Sebastian Kronschläger, Lukas Humer, Florian Eggetsberger, Oliver Doppelbauer

Bericht Angular

Vor- und Nachteile:

Vorteile:

- Angular folgt dem MVC Ansatz, das heißt Backend und Frontend sind voneinander zu trennen
- Single Page Applications lassen sich schreiben. Also es wird nur eine Seite immer geladen und die weiteren Components werden in der gleichen Seite geladen
- Dieses Framework ist im Trend

Nachteile:

- Unterstützt alte Browser nicht
- Für kleinere Projekte sind andere kleinere Frameworks besser geeignet
- Nicht für Programmieranfänger geeignet
- Schlechte Skalierbarkeit

Aufsetzen des Angular Frameworks

Im Terminal

1. Node.js installieren
2. **Installation**: npm install –g @angular/cli
3. **Aufruf:** ng serve

Wie funktionieren REST-Calls?

1. Man muss im app module das HttpClientModule importieren
   import { HttpClientModule } from '@angular/common/http';
2. Dann erstellt man eine Service Klasse, in der man HttpClient importieren muss.
3. Nun kann man in einer Methode z.B. **return this.http.get(this.serviceUrl);** schreiben (serviceUrl ist der Localhost auf dem das Backend läuft). Diese Methode kann man dann in seinem Component aufrufen.

Besonderheiten des Angular Frameworks

Das Angular Framework hat einige Besonderheiten gegenüber von anderen Frameworks. Diese Besonderheiten wären:

- **Cross-Platform:**
  Man kann mit Angular sowohl Progressive Web Applikationen als auch native Applikationen entwickeln. Ebenfalls ist es mit dem ionic SDK möglich mobile Apps zu entwickeln.

- **Sehr schnelle Ladegeschwindigkeit und Optimierte Performance:**
  Dadurch das Angular templates zu code macht, verbessern sich die laden enorm. Die Ladegeschwindigkeit einer Angular app ist schneller als jedes andere Framework.

- **Fehlersicherheit:**
  Durch die 11 eingebauten Testmodule die helfen einen fehlerfreien Code zu entwickeln.

Welche Probleme sind beim Arbeitsauftrag aufgetreten? Wie haben wir diese Gelöst?

Am Anfang war es für uns noch schwer einen Überblick über dieses Framework zu bekommen, da wir vorher noch nie etwas mit Angular gemacht haben, aber mit der Zeit wurde es immer besser.

Auch beim Design haben wir am Anfang Schwierigkeiten, haben dann aber Angular Material gefunden das hat uns die Aufgabe sehr erleichtert.

Ein weiteres Problem war die Übersicht der Components. Dies ist der Grund da wir mehrere Components in einer Datei hatten. Um dieses Problem zu lösen haben wir Extensions installiert um sogenannte “Regions” wie in C# einzufügen und so die Übersicht wieder herzustellen.

Unsere Empfehlung an Mitschüler für beste Benutzung von Angular

Auch wenn es sich sehr überflüssig anhört, es ist sehr wichtig alles auf eigene Components aufzuteilen. Der Grund dafür ist das die Übersichtlichkeit bei großen Projekten (für die man Angular hauptsächlich verwendet) sehr schnell verloren geht. Diese Art wird auch von Angular selbst empfohlen.

Eine persönliche Empfehlung wäre die Verwendung der IDE VSC (Visual Studio Code). Der Grund dafür ist die einfache Verwendung der IDE und der zahlreichen Erweiterungen die das Entwickeln um einiges verbessern

Ebenfalls empfehlen wir die Auseinandersetzung mit dem Tool “Angular Material”. “Angular Material” hat nämlich durch seine zahlreichen UI Komponenten eine große auswahl die UI um einiges zu verschönern/verbessern.

Coole Features des Angular Frameworks

Es gibt einige coole features für Angular Nutzung:

- **ng add und ng update:**
  Diese zwei einfachen CLI command lines ermöglicht das schnelle hinzufügen und updaten von packages zu einem Projekt.
  Eingabe der Command line: **ng add <package>**
  ng update <package>

- **Library Support:**
  Dadurch, dass dieses Feature schon oft angefragt wurde, ermöglicht diese Command Line das schnelle erstellen und veröffentlichen von Libraries:
  Eingabe der Command line: **ng generate library <name>**

- **Angular CDK:**
  Durch das update der Material.io platform beinhaltet dieses nun ein Component Development Kit (CDK). Jetzt kann man durch einfache command Lines ein Material setup in ein Angular projekt hinzuzufügen.
  Die CLI line: **ng add @angular/material**

Die Geschichte von Angular

Der Entwickler der ersten Version von Angular (AngularJS) war ein Google Mitarbeiter namens Miško Hevery. Er entwickelte dieses Nebenprojekt um ihm bei der Entwicklung von Webapplikationen zu helfen. Mit Hilfe von ein paar anderen Mitarbeitern entwickelte er ein paar interne Projekte und veröffentlichte schließlich AngularJS als Open Source Projekt in 2010.

Version 1 (20.Oktober.2010): Die Erscheinung von AngularJS, dem Vorgänger von Angular. Anders wie Angular war dieses Framework auf Javascript basiert.

Dieses Framework wurde sehr schnell bekannt. Durch die Entwicklung des Ionic Frameworks auf das Apache Cordova wurde es sehr beliebt da dies nun erlaubte AngularJS auch die Entwicklung von mobilen Anwendungen.

Durch die fortschritte von Javascript, jedoch fiel dadurch AngularJS immer weiter zurück. Ein weiterer Grund für diese Entwicklung war dass, das Team das sich um die Entwicklung des Angular Frameworks kümmerten einen Punkt erreicht haben wo sie das Framework nicht mehr verbessern konnten.

Jedoch mit der Version 2.0 schrieben sie das komplette Framework neu, um schwere Probleme beim entwickeln von großen cross-platform Anwendungen zu lösen. Dies hatte zur Folge ,dass Third-Projects wie “Angular Material” schlussendlich veraltet sind.

Version 2 (14.September.2016): Dies war das neu umschriebene Framework von “AngularJS”. Der Name wurde geändert auf “Angular”.

Jedoch hatten die Entwickler noch keine Migrationsstrategie von AngularJS auf das neue Angular 2.0. Dadurch haben viele Teams einen schnellen Wechsel auf das neue Angular 2.0, dass sich bislang noch in der Beta befand. Durch viele frustrierende Features wechselten viele Entwickler auf andere Libraries wie zum Beispiel React.

Version 4 (23.März.2017): Es wurde die Versionsnummer 3 übersprungen da ein NPM-Paket von Angular 2 bereits diese trug.

Version 5 (1.November.2017): Die wichtigste Verbesserungen dieser Version waren:

- Unterstützung für progressive Web-Apps
- ein Build-Optimizer
- Verbesserungen im Zusammenhang mit Material Design

Version 6 (3.Mai.2018): Änderungen und Verbesserungen in Version 6:

- Angular Elements sind im Framework integriert
- Framework Pakete liegen nun synchronisiert vor
- Zwei neue Command-Line-Befehle:
  - ng update
  - ng add

Version 7 (18.Oktober.2018): Änderungen und Verbesserungen in Version 7:

- Performance-Verbesserungen
- Neuerungen bei Angular Material und CDS
- Virtual Scrolling
- Typescript 3.1 und RxJS 6.3 wird verwendet

Nach einigen neuen Releasen wurde das Framework immer mehr verbessert wie zum Beispiel durch bessere build sizes, stabilere APIs und insgesamt besserer performance. Dadurch ,dass Angular von Grund auf neu designed und entwickelt wurde hat dieses neue Framework viel mehr Möglichkeiten als das AngularJS.

Ebenfalls hat es den Vorteil gegenüber anderen Libraries wie React ,dass es alles hat was man für eine große Anwendung benötigt anders als bei React wo man für verschiedene kleine Teile ein außenstehendes Projekt benötigt.

Hier darunter findet man noch die zwei neuesten Versionen von Angular:

Version 8 (28.Mai.2019): Änderungen und Verbesserungen in Version 8:

- Differential-Loading (kleinere Browserspezifische Script Pakete)
- inkrementelles Kompilieren
- Optimieren durch Ivy (Beta-Version)
- Unterstützung von Multithreading durch Web Workers (erlauben es Javascript code im Hintergrund vom Hauptthread auszuführen)
- Anpassungen beim Lazy Loading (Datenobjekte werden bereitgestellt aber erst bei konkreter Anfrage von der Datenquelle holen)
- Erweiterungen zur Angular Befehlszeile
- Typescript 3.4 und RxJS 6.5.1 wird verwendet
- Zahlreiche kleine Änderungen im Angular Ökosystem

Version 9 (6.Februar.2020): Änderungen und Verbesserungen in Version 9:

- Ivy (inkrementelles kompilieren und optimieren) nun einsetzbar. Bundles 40% kleiner und wesentlich schneller.
- Ivy ermöglicht auch effizienteres Lazy Loading (beim Start von einer Anwendung müssen nicht alle Pakete auf einmal geladen werden, sondern nur bei Benötigung)
- Integration von Lokalisierung (Anpassung in einem bestimmten umschriebenen geographischen oder ethnischen Absatz- oder Nutzgebiet) der Anwendung verbessert. Ebenfalls nur geladen bei benötigung-
- Anwendungen müssen auf neue Lokalisierung Schnittstellen angepasst werden
- Typescript 3.6/3.7 wird vorausgesetzt.

Community hinter Angular

Dadurch das Angular eine Open-Source-Software ist, besteht die Community aus vielen verschiedenen Einzelpersonen und Unternehmen. Das Hauptunternehmen davon ist Google

berühmte Applikationen mit Angular

Berühmte Applikationen die mit Angular entwickelt wurden:

- Deutsche Bank Developer Portal (AngularJS 1.7.8)
- GitHub Community Forum (AngularJS 1.7.3)
- Delta (Angular 6.1.9)
- Microsoft Office Home (Angular 7.1.3)
- VMware Clarity Design System (Angular 7.0.0)

- LUIS - Microsoft Azure (6.1.10)

Tooling Angular

Für Angular gibt es die unterschiedlichsten Tools die man verwenden kann:

- **Angular Materials:**
  Angular Material ist eines davon. Angular Materials bietet einige nützliche UI Komponenten die auf dem Material Design System basiert. Es besitzt eine CSS Library für Typography und anderer Elemente. Ebenfalls nutzt es ein responsives Layout mit flex grid.

- **Angular CLI:**
  Ein Tool das man ebenfalls verwenden könnte wäre Angular CLI. Diese CLI ermöglicht das schnelle aufsetzen von bootstrap projects, da es automatisch die build Konfiguration, testing Konfiguration und viele mehr anbietet. Dieses Tool wird auch bei Konzepte noch einmal besprochen.

- **Angular Augury**
  Augury ist ein praktisches Developer Tool das es möglich macht Angular Applikationen in Google Chrome und Mozilla Firefox zu debuggen. Das besondere an Angular Augury ist die möglichkeit die Applikation besser darzustellen durch Komponenten Bäume und visueller debugging tools.

IDEs die gut zum Programmieren von Angular geeignet sind:

- **Angular IDE:**
  Diese IDE wird sehr viel verwendet durch ihr schnelles und effektives Entwickeln. Es ist auch ein stand alone plugin das mit dem Eclipse plugin verwendet werden kann:

  Vorteile dieser IDE sind:
  - Echtzeit Fehlererkennung im Code
  - Einfach zu verstehen
  - Es hat ein auto-complete feature
  - Es benutzt viele Farben und hervorheben für syntax-aware source
  - Block und volle-Formatierung mit erweiterten Einstellungen
  - Unterstützt: Angular 7, TypeScript 3.0

- **Webstorm:**
  Eine von IntelliJ erfundene und von Jetbrains entwickelte IDE die sich exzellent für die Entwicklung von Angular 2 Applikationen eignet. Man benötigt ebenfalls keine Third-Party plugins dadurch ,dass diese IDE Typescript code kompilieren kann.

Vorteile dieser IDE sind:

- Unterstützung für Javascript, Node.js, HTML und CSS
- Es erlaubt robuste Navigation und Refactoring
- Es ermöglicht die integration in VSC (Visual Studio Code) und smart coding assistance
- Ebenfalls gibt es Plug-ins die eine komplette konfiguration, lokalen History feature und mehr
- Es unterstützt auch die neuesten Technologien von Debugging, Tracing und Testing support

- **Visual Studio Code:**
  Diese IDE unterstützt Typescript, unter anderem debugging, intelligent code completion basierend auf Typen, Funktionen und importierenden Modulen. Es hat ebenfalls unterschiedliche features wie, snipptes und code refactoring.

  Vorteile dieser IDE sind:
  - Unterstützung einer großen Anzahl an Sprachen
  - eingebaute Git commands
  - Schnelles Debugging
  - Einfach anzupassen

Geschwindigkeit (wie schnell)

Über die Geschwindigkeit von Angular lässt sich sagen ,dass sie durch die Einführung von Web Workers und server side rendering um einiges verbessert wurde.

Wie haben wir die Lernkurve von Angular empfunden

Es gab ein paar Anfangsschwierigkeiten da wir am Anfang noch nicht viel mit Javascript gearbeitet haben. Nachdem wir uns mit ein paar Tutorials auf den neuesten Stand gebracht haben, ging die Lernkurve immer weiter nach oben.

Welche Konzepte gibt es hinter Angular?

Es gibt 7 verschiedene Konzepte hinter Angular:

1. **Struktur einer Angular Anwendung:**
   Eine Angular-Anwendung teilt sich in 4 Bestandteile auf:
   - HTML Templates (Renderung der Komponenten)
   - Komponenten (erweitert Templates mit Variablen und Methoden)
   - Dienste (Anwendungslogik)
   - Module (umschließt Dienste und Komponenten)

Um eine Typescript Klasse als Komponente in Angular zu definieren muss man die Annotation @Component hinzufügen, dadurch wird die Klasse mit einer HTML-Template verbunden.

Ebenfalls wird im Browser ein HTML-Konstrukt in ein sogenanntes DOM (Data Object Model) umgewandelt. Durch dies ist es möglich ohne das aktualisieren der Seite die Änderungen zu sehen. Angular manipuliert ebenfalls das DOM.

2. **Angular CLI:**
   Die Angular CLI macht es möglich verschiedenste Teile eines Angular Projektes über command lines zu erstellen. Die wichtigsten Befehle sind hierbei:
   - **ng new:** erstellt neues Projekt
   - **ng generate:** erzeugt Komponenten, Services, usw.
   - **ng serve:** baut Grundgerüst der Application
   - **ng lint:** kontrolliert Code auf Einhaltung der Code Konventionen

   Die Verwendung einer solchen Commandline sieht zum Beispiel so aus:
   ng g component [name of Component]

3. **@Component-Annotation:**
   Durch diese Annotation wird eine Klasse als Angular-Komponente und definiert die verarbeitung und Instanziierung von dieser. In dieser Annotation gibt es vier Konzepte:
   1. **selector** (ermöglicht einfügung einer Instanz in andere HTML-Templates)
   2. **templateURL** (dies ist der Pfad zur HTML-Vorlage)
   3. **styleURLs** (dies ist der Pfad zu einer oder mehrerer CSS-Stylesheets)
   4. **providers** (Liste von Services, werden durch Dependency Injection bereitgestellt)

   Ein Beispiel für eine @Component-Annotation wäre:

```
@Component({
selector: 'app-root',
templateUrl: './app.component.html',
 styleUrls: ['./app.component.css']
})
```

4. **Bindings**
   Es gibt 7 verschiedene Möglichkeiten wie man Komponenten und HTML-Templates verbinden kann:
   1. **Interpolation Binding** (String Attribut “value:string” mit **{{value}}** verbinden)

2. **Property Binding** (Von der Komponente aus wird mit **[property] = "value"** eine Property verändert)
3. **Event Binding** (Events aus Template mit Komponenten verbinden. zum Beispiel ein Mausklick event: **(click) = "method(var)"**)
4. **Two-way Binding** (Man kann sowohl Event-Binding von Template zu Component schicken, als auch Property Binding anwenden **[(value)]**)
5. **Attribut Binding** (Bei Attribute von HTML-Befehlen kann man nicht einfach Property Binding machen. Hierbei wird Attribut Binding benötigt: **<td [attr.rowspan]={{value}}>**)
6. **Class Binding** (Hiermit kann man dynamische CSS Klassenattribute von DOM-Objekten entfernen als auch hinzufügen)
7. **Style Binding** (erlaubt das dynamische Aktivieren von CSS Styles)

5. **Structural Directives**
   Diese Directives entfernen, ersetzen oder fügen Elemente im DOM hinzu. Einige der häufig verwendeten structural Directives sind:
   - ***ngIf** (Elemente anzuzeigen oder aus der DOM zu entfernen)
   - ***ngFor** (Schleifen über Listen)
   - **ngSwitch** (Fallunterscheidungen abzuwickeln)

6. **Dependency Injection**
   Dadurch kann man Angular Instanzen zum Beispiel Services in Komponenten einspeisen. Dies wird über den Injector geregelt. Wenn man im Wurzelmodul in der provider:[...] Umgebung zum Beispiel die Serviceklassen definiert sind diese später auf globaler Ebene nutzbar. Ebenfalls existiert von jedem Dienst nur eine Instanz (Singleton)

   Auf Klassenebene wird die Einbindung in der @Component Annotation durchgeführt. Hierbei bekommt jede Komponente eine neue Instanz des Dienstes.

   Attribute können hierbei mit @Input() an die Attribute von anderen Klassen gebunden werden. Dies funktioniert so:

   import {input} from '@angular/core'

7. **Asynchrone Aufrufe:**
   Die Benutzerübersicht kommuniziert mit einer Datenhaltungsschicht. Um die UI währenddessen nutzbar zu machen wird diese asynchron verwendet. Das Konzept von Promise wird hierbei verwendet. zum Beispiel wird ein Service entfernt ein HTTP.GET Request, dann führt dieser seine Aufgabe asynchron durch und schickt zu einem späteren Zeitpunkt das Ergebnis zurück.

Wie findet ihr das Framework persönlich?

Unsere Meinung ist ,dass es ganz cool ist. Dadurch das wir noch nicht so viel Erfahrung haben mit den anderen Frameworks können wir es noch nicht ganz sagen aber nach einer Zeit findet man sich hinein.

Was wollen wir den Mitschülern noch weiters über das Framework erklären.

- Dadurch das Angular einen eigenen Styleguide besitzt wird definiert wie Klassen, Komponenten oder Tests benannt werden sollen und das eine Klasse pro Datei definiert werden soll. Die Klassen werden dann ebenfalls sinnvoll benannt zum Beispiel ein Service NameOfService.service.ts benannt.