# Angular Dependency Injection Demo

## 1. Setup Project

### 1.1 Install Dependencies

1. Change directory to calab:

* cd calab

1. Install dependencies by running the following command:

* npm install

## 2. Creating Injectable Service

### 2.1 Create a new Course Service and Implement it’s logic

1. Create a new CourseService using CLI:

* npx -p @angular/cli ng generate service course/course

1. Open src/app/course/course.service.ts file and do the following:
   * Import Mock course data from the following file:
   * import COURSES from './MOCK\_COURSE\_DATA.json';
   * Add the following method just after the constructor:
   * getCourses(){  
      return COURSES;  
     }

### 2.2 Implement Course Model Class

1. Inside src/app/course folder create a new file called course.ts.
2. Open course.ts and add the following code:

* export class Course{  
   course\_id: number | undefined;  
   course\_name: string | undefined;  
   course\_description: string | undefined;  
   course\_duration: number | undefined;  
  }

### 2.3 Inject Course Service Into AppComponent

1. Open src/app/app.component.ts file and do the following:
   * Import Course model:
   * import { Course } from './course/course';
   * Inside AppComponent class declare variable called courses with type of list of Courses:
   * courses: Course[] = [];
   * Add the CourseService as a parameter into the constructor.
   * constructor(private courseService: CourseService){}
   * Fetch list of courses by calling getCourses() method from courseService and assign responce to variable declared above.
   * this.courses = courseService.getCourses();

### 2.4 Render List of Courses

1. Open src/app/app.component.html file and just below the <div class="divider"> add the following code that loops list of courses and renders each course in a list:

* <ul>  
   @for (course of courses; track course.course\_id) {  
   <li>  
   <div>  
   <p><strong>Id</strong> {{ course.course\_id }}</p>  
   <p><strong>Title: </strong>{{ course.course\_name }}</p>  
   <p><strong>Description: </strong>{{ course.course\_description }}</p>  
   <p><strong>Duration: </strong>{{ course.course\_duration }} weeks</p>  
    
   <div class="divider" role="separator" aria-label="Divider"></div>  
   </div>  
    
   </li>  
   }  
   </ul>

### 2.5 Start The Application

1. Start Angular Development Server if not yet started:

* npx -p @angular/cli ng serve --host 0.0.0.0
* *Otherwise refresh the browser tab to see updated view.*

1. Inspect the Rendered Screen, you should see list of courses on your screen.

## 3. Injecting services in other services

### 3.1 Create a new Logger and Implement it’s logic

1. Create a new Logger using CLI:

* npx -p @angular/cli ng generate service logger/logger

1. Open src/app/logger/logger.service.ts file and do the following:
   * Create logging methods just below the constructor :
   * log(msg: unknown) { console.log(msg); }  
     error(msg: unknown) { console.error(msg); }  
     warn(msg: unknown) { console.warn(msg); }

### 3.2 Inject Logger Service Into Course Service and Log When Courses are Fetched

1. Open src/app/course/course.service.ts file and do the following:
   * Import Logger service:
   * import { LoggerService } from '../logger/logger.service';
   * Add the CourseService as a parameter into the constructor.
   * constructor(private logger: LoggerService) { }
   * Inside getCourses() method log that courses are getting fetched.
   * this.logger.log('Fetching Courses');

## 4. Configuring Dependency Providers

### 4.1 Creating Enhanced Logger

1. Create a new TimedLoggerService using CLI:

* npx -p @angular/cli ng generate service logger/timed-logger

1. Open src/app/logger/timed-logger.service.ts file and do the following:
   * Extend LoggerService with TimedLoggerService:
   * export class TimeLoggerService extends LoggerService {...}
   * Override logging methods just below the constructor:
   * constructor() {  
      super()  
     }  
     override log(msg: unknown) {   
      const date = Date.now();  
      console.log(`${date}: ${msg}`);   
     }  
     override error(msg: unknown) {   
      const date = Date.now();  
      console.error(`${date}: ${msg}`);   
     }  
     override warn(msg: unknown) {   
      const date = Date.now();  
      console.warn(`${date}: ${msg}`);   
     }

### 4.2 Configure an app-wide provider in the ApplicationConfig of bootstrapApplication, it overrides one configured for root in the @Injectable() metadata.

1. Open app.config.ts file and add the following:
   * Update providers with the following:

* providers: [provideRouter(routes), provideClientHydration(), {provide: LoggerService, useClass: TimeLoggerService}]

### 4.3 Review Changes

1. Start Angular Development Server if not yet started:

* npx -p @angular/cli ng serve --host 0.0.0.0
* *Otherwise refresh the browser tab to see updated view.*

1. Inspect console and see whether your application logs with new Enhanced Timed Logger.

* 1714649534570: Fetching Courses