|  |  |
| --- | --- |
| **Built-in directives** === **app.component.html** ====  <h1>**Built-in Directives**</h1>  <h2>**Built-in attribute directives**</h2>  <**h3**  id="**ngModel**">NgModel (two-way) Binding</**h3**>  <**fieldset**><h4>**NgModel examples**</h4>  <**p**>Current item name: {{**currentItem**.***name***}}</**p**>  <**p**>  <**label** **for**="***without***">**without NgModel:**</**label**>  <**input** [**value**]="**currentItem**.***name***" (**input**)="**currentItem**.***name***=$***event***.*target*.*value*" id="***without***">  <**/p**>  <**p**>  <**label** **for**="***example-ngModel***">**[(ngModel)]:**</label>  <**input** [(**ngModel**)]="**currentItem**.***name***" id="***example-ngModel***">  <**/p**>  <**p**>  <**label**  **for**="***example-bindon***">bindon-ngModel: </label>  <**input** **bindon-ngModel**="**currentItem**.***name***" id="***example-bindon***">  <**/p**>  <**p**>  <**label** **for**="***example-change***">(ngModelChange)="...name=$event":</label>  <**input** [**ngModel**]="**currentItem**.***name***" (**ngModelChange**)="**currentItem**.***name***=$*event*"  id="***example-change***">  <**/p**>  <**p**>  <**label**  **for**="***example-uppercase***">(ngModelChange)="setUppercaseName($event)"  <**input** [**ngModel**]="**currentItem**.***name***" (**ngModelChange**)="**setUppercaseName**($*event*)"  id="***example-uppercase***">  </**label**>  <**/p**>  <**/fieldset**>  <hr><h2 id="**ngClass**">**NgClass Binding**</h2>  <**p**>currentClasses is {{**currentClasses** | ***json***}}<**/p**>  <**div**  [**ngClass**]="**currentClasses**">This div is initially saveable, unchanged, and special.</div>  <**ul**>  <**li**>  <**label** **for**="***saveable***">**saveable**</label>  <**input** type="**checkbox**" [(**ngModel**)]="**canSave**" id="***saveable***">  </li>  <**li**>  <**label** **for**="***modified***">**modified:**</label>  <**input** type="**checkbox**" [**value**]="!**isUnchanged**" (**change**)="**isUnchanged**=!**isUnchanged**"  id="***modified***">  <**/li**>  <**li**>  <**label** **for**="***special***">special:  <**input** type="**checkbox**" [(**ngModel**)]="**isSpecial**" id="***special***"></label>  <**/li**>  <**/ul**>  <**button**  (**click**)="**setCurrentClasses**()">Refresh currentClasses<**/button**>  <**div** [**ngClass**]="**currentClasses**">  This div should be {{ **canSave** ? "": "not"}} saveable,  {{ **isUnchanged** ? "unchanged" : "modified" }} and  {{ **isSpecial** ? "": "not"}} special after clicking "Refresh".<**/div**>  <br><br>  *<!-- toggle the "special" class on/off with a property -->*  <**div** [**ngClass**]="**isSpecial** ? *'special'* : ''">This div is special</div>  <**div** **class**="*helpful study course*">Helpful study course</div>  <**div** [**ngClass**]="{'**helpful**':*false*, '**study**':*true*, '**course**':*true*}">Study course</div>  <!-- NgStyle binding -->  <hr><h3>**NgStyle Binding**</h3>  <**div** [**style**.font-size]="**isSpecial** ? '*x-large*' : '*smaller*'"> This div is x-large or smaller.  <**/div**>  <h4>[**ngStyle**] binding to currentStyles - CSS property names</h4>  <**p**>currentStyles is {{**currentStyles** | ***json***}}</p>  <**div**  [**ngStyle**]="**currentStyles**">  This div is initially italic, normal weight, and extra large (24px).  <**/div**>  <br>  <**label**>italic: <**input** type="**checkbox**" [(**ngModel**)]="**canSave**"></label> |  <**label**>normal: <**input**  type="**checkbox**" [(**ngModel**)]="**isUnchanged**"></label> |  <**label**>xlarge: <**input** type="**checkbox**" [(**ngModel**)]="**isSpecial**"></label>  <**button** (**click**)="**setCurrentStyles**()">Refresh currentStyles</button>  <br><br>  <**div** [**ngStyle**]="**currentStyles**">  This div should be {{ **canSave** ? "*italic*": "*plain*"}},  {{ **isUnchanged** ? "*normal weight*" : "*bold*" }} and,  {{ **isSpecial** ? "*extra large*": "*normal size*"}} after clicking "Refresh".</div>  <hr>  <h2>**Built-in structural directives**</h2>  <h3 id="**ngIf**">**NgIf Binding**</h3>  <**div**>  <**p**>If isActive is true, app-item-detail will render: </p>  <**app-item-detail** \***ngIf**="**isActive**" [**item**]="**item**">  <**/app-item-detail**>  <**button** (**click**)="**isActiveToggle**()">Toggle app-item-detail</button>  <**/div**>  <**p**>If currentCustomer isn't null, say hello to Laura:</p>  <**div** **\*ngIf**="**currentCustomer**">Hello, {{**currentCustomer**.***name***}}</div>  <**p**>nullCustomer is null by default. NgIf guards against null. Give it a value to show it:</p>  <**div**  **\*ngIf**="**nullCustomer**">  Hello, <**span**>{{**nullCustomer**}}</**span**>  <**/div**>  <**button** (**click**)="**giveNullCustomerValue**()">Give nullCustomer a value</button>  <h4>**NgIf binding with template (no \*)**</h4>  <**ng-template**  [**ngIf**]="**currentItem**">  Add {{**currentItem**.***name***}} with template  <**/ng-template**>  <hr>  <h4>**Show/hide vs. NgIf**</h4>  *<!-- isSpecial is true -->*  <**div** [**class**.*hidden*]="**!isSpecial**">Show with class</div>  <**div** [**class**.*hidden*]="**isSpecial**">Hide with class</div>  **+-**  <**p>***ItemDetail is in the DOM but hidden*</p>  <**app-item-detail** [**class**.*hidden*]="**isSpecial**"><**/app-item-detail**>  <**div** [**style**.*display*]="**isSpecial** ? *'block'* : *'none'*">Show with style</div>  <**div** [**style**.*display*]="**isSpecial** ? *'none'* : *'block'*">Hide with style</div>  <hr>  <h2 id="**ngFor**">**NgFor Binding**</h2>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items**">{{**item**.*name*}}</div>  <**/div**>  <**p**>\*ngFor with ItemDetailComponent element</p>  <**div** class="box">  <**app-item-detail** **\*ngFor**="let **item** of **items**" [**item**]="**item**">  <**/app-item-detail**>  <**/div**>  <h4 id="**ngFor-index**">**\*ngFor with index**</h4>  <p>with <i>**semi-colon**</i> **separator**</p>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items**; let **i**=**index**">  {{**i** + 1}} - {{**item**.*name*}}  </div>  <**/div**>  <p>with <i>**comma**</i> **separator**</p>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items**, let **i**=**index**">{{**i** + 1}} - {{**item**.*name*}}</div>  <**/div**>  <h4 id="**ngFor-trackBy**">**\*ngFor trackBy**</h4>  <**button** (**click**)="**resetList**()">Reset items</button>  <**button** (**click**)="**changeIds**()">Change ids</button>  <**button** (**click**)="**clearTrackByCounts**()">Clear counts</button>  <p><i>**without**</i> **trackBy**</p>  <**div** **class**="**box**">  <**div**  ***#noTrackBy*** **\*ngFor**="let **item** of **items**">({{**item**.*id*}}) {{**item**.*name*}}  </div>  <**div**  id="**noTrackByCnt**" **\*ngIf**="**itemsNoTrackByCount**" >  Item DOM elements change **#**{{**itemsNoTrackByCount**}} without trackBy  <**/div**>  <**/div**>  <p>**with trackBy**</p>  <**div** **class**="**box**">  <**div** ***#withTrackBy*** **\*ngFor**="let **item** of **items**; **trackBy**: **trackByItems**">  ({{**item**.*id*}}) {{**item**.*name*}}  </div>  <**div** id="**withTrackByCnt**" **\*ngIf**="**itemsWithTrackByCount**">  Item DOM elements change **#**{{**itemsWithTrackByCount**}} with trackBy  <**/div**>  <**/div**>  <br><br><br>  <p>**with trackBy and** <i>**semi-colon**</i> **separator**</p>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items**; **trackBy**: **trackByItems**">  ({{**item**.*id*}}) {{**item**.*name*}}  </**div**>  </**div**>  <p>**with trackBy and** <i>**comma**</i> **separator**</p>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items**, **trackBy**: **trackByItems**">({{**item**.*id*}}) {{**item**.*name*}}</div>  <**/div**>  <p>**with trackBy and** <i>**space**</i> **separator**</p>  <**div** **class**="**box**">  <**div** **\*ngFor**="let **item** of **items** , **trackBy**: **trackByItems**">({{**item**.*id*}}) {{**item**.*name*}}</div>  <**/div**>  <p>**with** <i>**generic**</i> **trackById function**</p>  <**div**  **class**="**box**">  <**div** \***ngFor**="let **item** of **items**, **trackBy**: **trackById**">({{**item**.*id*}}) {{**item**.*name*}}</div>  <**/div**>  <hr><h2>**NgSwitch Binding**</h2>  <p>**Pick your favorite item**</p>  <**div**>  <**label** **\*ngFor**="let **i** of **items**">  <**div**>  <**input**  type="**radio**" **name**="**items**" [(**ngModel**)]="**currentItem**" [**value**]="**i**">{{**i**.*name*}}  <**/div**>  <**/label**>  <**/div**>  <**div** [**ngSwitch**]="**currentItem**.*feature*">  <**app-stout-item** **\*ngSwitchCase**="**'stout'**" [**item**]="**currentItem**"><**/app-stout-item**>  <**app-device-item** **\*ngSwitchCase**="**'slim'**" [**item**]="**currentItem**"><**/app-device-item**>  <**app-lost-item** **\*ngSwitchCase**="**'vintage'**" [**item**]="**currentItem**"><**/app-lost-item**>  <**app-best-item** **\*ngSwitchCase**="**'bright'**" [**item**]="**currentItem**"><**/app-best-item**>  <**div** **\*ngSwitchCase**="**'bright'**"> Are you as bright as {{**currentItem**.*name*}}?<**/div**>  <**app-unknown-item** **\*ngSwitchDefault** [**item**]="**currentItem**"><**/app-unknown-item**>  <**/div**> | |
| ==== **item.ts** ====  **export** class **Item** {  static **nextId** = 0;  static **items**: **Item**[] = [  new **Item**( null, 'Teapot', 'stout' ),  new **Item**(1, 'Lamp', 'bright'),  new **Item**(2, 'Phone', 'slim' ),  new **Item**(3, 'Television', 'vintage' ),  new **Item**(4, 'Fishbowl')  ];  **constructor**(  public **id**?: number,  public **name**?: string,  public **feature**?: string,  public **url**?: string,  public **rate** = 100,  ) {  this.**id** = **id** ? **id** : **Item**.**nextId**++;  }  **clone**(): **Item** {  return **Object**.*assign*(new **Item**(), this);  }  } | |
| === **app.component.ts** ====  import { Component, OnInit } from '@angular/core';  **import** { **Item** } from './item';  @**Component**({  **selector**: 'app-root',  **templateUrl**: './app.component.html',  **styleUrls**: ['./app.component.css']  })  **export** class **AppComponent** *implements* **OnInit** {  **canSave** = *true*;  **isSpecial** = *true*;  **isUnchanged** = *true*;  **isActive** = *true*;  **nullCustomer** = *null*;  **currentCustomer** = {  **name**: *'Laura'*  };  **item**: **Item**; // defined to demonstrate template context precedence  **items**: **Item[]**;  **currentItem**: **Item**;  // trackBy change counting  **itemsNoTrackByCount** = 0;  **itemsWithTrackByCount** = 0;  **itemsWithTrackByCountReset** = 0;  **itemIdIncrement** = 1;  **currentClasses**: {};  **currentStyles**: {};  **ngOnInit**() {  this.**resetItems**();  this.**setCurrentClasses**();  this.**setCurrentStyles**();  this.**itemsNoTrackByCount** = 0;  }  **setUppercaseName**(***name***: string) {  this.**currentItem**.***name*** = ***name***.toUpperCase();  }  **setCurrentClasses**() {  // CSS classes: added/removed per current state of component properties  this.**currentClasses** = {  **saveable**: this.**canSave**,  **modified**: **!**this.**isUnchanged**,  **special**: this.**isSpecial**  };  }  **setCurrentStyles**() {  // CSS styles: set per current state of component properties  this.**currentStyles** = {  *'font-style'*: this.**canSave** ? *'italic'* : *'normal'*,  *'font-weight'*: **!**this.**isUnchanged** ? 'bold' : 'normal',  *'font-size'*: this.**isSpecial** ? *'24px'* : *'12px'*  };  }  **isActiveToggle**() {  this.**isActive** = **!**this.**isActive**;  }  **giveNullCustomerValue**() {  this.**nullCustomer** = *'Kelly'*;  }  **resetItems**() {  this.**items** = **Item**.**items**.*map*(**item** => **item**.*clone*());  this.**currentItem** = this.**items**[**0**];  this.**item** = this.**currentItem**;  }  **resetList**() {  this.**resetItems**();  this.**itemsWithTrackByCountReset** = 0;  this.**itemsNoTrackByCount** = **++**this.**itemsNoTrackByCount**;  }  **changeIds**() {  this.**items**.**forEach**(**i** => **i**.**id** += **1 \*** this.**itemIdIncrement**);  this.**itemsWithTrackByCountReset** = **-1**;  this.**itemsNoTrackByCount** = **++**this.**itemsNoTrackByCount**;  this.**itemsWithTrackByCount** = **++**this.**itemsWithTrackByCount**;  }  **clearTrackByCounts**() {  this.**resetItems**();  this.**itemsNoTrackByCount** = 0;  this.**itemsWithTrackByCount** = 0;  this.**itemIdIncrement** = 1;  }  **trackByItems**(***index***: number, ***item***: Item): number { return **item**.*id*; }  **trackById**(***index***: number, ***item***: any): number { return **item**.*id*; }  } | |
| ***https://stackblitz.com/angular/bbrjmoqrqqb?file=src%2Fapp%2Fapp.component.ts***  === **item-switch.component.ts** ==  import { Component, Input } from '@angular/core';  **import** { **Item** } from './item';  @**Component**({  selector: **'app-stout-item**',  template: `I'm a little {{**item**.***name***}}, short and stout!`  })  **export** class **StoutItemComponent** {  @**Input**() **item**: **Item**;  }  @**Component**({  selector: **'app-best-item'**,  template: `This is the brightest {{**item**.***name***}} in town.`  })  **export** class **BestItemComponent** {  @**Input**() **item**: **Item**;  }  @**Component**({  selector: **'app-device-item'**,  template: `Which is the slimmest {{**item**.***name***}}?`  })  **export** class **DeviceItemComponent** {  @**Input**() **item**: **Item**;  }  @**Component**({  selector: **'app-lost-item'**,  template: `Has anyone seen my {{**item**.***name***}}?`  })  **export** class **LostItemComponent** {  @**Input**() **item**: **Item**;  }  @**Component**({  selector: **'app-unknown-item'**,  template: `{{message}}`  })  **export** class **UnknownItemComponent** {  @**Input**() **item**: **Item**;  get **message**() {  *return* this.**item** && this.**item**.***name*** **?**  `${this.**item**.***name***} is strange and mysterious.` **:**  'A mystery wrapped in a fishbowl.'; }  }  **export** const **ItemSwitchComponents** =  [ **StoutItemComponent**, **BestItemComponent**, **DeviceItemComponent**, **LostItemComponent**, **UnknownItemComponent** ];  === **item-detail.component.ts** ===  import { Component, Input } from '@angular/core';  **import** { **Item** } from '../item';  @**Component**({  selector: **'app-item-detail'**,  templateUrl: './item-detail.component.html',  styleUrls: ['./item-detail.component.css']  })  **export** class **ItemDetailComponent** {  @**Input**() **item**: **Item**;  constructor() { }  }  === **item-detail.component.html** ===  <**div**>  <**span**>{{**item**?.***name***}}</**span**>  <**/div**> | |
| === **app.component.css** ===  **button** { font-size: 100%;  margin: 0 2px; }  **div**[ng-reflect-ng-switch], app-unknown-item {  margin: .5rem 0;  display: block;  }  #**noTrackByCnt**,  #**withTrackByCnt** { color: darkred;  max-width: 450px;  margin: 4px; }  **img** { height: 100px; }  .**box** { border: 1px solid black;  padding: 6px;  max-width: 450px; }  .**child-div** { margin-left: 1em;  font-weight: normal; }  .**context** { margin-left: 1em; }  .**hidden** {  display: none; }  .**parent-div** {  margin-top: 1em;  font-weight: bold; } | .**course** {  font-weight: bold;  font-size: x-large; }  .**helpful** {  color: red; }  .**saveable** {  color: limegreen; }  .**study**,  .**modified** {  font-family: "Brush Script MT";  font-size: 2rem; }  .**toe** {  margin-left: 1em;  font-style: italic; }  .**to-toc** {  margin-top: 10px;  display: block; } |