Directives

- reusable functions which can be used to change the behavior and appearance of DOM element
- Directives can implement all lifecycle hooks.
- Directives cannot have template

Directives

Structural

- used to change behavior of DOM
- may give performance issues
- contains *, *ngIf, *ngFor, *ngSwitch

Attribute

- widely used
- cannot add/remove element in DOM
- ngClass, ngStyle

Built-in directives

- *ngIf
- *ngFor
- *ngSwitch
- ngClass
- ngStyle

<div [hidden] = hideRooms> → The problem is it is still existing in your dom and if there are many such elements it may cause performance issues.

Therefore use ngIf instead.

```
* < div  *ngIf = "hidden == true">
           Display Rooms.
   </div>
```

Null coalescing (??)

Allows to use a kind of default value in place of another expression which is evaluated to null or undefined.

```
fullName?.length ?? -1 ;
```

⇒ return fullname.length if fullname exists else -1.

Optional chaining: (?.)

ability to immediately stop running an expression if a part of it evaluates to either null or undefined.

* **ngfor**

```
roomslist [  {no:1, type:'delux'},
             {no:2, type:'King'}
          ]

<tr   *ngfor="let room of roomslist, let i=index, let e=even, let o
                                                                =odd"
      <td> {{room.no}} </td>
      <td> {{room.type}}</td>
      <td> {{ i }} </td>
     <td> {{e ? "Even" : "Odd"}} </td>
```

* ngSwitch

```
   role = 'Admin'

  <div [ngSwitch]= "role>
     <div *ngSwitchCase = "'Admin'">
              welcome admin
     </div>
    <div *ngSwitchCase = "'User'">
              welcome User
    </div>
    <div   *ngSwitchDefault>
              Invalid role
    </div>
```

* ngClass

```
 .even {
      background-color: blue
      color: white
    }
 .odd
    {
      -- " ---
   }

 <tr [ngClass]=" e? 'even' : 'odd'"></tr>
```

* ngStyle

```
 <div [ngStyle] ="numRooms > 0 ? {'color':'green'} :
                                    {'color':'red'}"
```