

프론트 세미나 Week6

Form tag & Ajax

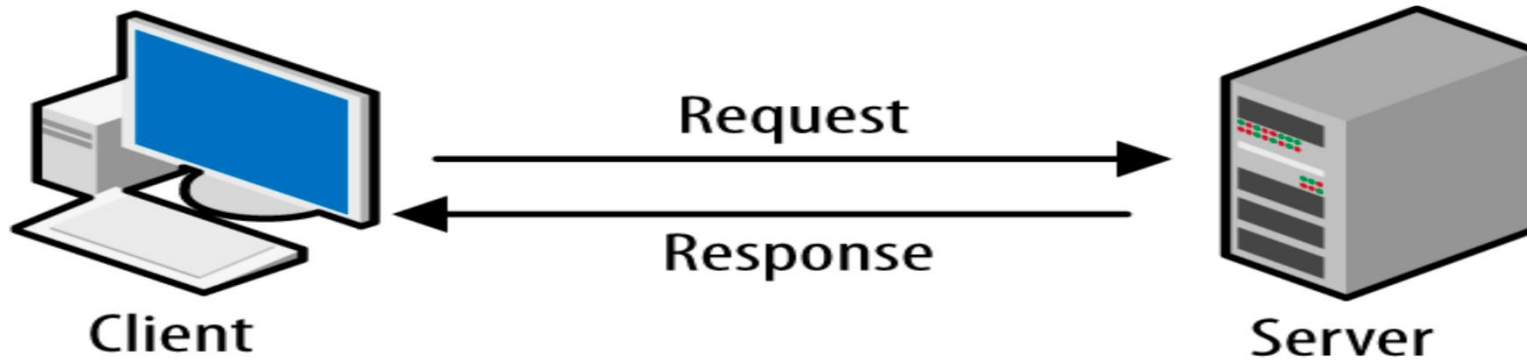
Review



Last lecture: JS 활용 & DOM

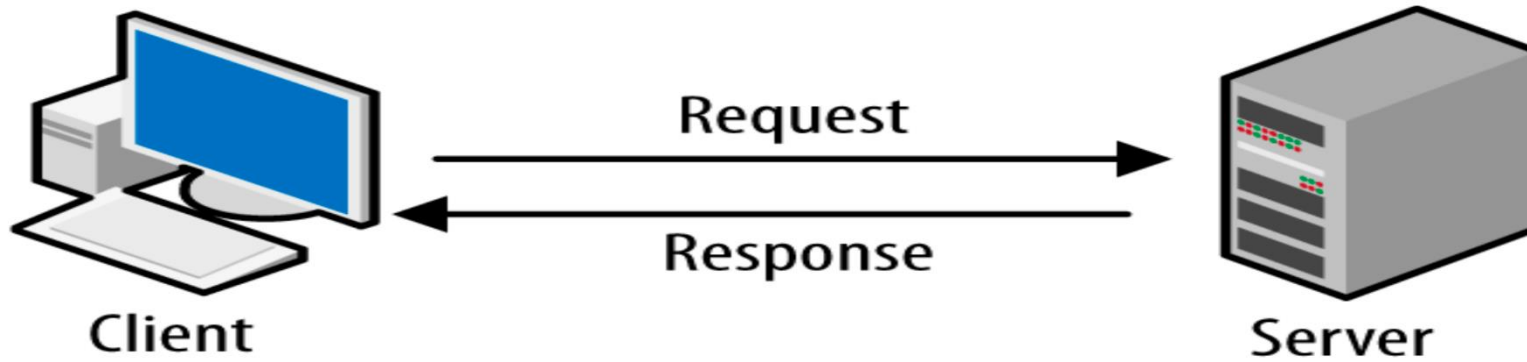
- 지금까지 배운 내용으로 우리는 HTML, CSS, JS를 활용하여 정적인 웹페이지를 구현할 수 있게 되었습니다!
- 오늘 세미나에서는 백엔드와 어떻게 데이터를 주고받을 수 있는지에 대해 알아보시다!
- 다음주 해커톤에서 오늘 배운 사항들을 활용하여 백엔드와 협업을 진행해보세요!

Server-Client








- 프론트에서 서버의 내용을 직접적으로 처리할 필요는 X
- 하지만, 서버로 내용을 보내주거나, 받아온 내용을 사용자에게 보여주는 것은 처리해주어야 함!
- 클라이언트에서 서버로 정보를 요청(request)하면 서버는 그에 맞는 정보를 클라이언트에 보냄(response)
- 서버로 내용을 보낼 때: form tag를 이용
- 서버에서 받아온 내용을 유저에게 보여줄 때: ajax를 이용

Server-Client



- 이 때, 백엔드와 어떤 주소를 이용할지 미리 논의가 필요
- ex1) server1.com에 중요한 정보가 저장되어 있으니 이를 유저에게 보여주세요!
- ex2) 사용자로부터 로그인 정보를 받으면 login.com으로 정보를 보내주세요!

Form Tag

Run »

Result Size: 625 x 461

```
<!DOCTYPE html>
<html>
<body>

<h1>The form element</h1>

<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
</form>

<p>Click the "Submit" button and the form-data will be sent to a page on
the
server called "action_page.php".</p>

</body>
</html>
```

The form element

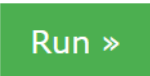





First name:

Last name:

Click the "Submit" button and the form-data will be sent to a page on the server called "action_page.php".

- form tag: 백엔드, 즉 서버로 정보를 전송하기 위한 용도로 사용
- action="데이터가 도착할 서버의 주소"
- form tag 안에 전송할 데이터를 여러 tag로 넣어줌

Form Tag



Result Size: 625 x 461

```
<!DOCTYPE html>
<html>
<body>

<h1>The form element</h1>

<form action="/action_page.php">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
</form>

<p>Click the "Submit" button and the form-data will be sent to a page on
the
server called "action_page.php".</p>

</body>
</html>
```

The form element

First name:

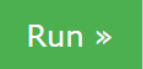





Last name:

Click the "Submit" button and the form-data will be sent to a page on the server called "action_page.php".

- type="submit" 버튼을 누를 때 서버로 데이터 전송
- 다양한 input type이 존재

<http://jun.hansung.ac.kr/CWP/htmls/HTML%20Input%20Types.html>

Form Tag



Result Size: 625 x 461

```
<!DOCTYPE html>
<html>
<body>

<h1>The form method="get" attribute</h1>

<form action="/action_page.php" method="get" target="_blank">
  <label for="fname">First name:</label>
  <input type="text" id="fname" name="fname"><br><br>
  <label for="lname">Last name:</label>
  <input type="text" id="lname" name="lname"><br><br>
  <input type="submit" value="Submit">
</form>

<p>Click on the submit button, and the input will be sent to a page on
the server called "action_page.php".</p>

</body>
</html>
```

The form method="get" attribute

First name:

Last name:

Click on the submit button, and the input will be sent to a page on the server called "action_page.php".

- method: 데이터를 어떻게 전송할지를 명시: GET(default) or POST
- GET: 데이터를 url pair로 전송 ex) URL?name1=value1&name2=value2&... → 보안상의 취약점
- POST: form data 별도로 전송

Form Tag

METHOD	역할
POST	POST를 통해 해당 URI를 요청하면 리소스를 생성합니다.
GET	GET를 통해 해당 리소스를 조회합니다. 리소스를 조회하고 해당 도큐먼트에 대한 자세한 정보를 가져온다.
PUT	PUT를 통해 해당 리소스를 수정합니다.
DELETE	DELETE를 통해 리소스를 삭제합니다.

- HTTP에 정의된 method와 연관됨
- 보통의 경우 form tag를 사용할 때는 method를 post로 지정하지만, 자세한 사항은 백엔드와의 협의가 필요!
- ex) 이 주소로 GET method를 지정하기로 했으면, 프론트에서는 단지 method를 get으로 지정해주기만 하면 됨

Form Tag

```
<form action="#" method="post">  
    <input type="hidden" name="_method" value="put" />  
  
</form>
```

- method 값으로 PUT, DELETE는 사용될 수 없음
- PUT, DELETE를 하려면 위와 같은 방식으로 적어주어야 함!

Form Tag

Elements

Console

Sources

Network

Performance

Memory

Application

Security

Lighthouse

Adblock Plus

</

✕ Headers Preview Response Initiator Timing

▼ General

Request URL: http://localhost:8000/

Referrer Policy: strict-origin-when-cross-origin

▼ Form Data view source view URL-encoded

first_name: aaa

last_name: bbbb

- 크롬 개발자도구의 network 탭에서 서버와 주고받는 정보를 확인 가능!

Ajax



- AJAX(Asynchronous Javascript and XML)
- 페이지의 내용이 바뀔 때마다 서버에 내용을 요청하여 페이지를 리로드하는 방식은 아주 비효율적
- 필요한 데이터만을 서버에서 가져와 바뀐 내용만 수정할 경우 페이지를 리로드하지 않으므로 효율적 → Ajax
- Asynchronous: 비동기적 처리, 서버에서 내용이 도달할 때까지 기다리지 않고 다른 작업들을 비동기적으로 처리함

Fetch API



- Ajax를 지원하는 API에는 XMLHttpRequest 등 다른 API들도 존재하지만, 본 세미나에서는 fetch API를 이용하겠습니다!

Fetch API

```
function fetchData(){
  var target=document.querySelector('.fetch');
  fetch('http://localhost:8000')
  .then(function(response){
    return response.json();
  })
  .then(function(info){
    target.innerHTML=
      "university: "+ info["university"]+ "<br>"+
      "group: "+info["group"]+"<br>"+
      "seminar: "+info["seminar"]+"<br>"
    console.log(info);
  })
}
```

- fetch("정보가 존재하는 주소")
- then → 서버로부터 정보가 전송될 때까지 무작정 기다리는 것이 아니라 정보가 전송되면 그 때 실행(Asynchronous)
- fetch API를 통해서 우리는 **response 객체**를 얻게 됨

Fetch API

```
function fetchData(){
  var target=document.querySelector('.fetch');
  fetch('http://localhost:8000')
  .then(function(response){
    return response.json();
  })
  .then(function(info){
    target.innerHTML=
      "university: "+ info["university"]+ "<br>"+
      "group: "+info["group"]+"<br>"+
      "seminar: "+info["seminar"]+"<br>"
    console.log(info);
  })
}
```

```
{ university: 'Postech', group: 'Poapper', seminar: 'front-end' }
```

- JSON: Javascript Object Notation, 서버에서 클라이언트로 데이터를 보낼 때 사용하는 형식
- 객체와 유사하므로 객체처럼 property에 접근할 수 있음!

Fetch API

```
function fetchData(){
  var target=document.querySelector('.fetch');
  fetch('http://localhost:8000')
    .then(function(response){
      return response.json();
    })
    .then(function(info){
      target.innerHTML=
        "university: "+ info["university"]+ "<br>"+
        "group: "+info["group"]+"<br>"+
        "seminar: "+info["seminar"]+"<br>"
      console.log(info);
    })
}
```

```
{ university: 'Postech', group: 'Poapper', seminar: 'front-end' }
```

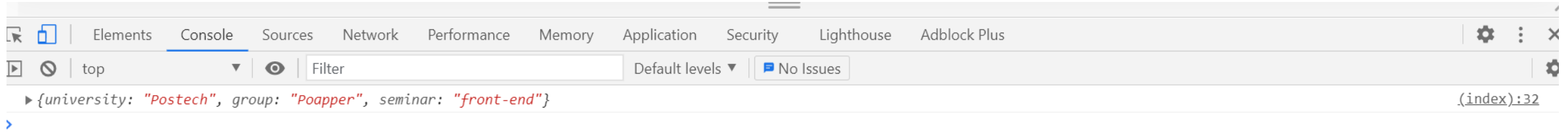
- res.json: 받은 response 객체를 JSON 형태로 바꿈

Fetch API

```
fetch('http://localhost:8000')  
  .then(res=>res.json())  
  .then(info=>{  
    target.innerHTML=  
      "university: "+ info["university"]+ "<br>"+  
      "group: "+info["group"]+"<br>"+  
      "seminar: "+info["seminar"]+"<br>"  
    console.log(info);  
  })
```

- cf) JS의 화살표 함수를 이용하면 앞선 예제의 함수를 조금 더 간결하게 작성할 수 있습니다!
- 인자 => return 값 or 인자 => {처리할 내용}
- res=>res.json() → function(res){return res.json()}

Fetch API



▼ General

Request URL: http://localhost:8000/

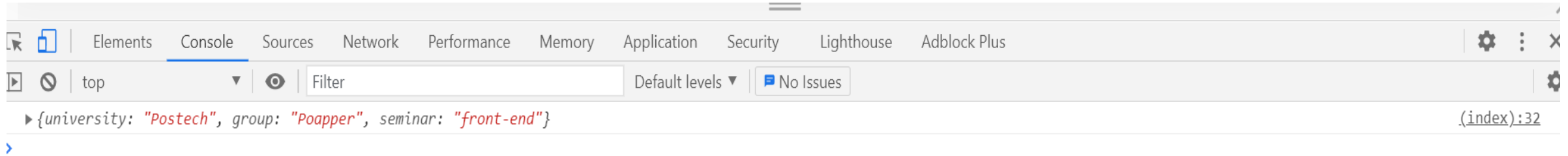
Request Method: GET

Status Code: ● 200 OK

Remote Address: [::1]:8000

- console.log를 통해 크롬 개발자 도구의 콘솔 창에서 변수의 값을 확인할 수 있습니다!

Fetch API



localhost	200	fetch	(index):25	210 B	6 ms
http://localhost:8000/					

- console.log를 통해 크롬 개발자 도구의 콘솔 창에서 변수의 값을 확인할 수 있습니다!
- form tag에서와 마찬가지로 네트워크 탭에서 서버와의 통신을 확인할 수 있습니다!

Fetch API



- 앞서 살펴본 예제는 request의 method를 따로 지정해주지 않아 default로 get method로 데이터를 받아옵니다!
- fetch API를 이용하면 get method 외에도 post, put, delete 등 다른 http method들도 서버로 전송할 수 있습니다!

HTTP Status

▼ General

Request URL: http://localhost:8000/

Request Method: GET

Status Code:  200 OK

Remote Address: [::1]:8000

- 서버에 보낸 request가 정상적으로 완료되었는지를 알려줌
- ex) 200: OK, 404: Not Found
- 서버와 통신이 잘 이루어지지 않을 때 체크!

Reference

Form tag

- https://www.w3schools.com/tags/tag_form.asp
- https://www.w3schools.com/tags/att_form_method.asp
- <https://private.tistory.com/74>

Arrow Function

- https://developer.mozilla.org/ko/docs/Web/JavaScript/Reference/Functions/Arrow_functions

Fetch API

- <https://opentutorials.org/course/3281/20562>

HTTP

- <https://developer.mozilla.org/ko/docs/Web/HTTP/Status>
- <http://tcpschool.com/html-tag-attrs/form-method>
- <https://developer.mozilla.org/ko/docs/Web/HTTP/Methods>
- <https://meetup.toast.com/posts/92>

과제

- 이번 주차에 제출해야 할 과제는 없습니다!!!
- 모두 해커톤 화이팅하세요!
- <https://opentutorials.org/course/3385>
- 시간이 나면 생활코딩 님의 HTTP 강좌를 가볍게 들어보시는 것을 추천드립니다! 해커톤을 하면서 백엔드와 협업을 할 때 미리 알고 있으면 도움이 될 수도 있을 것 같습니다! (30분 정도의 짧은 분량)