HTML5 + CSS3

jQuery

front-end programming

by hyunseok

jQuery

- site url : https://jquery.com/
- 2006년에 처음에 공개, 존레식(John Resig)
- 빠르고 작고 기능이 풍부한 JavaScript 라이브러리
- 여러 브라우저에서 작동하는 사용하기 쉬운 API를 사용하여 HTML문서 탐색 및 조작, 이벤트 처리, 애니메이션 및 Ajax와 같은 작업을 훨씬 더 간단하게 만든다.
- DOM 조작 및 Ajax 요청을 실행할 때 널리 쓰이는 라이브러리
- jQuery로 할 수 있는 일은 모두 DOM API로 할 수 있다.

jQuery 장점

- jQuery를 사용하면 브라우저 호환성을 걱정하지 않아도 된다. 특히 오래된 브라우저를 지원해야 할 때 골치 아픈 일이 줄어든다.
- jQuery가 제공하는 Ajax API는 무척 단순하다. 요즘은 웹 사이트에서 Ajax를 아주 많이 사용하므로 이 장점은 무시할 수 없다.
- jQuery는 내장된 DOM API를 더 유용하고 단순하게 바꾼 메소드를 제공한다.
- write one, do more
- 문법이 간결하다.
- 사용하기 쉽다.
- 빠르게 배울 수 있다.
- 다른 라이브러리와 충돌을 일으키지 않는다.
- 다양한 플러그인이 존재한다.
- 브라우저 호환성 문제를 해결해 준다.(jQuery 1, 2, 3 에 따라 다르다)
- MIT license

jQuery 현 상황

- 워낙 널리 퍼져있어서 언제든지 jQuery 코드에 부딪힐 가능성이 높다.
- 요즘은 jQuery를 걷어내거나 사용하지 않는 경우가 많다.
- DOM API로 직접 구현하려면 시간이 많이 걸리는 기능을 간편하게 제공
- 요즘은 Vue.js 나 React.js 같은 JavaScript 라이브러리를 많이 사용한다.

jQuery가 하는 일

- HTML 문서안의 요소를 찾는다.
- 찾은 요소를 조작한다.

jQuery 불러오기

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```

- jQuery를 불러오는 가장 쉬운 방법은 CDN을 사용하는 것임.
- jQuery는 2.X 버전부터 IE8 이하에 대한 지원을 중단함.
- IE 구 버전을 지원해야 한다면 jQuery 1.X버전을 써야한다.
- 현재 jQuery 버전은 3.6(2022년 5월현재)

jQuery 불러오기

- jQuery 라이브러리를 다운로드 받지 않고 인터넷이 연결되어 있으면 사용하는 방법
- CDN(Content Delivery Network) 으로 사용
  - Google Ajax API CDN
    - https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js
  - Microsoft CDN
    - https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.6.0.js
  - jQuery CDN
    - <script src="https://code.jquery.com/jquery-3.6.0.js" integrity="sha256-H+K7U5CnXl1h5ywQfKtSj8PCmoN9aaq30gDh27Xc0jk=" crossorigin="anonymous"></script>

jQuery 불러오기

CDN

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>jQuery 불러오기</title>
</head>
<body>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</body>
</html>
```

다운로드

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>jQuery 불러오기</title>
</head>
<body>
    <script src="js/jquery-3.6.0.min.js"></script>
</body>
</html>
```

jQuery 작성 위치

local

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>jQuery 불러오기</title>
</head>
<body>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        // 자신이 작성하고자 하는 jQuery 내용 작성
    </script>
</body>
</html>
```

- jQuery를 먼저 작성해서 로드한 후에 자신이 작성하고자 하는 내용을 작성한다.
- jQuery도 JavaScript와 마찬가지로 〈body〉태그 맨 마지막 바로 위에 위치시키고 사용

$

- $ 기호를 식별자로 쓸 수 있다.
- jQuery나 $ 표시를 사용한다.

```
<p></p>

$(“p”).innerHTML = “안녕!!”;


<span class=“resetBtn”></span>
$(“.resetBtn”).innerHTML = “버튼취소”;
```

```
<p></p>

jQuery(“p”).innerHTML = “안녕!!”;


<span class=“resetBtn”></span>
jQuery(“.resetBtn”).innerHTML = “버튼취소”;
```

엘리먼트 찾기

- HTML 문서안의 엘리먼트를 찾는 방법
- 태그 (body, div, span, a, 등등)
- 아이디 (id = “age”)
- 클래스 (class = “container”)

엘리먼트 찾기

- $(“a”) === `document.getElementsByTagName("a")`
- $(“#container”) === `document.getElementById("container")`

엘리먼트 찾기

- $(“a”)

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>jQuery 찾는 방법 - tag</title>
</head>
<body>
    <div id="name">
        손흥민
    </div>
    <div id="age">
        29
    </div>
    <a href="#">손흥민SNS</a>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $("a").css({background:'#e26500'});
    </script>
</body>
</html>
```

엘리먼트 찾기

- $(“a”); // 태그
- $(“#myId”); // id
- $(“.container”); // class
- $(“.container .btn-reset”); // class들의 조합
- $('div, .age_style').css('color','#e26500');

  // 태그와 클래스에 css에 모두 반영하기 위한 방법
  // div태그와 .age_style 클래스가 지정되어있는 모든 태그들 선택

jQuery - exam

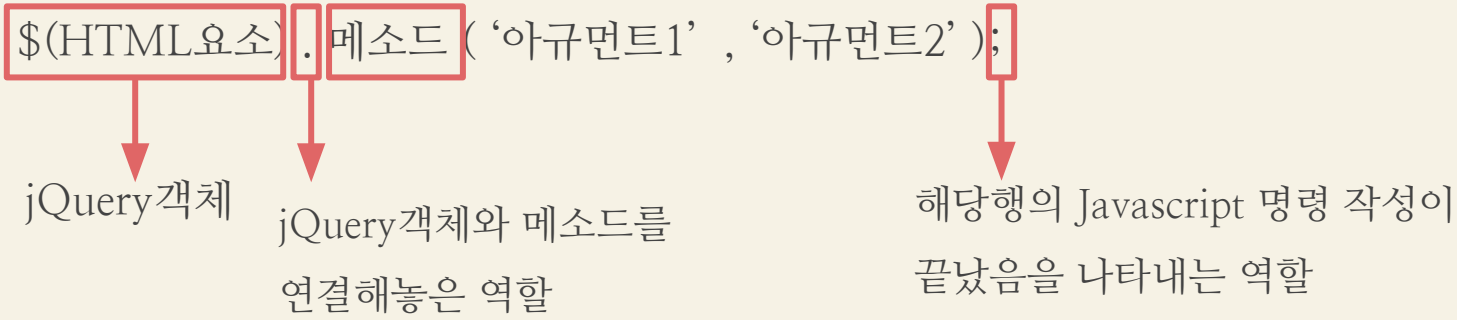
- 〈div〉태그들의 배경색을 #e26500 로 만들어 보자.


```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <title>jQuery 불러오기</title>
</head>
<body>
    <div>
        jquery 배경색 테스트 입니다.
    </div>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script>
        $("div").css({background:'#e26500'});
    </script>
</body>
</html>
```

jQuery 실행흐름

- $() 함수에서 명령 대상이 되는 html요소를 jQuery 객체로 변환
- 변환한 jQuery 객체에 메소드를 호출해서 html 요소를 변경한다.

- $(HTML요소) . 메소드 ('아규먼트1' , '아규먼트2');



DOMContentLoaded 처리

```
$(document).ready(function(){
    // 여기에 있는 코드는 HTML을 모두 불러오고
    // DOM이 구성된 다음 실행된다.
});
```

```
$(function(){
    // 여기 있는 코드는 HTML을 모두 불러오고
    // DOM이 구성된 다음 실행된다.
});
```

- 브라우저가 페이지를 완전히 읽고 DOM을 로드한 다음에 호출한다.
- 〈head〉〈/head〉태그에 script를 작성하려면 이와같이 작성한 후 그 안에 구현하면 오류없이 작동이 된다.

DOMContentLoaded 처리

- $(function(){
  // DOMContentLoaded가 된 후에 처리될 내용 작성하면 됨.
  });

```
$(function(){
    const $paras = jQuery('p');
    console.log($paras.length);
    console.log(typeof $paras);
});
```

```
$(function(){
    const $paras = $('p');
    console.log($paras.length);
    console.log(typeof $paras);
});
```

jQuery로 감싼 DOM 요소

```
const $paras = $('p');
console.log($paras.length);
console.log(typeof $paras);
```

```
const $paras = jQuery('p');
console.log($paras.length);
console.log(typeof $paras);
```

- jQuery로 가장 많이 사용하는 방법은 DOM 요소를 감싸는 방법
- '$' 나 'jQuery'로 DOM 요소를 감싼 것을 jQuery 객체라고 부른다.
- jQuery 함수를 호출할 때는 주로 CSS나 HTML을 사용한다.
- jQuery를 호출하면 해당 선택자에 일치하는 jQuery 객체가 반환된다.(document.querySelectorAll)

CSS 스타일

```
$('선택자').css('속성','값');
```

```
$('.box').css('color','red');
```

```
$('.btn-box .btn-reset').css('color','red');
```

- 값을 하나만 바꿀때 사용
- 선택자
  - css와 선택자를 지정
  - 지원되지 않기도 하지만, 특정요소를 지정하는데는 큰 문제는 없음
- 속성
  - css에 설정할 속성을 지정한다.
- 값
  - 속성에 설정할 값을 지정한다.

실행시점 제어

- 특정 요소에 마우스를 올려놓은 시점
- 특정 요소에서 마우스 포인터가 벗어나는 시점
- 특정 요소를 클릭하는 시점
- 마우스를 움직이는 시점
- 창 크기가 바뀌는 시점
- 스크롤 하는 시점

실행시점 제어

이 벤 트 명	이벤트가 발생하는 시점
keydown	해당 엘리먼트에서 키를 누를 때 발생
keyup	해당 엘리먼트에서 키를 누르고 뗄 때 발생
focus	해당 엘리먼트에 포커스가 일치했을 때
blur	해당 엘리먼트가 포커스를 잃어버렸을 때
change	입력내용이 변경되었을 때(textarea, input 엘리먼트, select 엘리먼트)
resize	요소의 크기를 다시 설정할 때
scroll	요소를 스크롤 했을 때

on() 메소드

```
$('선택자').on('발생할 이벤트','처리할 내용')
```

- $('#myId').on('mouseover',function(){...});
  - mouseover : on 메소드의 첫번째 아규먼트,발생할 이벤트명을 작성
  - function(){} : 첫번째 아규먼트의 이벤트가 발생할때 처리할 내용

메소드 체인

- 메소드 체인
  - 메소드를 체인처럼 연결하여 실행하는 방법
  - `$(‘선택자’).메소드A().메소드B().메소드C()`
  - `$('#myId').mouseover(function(){...}).mouseout(function(){...})`
- 메소드 체인의 장점
  - 함수를 한번만 작성해도 되므로 코드를 단순화 시킴
  - 프로그램의 처리속도를 향상시킬 수 있다.

메소드 체인

```
$(function(){
    $('#myId')
    .on('mouseover',function(){
        $('#myId').css({
          color:'blue',
          backgroundColor:'gray'
        });
    });
    .on('mouseover',function(){
        $('#myId').css({
          color: '',
          backgroundColor: ''
        });
    });
});
```

on()메소드를 사용해서 구현

```
$(function(){
    $('#myId')
    .mouseover(function(){
        $('#myId').css({
          color:'blue',
          backgroundColor:'gray'
        });
    })
    .mouseout(function(){
        $('#myId').css({
          color: '',
          backgroundColor: ''
        });
    });
});
```

on()메소드를 사용하지 않고 구현

fetch

```
fetch('http://example.com/movies.json')
  .then((response) => response.json())
  .then((data) => console.log(data));
```

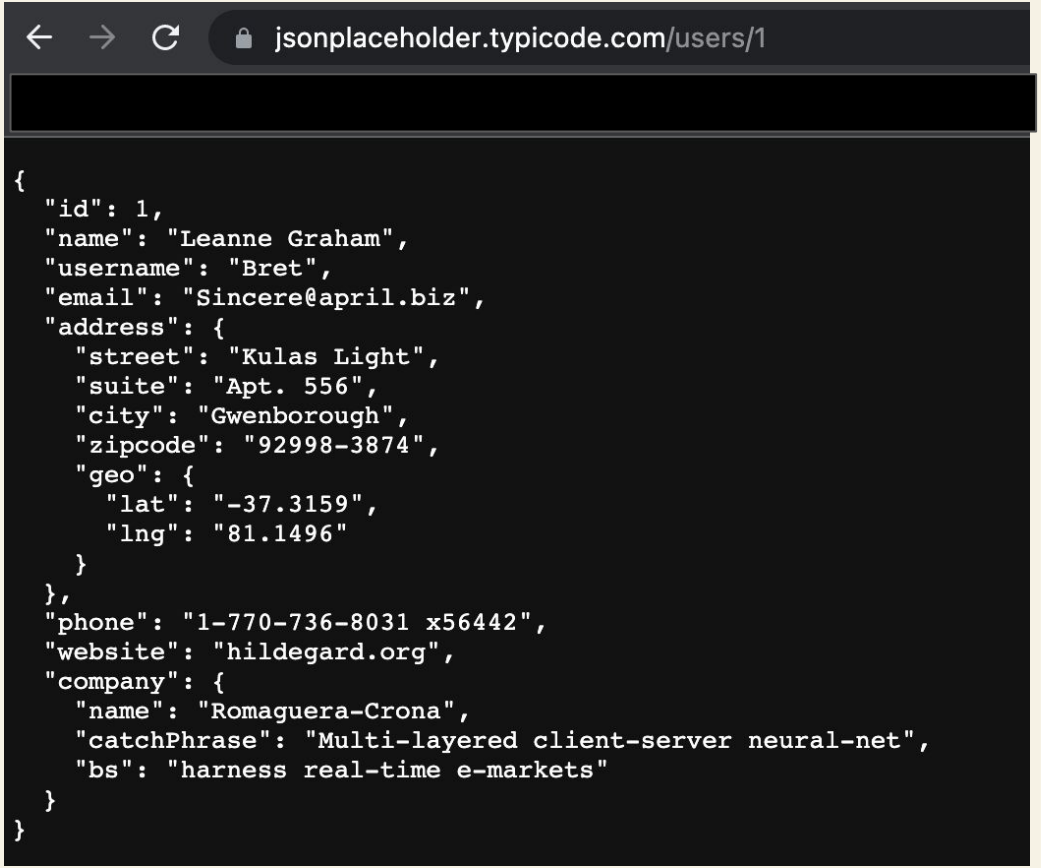
- 네트워크로 외부에 있는 리소스를 비동기로 쉽게 가져올 수 있는 API
- fetch() 이전에는 XMLHttpRequest를 사용
- fetch()와 jQuery.ajax()의 다른 점
    - fetch()는 404,500같은 HTTP오류에도 거부되지 않음

참고 - https://developer.mozilla.org/ko/docs/Web/API/Fetch_API/Using_Fetch

fetch

인증없이 데이터 가져올 수 있는 url

https://jsonplaceholder.typicode.com/users/1



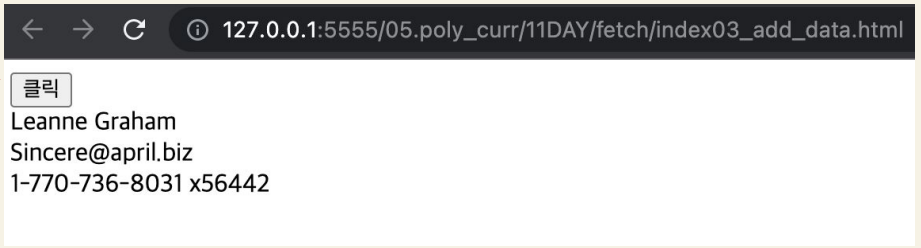
fetch

- 간단하게 데이터 가져오는 예제
- 예제를 그대로 파일로 작성하면 데이터를 가져와서 화면에 붙이는 것을 확인할 수 있다.
- 깜빡거림없이 그대로 동시에 처리가 되는 것을 볼 수 있다.
- 인증없이 데이터를 가져올 수 있는 url
    - https://jsonplaceholder.typicode.com/users/1

```
<!DOCTYPE html>
<html>
<body>
 <button onclick="getUser()">클릭</button>
 <div id="userInfo"></div>
 <script>
   // url로 접근해서 데이터를 가져와서
   // 가져온 데이터를 문서에 출력해 보기
   // 간단하게 ajax 형식의 통신을 확인해 볼 수 있는 예제
   function getUser() {
     const config = {
       method: "get"
     };
     fetch("https://jsonplaceholder.typicode.com/users/1", config)
       .then(response => response.json())
       .then(data => {
         const name = document.createElement("div");
         const email = document.createElement("div");
         const phone = document.createElement("div");
         name.textContent = data.name;
         email.textContent = data.email;
         phone.textContent = data.phone;
         const userInfo = document.getElementById("userInfo");
         userInfo.appendChild(name);
         userInfo.appendChild(email);
         userInfo.appendChild(phone);
       })
       .catch(error => console.log("fetch 에러!"));
   }
 </script>
</body>
</html>
```

fetch


127.0.0.1:5555/05.poly_curr/11DAY/fetch/index03_add_data.html
클릭


127.0.0.1:5555/05.poly_curr/11DAY/fetch/index03_add_data.html
클릭
Leanne Graham
Sincere@april.biz
1-770-736-8031 x56442

fetch

- json형태의 데이터를 POST 방식으로 전달하는 방법
- JSON.stringfy(data)는 JSON 객체 text형태로 변환시켜주는 API

```
const data = { username: 'example' };

fetch('https://example.com/profile', {
        method: 'POST', // 또는 'PUT'
        headers: {
          'Content-Type': 'application/json',
        },
        body: JSON.stringify(data),
})
.then((response) => response.json())
.then((data) => {
       console.log('성공:', data);
})
.catch((error) => {
       console.error('실패:', error);
});
```

JSON

- JSON(JavasScript Object Notation)
- 단순하게 데이터를 표기하는 방법
- 이전에는 XML을 많이 사용하다가 현재는 JSON 객체를 많이 사용한다.
- { “프로퍼티이름” : “값” } 형태로 구성되어 있고, 프로퍼티 이름과 값은 “”로 표시하고, 값중에 숫자형의 값은 “”없이 작성한다.

```
const news =
    {
        "title":"sbs",
        "no":2,
        "imgurl":"http://static.naver.net/newsstand/2017/0313/article_img/9054/173200/001.jpg",
        "newslist":[
             "[가보니] 가상 경주도 즐기고, 내 손으로 자동차도 만들고",
             "리캡차'가 사라진다.",
             "갤럭시S8’ 출시? ‘갤노트7’ 처리 계획부터 밝혀야",
             "블로코-삼성SDS, 블록체인 사업 ‘맞손’",
             "[블록체인 톺아보기] 퍼블릭 블록체인의 한계와 프라이빗 블록체인"
        ]
    }
```