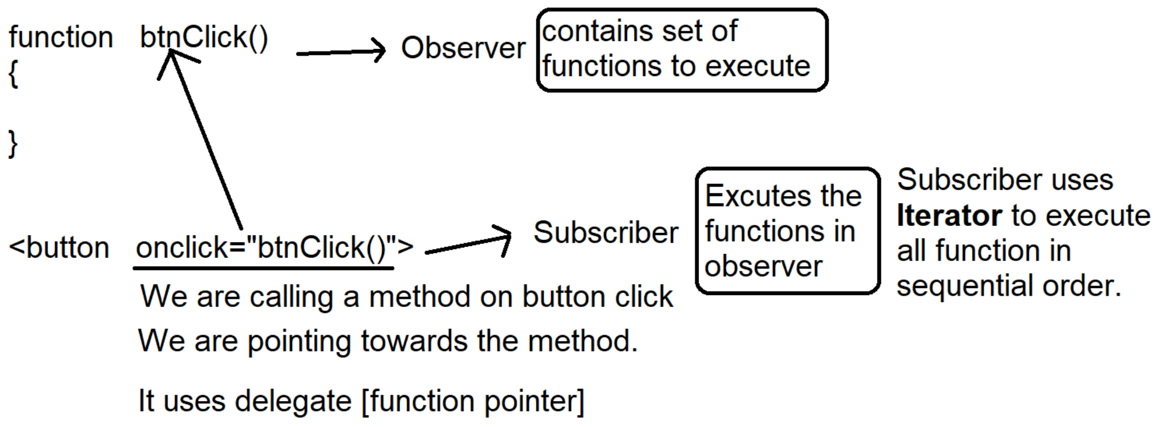
Angular Event Binding

What is Event?

- Event is a message sent by sender to its subscriber in order to notify the change.
- Event follows a software design pattern called "Observer".
- Observer is a communication pattern under "Behavioural Patterns".



function btnClick()
{
}
Observer
contains set of functions to execute
<button onclick="btnClick()">
Subscriber
Excutes the functions in observer
Subscriber uses **Iterator** to execute all function in sequential order.
We are calling a method on button click
We are pointing towards the method.
It uses delegate [function pointer]

- Angular implicitly controls all events by implementing "EventEmitter" base.
- You can create custom event by using EventEmitter.
- Angular uses all JavaScript browser events
    - Key Events
    - Mouse Events
    - Timer Events
    - Miscellaneous Events
- Angular events are bound to HTML element by using "Event Binding" technique

  Syntax:

  <button (click)="method()"> </button>

  <select (change)="method()"> </select>
- Angular provides only one event argument that is "$event"
- You can access event related properties by using "event" object.

- You can access object related properties by using “target” object.
  Syntax:

```
<button (click)=”method($event)”> </button>

public method(event) {
     event.eventProperties;
     event.target.objectProperties
}
```

Key Event Binding

- It defines the actions to perform when user is keying in the characters.
- The key events are

Event	Description
keyup	Specifies actions to perform when key is released
keydown	Specifies actions to perform when user hold down a key
keypress	Specifies actions to perform when user finish a key. [finish typing a key and use any another key]

- The key event properties
  - shiftKey
  - ctrlKey
  - altKey
  - keyCode
  - charCode
  - which

Note: Every key has a ASCII code you can get the code by using

- **keyCode [65-90 – A-Z]**

- **charCode**
- **which**

On Keyup and Keydown, the keyCode or Char code are not captured.

Key and char code are accessible to any every only on key press.

Ex:

Keydemo.component.ts

```
import { Component, OnInit } from '@angular/core';


@Component({
  selector: 'app-keydemo',
  templateUrl: './keydemo.component.html',
  styleUrls: ['./keydemo.component.css']
})
export class KeydemoComponent{
   users = [
     {UserName: 'john'},
     {UserName: 'john123'},
     {UserName: 'david'},
     {UserName: 'david_nit'}
   ];
   username;
   password;
   regExp = /(?=.*[A-Z])\w{4,10}/;
```

```
message = '';
validStyle = false;
invalidStyle = false;
showWarning = false;
passwordMessage = '';

VerifyUser(){
  for(var user of this.users){
    if(user.UserName == this.username) {
      this.message = 'User Name Taken - Try Another';
      this.invalidStyle = true;
      this.validStyle = false;
      break;
    } else {
     this.message = 'User Name Available';
     this.invalidStyle = false;
     this.validStyle = true;
    }
  }
}
VerifyCaps(e) {
   if(e.keyCode>=65 && e.keyCode<=90) {
    this.showWarning = true;
```

```
      } else {
        this.showWarning = false;
      }
  }
  VerifyPasswordStrength(){
      if(this.password.match(this.regExp)){
       this.passwordMessage = 'Strong Password';
      }  else {
          if(this.password.length<4) {
            this.passwordMessage = 'Poor Password';
          } else {
            this.passwordMessage = 'Weak Password';
          }
      }
  }
}
```

Keydemo.component.html

```
<div>
    <div class="form-register">
        <h2>Register User</h2>
        <div class="form-group">
            <label>User Name</label>
            <div>
            <input [ngClass]="{'valid-effect': validStyle, 'invalid-effect': invalidStyle}" (keyup)="VerifyUser()"  [(ngModel)]="username" type="text" class="form-control">
            <span [ngClass]="{'text-danger':invalidStyle, 'text-success': validStyle}">{{message}}</span>
          </div>
        </div>
        <div class="form-group">
          <label>Password</label>
          <div>
            <input [(ngModel)]="password" (keyup)="VerifyPasswordStrength()" (keypress)="VerifyCaps($event)" type="password" class="form-control">
            <div>
              <span *ngIf="showWarning" class="text-warning"> <span class="fa fa-exclamation-triangle"></span> Caps is ON </span>
            </div>
            <div>
              <span> {{passwordMessage}} </span>
            </div>
          </div>
        </div>
    </div>
</div>
```

Keydemo.component.css

```
.form-register {
   width: 300px;
   border:2px solid darkcyan;
   padding: 20px;
   margin:auto;
   margin-top: 20px;
   border-radius: 10px;
}
.valid-effect {
   border:2px solid green;
   box-shadow: 2px 3px 4px green;
}
.invalid-effect {
   border:2px solid red;
   box-shadow: 2px 3px 4px red;
}
```

Mouse Event Binding

- Angular can use all JavaScript mouse events to handle various interactions with regard to Mouse.
- The mouse events
  - mouseover
    - Specifies actions to perform when mouse pointer is over the element.
  - mouseout

        - Actions to perform when pointer leaves the element.
    - mousedown
        - Actions to perform when user hold down mouse button.
    - mouseup
        - Actions to perform when mouse button is released over element.
    - mousemove
        - Actions to perform when pointer is moving over element.
- The commonly used event properties
    - clientX
    - clientY

Ex:

Mousedemo.component.ts

```
import { Component, OnInit } from '@angular/core';


@Component({
  selector: 'app-mousedemo',
  templateUrl: './mousedemo.component.html',
  styleUrls: ['./mousedemo.component.css']
})
export class MousedemoComponent {
  styleObj = {
    'color': ''
  };
```

```
zoomObject = {
  'height': '',
  'width': ''
};
offerImage = 'assets/giftbox.png';
SetColor(e){
   this.styleObj = {
     'color': e.target.id
   };
}
startMarquee(e){
  e.target.start();
}
stopMarquee(e){
  e.target.stop();
}
zoomIN(){
  this.zoomObject = {
    'height': '400px',
    'width': '400px'
  };
}
zoomOut(){
  this.zoomObject = {
      'height': '100px',
      'width': '100px'
    };
  }
  showOffer(){
    this.offerImage = 'assets/offerbox.png';
  }
  hideOffer(){
    this.offerImage = 'assets/giftbox.png';
  }
}
```

Mousedemo.component.html

```
<div>
   <h2>Color Picker</h2>
   <div class="row" (mouseover)="SetColor($event)" style="width:
500px; margin:auto">
      <div class="col-4 bg-danger text-white" id="red">
         Red
      </div>
      <div class="col-4 bg-success text-white" id="green">
         Green
      </div>
      <div class="col-4 text-white" id="blue" style="background-color:
blue;">
```

```
          Blue
        </div>
    </div>
    <h2 align="center" [ngStyle]="styleObj" >Sample Text</h2>
    <h2>Mouse over and out</h2>
    <marquee (mouseover)="stopMarquee($event)"
(mouseout)="startMarquee($event)" scrollamount="10"
bgcolor="lightgreen">
        <img src="assets/shoe.jpg" width="100" height="100">
        <img src="assets/speaker.jpg" width="100" height="100">
        <img src="assets/shirt.jpg" width="100" height="100">
        <img src="assets/jeans.jpg" width="100" height="100">
    </marquee>
    <h2>Mouse Down and Up</h2>
    <div style="width: 400px;">
        <img [ngStyle]="zoomObject" (mousedown)="zoomIN()"
(mouseup)="zoomOut()" style="margin: auto;" src="assets/shoe.jpg"
width="100" height="100">
    </div>
    <h2>Mouse Down and Up</h2>
    <div>
        <img (mousedown)="showOffer()" (mouseup)="hideOffer()"
[src]="offerImage">
    </div>
</div>
```

Mousedemo.component.css

```
img {
   opacity: 0.8;
   margin-left: 30px;
}
```

Other JavaScript Events used by Angular

Event	Description
blur	Actions to perform when element loses the focus.
focus	Actions to perform when element gets focus.
change	Actions to perform when value is changed.
Cut	
Copy	
Paste	
Select	
Selectstart	
Submit	
Click	
Dblclick	
contextmenu	

Ex:

Evendemo.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-eventdemo',
  templateUrl: './eventdemo.component.html',
  styleUrls: ['./eventdemo.component.css']
})
export class EventdemoComponent{
  msg = '';
  username;
  products =
['assets/shoe.jpg','assets/shirt.jpg','assets/speaker.jpg','assets/earpo
ds.jpg'];
  imageSource = 'assets/shoe.jpg';
  sliderValue = 0;


  onFocus(){
    this.msg = 'Name in Block Letters';
  }
  onBlur(){
    this.msg = '';
    this.username = this.username.toUpperCase();
  }
  ChangeProduct(){
    this.imageSource = this.products[this.sliderValue];
  }
}
```

Evendemo.component.html

```
<div>
    <h2>User Name</h2>
    <div>
        <input (focus)="onFocus()" (blur)="onBlur()"
[(ngModel)]="username" placeholder="Name in Block Letters"
type="text" class="form-control">
        <span>{{msg}}</span>
    </div>
    <h2>Change</h2>
    <div>
        <label>Products</label>
        <input [(ngModel)]="sliderValue" (change)="ChangeProduct()"
min="0" value="0" max="3" type="range">
    </div>
    <div>
        <img [src]="imageSource" width="200" height="200">
    </div>
</div>
```