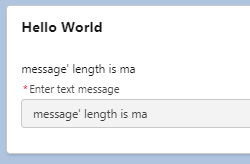
1



**“message” variable nın uzunluğu 20 karakteri geçerse disabled olacak**

<template>

  <lightning-card title="Hello World">

    <div class="slds-p-around\_medium">

    <p>{message} </p>

        <lightning-input

            label={inputLabel}

            value= 'type here'

            placeholder={placeholderVariable}

            required="true"

            onchange={changeHandler}

            disabled={isDisable}>

        </lightning-input>

    </div>

  </lightning-card>

</template>

import { LightningElement } from 'lwc';

export default class HelloWorld extends LightningElement {

    // global variable

  message = 'Hello';

  inputLabel = 'Enter text message';

  placeholderVariable = "Please enter message here......";

  isDisable = false;

  changeHandler(event){

     this.message = event.target.value;//Hello Vishal welcome

     if(event.target.value.length > 20){

        this.isDisable = true;

     }

  }

}

2

**Hesap makinası yapıldı. Onchange, onclick kullanıldı**

<br/>

            <lightning-button

                label="Addition"

                class="slds-p-around\_medium"

                title="Addition Button"

                variant="brand"

                onclick={addHandler}>

            </lightning-button>

            <lightning-button

                label="Subtraction"

                class="slds-p-around\_medium"

                title="Subtraction Button"

                variant="brand"

                onclick={subHandler}>

            </lightning-button>

            <lightning-button

                label="Devide"

                class="slds-p-around\_medium"

                title="Devide Button"

                variant="brand"

                onclick={divHandler}>

            </lightning-button>

            <lightning-button

                label="Multiplication"

                class="slds-p-around\_medium"

                title="Multiplication Button"

                variant="brand"

                onclick={mulHandler}>

            </lightning-button>

            <template if:true={isRef}>

            <lightning-button

                label="Reresh"

                class="slds-p-around\_medium"

                title="Fresh Button"

                variant="brand"

                onclick={refHandler}>

            </lightning-button>

            </template>

        </div>

    </lightning-card>

</template>

<template>

    <lightning-card title="Calculator">

        <!--{num1}===={num2}-->

        <div class="slds-p-around\_medium">

            <lightning-input

                label="Enter first number"

                type="number"

                required="true"

                placeholder="enter your first number"

                onchange={firstHandler}

                value ={num1}>

            </lightning-input>

            <lightning-input

                label="Enter second number"

                type="number"

                required="true"

                placeholder="enter your second number"

                onchange={secondHandler}

                value = {num2}>

            </lightning-input>

        <br/>

            <lightning-input

                label="Result"

                type="number"

                disabled="true"

                placeholder="Result"

                value = {result}>

            </lightning-input>

        <br/>

            <template if:true={isAdd}>

                Addition of two number : {result}

            </template>

            <template if:true={isSub}>

                Subtraction of two number : {result}

            </template>

            <template if:true={isDiv}>

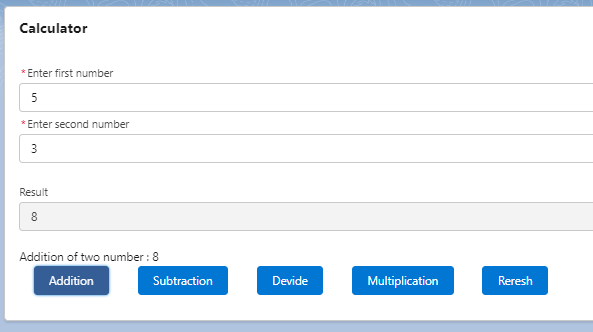
                Division of two number : {result}

            </template>

            <template if:true={isMul}>

                Multiplication of two number : {result}

            </template>



mulHandler(){

        this.result = parseInt(this.num1) \* parseInt(this.num2);

        this.isMul = true;

        this.isAdd = false;

        this.isDiv = false;

        this.isSub = false;

        this.isRef = true;

    }

    divHandler(){

        this.result = parseInt(this.num1) / parseInt(this.num2);

        this.isDiv = true;

        this.isAdd = false;

        this.isMul = false;

        this.isSub = false;

        this.isRef = false;

        this.isRef = true;

    }

    refHandler(){

        //alert('Hi i am refresh method');

        this.num1 = null;

        this.num2 = null;

        this.isAdd = false;

        this.isSub = false;

        this.isDiv = false;

        this.isMul = false;

        this.isRef = false;

    }

    firstHandler(event){

        this.num1 = event.target.value;

    }

    secondHandler(event){

        this.num2 = event.target.value;

    }

}

import { LightningElement, track } from 'lwc';

export default class Calculator extends LightningElement {

    @track num1;

    @track num2;

    @track result;

    @track isAdd = false;

    @track isSub = false;

    @track isMul = false;

    @track isDiv = false;

    @track isRef = false;

    addHandler(){

        this.result = parseInt(this.num1) + parseInt(this.num2);

        this.isAdd = true;

        this.isDiv = false;

        this.isMul = false;

        this.isSub = false;

        this.isRef = true;

    }

    subHandler(){

        this.result = parseInt(this.num1) - parseInt(this.num2);

        this.isSub = true;

        this.isAdd = false;

        this.isDiv = false;

        this.isMul = false;

        this.isRef = true;

    }

<template>

    <lightning-card >

        <div class="slds-p-around\_medium">

        {firstNameUstte}

            <lightning-input

                label="First Name"

                value="enter first name here.."

                onchange={firstNameMethod}>

            </lightning-input>

        {lastNameUstte}

            <lightning-input

                label="Last Name"

                value=""

                onchange={lastNameMethod}

                placeholder="type here">

            </lightning-input>

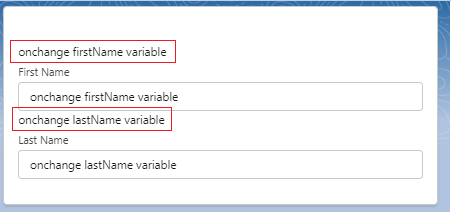
        </div>

    </lightning-card>

</template>

**onchange kullanıldı**

3



import { LightningElement } from 'lwc';

export default class EnginViewMessageText extends LightningElement {

    firstNameUstte;

    lastNameUstte;

    firstNameMethod(event){

        this.firstNameUstte = event.target.value;

    }

    lastNameMethod(event){

        this.lastNameUstte = event.target.value;

    }

}

<template>

    <lightning-card title="If condition in LWC">

        <div class="slds-p-around\_medium">

            <lightning-input

                type ="checkbox"

                label="click me"

                onchange={changeHandler}>

            </lightning-input>

            <template if:true={checkBoxValue}>

            <p>Hi i am true condition</p>

            </template>

            <template if:false={checkBoxValue}>

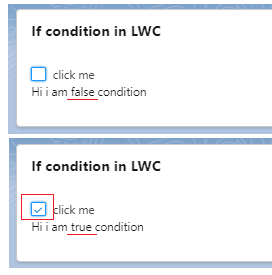
            <p>Hi i am false condition</p>

            </template>

        </div>

    </lightning-card>

</template>



**IF condition – check box tıklandığında açıklamada true çıksın, tıklanmadığında false çıksın**

4

import { LightningElement } from 'lwc';

export default class MyIfCondition extends LightningElement {

    checkBoxValue = false;

    changeHandler(event){

        this.checkBoxValue = event.target.checked; // true

    }

}

<template>

    <lightning-card title="For Each Loop">

        <div class="slds-p-around\_medium">

<template

for:each={accList} for:item="acc">

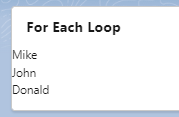
      <p key={acc}>{acc}</p>

      </template>

        </div>

    </lightning-card>

</template>



5

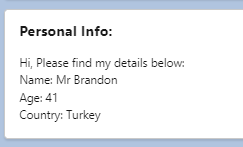
**For Each – Bir İsim Listesindeki isimleri alt alta yazdır**

import { LightningElement } from 'lwc';

export default class ForEachLoop extends LightningElement {

    accList = ['Mike','John', 'Donald'];

}



import { LightningElement } from 'lwc';

export default class MyInfo extends LightningElement {

    info = {firstName : 'Mr',

            lastName : 'Brandon',

            Age : '41',

            Country : 'Turkey'};

}

<template>

    <div class="myColor">

    <lightning-card title="Personal Info:" >

        <div class="slds-m-around\_medium">

            <p>Hi, Please find my details below:</p>

            <p>Name: {info.firstName} {info.lastName}</p>

            <p>Age: {info.Age}</p>

            <p>Country: {info.Country}</p>

        </div>

    </lightning-card>

    </div>

</template>

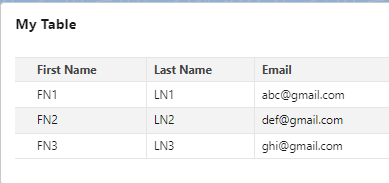
**Bir kişinin bilgileri bir object içerisinde first name last name age oluşturulup html de çağrıldı**

6

**For Each –**

**Çoklu liste çağrıldı**

7



<template>

    <lightning-card title = "My Table">

        <div class = "slds-p-around\_medium">

            <table class="slds-table

slds-table\_cell-buffer

                        slds-table\_bordered

                        slds-table\_col-bordered

                        slds-table\_striped">

                <thead>

                    <tr>

                        <th>First Name</th>

                        <th>Last Name</th>

                        <th>Email</th>

                    </tr>

                </thead>

                <tbody>

<template for:each={allPeopleInfo} for:item ="person">

         <tr key={person}>

         <td>{person.firstName}</td>

         <td>{person.lastName}</td>

         <td>{person.Email}</td>

         </tr>

</template>

                </tbody>

            </table>

        </div>

    </lightning-card>

</template>

import { LightningElement } from 'lwc';

export default class MultipleInfo extends LightningElement {

    allPeopleInfo = [

                    {firstName : 'FN1', lastName : 'LN1', Email : 'abc@gmail.com'},

                    {firstName : 'FN2', lastName : 'LN2', Email : 'def@gmail.com'},

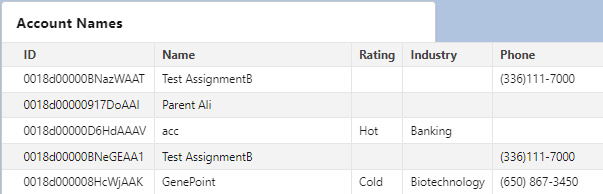
                    {firstName : 'FN3', lastName : 'LN3', Email : 'ghi@gmail.com'},

                    ]

}

**WIRE decorator. Bir class tan SOQL query ile bazı değerleri çağırıp org ta bir component ta gösterme**

8



**class**

public with sharing class getAllAccountsRecords {

    @AuraEnabled(cacheable=true)

    public static List<Account> getRecords(){

        List<Account> lstAccount = new List<Account>();

        lstAccount = [select Id, Rating, Name, Industry, Phone from Account limit 10];

        //

        // Account acc = new Account();  // burası hata için yazıldı. Çünkü dml yapılamaz

        // insert acc;

        return lstAccount;

    }

}

**js**

import { LightningElement, wire } from 'lwc';//Step 1: import Apex method

import getRecords from '@salesforce/apex/getAllAccountsRecords.getRecords';

export default class WireAProperty extends LightningElement {

    //Step 2: wire function execute when page loads

    @wire(getRecords) acc; // herhangi isim olabilir bu variable.

    // it will return 2 objects{data, Error}

    // Positive: 10 Accounts ==> data(10 Account Records), error (blank)

    // Negative it vill error ==> Error(Error Message), data(blank)

}

**html**

<template>

    <lightning-card title="Account Names">

        <template if:true={acc.data}>

            <table class="slds-table

                        slds-table\_cell-buffer

                        slds-table\_bordered

                        slds-table\_col-bordered

                        slds-table\_striped">

                <thead>

                    <tr>

                        <th>ID</th>

                        <th>Name</th>

                        <th>Rating</th>

                        <th>Industry</th>

                        <th>Phone</th>

                    </tr>

                </thead>

                <tbody>

    <template for:each={acc.data} for:item="each">

                        <tr key={each}>

                        <td>{each.Id}</td>

                        <td>{each.Name}</td>

                        <td>{each.Rating}</td>

                        <td>{each.Industry}</td>

                        <td>{each.Phone}</td>

                        </tr>

                    </template>

                </tbody>

            </table>

        </template>

    <template if:true={acc.error}>

        Error Message: {acc.error.body.message}

    </template>

    </lightning-card>

</template>

9

**WIRE decorator. Bir class tan birkaç objectten SOSL query ile bazı değerleri çağırıp org ta bir component ta gösterme**



**class**

public with sharing class getRecordsAccAndConMailForWire {

    @AuraEnabled(cacheable=true)

    public static List< List<sObject>> getRecords(){

        List< List<sObject>> searchList = [FIND 'com' IN ALL FIELDS RETURNING

                                            Account(Name, Website),

Contact(Name, Email)];

        return searchList;

    }

}

import { LightningElement, wire } from 'lwc';

import getRecords from '@salesforce/apex/getRecordsAccAndConMailForWire.getRecords';

export default class WireAccAndConMail extends LightningElement {

    accList;

    conList;

    error;

    @wire(getRecords)

    comCom({data, error}){

        if(data){

            this.accList = data[0];

            this.conList = data[1];

        }

    }

}

**js**

**html**

<template>

    <lightning-card title="com Of Account and Contact">

        <template if:true={conList}>

            <table class="slds-table

                        slds-table\_cell-buffer

                        slds-table\_bordered

                        slds-table\_col-bordered

                        slds-table\_striped">

                <thead>

                    <tr>

                        <th>Id</th>

                        <th>Name</th>

                        <th>Web/Email</th>

                    </tr>

                </thead>

                <tbody>

                    <template for:each={accList} for:item="each">

                        <tr key={each}>

                            <td>{each.Id}</td>

                            <td>{each.Name}</td>

                            <td>{each.Website}</td>

                        </tr>

                    </template>

                    <template for:each={conList} for:item="each">

                        <tr key={each}>

                            <td>{each.Id}</td>

                            <td>{each.Name}</td>

                            <td>{each.Email}</td>

                        </tr>

                    </template>

                </tbody>

            </table>

        </template>

        <template if:true={error}>

            Error Message: {error.body.message}

        </template>

    </lightning-card>

</template>

**IF kullanımı - değişken true ise true olan ifade gözükecek, false ise false olan ifade gözükecek**

<template>

    <lightning-card title="Show Me!!">

        <div if:true={status}>

            Show me when status is true

        </div>

        <div if:false={status}>

            Show me when status is true

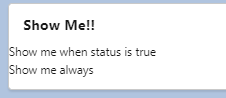
        </div>

        <div>Show me always</div>

    </lightning-card>

</template>

10



import { LightningElement } from 'lwc';

export default class EnginShowMe extends LightningElement {

    status=true;

}

import { LightningElement } from 'lwc';

export default class InnerSalesforce extends LightningElement {

}

<template>

    <lightning-card  title="SALESFORCE PARTS">

        <c-inner-compo-admin ><!-- class="slds-text-align\_center"-->

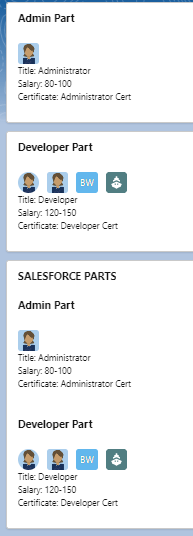
        </c-inner-compo-admin>

        <c-inner-compo-dev>

        </c-inner-compo-Dev>

    </lightning-card>

</template>



**Parent child component kullanımı.**

11

12