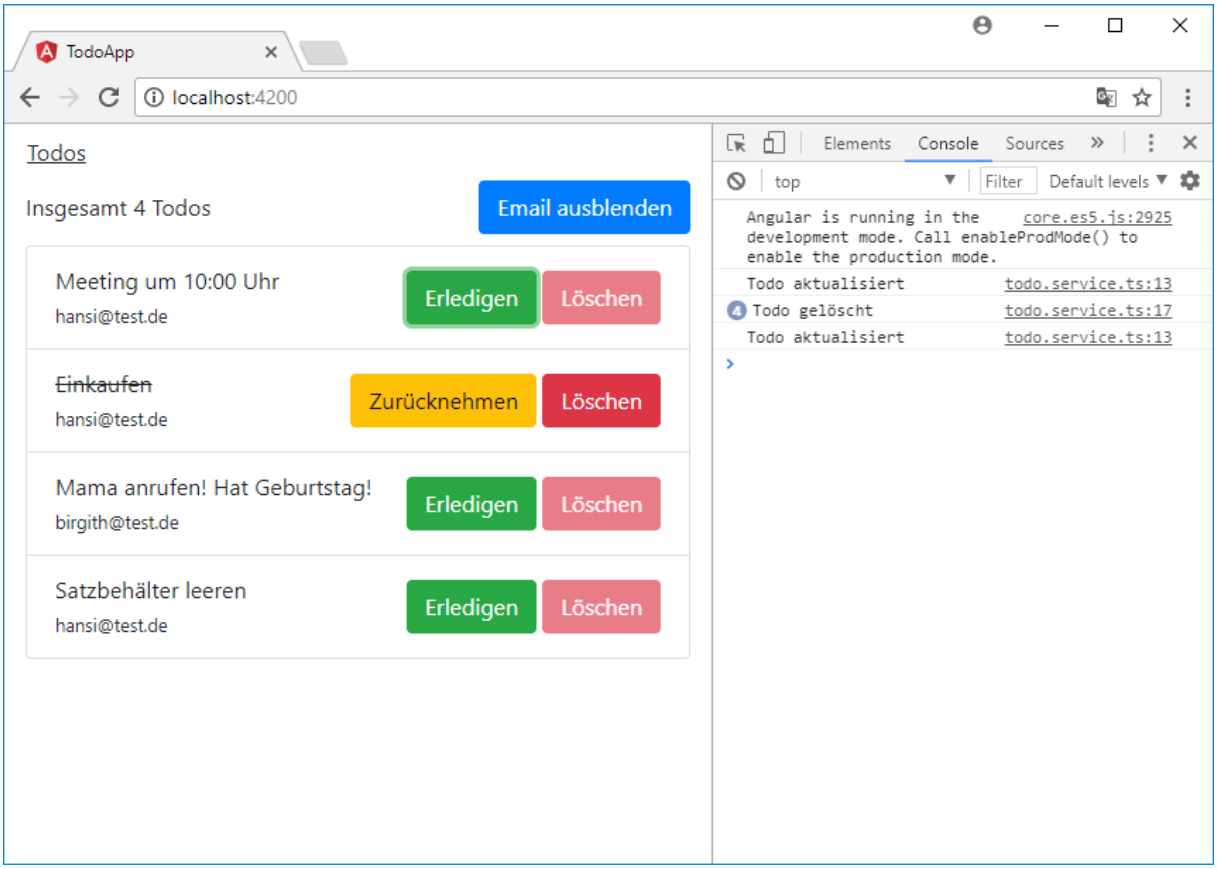
1. Services und DI [U-060]

> *Als Entwickler möchte ich eine Service Klasse in der Applikation vorfinden, welche die Zugriffe auf Daten der Todo-Liste steuert, um die Wartbarkeit der Applikation zu erhöhen und Datenkonsistenz sicherzustellen.*

1. Lernziele

- ✓ Sie verwenden Dependency Injection um eine Abhängigkeit zu initialisieren
- ✓ Sie beschreiben die Einsatzmöglichkeiten von Service-Klassen um Geschäftslogik zu kapseln.

2. Ergebnis



3. Benötigte Dateien

- `src/app/todo.service.ts`
- `src/app/app.module.ts`

- `src/app/todo-list.component.ts`

4. Anleitung

Erstellen einer Service-Klasse

Schritt 1: Erstellen einer neuen Service-Klasse `TodoService` und ihre Aufnahme in `AppModule`

1. Erstellen Sie die Datei `src/app/services/todo.service.ts` und erstellen Sie in dieser Datei die Klasse `TodoService.`
2. Ergänzen Sie die Klasse um den `@Injectable`-Dekorator. Hierdurch definieren Sie die Klasse als Singleton.
3. Übergeben Sie dem Dekorator ein Objekt mit dem Parameter `providedIn`, dem Sie den Wert '`root`' zuweisen.

Mock-Daten für TodoService

Schritt 2: Erstellen Sie eine Methode `getAll` in der Service-Klasse. Diese Methode gibt die Mock-Daten aus `src/app/shared/mock-todos.ts` zurück.

4. Definieren Sie die parameterlose Methode `getAll` mit dem Rückgabetyp `TodoItem[]` in der Service Klasse.
5. Importieren Sie die `mockTodos` analog zum Beispiel der `TodoListComponent`.
6. Geben Sie die `mockTodos` aus der Methode `getAll` zurück.

Schritt 3: Injizieren Sie den `TodoService` im Konstruktor der `TodoListComponent`.

7. Erweitern Sie die Klasse `TodoListComponent` um eine Konstruktorfunktion.

8. Spezifizieren Sie als Parameter des Konstruktors eine `private` Variable `todoService` vom Typ `TodoService`. Das Framework erstellt durch die Spezifizierung des Sichtbarkeitsmodifiers eine private Objektvariable namens `todoService` in der Serviceklasse. Über diese überall im Objekt sichtbare Referenz auf den Service erfolgen alle Zugriffe auf APIs oder wie hier Daten.

Schritt 4: Implementieren des `OnInit`-Interfaces mit der Methode `ngOnInit()`, in der `TodoListComponent`. Diese Methode ruft die `getAll`-Methode des Services auf und speichert den Rückgabewert in der Objektvariablen `todoList`. Entfernen des `mockTodo`-Imports.

9. Implementieren Sie in der Klasse `TodoListComponent` das `OnInit`-Interface aus `@angular/core`.
10. Erstellen Sie die für die Implementierung des Interfaces erforderliche Methode `ngOnInit()`.
11. Rufen Sie innerhalb der neu erstellten Methode die Service-Methode `getAll()` auf und weisen deren Rückgabewert einer neuen, `public` Objektvariablen namens `todoList` vom Typ `TodoItem[]` zu.
12. Löschen Sie die Zuweisung von `mockTodos` zu `todoList` und entfernen den zugehörigen `import` von `mockTodos`.

Schritt 5: Implementierung einer `delete()` Methode mit dem Übergabe-Parameter `todo` vom Typ `TodoItem` in `TodoService`, welche in `onDeleteClick` der `TodoListComponent` aufgerufen wird

13. Definieren Sie in der Klasse `TodoService` die Methode `delete` mit dem Parameter `todo` vom Typ `TodoItem`. Diese besitzt keinen Rückgabeparameter.
14. Verschieben Sie die Implementierung der Methode `onDeleteClick` der Komponente `TodoListComponent` in die neu erstellte Methode.
15. Rufen Sie die Methode `delete()` der Serviceklasse in der Methode `onDeleteClick` der Klasse `TodoListComponent` auf.

Schritt 6: Implementieren einer `update()`-Methode mit dem Parameter `todo` vom Typ `TodoItem` in `TodoService`. Diese Methode wird in `onCompleteClick` der `TodoListComponent` aufgerufen

16. Definieren Sie in der Klasse `TodoService` die Methode `update` mit dem Parameter `todo` vom Typ `TodoItem`
17. Führen Sie in der Methode ein `console.log` aus, welches die Nachricht „Todo aktualisiert“ ausgibt.
18. Fügen Sie einen Aufruf der Methode `update()` in der Methode `onCompleteClick` der Klasse `TodoListComponent` hinzu.

5. Kontrollfragen

- Wie werden Abhängigkeiten in Angular geladen?
- Welche Konzeptionellen Unterschiede bestehen zwischen Services und Komponenten?

6. Weiterführende Materialien

- Angular Dokumentation: Dependency Injection
  `https://tinyurl.com/gs-angular-di`