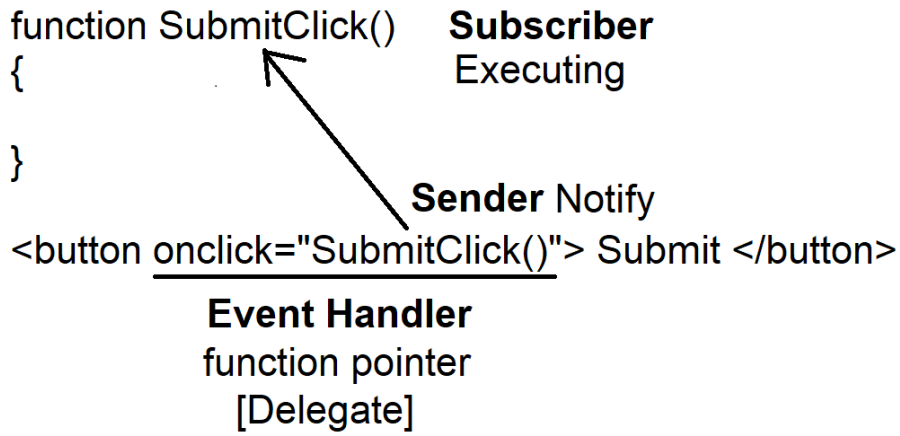
Angular Event Binding

- Event is a message sent by sender to its subscriber in order to notify the change.
- Event follows a software design pattern called "**Observer**".
- It is a common communication pattern [Behavioral Patterns] in software design patterns.



onclick is event that handles SubmitClick()

- Event handler uses "Delegate" mechanism. [Function Pointer]
- **Angular** can make use of all JavaScript Browser Events.
  - Key Events
  - Mouse Events
  - Timer Events
  - Miscellaneous Events
- **Angular** can bind any event by using "( )"
  Syntax:
  <button (click)="function()">
  <select (change)="function()">
- **Event** handler in JavaScript have 2 arguments to handle.

    - this: It can provide access to all members of current object.
    - event: It can provide access to all members of current event.
      Syntax
      <button onclick=”btnClick(this, event)”> Submit </button>
      this – sends button related details to the function like: value, id, name etc.
      event – sends onclick related details to the function like: clientX, altKey, screenX etc.
- **Event** handler in Angular can have only one argument by default “$event”
- **$event** directly provides access to event members.
- If you want to access object related members then you have to use “target” object of “$event”.
  $event.target.objectProperties/methods
  $event.eventProperties/methods

Ex:

Eventdemo.component.css

```
img:hover{
   cursor:grab;
}
```

Eventdemo.component.ts

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

```
@Component({
  selector: 'app-eventdemo',
  templateUrl: './eventdemo.component.html',
  styleUrls: ['./eventdemo.component.css']
})
export class EventdemoComponent{
  public buttonDetails;
  public eventDetails;
  public RegisterClick(obj){
     this.buttonDetails = obj.target;
     this.eventDetails = obj;
  }
  public ImageClick(e){
    if(e.ctrlKey){
      window.open('assets/shoe.jpg','Nike','width=400
height=300');
    } else {
      alert('Use Ctrl + Click');
    }
  }
}
```

Eventdemo.component.html

```
<div class="container">
  <h2>Event Demo</h2>
  <button (click)="RegisterClick($event)" class="btn btn-primary" name="btnRegister" value="Register">
    Register
  </button>
  <h2>Button Details</h2>
  <dl>
    <dt>Name</dt>
    <dd>{{buttonDetails.name}}</dd>
    <dt>Value</dt>
    <dd>{{buttonDetails.value}}</dd>
  </dl>
  <h2>Event Details</h2>
  <dl>
    <dt>X Position</dt>
    <dd>{{eventDetails.clientX}}</dd>
    <dt>Ctrl Key</dt>
    <dd>{{eventDetails.ctrlKey}}</dd>
  </dl>
  <img (click)="ImageClick($event)" src="assets/shoe.jpg" width="50" height="50">
```

```
    <p>Use Ctrl + Click to View Large</p>
</div>
```

- Angular uses all JavaScript browser events.
- Angular also provides events to handle various interactions. These angular events are defined with “Ng” reference, like “ngSubmit”.
- Angular Events are derived from “**EventEmitter”** base class.
- You can configure and create custom events by implementing EventEmitter.
- EventEmitter can emit any type of value for specific situation, which is used as Event Argument.
- Angular requires custom events for custom components.

Angular Events

- Key Events
- Mouse Events
- Timer Events
- Miscellaneous Events

Key Event Binding

- You can configure various functionalities based on different key events.
- Angular identifies the keystrokes and the characters user keying in and can perform functionality accordingly.
- Several interactions in application are managed on key events.

- The key events are:

Event	Description
keyup	It specifies the actions to perform when key up on any element. [Key is released over element].
Keydown	It specifies action to perform when user hold down the key.
keypress	It specifies the actions to perform when user finish a key.

- The key events are frequently used with event properties like

Property	Description
keyCode	It returns the actual key code. Ex: A=65, Z=90
charCode	It returns the character code as per UTF standards. Every key doesn't have a char code.
which	It is similar to keyCode but used with different keyboard layout.
shiftKey	Return true when shift used
altKey	Return true when Alt used.
ctrlKey	Return true when ctrl used.

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{
   public users = [
     {UserName: 'john'},
     {UserName: 'john12'},
     {UserName: 'john_nit'},
     {UserName: 'david'}
   ];
   public txtUserName;
   public userMsg;
   public isUserValid = false;
   public isUserInvalid = false;

   public password;
   public showPasswordWarning = false;
```

```
public VerifyUserName(){
  if(this.txtUserName.length<=3){
    this.userMsg = 'Name too short..';
    this.isUserInvalid = true;
    this.isUserValid= false;
  } else {
    for(var user of this.users){
      if(user.UserName==this.txtUserName){
        this.userMsg = 'User Name Taken - Try Another';
        this.isUserInvalid = true;
        this.isUserValid = false;
        break;
      } else {
        this.userMsg = 'User Name Available';
        this.isUserInvalid = false;
        this.isUserValid = true;
      }
    }
  }
}
public VerifyPassword(e){
  if(e.keyCode>=65 && e.keyCode<=90) {
```

```
      this.showPasswordWarning = true;
    } else {
      this.showPasswordWarning = false;
    }
  }
}
```

Keydemo.component.html

```
<div class="container-fluid">
  <h2>Register User</h2>
  <div class="form-group">
    <label>User Name</label>
    <div>
      <input (keyup)="VerifyUserName()" [(ngModel)]="txtUserName" type="text" class="form-control">
      <span [ngClass]="{'text-danger':isUserInvalid, 'text-success':isUserValid}">{{userMsg}}</span>
    </div>
  </div>
  <div class="form-group">
    <label>Password</label>
    <div>
```

```
      <input (keypress)="VerifyPassword($event)"
[(ngModel)]="password" type="password" class="form-
control">
      <div *ngIf="showPasswordWarning">
        <span class="fa fa-exclamation-triangle text-
warning"></span>
        <span class="text-warning">CAPS is ON</span>
      </div>
   </div>
  </div>
</div>
```

Keydemo.component.css

```
.container-fluid {
   margin: auto;
   padding:20px;
   width: 300px;
   align-items: center;
   justify-content: center;
}
label {
   font-weight: bold;
}
```