Advanced Angular Development - Modules

Im Seminar "Advanced Angular Development" bringen wir Ihre Angular Kenntnisse auf Experten-Level. Großes Augenmerk legen wir darauf, in den Demos & Labs aktuelle Coding-Styles & Patterns zu verwenden. Wir besprechen mögliche Refactorings & Schematics für die Migration bestehender Lösungen und setzen dies fallweise in Form von Live-Coding um.

Wir beginnen mit der Implementierung eines Angular Material Themes und lernen dabei die Kurs-Demo-App kennen, welche wir im Laufe des Kurses erweitern. Standalone Components sowie, deren Konzepte und Migration bilden den Einstieg ins Module Components & Forms Deep Dive.

Ein Schwerpunkt des Kurses ist der Themenblock Reactive Programming mit RxJs und State Management mit NgRx, sowie Advanced Routing und App Initialization.

Wir diskutieren die Implementierung von Authentifizierung mit Cloud Identities und Tests mit Jasmine, Jest, Cypress. Zusätzlich behandeln wir die Themen Reusability mit Libraries, Schematics, Nx & Angular Elements und Optimierung von Anwendungen, Server Side Rendering, sowie A11y.

Zum Abschluss implementieren wir ein Real Time connected Mikro-Frontend als Progressive Web App und publizieren Angular Apps in die Cloud mittels Containern und Config Injection.

Nach Abschluss des Kurses haben die Teilnehmer Kenntnisse zu folgenden Themen:

Theming Angular Apps Standalone Components: Concepts & Migration Components & Forms Deep Dive Mastering Reactive Programming using Signals & RxJs Advanced State Management using NgRx Advanced Routing and App Initialization Securing Angular using Cloud Identities Advanced Testing with Jasmine, Jest, Cypress and NgRx Reusability with Libraries, Schematics, Nx & Angular Elements Optimizing Applications & Server Side Rendering & A11y Implementing & publishing a Real Time connected Micro-Frontend as a Progressive Web App Publishing Angular Apps to the Cloud using Containers and Config Injection

Neue Themengebiete werden anhand von Folien und Demos erarbeitet. Am Ende der Module werden die erlernten Inhalte als Lab in eine durchgängige Anwendung integriert, welches am Ende in die Cloud publiziert werden kann.

Theming Angular Apps

- Style inheritance and View Encapsulation
- Comparing Angular Material & Bootstrap
- Material Theming Overview
- Using Material Colors to define Primary, Accent and Warning Colors
- Building a Reusable Material Theme
- Define Alternative Themes
- Theming Custom Components and override Material Components
- Material Design Migration to MDC
- Using Bootstrap with Angular

Standalone Components: Concepts & Migration

- Standalone Components vs Modules
- Creation, Bootstrapping
- Providers & Dependency Injection
- Routing & Lazy Loading
- Migration of an existing Project to Standalone Components

Components & Forms Deep Dive

- Using & Migrating to Control Flow Syntax
- Standalone Directives & Directives Composition Api
- Components and Required Inputs
- Content Projection
- Templates TemplateRef, *ngTemplateOutlet
- Comparison: ng-template vs ng-content - pro / cons
- ViewChild, -Children, ContentChild, -Children
- HostBinding & HostListener
- Recap Reactive Forms Revisited (FormGroup, Form Builder, FormControl, FormArray)
- Dynamic Component Loading & DataBinding
- Untyped Forms vs Typed Forms
- Typed Forms Nullability, NonNullableFormBuilder, GetRawValue
- Partial Values, Optional Controls, Dynamic Groups and FormRecord
- Cascading Form Controls
- Implementing Custom Controls using ControlValueAccessor
- Typed Forms Validation & Custom Validators
- Handling FormErrors & ErrorStateMatcher

Mastering Reactive Programming using Signals & RxJS

- Imperative vs Functional Programming
- Immutability & Pure Functions
- Introduction to RxJS
- Observables, Observers & Use Cases
- Data- vs Action-Streams
- Mouse & DOM Events as Observables
- Implementing Side Effects using tap
- Base Operators: Mapping, Filtering, Merging, Scanning, ...
- Unsubscribing (takeUntil, DestroyRef, takeUntilDestroyed)
- Introduction to Signals
- Imperative vs Declarative Reactive Programming
- Signals vs Observables: Synchronous & Asynchronous Reactive Programming
- Understanding Marble Diagrams & Debugging Observables
- Marble-testing RxJS
- Combination & Transformation Operators
- Retry & Error Handling Strategies
- Implementing & Testing Custom Observable Operators
- Communication between using Event Bus Pattern
- Stateful Services using Behavior Subjects and Signals

Advanced State Management using NgRx

- Overview State Management Patterns
- Introduction to the Redux Pattern & NgRx
- Feature State and ActionReducerMap
- Implementing NgRx Store, Reducers & Selectors using createFeature
- Actions & createActionGroup
- Debugging NgRx using Redux Dev Tools
- Effects, Facades, @ngrx/entity adapters
- Simplifying Data Access with @ngrx/data
- NgRx Container Presenter Best Practices
- NgRx and Signals Interoperability
- @ngrx/component-store vs classic NgRx Store
- Using @ngrx/component-store

Advanced Routing and App Initialization

- Dependency Injection in Depth: Resolution modifiers and Dependency providers
- Using Constructor vs inject for DI
- APP_INITIALIZER, Injection & forwardRef
- Implementing Global Error Handling and Retry-Patterns
- Modules & Code Splitting
- Introduction to @ngrx/router-store
- Routing using NgRx Actions
- Binding Router-Params to Component Inputs
- Functional Route Guards & Interceptors
- Integrating Route Guards & Interceptors with NgRx
- Chaining Route Guards & Interceptors
- Auxiliary Routes: Common use cases
- Preloading Component Data from NgRx using Functional Resolvers
- Using Preloading Strategies
- Router Animations & Anchor Scrolling
- Introduction to Visual Feedback (Loading-, Saving-, ...-Indicator)

Securing Angular using Cloud Identities

- Recap Jwt, OAuth 2.0 & OpenID Connect
- Token based Authentication in Angular with NgRx
- Implementing an AuthModule using a Facade Service, Components, Guards & Interceptors
- Optimizing Application Flow for Authentication
- Authentication using Microsoft Entra ID

Advanced Testing with Jest, Cypress and NgRx

- Introduction Angular Testing Tools (Jasmine, Karma, Jest & Cypress)
- Testing Classes, Pipes, Directives
- Testing Services using HttpClientTestingModule & HttpTestingController
- Mocking vs Spies

- Testing Component Interaction (Read, Write, Emit, Inputs)
- Complex Forms Testing
- Testing Observables & BehaviourSubjects
- Material Testing using Component Harnesses
- Async Component Testing (done, fakeAsync, waitForAsync)
- Components Marble Testing
- Testing NgRx: Mock Store, Mock Selectors, Reducers, Effects, Facades
- Using Jest for Unit Testing (Setup, Changes in spec, Snapshot Tests)
- Introduction to End-2-End Testing using Cypress
- Cypress Component Tests

Reusability with Libraries, Nx, Schematics & Angular Elements

- Angular Building Blocks: Workspace, Apps, Libraries
- Reusable Artifacts using Angular Libraries
- Implementing, Publishing and Consuming Libraries to / from GitHub Packages
- Understanding Monorepos: Pro / Cons
- Introduction to Nx Workspaces
- Understanding and Implementing Schematics
- Implementing Web Components using Angular Elements

Implementing a Real Time connected Micro-Frontend as a Progressive Web App

- Introduction to Micro-Frontend and Event Driven Architecture (EDA)
- Implementing a Real Time connected Micro-Frontend listening to Cloud Events
- Using `@ngrx/component-store`
- Introduction to Progressive Web Apps
- Understanding and Configuring Service Workers & Manifests
- Installing & Updating Progressive Web Apps
- Introduction to Module Federation

Building & Optimizing Applications

- Using Chrome Dev Tools & Lighthouse for Performance Optimization
- Analyzing and Optimizing Bundles & Modules
- Deferred Loading
- Understanding & Using Page Traces
- Optimizing Images using NgOptimizedImage
- Logging NgRx to custom destinations using Meta-Reducers
- Virtual- & Infinite Scrolling
- Understanding, Profiling & Optimizing Angular Change Detection
- Optimize Change Detection using ngrxPush, ChangeDetectioRef
- Change Detection and Signals
- Using Linting and Autoformat with Prettier
- Accessibility A11y: Best Practices & Linting
- Introduction to Server Side Rendering (SSR) and Non-destructive hydration
- Why Server Side Rendering

- Configure Server Side Rendering & Pre-rendering

Publishing Angular App using Containers and Config Injection

- Deployment Overview & Cloud Hosting Options
    - Using ng deploy to publish to Firebase
    - Deploy to Azure Static Webapp
- Configuration Management and Config Injection Options
    - Using a config service
    - Creating an Angular Multi-Stage Docker Image
    - Overriding config in containers using environment variables
- Deploy to a Cloud Container Host (Azure Container Apps)
    - Azure Container Apps Overview
    - Publish & Configure Api & Angular UI Containers