DevCon 2016 – Bangalore

Date – 30th Nov – 2nd Dec

Introduction to Angular 2

Name of the Speaker : Naveen Pete

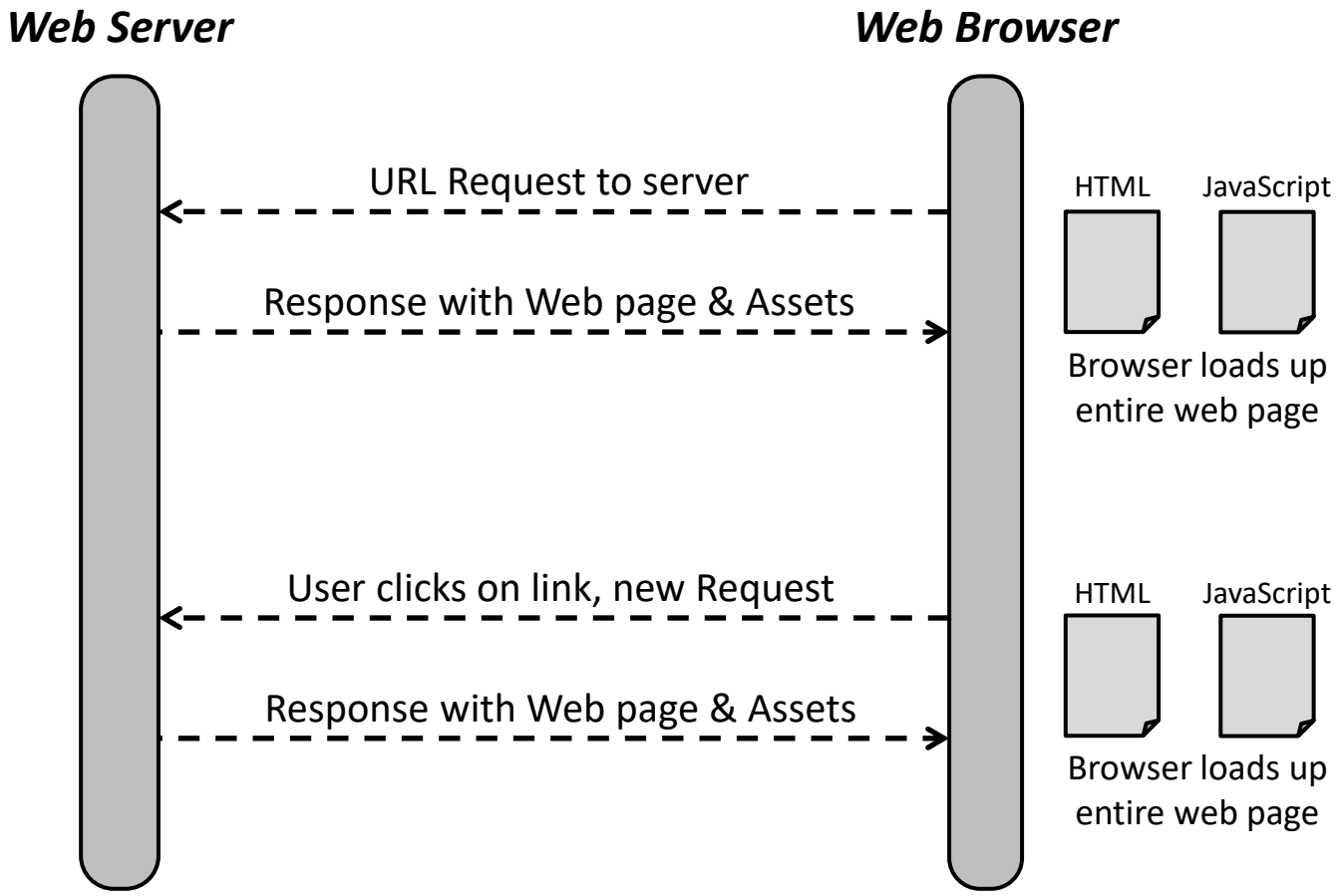
Place: Bengaluru

Agenda

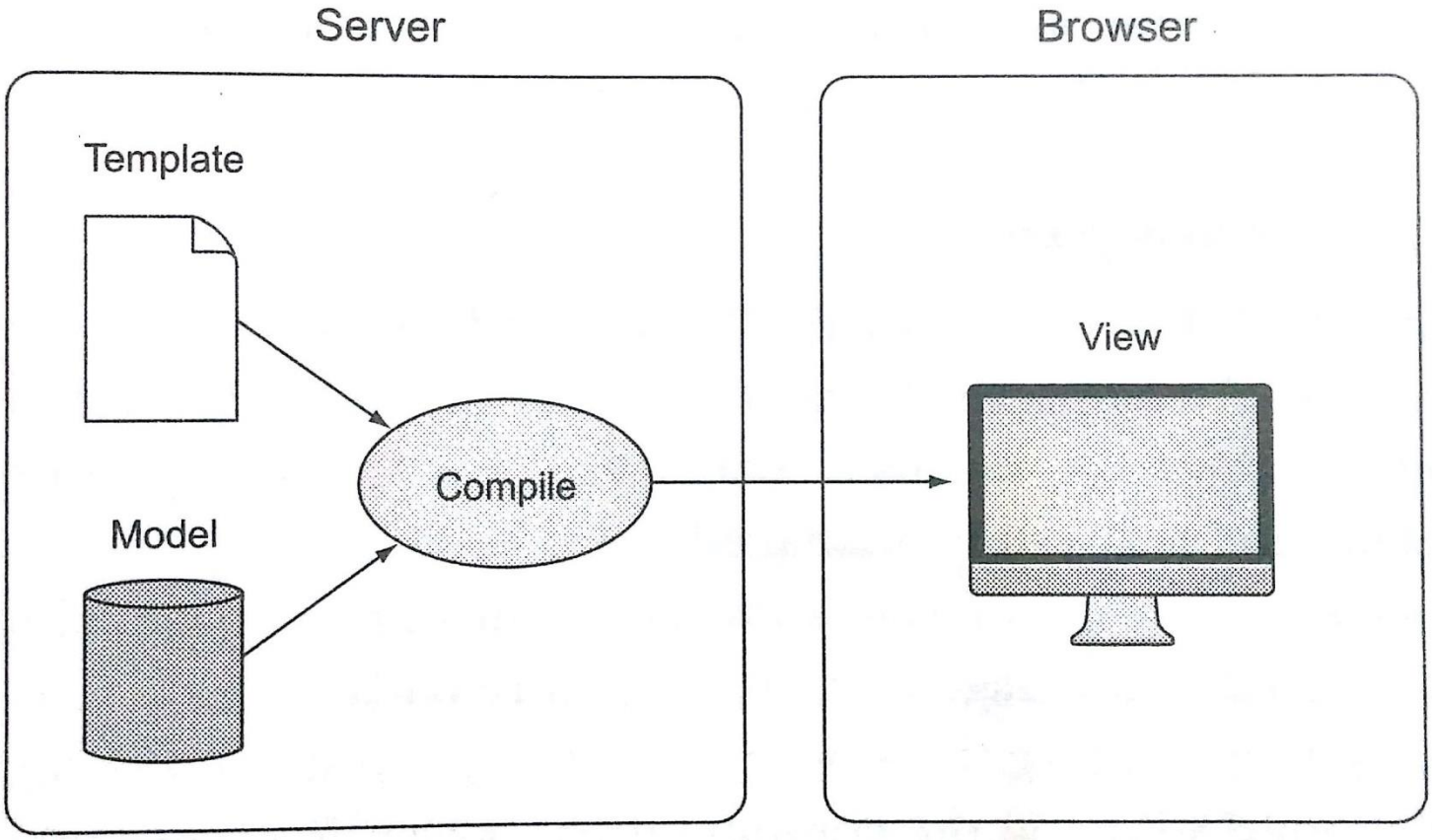
1. Why Library or Framework?
2. Introducing Angular
3. TypeScript
4. Setting up Angular 2
5. Angular 2 Building Blocks
6. Code Walk-thru & Demo
7. Q & A

1. Why Library or Framework?

Traditional Page Refresh



Data Binding



Benefits of Library or Framework

- Abstracts complexities of development
- Increases developer productivity
- Moving the application code forward in the stack
  - Reduces server load, thus reducing cost
  - Crowd-sourcing of computational power

2. Introducing Angular

Introducing Angular

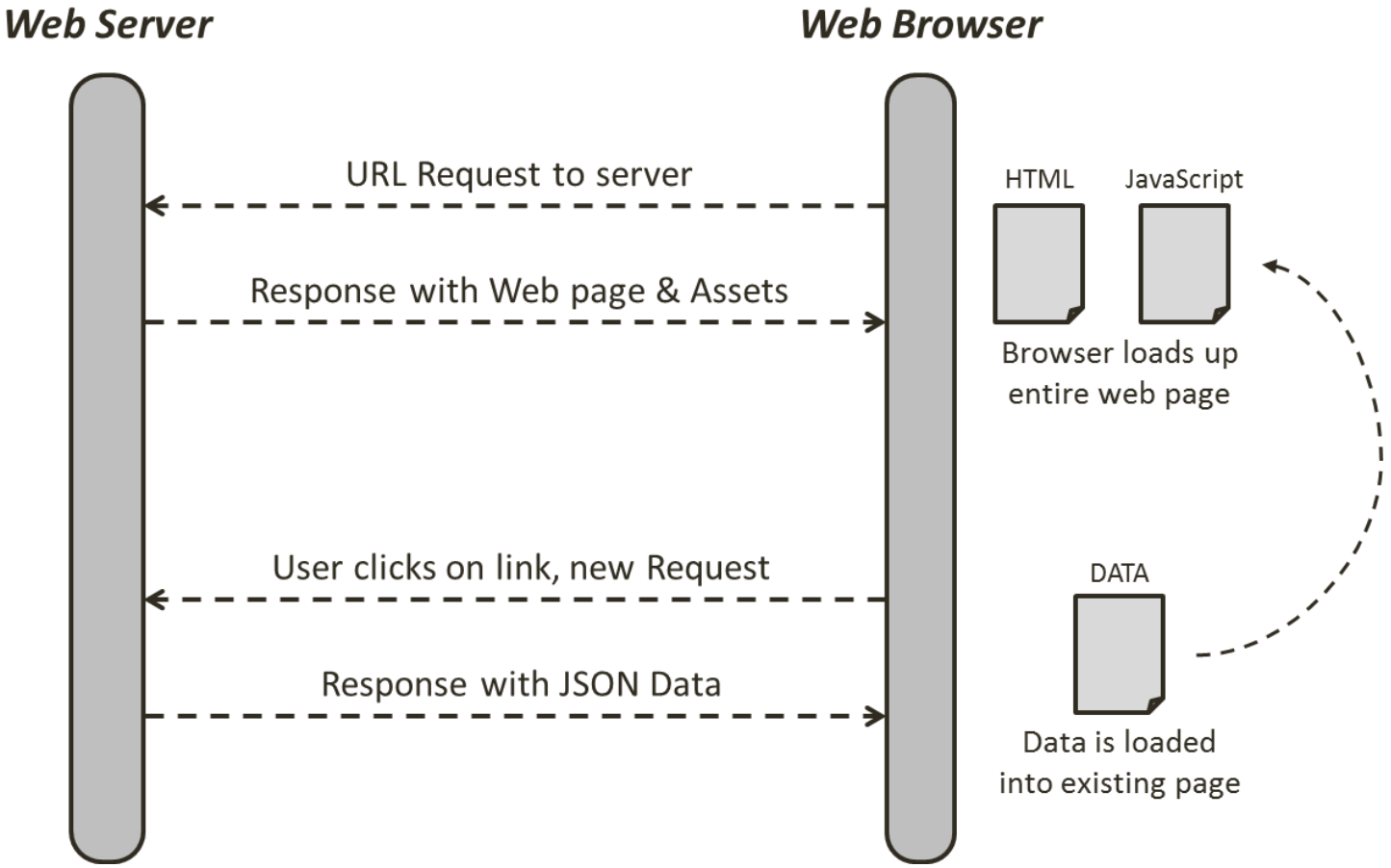
- Developed in 2009 by Misko Hevery
- Structural framework for building dynamic web apps
- Front-end SPA, RIA framework
- Build modular, maintainable and testable apps

- Current Release Versions
  - Angular 1.5.9
  - Angular 2.2

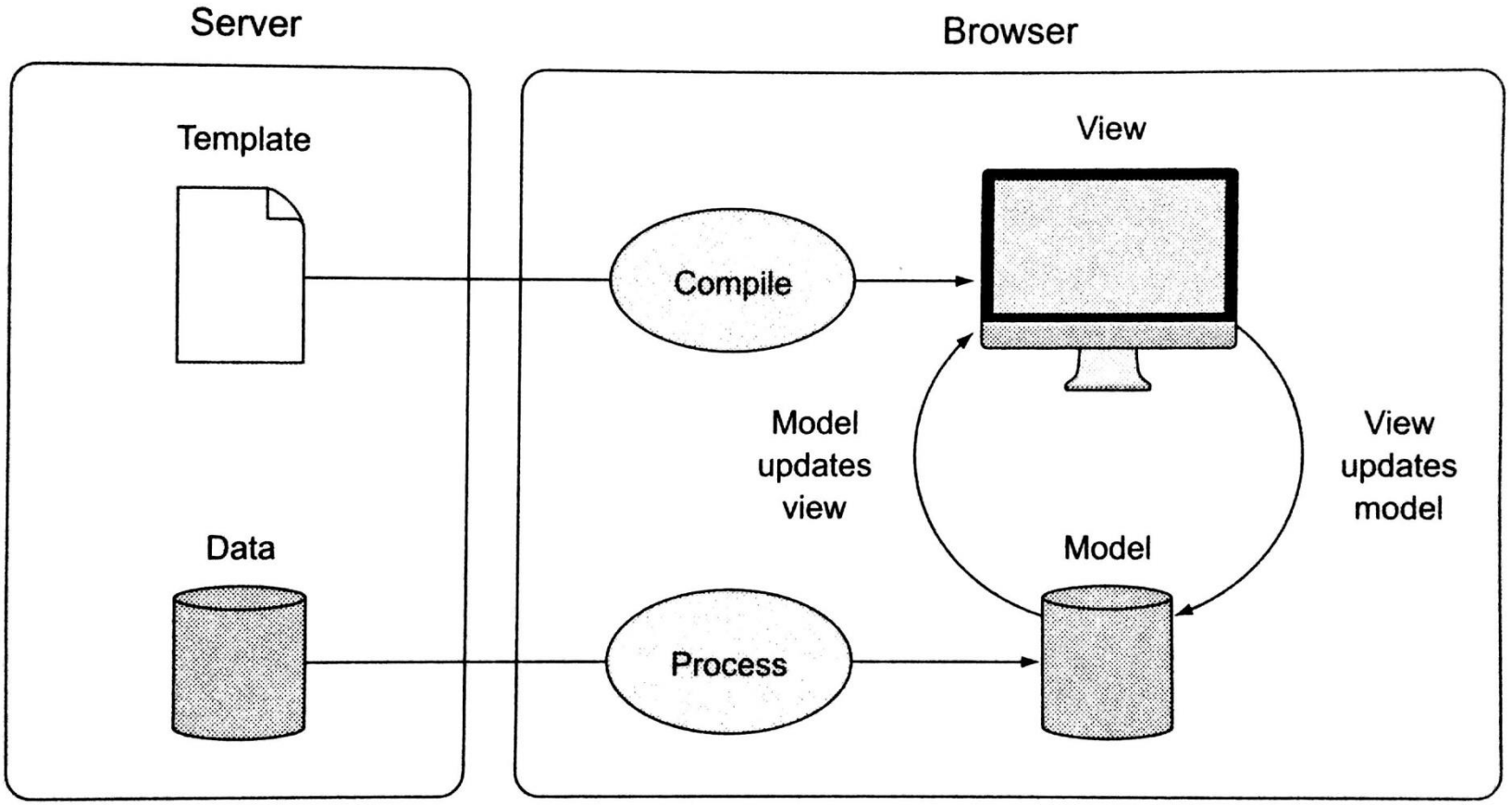
Angular Benefits

- Build responsive apps for different platforms
- Helps you organize your code
- Data Binding and Dependency Injection eliminates much of the manual code
- Decouple DOM manipulation from app logic
  - Improves testability
- Decouple client side of an app from the server side
  - Allows reuse
  - Allows parallel development

Responsive Page using Angular (or any other framework)



Two-Way Data Binding



3. TypeScript

TypeScript

- Superset of JavaScript
- Chosen as main language by Angular 2
- By far most documentation & example-base uses TypeScript

- Why TypeScript?
  - *Strong Typing*
    - reduces compile-time errors, provides IDE support
  - *Next Gen JS Features*
    - Modules, Classes, Import, Export, …
  - *Missing JS Features*
    - Interfaces, Generics, …

Install TypeScript

```
[sudo] npm install -g typescript
```

4. Setting up Angular 2

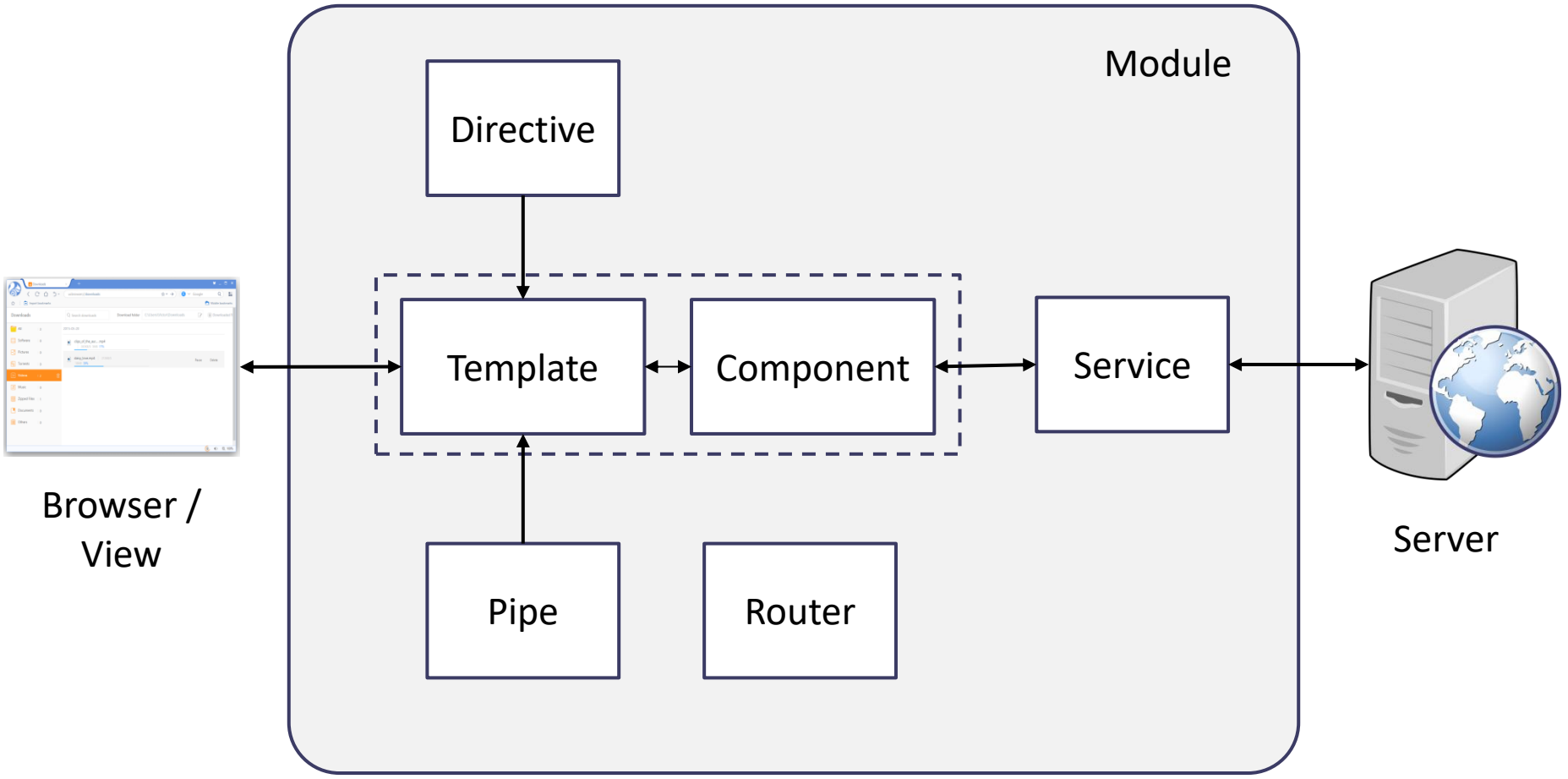
Setting up Angular

- Install Node (https://nodejs.org/en/)
- Install TypeScript – This is optional
- Install Angular CLI (https://cli.angular.io/)

- Create new Angular App
- Start the app
  - ng serve
    - Starts up development server
    - Builds the app

```
> npm install -g typescript
> npm install -g angular-cli
> ng new first-app
> cd first-app
> ng serve
```

5. Angular 2 Building Blocks


Module
Directive
Template
Component
Service
Pipe
Router
Browser / View
Server

Module (NgModule)

- A block of highly related classes
- Organizes an application into cohesive blocks of functionality
- Every app has at least one module, called AppModule

```
@NgModule({
   imports: [module1, module2, ...],
   declarations: [
      component(s), directive(s), pipe(s), ...
   ],
   exports: [class1, class2, ...],
   providers: [service1, service2, ...]
})
export class AppModule{ }
```

```
AppModule
```

Component

- Encapsulates the template, data and the behavior of a view
- Every app has at least one component, called AppComponent
- Completely decoupled from the DOM

Creating Component

```
> ng g c product
```

```
AppComponent
HeaderComponent
RecipesComponent
ShoppingListComponent
```

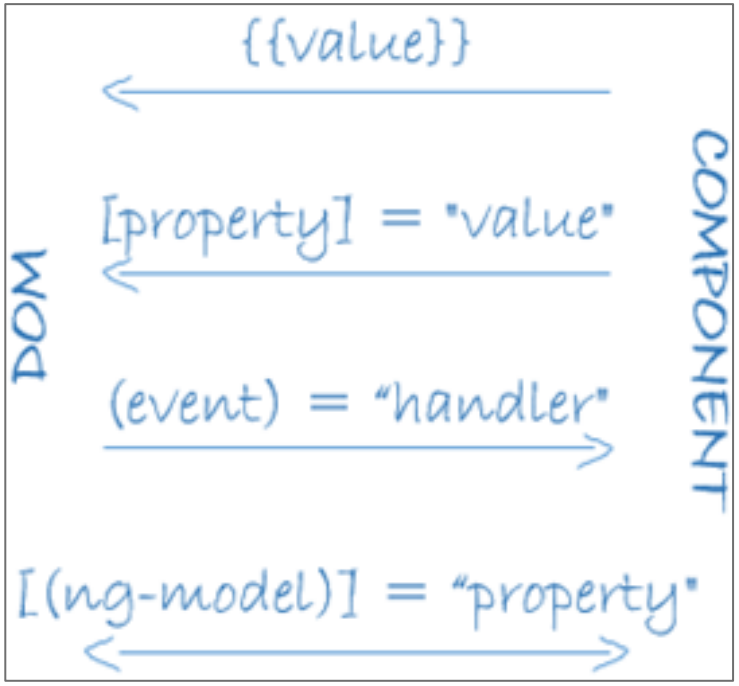
```
@Component({
    selector: 'rating',
    templateUrl: './rating.component.html',
    styleUrls: ['./rating.component.css']
})
export class RatingComponent {
    averageRating: number;

    setRating(value) {
        ...
    }
}
```

Template & Data Binding

- Defines component's view
- Uses HTML and Angular's template elements & attributes

- Data Binding
  - Interpolation
    - <h1>{{hero.name}}</h1>
  - Property binding
    - <img [src]="heroImageUrl">
  - Event binding
    - <li (click)="selectHero(hero)">
  - Two-way data binding
    - <input [(ngModel)]="hero.name">



Template & Data Binding

```
Examples:

ShoppingListComponent
RecipeItemComponent
RecipeDetailComponent
```

Directive

- Helps you to extend HTML to support dynamic behavior
- Transforms the DOM according to instructions given
- Can be built-in or custom
- Two kinds
  - Structural – alter the layout. E.g. *ngFor, *ngIf
  - Attribute – alter the appearance or behavior. E.g. ngModel, ngClass

Creating Custom Directive

```
> ng g d rating
```

```
@Directive({
   selector: '[appRating]`
})
export class RatingDirective {
   ...
}
```

```
ShoppingListComponent
ShoppingListAddComponent
```

Service

- Allows organizing and sharing code across an app
  - AJAX calls
  - Business rules
  - Calculations
  - Share data between components

Creating Service

```
> ng g s product
```

```
RecipeService
ShoppingListService
```

```
@Injectable()
export class ProductService {
    ...
}
```

Dependency Injection

- **Dependency**: An object that can be used (a service)
- **Injection**: Passing of a dependency to a dependent object so that it can use it. The client does not need to build the object
- Angular 2 uses constructor injection

```
@Component({
   selector: 'rb-recipe-list',
   templateUrl: './recipe-list.component.html‘,
   providers: [RecipeService]
})
export class RecipeListComponent implements OnInit {
   constructor(private recipeService: RecipeService) { }
}
```

```
RecipeListComponent
ShoppingListComponent
```

Pipe

- Transforms displayed values within a template
- Does not modify underlying data
- Built-in pipes
  - CurrencyPipe, DatePipe, JsonPipe, LowerCasePipe, UpperCasePipe

Creating Custom Pipe

```
> ng g p my-currency
```

Router

- Enables navigation from one view to the next as users perform application tasks
- Maps a URL path to a component

- Steps
  - Define array of routes using 'Routes'
  - Register routes with router module using 'Router.forRoot()'
  - Add the resulting module to 'imports' array of 'AppModule'
  - Add <router-outlet> element to the template
  - Use 'routerLink' attribute directive in <a> tag to navigate to a specific route
    - <a [routerLink]="['/recipes']">Recipes

```
app.routing.ts
app.module.ts
app.component.html
header.component.html
```

```
recipe-item.component.html
recipe-detail.component.ts
```

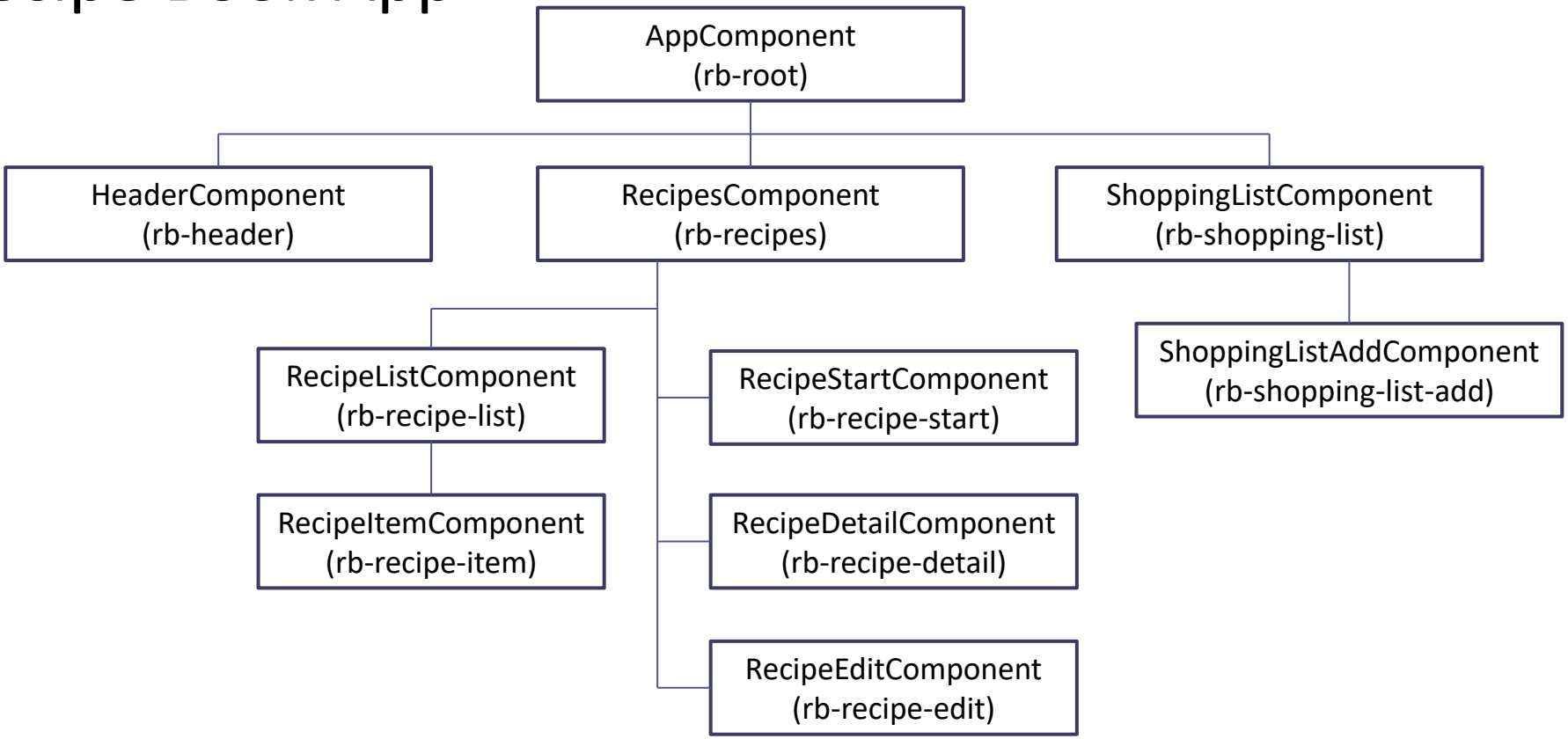
Server Communication – Angular Http Client

- Communicates with remote servers using HTTP protocol
- Uses browser's XmlHttpRequest object
- Methods
  - get()
  - post()
  - put()
  - delete()
- Methods return Observable<Response>

```
RecipeService
```

6. Code Walk-thru & Demo

Recipe Book App




```
https://github.com/naveen-pete/ng-2 (Project: recipe-book)
```

7. Q & A

THANK YOU

Speaker Name: Naveen Pete

Linked In: https://www.linkedin.com/in/naveen-pete

Organized by

UNICOM Trainings & Seminars Pvt. Ltd.
contact@unicomlearning.com