ANGULAR

The Super-heroic Framework

TRAINING AGENDA

- Angular Introduction
- Angular Building Blocks
  - Components
  - Directives
  - Pipes
  - Services
  - Modules
- Data Bindings
- Forms
- Remote Server Calls
- Single Page Apps
- Debugging
- Optimizing Apps
- Testing

Why Angular?

- Easy to understand
- ES2015+ features
- Corporate Care-taker
- Performance
- Maintenance
- App Architecture

What is Angular?

Angularis a structural framework for dynamic (SPA) web apps.

Extension of standard HTML markups.

Keeps the front-end clean; separation of concerns.

Why TypeScript?

Microsoft Extension of JS

Object-oriented features

ES6+ Features

Type Definition

Angular itself programmed in TypeScript

TypeScript Features

Classes and Inheritance

Module System

Type Definition

ES6+ Features

Decorators

Additional Types

Angular Environment

Let’s understand Angular environment set-up

ANGULAR

Environment Setup

- NodeJS
- TypeScript
- Webpack
- RxJS
- ZoneJS
- Angular Packages

Components

Reusable piece of code

Angular Components

- A component controls a patch of screen real estate that we could call a view and declares reusable UI building blocks for an application.

- Passing data to/from components
  - Property binding
  - Event binding
  - Two-way data-binding

Angular Components [Cntd..]

Nested Component Communication

- Parent-to-child communication
  - Use of Input Decorator
  - Property Binding Syntax – [ ]
- Child-to-parent communication
  - Use of Output Decorator
  - Use of EventEmitter Class
  - Event Binding Syntax – ( )

Angular Components [Cntd...]

Data Projection / Content Projection

- Content projection is a pattern in which you insert, or *project*, the content you want to use inside another component
- Data / Content projection allows a directive to make use of templates while still being able to clone the original content and add it to the DOM
- For example, you could have a Card component that accepts content provided by another component.

Angular Component Life Cycle

ngOnChanges	called when an input binding value changes
ngOnInit	after the first ngOnChanges
ngDoCheck	after every run of change detection
ngAfterContentInit	after component content initialized
ngAfterContentChecked	after every check of component content
ngAfterViewInit	after component's view(s) are initialized
ngAfterViewChecked	after every check of a component's view(s)
ngOnDestroy	just before the component is destroyed

Directives

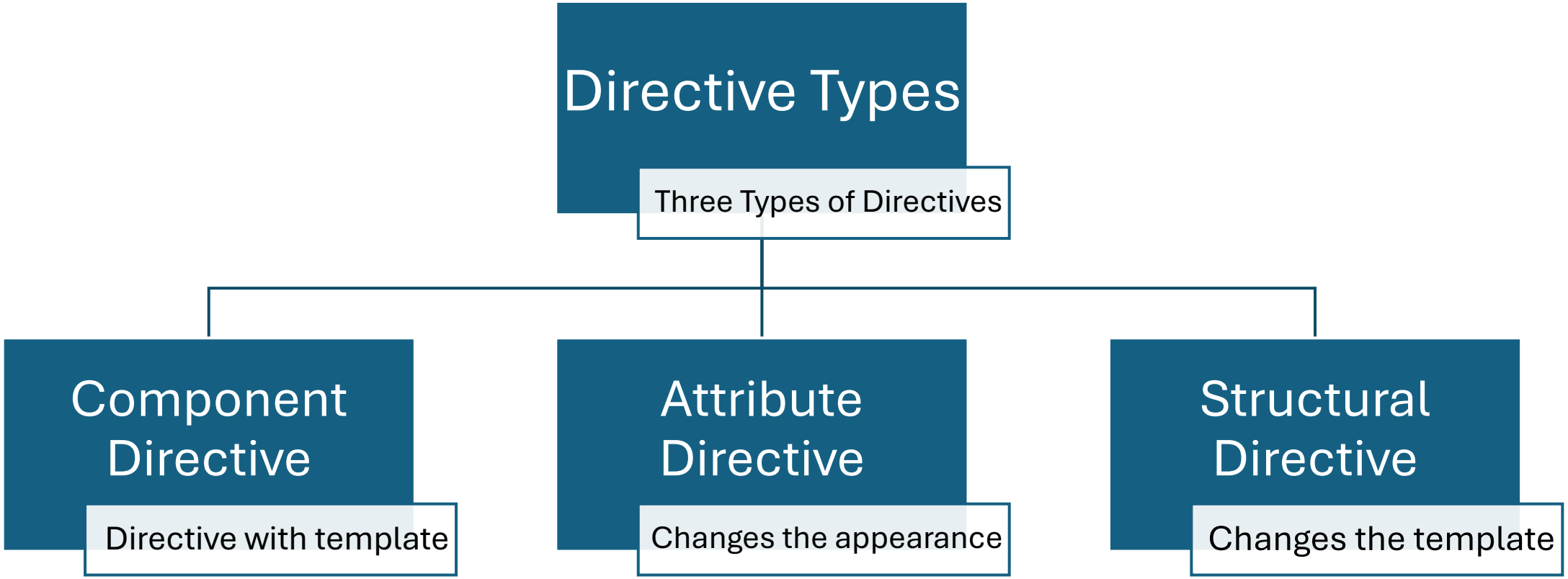
A Marker on the UI

Directives

A directive is a marker available on the UI

A Directive modifies the DOM to change appearance, behaviour or layout of DOM elements

Directives Types



Directive Types [Cntd...]

Attribute Directives

Directives that change the behaviour of a component or element but don't affect the template

Built-in attribute directives examples

- ngStyle
- ngClass

Structural Directives

Directives that change the behavior of a component or element by affecting how the template is rendered

Built-in structural directive examples

- *ngIf
- *ngFor
- *ngSwitch

Pipes

Let’s Format the UI

Pipes

Pipes are used to format data for template

Pipes can be supplied with interpolation syntax and property binding syntax as well

Pipes can be used with Pipe (|) Operator followed by the pipe name

Pipes [Cntd...]

Built-in Pipes

Angular provides serveral built-in pipes

- uppercase
- lowercase
- titlecase
- json
- percent
- async
- number
- currency
- date

Custom Pipes

We can also create our own custom pipe to meet the application requirement

- Pure Pipes : by default, all pipes are pure
- Impure Pipes : we can make impure pipes by supplying pure property to false in the pipe decorator

Forms

Accepting inputs from UI

Forms – Classes and States

ngTouched / ngUntouched	→	Toggles when the form element is focused or loss focus
ngDirty / ngPristine	→	Toggles when the form element is in initial state or the value of the element is changed
ngValid / ngInvalid	→	Toggles when the form element satisfy the validation or not

Angular Forms [Cntd...]

Template Driven Forms	Model Driven / Reactive Forms
Angular infers the Form Object from the DOM - "ngForm"	Form is created programmatically and sync with the DOM
App logic resides inside the template	App logic resides inside the component
Uses HTML5 Validations	Use of FormControl, FormGroup, FormBuilder

Services

Business logic and more

Angular Services

- Service in Angular, can be as simple as writing ES6 class
- Services provides singleton effect
- Services implements Dependency Injection in Angular App
- Services can be self registered or they can be registered in Angular Modules / Components

Angular Services – Hierarchical Injector

Root Module	Root Component	Other Components
• Same instance of service is available Application wide	• Same instance of service is available for all components (but not for other services)	• Same instance of service is available for the self-component and its own child components

RxJS Observables

Stream of Events

RxJS Observables

Reactive Extensions Library for JavaScript

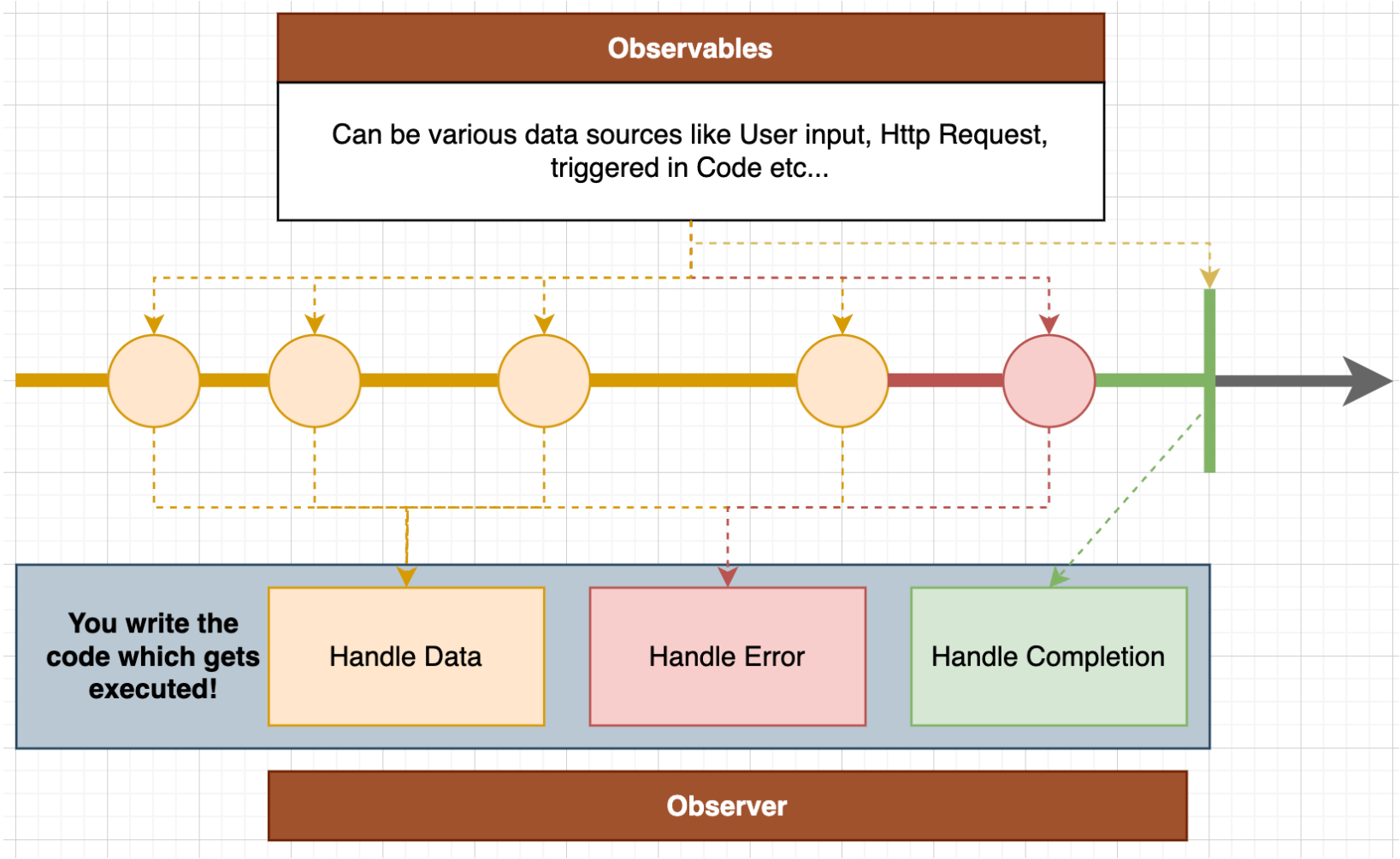
Event-based programming

Observables are Lazy

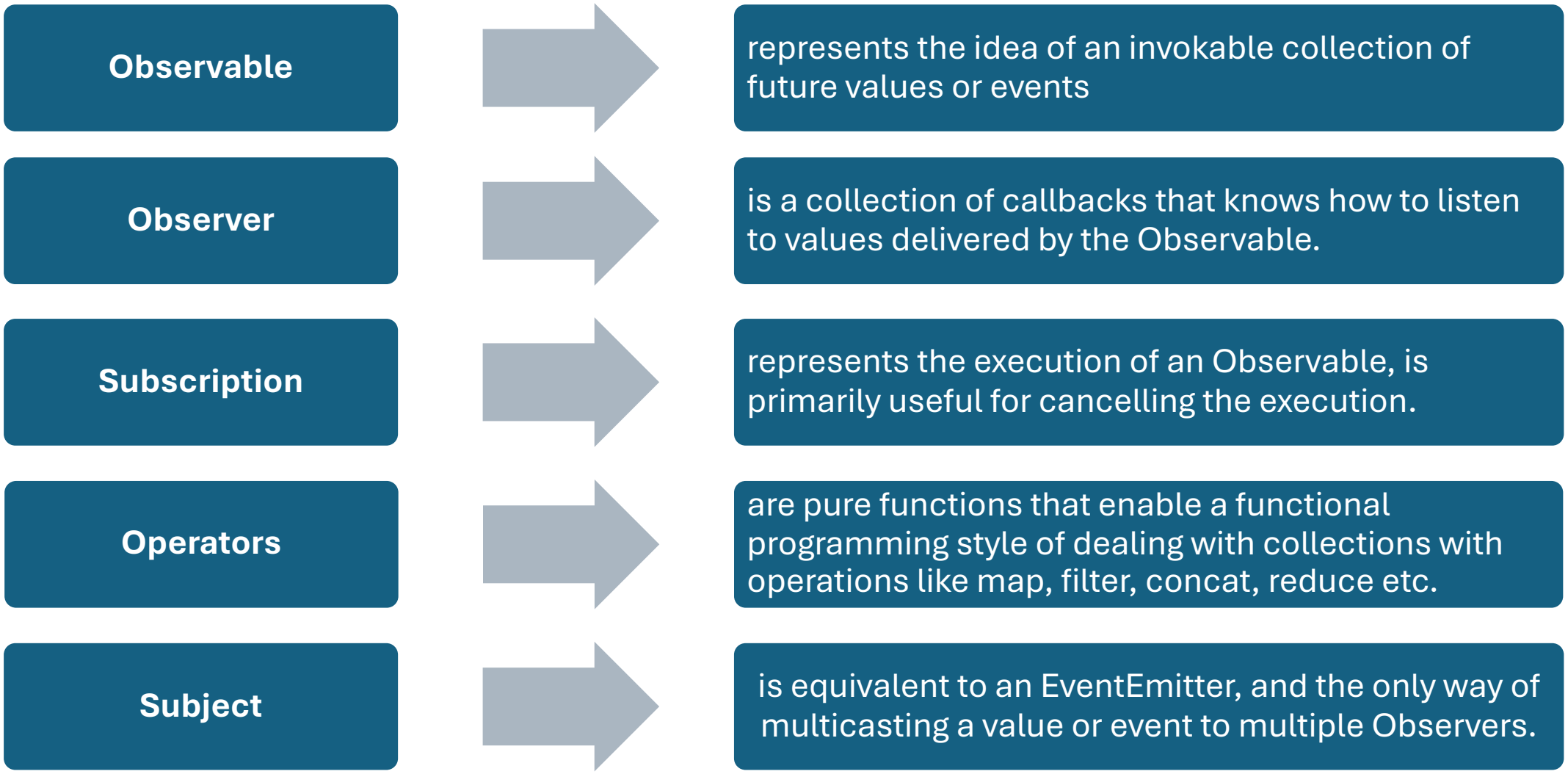
Both Synchronous and Asynchronous

Can be cancelled anytime

RxJS Observables [Cntd...]



RxJS Observables [Cntd...]



HttpClient Service

Let’s make a call

HttpClient Service - Introduction

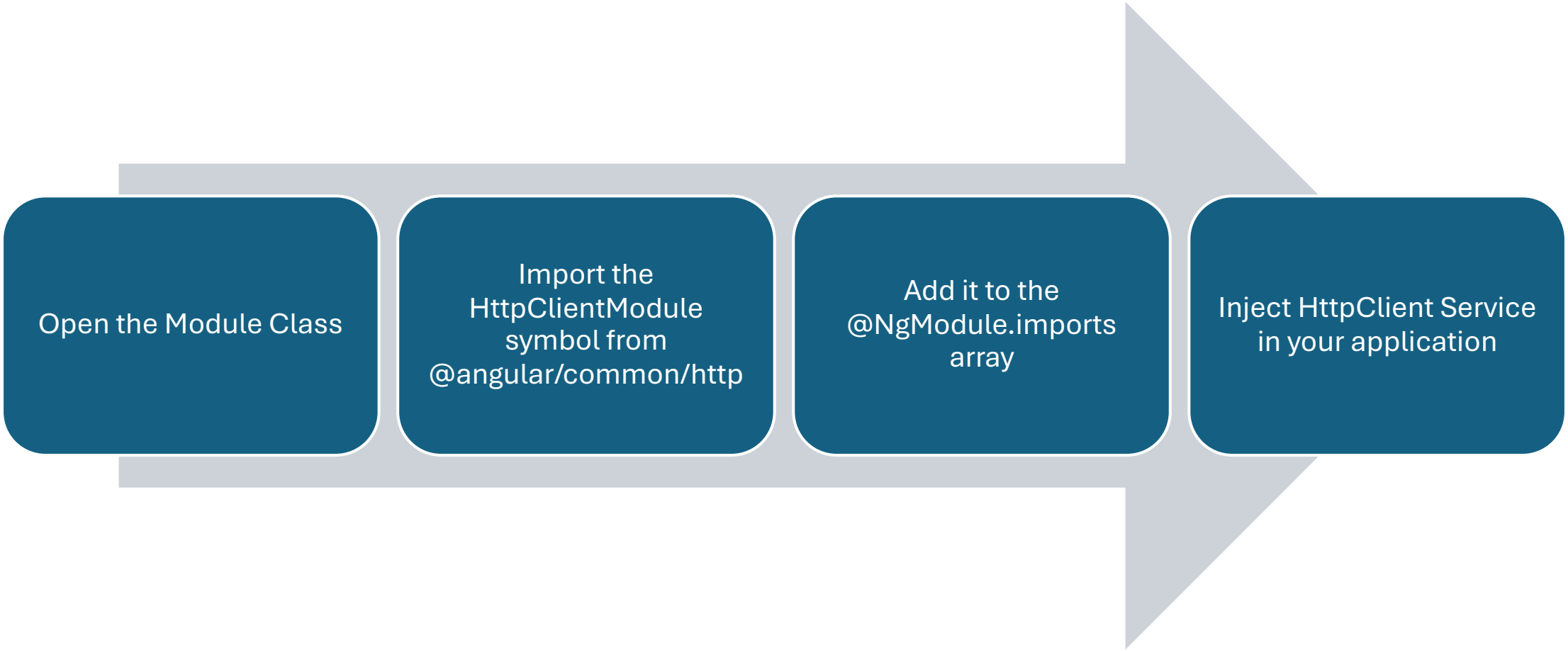
The HttpClient in @angular/common/http offers a simplified client HTTP API for Angular applications that rests on the *XMLHttpRequest* interface exposed by browsers.

HttpClient Service [Cntd...]

The Benefits of using HttpClient over other ways for fetching data from REST endpoints are -

- Typed request and response objects
- Request and response interception
- Streamlined error handling.
- Observable APIs

HttpClient Service [Contd...]



Routing and Navigation

Creating Single Page Application

Angular Router - Introduction

Angular Routing allows us -

- Maintain the state of the application
- Implement modular applications
- Implement the application based on the roles

Angular Router – 5 Steps Routing

Step 01

- Checking the *base href* tag in index file
- Open index.html and set the base tag to your domain / sub-domain

Step 02

- Configuring routes with components
- Use the Routes Config from Angular Router Package

Step 03

- Tell angular about routing app
- Import RouterModule and supply the created Routes using forRoot / forChild function

Step 04

- Setting up the routing links
- Use router-link attribute instead of href on anchor element. Prevents reloading of page.

Step 05

- Provide space on template to load the component
- Use <router-outlet> component provided by Angular Router

Angular Router Terminologies

Terminology	Description
Router Service	• Able to navigate the user programmatically without clicking any link from code
ActivatedRoute Service	• Able to access the current URL • Access Route parameters and Query parameters
Route Guards	• Route protection guards available with Angular Router package • CanActivate, canDeactivate, canResolve etc are the examples
loadChildren Property	• To load the module lazily (on-demand)
pathMatch Property	• Matched the complete path or prefix available in URL
redirect Property	• Redirect the User to a given path

Modules

Let’s split application features

Angular Modules - Introduction

A module is a mechanism to group components, directives, pipes and services that are related

Module Types -

- **Root Module:** One Module per angular application
- **Feature Modules:** depending upon the features of the application

Modules can be instantiated lazily

Debugging

Finding bugs and debugging

Debugging Angular App

Prevent Bugs with TypeScript

Using Debugger Statements to Stop JavaScript Execution

Inspect Data with the JSON pipe

Console Debugging

Debugging RxJSObservables using ‘tap’ operator

Angular Chrome Plugin

Optimization

Optimizing the Angular App

Optimizing Angular App Performance

- Using onPush change detection strategy
- Using trackBy function with ngFor
- Using lazy loading
- Avoid computing values in templates
- Disable change detection (if required)

References

Books

- Rangle's Angular Gitbook
- Ngbook

Web

- http://angular.io
- http://rangle.io
- http://reactivex.io
- http://learnrxjs.io
- https://redux.js.org