기초 웹

22 Winter CNU 기초 스터디

21 남정연
21 박준서

Blocking Code

```
const btn = document.querySelector('button');
btn.addEventListener('click', () => {
  let myDate;
  for(let i = 0; i < 10000000; i++) {
    let date = new Date();
    myDate = date;
  }

  console.log(myDate);

  let pElem = document.createElement('p');
  pElem.textContent = 'This is a newly-added paragraph.';
  document.body.appendChild(pElem);
});
```

Causing code blocking

Threads

```
Task A --> Task B --> Task C
```

```
Thread 1: Task A --> Task B
Thread 2: Task C --> Task D
```

```
  Main thread: Task A --> Task C
Worker thread: Expensive task B
```

Worker is Synchronous

```
Main thread: Task A --> Task C
Worker thread: Expensive task B
```

```
<script>
  const btn = document.querySelector('button');
  const worker = new Worker('worker.js');

  btn.addEventListener('click', () => {
    worker.postMessage('Go!');  A

    let pElem = document.createElement('p');  C
    pElem.textContent = 'This is a newly-added paragraph.';
    document.body.appendChild(pElem);
  });

  worker.onmessage = function(e) {
    console.log(e.data);  B
  }
</script>
```

Asynchronous Code

```
  Main thread: Task A --> Task B --> |Task D|
Worker thread: Task C ----------> |      |
```

```
Main thread: Task A                    Task B
    Promise:      |__async operation__|
```

Async Callbacks

```
function loadAsset(url, type, callback) {
  let xhr = new XMLHttpRequest();
  xhr.open('GET', url);
  xhr.responseType = type;

  xhr.onload = function() {
    callback(xhr.response);
  };

  xhr.send();
}

function displayImage(blob) {
  let objectURL = URL.createObjectURL(blob);

  let image = document.createElement('img');
  image.src = objectURL;
  document.body.appendChild(image);
}

loadAsset('coffee.jpg', 'blob', displayImage);
```

Promises

```
fetch('products.json').then(function(response) {
  return response.json();
}).then(function(json) {
  let products = json;
  initialize(products);
}).catch(function(err) {
  console.log('Fetch problem: ' + err.message);
});
```

Promises

```
console.log ('Starting');
let image;

fetch('coffee.jpg').then((response) => {
  console.log('It worked :)')
  return response.blob();
}).then((myBlob) => {
  let objectURL = URL.createObjectURL(myBlob);
  image = document.createElement('img');
  image.src = objectURL;
  document.body.appendChild(image);
}).catch((error) => {
  console.log('There has been a problem with your fetch operation: ' + error.message);
});

console.log ('All done!');
```

setTimeout()

```
function sayHi(who) {
  alert(`Hello ${who}!`);
}
```

```
let myGreeting = setTimeout(sayHi, 2000, 'Mr. Universe');
```

```
clearTimeout(myGreeting);
```

setInterval()

```
function displayTime() {
   let date = new Date();
   let time = date.toLocaleTimeString();
   document.getElementById('demo').textContent = time;
}

const createClock = setInterval(displayTime, 1000);
```

```
const myInterval = setInterval(myFunction, 2000);

clearInterval(myInterval);
```