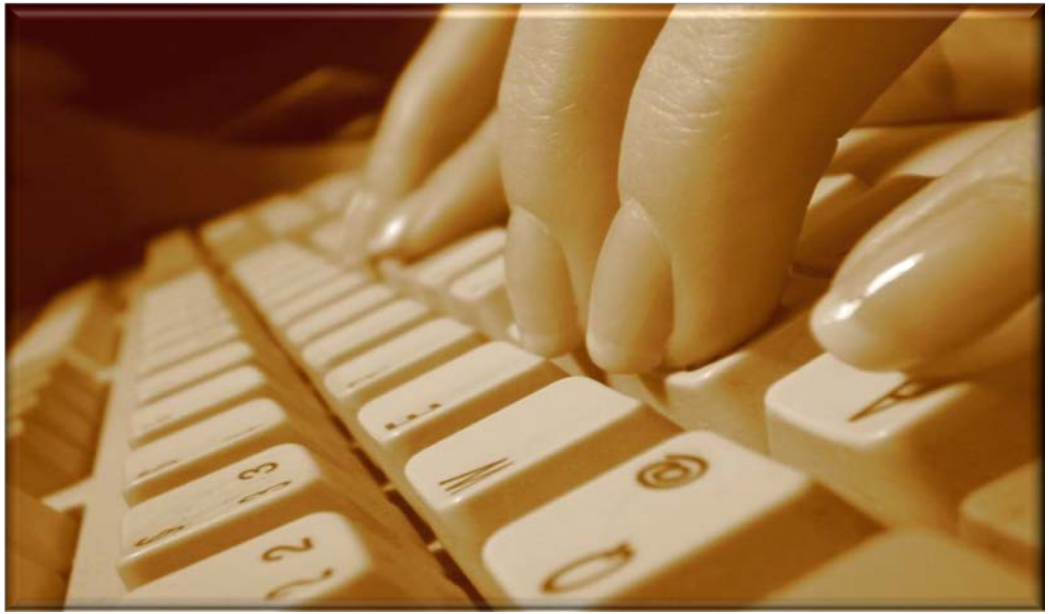
- This MOOC Specialization has certain expectations of learners
  - Self-motivated
  - Interested in both concepts & practice
  - Curious about entire Android software stack
  - Are willing/able to read the FAQ & follow instructions



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

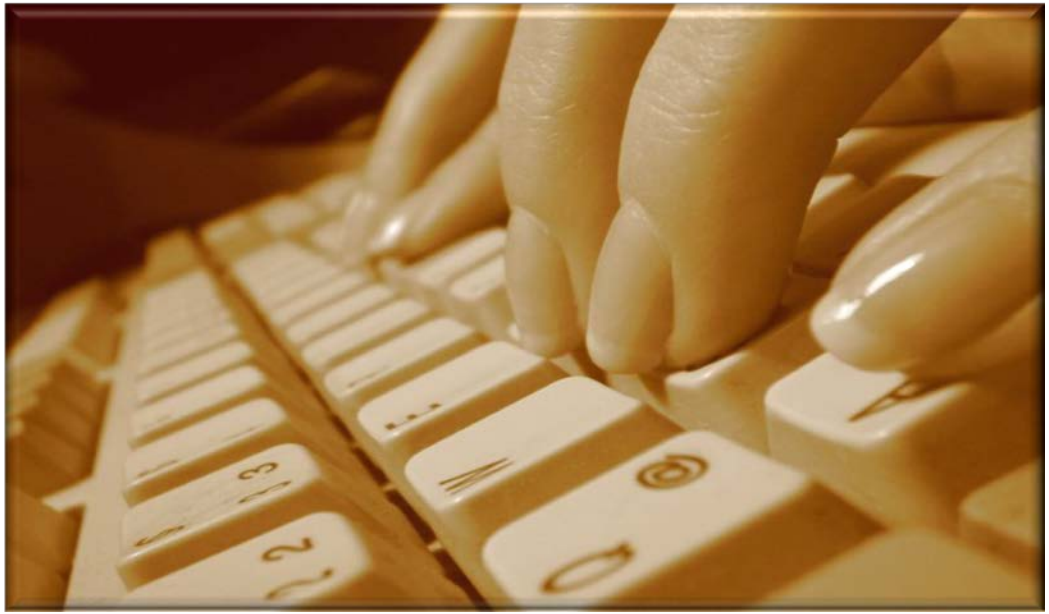
- This MOOC Specialization has certain expectations of learners
  - Self-motivated
  - Interested in both concepts & practice
  - Curious about entire Android software stack
  - Are willing/able to read the FAQ & follow instructions
  - Have computer literacy skills



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

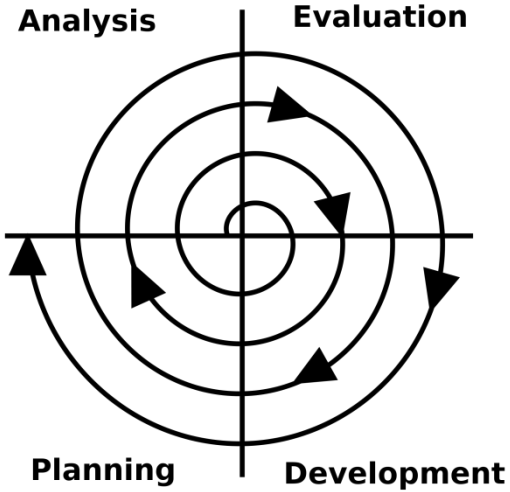
- This MOOC Specialization has certain expectations of learners
  - Self-motivated
  - Interested in both concepts & practice
  - Curious about entire Android software stack
  - Are willing/able to read the FAQ & follow instructions
  - Have computer literacy skills
    - e.g., send/receive emails, browse the web, participate in online discussion forums, & upload/ download files from websites

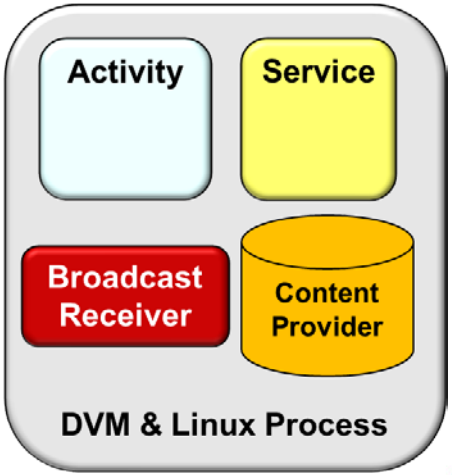


See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#5

MOOC Specialization Prerequisites & Expectations

- After completing this MOOC Specialization we expect you'll know Java for programming core Android components, Android Studio, Git, et al.





Overview of the Assignments & Assessments

Overview of Assignments & Assessments

- Programming assignments should be written in Java using Android Studio

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#23

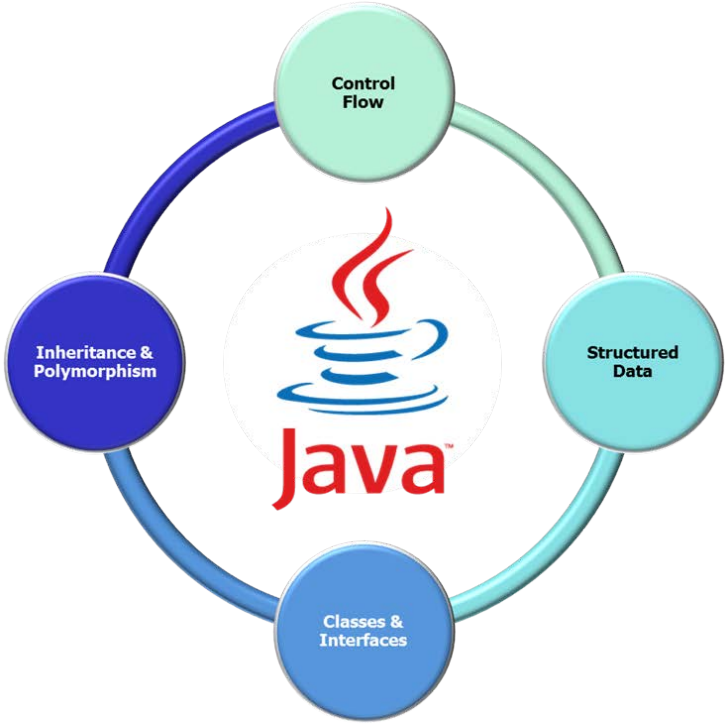
Overview of Assignments & Assessments

- Source code for assignments, examples, Android, & Java is online

See github.com/douglascraigschmidt/Android-App-Development/wiki/Source-Code

Overview of Assignments & Assessments

- The programming assignments provide experience developing apps using a range of Java & Android features



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#10

Overview of Assignments & Assessments

- Stand-alone quizzes will be assessed via auto-grading

Overview of Assignments & Assessments

- Stand-alone quizzes will be assessed via auto-grading
- Programming assignments will be assessed via auto-grading & peer-grading

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#34

Overview of Assignments & Assessments

- Verified Certificates require doing the quizzes & assignments

Verified Certificate

douglascraigschmidt edited this page 7 minutes ago · 1 revision

Coursera offers the Signature Track, which verifies the identity of students. Students in the Signature Track can receive a Verified Certificate if they meet the criteria described below. It's mandatory for students in the Signature Track to successfully achieve a Verified Certificate in all MOOCs in the Specialization to take the Capstone project at the end of the Specialization.

This MOOC consists of lecture videos with integrated quiz questions designed to ensure students understand the material covered in the videos. The estimated time commitment is roughly 8 -- 12 hours per week, depending on background and interest level. In addition to completing the auto-graded weekly quizzes, students will also complete auto-/peer-graded programming assignments. These programming assignments will involve writing concurrent Android applications using its pattern-oriented frameworks written in Java. The final grade for the course will be calculated as follows:

See github.com/douglascraigschmidt/Android-App-Development/wiki/Verified-Certificate

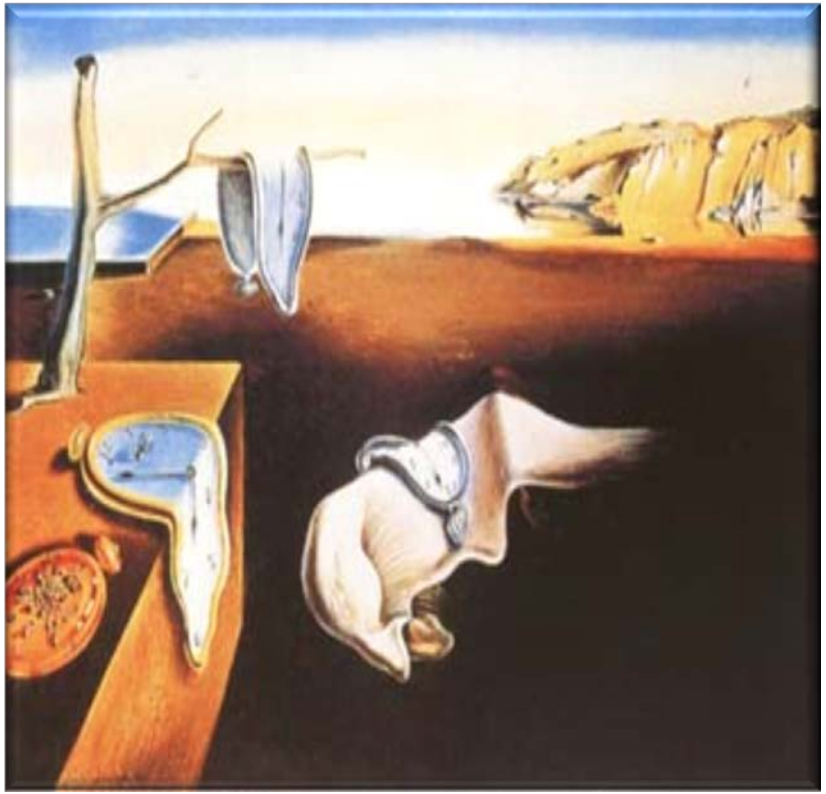
Overview of Assignments & Assessments

- Learners are expected to devote time & effort to these MOOCs each week

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#38

Overview of Assignments & Assessments

- Learners are expected to devote time & effort to these MOOCs each week
  - However, they have different learning styles, backgrounds, motivations, aptitudes, & time commitments



See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#38

Overview of Assignments & Assessments

- Learners are expected to devote time & effort to these MOOCs each week
  - However, they have different learnin styles, backgrounds, motivations, aptitudes, & time commitments
  - We can't precisely estimate the amount of time for this MOOC

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#38

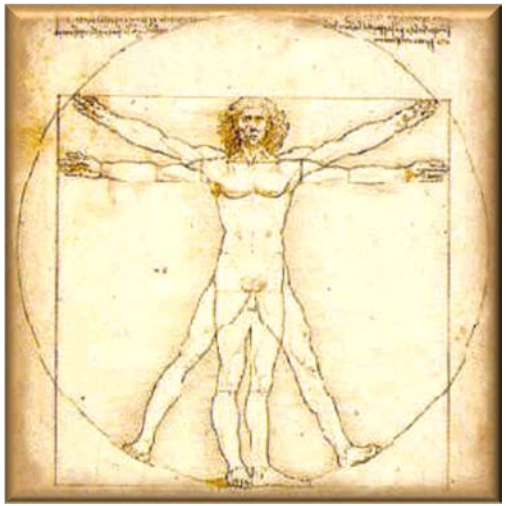
Overview of Assignments & Assessments

- The more time & effort you expend the more benefits you'll receive

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#38

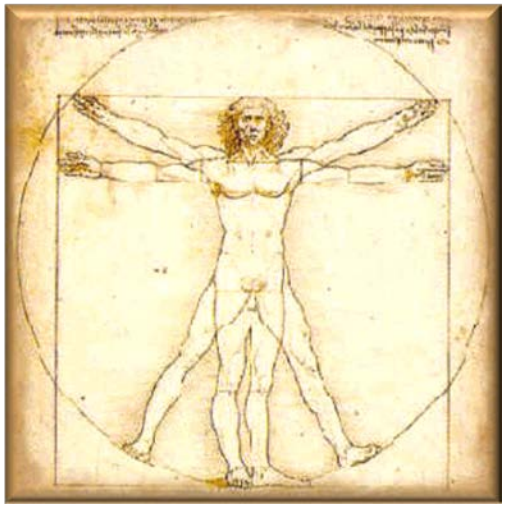
Overview of Assignments & Assessments

- You can take this MOOC solely to expand your knowledge of Java & Android



Overview of Assignments & Assessments

- You can take this MOOC solely to expand your knowledge of Java & Android
  - i.e., without submitting quizzes or programming assignment solutions



Strategies for Learning All this Material

Strategies for Learning All this Material

- This MOOC Specialization covers a lot of material that requires careful thinking & practice to master

Strategies for Learning All this Material

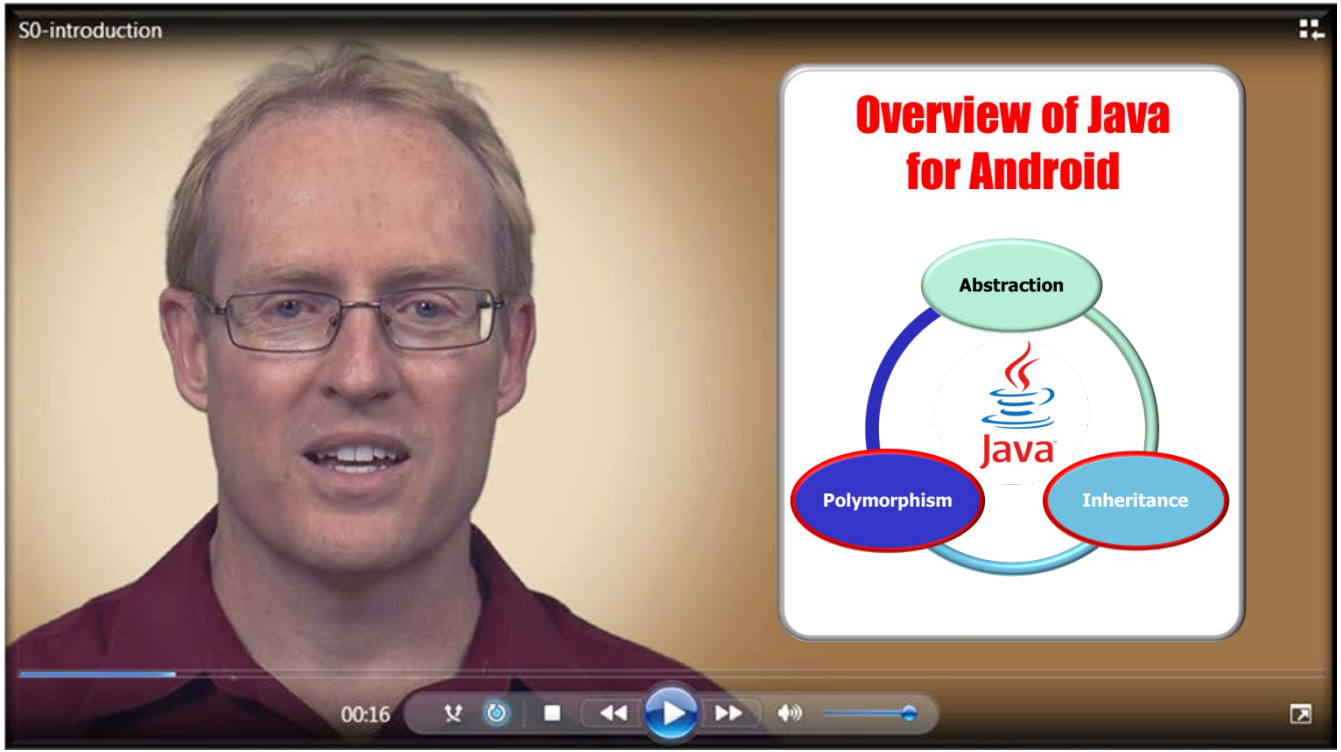
Strategies for Learning All this Material

- Watch the lessons carefully

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#6

Strategies for Learning All this Material

- Watch the lessons carefully

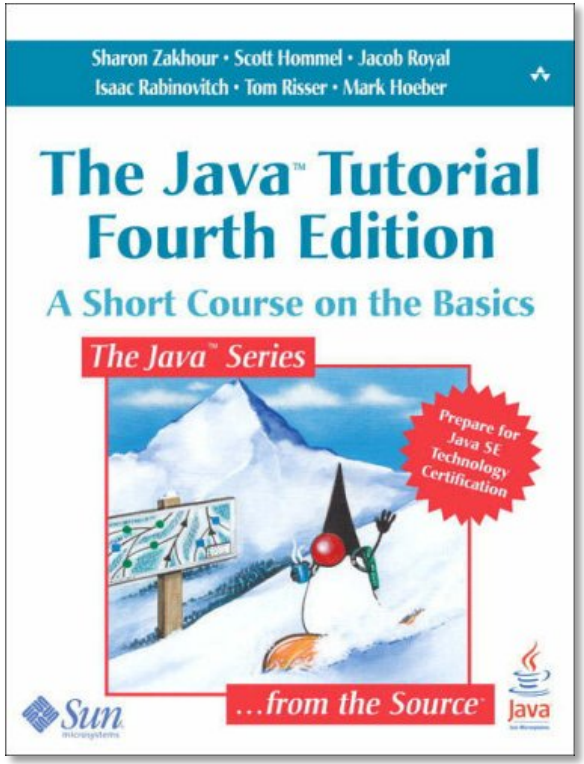


Consider watch some lessons multiple times & at different speeds!

Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources

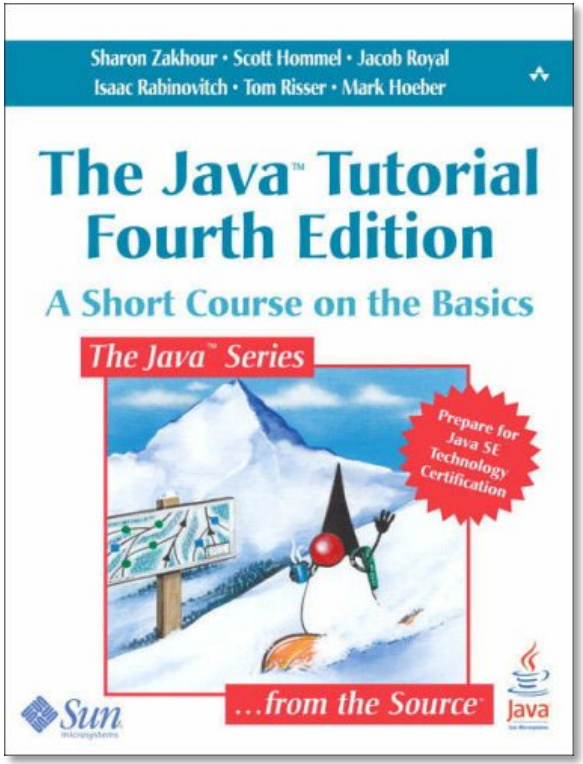
java.util.concurrent	Utility classes commonly useful in concurrent programming.
java.util.concurrent.atomic	A small toolkit of classes that support lock-free thread-safe programming on single variables.
java.util.concurrent.locks	Interfaces and classes providing a framework for locking and waiting for conditions that is distinct from built-in synchronization and monitors.



Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
  - e.g., articles, tutorials, source code, documentation, etc.

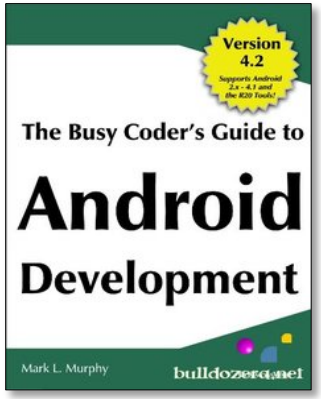
java.util.concurrent	Utility classes commonly useful in concurrent programming.
java.util.concurrent.atomic	A small toolkit of classes that support lock-free thread-safe programming on single variables.
java.util.concurrent.locks	Interfaces and classes providing a framework for locking and waiting for conditions that is distinct from built-in synchronization and monitors.

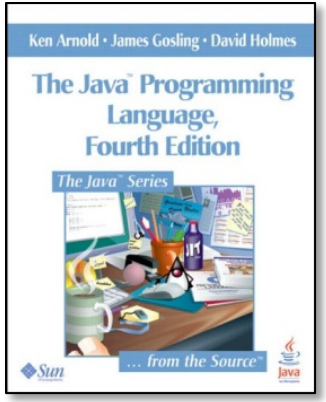


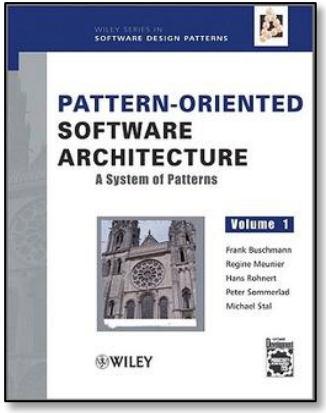
See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ#8

Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
- Read suggested books







See github.com/douglascraigschmidt/Android-App-Development/wiki/Suggested-Reading

Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
- Read suggested books
- Watch online videos

Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
- Read suggested books
- Watch online videos
  - e.g., material from other MOOCs & courses at Vanderbilt

Digital Learning Offerings

Douglas C. Schmidt
(d.schmidt@vanderbilt.edu)
Associate Chair of Computer Science and Engineering,
Professor of Computer Science, and Senior Researcher
in the Institute for Software Integrated Systems (ISIS)
at Vanderbilt University

Coursera MOOCs on Pattern-Oriented Software Architecture (POSA)

- Mobile Cloud Computing with Android Specialization
- Spring 2015 Offering of Programming Mobile Services for Android Handheld Systems: Concurrency
- Spring 2015 Offering of Programming Mobile Services for Android Handheld Systems: Communication
- Spring 2014 Offering of Pattern-Oriented Software Architecture: Programming Mobile Services for Android Handheld Systems
- Spring 2013 Offering of Pattern-Oriented Software Architectures for Concurrent and Networked Software

Vanderbilt University Courses

- Playlist from my YouTube Channel videos from CS 251: Intermediate Software Design with Java
- Playlist from my YouTube Channel videos from CS 282: Concurrent Java Network Programming in Android
- Playlist from my YouTube Channel videos from CS 251: Intermediate Software Design with C++
- Playlist from my YouTube Channel videos from CS 282: Systems Programming for Android

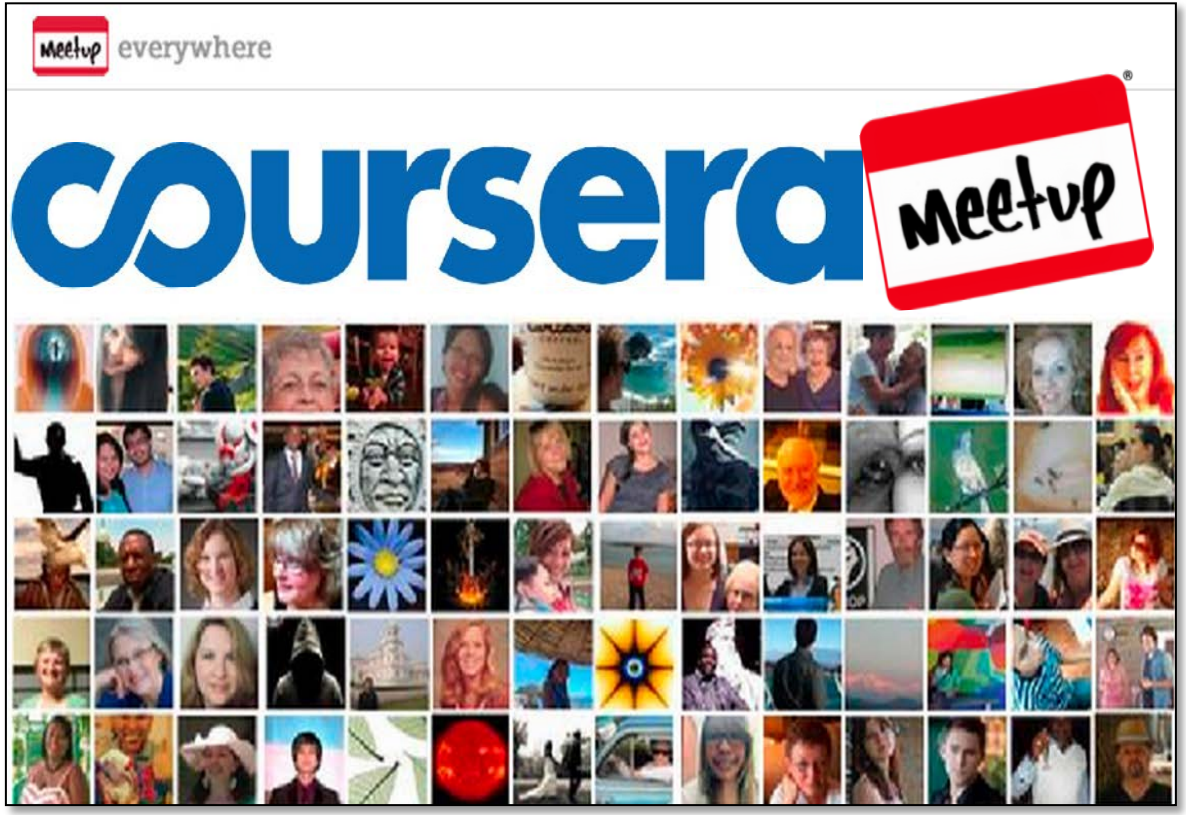
Pearson LiveLessons Courses

- Design Patterns in Java
- Concurrent Programming in Java

See www.dre.vanderbilt.edu/~schmidt/DigitalLearning

Strategies for Learning All this Material

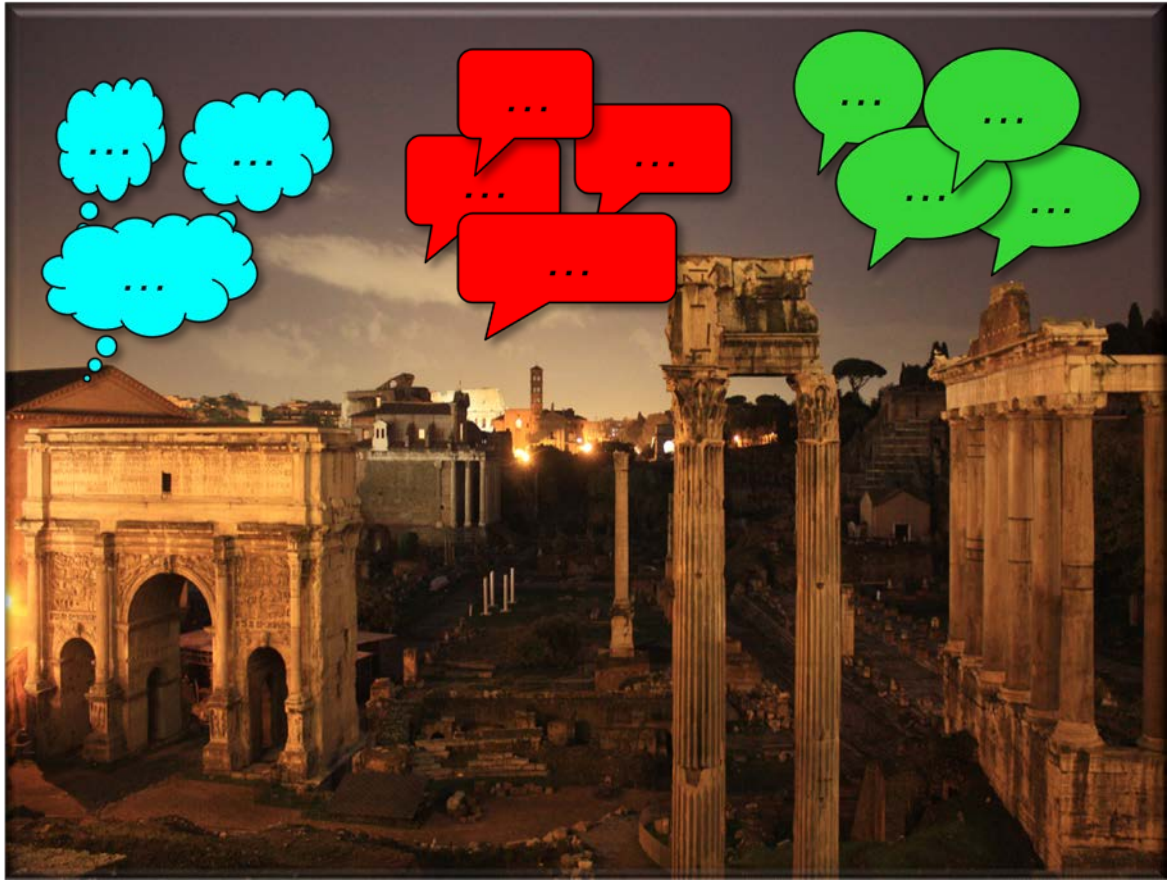
- Watch the lessons carefully
- Explore online resources
- Read suggested books
- Watch online videos
- Join a “meetup group”



See www.meetup.com/coursera

Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
- Read suggested books
- Watch online videos
- Join a “meetup group”
- Participate in online discussion forums



Strategies for Learning All this Material

- Watch the lessons carefully
- Explore online resources
- Read suggested books
- Watch online videos
- Join a “meetup group”
- Participate in online discussion forums
  - Please be respectful of fellow students & staff

See github.com/douglascraigschmidt/POSA-15/wiki/POSA-15-FAQ#9

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

Learning Objectives

- Understand the MOOC's structure & topics covered
- Recognize what's expected of you to complete the MOOC successfully
- Know where to find more information on developing Android apps in Java

MOOC Organization, Content, & Strategies

FAQ

douglascraigschmidt edited this page 2 days ago · 13 revisions

Table of Contents

See github.com/douglascraigschmidt/Android-App-Development/wiki/FAQ