

K-Digital Training 웹 풀스택 과정

Form 전송

WITH 팀 리처드



body-parser

body-parser

- 데이터를 쉽게 처리할 수 있도록 도와주는 라이브러리
- Post로 정보를 전송할 때 요청의 body(req.body)로 받을 수 있게 도와 줌
- express 4.x 부터 body-parser가 내장되어 있어 설치 필요 없음

```
app.use(express.urlencoded({ extended: true }));  
app.use(express.json());
```

app.js

form

<form>

- 입력된 데이터를 한 번에 서버로 전송하기 위해 사용
- 즉, 클라이언트가 서버에게 정보를 전달할 때 사용
- 속성 : action, name, target, method
- 폼 요소 : <input>, <select>, <textarea>, <button> 등등

```
<form action="/폼_전송할_서버주소" name="폼이름" method="폼_전송방식">
  <input type="text" />
  <textarea name="" id="" cols="30" rows="10"></textarea>
  <select name="" id="">
    <option value=""></option>
    <option value=""></option>
  </select>
  <button type="submit"></button>
</form>
```

폼 사용 예시

<form> 속성

- **action**

- 폼을 전송할 서버 주소 지정

- **name**

- 폼을 식별하기 위한 이름

```
document.forms['form태그_name속성값']
```

JavaScript에서 위와 같이 폼 선택 가능함

- **method**

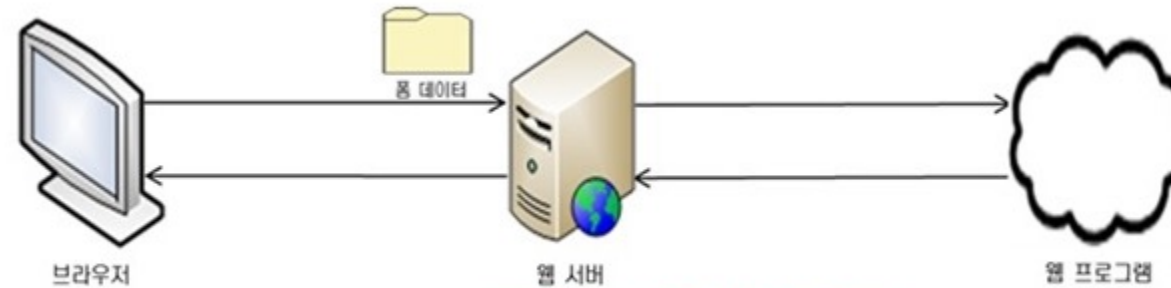
- 폼을 서버에 전송할 http 메소드 지정

- **target**

- action 속성값에 지정한 스크립트 파일을 현재 창이 아닌 다른 위치에서 열 수 있도록 함
- _blank, _self

<form> method

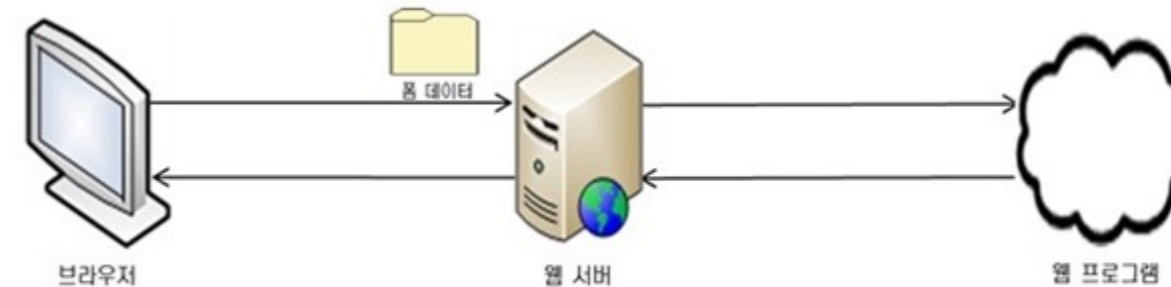
- GET



`http://localhost:8080/form.jsp?name=joon&address=suwon`

-> 폼에 입력한 정보가 url에 추가되어 보인다!!

- POST



`http://localhost:8080/form.jsp`

폼 요소 - <input>

- 사용자가 다양하게 폼 태그를 입력할 수 있게 하는 것
- 입력창
- 가장 기본적인 form 요소

폼 요소 - <input> 속성

- **type** : 인풋 타입
- **name** : 이름 지정. backend에서 **name**으로 **key**가 설정된다.
- **readonly** : 읽기 전용 (수정 불가)
- **autofocus** : 자동 focus
- **placeholder** : 짧은 도움말

폼 요소 - <input> type 속성 값 종류

- text

- radio

☐

남자

- checkbox

☐

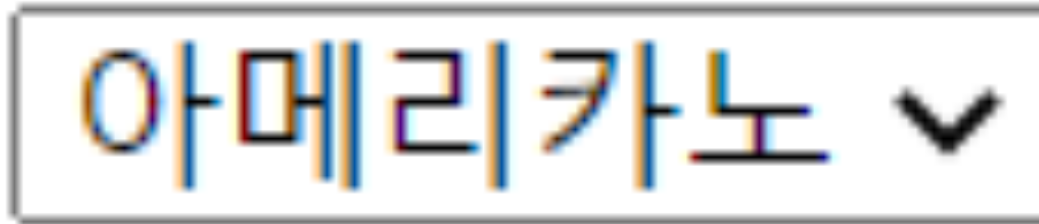
야구

- 등등

[kdt3rd-study/05_JAVASCRIPT_OBJECT/form.html](https://github.com/kdt3rd-study/05_JAVASCRIPT_OBJECT/tree/main/form.html) 파일 확인

폼 요소 - <select>

- 선택창
- 서버가 지정한 특정 값만을 선택할 수 있는 요소
- input이 주관식이라면 select는 객관식



kdt3rd-study/O5_JAVASCRIPT_OBJECT/form.html 파일 확인

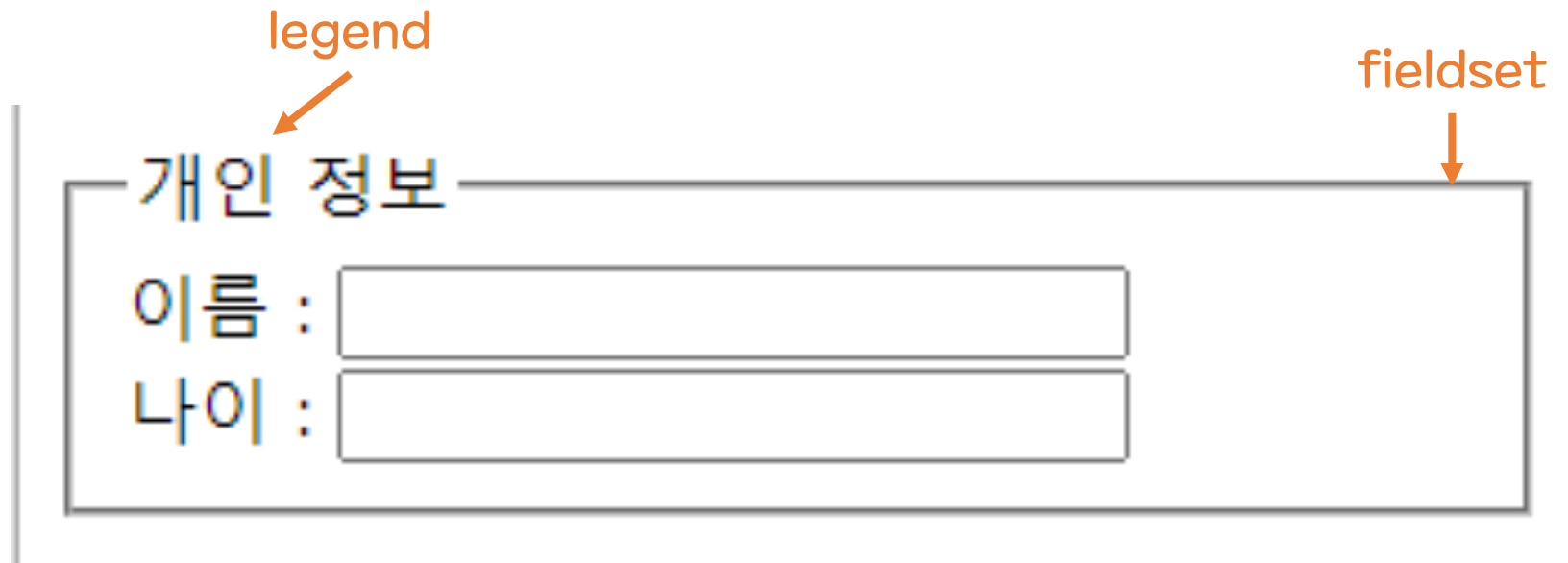
폼 요소 - <label>

- 폼 양식에 이름을 붙일 수 있다.
- for 속성
 - for 속성에 연결할 요소의 id를 적어 label을 클릭해도 해당 요소로 가게 만들 수 있다.

[kdt3rd-study/05_JAVASCRIPT_OBJECT/form.html](#) 파일 확인

폼 요소 - <fieldset> <legend>

- <fieldset> : 폼 태그 안에 있는 요소들을 **그룹화**할 때 사용한다.
- <legend> : <fieldset> 안에 들어가는 태그로, 목적에 맞게 **이름**을 지정할 수 있다.



The diagram illustrates the use of the <fieldset> and <legend> HTML tags. It shows a rectangular box representing a form fieldset. Inside the box, at the top left, is the text '개인 정보' (Personal Information). Below this text are two input fields, one for '이름' (Name) and one for '나이' (Age). An orange arrow labeled 'legend' points to the text '개인 정보'. Another orange arrow labeled 'fieldset' points to the top right corner of the box.

form 받기

Step1. form 정보 받기

폼 실습을 해봅시다

GET 요청

localhost:8000/getForm?id=apple&pw=1234

get 요청 성공!

post 요청

localhost:8000/postForm

post 요청 성공!

index.ejs

result.ejs

Step1. form 정보 받기

```
app.get('/getForm', function (req, res) {  
  console.log(req.query);  
  res.send('get 요청 성공!');  
});  
  
app.post('/postForm', function (req, res) {  
  console.log(req.body);  
  res.send('post 요청 성공!');  
});
```

app.js

```
{ id: 'apple', pw: '1234' }
```

```
{ id: 'banana', pw: '4321' }
```


Step2. 결과 페이지에 데이터 추가

폼 실습을 해봅시다

GET 요청

post 요청

index.ejs

← → ↻ localhost:8000/getForm?id=apple&pw=1234

GET 요청 폼 결과 확인하기

apple님 환영합니다!
비밀번호는 1234이군요!!

← → ↻ localhost:8000/postForm

POST 요청 폼 결과 확인하기

banana님 환영합니다!
비밀번호는 4321이군요!!

result.ejs

Step2. 결과 페이지에 데이터 추가

```
app.get('/getForm', function (req, res) {
  console.log(req.query);
  // res.send('get 요청 성공!');
  res.render('result', {
    title: 'GET 요청 폼 결과 확인하기',
    userInfo: req.query, { id: 'apple', pw: '1234' }
  });
});

app.post('/postForm', function (req, res) {
  console.log(req.body);
  // res.send('post 요청 성공!');
  res.render('result', {
    title: 'POST 요청 폼 결과 확인하기',
    userInfo: req.body, { id: 'banana', pw: '4321' }
  });
});
```

app.js

```
<!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>
      span {
        color: blue;
        font-weight: 700;
      }
    </style>
  </head>
  <body>
    <h1><%= title %></h1>
    <div><span><%= userInfo.id %></span>님 환영합니다!</div>
    <div>비밀번호는 <span><%= userInfo.pw %></span>이군요!!</div>
  </body>
</html>
```

views/result.ejs

폼 요소 입력없이 버튼을 누르면?

빈문자열이 들어감! 😱

```
{ id: '', pw: '' }
```

localhost:8000/getForm?id=&pw=

폼 실습을 해봅시다

GET 요청

GET 요청 폼 결과 확인하기

님 환영합니다!
비밀번호는 이군요!!

post 요청

localhost:8000/postForm

POST 요청 폼 결과 확인하기

님 환영합니다!
비밀번호는 이군요!!

index.ejs

result.ejs

form validation

유효성 검사

form validation (유효성 검사)

- form 요소들에 **정보가 올바르게 입력되었는지 검사**하는 것
 - ex. 비밀번호: 8자리 이상, 특수문자 및 대문자 1개 이상 포함
 - ex. 이메일: @ 기호 반드시 포함

form validation (유효성 검사)

input 태그에 지정 가능한 유효성 검사 기능

- **required** : 필수 값
- **minlength / maxlength** : 최소/최대 문자수
- **min / max** : 최소/최대 값
- **type** : 입력받는 정보 타입
- **pattern** : 정규식으로 검사

pattern 속성에서 정규식 사용

정규식(정규표현식): **regex**(regular expression)

https://developer.mozilla.org/ko/docs/Web/JavaScript/Guide/Regular_Expressions

- `/`: 정규식을 사용한다는 의미 (정규표현식을 `//` 으로 감싼다)
- `^`: 시작
- `$`: 끝
- `[]`: 범위
 - ex) `[a-z]`: a부터 z까지 중 문자 하나
 - ex) `[가-힣]`: 한글의 모든 음절 범위를 표현
- `{}`: 개수
 - ex) `{2, 4}`: 2개 부터 4개까지
- `()`: 그룹 검색 및 분류
- `.`: 임의의 글자 하나
- `+`: 1개 이상의 글자
- `*`: 0개 이상의 모든 문자
- `?`: 0~1번 반복되는 문자열

```
<form action="/postForm" name="login" method="post">
  <input
    type="text"
    name="id"
    placeholder="ID"
    pattern="^[a-zA-Z0-9가-힣]{4,}$"
    title="대소문자, 한글, 숫자, 4글자 이상"
    required
  />
  <input
    type="password"
    name="pw"
    placeholder="PW"
    pattern="^[a-z0-9]{8,12}$"
    title="소문자, 숫자로 8자리 이상 12자리 이하"
    required
  />
  <button type="submit">제출</button>
</form>
```

nodemon

nodemon 패키지

- 서버측 코드 (app.js 등) 가 변경될 때마다 ctrl + c 단축키로 node 명령어를 종료하고 node app.js 명령을 다시 입력하기 귀찮음
- 파일들을 모니터링 하다가 소스코드 변경시 자동으로 node 재실행하는 패키지

```
npm install -g nodemon # 전역 설치  
nodemon -v # 설치 확인(버전 확인)  
nodemon app.js # 진입점 파일 실행
```

참고) 지역 설치 vs. 전역 설치

- **지역 설치** : 해당 프로젝트의 node_modules/ 폴더 안에 패키지 설치
 - 해당 프로젝트 내에서만 사용 가능
- **전역 설치** : **-g** 옵션(global)으로 전역 node_modules/ 폴더 안에 패키지 설치
 - 모든 프로젝트가 공통으로 사용 가능

```
npm install [패키지명] # 지역 설치
```

```
npm install -g [패키지명] # 전역 설치
```