SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

JavaScript

SAMSUNG SW ACADEMY FOR YOUTH

Web - FrontEnd

JavaScript

- ES6문법
- fetch

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

ES6 문법

ES6 문법

- let, const
- arrow function
- spread/rest operator(...)
- for/of
- class
- promise

http://es6-features.org/

Spread/rest operator(...)

- **spread** : 배열로 묶여 있는 값들을 각각의 개별 값으로 풀어줌, 문자열은 각각의 문자로 나눔
- **rest**: 나머지 값들을 모아서 배열로 만듦

```
var params = [ "hello", true, 7 ]
var other = [ 1, 2, ...params ] // [ 1, 2, "hello", true, 7 ]

function f (x, y, ...a) {
    return (x + y) * a.length
}
f(1, 2, ...params) === 9

var str = "foo"
var chars = [ ...str ] // [ "f", "o", "o" ]
```

- for/of 문법

```
for (variable of iterable) {
  // code block to be executed
}
```

```
const cars = ["BMW", "Volvo", "Mini"];
let text = "";

for (let x of cars) {
  text += x + " ";
}
```

- class: 자바스크립트 객체의 틀

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Promise

콜백이란?

- 함수를 매개변수로 전달하여,
- 나중에 실행하도록 하는 것
- 콜백이 중첩되면, 콜백 헬이 되어 해석하고 유지보수하기 힘든 코드가 될 우려(스파게티 코드)

```
function fun(successCallback, failureCallback){
    if(작업 성공시) {
        successCallback();
    } else {
        failureCallback();
}
```

Promise Object

- 비동기 작업을 마치 동기 작업처럼 값을 반환해서 사용 형태
- 미래의 완료 또는 실패와 그 결과 값을 나타냄.
- 미래의 어떤 상황에 대한 약속

- new Promise(function (resolve, reject) { })

- resolve (성공 시 사용)
- reject (실패 시 사용)

https://developer.mozilla.org/ko/docs/Web/JavaScript/Reference/Global_Objects/Promise

Promise Object

```
const promise = new Promise((resolve, reject) => {
  resolve('resolve');   -> then 부분을 실행
  // reject('reject');   -> catch 부분을 실행
});

promise
  .then((data) => {
    console.log(data);
  })
  .catch((data) => {
    console.log(data);
  });
```

Promise Methods

- .then(callback)
  - ✓ Promise 객체를 리턴하고 두 개의 콜백 함수를 인수로 받는다. (이행 했을 때, 거부 했을 때)
  - ✓ 콜백 함수는 이전 작업의 성공 결과를 인자로 전달 받음.
- .catch(callback)
  - ✓ .then 이 하나라도 실패하면(거부 되면) 동작 (예외 처리 구문 유사)
  - ✓ 이전 작업의 실패로 인해 생성된 error 객체는 catch 블록 안에서 사용 가능
- .finally(callback)
  - ✓ Promise 객체 반환
  - ✓ 결과 상관없이 무조건 실행
- 체이닝 가능

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

fetch

fetch API

- XMLHttpRequest보다 강력하고 유연한 조작이 가능
- Promise를 지원하므로 콜백 패턴에서 자유로움
- ES6문법은 아니고, BOM (Browser Object Model) 객체 중의 하나임.
- fetch() 메서드를 사용함
- fetch() 메서드는 HTTP 응답을 나타내는 Response 객체를 래핑한 Promise 객체를 반환

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch

- fetch(resource, options) 메서드
  - resource: 리소스가 위치한 url 지정
  - options: 옵션을 지정
    - method: HTTP method
    - headers: 요청 헤더 지정
    - body: 요청 본문 지정
  - fetch 메서드는 Promise 객체를 반환

- fetch() 가 반환하는 Promise 객체
  - 성공시 then() 을 이용해 처리
  - 실패시 catch()를 이용해 처리

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch

fetch 사용 예

- fetch 메서드는 HTTP 응답을 나타내는 Response 객체를 래핑한 Promise 객체를 반환
- response.text() : Response의 Body를 텍스트의 형태로 반환
- response.json() : Response의 Body를 JSON 파싱하여 반환

```
fetch("https://jsonplaceholder.typicode.com/posts/1")
        .then((response) => response.text())
        .then((text) => JSON.parse(text))
        .then((body) => console.log(body));
```

https://developer.mozilla.org/en-US/docs/Web/API/Response

https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

다음 방송에서 만나요!

삼성 청년 SW 아카데미