GNB
웹 프로그래밍

세번째 수업 자료

Made by 박은빈
Copy from 양한솔

INDEX

1. 실습

- 목표1 : 메인 페이지 요청 -> Read -> GET # 저번시간 까지
- 목표2 : 게시글 작성 -> Create -> POST
- 목표3 : 게시글 조회 -> Read -> GET # 오늘은 여기까지
- 목표4 : 게시글 수정 -> Update -> PUT
- 목표5 : 게시글 삭제-> Delete -> Delete

1. 실습

1) 게시글 작성하기

mainpage.ejs | index.js

hellognb > routes > index.js > ...

```
1   var express = require('express');
2   var router = express.Router();
3
4   router.get('/',function(req,res){
5     res.render('mainpage');
6   })
7
8   router.post('/',function(req,res){
9     res.render('mainpage');
10  })
11
12  module.exports = router;
13
```

post메소드 사용

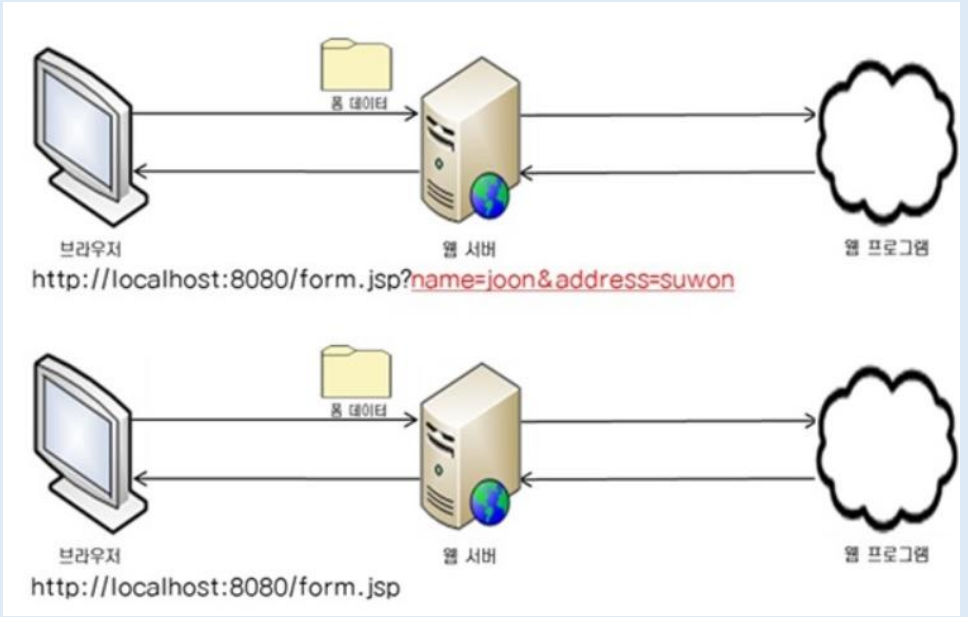
post를 요청 했을 때 시행- mainpage를 띄어줌

1. 실습

- 폼 태그 동장방법: 폼이 있는 웹페이지 방문-> 폼 내용 입력-> 폼 안에 있는 데이터를 웹 서버로 보냄-> 서버에서 데이터 처리->···

- 폼 태그 속성: action: 폼을 전송할 서버 쪽 파일-'/' (localhost:포트번호)에 폼을 전송
  method: 폼을 서버에 전송할 http메소드를 정한다.(GET 또는 POST)
  (get은 눈에 보이게 보내고 post는 내부적으로 보냄)

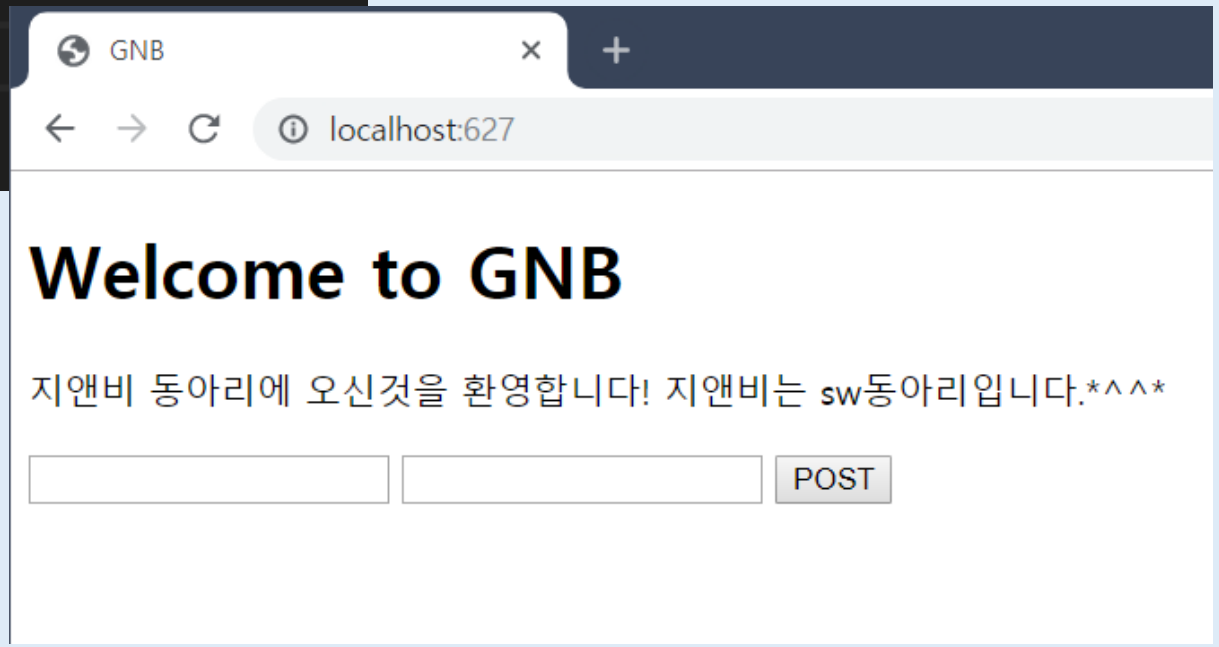
- 참고: http://www.nextree.co.kr/p8428/



1. 실습

- mainpage.ejs에 form 생성 : 제목과 내용을 받아 오기 위한 textbox생성 + 제출 버튼 생성

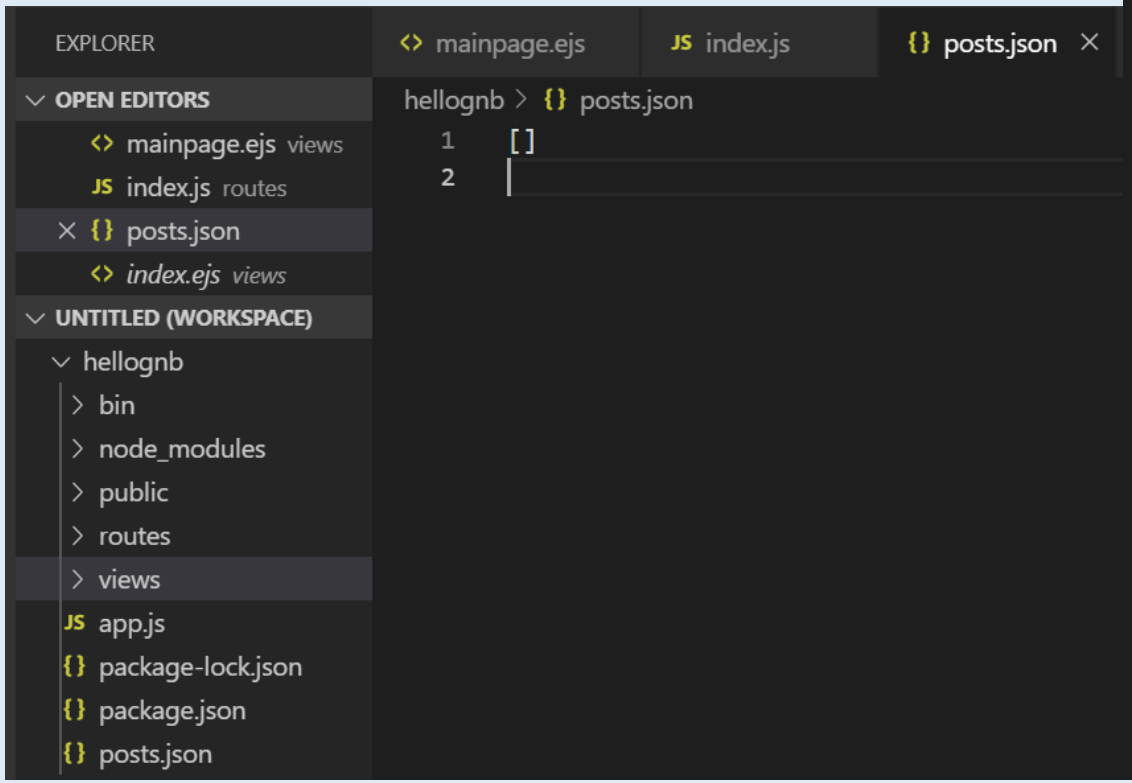
```
<body>
    <h1>Welcome to GNB</h1>
    <p>지앤비 동아리에 오신것을 환영합니다! 지앤비는 sw동
    <form action="/" method="POST">
        <input type="text" name="title">
        <input type="text" name="body">
        <button>POST</button>
    </form>
```



- 내용 입력하고 post하면 ?
- Mainpage.ejs로 감
- Index.js에서 그렇게 설정 했었음!

1. 실습

- form의 정보를 받아서 저장해보자!
- 정보를 담을 post.json파일을 젤 상위 폴더 hellognb에 생성
- 정보를 담을 빈 리스트 생성
- Index.js에서 posts.json불러오기 객체 이름은 posts




```
hellognb > routes > JS index.js > ...
1   var express = require('express');
2   var router = express.Router();
3
4   var posts= require('../posts.json');
5
6   router.get('/',function(req,res){
7     res.render('mainpage');
8   })
9
10  router.post('/',function(req,res){
11    res.render('mainpage');
12  })
13
14  module.exports = router;
```

1. 실습

- form의 정보를 받아서 저장해보자!

- File System 모듈 사용 : https://www.w3schools.com/nodejs/nodejs_filesystem.asp

hellognb > routes > JS index.js > router.post('/') callback

```
1   var express = require('express');
2   var router = express.Router();
3   var fs= require('fs');
4   var posts= require('../posts.json');
5
6   router.get('/',function(req,res){
7     res.render('mainpage');
8   })
```

- fs.writeFile(파일명,넣을 데이터,callback함수)

```
fs.writeFile('posts.json',JSON.stringify(posts),function(){
  res.render('mainpage')
})
```

JSON.stringify()
JavaScript 값이나 객체를 JSON 문자열로 변환

1. 실습

- form의 정보를 받아서 저장해보자!

```
9
10  router.post('/',function(req,res){
11    var newpost ={
12      title: req.body.title,
13      body: req.body.body
14    }
15    posts.push(newpost)
16
17    fs.writeFile('post.json',JSON.stringify(posts),function(){
18      res.render('mainpage')
19    })
20  })
```

newpost라는 리스트형태의 변수 생성

form의 title과 body를 받아옴

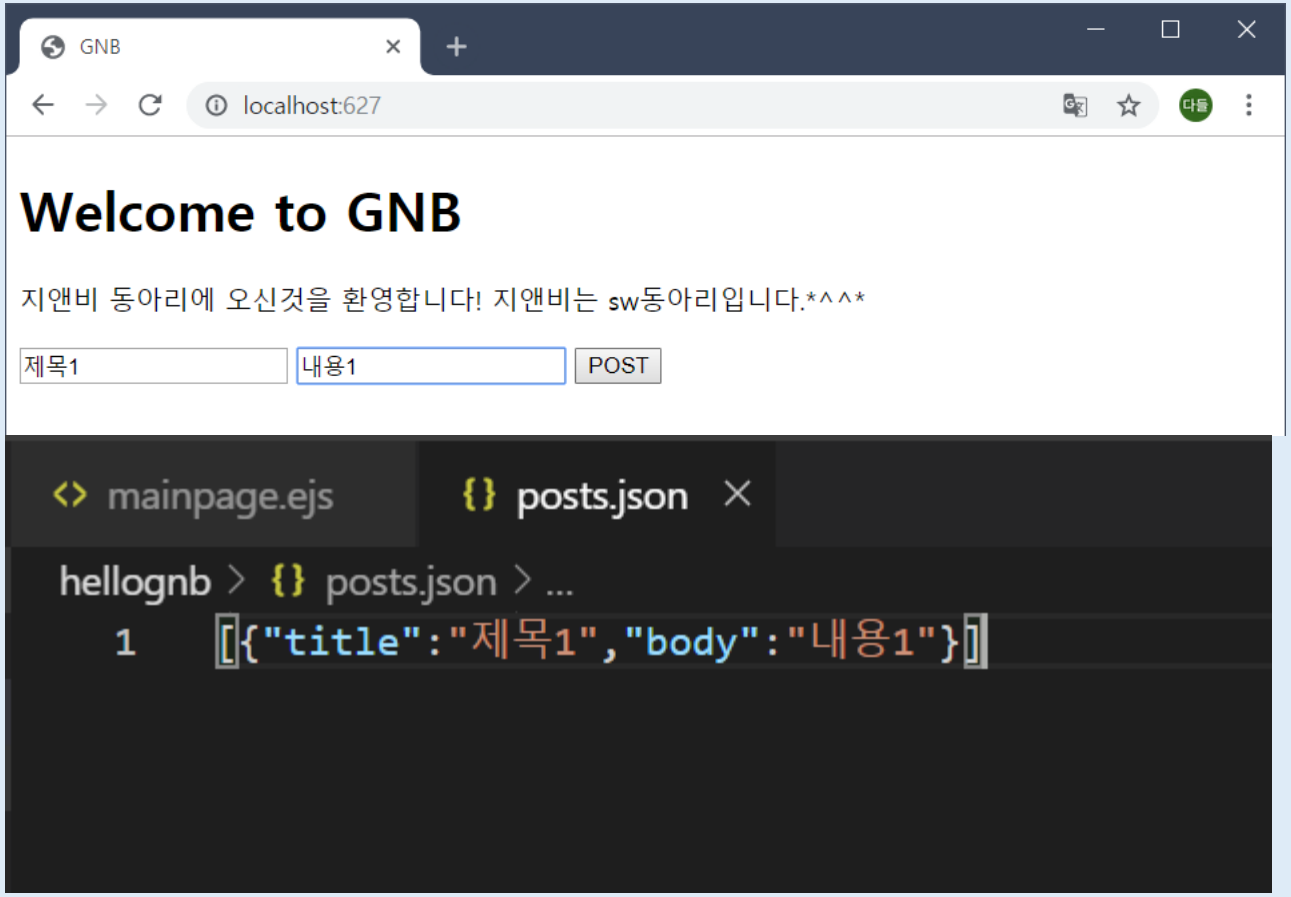
posts에 newpost의 정보를 넣어줌

'post.json' 파일에 posts의 정보를 넣어준다! 단, posts를 json의 문자열 형태로 바꿔주고 넣는다.

그동안 'mainpage'렌더링

1. 실습

- form의 정보를 받아서 저장해보자!
- 실행 하고 폼을 작석해서 post해보자 -> posts.json 에 기록 됨!



1. 실습

2) 게시글 조회하기

- 글 마다 고유한 id가 필요하다!
- Uuid 모듈을 사용한다.
- 터미널 창에서 설치

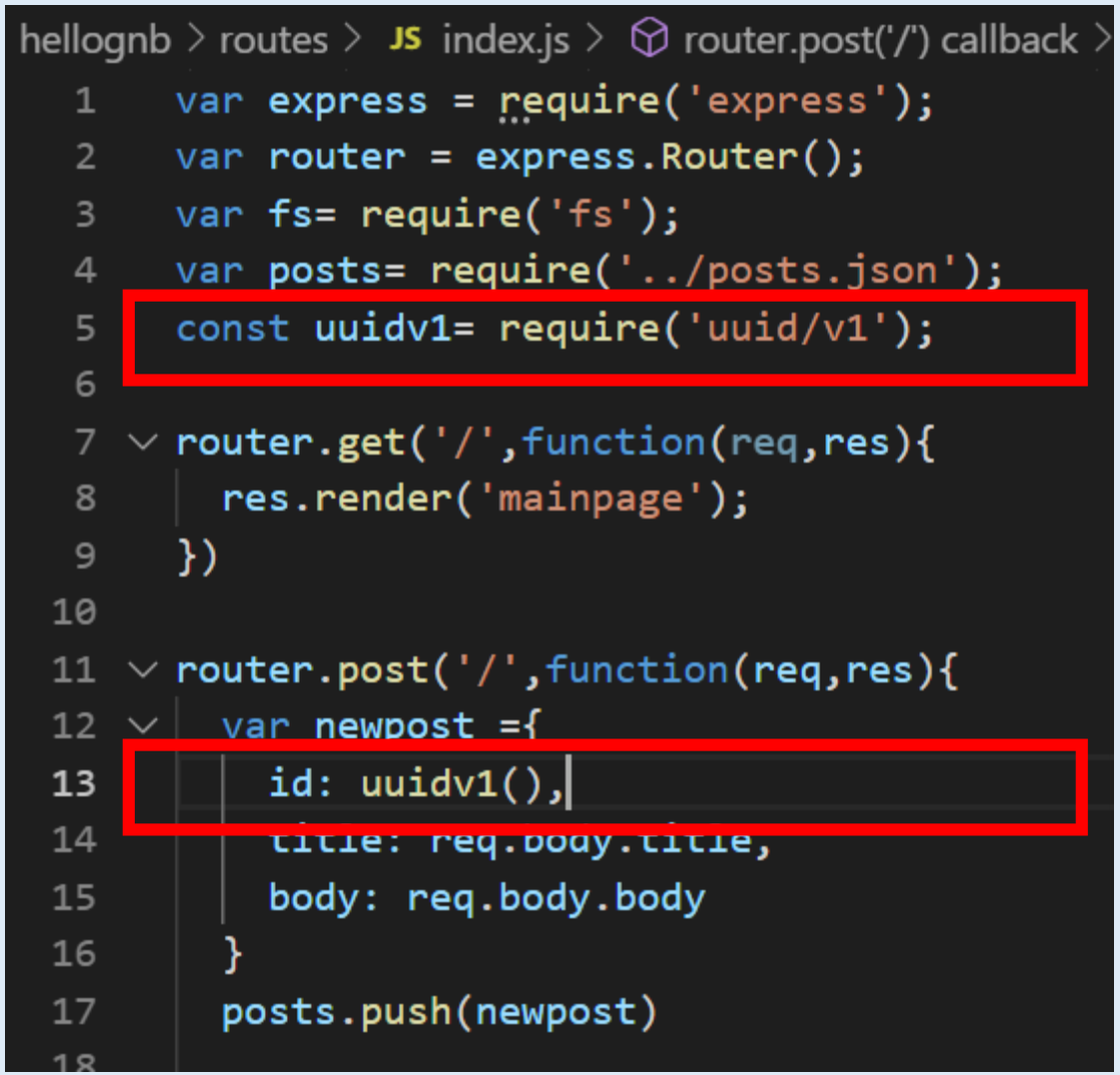
```
PS C:\Users\beb99\hellognb> npm install uuid
+ uuid@3.3.3
added 1 package from 5 contributors and audited 142 packages in 2.4s
found 0 vulnerabilities

PS C:\Users\beb99\hellognb>
```

- uuid/v1 : timestamp
- 그외 참고: https://www.npmjs.com/package/uuid

1. 실습

2) 게시글 조회하기



새로 입력 될 때 마다 고유한 id 값을 준다

1. 실습

2) 게시글 조회하기

```
router.get('/',function(req,res){
  res.render('mainpage',{posts:posts});
})

router.post('/',function(req,res){
  var newpost ={
    id: uuidv1(),
    title: req.body.title,
    body: req.body.body
  }
  posts.push(newpost)

  fs.writeFile('posts.json',JSON.stringify(posts),function(){
    res.render('mainpage',{posts:posts})
  })
})
```

posts.json에 있는 정보를 mainpage.ejs로 넘겨 줘야 조회할 수 있음

객체{사용할 이름: 넘길 정보}형태로 넘겨준다.

1. 실습

2) 게시글 조회하기

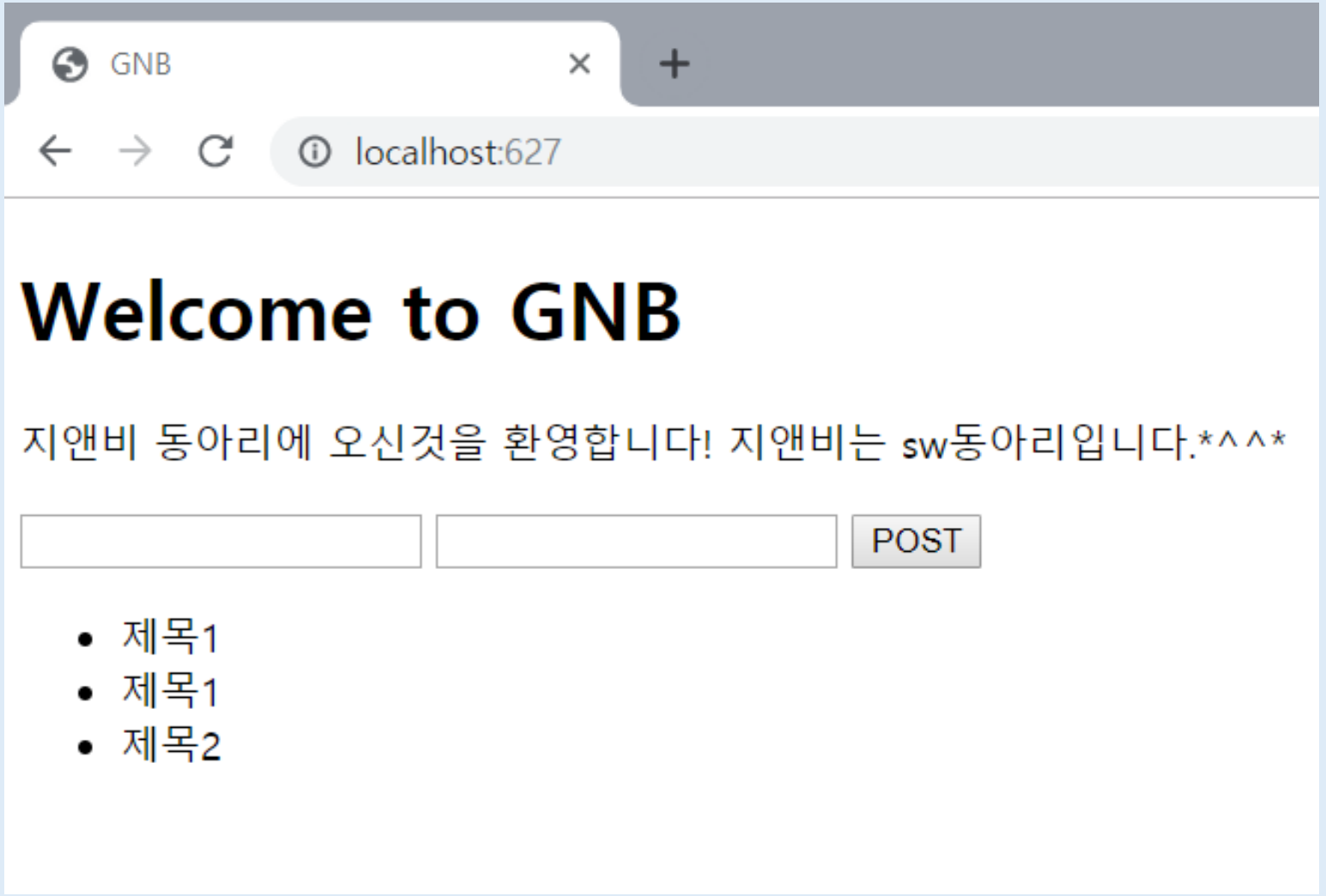
- mainpage.ejs에서 posts정보를 리스트 형태로 나타내고 싶다.
- Ejs에서 js 문법을 사용하기 위해 <% %>를 사용, 출력하기 위해선 <%= %>을 사용
- Ejs: js문법을 사용하기위해 사용하는 view engine

```
        <button>POST</button>
    </form>
    <ul>
        <% for( var i=0; i<posts.length; i++){ %>
            <li><%= posts[i].title %></li>
            </a>
        <% } %>
    </ul>
</body>
```

1. 실습

2) 게시글 조회하기

-실행 시켜 보기



1. 실습

2) 게시글 조회하기

- 제목 눌렀을 때 상세 페이지로 들어가고 싶다!
- <a> 태그 사용: 하이퍼링크를 걸어주는 태그
- <a>속성: href: 클릭 시 이동 할 링크
- 참고: https://ofcourse.kr/html-course/a-%ED%83%9C%EA%B7%B8

```
</form>
<ul>
    <% for( var i=0; i<posts.length; i++){ %>
        <a href="/<%= posts[i].id %>">
        <li><%= posts[i].title %></li>
        </a>
    <% } %>
</ul>
</body>
```

'/' 뒤에 post의 각 아이디 값을 가지는 주소로 이동

1. 실습 2) 게시글 조회하기

- <a>로 연결된 주소를 요청할 때 상세정보가 있는 'detail.ejs'로 가고 싶음.
- View에 제목과 내용 html정보가 있는 detail.ejs파일을 생성
- 서버에 들어오는 GET요청에 대응하는 메소드를 구현하기 위해 Router::get메소드 사용
- 'detail'에 해당하는 path로 렌더링

hellognb > views > detail.ejs > html

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <meta http-equiv="X-UA-Compatible" content="ie=edge">
7       <title>title</title>
8   </head>
9   <body>
10      <h1>제목: title </h1>
11      <p>내용 : body </p>
12  </body>
13  </html>
```

```
router.get('/',function(req,res){
  res.render('mainpage',{posts:posts});
})

router.get('/:post_id',function(req,res){
  res.render('detail')
})
```

1. 실습

- <a>로 연결된 주소가 매번 다름! 하나하나씩 get함수를 써주기 번거롭다.
- 파라미터를 사용한다!

파라미터 - post_id 라는 속성 값을 req.params에 넣어 줌

```
11    router.get('/:post_id',function(req,res){
12      console.log(req)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
params: { post_id: 'bc9d8f80-e73b-11e9-af7e-bf56d5464147' },
query: {},
```

1. 실습

```
router.get('/:post_id',function(req,res){
  var targetIndex= posts.findIndex(function(element){
    return element.id==req.params.post_id
  })
  var targetPost=posts[targetIndex]
  res.render('detail')
})
```

targetIndex에 원하는 post_id가 있는 요소의 인덱스를 넣는다.

findIndex(조건): 조건과 일치하는 인덱스를 반환한다.

targetPost에 posts[위에서 찾은 인덱스]의 정보를 넣는다.

1. 실습

- Missinon : targetPost에 담긴 정보를 detail.ejs로 보내고 주소를 요청했을 때 그 내용이 화면에 보여지게 해보자

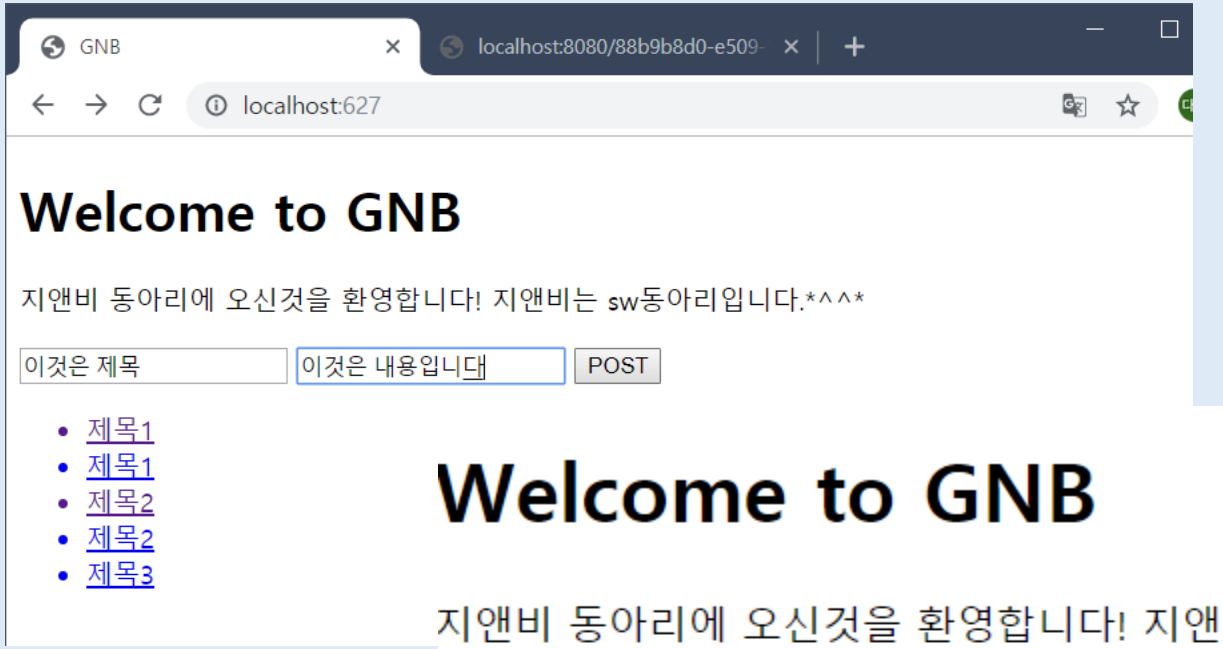
- Hint: {사용할 이름: 보낼 정보}

1. 실습

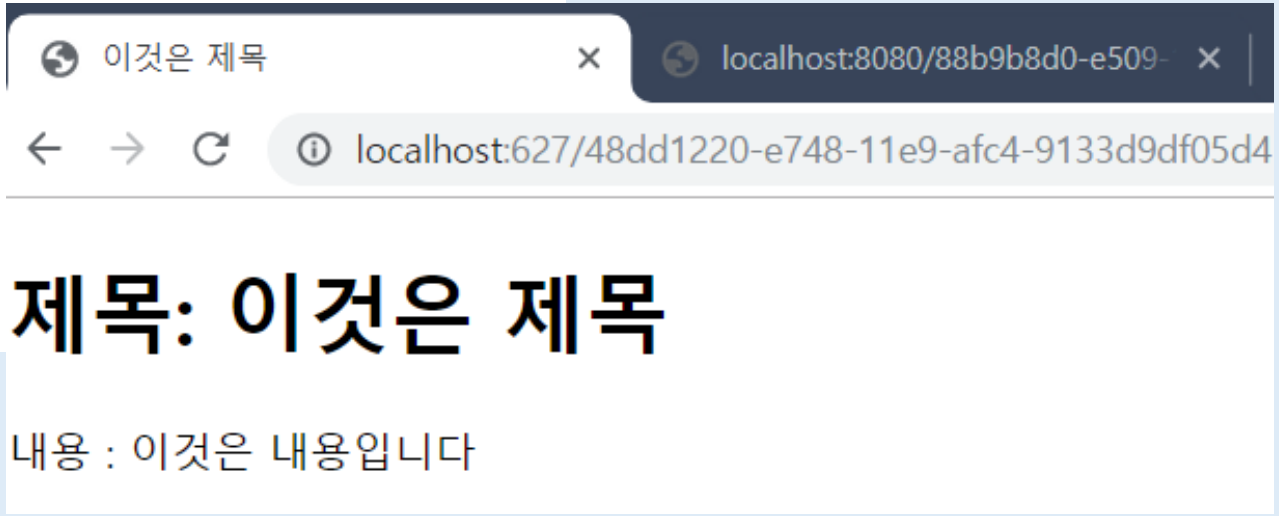
```
router.get('/:post_id',function(req,res){
  var targetIndex= posts.findIndex(function(element){
    return element.id==req.params.post_id
  })
  var targetPost=posts[targetIndex]
  res.render('detail' {post:targetPost})
})
```

hellognb > views > detail.ejs > html

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initial-scale=1.0">
6       <meta http-equiv="X-UA-Compatible" content="ie=edge">
7       <title><%= post.title %></title>
8   </head>
9   <body>
10      <h1>제목: <%= post.title %> </h1>
11      <p>내용 : <%= post.body %> </p>
12  </body>
13  </html>
```


GNB
localhost:8080/88b9b8d0-e509-
localhost:627
Welcome to GNB
지앤비 동아리에 오신것을 환영합니다! 지앤비는 sw동아리입니다.*^^*
이것은 제목
이것은 내용입니대
POST
제목1
제목1
제목2
제목2
제목3


Welcome to GNB
지앤비 동아리에 오신것을 환영합니다! 지앤비는 sw동아리입니다.*^^*
POST
제목1
제목1
제목2
제목2
제목3
이것은 제목


이것은 제목
localhost:8080/88b9b8d0-e509-
localhost:627/48dd1220-e748-11e9-afc4-9133d9df05d4
제목: 이것은 제목
내용 : 이것은 내용입니다

오늘은 여기까지!
수고했어요*^^*