Angular

Binding, Routing, Directives

.NET CORE

Data-binding is a mechanism for coordinating what users see, specifically with application data values.

HTTPS://ANGULAR.IO/GUIDE/BINDING-SYNTAX#BINDING-SYNTAX-AN-OVERVIEW

Modeling – Data Binding

https://angular.io/guide/template-syntax#property-binding
https://angular.io/tutorial/toh-pt3#update-the-heroescomponent-template

The double curly braces ({{ }}) are ***Angular's*** interpolation binding syntax. This interpolation binding presents the component's property ***values*** inside the accompanying HTML Doc.

Property binding with [] around the property to be bound. This is one-way.

```
[class.selected]="hero === selectedHero"
```

Event binding based on events like 'click' or 'hover' to methods in the .ts file) using ().

```
<button (click)="addToCart(product)">Buy</button>
```

Two-Way Binding.

```
<input [(ngModel)]="hero.name" placeholder="name"/>
```

Angular Templates - Data Binding

https://angular.io/tutorial/toh-pt1#two-way-binding

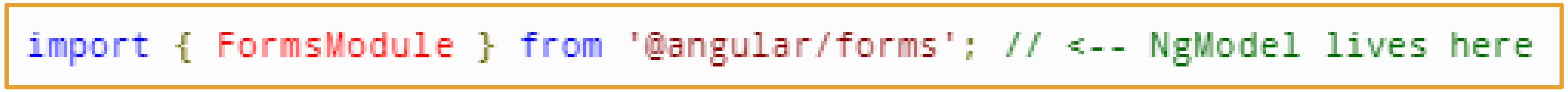
[(ngModel)] is Angular's two-way ***data binding*** syntax. It ***binds*** the property to the HTML so that data can flow in both directions: from the property to the textbox, and from the textbox back to the property.

ngModel isn't available by default do you have to import its module into the app. ***ngModel*** belongs to ***FormsModule*** and you have to opt-in (***import*** it) to use it.

@ngModule decorators have the needed metadata for the app to function.

The most important ***@NgModule decorator*** annotates the top-level ***AppModule*** class.

In app.module.ts, import the ***FormsModule***.



```
import { FormsModule } from '@angular/forms'; // <-- NgModel lives here
```

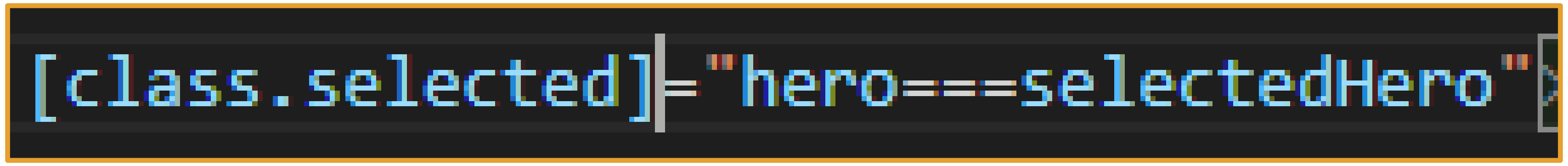
Then, add ***FormsModule*** to the imports array in the same file.

Angular Templates- Class Binding

https://angular.io/guide/template-syntax#class-binding

You can add and remove CSS class names from an element's class attribute with a ***class binding***.

To create a single ***class binding***, start with the prefix class followed by a dot (.) and the name of the ***CSS class*** ([class.foo]=“condition"). ***Angular*** adds the class when the bound expression is ***truthy***, and it removes the class when the expression is ***falsy***.

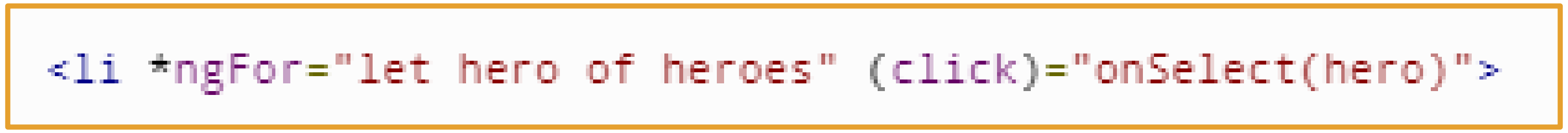


```
[class.selected]="hero===selectedHero"
```

Angular Templates - Event Binding

https://angular.io/tutorial/toh-pt2#add-a-click-event-binding
https://angular.io/guide/template-syntax#event-binding

The parentheses, (), around ***click*** tell ***Angular*** to listen for the **<*li*>** element's click event. When the user clicks in the **<*li*>**, ***Angular*** executes the onSelect(hero) expression in the components ***.ts*** file.



```
<li *ngFor="let hero of heroes" (click)="onSelect(hero)">
```

Angular dynamically changes the HTML ***template*** markup when the conditions change.

Modeling – Decorators

https://angular.io/guide/template-syntax#inputs-outputs
https://docs.angularjs.org/guide/decorators

Decorators are used to separate modification or decoration of a class without modifying the original source code. In AngularJS, ***decorators*** are functions that allow a service, directive or filter to be modified prior to its usage.

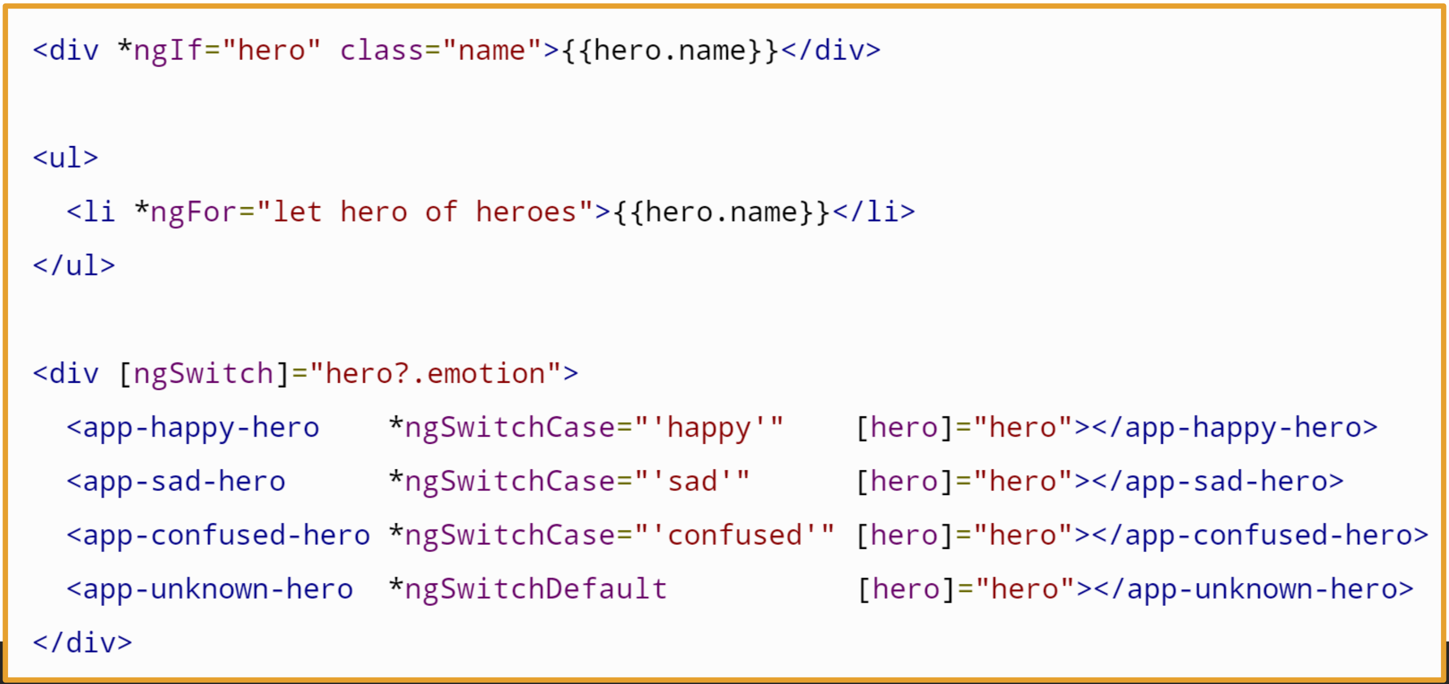
@Component - This indicates that the following class is a component. It provides metadata about the component, including its selector, templates, and styles.

- The selector identifies the component. The selector is the name you give the Angular component when it is rendered as an HTML element on the page. By convention, Angular component selectors begin with the prefix app-, followed by the component name.
- The template and style filenames reference the HTML and CSS files that StackBlitz generates.

Structural Directives

https://angular.io/api/common/NgIf
https://angular.io/api/common/NgForOf
https://angular.io/guide/template-syntax#ngSwitch
https://angular.io/guide/structural-directives

Structural directives shape or reshape the DOM's structure, typically by adding, removing, and manipulating the elements to which they are attached. Directives with an asterisk, *, are ***structural directives***.

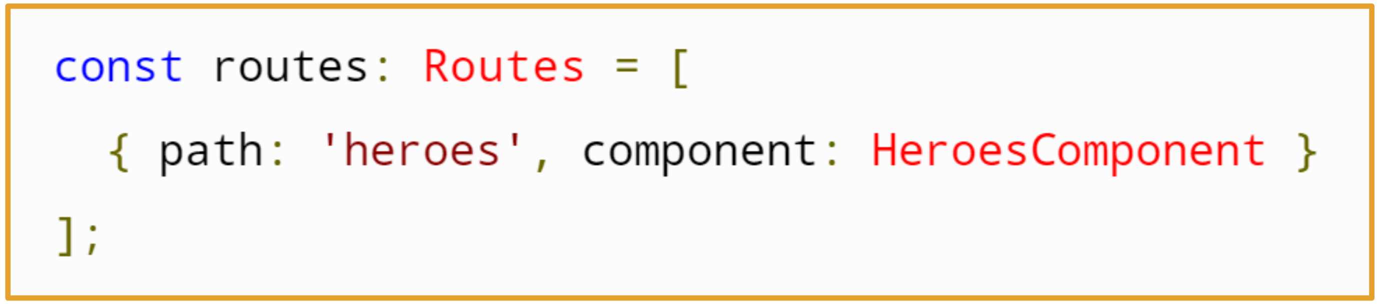


```
<div *ngIf="hero" class="name">{{hero.name}}</div>

<ul>
  <li *ngFor="let hero of heroes">{{hero.name}}</li>
</ul>

<div [ngSwitch]="hero?.emotion">
  <app-happy-hero    *ngSwitchCase="'happy'"    [hero]="hero"></app-happy-hero>
  <app-sad-hero      *ngSwitchCase="'sad'"      [hero]="hero"></app-sad-hero>
  <app-confused-hero *ngSwitchCase="'confused'" [hero]="hero"></app-confused-hero>
  <app-unknown-hero  *ngSwitchDefault           [hero]="hero"></app-unknown-hero>
</div>
```

Angular Routing

https://angular.io/start/start-routing
https://angular.io/guide/router
https://angular.io/start/start-data#services

Register a route in ***app.module.ts***. A route associates one (or more) URL paths with a component.



```
const routes: Routes = [
  { path: 'heroes', component: HeroesComponent }
];
```

The ***RouterLink*** directive in the .html template gives the ***router*** control over the ***anchor*** element. Put ***RouterLink*** in the <a> element where you want to redirect to another (registered) URL.

Inject the ***ActivatedRoute*** into the constructor of the component where it will be used. It is specific to each routed component that the Angular Router loads. By injecting the ***ActivatedRoute***, you are configuring the component to use a service.

Angular Routing

https://angular.io/start/start-routing
https://angular.io/guide/router

Routes tell the ***Router*** which view to display when a user clicks a link.

A typical Angular ***Route*** has two properties:

- path: a string that matches the URL in the browser address bar.
- component: the component that the router should create when navigating to this route.

- The @NgModule (in ***app-routing.module.ts***) metadata initializes the router and starts it listening for browser location changes.

```
imports: [ RouterModule.forRoot(routes) ],
```

The forRoot() method supplies the service providers and directives needed for routing, and performs the initial navigation based on the current browser URL

Routing Step-by-step

https://angular.io/tutorial/toh-pt5#add-the-approutingmodule

1. Add a module called app-routing with
   - ng generate module app-routing --flat --module=app
2. Make sure RouterModule and Routes are imported into app-routing.module with
   - import { RouterModule, Routes } from '@angular/router';
   - Also import whatever ***component*** (from its relative location) you will be routing to into app-routing.module.ts
3. You can delete the CommonModule references and declarations array.
4. Configure routes in const routes: Routes = [{ path:'link', component: AssociatedComponent }]; in app-routing.module
5. Add imports: [RouterModule.forRoot(routes)], under @NgModule.
6. Also under @NgModule add exports: [RouterModule].
7. Add <a routerLink="/[link]">NameOfLink</a> to whatever page you want to add a link to.
8. Add ...