Blueprint for a 4-Day Intensive TypeScript Course

Course Title: TypeScript Fundamentals in 4 Days

Instructor: Joseph Opio

Course Description

This intensive four-day workshop provides a comprehensive introduction to TypeScript, a powerful typed superset of JavaScript. Over four one-hour lessons, participants will transition from foundational concepts to more advanced features, enabling them to write more robust, scalable, and maintainable code. The course is designed for developers with a working knowledge of JavaScript who want to leverage the benefits of a statically typed language. Each session will include theoretical explanations and practical examples to solidify understanding.

Prerequisites

- Solid understanding of JavaScript fundamentals (variables, functions, objects, arrays, etc.).
- A code editor (e.g., Visual Studio Code) installed.
- Node.js and npm (or yarn) installed to set up a TypeScript environment.

Course Objectives

By the end of this four-day course, participants will be able to:

- Understand the core concepts and benefits of TypeScript.
- Set up a TypeScript development environment.
- Utilize basic and complex types to create strongly typed variables and functions.
- Work with interfaces and classes to build structured and object-oriented code.
- Implement advanced TypeScript features like generics and utility types.

Daily Lesson Plan

Day 1: Introduction to TypeScript and Basic Types

Objective: To understand the fundamentals of TypeScript, set up the development environment, and learn how to use basic types.

Introduction to TypeScript (15 minutes)

- What is TypeScript and why use it?
- The relationship between TypeScript and JavaScript.
- Overview of the TypeScript compiler (tsc).
- **Activity:** Setting up a basic TypeScript project.

Basic Types and Type Annotations (30 minutes)

- Number, String, Boolean, Null, Undefined.
- The any and unknown types.
- Type Inference: Letting TypeScript figure out the types.
- Working with Arrays and Tuples.
- **Hands-on:** Declaring variables with different basic types and creating typed arrays.

Q&A and Wrap-up (15 minutes)

- Review of key concepts.
- Preview of Day 2.

Day 2: Functions, Objects, and Interfaces

Objective: To learn how to apply types to functions and structure data with objects and interfaces.

Typing Functions (20 minutes)

- Adding types to function parameters and return values.
- Optional and default parameters.
- Function overloading.
- **Hands-on:** Creating and typing various functions.

Working with Objects (15 minutes)

- Defining object types.
- Readonly properties.
- **Activity:** Creating typed objects.

Interfaces and Type Aliases (25 minutes)

- Defining custom types with `type` and `interface`.
- Extending interfaces.
- Difference between `type` and `interface`.
- **Hands-on:** Building a simple application structure with interfaces.

*Q&A and Wrap-up (5 minutes)***

- Recap of functions, objects, and interfaces.
- Preview of Day 3.

Day 3: Classes, Enums, and Union/Intersection Types

Objective: To explore object-oriented programming with classes, learn about enums, and combine types using union and intersection.

Classes in TypeScript (25 minutes)

- Defining classes and their properties.
- Constructors and methods.
- Access modifiers: `public`, `private`, and `protected`.
- Implementing interfaces in classes.
- **Hands-on:** Creating a class with properties, methods, and access modifiers.

Enums (10 minutes)

- What are enums and when to use them?
- Numeric and string enums.
- **Activity:** Defining and using an enum.

Union and Intersection Types (20 minutes)

- Allowing a variable to be one of several types (Union).
- Combining multiple types into one (Intersection).
- Type narrowing with `typeof` and `instanceof`.
- **Hands-on:** Writing a function that accepts a union type and performs different actions based on the type.

Q&A and Wrap-up (5 minutes)

- Review of classes, enums, and advanced types.
- Preview of Day 4.

Day 4: Generics, Utility Types, and Final Review

Objective: To understand how to write reusable, type-safe code with generics and leverage built-in utility types.

Generics (25 minutes)

- Introduction to generics for creating reusable components.
- Generic functions and interfaces.
- Generic constraints to limit the types that can be used.
- **Hands-on:** Creating a generic function that can work with different data types while maintaining type safety.

Utility Types (20 minutes)

- Overview of commonly used utility types: `Partial<T>`, `Readonly<T>`, `Pick<T>`, `Omit<T>`.
- Practical examples of how utility types can simplify your code.
- **Activity:** Using utility types to create new types from existing ones.

Course Review and Next Steps (15 minutes)

- Recap of all topics covered.
- Best practices and further learning resources.
- Final Q&A session.