What Is TypeScript?

Adds new Features + Advantages to JavaScript

Browser CAN'T execute it!

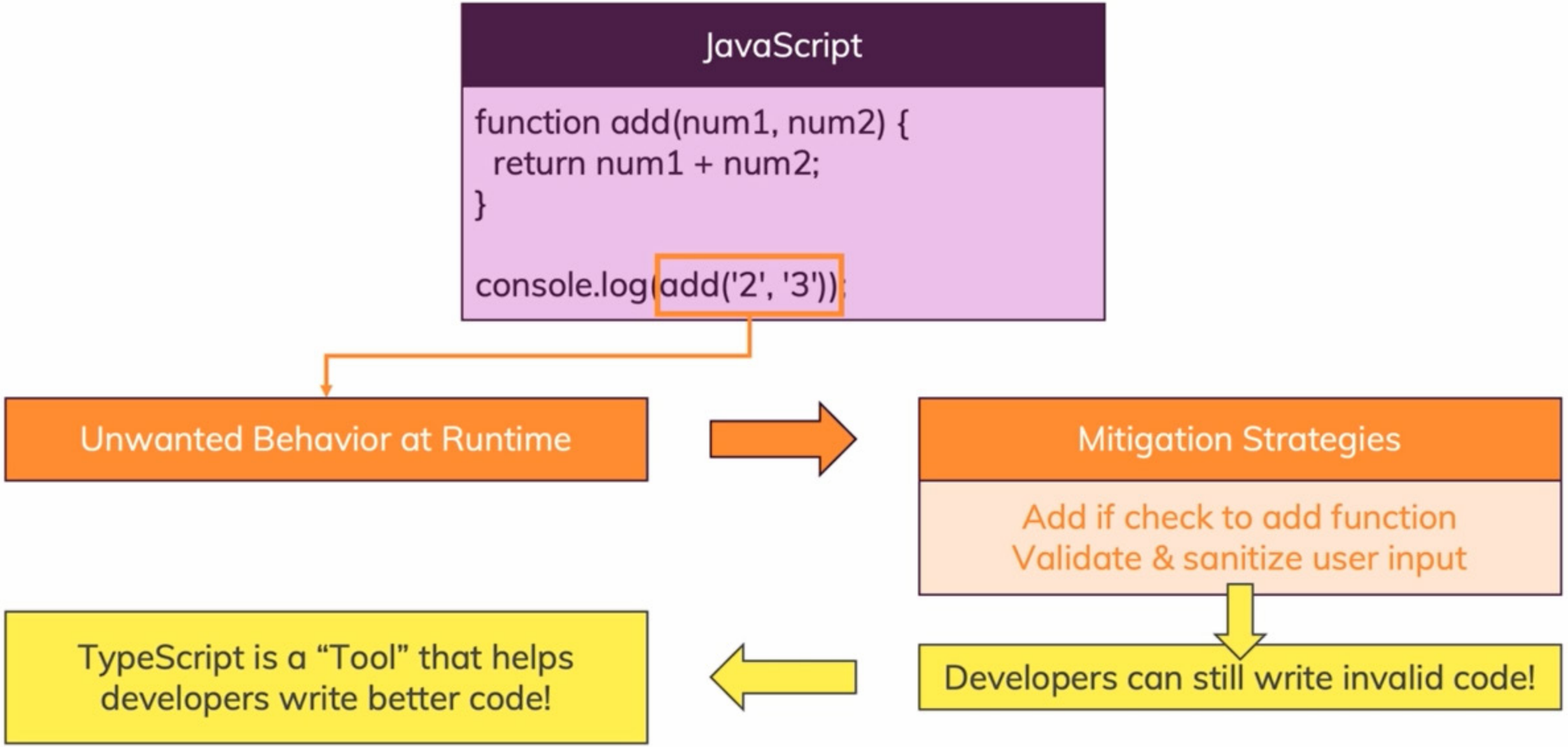
A JavaScript Superset

A Language building up on JavaScript

Features are compiled to JS "workarounds", possible errors are thrown

Compiled to

Why TypeScript?

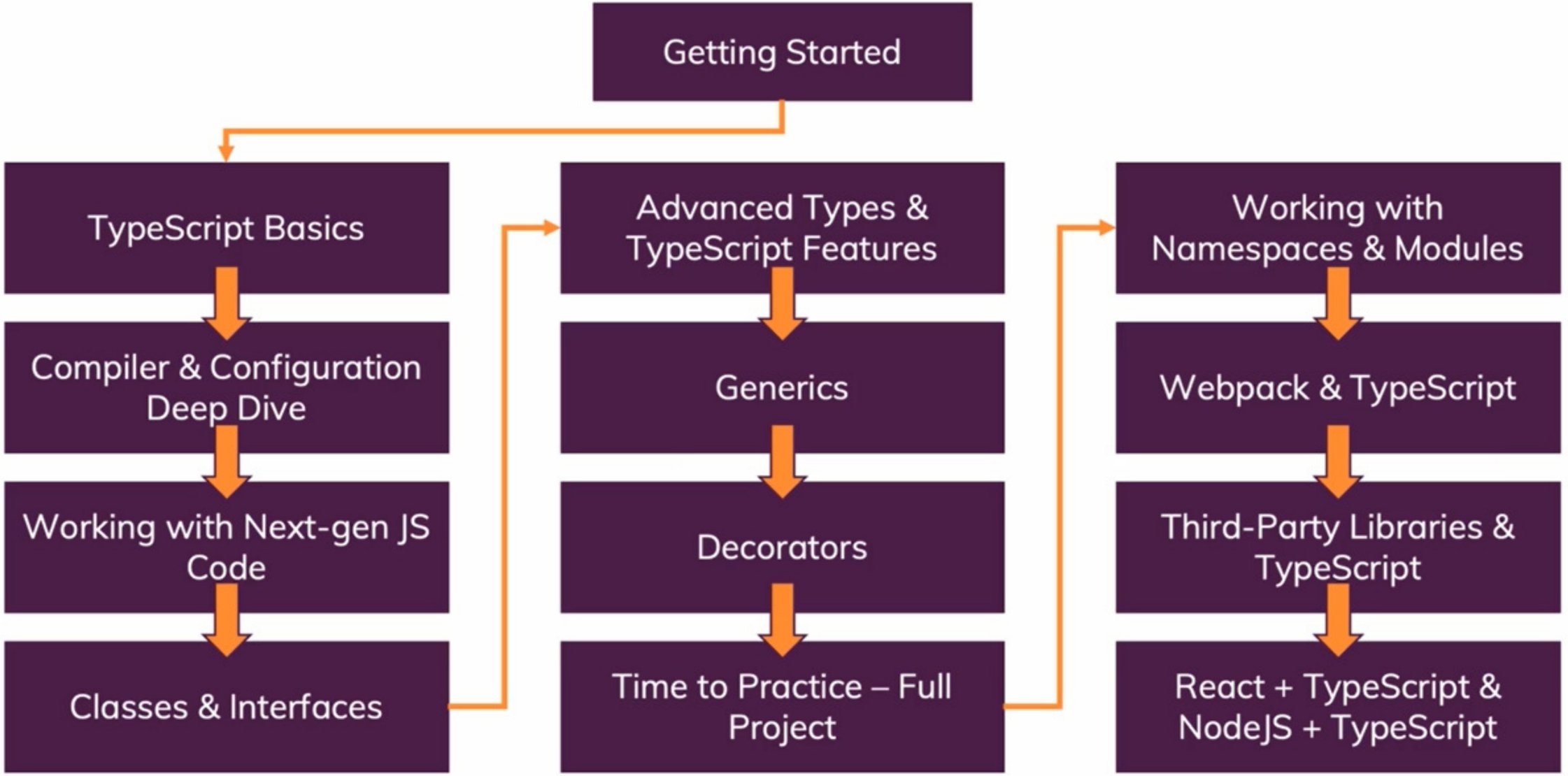


TypeScript Overview

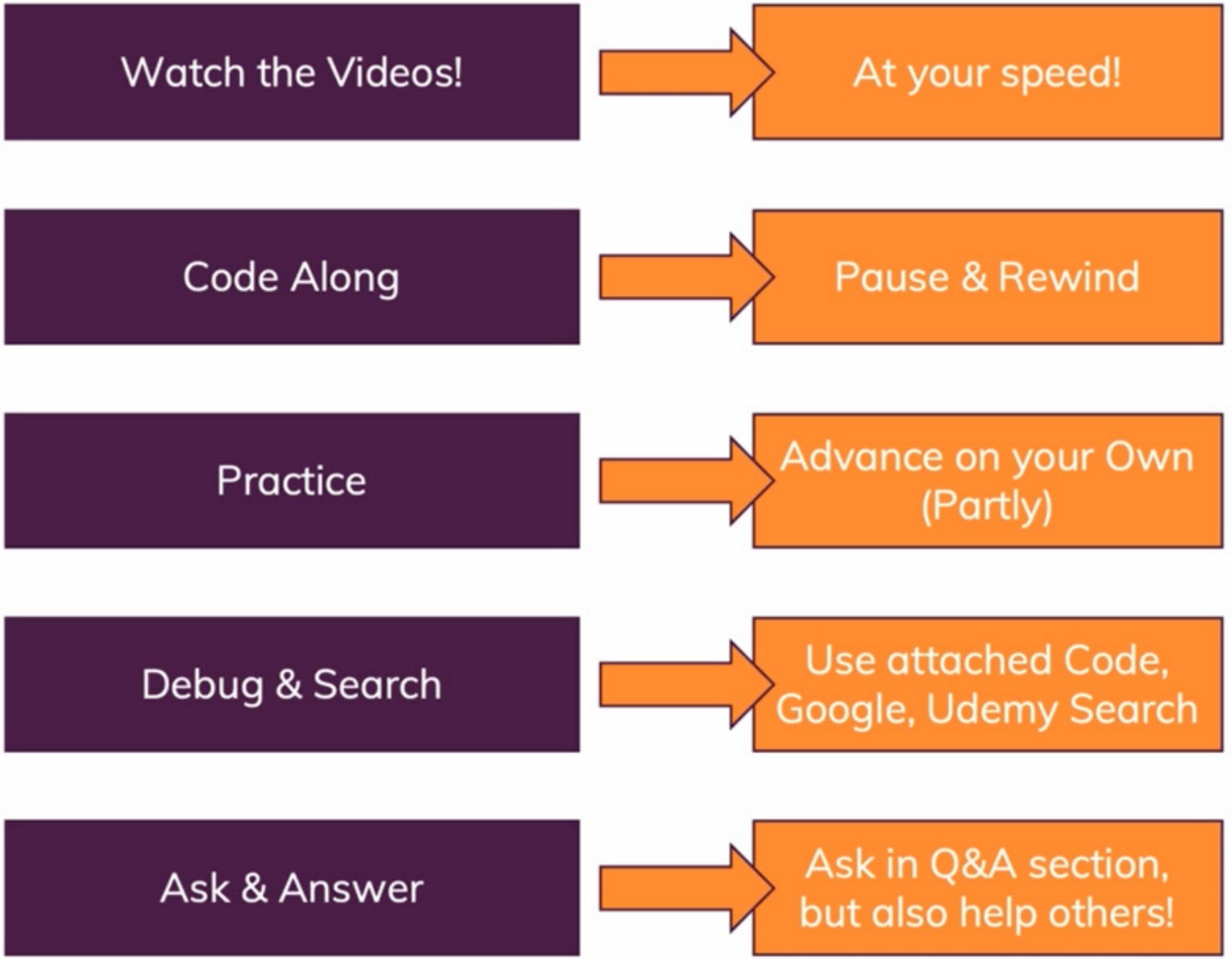
TypeScript adds...

- Types!
- Non-JavaScript Features like Interfaces or Generics
- Rich Configuration Options
- Next-gen JavaScript Features (compiled down for older Browsers)
- Meta-Programming Features like Decorators
- Modern Tooling that helps even in non-TypeScript Projects

Course Outline



How To Get The Most Out Of The Course



Working with Types

Core Syntax & Features

Core Types

number	1, 5.3, -10	All numbers, no differentiation between integers or floats
string	'Hi', "Hi", `Hi`	All text values
boolean	true, false	Just these two, no "truthy" or "falsy" values
object	{age: 30}	Any JavaScript object, more specific types (type of object) are possible
Array	[1, 2, 3]	Any JavaScript array, type can be flexible or strict (regarding the element types)
Tuple	[1, 2]	Added by TypeScript: Fixed-length array
Enum	enum { NEW, OLD }	Added by TypeScript: Automatically enumerated global constant identifiers
Any	*	Any kind of value, no specific type assignment

The TypeScript Compiler

Configuring & Using the TypeScript Compiler

TypeScript & Modern JavaScript

Using Next-Gen JavaScript Syntax

Classes & Interfaces

Working with Objects

Module Content

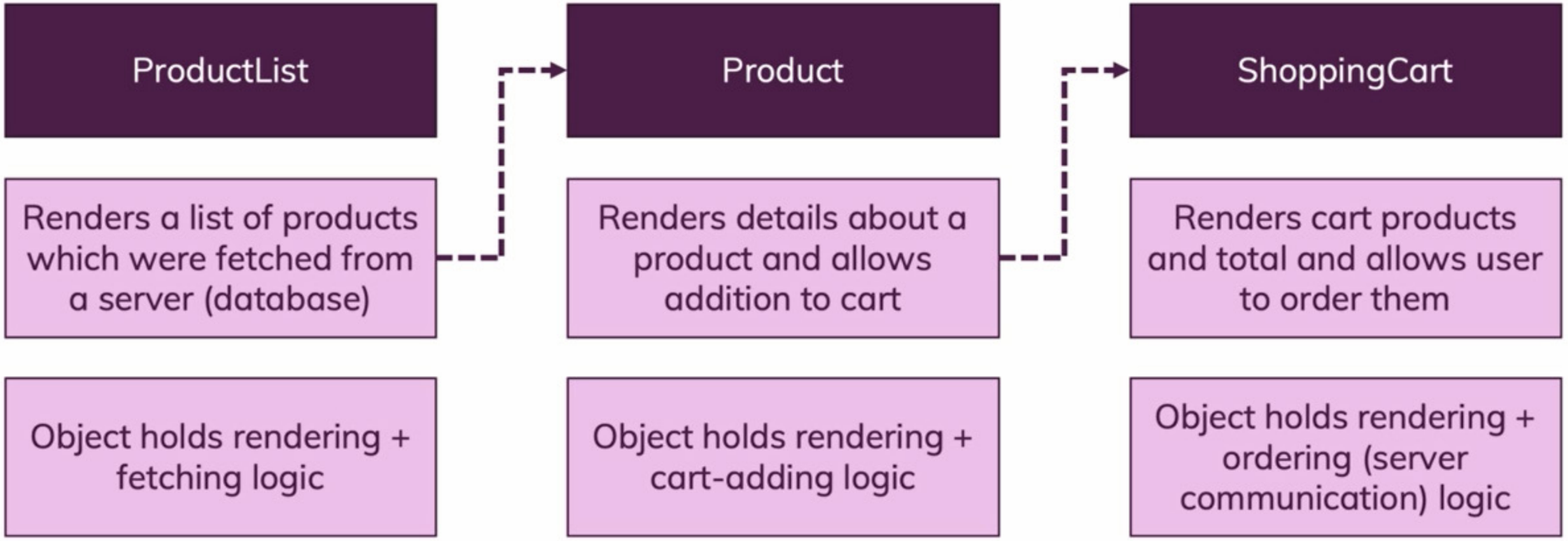
What & Why?

Classes & Inheritance

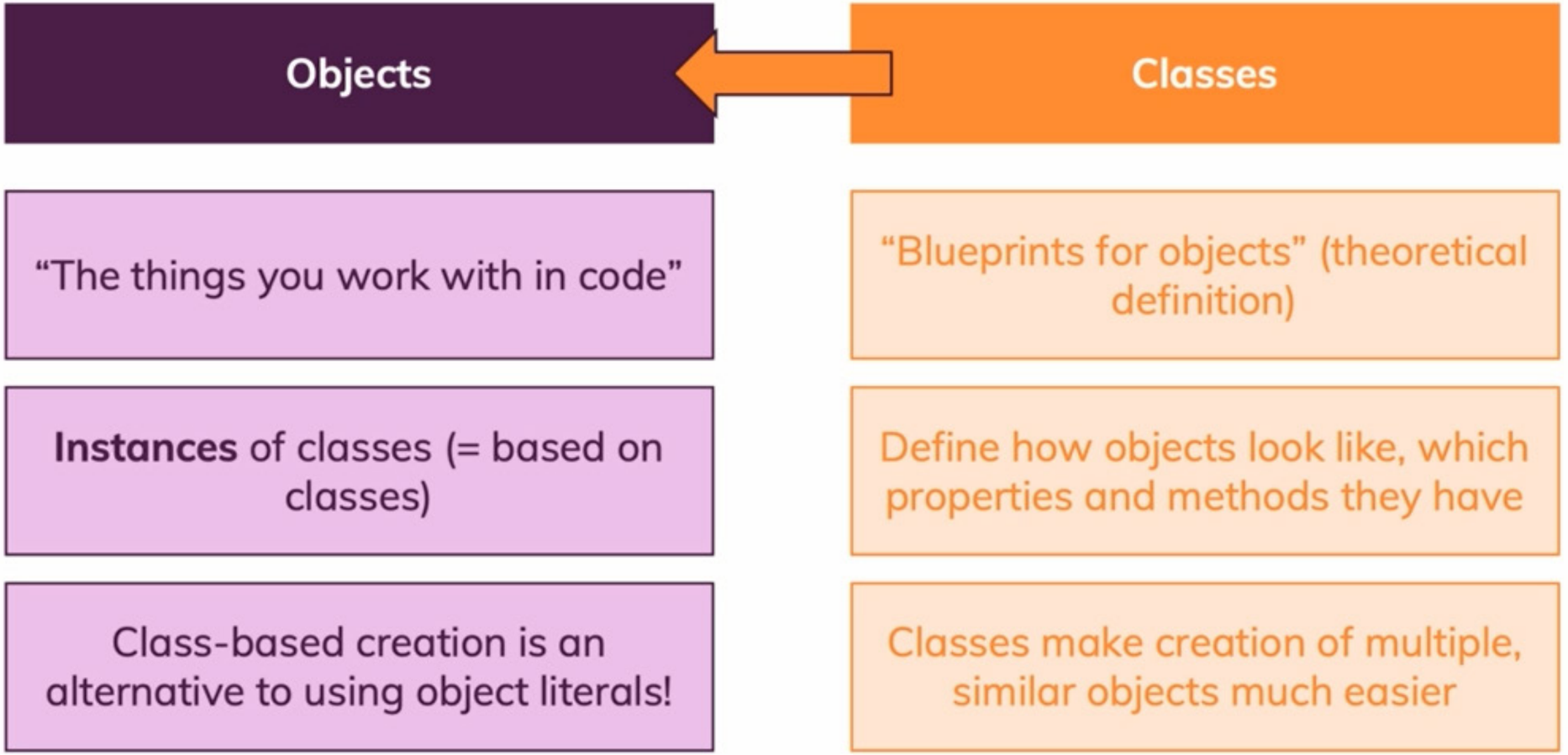
Interfaces

What's Object-oriented Programming (OOP)?

Work with (real-life) Entities in your Code



Classes & Instances



Advanced Typing Concepts

Beyond the Very Basics

Module Content

- Intersection Types
- Type Guards
- Discriminated Unions
- Type Casting
- Function Overloads

Generics

Flexible & Re-Usable Code

Module Content

- What?
- Generic Functions & Classes
- Constraints
- Special TypeScript Types

Decorators

Meta-Programming

Module Content

What?

Decorator Usage

Examples!

Practice Time!

Using TypeScript In A Project

Writing Modular Code

Splitting Your Code Into Modules

Module Content

Two Options to Organize Code In Multiple Files

Demo Time!

Splitting Code Into Multiple Files

Namespaces & File Bundling	ES6 Imports/ Exports
Use “namespace” code syntax to group code	Use ES6 import/ export syntax
Per-file or bundled compilation is possible (less imports to manage)	Per-file compilation but single <script> import
	Bundling via third-party tools (e.g. Webpack) is possible!

TypeScript & Webpack

A Modern Build Workflow

Module Content

What is Webpack?

Example Setup

What is Webpack?

Webpack is a Bundling & “Build Orchestration” Tool

“Normal” Setup	With Webpack
Multiple .ts files & imports (Http requests)	Code bundles, less imports required
Unoptimized code (not as small as possible)	Optimized (minified) code, less code to download
”External” development server needed	More build steps can be added easily

TypeScript & 3rd Party Libraries

Utilizing TypeScript's Power!

Module Content

"Normal" Libraries & Using them with TypeScript

TypeScript-specific Libraries

Time to Practice!

Using 3rd Party Libraries

Using React with TypeScript

A Popular Combination

Using Node.js & Express with TypeScript

A Popular Combination