Angular Interview Questions and Answers

Prepared by ChatGPT

July 1, 2025

Questions and Answers

1. **What are pipes in Angular? Give me simple examples.**
   Pipes are used to transform data in templates.
   Example:

   ```
   {{ 'angular' | uppercase }} → 'ANGULAR'
   {{ today | date:'dd/MM/yyyy' }}
   ```

   Common built-in pipes: uppercase, lowercase, date, currency, number.

2. **Example of property binding and event binding.**
   Property Binding:

   ```
   <input [disabled]="isDisabled">
   ```

   Event Binding:

   ```
   <button (click)="onClick()">Click Me</button>
   ```

3. **Simple toggle example.**
   Component:

   ```
   status = true;

   Disappear() {
     this.status = !this.status;
   }
   ```

   Template:

   ```
   <p [ngClass]="{'active': status}">This is an active interview!</p>
   <button (click)="Disappear()">Click ME</button>
   ```

4. **Two-way data binding example.**

```
<input [(ngModel)]="IsChecked" type="checkbox">
<p>{{ IsChecked ? 'Active' : 'Inactive' }}</p>
```

5. **Shortcut to save all files in VSCode.**
   Windows/Linux: `Ctrl + K` followed by `Ctrl + S`
   Mac: `Cmd + Option + S`

6. **What is constructor and example?**
   Constructor is a special method called when an instance of a class is created.
   Example:

```
class Person {
  name: string;
  constructor(name: string) {
    this.name = name;
  }
}
const person1 = new Person('Kiran');
console.log(person1.name); // Outputs: Kiran
```

7. **Why does console show object instead of just a value?**
   When logging an instance, console shows the object. Use:

```
console.log(totalRounds.rounds);
```

   to print the specific property.

8. **What is Dependency Injection?**
   Dependency Injection (DI) is a design pattern where Angular injects services or dependencies into components. It promotes loose coupling and easier testing.

9. **Angular Directives and Attribute Directives.**
   Directives add behavior to elements. Attribute directives change appearance or behavior. Examples include `[ngClass]`, `[ngStyle]`.

10. **Explain pipes in an interview.**
    Pipes are used to transform data directly in the template, like formatting dates, currency, or text cases.

11. **Examples of built-in pipes.**

- UpperCasePipe: `'angular' | uppercase` → 'ANGULAR'
- LowerCasePipe: `'ANGULAR' | lowercase` → 'angular'
- DatePipe: `today | date:'dd/MM/yyyy'`
- CurrencyPipe: `amount | currency:'USD'`
- DecimalPipe: `3.14159 | number:'1.2-2'` → 3.14

12. **What are Reactive Forms in Angular?**
    Reactive forms are model-driven forms created using `FormGroup` and `FormControl` in the component class.

13. **Explain Reactive Forms in an interview.**
    Reactive forms allow better control, dynamic validations, and scalability, as the form model is defined in the component class.

14. **Enhanced way to create forms with FormBuilder.**

```
constructor(private fb: FormBuilder) {
  this.loginForm = this.fb.group({
    username: ['', Validators.required],
    password: ['', Validators.required]
  });
}
```

15. **CLI command to create a new Angular project.**
    `ng new project-name`
    Example:

```
ng new my-first-app
cd my-first-app
ng serve
```

16. **What is Observable? How is it used in Angular?**
    Observable is an RxJS stream that emits data asynchronously. Angular uses it for HTTP calls and reactive programming.

17. **Explain Angular lifecycle hooks and important ones.**
    Lifecycle hooks are methods that get called at specific moments in a component's lifecycle. Important hooks:

    - ngOnInit - Initialization
    - ngOnChanges - On input property change
    - ngOnDestroy - Cleanup
    - ngAfterViewInit - After view initialization

18. **How to explain lifecycle hooks in an interview.**
    Lifecycle hooks are methods that let us run logic during component initialization, input changes, view initialization, or cleanup.

19. **Examples of ngOnInit, ngOnChanges, ngOnDestroy.**

    - ngOnInit - Fetch API data or initialize values.
    - ngOnChanges - Detect changes in @Input properties.
    - ngOnDestroy - Unsubscribe from observables or clear timers.

20. **How to share data between Angular components.**

    - Parent to Child - `@Input()`
    - Child to Parent - `@Output()` with `EventEmitter`
    - Sibling Components - Shared service using `Subject` or `BehaviorSubject`
    - Route Parameters - Using `ActivatedRoute`
    - Local Storage / Session Storage

21. **Fix error: Can't bind to 'message'.**
    Ensure:

    - Child component has `@Input() message` with correct case.
    - Child component is declared in the module.
    - Selector name matches usage.

22. **What is ViewChild and ContentChild?**
    `@ViewChild` accesses a template reference or child component declared in the same component's view.
    `@ContentChild` accesses projected content passed from a parent using `<ng-content>`.