Webseite

xyz

Angular

Angular ist ein TypeScript-basiertes Framework, welches von Google für die komponentenbasierte Entwicklung von Single-Page-Anwendungen entwickelt wurde. Mithilfe von Angular lassen sich Teile bzw. Komponenten der Anwendung einfach austauschen und kombinieren, wodurch mit wenig Aufwand eine strukturierte und wartbare Anwendung erstellt werden kann.

Komponenten

Komponenten sind die Grundbausteine einer Angular-Anwendung und stellen eine unabhängige Einheit dar. Mit ihnen lassen sich Ansichten definieren; das ist der Bereich, den Nutzer letztendlich sehen. Eine Komponente besteht aus folgenden drei Teilen: die Typescript-Klasse, das Template und die Styles. Diese Dateien sollten immer gemeinsam in einem eigenen Ordner untergebracht werden, damit zu sehen ist, welche Dateien zu der Komponente gehören. Die Namen der Dateien werden durch den Komponentennamen definiert, jedoch tragen TypeScript-Klassen die Endung .ts, Templates .html und Styles .css. Der Befehl zum Erstellen einer neuen Komponente lautet: ng generate component NAME-DER-KOMPONENTE

Routing

Single-Page-Anwendungen werden vom Routing unterstützt. Routing ist das Aufspalten der Anwendung auf mehrere Seiten, die separat angefragt werden können.Beim Routing werden Ansichten der Anwendung abhängig von ihrem Zustand geladen. Dazu wird ein Router benötigt, der den Zustand der Anwendung verwaltet. Dieser Router ermöglich das Navigieren zwischen verschiedenen Ansichten, indem er die geladene Komponente automatisch mit einer anderen austauscht. Um den Router zu verwenden, müssen zunächst mehrere Komponenten vorhanden sein. Das heißt also, mehrere Ansichten, die im Browser angezeigt werden sollen. Darüber hinaus muss das Routing-Module, in dem der Router konfiguriert wird, erstellt und in die Anwendung eingebaut werden. Bei der Konfiguration des Routers wird einer zu ladenden Komponente ein URL-Pfad zugewiesen. Diese Zuordnung erfolgt mit der Routendefinition, bei der eine Route als Objekt notiert wird. In diesem Objekt wird der URL-Pfad und die Komponente, die durch diesen URL bzw. diese Route geladen werden soll, angegeben. Damit der Name der Komponente im Objekt verwendet werden kann, muss die Komponente importiert werden. Da sich aufgrund der vielen Komponenten auch mehrere Routen in einer Anwendung befinden, werden die Objekte bzw. Routendefinitionen in einem Array abgelegt. Für das Array wird der Typ Routes festgelegt, der aus dem Paket @angular/router importiert wird. Da z.B. kein Slash vor dem Pfad stehen darf, wird durch diese Typbindung sichergestellt, dass sich in dem Array nur wohlgeformte Routendefinitionen befinden. Wird die Anwendung gestartet, wird normalerweise die Root-URL / aufgerufen. Für diesen Pfad ist jedoch in Listing 12 noch keine Route definiert worden. Um die Anwendung mit der Information zu versorgen,

welche Route beim Aufruf der Root-URL geladen werden soll, muss für diesen URL eine Route angelegt werden. Durch das Routing werden Komponenten dynamisch geladen, weshalb eine stelle im Template festgelegt werden muss, an der die geladenen Komponenten eingesetzt werden sollen. Das geschieht mit der RouterOutlet-Direktive. Die Direktive ist Teil des Routing-Moduls und dient als Platzhalter, welcher dynamisch vom Router durch die geladene Komponente ersetzt wird. Sie kann nur eine Komponente aufeinmal anzeigen. Der RouterOutlet-Tag ist der Selektor für die RouterOutlet-Direktive. Um Links innerhalb der Anwendung verwenden zu können, wird anstelle des href-Attributs die RouterLink-Direktive genutzt. Der Router wird automatisch informiert eine neue Route zu laden, sobald der Link angeklickt wird. Der übrige teil der Anwendung bleibt bestehen und nur die zu ladende Komponente wird vom Server abgerufen. Um das Routing-Modul, in dem die Routendefinitionen stehen, nutzen zu können, wird das Routing-Modul in das Haupt-Angular-Modul importiert. Damit ist das Routing global in der Anwendung aktiviert.

Strukturdirektiven

Strukturdirektiven fügen Elemente hinzu oder entfernen diese, um die Struktur des DOM zu verändern (vgl. Built-in structural directives, o.D.). Zu den Strukturdirektiven gehört die NgForOf-Direktive, eine wiederholende Direktive. Sie wird in der Kurzform *ngFor genutzt und erleichtert die Arbeit mit Listen. Die Direktive erhält in der TypeScript-Datei einer Komponente ein Array als Parameter. Im Template wird ein HTML-Block definiert. Dieser Block wiederum definiert, wie ein einzelnes Element angezeigt werden soll und weist Angular an den HTML-Block als Vorlage zum Rendern jedes Elements in der Liste zu verwenden. Die Zeichenfolge nach dem *ngFor ist ein Mikrosyntax. Das ist eine eigene Sprache, die Angular interpretiert. Die Zeichenfolge row of rows bedeutet, dass jedes Element aus dem Array rows genommen, in der lokalen row Schleifenvariable gespeichert und es für jede Iteration dem HTML-Template zur Verfügung gestellt werden soll. Angular verwendet diese Vorlage dann wiederholt, um einen neuen Satz von Elementen in der Liste zu erstellen. Das Schlüsselwort let erstellt eine Eingabevariable namens row. Die ngFor-Direktive durchläuft das zurückgegebene Array rows und setzt row während jeder Iteration auf das aktuelle Element.