Դաս 2: JS Իրադարձություններ (Events)

1. Իրադարձություններ ներածություն
JS Իրադարձությունը դա գործողություններ կամ իրադարձություններ է, որը կարող է առաջանակ օգտատիրոջ կամ ծրագրի միջոցով: JS-ում մենք կարող ենք արձագանքել այդ իրադարձություններին ավելացնելով իրադարձության մշակիչներ, որոնք կկատարեն ծրագրավորողի կողմից նախատեսված գործողությունները:
Իրադարձությունները կարող են առաջանալ տարբեր պատճառներով.
Օգտատերը որևէ կերպ ինտերֆեյսի հետ փոխազդելիս (օրինակ՝ կոճակի սեղմում, տեքստի մուտքագրում, մկնիկի շարժում):
Փաստաթղթի վիճակի փոփոխություններ (օրինակ՝ էջի բեռնում կամ պատուհանի չափի փոխում):
Հիմնական դիրադարձություններից են **click**, **mouseover**, **keydown**.

2. Onclick և **addEventListener** -ի տարբերությունը՝

onclick-ը թույլ է տալիս նշանակել միայն մեկ իրադարձություն: Եթե մենք նշանակեք նոր մշակիչ (обработчик), ապա նախորդը կվերացրվի:
addEventListener-ը թույլ է տալիս մեկ տարրի վրա մեկ իրադարձության համար ավելացնել բազմաթիվ մշակիչներ (обработчик): Նրանք բոլորը կկանչվեն կարգի.

```
<button id="myButton">Սեղմիր ինձ</button>
<script>
    const button = document.getElementById('myButton');

    button.onclick = () => {
        alert('Առաջին մշակիչը onclick');
    };

    button.onclick = () => {
```

```
            alert('Երկրորդ մշակիչը onclick');
        };
    </script>
```

Երբ անհրաժեշտ է հեռացնել մշակիչը, պետք է՝
Onclick -ի համար

```
button.onclick = null;
```

addEventListener-ի միջոցով ավելացված մշակիչը հեռացնելու համար պետք է օգտագործել **removeEventListener()** մեթոդը: Սակայն այստեղ կարևոր է հիշել, որ **removeEventListener()**-ը աշխատում է միայն, եթե նույն ֆունկցիայի հղումը փոխանցել եք **addEventListener**-ի և **removeEventListener**-ի մեջ:
Եթե օգտագործում ենք անանուն (lambda) ֆունկցիաներ **addEventListener**-ի մեջ, դրանք հետագայում չեք կարողանա հեռացնել **removeEventListener**-ի միջոցով, քանի որ դրանց համար ֆունկցիայի հղում չի պահպանվում:

3. Անանուն (**lambda**) ֆունկցիաների և **սովորական** ֆունկցիաների տարբերությունները
Անանուն ֆունկցիաները և սովորական ֆունկցիաները javaScript-ում ունեն կարևոր տարբերություններ: Հիմնական տարբերությունը կայանում է this -ի աշխատանքի մեջ: Այս տարբերությունները սովորաբար դառնում են կրիտիկական, երբ սկսում ենք աշխատել իրադարձությունների կամ օբյեկտների հետ:

Անանուն ֆունկցիաները չունեն սեփական this: Անանուն ֆունկցիաները this -ի արժեքը ստանում են իր արտաքին կոնտեկստից, այլ կերպ ասած այնտեղից, որտեղ իրեն հայտարարել են: Այն շատ հարմար է մշակիչների հետ աշխատանքի դեպքում, քանի որ անանուն ֆունկցիաները ավտոմատ կերպով են վերագրում this-ը արտաքին կոնտեկստից:

```
<button id="btn">Սեղմիր ինձ</button>
```

```
<script>
  let obj = {
      value: 44,
      method: function() {
          document.querySelector('#btn')
          .addEventListener('click', () => {
              console.log(this.value);
          })
      }
  }
  obj.method()
</script>
```

Սովորական ֆունկցիաները ունեն սեփական this: Սովորական ֆունկցիայի this-ը կախված է թե ինչ օբյեկտի (տեգի) մեջ է կանչվել: Սովորական ֆունկցիայի this -ը միշտ ցույց է տալիս այն տեգը որի մեջ իրեն կանչել ենք:

```
let obj2 = {
  value: 55,
  method: function () {
    document.querySelector('#btn')
      .addEventListener('click', function () {
        console.log(this.value);
      })
  }
}

obj2.method()
```

JS իրադարձությունների ցուցակ ռուսերեն լեզվով | MDN

4. Համեմատաբար բարդ event-ներ

[keydown и keyup -ի ինֆորմացիոն կայք]

Keydown, **Keyup** իրադարձությունները հետևում են, երբ օգտատերը սեղմում կամ արձակում է ստեղնաշարի ստեղները:

```
    <h1>Ստեղնաշարի ստեղներ</h1>
    <p>Պետք է բացել կոնսոլը և սեղմել ստեղնաշարի ստեղները</p>

    <script>
        document.addEventListener('keydown', (event) => {
            console.log(`"${event.key}" ստեղնը սեղմված է (keydown)`);
        });

        document.addEventListener('keyup', (event) => {
            console.log(`"${event.key}" ստեղնը բաց թողնվեց (keyup)`);
        });

        document.addEventListener('keypress', (event) => {
            console.log(`"${event.key}" սեղմվել է (keypress) համարվում է հնացած իրադարձություն`);
        });
    </script>
```

Input, **change** իրադարձությունները կիրառվում են տեքստային դաշտերի հետ: Նրանք թույլ են տալիս վերահսկել մուտքագրվող տեքստը և հարկ եղած դեպքում նաև որոշակի լոգիկա ավելացնել: **Input** -ը աշխատում է տեքստային դաշտում ցանկացած փոփոխության դեպքում, իսկ **change** -ը աշխատում է տեքստային դաշտում փոփոխության առկայության և ֆոկուսի հեռացման դեպքում:

```
    <input type="text" name="" id="textInput">
    <script>
        document.getElementById("textInput").addEventListener('input', (event) => {
            console.log(`Ներկա պահին փոփոխված տվյալ: ${event.target.value}`);
        });
        document.getElementById("textInput").addEventListener('change', (event) => {
            console.log(`Վերջնական փոփոխված տվյալ: ${event.target.value}`);
        });
    </script>
```

Beforeinput իրադարձությունը համեմատաբար նոր իրադարձություն է, որը բրաուզերներում ճանաչվում և գործարկվում է 2020 թվականից սկսած: Այն հնարավորություն է

տալիս իրականացնել որոշակի գործողություն մինչև տարրում մուտքագրված արժեքի երևալը:

```
        document.getElementById("textInput").addEventListener('beforeinput', (event) =>
{
            console.log(`Միճչև մուսfագրված տառը կերևա: ${event.data}`);
            if (event.data === "a") {
                // Ազելակում եճf տառի մուսfագրումը եթե այճ ա տառ է
                event.preventDefault();}
        });
```

Mousemove, **mouseover**, **mouseout** իրադարձությունները թույլ են տալիս վերահսկել մկնիկի դիրքը և կատարած գործողությունները:

5. Իրադարձությունների համակարգում:

Որպեսզի օպտիմիզացնենք մեր կոդը, մենք կարող ենք վերագրել addEventListener -ը ոչ թե անհատապես ամեն տարրի, այլ անմիջապես իր ծնող տարրին: Այս մոտեցումը թույլ կտա էականորեն կրճատել eventListener-ների թիվը արագացնելով բրաուզերի աշխատանքը:

```
    <ul id="itemList">
        <li>Ադաջիճ li</li>
        <li>երկրորդ li</li>
        <li>երրորդ li</li>
    </ul>

    <div id="output"></div>

    <script>
        const itemList = document.getElementById('itemList');
        const output = document.getElementById('output');

        itemList.addEventListener('click', (event) => {
            if (event.target.tagName === 'LI') {
                output.textContent = `Դուf հեճg ճոր fլիկ արեցիf`
${event.target.textContent} -իճ`;
            }
        });
    </script>
```