웹퍼블리싱 강의

2022. 6

강 현 준

human@human.or.kr

강의 목차

1. HTML
2. CSS
3. JAVA SCRIPT
4. JQuery
5. 비동기 통신

5장. 비동기 통신

5.0.1. 프로그램 언어별 포지션

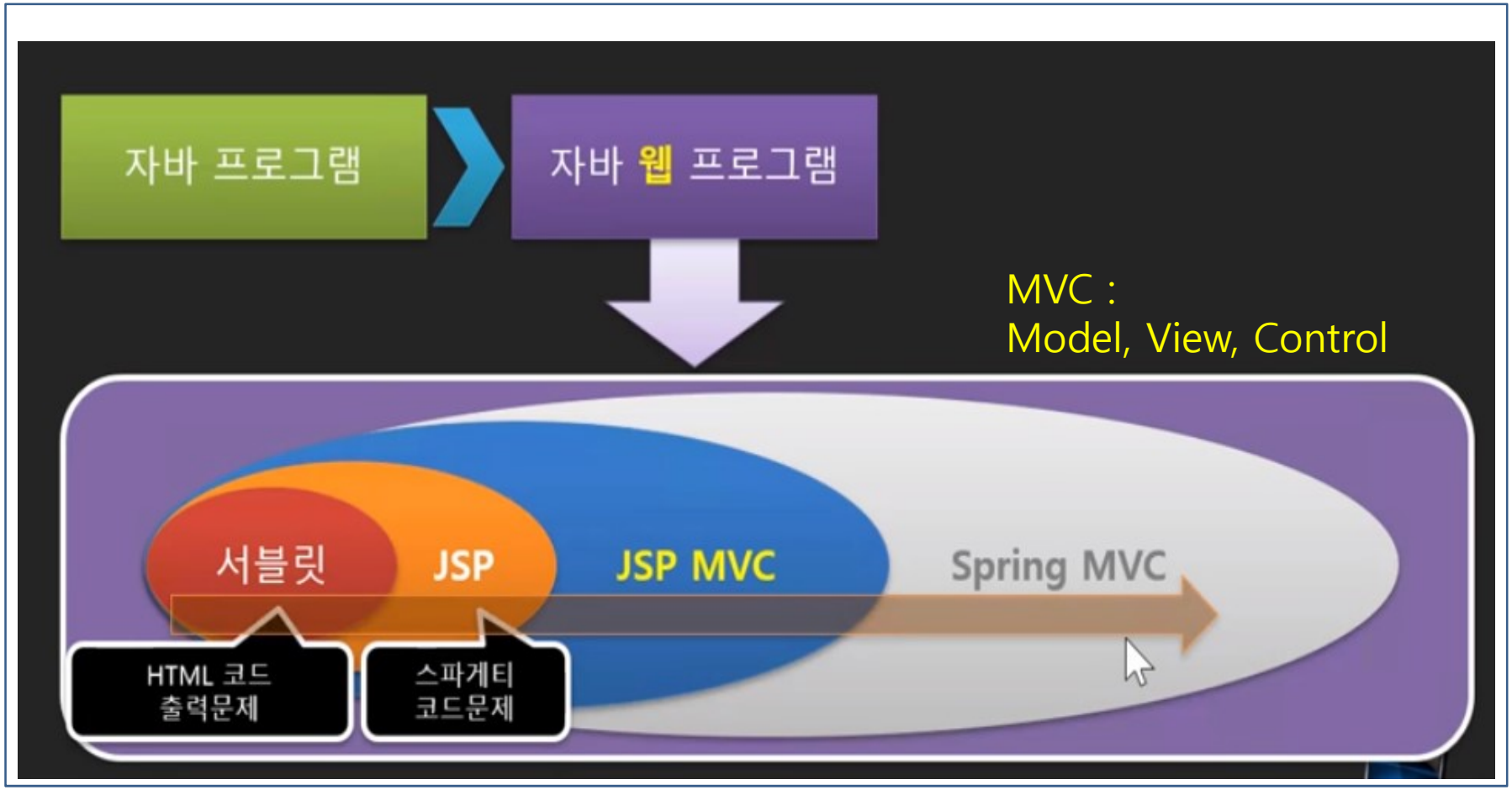
- Front-End : Web Browser 등 사용자와 접점을 가지고 있는 부분을 개발
- Back-End : 데이터 기반의 로직을 기반으로 솔루션의 엔진을 담당함.

프로그램 언어별 포지션

5.0.2. JAVA 언어의 단계

- 현재 취업시장에서 가장 많은 것은 웹 프로그래머
- 웹 프로그램은 서블릭 ➔ JSP ➔ Spring의 단계로 교육이 진행될 예정

.



5.1. 비동기(Asynchronous) 통신이란?

- AJAX (Asynchronous Javascript And XML) : XHR(XMLHttpRequest), AXIOS, FETCH 방식
- 비동기 애플리케이션을 만들기 위해 클라이언트에서 단에서 쓰이는 웹개발 기술들.

AJAX

AJAX 예

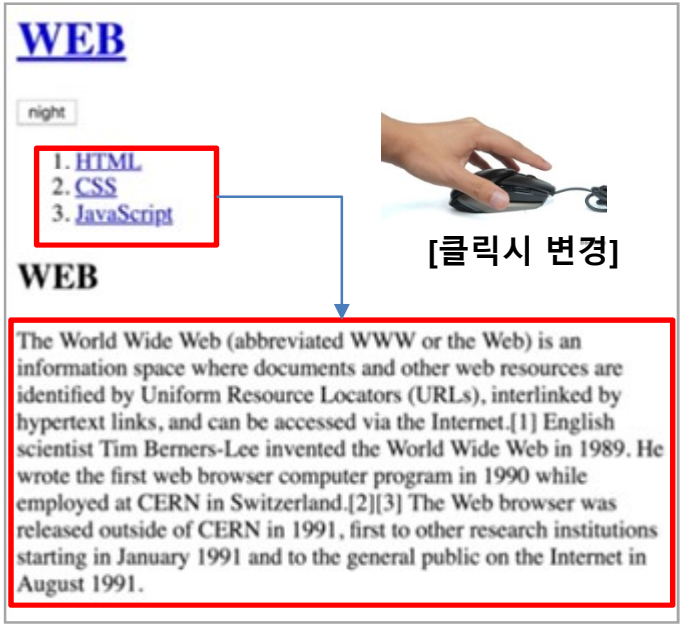
이게 어떻게 나오지?
수많은 데이터를 웹문서에 담아둘 수는 없음.
글자를 타이핑할 때마다
웹페이지는 호출하고,
호출한 후 응답이 올때까지 다른 일을 함.
(비동기식의 의미임)

5.2. 비동기 통신의 탄생배경

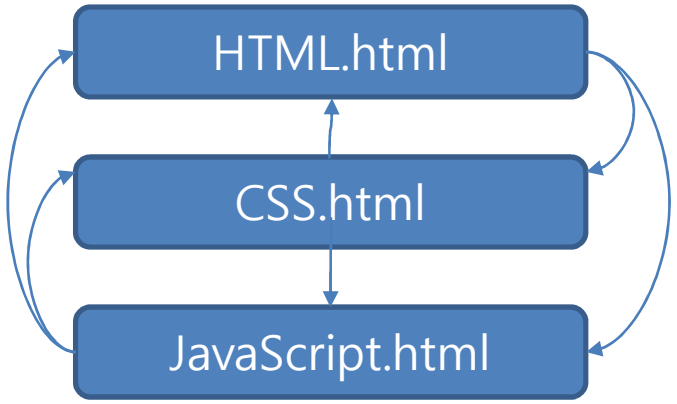
- 기존 웹문서는 구조가 바뀌면 모든 문서를 변경해야 하는 한계가 있음. (유지관리 비용 증가)
- 한계를 극복하고자 비동기 통신이 탄생함.

비동기 통신 탄생 배경

비동기 통신탄생 배경



동일한 문서를 내용만 달리하여
서로 Link로써 수행함



1. 조금만 모든 문서를 바꾸어야 함
2. 항목이 추가된다면.
   별도로 같은 것을 만들어야 함.
3. 추가된 것을 또 모든 문서가
   링크가 되어야 함.

➔ 해결 방법은?

5.3. 비동기 통신 구현 원리

- innerHTML을 활용하여 이벤트 발생시 article 태그에 문서 삽입함.
- 문서 삽입은 querySelector를 활용함.

비동기 통신 구현 원리

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        li{
            font-size: 40px;
            font-weight: bold;
        }
    </style>
</head>
<body>
    <ol>
        <li><a onclick = "document.querySelector('article').innerHTML='<H1>HTML</H1><P>HTML, ...';">
            HTML</a></li>
        <li><a onclick = "document.querySelector('article').innerHTML='<H1>CSS</H1><P>CSS, ...';">
            CSS</a></li>
        <li><a onclick = "document.querySelector('article').innerHTML='<H1>JAVA Script</H1><P>JAVA Script, ...';">
            JAVA Script</a></li>
    </ol>
    <article>
    </article>
</body>
</html>
```

5.4.1. 비동기 통신 구현을 위한 TEST - 1

- callback 함수를 통한 로그 확인. (JAVASCRIPT 문서 확인함.)
- 로그가 먼저 찍히고, response OK가 됨. (비동기식이므로)

TEST-1

```
<!-- 서버와 통신하기 위해 XMLHttpRequest 객체를 사용하는 것을 말합니다. 단, 비동기식임. -->
<input type = "button" value="click_callbackme"
        onclick="fetch('JAVASCRIPT').then(callbackme);
                function callbackme(){
                    console.log('response_ok');
                }
                console.log(1)
                console.log(2)
                console.log(3)
                "
```

나중에 수행됨

먼저 수행됨.

```
1
2
3
response_ok
```

5.4.2. 비동기 통신 구현을 위한 TEST - 2

- CSS 문서 확인하여 text란 변수에 넣어라.
- response.status = 200은 통신에 성공했다는 의미임.

TEST-2

```
<!-- 버튼 클릭시 HTML 파일을 찾아서 text라는 변수에 넣고 innerHTML로 하여서 읽어온 것을 출력하라 -->
    <input type = "button" value="click_ajax_article"
        onclick="fetch('CSS').then(function(response) {
                        response.text().then(
                            function(text) {
                                if (response.status ==200) {
                                    document.querySelector('article').innerHTML = text;
                                    console.log('SUCCESS!!')
                                }
                                else console.log('connect error 발생')

                        })
                })"
    >
    <article>

    </article>
```

5.4.3. 비동기 통신 구현을 위한 TEST - 3

- FETCH API는 전달하는 문서를 text로 받고, inner HTML을 통해 article tag에 문서를 포함
- 이 때 response 객체를 통해서 성공여부를 확인할 수 있음. (AJAX의 기본은 마무리 됨.)

TEST-3

```
<body>
    <ol>
        <li><a onclick = "fetch('HTML').then(function(response) {
                            response.text().then(
                                function(text) {
                                        document.querySelector('article').innerHTML = text;
                                })
                        });">
            HTML</a></li>
        <li><a onclick = "fetch('CSS').then(function(response) {
            response.text().then(
                function(text) {
                        document.querySelector('article').innerHTML = text;
            })
        });">
            CSS</a></li>
        <li><a onclick = "fetch('JAVASCRIPT').then(function(response) {
            response.text().then(
                function(text) {
                        document.querySelector('article').innerHTML = text;
            })
        });">
            JAVA Script</a></li>
    </ol>
    <article>
    </article>
</body>
```

공통되는 부분이 많음

공통되는 부분이 많음

공통되는 부분이 많음

5.4.4.비동기 통신 구현을 위한 TEST - 4

- 공통된 부분을 하나로 합쳐서 처리함

TEST-4

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        li{
            font-size: 40px;
            font-weight: bold;
        }
    </style>
    <script>
        // 함수로 호출하여 간략하게 만듦
        function click_page(file) {
            fetch(file).then(function(response) {
                response.text().then(
                    function(text) {
                        document.querySelector('article').innerHTML = text;
                    })
            });
        }
    </script>
</head>
<body>
    <ol>
        <li><a onclick = "click_page('HTML')">HTML</a></li>
        <li><a onclick = "click_page('CSS')">CSS</a></li>
        <li><a onclick = "click_page('JAVASCRIPT')">JAVASCRIPT</a></li>
    </ol>
    <article>
    </article>
</body>
```

참조하려는 문서를
인자로 전달하여 처리