Progressive Part Practice Assignments

Difficulty: Easy

Set up the environment for your first Angular Application. Create an angular application and execute the app using ng serve in CLI.

Difficulty: Medium

Difficulty: Hard

Introduction to TypeScript

Difficulty: Easy

Write a typescript program to split a sentence and convert the first letter of each word to upper case. Display the result in console. (Use let instead of var).

Difficulty: Medium

Write a typescript program to check if a string is a palindrome or not. (Example for palindrome: madam, malayalam). Display the result in console.

Difficulty: Hard

Write a typescript program to check if two strings are anagrams. (Example for anagrams: tea and eat). Display the result in console.

Declaring Variables

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Types

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Type Assertions

Difficulty: Easy

Create an arrow function that accepts a string and convert the first letter of each word to upper case. Return the result string. Use type assertion wherever required.

Difficulty: Medium

Create an arrow function to check if a string is a palindrome or not. (Example for palindrome: madam, malayalam). Return the result. Use type assertion wherever needed.

Difficulty: Hard

Create an arrow function to check if two strings are anagrams. (Example for anagrams: tea and eat). Return the result and use type assertion wherever needed.

Arrow Functions

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Interfaces

Difficulty: Easy

Create a function named as calc1 that accepts a number as parameter. The function should find the square, cube and square root of the parameter. The result should be return as Result type. (Create an interface named as Result)

Difficulty: Medium

Create a function that accepts id, name and designation as parameters. The function should return an employee object. (Hint: use interface).

Difficulty: Hard

Create a function that accepts id, name, designation, department id, department name. The function should return an employee object that contains department object as aggregation. (Hint: use interfaces).

Classes

Difficulty: Easy

Create a typescript class "Greeter" and declare a field "title" of type string. As soon as the Greeter class is instantiated, the title must be initialized. Implement this using the constructor of the class. Create a function "greet" that displays the title in console. Create an instance of Greeter with a title "Hello Angular".

Difficulty: Medium

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title and name must be initialized. The age is optional. Implement this using the constructor of the class. Create a function "greet" that displays the title and name in console. Create an instance of Greeter with a title "Hello Angular".

Difficulty: Hard

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title and name must be initialized. The age is optional. Implement this using the constructor of the class. Create a function "greet" that displays the title and name in console. If age is not given, it should be initialized to 0. If given, the greet method should display age also. Test this method by executing the program.

Objects

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Constructors

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Access Modifiers

Difficulty: Easy

Create a typescript class "Greeter" and declare the "title" and "name" fields as public. Create a private property "age" whose value can be set only in the constructor. Later the value cannot be changed and hence it should be a read only property. (hint: create only getter and there is no setter for age in this class). Create an object and call the "greet" method that displays all the details in console.

Difficulty: Medium

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title, name age name must be initialized. The age is declared as private and others are public. Create a getter for age and there is no setter for age. Implement this using the constructor of the class. Create a function "greet" that displays the title, name and age in console. Create an instance of Greeter and call the "greet" method to display the details.

Difficulty: Hard

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title and name must be initialized. Implement this using the constructor of the class. All the fields are declared as private. Allow access to these properties using getters and setters. Make the age property as read only. Create a function "greet" that displays the title and name in console. If age is not given, it should be initialized to 0. If given, the greet method should display age also. Test this method by executing the program.

Properties

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Modules

Difficulty: Easy

Create a typescript class "Greeter" and declare the "title" and "name" fields as public. Create a private property "age" whose value can be set only in the constructor. Later the value cannot be changed and hence it should be a read only property. (hint: create only getter and there is no setter for age in this class). Create an object and call the "greet" method that displays all the details in console.

Export the class and import it in main.ts.

Difficulty: Medium

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title, name age name must be initialized. The age is declared as private and others are public. Create a getter for age and there is no setter for age. Implement this using the constructor of the class. Create a function "greet" that displays the title, name and age in console. Create an instance of Greeter and call the "greet" method to display the details.

Export the class as greeter.ts and import it from main.ts

Difficulty: Hard

Create a typescript class "Greeter" and declare the fields "title","name" and "age" inside the class. As soon as the Greeter class is instantiated, the title and name must be initialized. Implement this using the constructor of the class. All the fields are declared as private. Allow access to these properties using getters and setters. Make the age property as read only. Create a function "greet" that displays the title and name in console. If age is not given, it should be initialized to 0. If given, the greet method should display age also. Test this method by executing the program.

Export this class from greeter.ts and import it from main.ts

Progressive Part Practice Assignments

Difficulty: Easy

Create a module hrms and create Employee class with Department interface as an aggregation. Declare all the variables as private and create getters and setters and a constructor to initialized the fields. Create a function toString to display all the details. Export the class and interface to access it from another typescript file. Instantiate the class Employee and supply values to the constructor. Invoke the toString function to check the values in console.

Difficulty: Medium

Difficulty: Hard

Angular Fundamentals

Difficulty: Easy

Create an Angular project and create a component named "hello-world" and use it in app root. Use ng serve too view the app in browser.

Difficulty: Medium

Create an Angular project and create a component named "hello-world". Use template instead of templateUrl and display a h1 size heading in the component. Run the application and check output.

Difficulty: Hard

Create an Angular project and create a component named "hello-world" with more than one css in styleUrls. Execute the app to see the output.

Components

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Services

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Progressive Part Practice Assignments

Difficulty: Easy

Create an Angular project hrms and create Employee component and create a data entry form in the template. Use css to present the form. Run the application using ng serve and view the output in localhost:4200 in browser.

Difficulty: Medium

Difficulty: Hard

Displaying Data and Handling Events

Difficulty: Easy

Display the list of Employees in tabular format. Employee details include employeeId, firstName, lastName and dateOfJoining. Bind the DOM properties in html template to present the table cellpadding and bordering.

Difficulty: Medium

Display the list of Employees in tabular format. Employee details include employeeId, firstName, lastName and dateOfJoining. Bind the html attribute to border the table using "attr".

Difficulty: Hard

Display the list of Employees in tabular format. Employee details include employeeId, firstName, lastName and dateOfJoining. Bind the DOM properties in html template to present the table cellpadding. Bind the html attribute to border the table using "attr".

Class and Style Binding

Difficulty: Easy

Create a button in html template. Bind the class to give color to the button. Change the value in the component class to see the change in button color in browser.

Difficulty: Medium

Create multiple styles related to button and bind it to an array of styles in component class. Check browser for change in values of class.

Difficulty: Hard

Allow the user to choose the color, border, etc for a button. Bind the style to an array of values in component class and apply to css using style binding.

Event Binding and Filtering

Difficulty: Easy

Validate the login (uid: ravi pwd: raja) using event binding on a button click. Also validate while an "Enter" key is pressed, using filtering.

Difficulty: Medium

Validate the Sign up form. Confirm password should be checked with password while leaving the text box. Also validate while an "Enter" key is pressed from any input field using filtering.

Difficulty: Hard

Validate the Sign up form. Confirm password should be checked with password while leaving the text box. Password strength should be reflected in red, yellow, orange and green while typing itself. Also while an "Enter" key is pressed from any input field, the validations should happen using filtering.

Two-Way Binding and Template Variables

Difficulty: Easy

Create a reference (#) in the input control and access its value in the component function. Display its "value".

Difficulty: Medium

Implement two-way binding between email address form field and component class variable. Implement this by importing FormsModule from forms library.

Difficulty: Hard

Implement two-way binding between template variable and the component class variable. Bind Employee class variables eid, fname and lname with the template variables and display the json object in console.

Pipes

Difficulty: Easy

Accept the user input "date of Birth" and bind it to the class variable "dob". Display the dob in "dd-MMM-yyyy" format using appropriate filter.

Difficulty: Medium

Create a signup form with firstName, lastName, emailId, dateOfBirth. Bind them to the corresponding class variables. Display them in console as firstName in lowercase, lastName in UPPERCASE, dateOfBirth in "MM/dd/yy" format. Use appropriate filter.

Difficulty: Hard

Create a signup form with firstName, lastName, emailId, dateOfBirth. Bind them to the corresponding class variables. Display them as a JSON object in console . Also display them as firstName in lowercase, lastName in UPPERCASE, dateOfBirth in "MM/dd/yy" format. Use appropriate filter.

Custom Pipes

Difficulty: Easy

Display a list of Employees and allow the user to filter the list based on search pipe. Implement this by generating a custom pipe.

Difficulty: Medium

Display a list of Employees and allow the user to filter the list based on search pipe. Implement this by generating a custom pipe. The user input is always in lower case and the search should be case insensitive.

Difficulty: Hard

Display a list of Employees and allow the user to filter the list based on search pipe. Implement this by generating a custom pipe. The user input is either in lower or upper case and the search should be case insensitive.

Progressive Part Practice Assignments

Difficulty: Easy

Create an Employee master form with fields accepting user inputs for employeeId, firstName, lastName, dateOfJoining. Implement two-way binding with the component class properties and validate the firstName and lastName are not same. On submit, display the list of employees in tabular format below the form. Allow the user to filter the list by search on firstName in either case.

Difficulty: Medium

Difficulty: Hard

Building Re-usable Components

Difficulty: Easy

Create a Login component whose template shows User Name and Password input fields and submit reset buttons. Validate the fields for required and password length to be eight characters.

Difficulty: Medium

Create a Login component whose template shows User Name and Password input fields and submit reset buttons. Validate the form fields for required and password length to be eight characters. Authenticate the login with user name and password given in component class and display the result below the form.

Difficulty: Hard

Create a Login component whose template shows User Name and Password input fields and submit reset buttons. Validate the form fields for required and password length to be eight characters. Authenticate the login with user name and password given in component class and display the result in green color if success and red if failure, below the form.

Output Properties and Passing Event Data

Difficulty: Easy

In order to validate the custom login component, allow the component user to access the user name and password using output properites.

Difficulty: Medium

In order to validate the custom login component, all the component user to access the user name and password using output properties. The property should be a json. Use output decorator. Use templateUrl, styleUrls and encapsulation

Difficulty: Hard

In order to validate the custom login component, all the component user to access the user name and password using output properties. The property should be a json. Use output decorator and EventEmitter. Use templateUrl, styleUrls and encapsulation, ngContent and ngContainer to create the Login component.

Templates, Styles, and View Encapsulation

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

ngContent and ngContainer

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Progressive Part Practice Assignments

Difficulty: Easy

Create a custom component Signup to display sign up form elements in templateUrl, style it using styleUrls and use ngContent and ngContainer to allow access to the component properties.

Difficulty: Medium

Difficulty: Hard

Directives

Difficulty: Easy

Create an array of Employees in the component class. Display the employees in template in tabular format using \*ngFor directive.

Difficulty: Medium

Create an array of Employees in the component class. Display the employees in template in tabular format using \*ngFor directive. Display the first and last row in silver background.

Difficulty: Hard

Create an array of Employees in the component class. Display the employees in template in tabular format using \*ngFor directive. Display the odd rows in silver background and even rows in white background.

ngIf, Hidden Property and ngSwitchCase

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Creating Custom Directives

Difficulty: Easy

Create a custom directive to remove the unnecessary spaces in front and rear of the text typed in the firstName and lastName of the Employee form.

Difficulty: Medium

Create a custom directive to convert the text to a sentence case. A sentence case is such that the first letter of each word to be in upper case.

Difficulty: Hard

Create a custom directive for the password strength. When the user types the password, the text color should be in red, yellow, orange and green according to the password strength. This should happen in the keyup event.

Progressive Part Practice Assignments

Difficulty: Easy

Create custom directives to ensure the password strength is checked in each keyup event and also remove spaces typed in the firstName. Also the text box background color to be changed to blue when it gets focus and back to white when it loses focus.

Difficulty: Medium

Difficulty: Hard

Template-Driven Forms

Difficulty: Easy

Create a template driven form to accept the Login user name and password and use ngForm and ngModel to bind the data entered with the component.

Difficulty: Medium

Create a template driven form to accept sign up details and bind the form elements to the component class using ngForm and ngModel.

Difficulty: Hard

Create a template driven form to accept sign up details and bind the form elements to the component class using ngForm and ngModel. Validate the form and use styles to complete the form view.

ngForm and ngModelGroup

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Binding Data to Dropdown List

Difficulty: Easy

This video is missing

Difficulty: Medium

Difficulty: Hard

Progressive Part Practice Assignments

Difficulty: Easy

Create template driven signup and signin forms and bind the form with component using ngForm and ngModel directives.

Difficulty: Medium

Difficulty: Hard

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Whole Practice Assignment

Difficulty: Easy

Create an angular project for HRMS and create signup, signin, employee master form to enter the new employee details. Display the employees list in tabular format below the form from the employeeList array from component class. Allow user to search employee using filter search which should be case insensitive.

Difficulty: Medium

Difficulty: Hard

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard

Difficulty: Easy

Difficulty: Medium

Difficulty: Hard