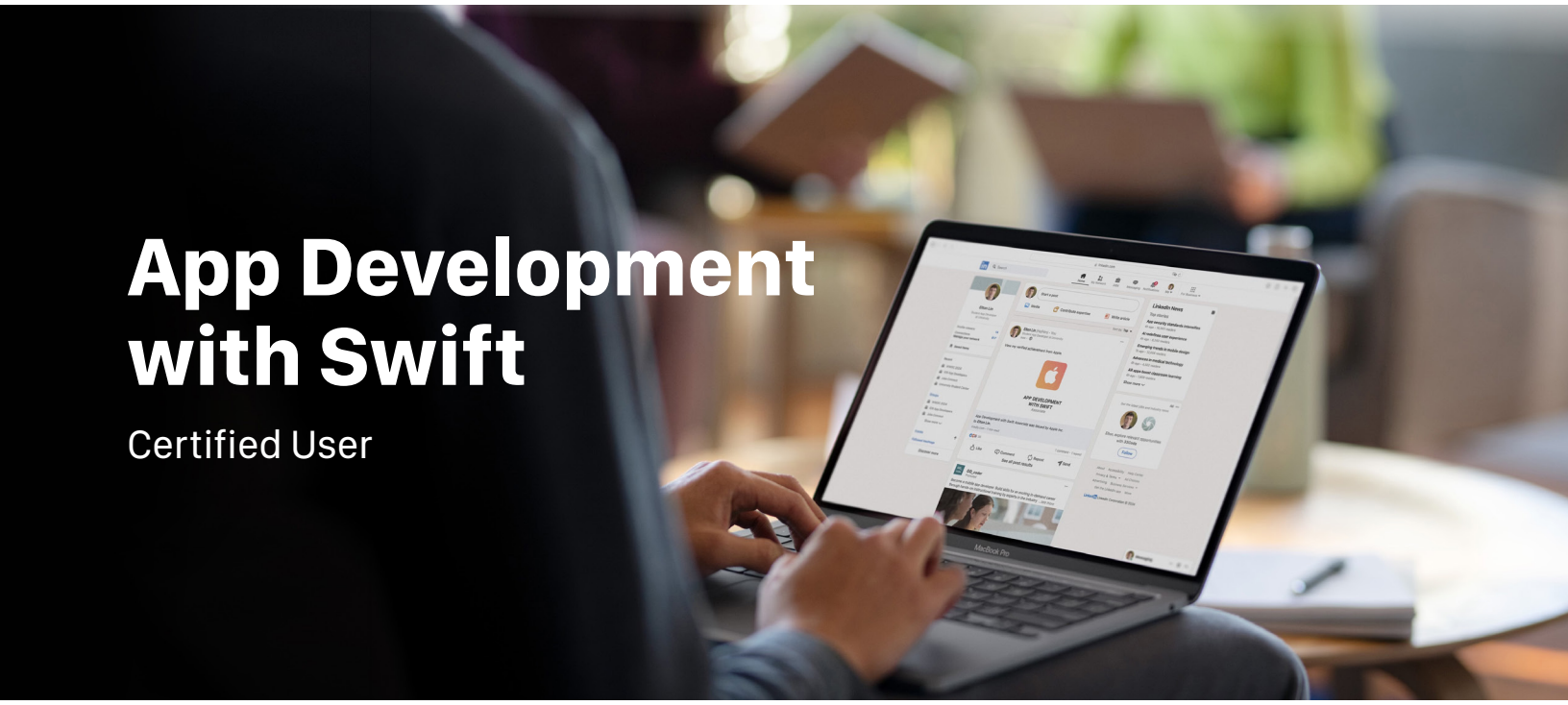
CERTIPORT
A PEARSON VUE BUSINESS



App Development with Swift

Certified User

Objective Domains

Earning the App Development with Swift Certified User certification demonstrates knowledge of common concepts and practices that Swift developers use while building basic fluency in Xcode source and UI editors. Students will be able to create iOS apps that adhere to standard practices, including the use of stock UI elements, layout techniques, and common navigation interfaces.

Xcode Developer Tools

1.1. Identify and use the features of the Xcode interface

- 1.1.1. Navigate Xcode
- 1.1.2. Create and modify views with Interface Builder
- 1.1.3. Demonstrate how to access documentation and help

1.2. Demonstrate how to build and run an app

- 1.2.1. on the iOS simulator
- 1.2.2. on the iOS device

1.3. Use debugging techniques including, but not limited to, breakpoints, watchpoints, and logging to resolve errors

- 1.3.1. Set breakpoints and step through code line by line

Swift Programming Language

2.1. Declare and use basic Swift types

- 2.1.1. Describe and use data types and operators
- 2.1.2. Demonstrate the use of type casting in both safe and unsafe ways
- 2.1.3. Demonstrate when to use constants and variables
- 2.1.4. Interpret and use basic types

2.2. Manage data using collection types

- 2.2.1. Arrays
- 2.2.2. Dictionaries

2.3. Know how and when to apply control flow and loops

- 2.3.1. Use logical operators
- 2.3.2. Use Guard
- 2.3.3. Use range operators

APP DEVELOPMENT
WITH SWIFT
Certified User

CERTIPORT
A PEARSON VUE BUSINESS

App Development with Swift

Certified User

Swift Programming Language (Continued)

2.4. Use functions

- 2.4.1 Organize and structure code
- 2.4.2 Create and call a function
- 2.4.3 Demonstrate how to use a function's return value
- 2.4.4 Customize internal, external, and anonymous naming of parameters in functions
- 2.4.5 Implement default parameter values

2.5. Demonstrate proper use of structs, classes

- 2.5.1. Define and use properties and methods
- 2.5.2. Differentiate between structures and classes
- 2.5.3. Differentiate between various initializers
- 2.5.4. Define and use property observers

2.6. Demonstrate the use of Optional types

- 2.6.1. Demonstrate how to unwrap Optionals safely
- 2.6.2. Apply Optional binding and Optional chaining (including but not limited to if let, guard let)

2.7. Evaluate variable scope and shadowing

View Building with SwiftUI

3.1. Position and/or layout a single SwiftUI View with standard Views and modifiers

3.2. Create multiple Views to implement app logic

3.3. Use List Views to iterate through collections

3.4. Extract Subviews to simplify the structure of an overlarge View

3.5. Create a multi-view app with navigation Stacks, Links, and/or Sheets

3.6. Use @State, @Binding, @Environment, and/or Observable to share data between Views