MOBILE IOS DEVELOPER CERTIFICATE OF COMPLETION
COURSE EXAMPLE SYLLABUS
Comprehensive Curriculum

Curriculum Includes Job-readiness/Soft Skills/Health & Safety

All lesson plans and syllabi are subject to change based upon Diploma and Certificate of Completion choices

Curriculum Description

The Java course is an introduction to software engineering, using the Java™ programming language. It covers concepts useful to students and will teach the fundamentals of Java. The focus is on developing high quality, working software that solves real problems. The course is designed for students with some programming experience, but if you have none and are motivated you will do fine. This course will combine lecture and practical applications and modalities of training.

The JavaScript gets you started with an introduction to JavaScript. We assume that you're new to the language, so it gets you started with basic functionality such as creating functions, creating variables, and calling these lines of code from your standard HTML pages. We talk about events and triggers for custom event handling. We also discuss pattern matching, searching for text within a page, flow control and the document object model (DOM).
We start off with the basics and move on to more complex functionality such as arrays and objects. We then discuss how to script common elements with JavaScript such as forms and tables. At the very end, we discuss major libraries such as Ajax, which allows you to make asynchronous calls to server-side scripts without reloading the web page on the server. Whether you're just getting started in web design or want to learn how to code JavaScript, this course is for you.

The Angular JS course is a structural framework for creating dynamic web applications. HTML is a great declarative language for static pages. It does not contain much for creating a dynamic application. So Angular will be filling that gap. Agular's data binding and dependency injection eliminate much of the code than we would write. The best part is that it all happens in the browser by making it an ideal partner with any server technology.

The React Course allows you to learn all the aspects of React and React Native. The class will start with basics such as learning how to compose components which are the foundations of React. During this course, students will build two fully working apps: a desktop application using React, and an iOS and Android app using React Native.

Curriculum Objectives

After completing this curriculum, you will be able to:

- Understand Components
- Learn how to launch an IOS project
- Learn how to use the hardware and software
- Front and Back-end development
- Create concepts and component composition
- Pass data between components
- How to unit test
- Reduce the amount of code you write to build rich user interface applications
- Increase the reliability and maintainability of UI by using data binding
- Retrieve data from back-end server, manipulate it and display it with ease.
- Modularize your code with the custom services and directives
- Providing two-way binding of data.
- Create Single Page Applications (SPA)
- Summarize variable naming rules and JavaScript data types
- Identify expressions and operators
- Summarize flow control
- Demonstrate objects and arrays usage
- Define functions and methods
- Define constructors and inheritance
- Demonstrate usage of pattern matching with regular expressions
- Describe the document object model (DOM)
- How to get input and output
- Summarize managing web page styles using JavaScript and CSS
- Demonstrate handling web page events
- Demonstrate how to script tables and script forms
- Summarize Ajax and its use
- Mastery of lesson content

Prerequisites
To successfully engage and complete the IOS mobile Developer course, you will need to have working knowledge of the computer, both hardware and software components. Some previous experience with the developer language is preferred but not necessary. Ability to problem solve, and have adequate time for class preparation, course work and homework as necessary. This course is jampacked and rigorous, so the student should be well prepared and committed to the program.

Instructional Methods
The content that will be used in this course are lecture and practical demonstration. All courses within the program are instructor-led and will have adequate hands-on interactive labs for real-world practical application techniques.

Week 1
Introduction to Curriculum/ Instructor
Meeting Places/Times/Contact Info
Icebreaking- Students/Syllabus Distribution/Assessment/Email Setup

Chapter 1: Java Language/ What Should I Know
Module A: Fundamentals of Java- what should I know
Module B: Java Class Design- Interfaces
Module C: Java Class Design- Abstract Classes
Module D: Exceptions and Assertions

Chapter 2: Data Structures: Generics and Collections
Module A: Generics
Module B: Collections - Part 1
Module C: Collections - Part 2
Module D: Sorting and Searching

Week 2

Chapter 3: Strings, Regular Expressions, and Recursion
Module A: String Processing
Module B: Use Regular Expressions
Module C: Recursions

Chapter 4: Input and Output
Module A: Basics of Input and Output
Module B: Input and Output Fundamentals
Module C: Deploying an application

Week 3

Chapter 5: JDBC
Module A: JDBC Introduction
Module B: JDBC Basics

Chapter 6: Java Memory and the JVM
Module A: Introduction to JVM Architecture
Module B: Java memory structure

Week 4

Chapter 7: Class File and the JDK
Module A: JDK Tools
Module B: Class File

Chapter 8: Bytecode and Class Loader
Module A: Java Bytecode
Module B: Class Loader

Week 5

Chapter 9: JavaScript Language and the benefits of the language
Module A: JavaScript introduction

Chapter 10: Language Syntax
Module A: JavaScript language syntax
Module B: How to use the program level
Module C: Understanding the arrays (operators, Variable declaration, Control Statements, error handling, Function Declaration)

Week 6

Chapter 11: Built in Functions
Module A: Built in Functions in JavaScript
Module B: How to use the functions
Module C: Standard date and time functions

Chapter 12: HTML forms
Module A: HTML document object model
Module B: Working with HTML form and its elements

Week 7

Chapter 13: HTML DOM
Module A: HTML document object model
Module B: Working with HTML form and its elements
Module C: Other document object model

Week 8

Chapter 14: Cookies
Module A: Working with cookies

Chapter 15: Working with Objects and Classes
Module A: Working with Objects
Module B: Call method in JavaScript
Module C: Inheritance in JavaScript using prototype

Week 9

Chapter 16: ES61 Typescript, Angular-CU, Angular Components
Module A: Module system/ classes/ variable declaration/arrow functions/template strings
Module B: Type safety, inference and intelligence
Module C: Interfaces, Decorators
Module D: Creating new projects, project settings, bootstrapping, building and serving, component-based architecture, angular building blocks, generating project elements, root angular module
Module E: Components definition, component types, template syntax, data property and event binding, using directives and pipes, inputs, outputs, component style
Module F: Data projection, building a wrapper component, querying view and children, dynamic components

Week 10

Chapter 17: Providers, Dependency Injection, Observables
Module A: Understanding the role of the provider, the injector tree, creating and using a class provider service, other provider strategies, configuring providers
Module B: Subscription, RxJS Operators, Creating Subjects and Observables
Module C: Http requests (GET, PUT, POST), configuring headers, interceptors, progress events

Week 11

Chapter 18: Angular Router
Module A: Setting up the router, navigation, child routes, routing params, lazy loading, guards, and hooks
Module B: Angular Modules, root module vs, feature module, module definition and configuration, directives (type and built-in), writing your own directives, pipes (sync and async, built-in, writing your own)

Week 12

Chapter 19: Angular Forms and state management with ngRX
Module A: Template-driven, reactive, Form Builder, Form Validation, custom and async validators
Module B: Understanding Redux architecture, ngRx Store, Actions & Reducers, Middleware, Effects and Facades, Action Splitters

Week 13

Chapter 20: Why React
Module A: Identify why React was built, use compositions and build complex functions from simple ones, leverage declarative code to express logic without control flow, recognize that React is just JavaScript
Module B: Rendering UI with React (Use create-react-app to create application and reusable, focused Class components with compositions, leverage JSX to describe U)

Chapter 21: State Management, Render UI with External Data, Manage App location with React Router
Module A: Manage state in applications, use props for data passing, and components, key words, access component data and properties, update state with set State, use prop Types to type check and debug components, use controlled components to manage input form elements
Module B: Conceptualize the lifecycle of a component, use Reacts component Did Mount lifecycle hook for HTTP requests
Module C: Use React Router to add different applications, use state to dynamically render a different page, use React Routers link and route component

Week 14

Chapter 22: Managing State, UI + Redux, Redux Middleware
Module A: Recognize how state predictability improves applications, create a store to manage an applications state, leverage store API, get State, dispatch, and subscribe, create actions and action creators that describe state changes, create reducers that return state, use reducer composition to handle independent parts of state
Module B: Combine Redux with user interface, build intuition for when to use Redux
Module C: Identify the benefits of implementing middleware in applications, identify the role of middleware within the Redux cycle, apply middleware to a Redux application, build your own Redux middleware

Week 15

Chapter 23: Redux with React and Asynchronous Redux
Module A: Combine Redux with the popular React library
Module B: Identify when to use component state vs. Redux state
Module C: Learn the pitfall of asynchronous requests in Redux, Leverage Thunk middleware to support asynchronous requests, fetch data from a remote APi

Week 16

Chapter 24: React-Redux and Real World Redux

Module A: Install the react-redux bindings, leverage react-redux bindings to extend app functionality, use the provider to pass a store to component trees, use connect to access store context set by the provider

Module B: Build a complex, real-world application with Tyler, add Redux to an application scaffolded with Create React App, normalize state shape to keep application logic simple with scale

Review/Exam Prep

Our scheduled class time will be spent preparing for the final exam for all the courses taught (Java, JavaScript, Angular JS, React).

Soft skills, Exit Interviews

Scheduled class time will be spent conducting exit interviews, filling out exit surveys, engaging in soft skills job interview techniques, resume writing, proper dress for interview, etc.), and final thoughts.