목적

새로운 웹 프레임워크 및 라이브러리를 사용하여 최단기간 crud 한본 생성을 목표로 함

기술

Java1.7/SpringBoot2.6.3/React17.0.2/Gradle/h2 database/Jpa/Lombok/npm/toast-ui/Material-ui 등

작성자

22/03/23 최초작성 정용후

추후 보완

ㄱ. 리액트 부분 class 폴더 생성후 과정 추가 및 가이드 문서 react부분 재작성

ㄴ. 가이드 문서 상세 설명 추가

ㄷ. git추가 / 추후 버전업을 한다면 환경설정이나 코드분리, 공통모듈 부분 추가 작성

출처 : 생활코딩-React (Egoing Lee) / 인프런-스프링 입문 (김영한)

목차

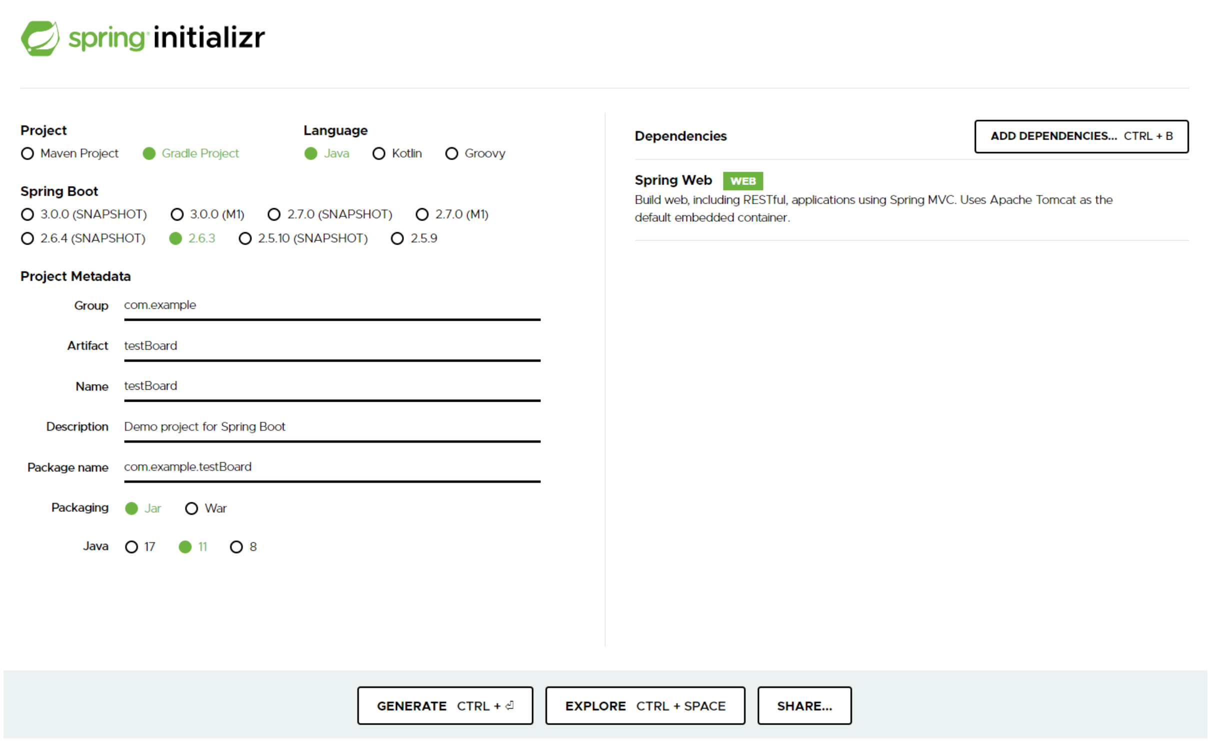
1. Spring Boot

1-1. 자바 11설치

https://blog.naver.com/qodlgks123/222616577711

1-2. 스프링부트 프로젝트 시작

Gradle이나 Maven과 같은 빌드 도구를 이용할 수 있지만 조금 더 손쉽고 편한 방법으로 프로젝트를 만들기 위해 스프링 부트 프로젝트를 Spring initializr를 이용
https://start.spring.io



Project: Maven Project [✓ Gradle Project]

Language: [✓ Java] Kotlin Groovy

Spring Boot: 3.0.0 (SNAPSHOT) 3.0.0 (M1) 2.7.0 (SNAPSHOT) 2.7.0 (M1) 2.6.4 (SNAPSHOT) [✓ 2.6.3] 2.5.10 (SNAPSHOT) 2.5.9

Project Metadata

Group	com.example
Artifact	testBoard
Name	testBoard
Description	Demo project for Spring Boot
Package name	com.example.testBoard
Packaging	[✓ Jar] War
Java	17 [✓ 11] 8

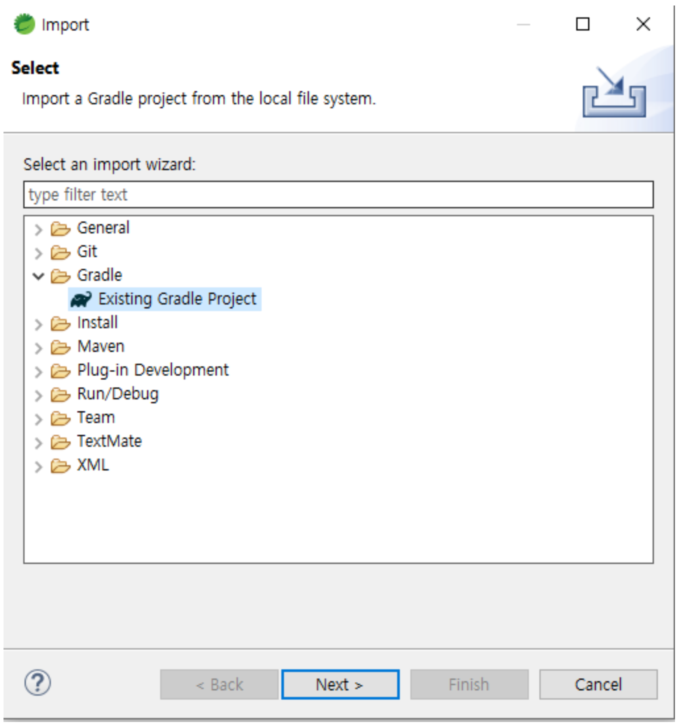
Dependencies

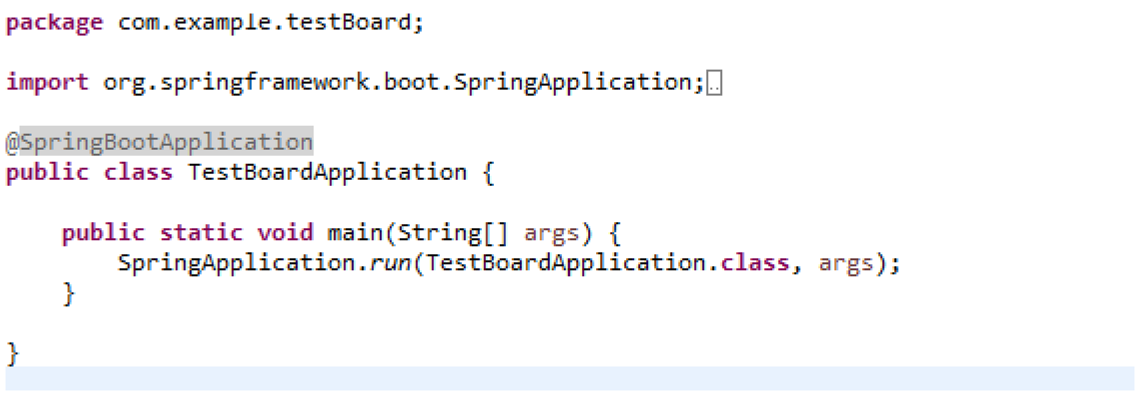
Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

- Group: 보통 기업의 도메인 명
- Artifact: 빌드 결과물 이름, 보통 프로젝트명
- Name: 프로젝트 이름
- Description: 프로젝트 설명
- Package name: 패키지 이름, 초기 소스들이 만들어질 기본 패키지명
- Packaging: 배포 형태 (war, jar)
- Java: 자바 버전 선택
- Dependencies : 이후에 설정 파일(ex: build.gradle)에 직접 등록해도 되지만, 프로젝트 생성 전에도 라이브러리를 미리 주입할 수 있다.

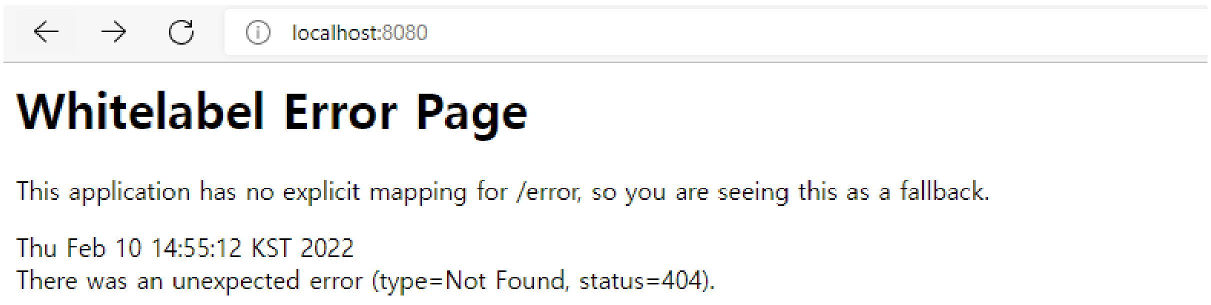
1-3. 프로젝트 임포트 및 실행

testBoard.zip





```java
package com.example.testBoard;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class TestBoardApplication {

    public static void main(String[] args) {
        SpringApplication.run(TestBoardApplication.class, args);
    }

}
```



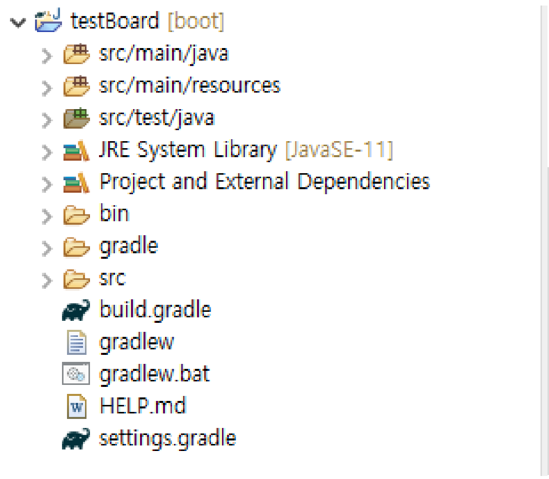
localhost:8080

Whitelabel Error Page

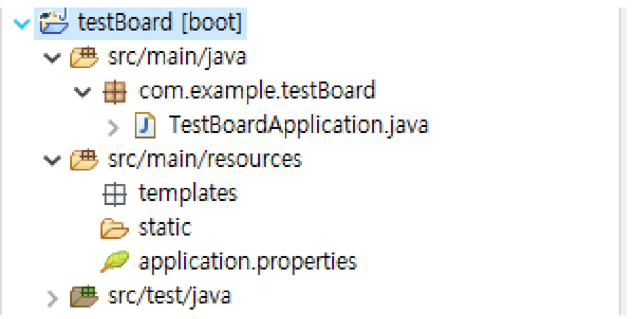
This application has no explicit mapping for /error, so you are seeing this as a fallback.

Thu Feb 10 14:55:12 KST 2022
There was an unexpected error (type=Not Found, status=404).

1-4. 테스트 프로젝트 구조

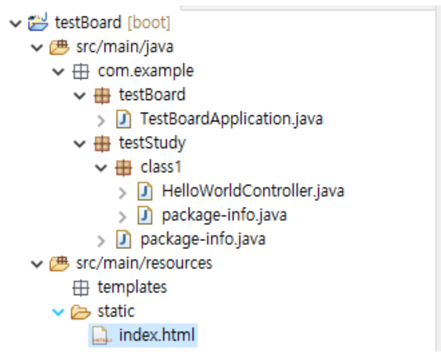


```
testBoard [boot]
  src/main/java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE-11]
  Project and External Dependencies
  bin
  gradle
  src
  build.gradle
  gradlew
  gradlew.bat
  HELP.md
  settings.gradle
```

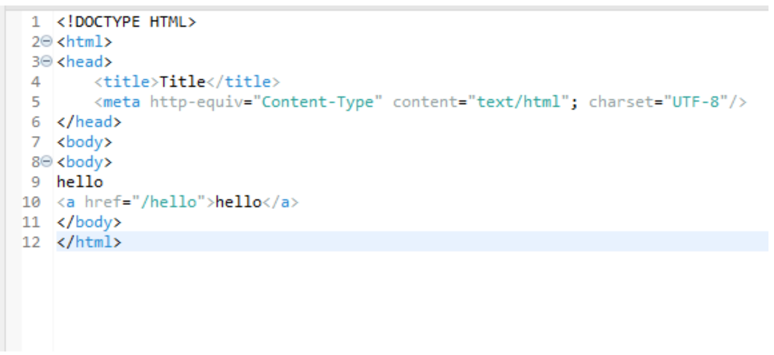


```
testBoard [boot]
  src/main/java
    com.example.testBoard
      TestBoardApplication.java
  src/main/resources
    templates
    static
    application.properties
  src/test/java
```

Static/index.html 웰컴페이지 기능

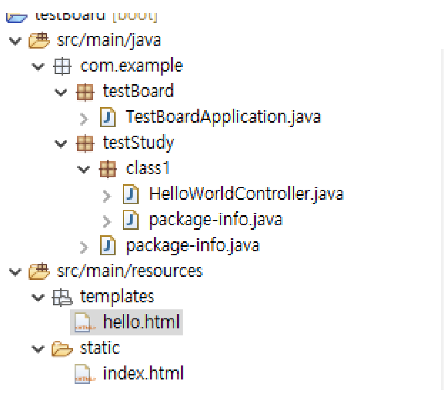


```
testBoard [boot]
  src/main/java
    com.example
      testBoard
        TestBoardApplication.java
      testStudy
        class1
          HelloWorldController.java
          package-info.java
        package-info.java
  src/main/resources
    templates
    static
      index.html
```



```html
<!DOCTYPE HTML>
<html>
<head>
    <title>Title</title>
    <meta http-equiv="Content-Type" content="text/html"; charset="UTF-8"/>
</head>
<body>
<body>
hello
<a href="/hello">hello</a>
</body>
</html>
```

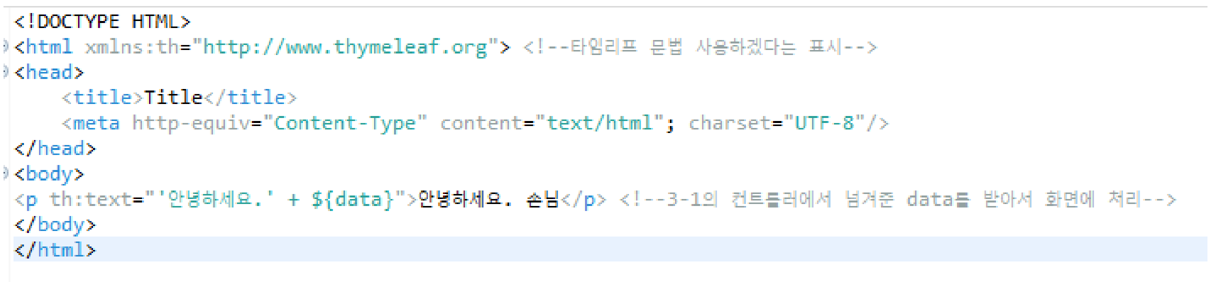
Controller.class 생성후 hello 페이지 이동



```
testBoard [boot]
  src/main/java
    com.example
      testBoard
        TestBoardApplication.java
      testStudy
        class1
          HelloWorldController.java
          package-info.java
        package-info.java
  src/main/resources
    templates
      hello.html
    static
      index.html
```

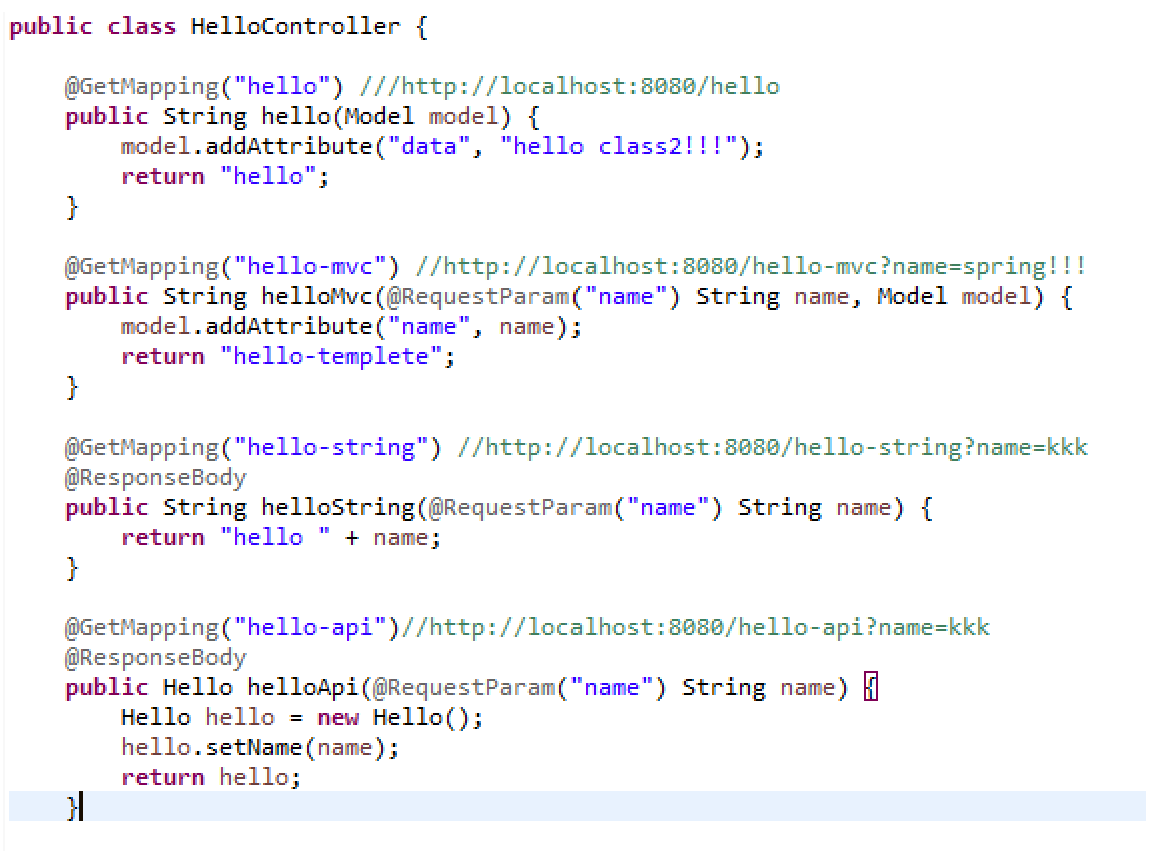
hello hello

1-5. 타임리프



```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org"> <!--타임리프 문법 사용하겠다는 표시-->
<head>
    <title>Title</title>
    <meta http-equiv="Content-Type" content="text/html"; charset="UTF-8"/>
</head>
<body>
<p th:text="'안녕하세요.' + ${data}">안녕하세요. 손님</p> <!--3-1의 컨트롤러에서 넘겨준 data를 받아서 화면에 처리-->
</body>
</html>
```

1-6. Controller.java 데이터 전달방식



```
public class HelloController {

    @GetMapping("hello") ///http://localhost:8080/hello
    public String hello(Model model) {
        model.addAttribute("data", "hello class2!!!");
        return "hello";
    }

    @GetMapping("hello-mvc") //http://localhost:8080/hello-mvc?name=spring!!!
    public String helloMvc(@RequestParam("name") String name, Model model) {
        model.addAttribute("name", name);
        return "hello-templete";
    }

    @GetMapping("hello-string") //http://localhost:8080/hello-string?name=kkk
    @ResponseBody
    public String helloString(@RequestParam("name") String name) {
        return "hello " + name;
    }

    @GetMapping("hello-api")//http://localhost:8080/hello-api?name=kkk
    @ResponseBody
    public Hello helloApi(@RequestParam("name") String name) {
        Hello hello = new Hello();
        hello.setName(name);
        return hello;
    }
```

참고 : spring-boot-devtools 라이브러리 추가

1-7. 빌드

cmd 해당폴더 gradlew build

build - libs 에 가면 jar파일 생성됨 - 서버에 올리고 java –jar 해서 실행

2. React

2-1. 리액트 설치

https://ko.reactjs.org/docs/getting-started.html

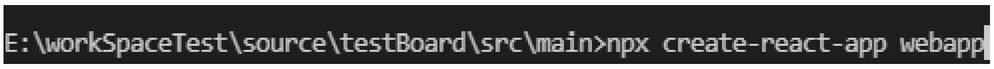
https://github.com/facebook/create-react-app 퀵실행해도 되지만 create react app사용

window+r cmd npm-v(node-v) - nodejs - npm - create react app

npm install -g create-react-app (어디서든 사용하게)

sudo npm install -g create-react-app (권한이 없어서 에러가 뜰때) - 비번

create-react-app -V



```
E:\workSpaceTest\source\testBoard\src\main>npx create-react-app webapp
```

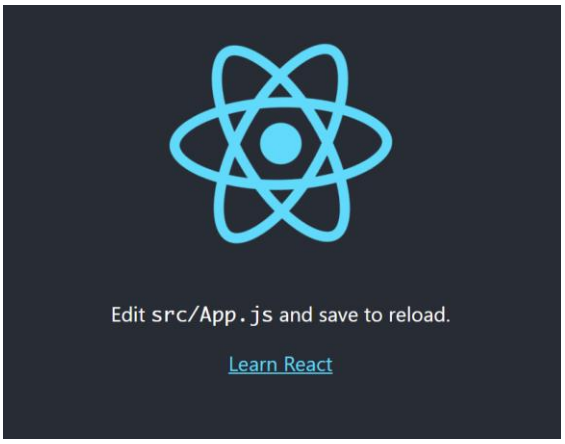
2-2. visual studio code 에디터 설치

view -appreance - terminal -> 터미널 제어 가능

2-3. 기본 서버 실행 및 빌드

npm run start – 서버 실행

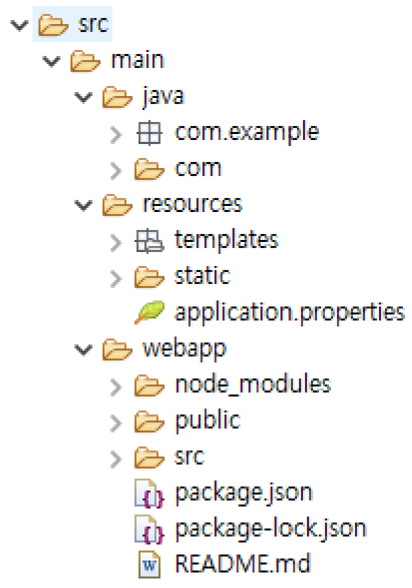
npm run build - 빌드파일 - 이걸 실제 웹서버의 root에 빌드 안쪽 파일에 위치하면 구동됨



Edit src/App.js and save to reload.

Learn React

2-4. create-react-app 구조 - 5장에서 추가 기술

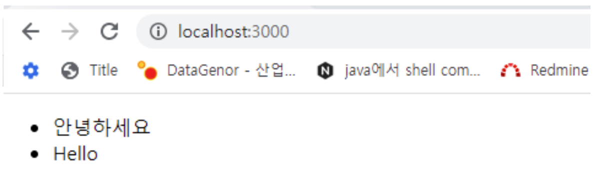


```
src
  main
    java
      com.example
      com
    resources
      templates
      static
      application.properties
    webapp
      node_modules
      public
      src
      package.json
      package-lock.json
      README.md
```

https://kth990303.tistory.com/210

2-5. 리액트가 빠른 이유

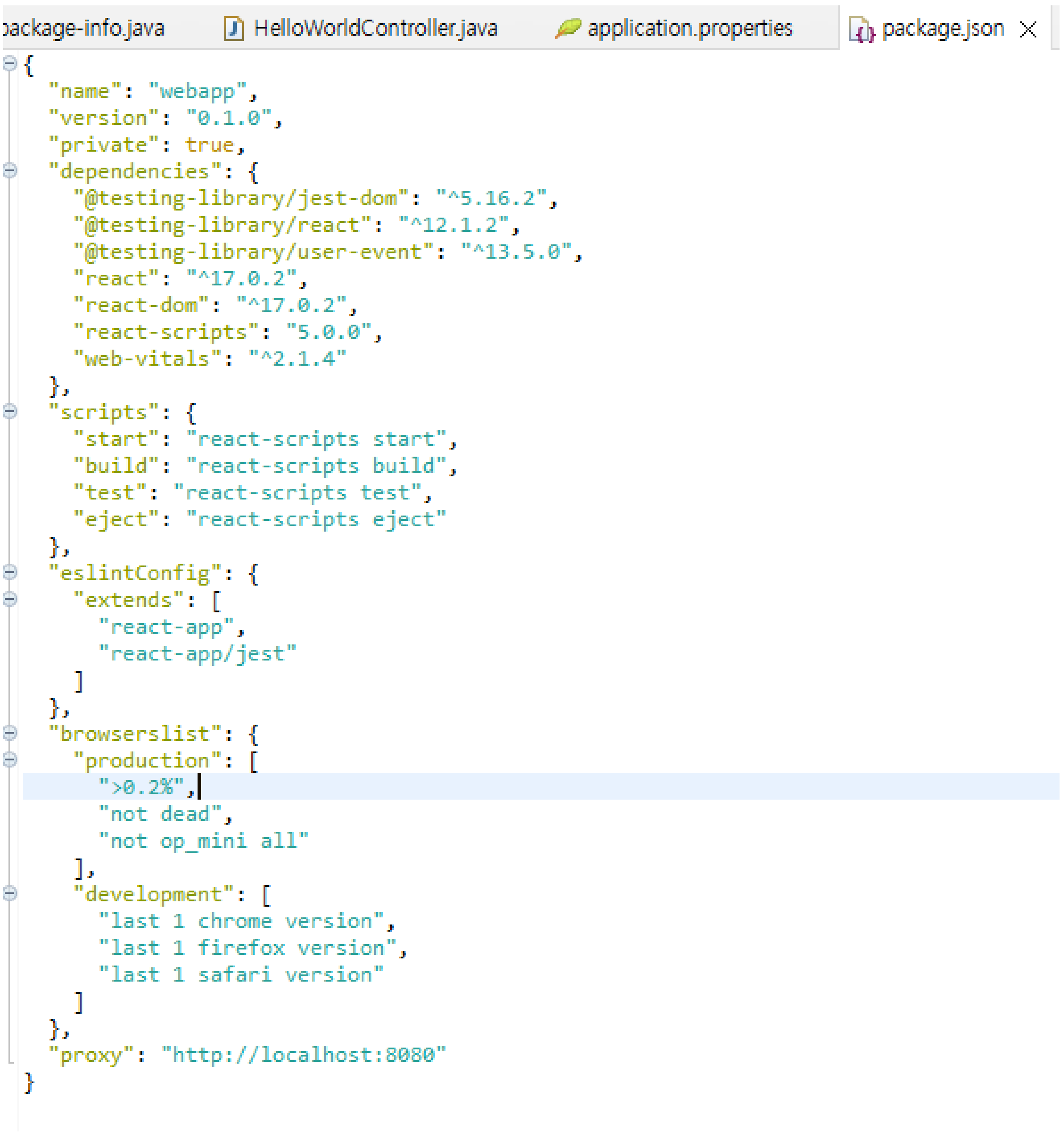
https://ssungkang.tistory.com/entry/React-React-%EC%8B%9C%EC%9E%91%ED%95%98%EA%B8%B0-create-react-app



- 안녕하세요
- Hello

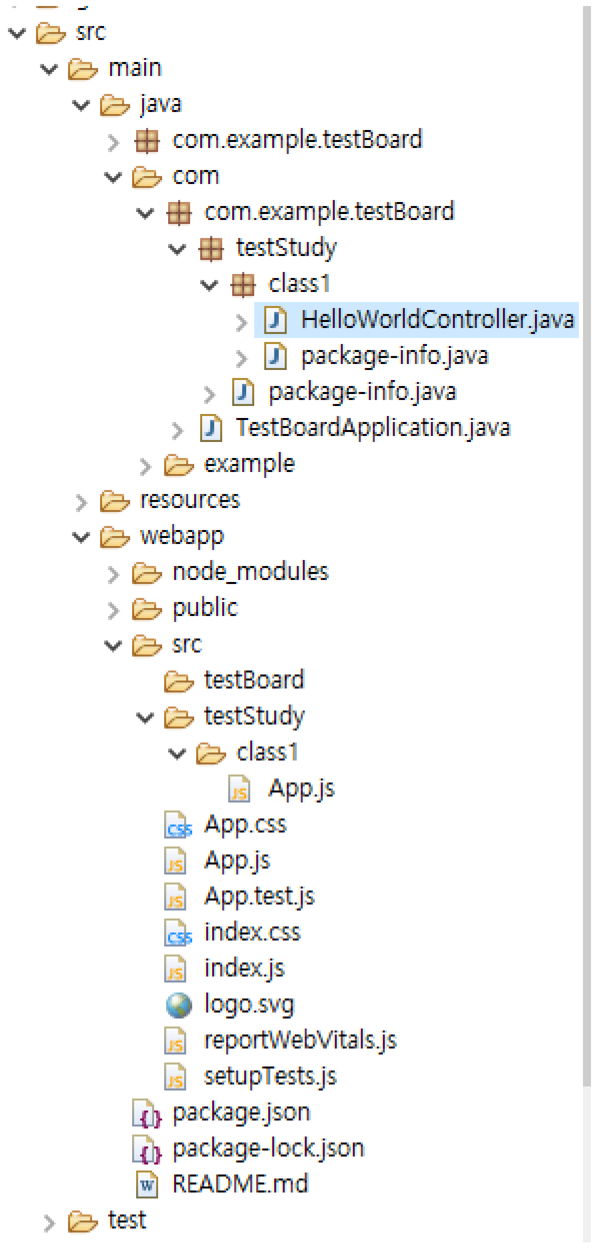
3. React-Spring Boot 연동

3-1. package.json



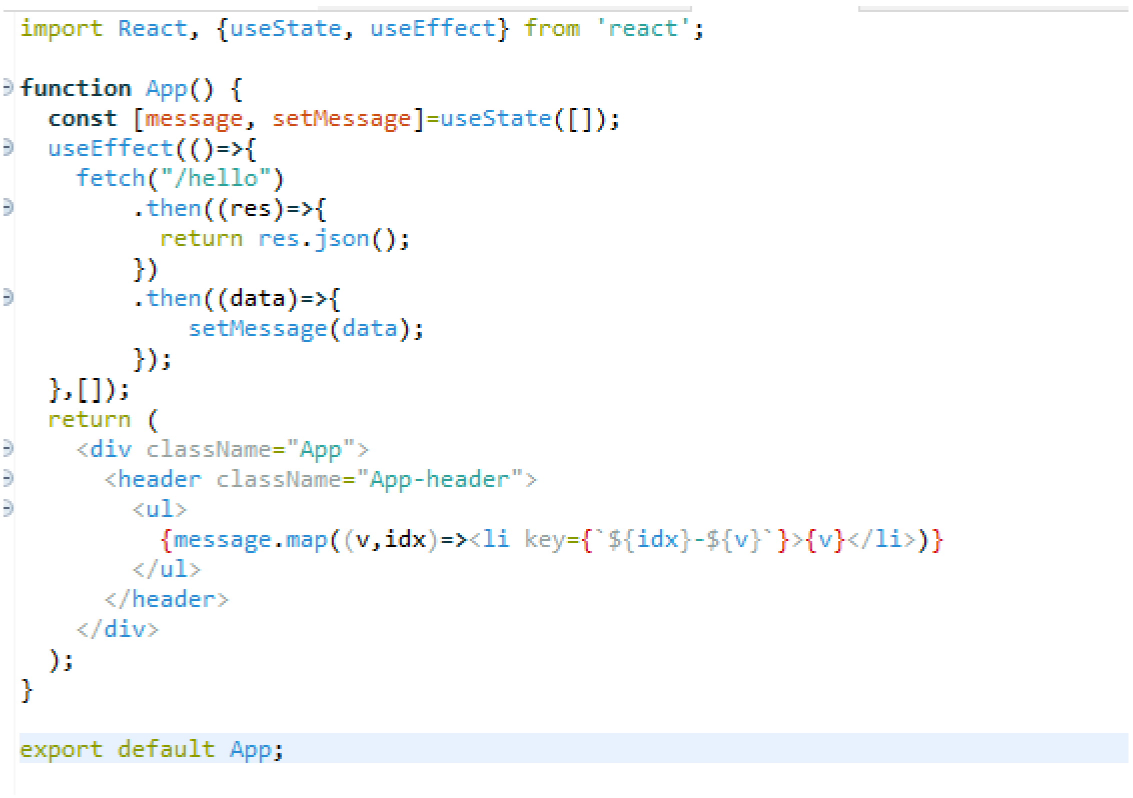
```json
{
  "name": "webapp",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.2",
    "@testing-library/react": "^12.1.2",
    "@testing-library/user-event": "^13.5.0",
    "react": "^17.0.2",
    "react-dom": "^17.0.2",
    "react-scripts": "5.0.0",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  },
  "proxy": "http://localhost:8080"
}
```

3-2. 현재 패키지 구조

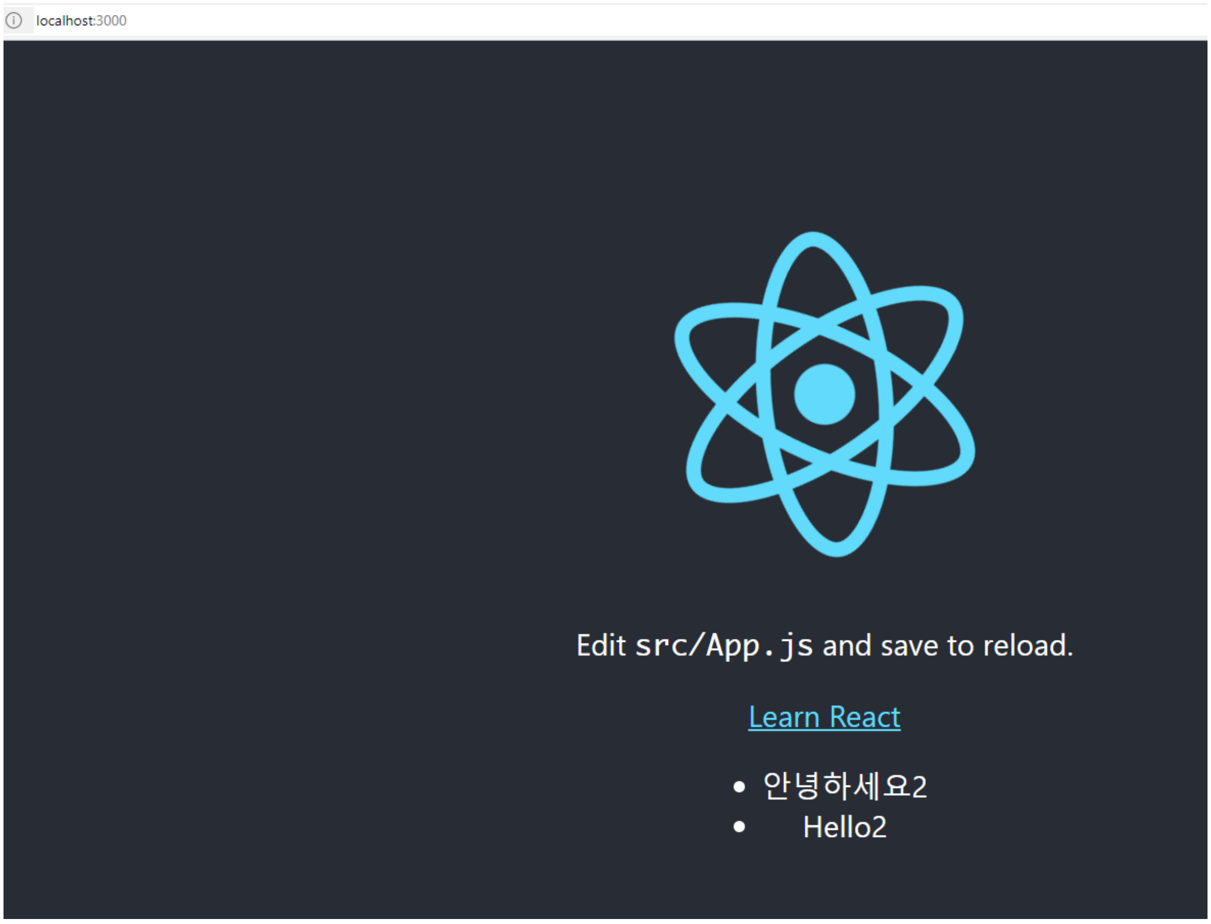


```
src
  main
    java
      com.example.testBoard
      com
        com.example.testBoard
          testStudy
            class1
              HelloWorldController.java
              package-info.java
            package-info.java
          TestBoardApplication.java
        example
    resources
    webapp
      node_modules
      public
      src
        testBoard
        testStudy
          class1
            App.js
        App.css
        App.js
        App.test.js
        index.css
        index.js
        logo.svg
        reportWebVitals.js
        setupTests.js
      package.json
      package-lock.json
      README.md
  test
```

3-3. App.js 그외 소스는 class2참고



```
import React, {useState, useEffect} from 'react';

function App() {
  const [message, setMessage]=useState([]);
  useEffect(()=>{
    fetch("/hello")
        .then((res)=>{
          return res.json();
        })
        .then((data)=>{
            setMessage(data);
        });
  },[]);
  return (
    <div className="App">
      <header className="App-header">
        <ul>
          {message.map((v,idx)=><li key={`${idx}-${v}`}>{v}</li>)}
        </ul>
      </header>
    </div>
  );
}

export default App;
```



localhost:3000

Edit `src/App.js` and save to reload.

Learn React

- 안녕하세요2
- Hello2

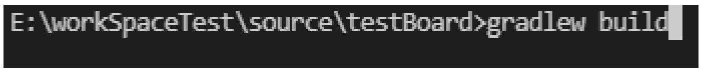
설정한 build.gradle의 내용은 SpringBoot 프로젝트가 build 될 때 React 프로젝트를 먼저 build 하고 결과물을 SpringBoot 프로젝트 build 결과물에 포함시킨다는 스크립트

3-4. React-spring build

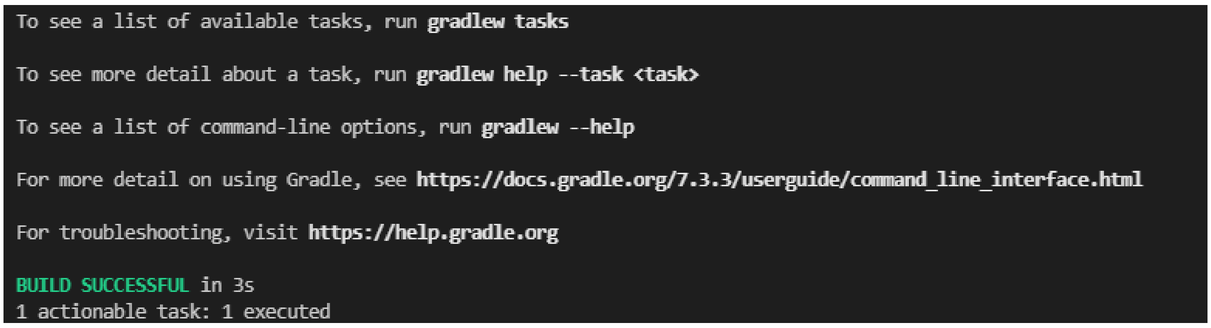
```
def frontendDir = "$projectDir/frontend"

sourceSets {
        main {
                resources {
                        srcDirs = ["$projectDir/src/main/resources"]
                }
        }
}

processResources {
        dependsOn "copyReactBuildFiles"
}

task installReact(type: Exec) {
        workingDir "$frontendDir"
        inputs.dir "$frontendDir"
        group = BasePlugin.BUILD_GROUP
        if (System.getProperty('os.name').toLowerCase(Locale.ROOT).contains('window
s')) {
                commandLine "npm.cmd", "audit", "fix"
                commandLine 'npm.cmd', 'install'
        } else {
                commandLine "npm", "audit", "fix"
                commandLine 'npm', 'install'
        }
}

task buildReact(type: Exec) {
        dependsOn "installReact"
        workingDir "$frontendDir"
        inputs.dir "$frontendDir"
        group = BasePlugin.BUILD_GROUP
        if (System.getProperty('os.name').toLowerCase(Locale.ROOT).contains('window
s')) {
                commandLine "npm.cmd", "run-script", "build"
        } else {
                commandLine "npm", "run-script", "build"
        }
}

task copyReactBuildFiles(type: Copy) {
        dependsOn "buildReact"
        from "$frontendDir/build"
        into "$projectDir/src/main/resources/static"
}
```

설정한 build.gradle의 내용은 SpringBoot 프로젝트가 build 될 때 React 프로젝트를 먼저 build 하고 결과물을 SpringBoot 프로젝트 build 결과물에 포함시킨다는 스크립트

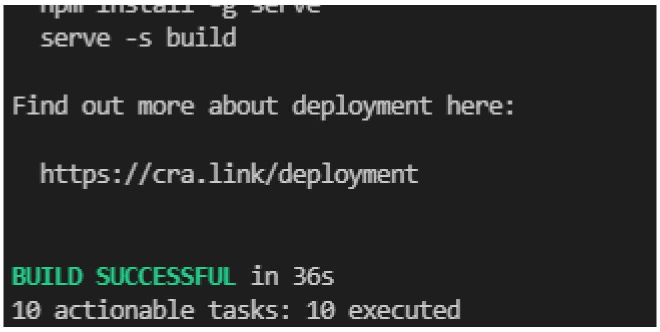
빌드



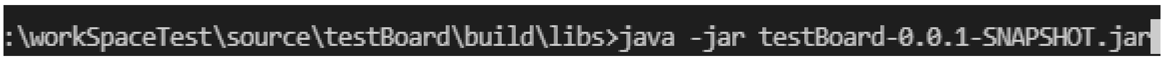
```
E:\workSpaceTest\source\testBoard>gradlew build
```



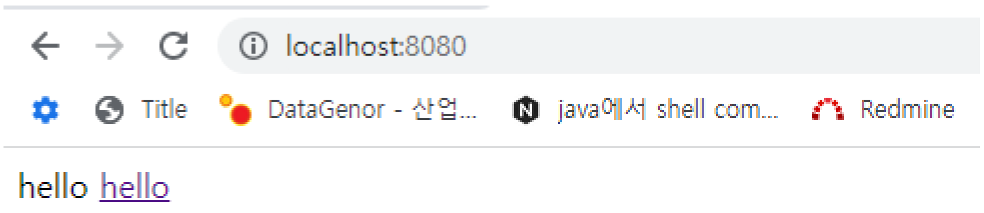
```
To see a list of available tasks, run gradlew tasks

To see more detail about a task, run gradlew help --task <task>

To see a list of command-line options, run gradlew --help

For more detail on using Gradle, see https://docs.gradle.org/7.3.3/userguide/command_line_interface.html

For troubleshooting, visit https://help.gradle.org

BUILD SUCCESSFUL in 3s
1 actionable task: 1 executed
```



```
  serve -s build

Find out more about deployment here:

  https://cra.link/deployment


BUILD SUCCESSFUL in 36s
10 actionable tasks: 10 executed
```



```
:\workSpaceTest\source\testBoard\build\libs>java -jar testBoard-0.0.1-SNAPSHOT.jar
```

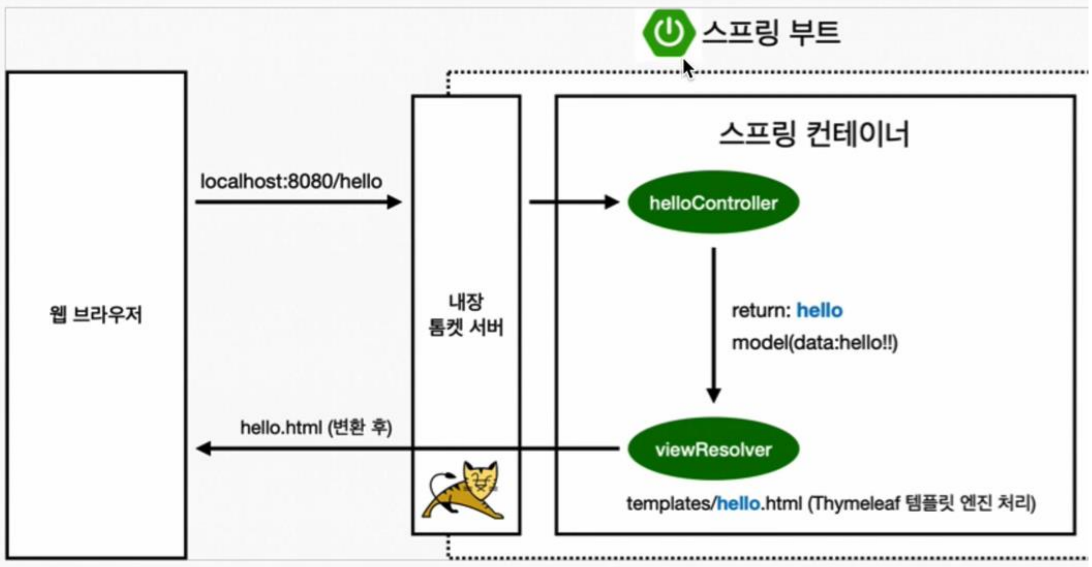


hello hello

참고 : https://7942yongdae.tistory.com/136

3-5. 동작 프로세스

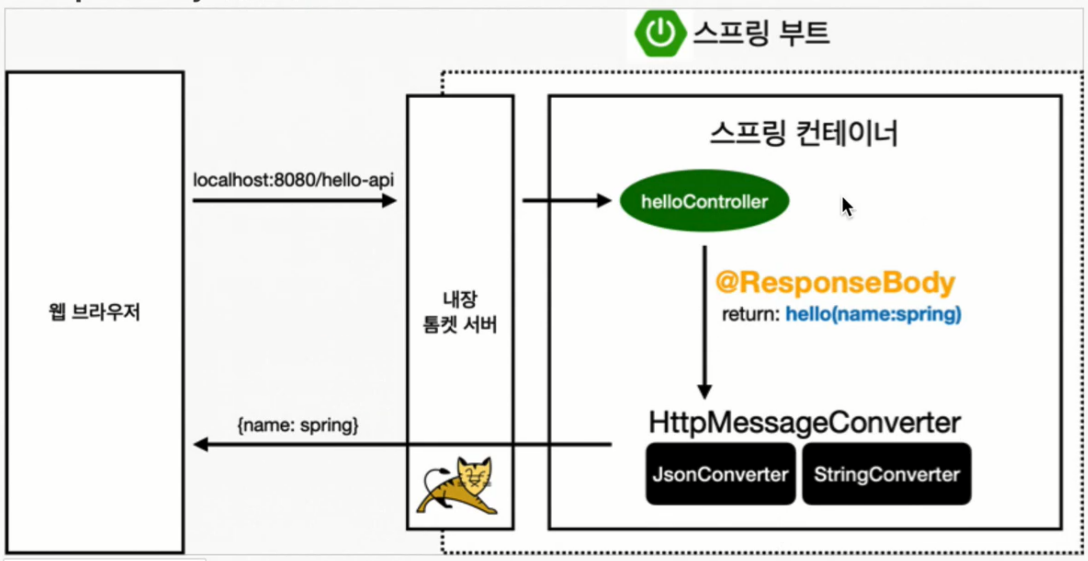
동작 환경 그림



- 컨트롤러에서 리턴 값으로 문자를 반환하면 뷰 리졸버(`viewResolver`)가 화면을 찾아서 처리한다.
    - 스프링 부트 템플릿엔진 기본 viewName 매핑
    - `resources:templates/` +{ViewName}+ `.html`

> 참고: `spring-boot-devtools` 라이브러리를 추가하면, `html` 파일을 컴파일만 해주면 서버 재시작 없이 View 파일 변경이 가능하다.
> 인텔리J 컴파일 방법: 메뉴 build → Recompile

@ResponseBody 사용 원리

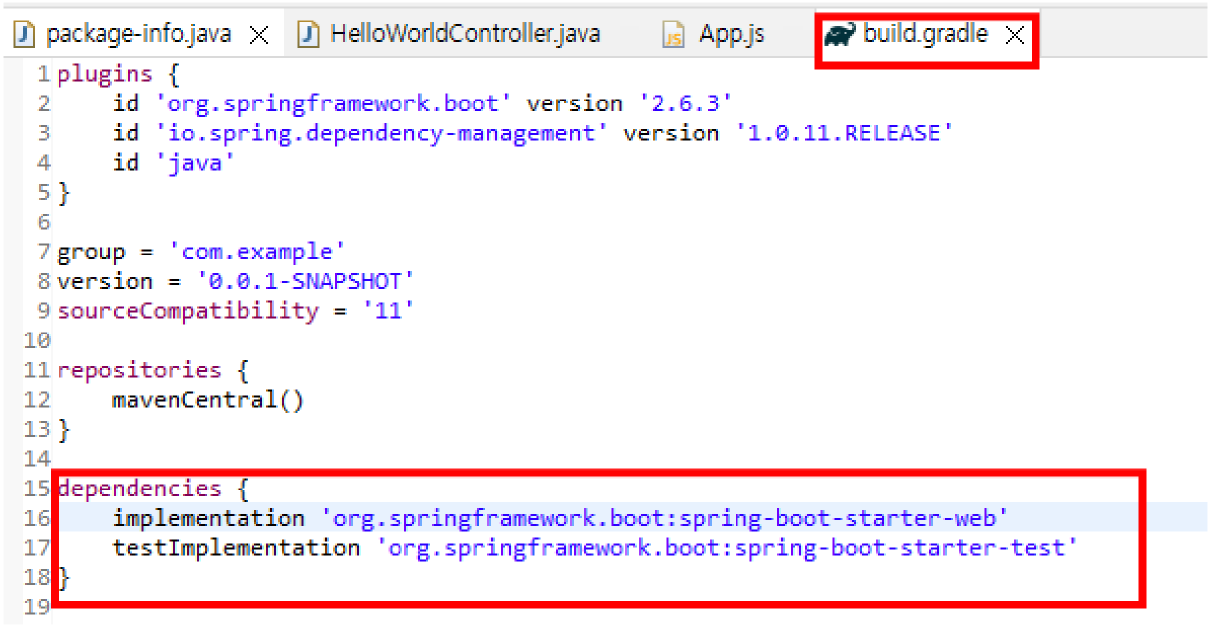


- `@ResponseBody` 를 사용

4.백엔드 스프링

4.1 class1

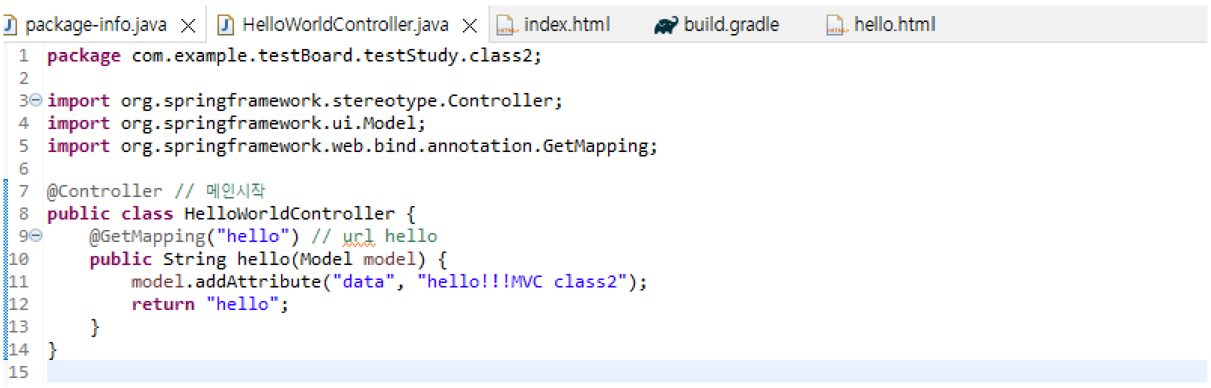
4.2 class2



package-info.java | HelloWorldController.java | App.js | build.gradle

```
plugins {
    id 'org.springframework.boot' version '2.6.3'
    id 'io.spring.dependency-management' version '1.0.11.RELEASE'
    id 'java'
}

group = 'com.example'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '11'

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-web'
    testImplementation 'org.springframework.boot:spring-boot-starter-test'
}
```

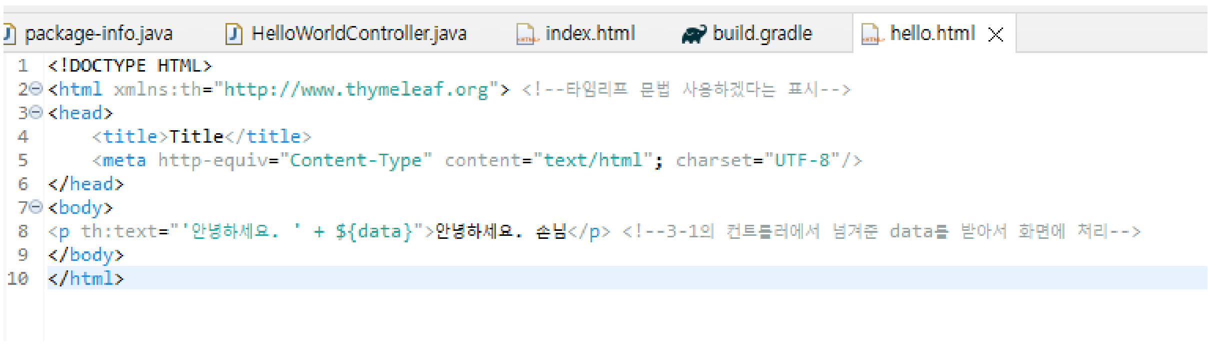
helloworldController



package-info.java | HelloWorldController.java | index.html | build.gradle | hello.html

```
package com.example.testBoard.testStudy.class2;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@Controller // 메인시작
public class HelloWorldController {
    @GetMapping("hello") // url hello
    public String hello(Model model) {
        model.addAttribute("data", "hello!!!MVC class2");
        return "hello";
    }
}
```

Templetes/hello.html



package-info.java | HelloWorldController.java | index.html | build.gradle | hello.html

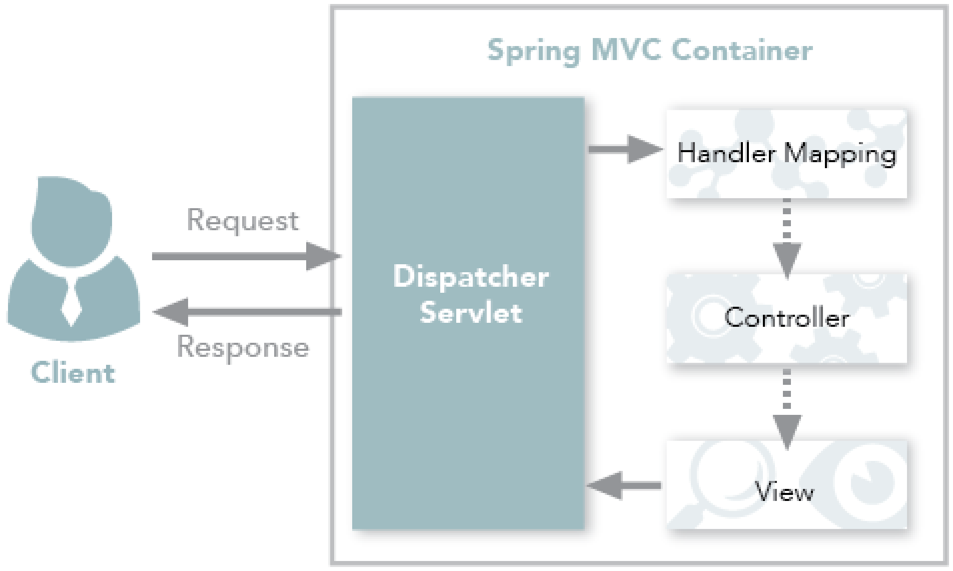
```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org"> <!--타임리프 문법 사용하겠다는 표시-->
<head>
    <title>Title</title>
    <meta http-equiv="Content-Type" content="text/html"; charset="UTF-8"/>
</head>
<body>
<p th:text="'안녕하세요. ' + ${data}">안녕하세요. 손님</p> <!--3-1의 컨트롤러에서 넘겨준 data를 받아서 화면에 처리-->
</body>
</html>
```

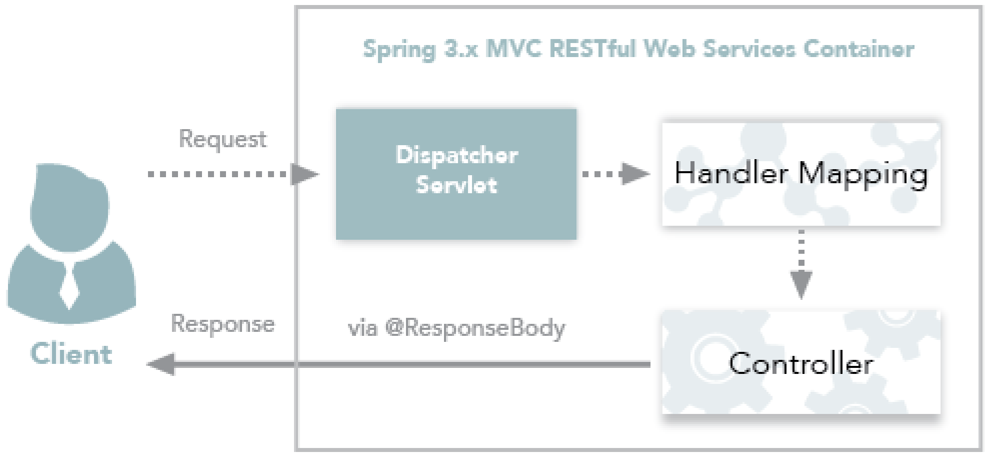
4.3 RestController 과 Controller 차이

@RestController 은 @Controller 와 @ResponseBody 의 조합입니다.

Spring 프레임 워크에서 RESTful 웹 서비스를 보다 쉽게 개발할 수 있도록 Spring 4.0 에서 추가되었습니다.

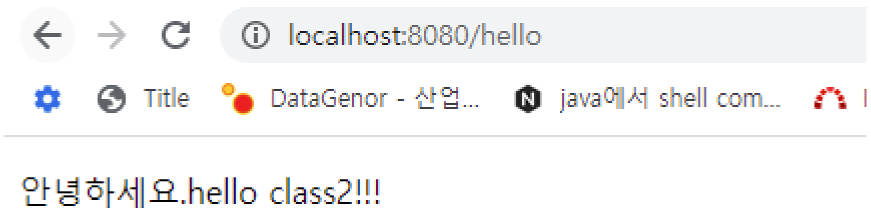
근본적인 차이점은 @Controller 의 역할은 Model 객체를 만들어 데이터를 담고 View 를 찾는 것이지만, @RestController 는 단순히 객체만을 반환하고 객체 데이터는 JSON 또는 XML 형식으로 HTTP 응답에 담아서 전송합니다. 물론 @Controller 와 @ResponseBody 를 사용하여 만들 수 있지만 이러한 방식은 RESTful 웹서비스의 기본 동작이기 때문에 Spring 은 @Controller 와 @ResponseBody 의 동작을 조합한 @RestController 을 도입했습니다.



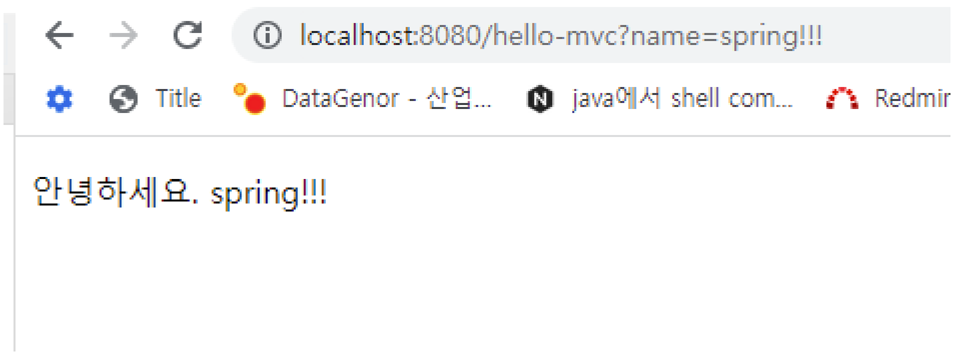


4-4. class2 – 소스참조

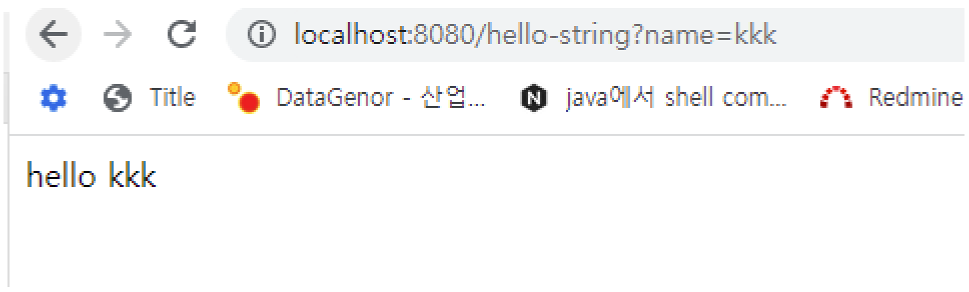
```
runtimeOnly('org.springframework.boot:spring-boot-devtools')
```



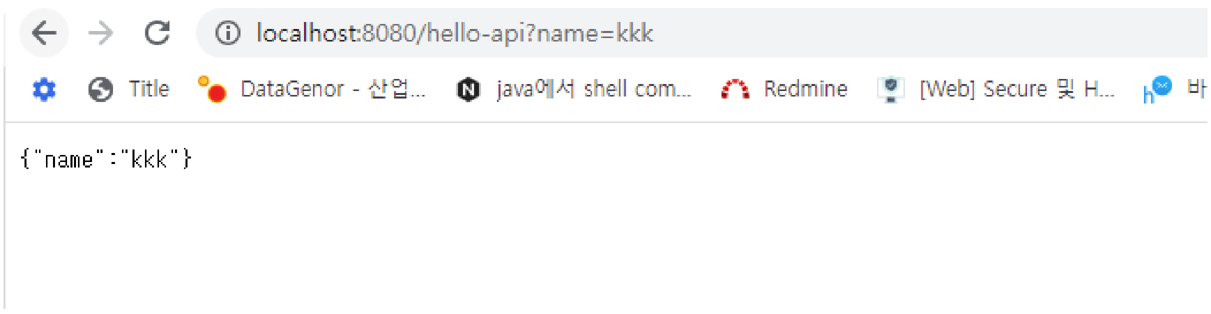
안녕하세요.hello class2!!!



안녕하세요. spring!!!

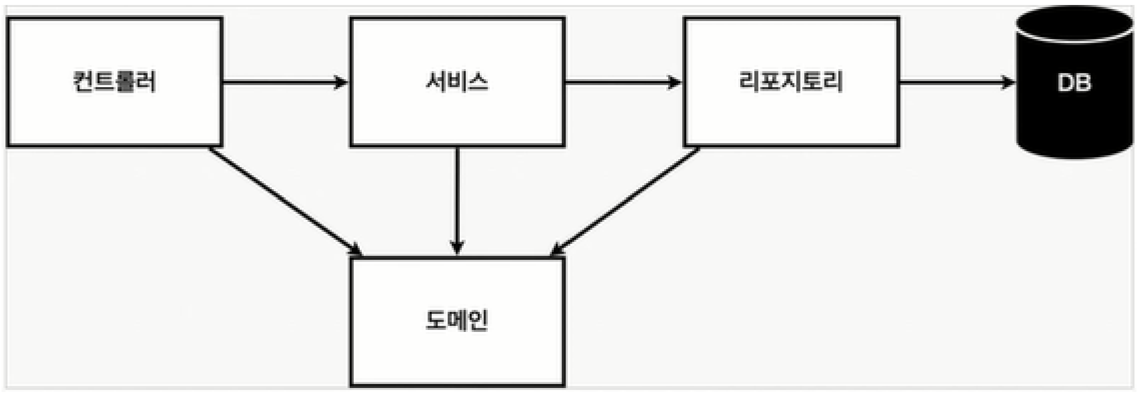


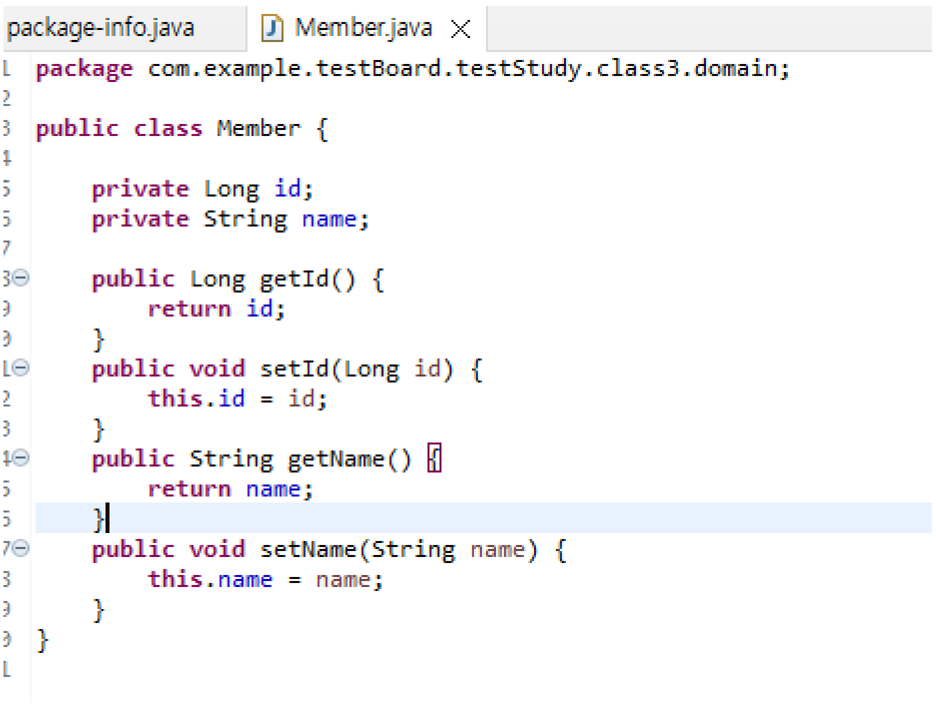
hello kkk



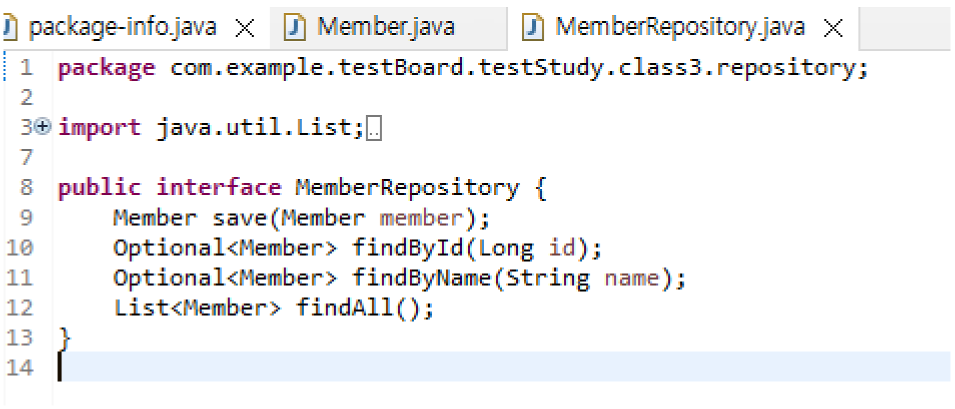
```
{"name":"kkk"}
```

4-5. 회원관리 예제 – 백앤드

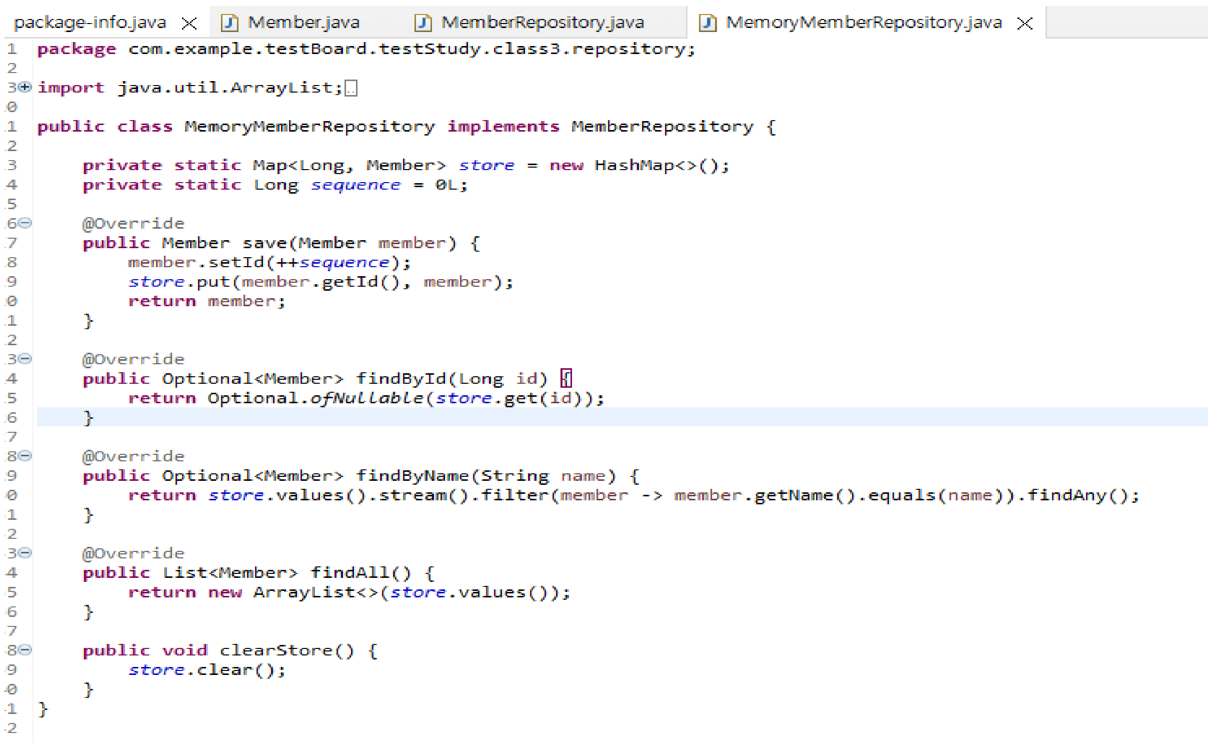




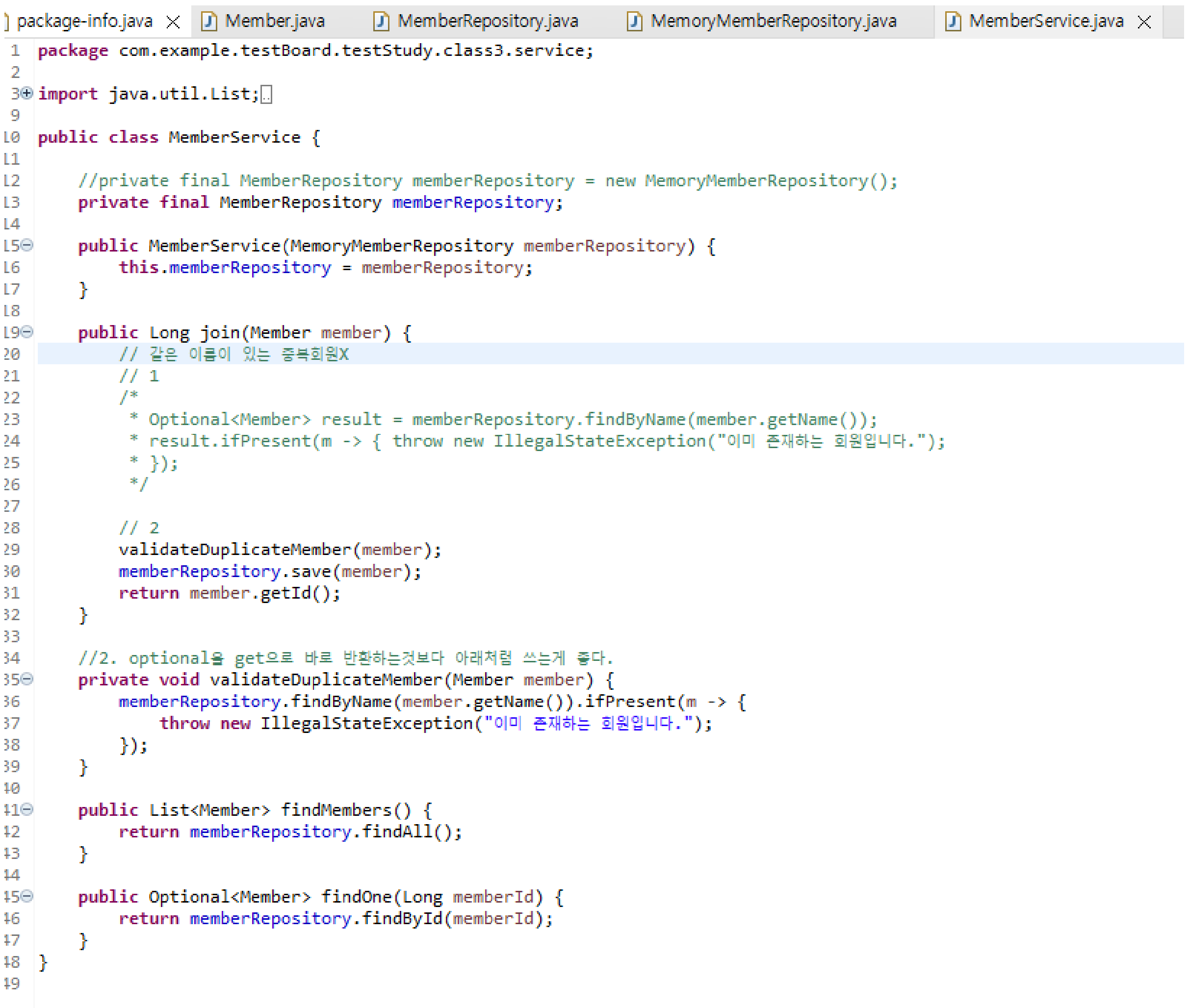
```java
package com.example.testBoard.testStudy.class3.domain;

public class Member {

    private Long id;
    private String name;

    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```



```java
package com.example.testBoard.testStudy.class3.repository;

import java.util.List;

public interface MemberRepository {
    Member save(Member member);
    Optional<Member> findById(Long id);
    Optional<Member> findByName(String name);
    List<Member> findAll();
}
```

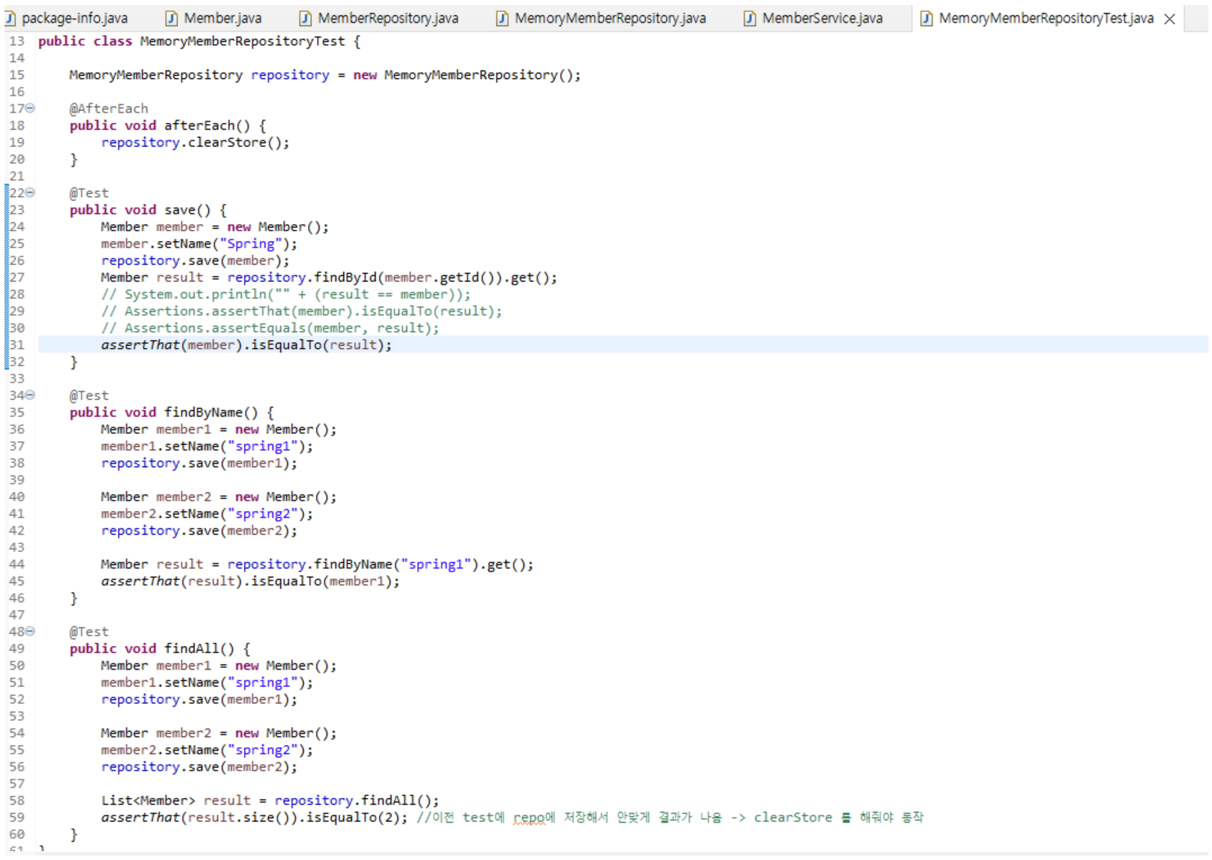

package-info.java   Member.java   MemberRepository.java   MemoryMemberRepository.java

```java
package com.example.testBoard.testStudy.class3.repository;

import java.util.ArrayList;

public class MemoryMemberRepository implements MemberRepository {

    private static Map<Long, Member> store = new HashMap<>();
    private static Long sequence = 0L;

    @Override
    public Member save(Member member) {
        member.setId(++sequence);
        store.put(member.getId(), member);
        return member;
    }

    @Override
    public Optional<Member> findById(Long id) {
        return Optional.ofNullable(store.get(id));
    }

    @Override
    public Optional<Member> findByName(String name) {
        return store.values().stream().filter(member -> member.getName().equals(name)).findAny();
    }

    @Override
    public List<Member> findAll() {
        return new ArrayList<>(store.values());
    }

    public void clearStore() {
        store.clear();
    }
}
```

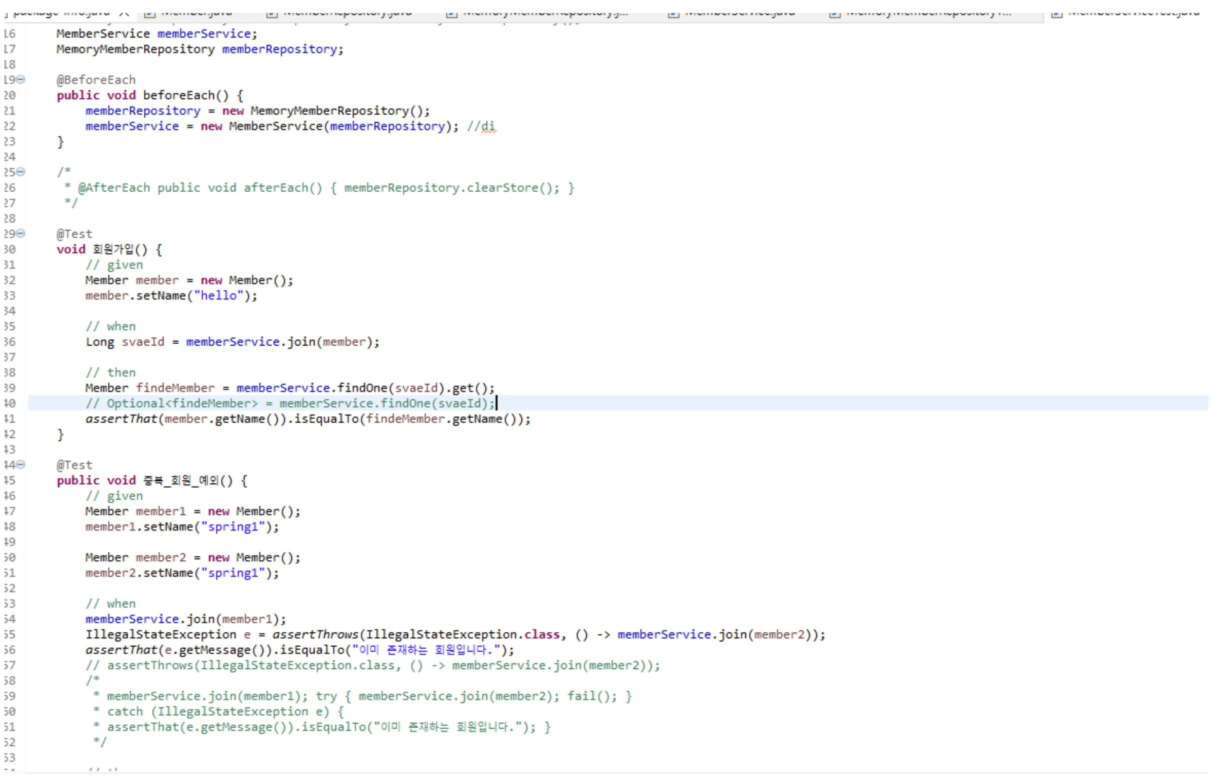

package-info.java   Member.java   MemberRepository.java   MemoryMemberRepository.java   MemberService.java

```java
package com.example.testBoard.testStudy.class3.service;

import java.util.List;

public class MemberService {

    //private final MemberRepository memberRepository = new MemoryMemberRepository();
    private final MemberRepository memberRepository;

    public MemberService(MemoryMemberRepository memberRepository) {
        this.memberRepository = memberRepository;
    }

    public Long join(Member member) {
        // 같은 이름이 있는 중복회원X
        // 1
        /*
         * Optional<Member> result = memberRepository.findByName(member.getName());
         * result.ifPresent(m -> { throw new IllegalStateException("이미 존재하는 회원입니다.");
         * });
         */

        // 2
        validateDuplicateMember(member);
        memberRepository.save(member);
        return member.getId();
    }

    //2. optional을 get으로 바로 반환하는것보다 아래처럼 쓰는게 좋다.
    private void validateDuplicateMember(Member member) {
        memberRepository.findByName(member.getName()).ifPresent(m -> {
            throw new IllegalStateException("이미 존재하는 회원입니다.");
        });
    }

    public List<Member> findMembers() {
        return memberRepository.findAll();
    }

    public Optional<Member> findOne(Long memberId) {
        return memberRepository.findById(memberId);
    }
}
```

4-6. junit

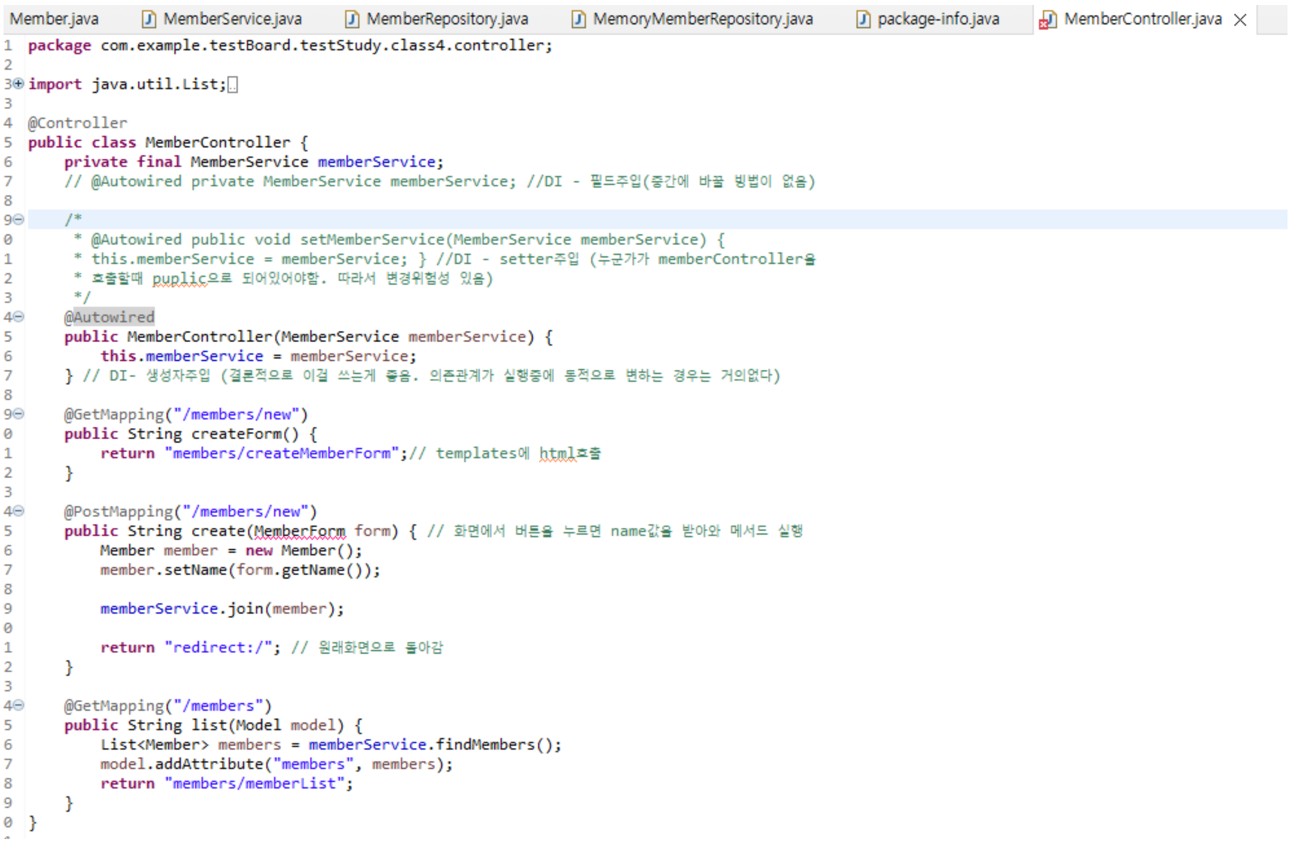


```java
public class MemoryMemberRepositoryTest {

    MemoryMemberRepository repository = new MemoryMemberRepository();

    @AfterEach
    public void afterEach() {
        repository.clearStore();
    }

    @Test
    public void save() {
        Member member = new Member();
        member.setName("Spring");
        repository.save(member);
        Member result = repository.findById(member.getId()).get();
        // System.out.println("" + (result == member));
        // Assertions.assertThat(member).isEqualTo(result);
        // Assertions.assertEquals(member, result);
        assertThat(member).isEqualTo(result);
    }

    @Test
    public void findByName() {
        Member member1 = new Member();
        member1.setName("spring1");
        repository.save(member1);

        Member member2 = new Member();
        member2.setName("spring2");
        repository.save(member2);

        Member result = repository.findByName("spring1").get();
        assertThat(result).isEqualTo(member1);
    }

    @Test
    public void findAll() {
        Member member1 = new Member();
        member1.setName("spring1");
        repository.save(member1);

        Member member2 = new Member();
        member2.setName("spring2");
        repository.save(member2);

        List<Member> result = repository.findAll();
        assertThat(result.size()).isEqualTo(2); //이전 test에 repo에 저장해서 안맞게 결과가 나음 -> clearStore 를 해줘야 동작
    }
```



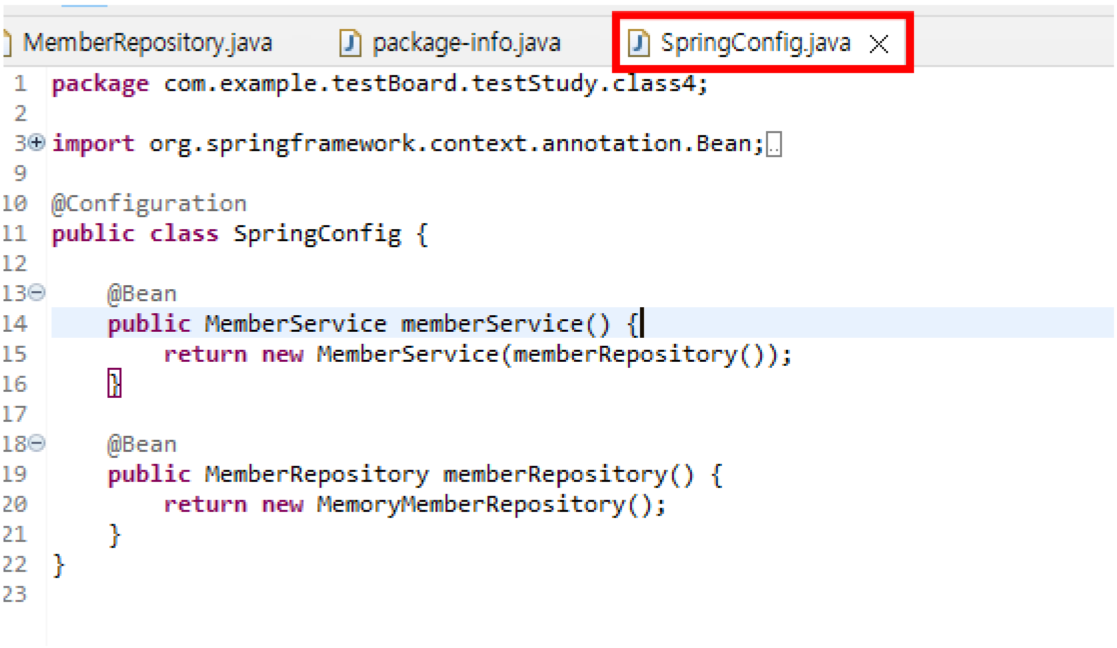
```java
    MemberService memberService;
    MemoryMemberRepository memberRepository;

    @BeforeEach
    public void beforeEach() {
        memberRepository = new MemoryMemberRepository();
        memberService = new MemberService(memberRepository); //di
    }

    /*
     * @AfterEach public void afterEach() { memberRepository.clearStore(); }
     */

    @Test
    void 회원가입() {
        // given
        Member member = new Member();
        member.setName("hello");

        // when
        Long svaeId = memberService.join(member);

        // then
        Member findeMember = memberService.findOne(svaeId).get();
        // Optional<findeMember> = memberService.findOne(svaeId);
        assertThat(member.getName()).isEqualTo(findeMember.getName());
    }

    @Test
    public void 중복_회원_예외() {
        // given
        Member member1 = new Member();
        member1.setName("spring1");

        Member member2 = new Member();
        member2.setName("spring1");

        // when
        memberService.join(member1);
        IllegalStateException e = assertThrows(IllegalStateException.class, () -> memberService.join(member2));
        assertThat(e.getMessage()).isEqualTo("이미 존재하는 회원입니다.");
        // assertThrows(IllegalStateException.class, () -> memberService.join(member2));
        /*
         * memberService.join(member1); try { memberService.join(member2); fail(); }
         * catch (IllegalStateException e) {
         * assertThat(e.getMessage()).isEqualTo("이미 존재하는 회원입니다."); }
         */
```

4-7. 스프링빈 의존성설정



```java
package com.example.testBoard.testStudy.class4.controller;

import java.util.List;

@Controller
public class MemberController {
    private final MemberService memberService;
    // @Autowired private MemberService memberService; //DI - 필드주입(중간에 바꿀 방법이 없음)

    /*
     * @Autowired public void setMemberService(MemberService memberService) {
     * this.memberService = memberService; } //DI - setter주입 (누군가가 memberController을
     * 호출할때 puplic으로 되어있어야함. 따라서 변경위험성 있음)
     */
    @Autowired
    public MemberController(MemberService memberService) {
        this.memberService = memberService;
    } // DI- 생성자주입 (결론적으로 이걸 쓰는게 좋음. 의존관계가 실행중에 동적으로 변하는 경우는 거의없다)

    @GetMapping("/members/new")
    public String createForm() {
        return "members/createMemberForm";// templates에 html호출
    }

    @PostMapping("/members/new")
    public String create(MemberForm form) { // 화면에서 버튼을 누르면 name값을 받아와 메서드 실행
        Member member = new Member();
        member.setName(form.getName());

        memberService.join(member);

        return "redirect:/"; // 원래화면으로 돌아감
    }

    @GetMapping("/members")
    public String list(Model model) {
        List<Member> members = memberService.findMembers();
        model.addAttribute("members", members);
        return "members/memberList";
    }
}
```

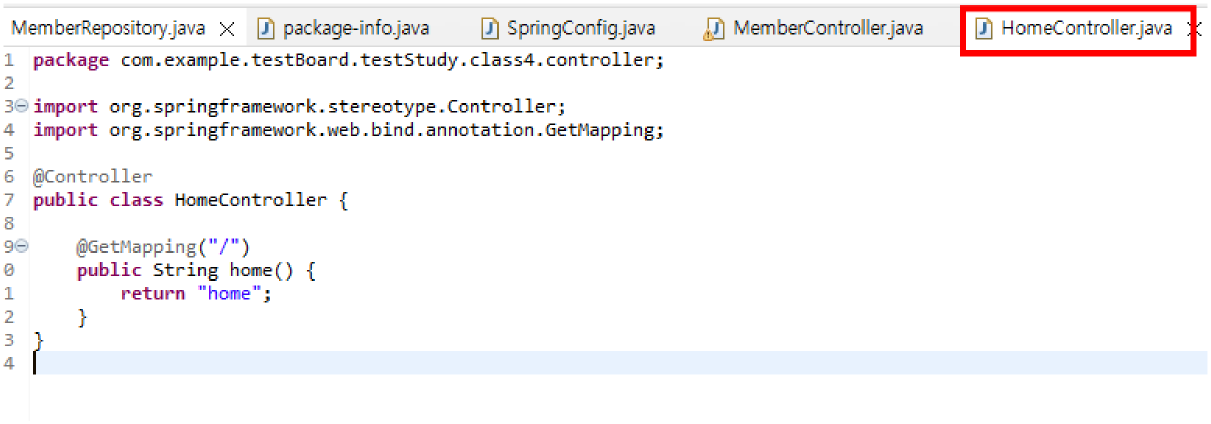
4-8. 환경설정



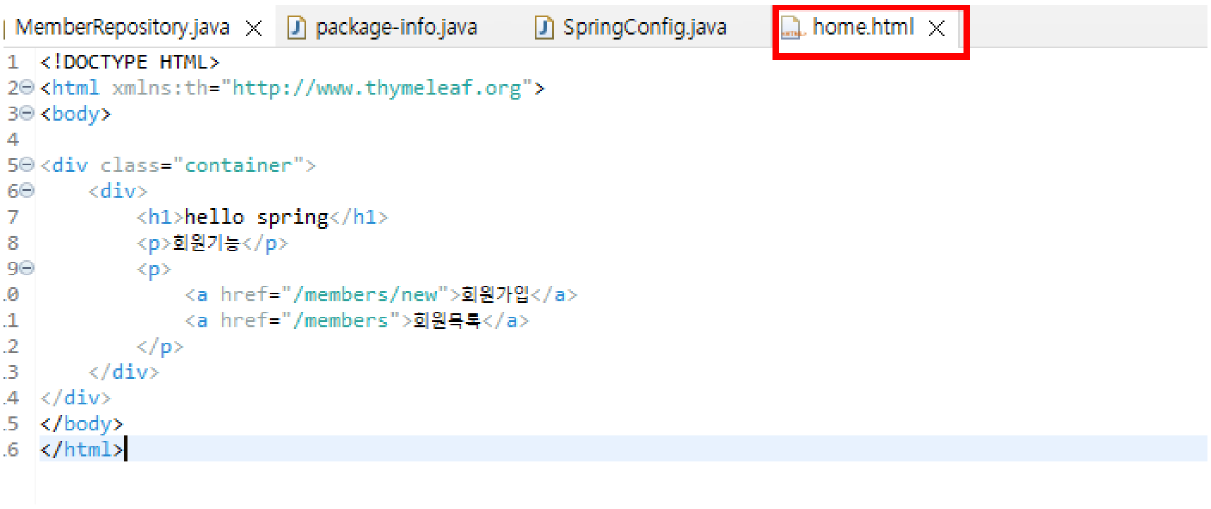
```java
package com.example.testBoard.testStudy.class4;

import org.springframework.context.annotation.Bean;

@Configuration
public class SpringConfig {

    @Bean
    public MemberService memberService() {
        return new MemberService(memberRepository());
    }

    @Bean
    public MemberRepository memberRepository() {
        return new MemoryMemberRepository();
    }
}
```

4-9. DI

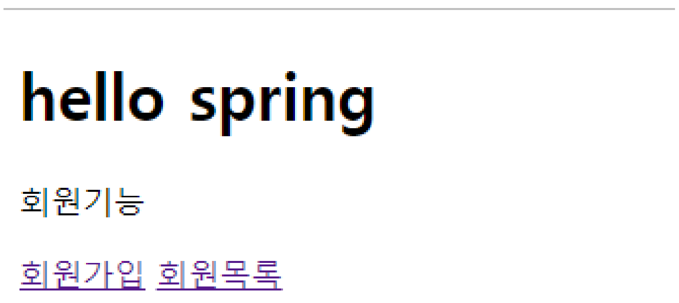
```
@Controller
public class MemberController {
    private final MemberService memberService;
    //@Autowired private MemberService memberService; //DI - 필드주입(중간에 바꿀 빙법이 없음)

/*    @Autowired
    public void setMemberService(MemberService memberService) {
        this.memberService = memberService;
    }*/ //DI - setter주입 (누군가가 memberController를 호출할때 puplic으로 되어있어야함. 따라서 변경위험성 있음)

    @Autowired
    public MemberController(MemberService memberService) {
        this.memberService = memberService;
    } //DI- 생성자주입 (결론적으로 이걸 쓰는게 좋음. 의존관계가 실행중에 동적으로 변하는 경우는 거의없다)
}
```



```java
package com.example.testBoard.testStudy.class4.controller;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class HomeController {

    @GetMapping("/")
    public String home() {
        return "home";
    }
}
```



```html
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<body>

<div class="container">
    <div>
        <h1>hello spring</h1>
        <p>회원기능</p>
        <p>
            <a href="/members/new">회원가입</a>
            <a href="/members">회원목록</a>
        </p>
    </div>
</div>
</body>
</html>
```

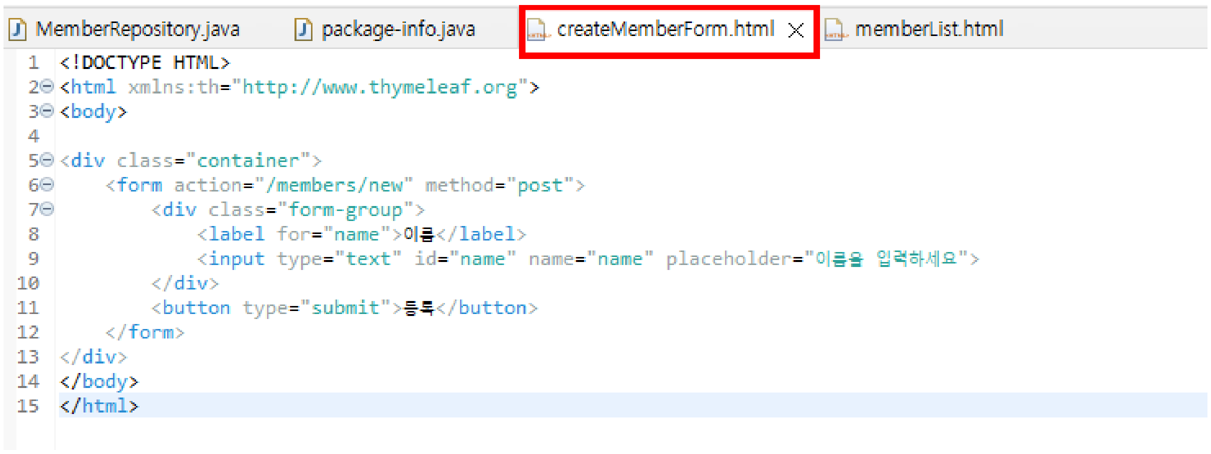


hello spring

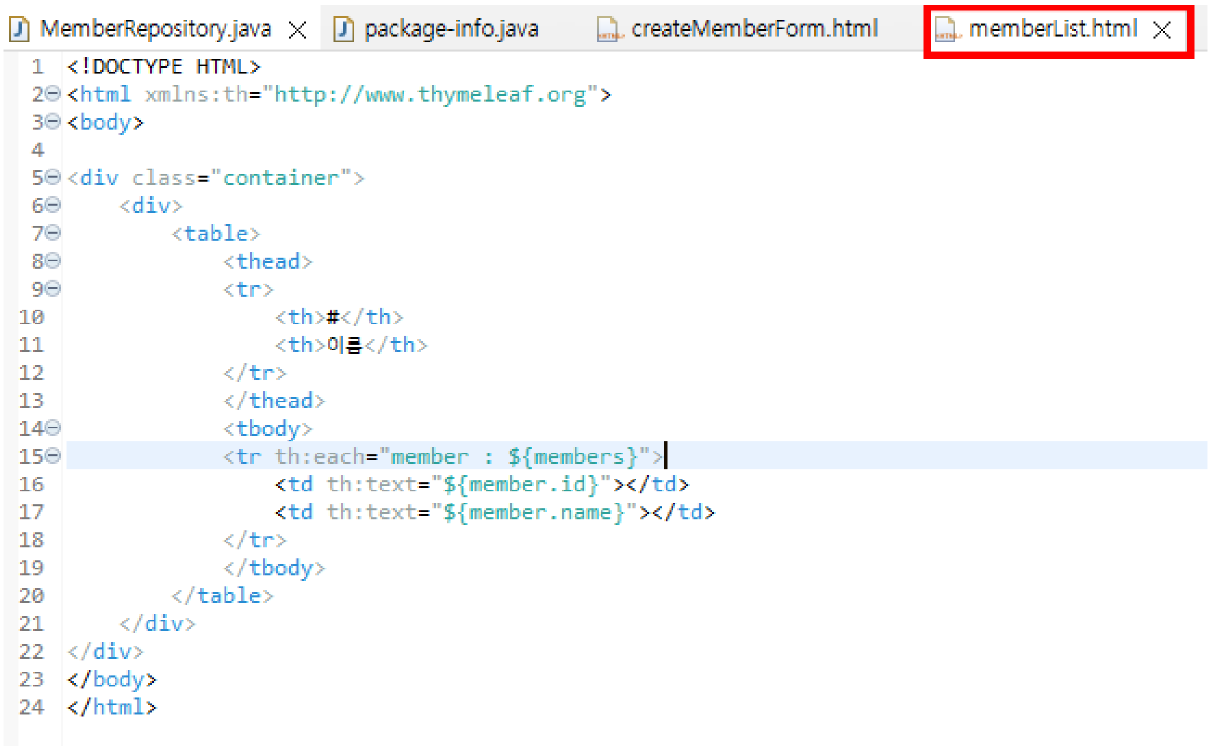
회원기능

회원가입 회원목록

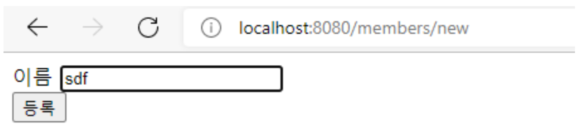
4-10. 메모리 기본 cr



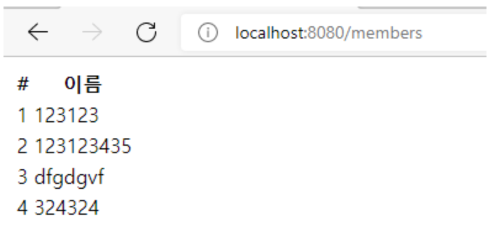
```html
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<body>

<div class="container">
    <form action="/members/new" method="post">
        <div class="form-group">
            <label for="name">이름</label>
            <input type="text" id="name" name="name" placeholder="이름을 입력하세요">
        </div>
        <button type="submit">등록</button>
    </form>
</div>
</body>
</html>
```



```html
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<body>

<div class="container">
    <div>
        <table>
            <thead>
            <tr>
                <th>#</th>
                <th>이름</th>
            </tr>
            </thead>
            <tbody>
            <tr th:each="member : ${members}">
                <td th:text="${member.id}"></td>
                <td th:text="${member.name}"></td>
            </tr>
            </tbody>
        </table>
    </div>
</div>
</body>
</html>
```


localhost:8080/members/new

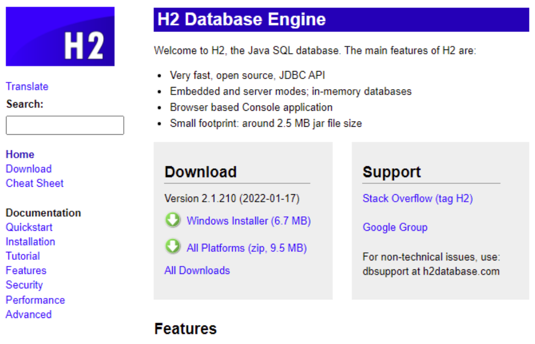
이름 sdf


localhost:8080/members

```
#   이름
1 123123
2 123123435
3 dfgdgvf
4 324324
```

4-11. 디비 연동 – h2 db

https://www.h2database.com



H2 Database Engine

Welcome to H2, the Java SQL database. The main features of H2 are:

- Very fast, open source, JDBC API
- Embedded and server modes; in-memory databases
- Browser based Console application
- Small footprint: around 2.5 MB jar file size

Download

Version 2.1.210 (2022-01-17)

Windows Installer (6.7 MB)

All Platforms (zip, 9.5 MB)

All Downloads

Support

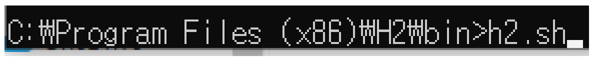
Stack Overflow (tag H2)

Google Group

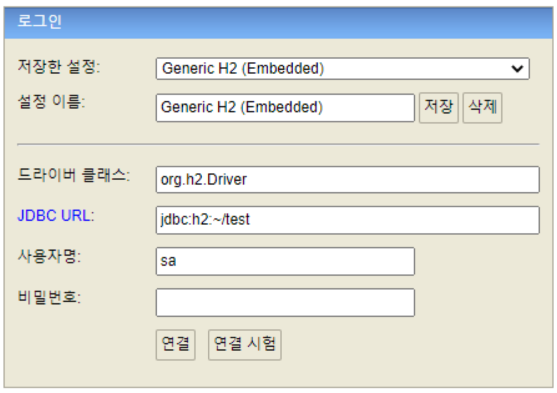
For non-technical issues, use:
dbsupport at h2database.com

Features

/

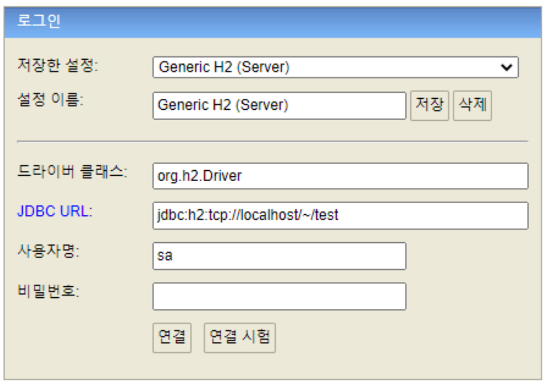


```
C:₩Program Files (x86)₩H2₩bin>h2.sh
```



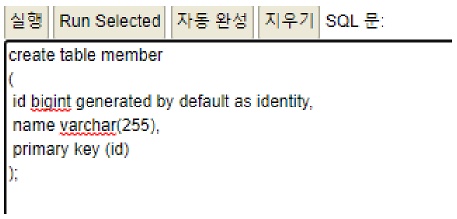
로그인

저장한 설정:	Generic H2 (Embedded)
설정 이름:	Generic H2 (Embedded)
드라이버 클래스:	org.h2.Driver
JDBC URL:	jdbc:h2:~/test
사용자명:	sa
비밀번호:	



로그인

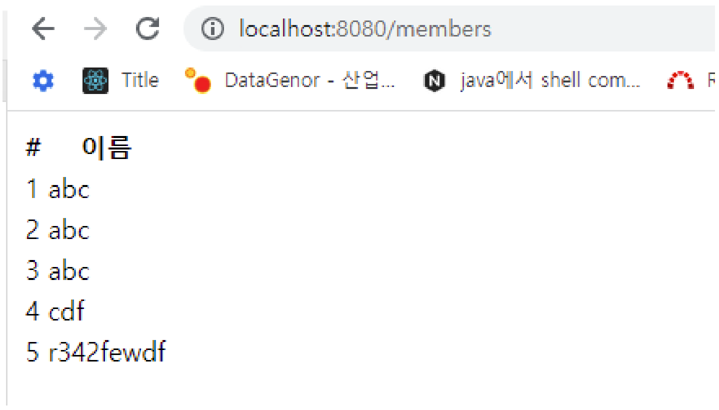
저장한 설정:	Generic H2 (Server)
설정 이름:	Generic H2 (Server)
드라이버 클래스:	org.h2.Driver
JDBC URL:	jdbc:h2:tcp://localhost/~/test
사용자명:	sa
비밀번호:	



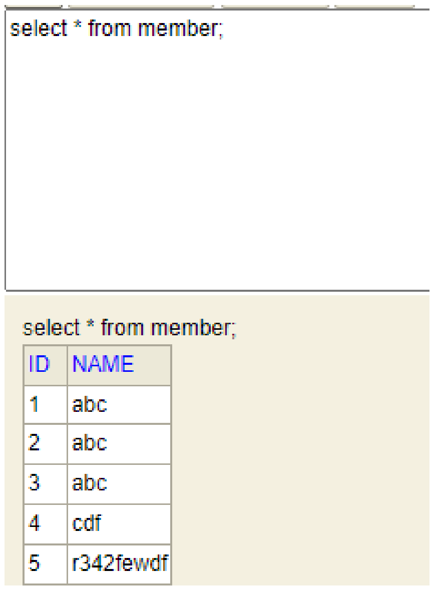
```
create table member
(
 id bigint generated by default as identity,
 name varchar(255),
 primary key (id)
);
```

```
1-2. build.gradle
    implementation 'org.springframework.boot:spring-boot-starter-jdbc'
    runtimeOnly 'com.h2database:h2'
1-3. application.properties
    spring.datasource.url=jdbc:h2:tcp://localhost/~/test
    spring.datasource.driver-class-name=org.h2.Driver
```

```
dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
    implementation 'org.springframework.boot:spring-boot-starter-web'
    implementation 'org.springframework.boot:spring-boot-starter-jdbc'

    runtimeOnly('org.springframework.boot:spring-boot-devtools')
    runtimeOnly 'com.h2database:h2'

    testImplementation('org.springframework.boot:spring-boot-starter-test') {
        exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
    }
}
```



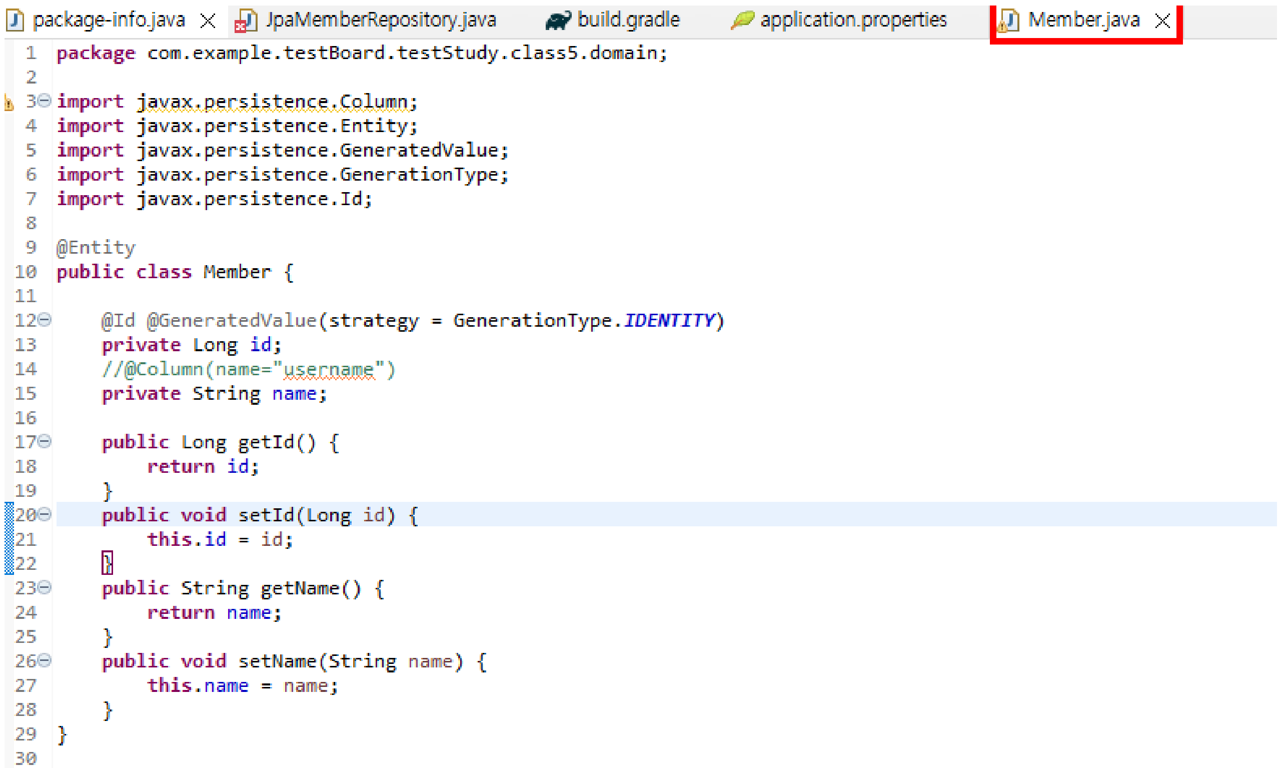
```
#   이름
1 abc
2 abc
3 abc
4 cdf
5 r342fewdf
```



```
select * from member;
```

select * from member;

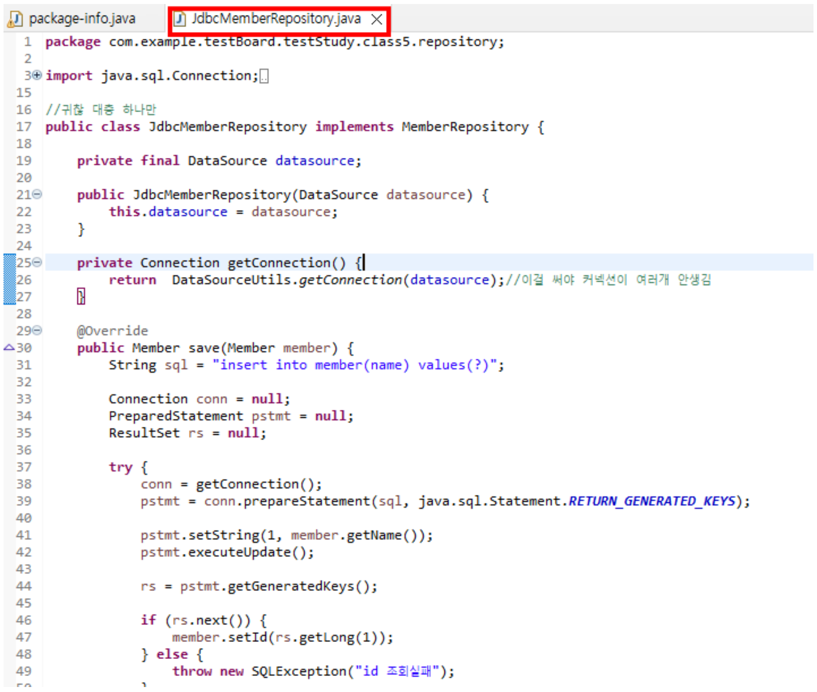
ID	NAME
1	abc
2	abc
3	abc
4	cdf
5	r342fewdf



```java
package com.example.testBoard.testStudy.class5.domain;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Member {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    //@Column(name="username")
    private String name;

    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}

```

4-12. Repository교체

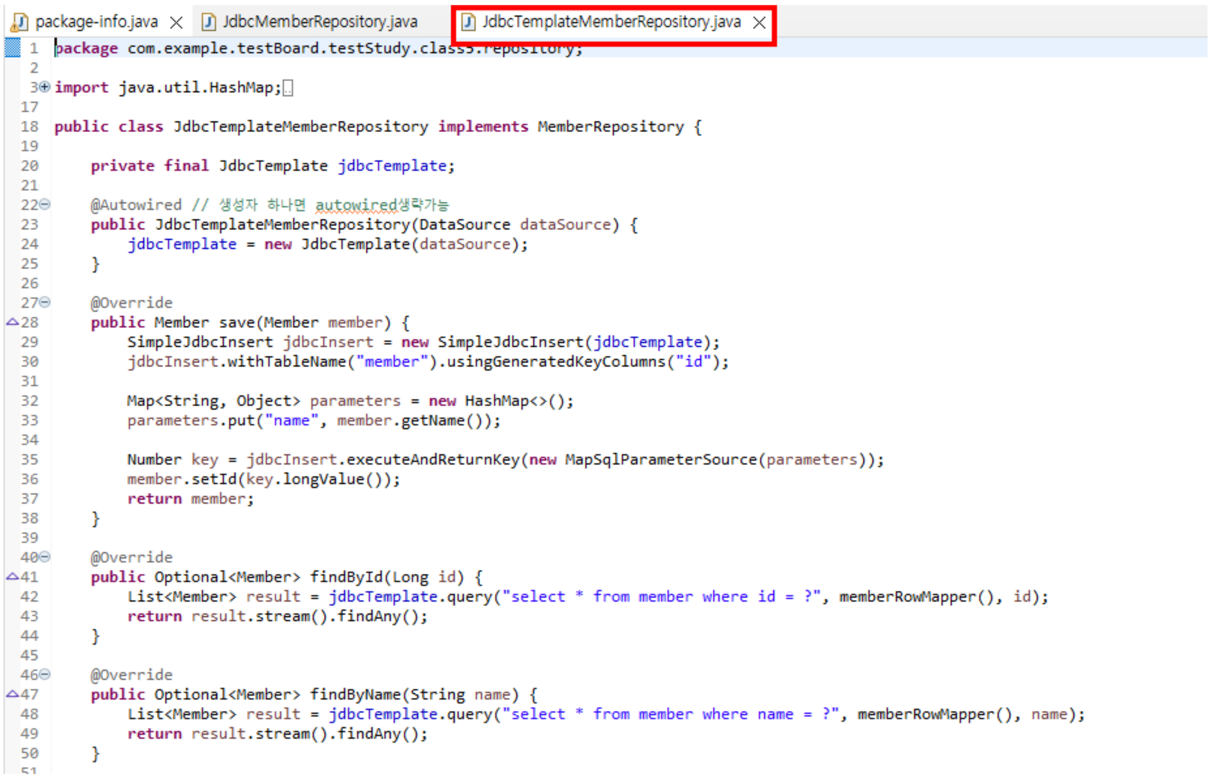
4-12-1. Jdbc



package-info.java | JdbcMemberRepository.java

```java
package com.example.testBoard.testStudy.class5.repository;

import java.sql.Connection;

//귀찮 대충 하나만
public class JdbcMemberRepository implements MemberRepository {

    private final DataSource datasource;

    public JdbcMemberRepository(DataSource datasource) {
        this.datasource = datasource;
    }

    private Connection getConnection() {
        return  DataSourceUtils.getConnection(datasource);//이걸 써야 커넥션이 여러개 안생김
    }

    @Override
    public Member save(Member member) {
        String sql = "insert into member(name) values(?)";

        Connection conn = null;
        PreparedStatement pstmt = null;
        ResultSet rs = null;

        try {
            conn = getConnection();
            pstmt = conn.prepareStatement(sql, java.sql.Statement.RETURN_GENERATED_KEYS);

            pstmt.setString(1, member.getName());
            pstmt.executeUpdate();

            rs = pstmt.getGeneratedKeys();

            if (rs.next()) {
                member.setId(rs.getLong(1));
            } else {
                throw new SQLException("id 조회실패");
```

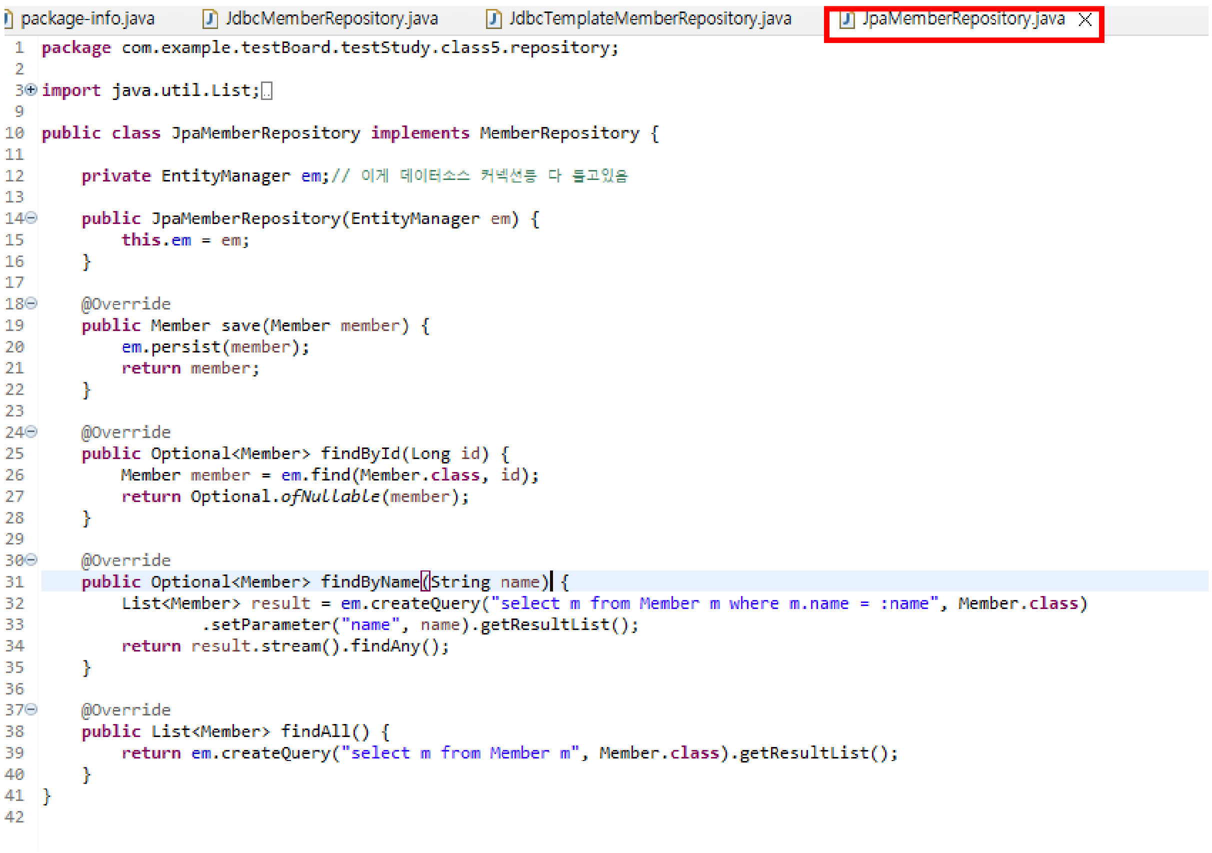
4-12-2. JdbcTemplate



package-info.java | JdbcMemberRepository.java | JdbcTemplateMemberRepository.java

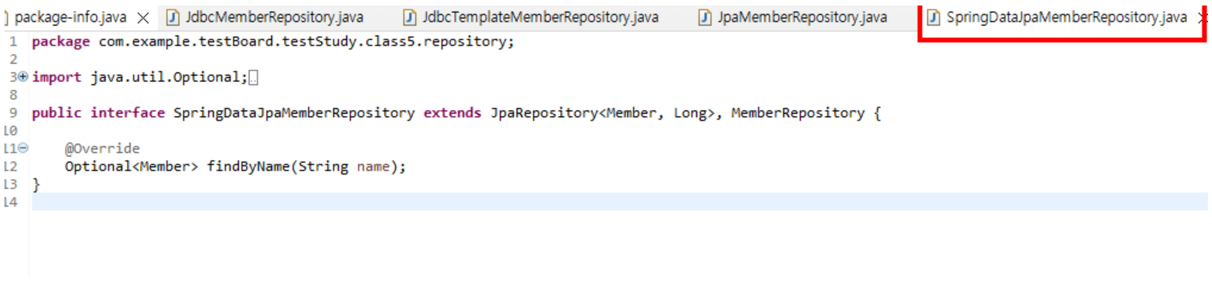
```java
package com.example.testBoard.testStudy.class5.repository;

import java.util.HashMap;

public class JdbcTemplateMemberRepository implements MemberRepository {

    private final JdbcTemplate jdbcTemplate;

    @Autowired // 생성자 하나면 autowired생략가능
    public JdbcTemplateMemberRepository(DataSource dataSource) {
        jdbcTemplate = new JdbcTemplate(dataSource);
    }

    @Override
    public Member save(Member member) {
        SimpleJdbcInsert jdbcInsert = new SimpleJdbcInsert(jdbcTemplate);
        jdbcInsert.withTableName("member").usingGeneratedKeyColumns("id");

        Map<String, Object> parameters = new HashMap<>();
        parameters.put("name", member.getName());

        Number key = jdbcInsert.executeAndReturnKey(new MapSqlParameterSource(parameters));
        member.setId(key.longValue());
        return member;
    }

    @Override
    public Optional<Member> findById(Long id) {
        List<Member> result = jdbcTemplate.query("select * from member where id = ?", memberRowMapper(), id);
        return result.stream().findAny();
    }

    @Override
    public Optional<Member> findByName(String name) {
        List<Member> result = jdbcTemplate.query("select * from member where name = ?", memberRowMapper(), name);
        return result.stream().findAny();
    }
```

4-12-3. Jpa



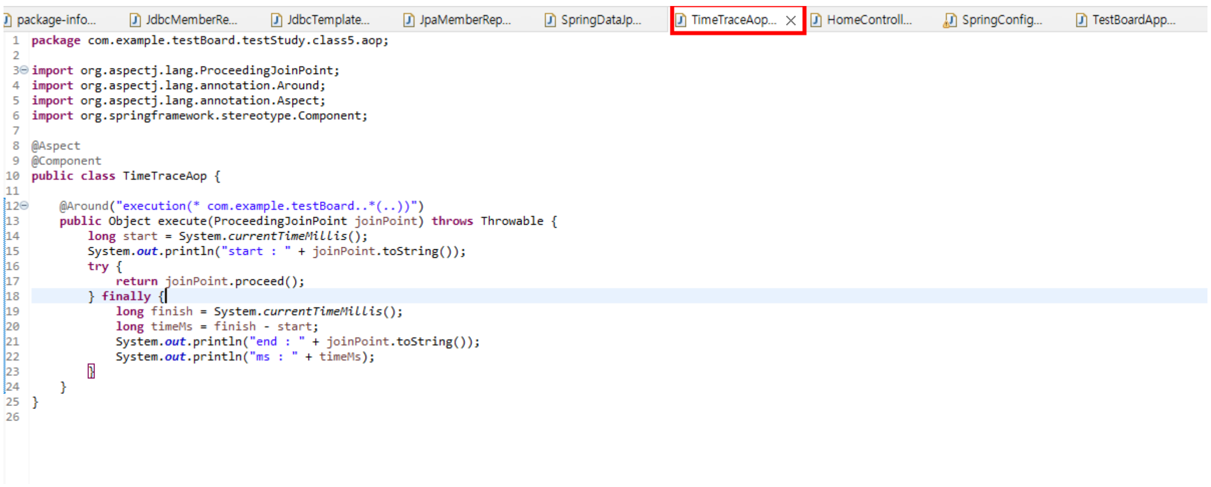
```java
package com.example.testBoard.testStudy.class5.repository;

import java.util.List;

public class JpaMemberRepository implements MemberRepository {

    private EntityManager em;// 이게 데이터소스 커넥션등 다 들고있음

    public JpaMemberRepository(EntityManager em) {
        this.em = em;
    }

    @Override
    public Member save(Member member) {
        em.persist(member);
        return member;
    }

    @Override
    public Optional<Member> findById(Long id) {
        Member member = em.find(Member.class, id);
        return Optional.ofNullable(member);
    }

    @Override
    public Optional<Member> findByName(String name) {
        List<Member> result = em.createQuery("select m from Member m where m.name = :name", Member.class)
                .setParameter("name", name).getResultList();
        return result.stream().findAny();
    }

    @Override
    public List<Member> findAll() {
        return em.createQuery("select m from Member m", Member.class).getResultList();
    }
}
```

4-12-4. SpringDataJpa

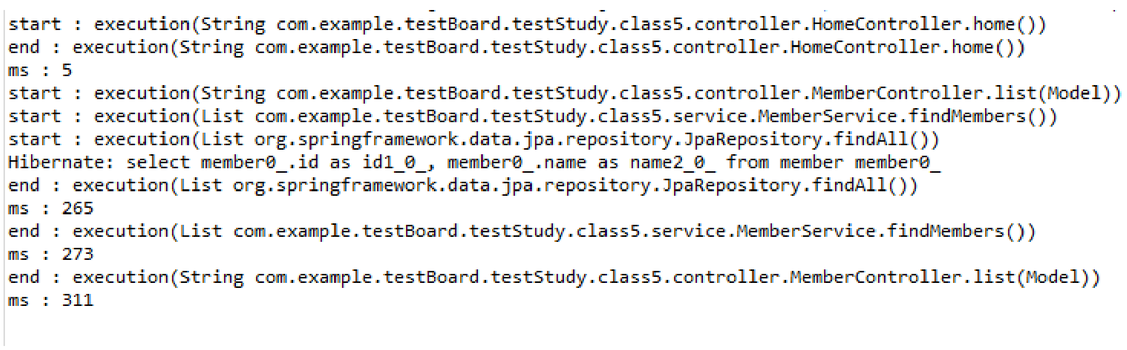

package-info.java
JdbcMemberRepository.java
JdbcTemplateMemberRepository.java
JpaMemberRepository.java
SpringDataJpaMemberRepository.java

```java
package com.example.testBoard.testStudy.class5.repository;

import java.util.Optional;

public interface SpringDataJpaMemberRepository extends JpaRepository<Member, Long>, MemberRepository {

    @Override
    Optional<Member> findByName(String name);
}
```

4-13. Aop

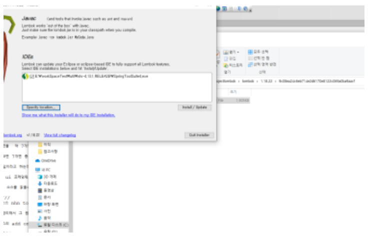


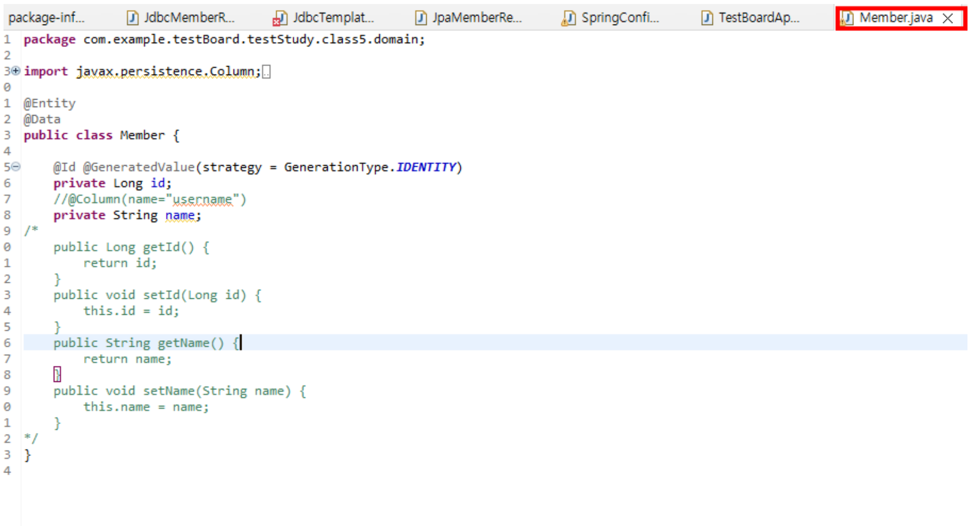
```java
package com.example.testBoard.testStudy.class5.aop;

import org.aspectj.lang.ProceedingJoinPoint;
import org.aspectj.lang.annotation.Around;
import org.aspectj.lang.annotation.Aspect;
import org.springframework.stereotype.Component;

@Aspect
@Component
public class TimeTraceAop {

    @Around("execution(* com.example.testBoard..*(..))")
    public Object execute(ProceedingJoinPoint joinPoint) throws Throwable {
        long start = System.currentTimeMillis();
        System.out.println("start : " + joinPoint.toString());
        try {
            return joinPoint.proceed();
        } finally {
            long finish = System.currentTimeMillis();
            long timeMs = finish - start;
            System.out.println("end : " + joinPoint.toString());
            System.out.println("ms : " + timeMs);
        }
    }
}
```



```
start : execution(String com.example.testBoard.testStudy.class5.controller.HomeController.home())
end : execution(String com.example.testBoard.testStudy.class5.controller.HomeController.home())
ms : 5
start : execution(String com.example.testBoard.testStudy.class5.controller.MemberController.list(Model))
start : execution(List com.example.testBoard.testStudy.class5.service.MemberService.findMembers())
start : execution(List org.springframework.data.jpa.repository.JpaRepository.findAll())
Hibernate: select member0_.id as id1_0_, member0_.name as name2_0_ from member member0_
end : execution(List org.springframework.data.jpa.repository.JpaRepository.findAll())
ms : 265
end : execution(List com.example.testBoard.testStudy.class5.service.MemberService.findMembers())
ms : 273
end : execution(String com.example.testBoard.testStudy.class5.controller.MemberController.list(Model))
ms : 311
```

4-14. Lombok





```java
package com.example.testBoard.testStudy.class5.domain;

import javax.persistence.Column;

@Entity
@Data
public class Member {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    //@Column(name="username")
    private String name;
/*
    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
*/
}
```

5. React

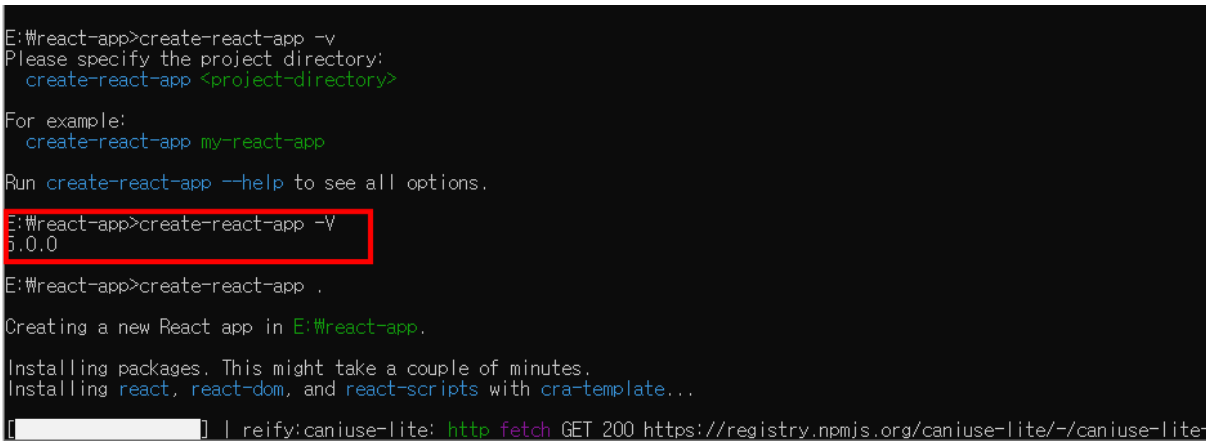
5-1. Creat react app 실행

https://ko.reactjs.org/docs/create-a-new-react-app.html#create-react-app

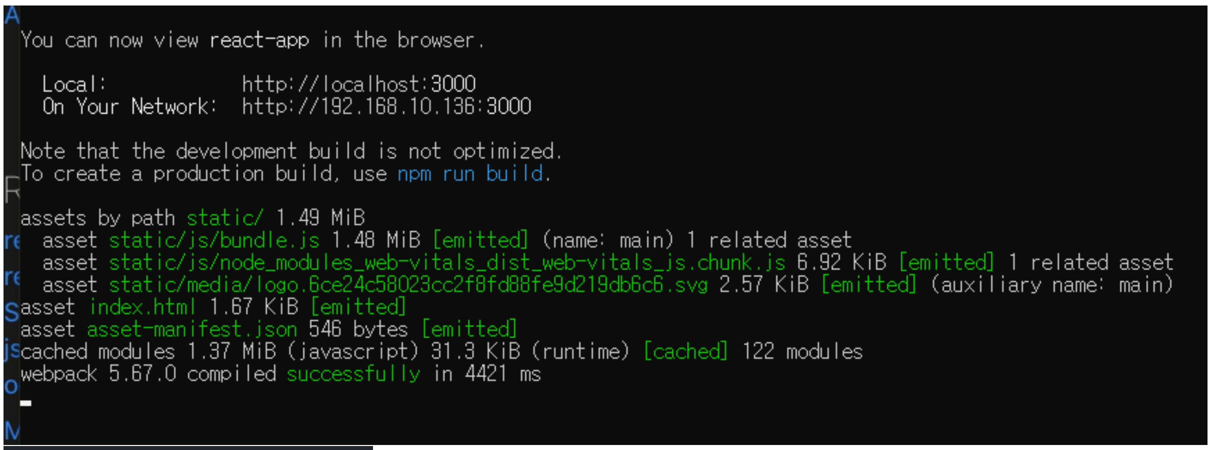
npm install -g create-react-app (어디서든 사용하게)

sudo npm install -g create-react-app (권한이 없어서 에러가 뜰때) - 비번

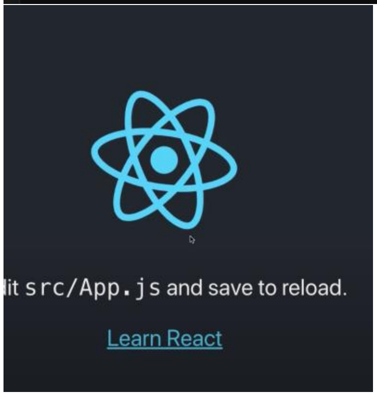
create-react-app -V



```
E:₩react-app>create-react-app -v
Please specify the project directory:
  create-react-app <project-directory>

For example:
  create-react-app my-react-app

Run create-react-app --help to see all options.

E:₩react-app>create-react-app -V
5.0.0

E:₩react-app>create-react-app .

Creating a new React app in E:₩react-app.

 Installing packages. This might take a couple of minutes.
 Installing react, react-dom, and react-scripts with cra-template...

[                   ] | reify:caniuse-lite: http fetch GET 200 https://registry.npmjs.org/caniuse-lite/-/caniuse-lite
```



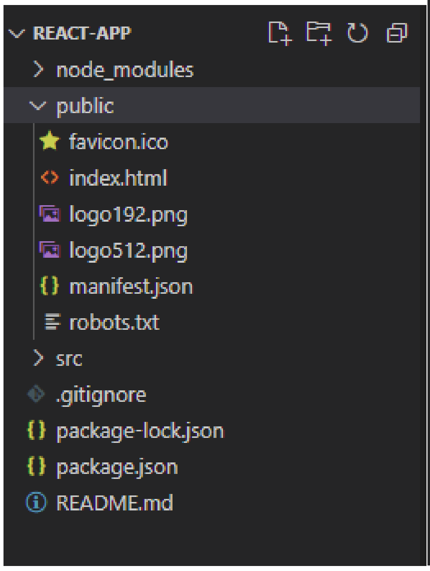
```
 You can now view react-app in the browser.

   Local:            http://localhost:3000
   On Your Network:  http://192.168.10.136:3000

 Note that the development build is not optimized.
 To create a production build, use npm run build.

 assets by path static/ 1.49 MiB
   asset static/js/bundle.js 1.48 MiB [emitted] (name: main) 1 related asset
   asset static/js/node_modules_web-vitals_dist_web-vitals_js.chunk.js 6.92 KiB [emitted] 1 related asset
   asset static/media/logo.6ce24c58023cc2f8fd88fe9d219db6c6.svg 2.57 KiB [emitted] (auxiliary name: main)
 asset index.html 1.67 KiB [emitted]
 asset asset-manifest.json 546 bytes [emitted]
 cached modules 1.37 MiB (javascript) 31.3 KiB (runtime) [cached] 122 modules
 webpack 5.67.0 compiled successfully in 4421 ms
```



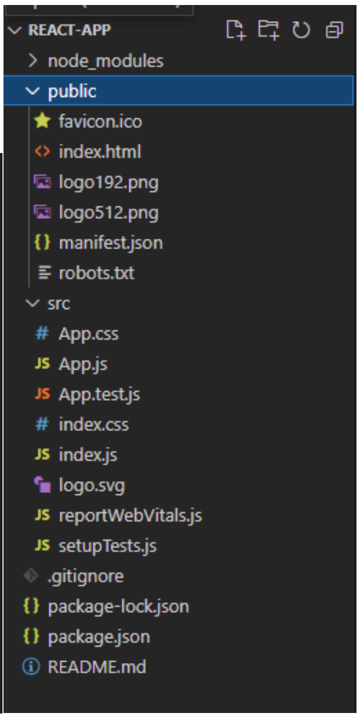
lit src/App.js and save to reload.

Learn React

5-2. 프로젝트 구조



```
REACT-APP
> node_modules
v public
    favicon.ico
    index.html
    logo192.png
    logo512.png
    manifest.json
    robots.txt
> src
  .gitignore
  package-lock.json
  package.json
  README.md
```



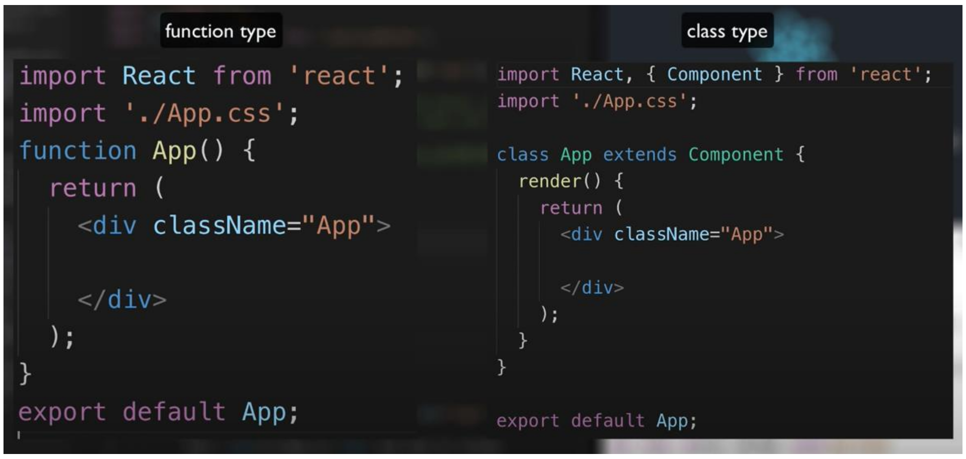
```
REACT-APP
> node_modules
v public
    favicon.ico
    index.html
    logo192.png
    logo512.png
    manifest.json
    robots.txt
v src
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
```

package.json: 스프링에서의 build.gradle 과 같은 기능

gitignore: 여기에 적힌 파일들은 깃허브에 올리지 않는다.,

readme.md: 프로젝트 설명 (깃허브 리드미)이다.

5-2-1. 리액트 소스 타입



function type

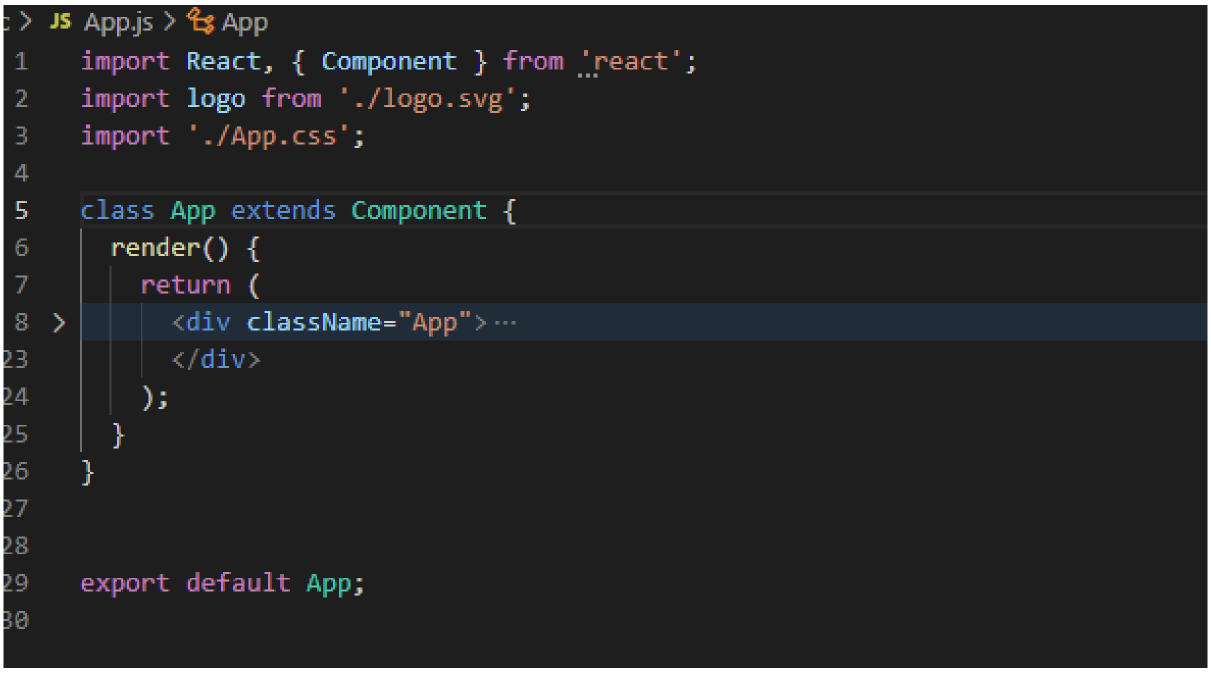
```
import React from 'react';
import './App.css';
function App() {
  return (
    <div className="App">

    </div>
  );
}
export default App;
```

class type

```
import React, { Component } from 'react';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">

      </div>
    );
  }
}

export default App;
```

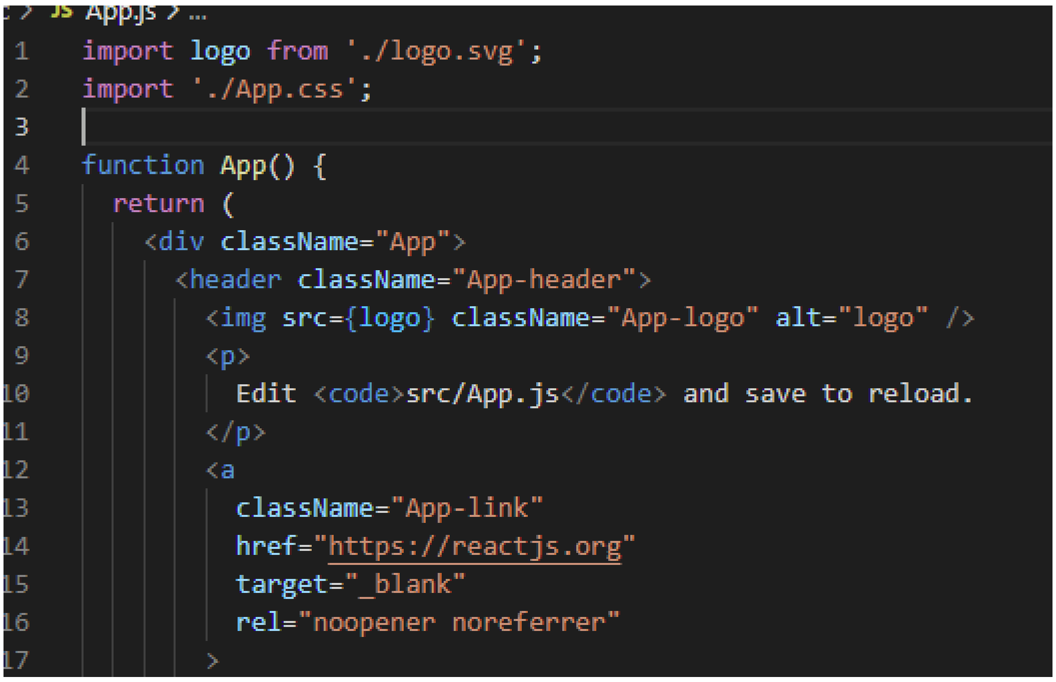
5-3. 기본 시작

Index.js



```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';

class App extends Component {
  render() {
    return (
      <div className="App">···
      </div>
    );
  }
}


export default App;
```

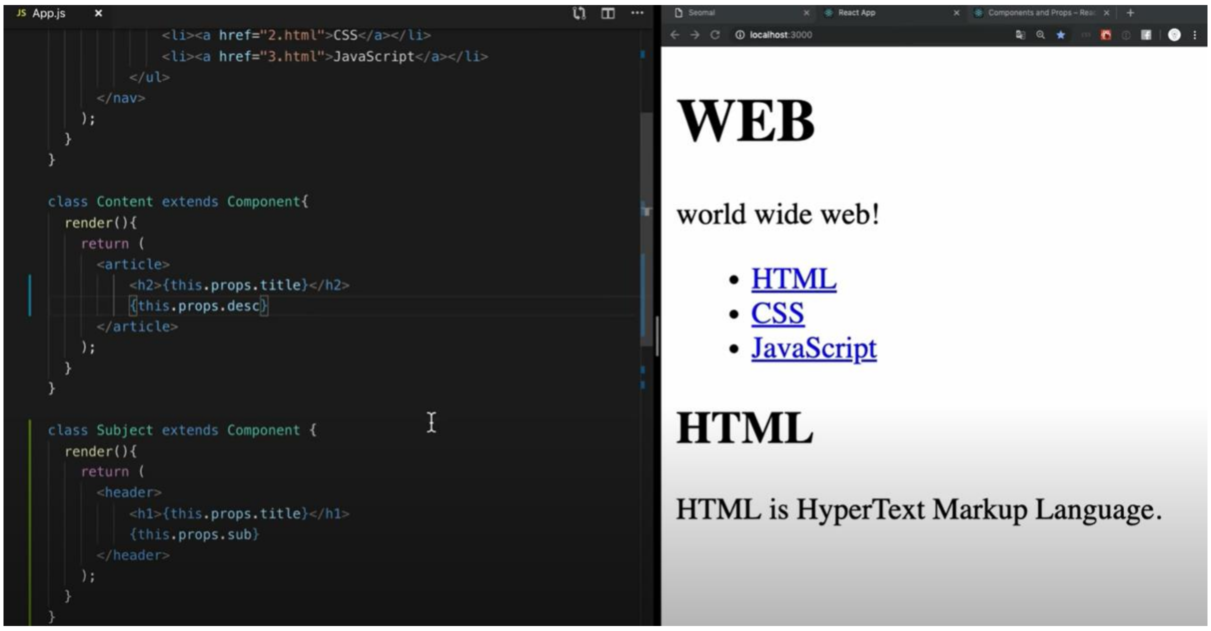
App.js



```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
```

5-4. props vs state

props



```
        <li><a href="2.html">CSS</a></li>
        <li><a href="3.html">JavaScript</a></li>
      </ul>
    </nav>
  );
  }
}

class Content extends Component{
  render(){
    return (
      <article>
          <h2>{this.props.title}</h2>
          {this.props.desc}
      </article>
    );
  }
}

class Subject extends Component {
  render(){
    return (
      <header>
          <h1>{this.props.title}</h1>
          {this.props.sub}
      </header>
    );
  }
}
```

WEB

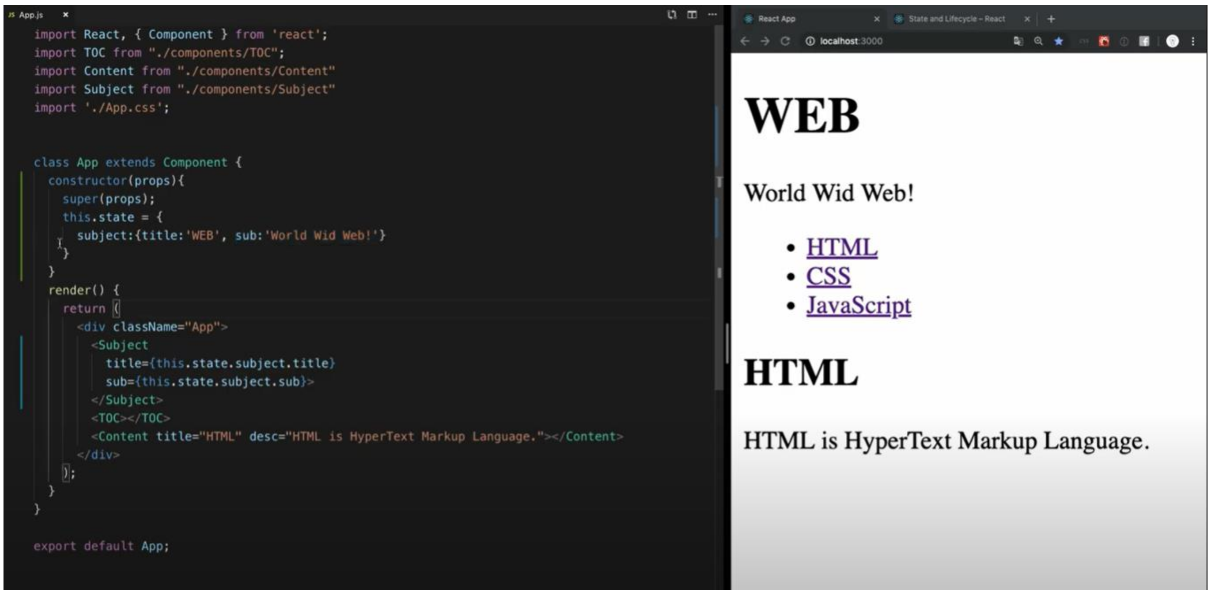
world wide web!

- HTML
- CSS
- JavaScript

HTML

HTML is HyperText Markup Language.

state



```
import React, { Component } from 'react';
import TOC from "./components/TOC";
import Content from "./components/Content"
import Subject from "./components/Subject"
import './App.css';


class App extends Component {
  constructor(props){
    super(props);
    this.state = {
      subject:{title:'WEB', sub:'World Wid Web!'}
    }
  }
  render() {
    return (
      <div className="App">
        <Subject
          title={this.state.subject.title}
          sub={this.state.subject.sub}>
        </Subject>
        <TOC></TOC>
        <Content title="HTML" desc="HTML is HyperText Markup Language."></Content>
      </div>
    );
  }
}

export default App;
```

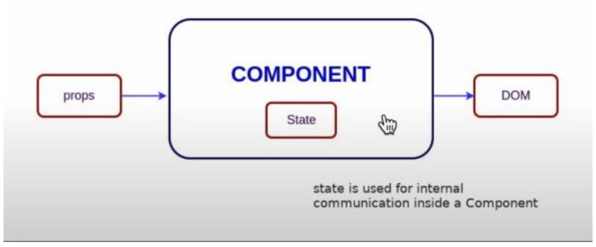
WEB

World Wid Web!

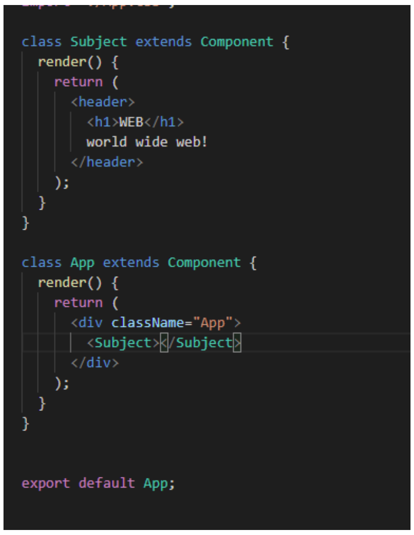
- HTML
- CSS
- JavaScript

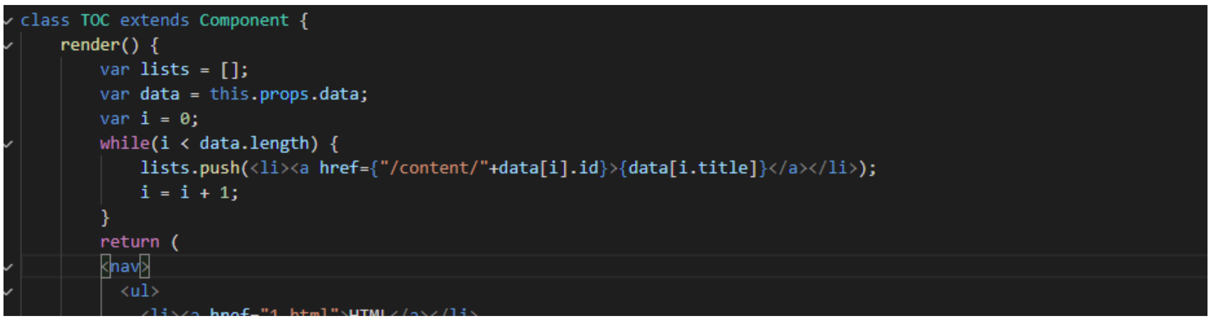
HTML

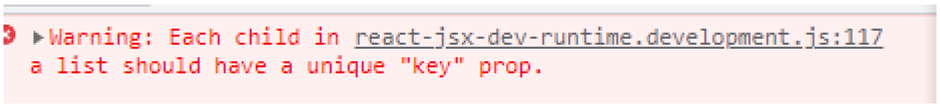
HTML is HyperText Markup Language.

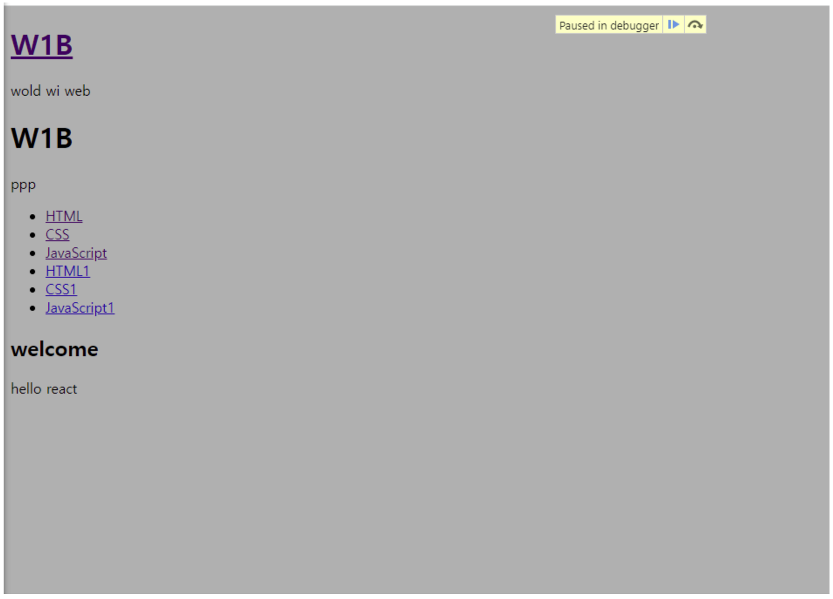


5-5. 컴퍼넌트 생성


```
class Subject extends Component {
  render() {
    return (
      <header>
        <h1>WEB</h1>
        world wide web!
      </header>
    );
  }
}

class App extends Component {
  render() {
    return (
      <div className="App">
        <Subject></Subject>
      </div>
    );
  }
}


export default App;
```


```
class TOC extends Component {
    render() {
        var lists = [];
        var data = this.props.data;
        var i = 0;
        while(i < data.length) {
            lists.push(<li><a href={"/content/"+data[i].id}>{data[i.title]}</a></li>);
            i = i + 1;
        }
        return (
        <nav>
          <ul>
```


```
▶Warning: Each child in react-jsx-dev-runtime.development.js:117
a list should have a unique "key" prop.
```



W1B

wold wi web

W1B

ppp

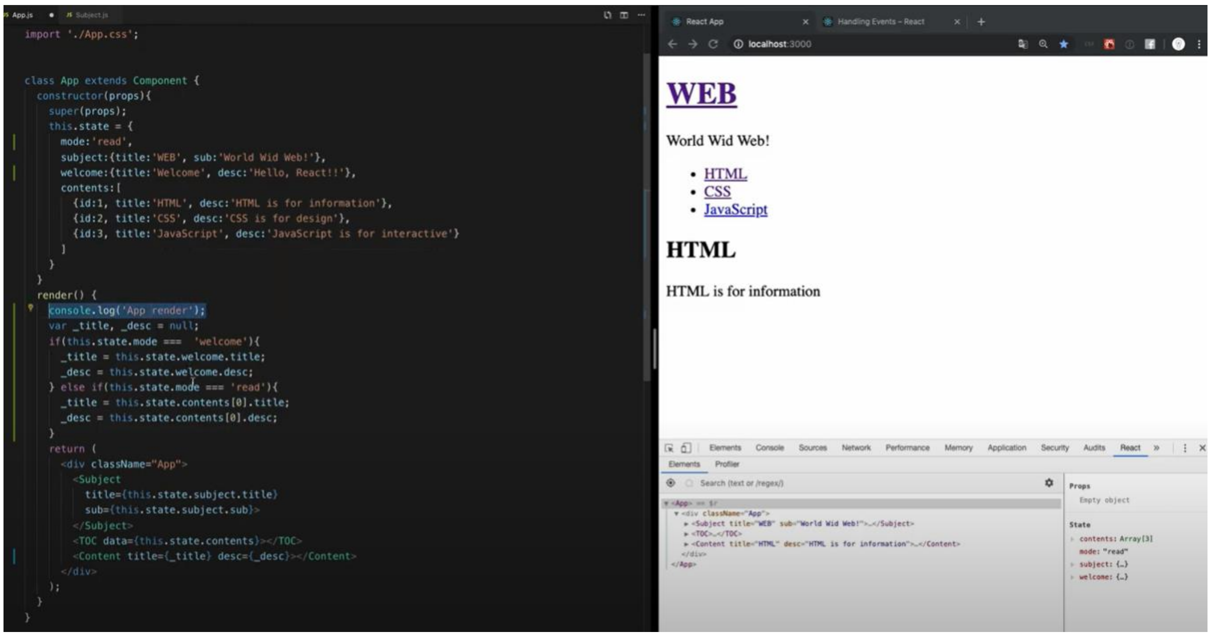
- HTML
- CSS
- JavaScript
- HTML1
- CSS1
- JavaScript1

welcome

hello react

5-6. 이벤트 설치

Render 호출 확인



```
import './App.css';


class App extends Component {
  constructor(props){
    super(props);
    this.state = {
      mode:'read',
      subject:{title:'WEB', sub:'World Wid Web!'},
      welcome:{title:'Welcome', desc:'Hello, React!!'},
      contents:[
        {id:1, title:'HTML', desc:'HTML is for information'},
        {id:2, title:'CSS', desc:'CSS is for design'},
        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'}
      ]
    }
  }
  render() {
    console.log('App render');
    var _title, _desc = null;
    if(this.state.mode ===  'welcome'){
      _title = this.state.welcome.title;
      _desc = this.state.welcome.desc;
    } else if(this.state.mode === 'read'){
      _title = this.state.contents[0].title;
      _desc = this.state.contents[0].desc;
    }
    return (
      <div className="App">
        <Subject
          title={this.state.subject.title}
          sub={this.state.subject.sub}>
        </Subject>
        <TOC data={this.state.contents}></TOC>
        <Content title={_title} desc={_desc}></Content>
      </div>
    );
  }
}
```

WEB

World Wid Web!

- HTML
- CSS
- JavaScript

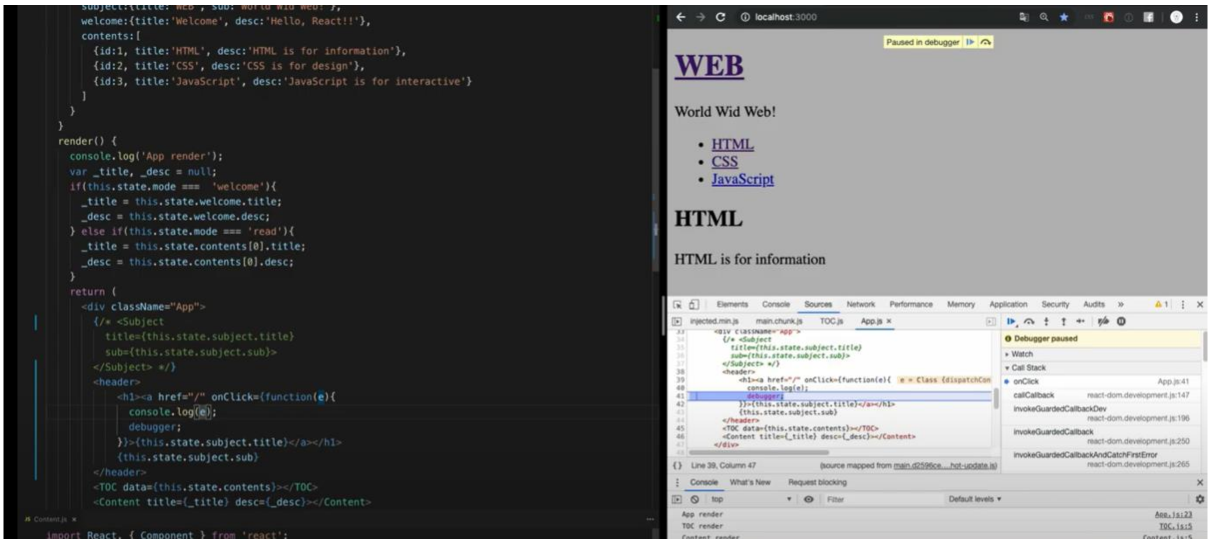
HTML

HTML is for information

```
▼ <App> == $r
  ▼ <div className="App">
    ▶ <Subject title="WEB" sub="World Wid Web!">…</Subject>
    ▶ <TOC>…</TOC>
    ▶ <Content title="HTML" desc="HTML is for information">…</Content>
    </div>
  </App>
```

```
Props
  Empty object

State
▶ contents: Array[3]
  mode: "read"
▶ subject: {…}
▶ welcome: {…}
```

onClick



```
      welcome:{title:'Welcome', desc:'Hello, React!!'},
      contents:[
        {id:1, title:'HTML', desc:'HTML is for information'},
        {id:2, title:'CSS', desc:'CSS is for design'},
        {id:3, title:'JavaScript', desc:'JavaScript is for interactive'}
      ]
    }
  }
  render() {
    console.log('App render');
    var _title, _desc = null;
    if(this.state.mode ===  'welcome'){
      _title = this.state.welcome.title;
      _desc = this.state.welcome.desc;
    } else if(this.state.mode === 'read'){
      _title = this.state.contents[0].title;
      _desc = this.state.contents[0].desc;
    }
    return (
      <div className="App">
        {/* <Subject
          title={this.state.subject.title}
          sub={this.state.subject.sub}>
        </Subject> */}
        <header>
            <h1><a href="/" onClick={function(e){
              console.log(e);
              debugger;
            }}>{this.state.subject.title}</a></h1>
            {this.state.subject.sub}
        </header>
        <TOC data={this.state.contents}></TOC>
        <Content title={_title} desc={_desc}></Content>
```

```
import React, { Component } from 'react';
```

Paused in debugger

WEB

World Wid Web!

- HTML
- CSS
- JavaScript

HTML

HTML is for information

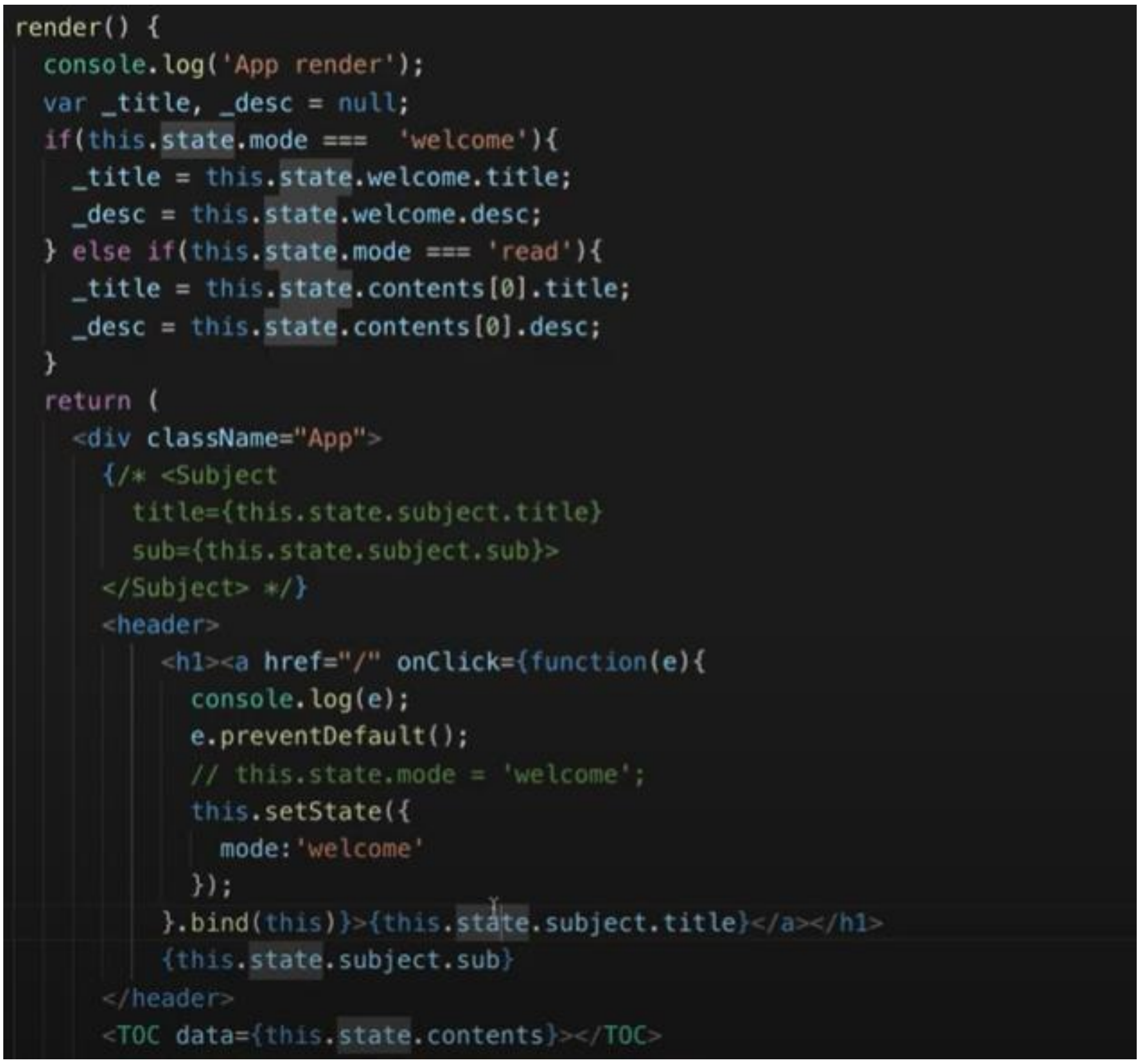
```
        <h1><a href="/" onClick={function(e){  e = Class {dispatchCon
          console.log(e);
          debugger;
        }}>{this.state.subject.title}</a></h1>
        {this.state.subject.sub}
      </header>
      <TOC data={this.state.contents}></TOC>
      <Content title={_title} desc={_desc}></Content>
    </div>
```

Line 39, Column 47 (source mapped from main.d2596ce….hot-update.js)

```
Debugger paused
▶ Watch
▼ Call Stack
● onClick                               App.js:41
  callCallback          react-dom.development.js:147
  invokeGuardedCallbackDev
                        react-dom.development.js:196
  invokeGuardedCallback
                        react-dom.development.js:250
  invokeGuardedCallbackAndCatchFirstError
                        react-dom.development.js:265
```

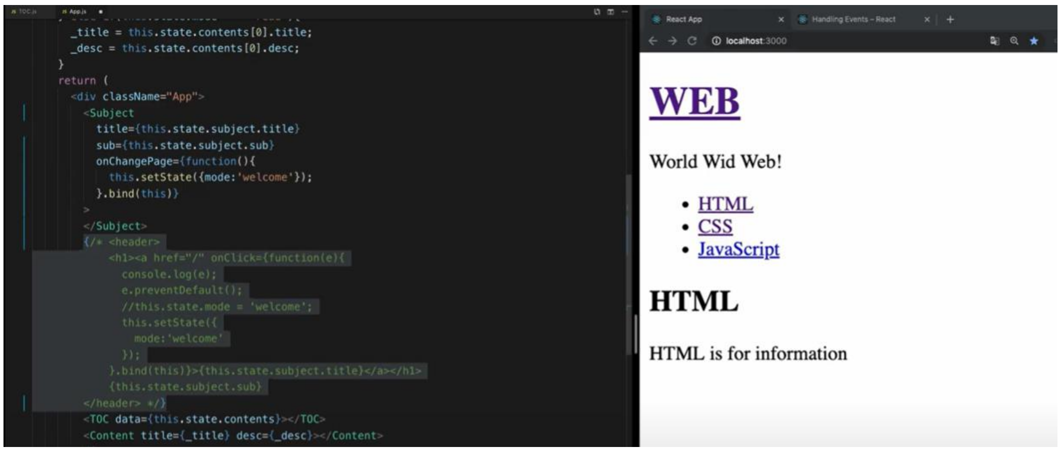
```
App render                                  App.js:23
TOC render                                  TOC.js:5
Content render                              Content.js:5
```

setState



```
render() {
  console.log('App render');
  var _title, _desc = null;
  if(this.state.mode ===  'welcome'){
    _title = this.state.welcome.title;
    _desc = this.state.welcome.desc;
  } else if(this.state.mode === 'read'){
    _title = this.state.contents[0].title;
    _desc = this.state.contents[0].desc;
  }
  return (
    <div className="App">
      {/* <Subject
        title={this.state.subject.title}
        sub={this.state.subject.sub}>
      </Subject> */}
      <header>
          <h1><a href="/" onClick={function(e){
            console.log(e);
            e.preventDefault();
            // this.state.mode = 'welcome';
            this.setState({
              mode:'welcome'
            });
          }.bind(this)}>{this.state.subject.title}</a></h1>
          {this.state.subject.sub}
      </header>
      <TOC data={this.state.contents}></TOC>
```

5-7. 컴퍼넌트 만들기



```
      _title = this.state.contents[0].title;
      _desc = this.state.contents[0].desc;
    }
    return (
      <div className="App">
        <Subject
          title={this.state.subject.title}
          sub={this.state.subject.sub}
          onChangePage={function(){
            this.setState({mode:'welcome'});
          }.bind(this)}
        >
        </Subject>
        {/* <header>
            <h1><a href="/" onClick={function(e){
              console.log(e);
              e.preventDefault();
              //this.state.mode = 'welcome';
              this.setState({
                mode:'welcome'
              });
            }.bind(this)}>{this.state.subject.title}</a></h1>
            {this.state.subject.sub}
        </header> */}
        <TOC data={this.state.contents}></TOC>
        <Content title={_title} desc={_desc}></Content>
```

WEB

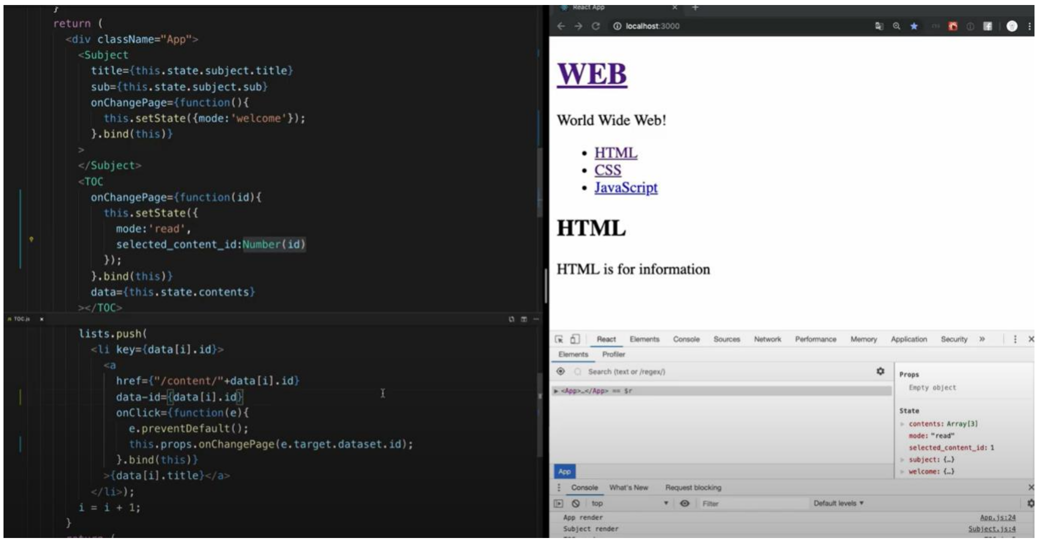
World Wid Web!

- HTML
- CSS
- JavaScript

HTML

HTML is for information

Toc



```
    return (
      <div className="App">
        <Subject
          title={this.state.subject.title}
          sub={this.state.subject.sub}
          onChangePage={function(){
            this.setState({mode:'welcome'});
          }.bind(this)}
        >
        </Subject>
        <TOC
          onChangePage={function(id){
            this.setState({
              mode:'read',
              selected_content_id:Number(id)
            });
          }.bind(this)}
          data={this.state.contents}
        ></TOC>
```

```
        lists.push(
          <li key={data[i].id}>
            <a
              href={"/content/"+data[i].id}
              data-id={data[i].id}
              onClick={function(e){
                e.preventDefault();
                this.props.onChangePage(e.target.dataset.id);
              }.bind(this)}
            >{data[i].title}</a>
          </li>);
        i = i + 1;
      }
```

WEB

World Wide Web!

- HTML
- CSS
- JavaScript

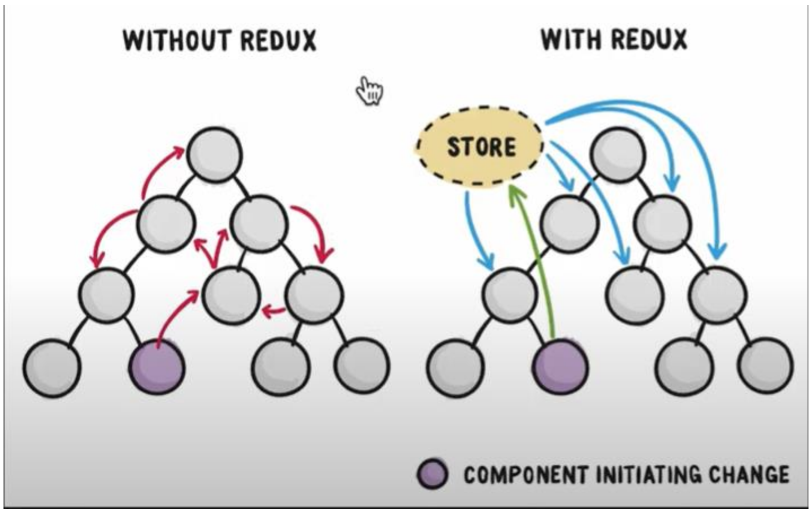
HTML

HTML is for information

```
▶ <App>…</App> == $r

Props
  Empty object

State
▶ contents: Array[3]
  mode: "read"
  selected_content_id: 1
▶ subject: {…}
▶ welcome: {…}
```

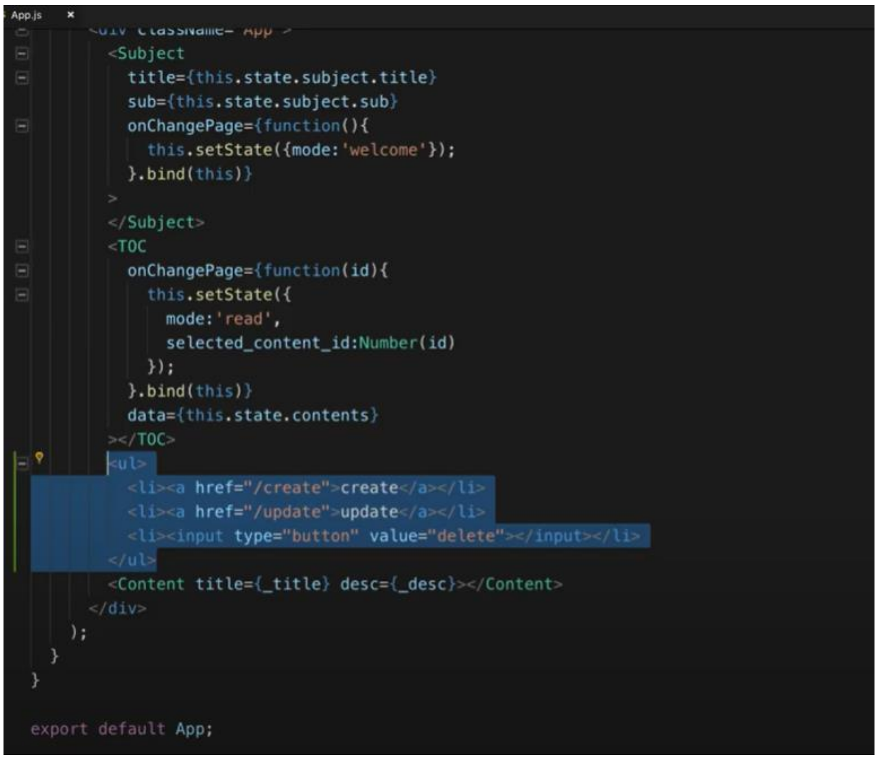
```
App render                                   App.js:24
Subject render                           Subject.js:4
```

5-7-1. Redux



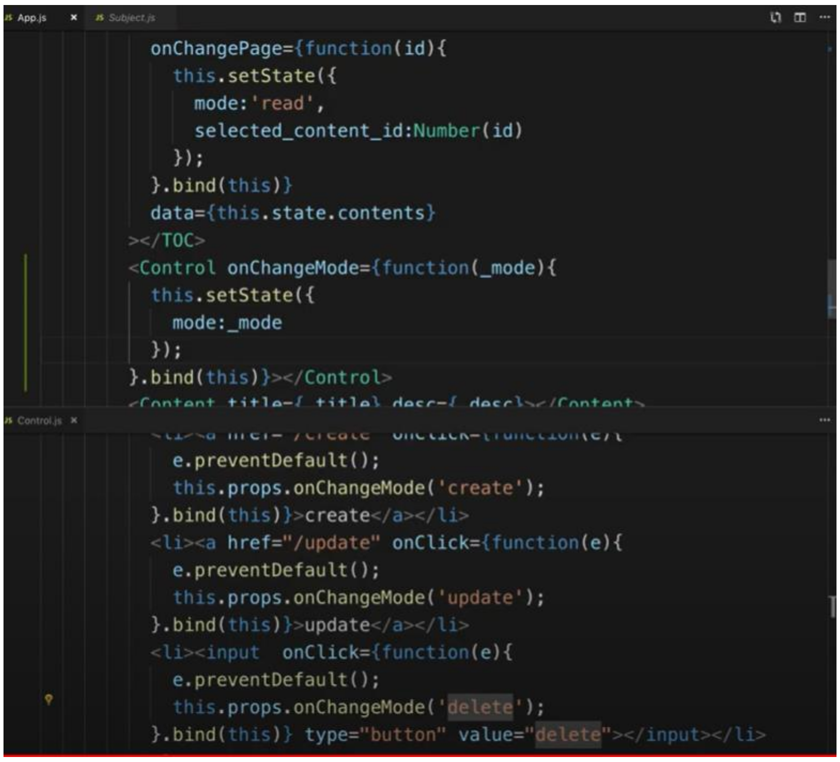
5-8. crud – create

App.js



```
    <div className="App">
      <Subject
        title={this.state.subject.title}
        sub={this.state.subject.sub}
        onChangePage={function(){
          this.setState({mode:'welcome'});
        }.bind(this)}
      >
      </Subject>
      <TOC
        onChangePage={function(id){
          this.setState({
            mode:'read',
            selected_content_id:Number(id)
          });
        }.bind(this)}
        data={this.state.contents}
      ></TOC>
      <ul>
        <li><a href="/create">create</a></li>
        <li><a href="/update">update</a></li>
        <li><input type="button" value="delete"></input></li>
      </ul>
      <Content title={_title} desc={_desc}></Content>
    </div>
  );
 }
}

export default App;
```

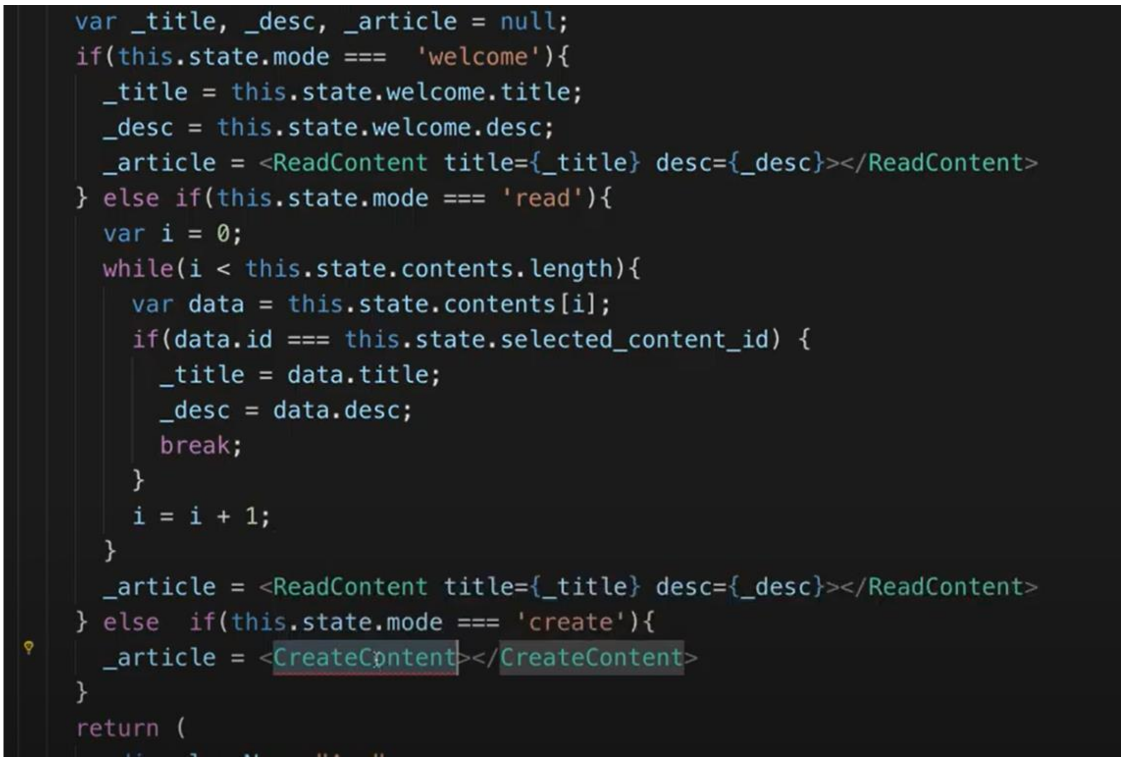
Subject.js



```
        onChangePage={function(id){
          this.setState({
            mode:'read',
            selected_content_id:Number(id)
          });
        }.bind(this)}
        data={this.state.contents}
      ></TOC>
      <Control onChangeMode={function(_mode){
        this.setState({
          mode:_mode
        });
      }.bind(this)}></Control>
      <Content title={_title} desc={_desc}></Content>
```

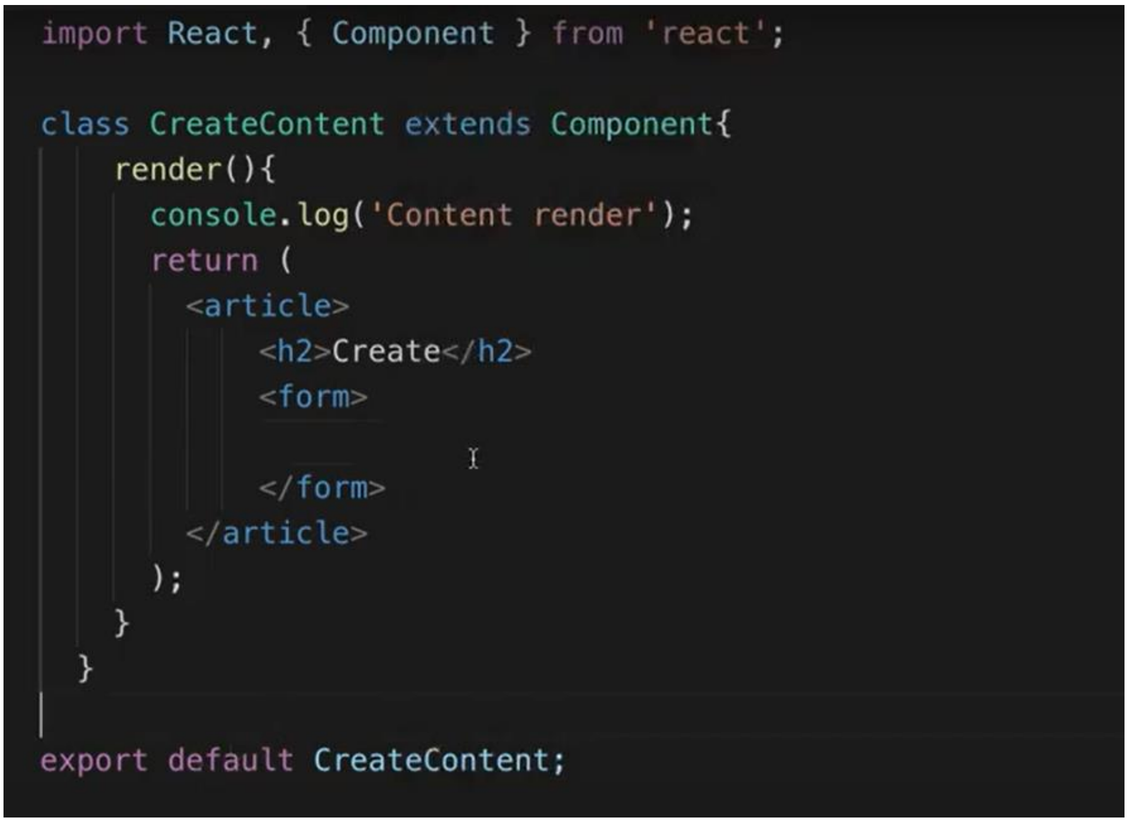
```
        <li><a href="/create" onClick={function(e){
          e.preventDefault();
          this.props.onChangeMode('create');
        }.bind(this)}>create</a></li>
        <li><a href="/update" onClick={function(e){
          e.preventDefault();
          this.props.onChangeMode('update');
        }.bind(this)}>update</a></li>
        <li><input  onClick={function(e){
          e.preventDefault();
          this.props.onChangeMode('delete');
        }.bind(this)} type="button" value="delete"></input></li>
```

App.js

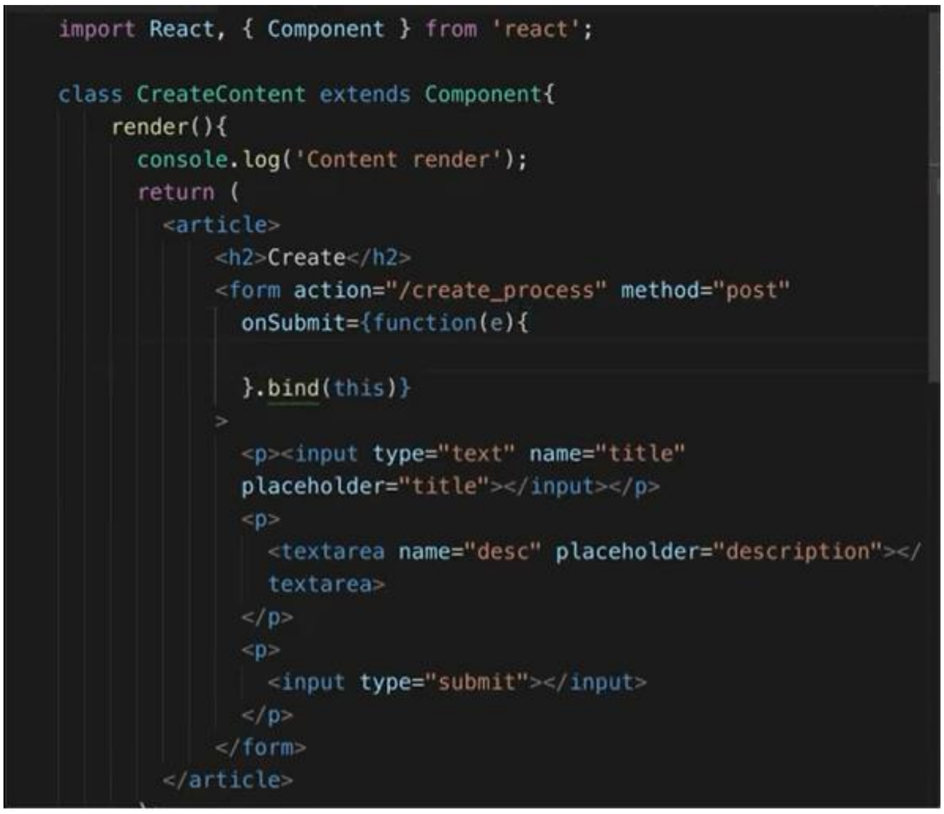


```
var _title, _desc, _article = null;
if(this.state.mode ===  'welcome'){
  _title = this.state.welcome.title;
  _desc = this.state.welcome.desc;
  _article = <ReadContent title={_title} desc={_desc}></ReadContent>
} else if(this.state.mode === 'read'){
  var i = 0;
  while(i < this.state.contents.length){
    var data = this.state.contents[i];
    if(data.id === this.state.selected_content_id) {
      _title = data.title;
      _desc = data.desc;
      break;
    }
    i = i + 1;
  }
  _article = <ReadContent title={_title} desc={_desc}></ReadContent>
} else  if(this.state.mode === 'create'){
  _article = <CreateContent></CreateContent>
}
return (
```

CreateContent.js

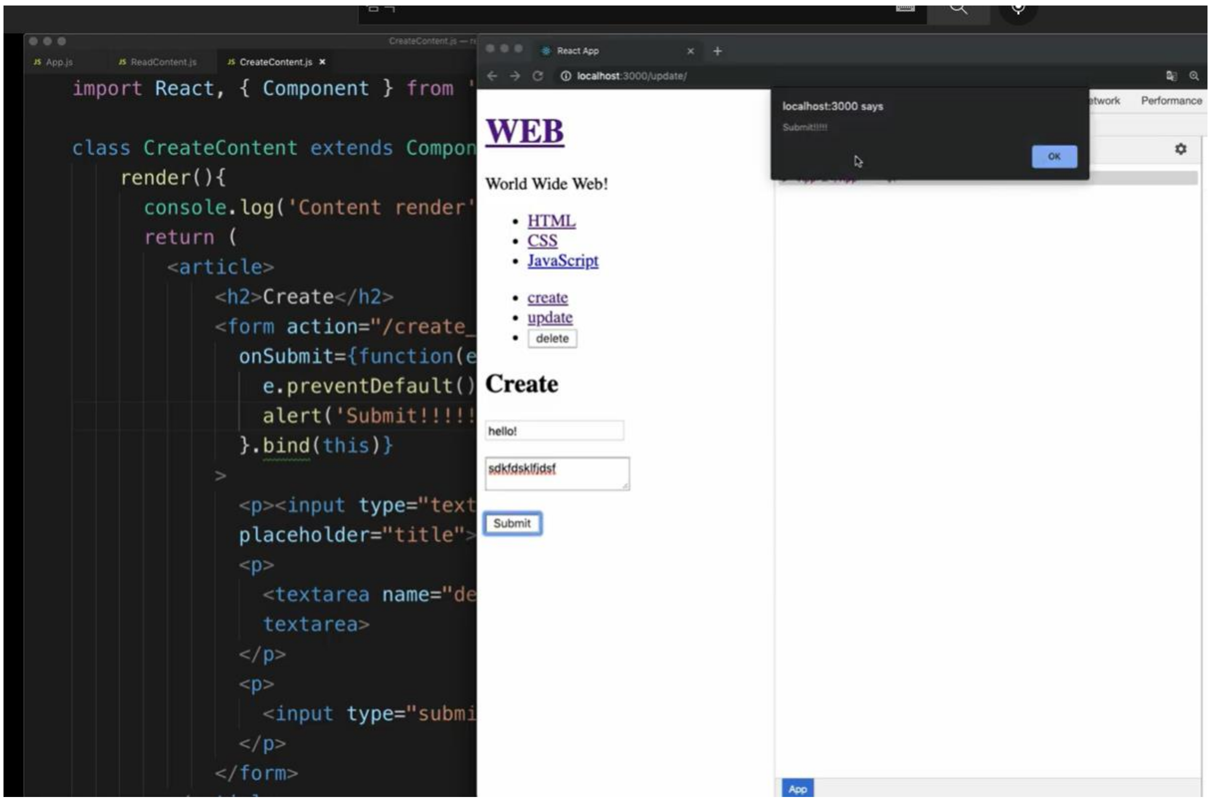


```
import React, { Component } from 'react';

class CreateContent extends Component{
    render(){
      console.log('Content render');
      return (
        <article>
            <h2>Create</h2>
            <form>

            </form>
        </article>
      );
    }
  }

export default CreateContent;
```



```
import React, { Component } from 'react';

class CreateContent extends Component{
    render(){
      console.log('Content render');
      return (
        <article>
            <h2>Create</h2>
            <form action="/create_process" method="post"
              onSubmit={function(e){

              }.bind(this)}
            >
              <p><input type="text" name="title"
              placeholder="title"></input></p>
              <p>
                <textarea name="desc" placeholder="description"></
                textarea>
              </p>
              <p>
                <input type="submit"></input>
              </p>
            </form>
        </article>
```

실행



App.js ReadContent.js CreateContent.js

```
import React, { Component } from '
class CreateContent extends Compon
    render(){
      console.log('Content render'
      return (
        <article>
            <h2>Create</h2>
            <form action="/create_
              onSubmit={function(e
                e.preventDefault()
                alert('Submit!!!!!
              }.bind(this)}
            >
              <p><input type="text
              placeholder="title">
              <p>
                <textarea name="de
                textarea>
              </p>
              <p>
                <input type="submi
              </p>
            </form>
```

React App
localhost:3000/update/

WEB

World Wide Web!

- HTML
- CSS
- JavaScript

- create
- update
- delete

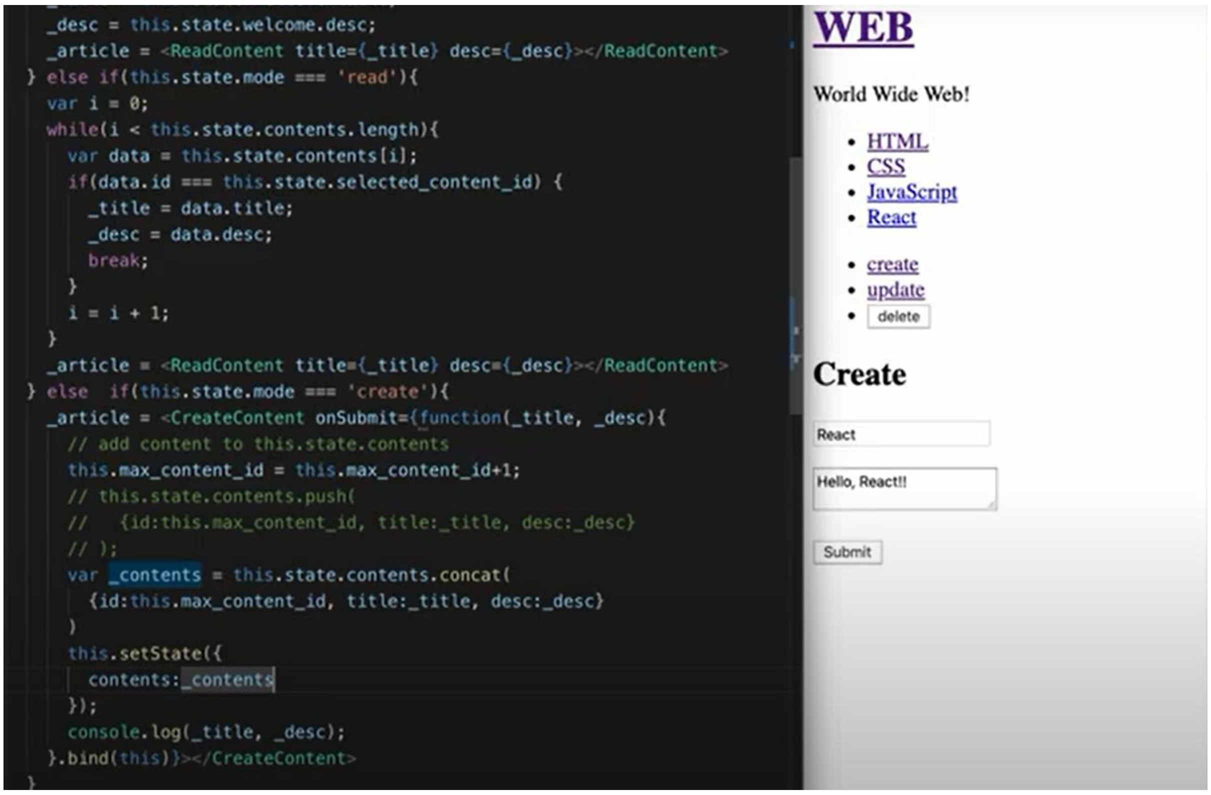
Create

hello!

sdkfdsklfjdsf

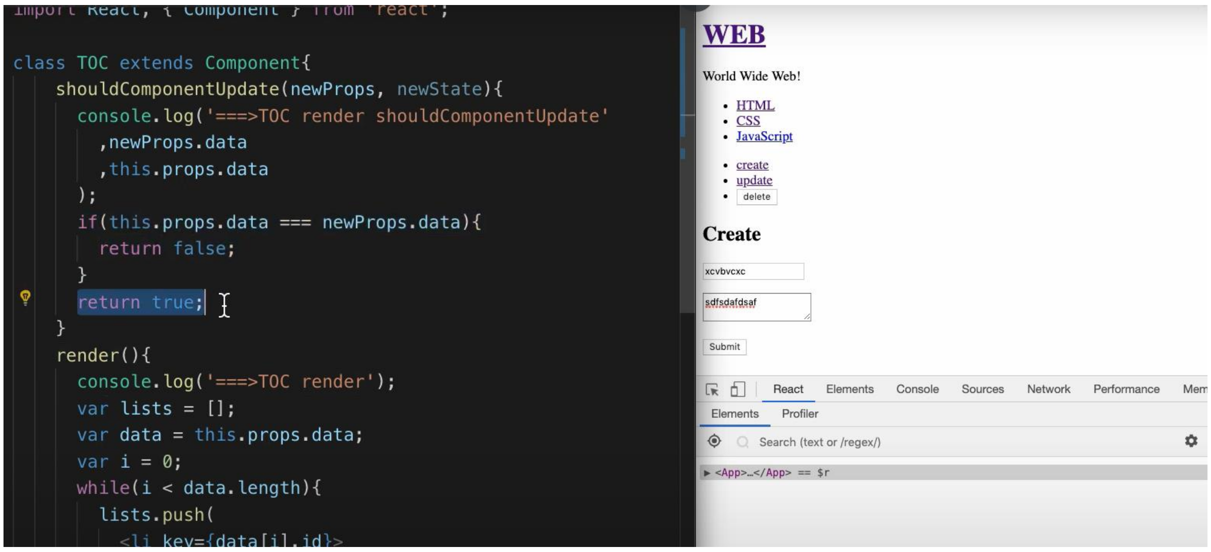
Submit

localhost:3000 says
Submit!!!!!
OK



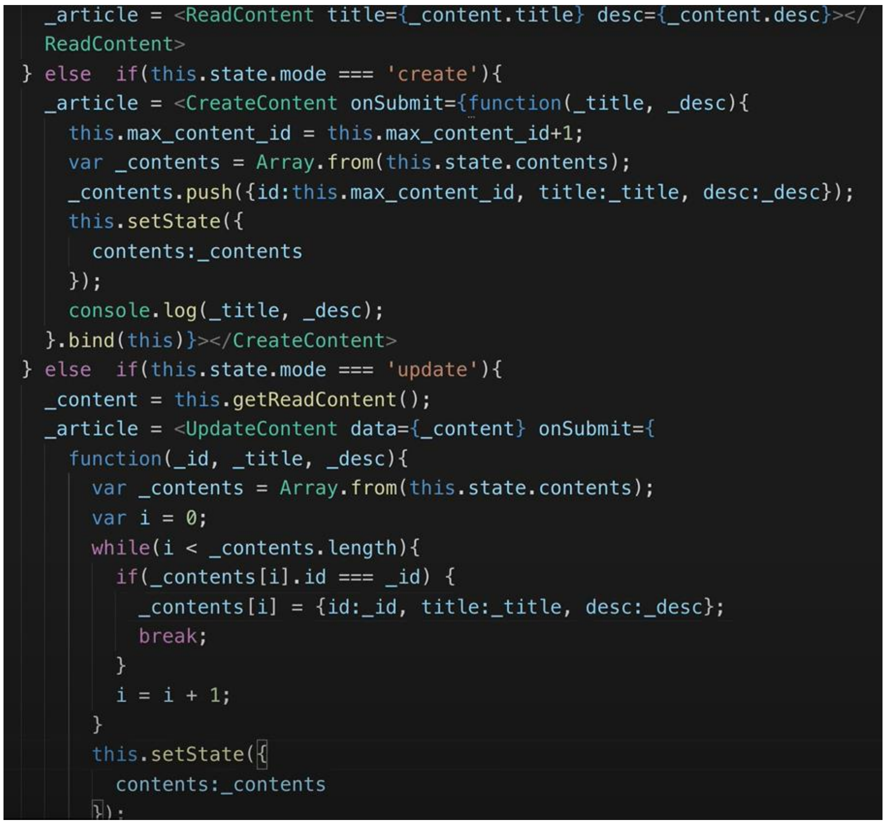
```
  _desc = this.state.welcome.desc;
  _article = <ReadContent title={_title} desc={_desc}></ReadContent>
} else if(this.state.mode === 'read'){
  var i = 0;
  while(i < this.state.contents.length){
    var data = this.state.contents[i];
    if(data.id === this.state.selected_content_id) {
      _title = data.title;
      _desc = data.desc;
      break;
    }
    i = i + 1;
  }
  _article = <ReadContent title={_title} desc={_desc}></ReadContent>
} else  if(this.state.mode === 'create'){
  _article = <CreateContent onSubmit={function(_title, _desc){
    // add content to this.state.contents
    this.max_content_id = this.max_content_id+1;
    // this.state.contents.push(
    //   {id:this.max_content_id, title:_title, desc:_desc}
    // );
    var _contents = this.state.contents.concat(
      {id:this.max_content_id, title:_title, desc:_desc}
    )
    this.setState({
      contents:_contents
    });
    console.log(_title, _desc);
  }.bind(this)}></CreateContent>
}
```

shouldComponetUpdate



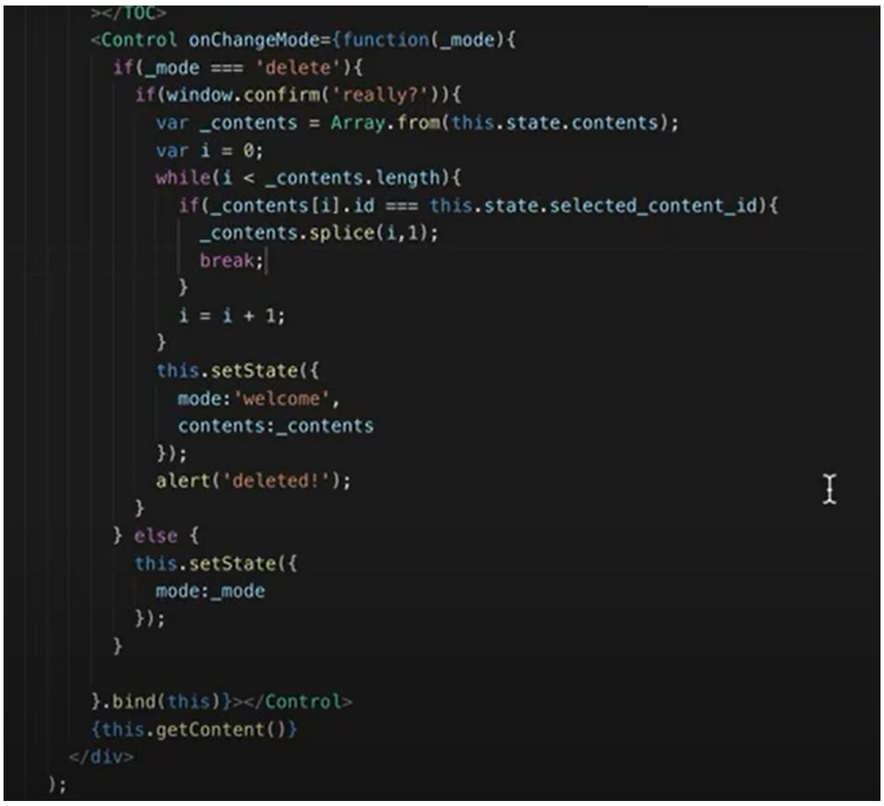
```
import React, { Component } from 'react';

class TOC extends Component{
    shouldComponentUpdate(newProps, newState){
      console.log('===>TOC render shouldComponentUpdate'
        ,newProps.data
        ,this.props.data
      );
      if(this.props.data === newProps.data){
        return false;
      }
      return true;
    }
    render(){
      console.log('===>TOC render');
      var lists = [];
      var data = this.props.data;
      var i = 0;
      while(i < data.length){
        lists.push(
          <li key={data[i].id}>
```

5-9. crud – update



```
  _article = <ReadContent title={_content.title} desc={_content.desc}></
  ReadContent>
} else  if(this.state.mode === 'create'){
  _article = <CreateContent onSubmit={function(_title, _desc){
    this.max_content_id = this.max_content_id+1;
    var _contents = Array.from(this.state.contents);
    _contents.push({id:this.max_content_id, title:_title, desc:_desc});
    this.setState({
      contents:_contents
    });
    console.log(_title, _desc);
  }.bind(this)}></CreateContent>
} else  if(this.state.mode === 'update'){
  _content = this.getReadContent();
  _article = <UpdateContent data={_content} onSubmit={
    function(_id, _title, _desc){
      var _contents = Array.from(this.state.contents);
      var i = 0;
      while(i < _contents.length){
        if(_contents[i].id === _id) {
          _contents[i] = {id:_id, title:_title, desc:_desc};
          break;
        }
        i = i + 1;
      }
      this.setState({
        contents:_contents
      });
```

5-10. crud – delete

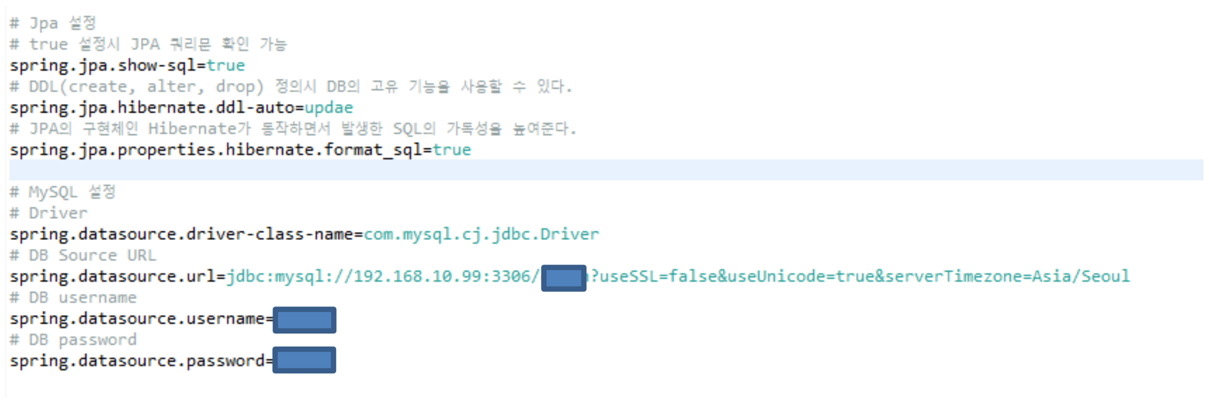


```
    ></TOC>
    <Control onChangeMode={function(_mode){
      if(_mode === 'delete'){
        if(window.confirm('really?')){
          var _contents = Array.from(this.state.contents);
          var i = 0;
          while(i < _contents.length){
            if(_contents[i].id === this.state.selected_content_id){
              _contents.splice(i,1);
              break;
            }
            i = i + 1;
          }
          this.setState({
            mode:'welcome',
            contents:_contents
          });
          alert('deleted!');
        }
      } else {
        this.setState({
          mode:_mode
        });
      }

    }.bind(this)}></Control>
    {this.getContent()}
  </div>
);
```

6. 사용자 관리 화면

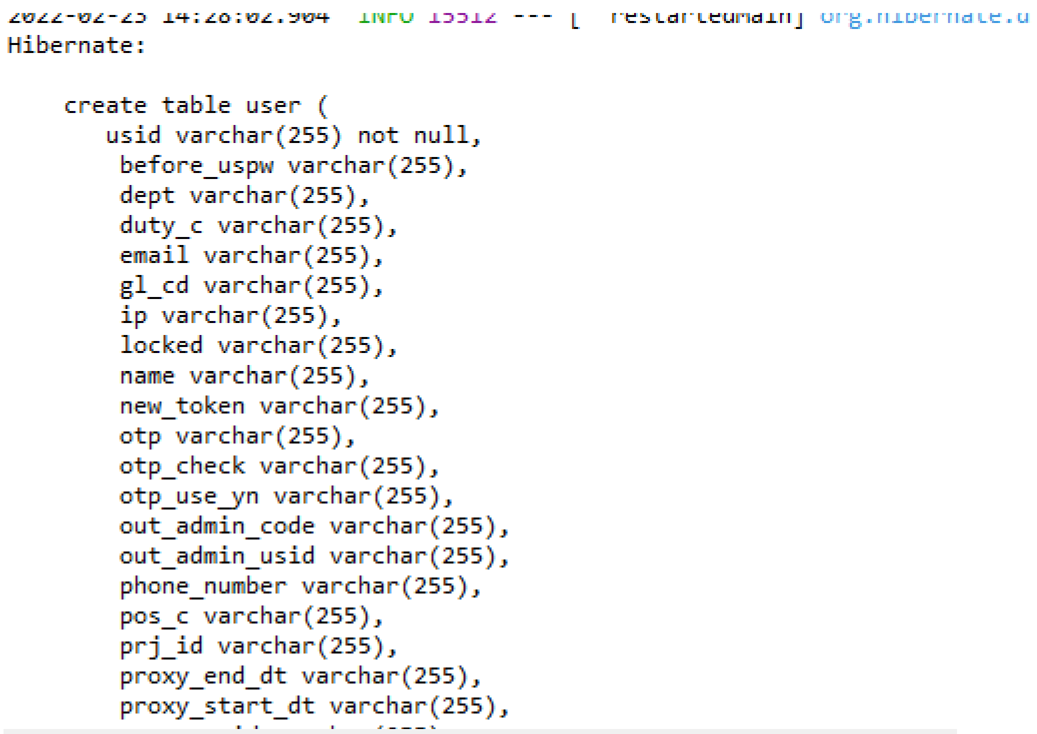
6-1. 의존성 추가 및 mysql 설정

implementation'mysql:mysql-connector-java'

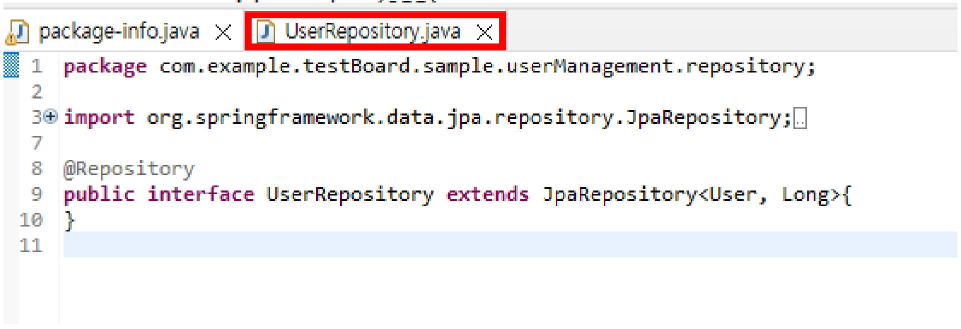


```
# Jpa 설정
# true 설정시 JPA 쿼리문 확인 가능
spring.jpa.show-sql=true
# DDL(create, alter, drop) 정의시 DB의 고유 기능을 사용할 수 있다.
spring.jpa.hibernate.ddl-auto=updae
# JPA의 구현체인 Hibernate가 동작하면서 발생한 SQL의 가독성을 높여준다.
spring.jpa.properties.hibernate.format_sql=true

# MySQL 설정
# Driver
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
# DB Source URL
spring.datasource.url=jdbc:mysql://192.168.10.99:3306/[redacted]?useSSL=false&useUnicode=true&serverTimezone=Asia/Seoul
# DB username
spring.datasource.username=[redacted]
# DB password
spring.datasource.password=[redacted]
```

6-2. 롬복 및 스프링데이터 jpa



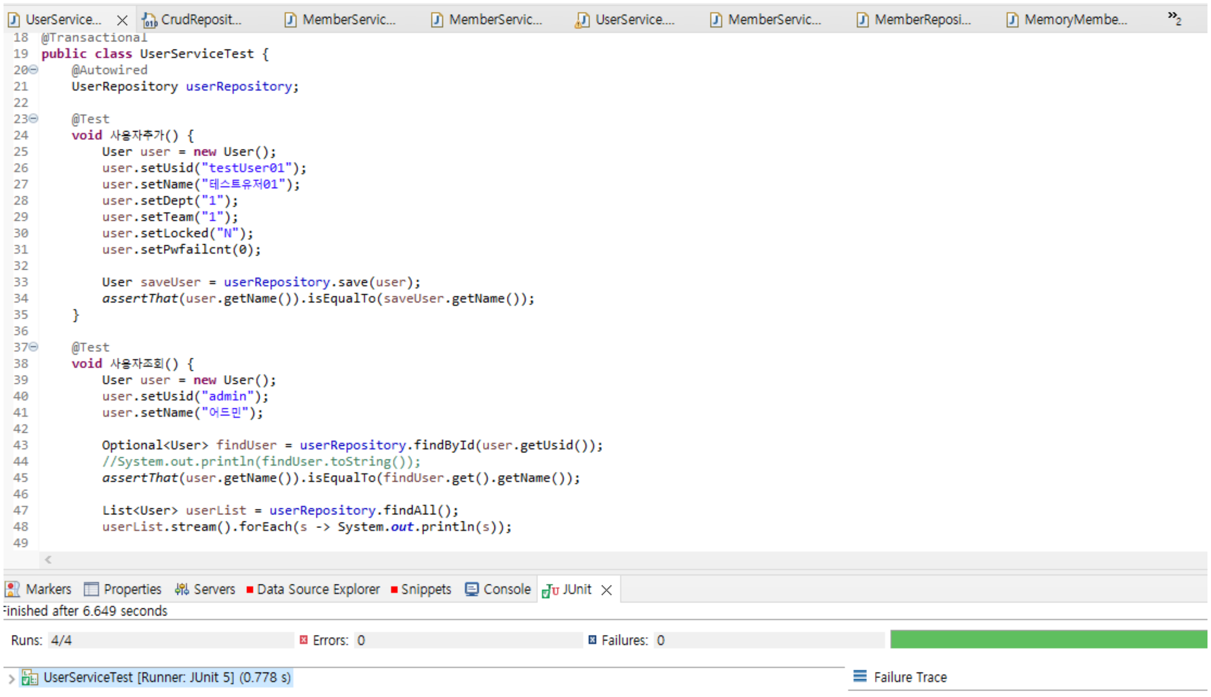
```
2022-02-23 14:28:02.904  INFO 15512 --- [  restartedMain] org.hibernate.d
Hibernate: 

    create table user (
       usid varchar(255) not null,
        before_uspw varchar(255),
        dept varchar(255),
        duty_c varchar(255),
        email varchar(255),
        gl_cd varchar(255),
        ip varchar(255),
        locked varchar(255),
        name varchar(255),
        new_token varchar(255),
        otp varchar(255),
        otp_check varchar(255),
        otp_use_yn varchar(255),
        out_admin_code varchar(255),
        out_admin_usid varchar(255),
        phone_number varchar(255),
        pos_c varchar(255),
        prj_id varchar(255),
        proxy_end_dt varchar(255),
        proxy_start_dt varchar(255),
```



```
package com.example.testBoard.sample.userManagement.repository;

import org.springframework.data.jpa.repository.JpaRepository;

@Repository
public interface UserRepository extends JpaRepository<User, Long>{
}

```

6-3. jpa 메서드 및 junit 테스트

- save : entity를 저장하는 메소드(insert, update)
- flush : EntityManager의 내용을 DB에 동기화하는 메소드
- saveAndFlush : entity에 대한 저장 작업 후 flush
- delete : entity를 삭제하는 메소드(delete)
- deleteAll : DB의 모든 레코드를 persistence context로 읽어와 삭제하는 메소드
- deleteInBatch : persistence context로 읽어오지 않고 DB의 모든 레코드를 삭제하는 메소드
- findOne : primary key로 DB에서 Entity를 찾아오는 메소드(select)
- findAll : 모든 entity를 찾아오는 메소드(select)
- exists : primary key에 해당하는 entity가 존재하는 확인하는 메소드
- count : entity의 갯수를 확인하는 메소드



```java
@Transactional
public class UserServiceTest {
    @Autowired
    UserRepository userRepository;

    @Test
    void 사용자추가() {
        User user = new User();
        user.setUsid("testUser01");
        user.setName("테스트유저01");
        user.setDept("1");
        user.setTeam("1");
        user.setLocked("N");
        user.setPwfailcnt(0);

        User saveUser = userRepository.save(user);
        assertThat(user.getName()).isEqualTo(saveUser.getName());
    }

    @Test
    void 사용자조회() {
        User user = new User();
        user.setUsid("admin");
        user.setName("어드민");

        Optional<User> findUser = userRepository.findById(user.getUsid());
        //System.out.println(findUser.toString());
        assertThat(user.getName()).isEqualTo(findUser.get().getName());

        List<User> userList = userRepository.findAll();
        userList.stream().forEach(s -> System.out.println(s));

```

```
Finished after 6.649 seconds

Runs:  4/4        Errors:  0        Failures:  0

> UserServiceTest [Runner: JUnit 5] (0.778 s)
```

6-4. 현재 Package.json

```json
{
  "name": "webapp",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.16.2",
    "@testing-library/react": "^12.1.2",
    "@testing-library/user-event": "^13.5.0",
    "react": "^17.0.2",
    "react-dom": "^17.0.2",
    "react-scripts": "5.0.0",
    "web-vitals": "^2.1.4"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  },
  "proxy": "http://localhost:8080"
}
```

6-5. toast-ui적용

```
npm install --save @toast-ui/react-grid
```

종속성에러발생시

npm install --save --legacy-peer-deps @toast-ui/react-grid

출처 : tui.grid/패키지/토스트 ui.react-grid at master · nhn/tui.grid · 기트허브 (github.com)

App.js추가

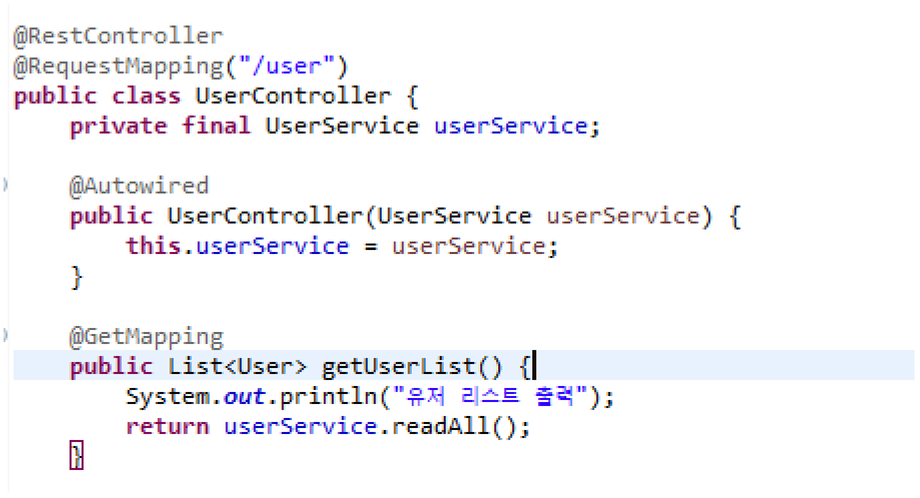
```
import 'tui-grid/dist/tui-grid.css';
import Grid from '@toast-ui/react-grid';


const data = [
  {id: 1, name: 'Editor'},
  {id: 2, name: 'Grid'},
  {id: 3, name: 'Chart'}
];

const columns = [
  {name: 'id', header: 'ID'},
  {name: 'name', header: 'Name'}
];

const MyComponent = () => (
  <Grid
    data={data}
    columns={columns}
    rowHeight={25}
    bodyHeight={100}
    heightResizable={true}
    rowHeaders={['rowNum']}
  />

);
```

////////////////////////

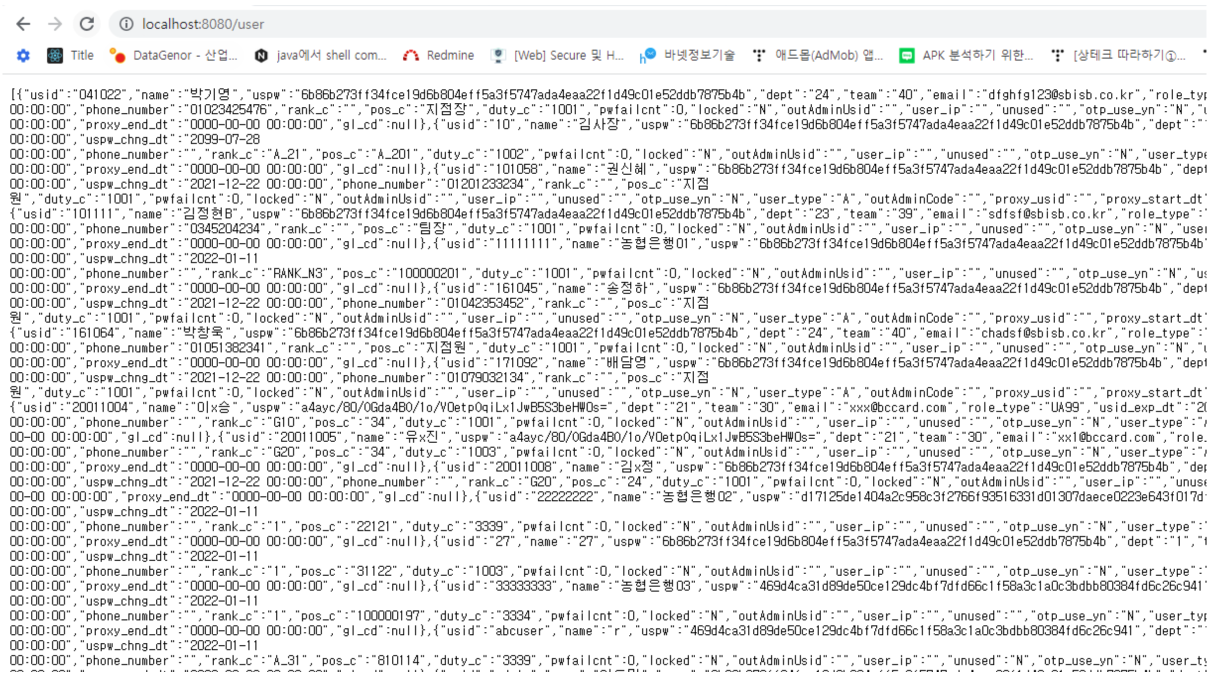
```
class MyComponent extends React.Component {
  gridRef = React.createRef();

  handleAppendRow = () => {
    this.gridRef.current.getInstance().appendRow({});
  };

  render() {
    return (
      <>
        <Grid ref={this.gridRef} data={data} columns={columns} />
        <button onClick={this.handleAppendRow}>Append Row</button>
      </>
    );
  }
}
```

MyComponent -> App, `React.Component -> Component` 변경

////class7////

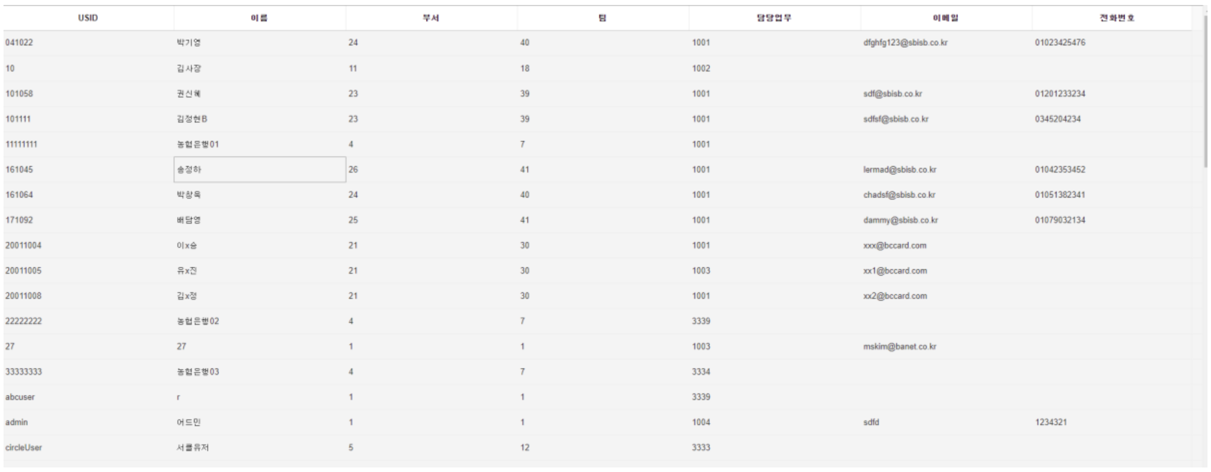


```
@RestController
@RequestMapping("/user")
public class UserController {
    private final UserService userService;

    @Autowired
    public UserController(UserService userService) {
        this.userService = userService;
    }

    @GetMapping
    public List<User> getUserList() {
        System.out.println("유저 리스트 출력");
        return userService.readAll();
    }
```



```
[{"usid":"041022","name":"박기영","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept":"24","team":"40","email":"dfghfg123@sbisb.co.kr","role_typ
00:00:00","phone_number":"01023425476","rank_c":"","pos_c":"지점장","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","u
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"10","name":"김사장","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept":"
00:00:00","uspw_chng_dt":"2099-07-28
00:00:00","phone_number":"","rank_c":"A_21","pos_c":"A_201","duty_c":"1002","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"101058","name":"권신혜","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept
00:00:00","uspw_chng_dt":"2021-12-22 00:00:00","phone_number":"01201233234","rank_c":"","pos_c":"지점
원","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"A","outAdminCode":"","proxy_usid":"","proxy_start_dt"
{"usid":"101111","name":"김정현B","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept":"23","team":"39","email":"sdfsf@sbisb.co.kr","role_type":"
00:00:00","phone_number":"0345204234","rank_c":"","pos_c":"팀장","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"11111111","name":"농협은행01","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b"
00:00:00","uspw_chng_dt":"2022-01-11
00:00:00","phone_number":"","rank_c":"RANK_N3","pos_c":"100000201","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","us
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"161045","name":"송정하","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept
00:00:00","uspw_chng_dt":"2021-12-22 00:00:00","phone_number":"01042353452","rank_c":"","pos_c":"지점
원","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"A","outAdminCode":"","proxy_usid":"","proxy_start_dt"
{"usid":"161064","name":"박창욱","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept":"24","team":"40","email":"chadsf@sbisb.co.kr","role_type":"
00:00:00","phone_number":"01051382341","rank_c":"","pos_c":"지점원","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","u
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"171092","name":"배담영","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept
00:00:00","uspw_chng_dt":"2021-12-22 00:00:00","phone_number":"01079032134","rank_c":"","pos_c":"지점
원","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"A","outAdminCode":"","proxy_usid":"","proxy_start_dt"
{"usid":"20011004","name":"이x승","uspw":"a4ayc/80/0Gda4B0/1o/V0etp0qiLx1JwB5S3beHW0s=","dept":"21","team":"30","email":"xxx@bccard.com","role_type":"UA99","usid_exp_dt":"20
00:00:00","phone_number":"","rank_c":"G10","pos_c":"34","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"
00-00 00:00:00","gl_cd":null},{"usid":"20011005","name":"유x진","uspw":"a4ayc/80/0Gda4B0/1o/V0etp0qiLx1JwB5S3beHW0s=","dept":"21","team":"30","email":"xx1@bccard.com","role_
00:00:00","phone_number":"","rank_c":"G20","pos_c":"34","duty_c":"1003","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"20011008","name":"김x정","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dep
00:00:00","uspw_chng_dt":"2021-12-22 00:00:00","phone_number":"","rank_c":"G20","pos_c":"24","duty_c":"1001","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unuse
00-00 00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"22222222","name":"농협은행02","uspw":"d17125de1404a2c958c3f2766f93516331d01307daece0223e643f017d
00:00:00","uspw_chng_dt":"2022-01-11
00:00:00","phone_number":"","rank_c":"1","pos_c":"22121","duty_c":"3339","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"27","name":"27","uspw":"6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b","dept":"1","
00:00:00","uspw_chng_dt":"2022-01-11
00:00:00","phone_number":"","rank_c":"1","pos_c":"31122","duty_c":"1003","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_type":"
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"33333333","name":"농협은행03","uspw":"469d4ca31d89de50ce129dc4bf7dfd66c1f58a3c1a0c3bdbb80384fd6c26c941"
00:00:00","uspw_chng_dt":"2022-01-11
00:00:00","phone_number":"","rank_c":"1","pos_c":"100000197","duty_c":"3334","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"","otp_use_yn":"N","user_typ
00:00:00","proxy_end_dt":"0000-00-00 00:00:00","gl_cd":null},{"usid":"abcuser","name":"r","uspw":"469d4ca31d89de50ce129dc4bf7dfd66c1f58a3c1a0c3bdbb80384fd6c26c941","dept":"
00:00:00","uspw_chng_dt":"2022-01-11
00:00:00","phone_number":"","rank_c":"A_31","pos_c":"810114","duty_c":"3339","pwfailcnt":0,"locked":"N","outAdminUsid":"","user_ip":"","unused":"N","otp_use_yn":"N","user_ty
```

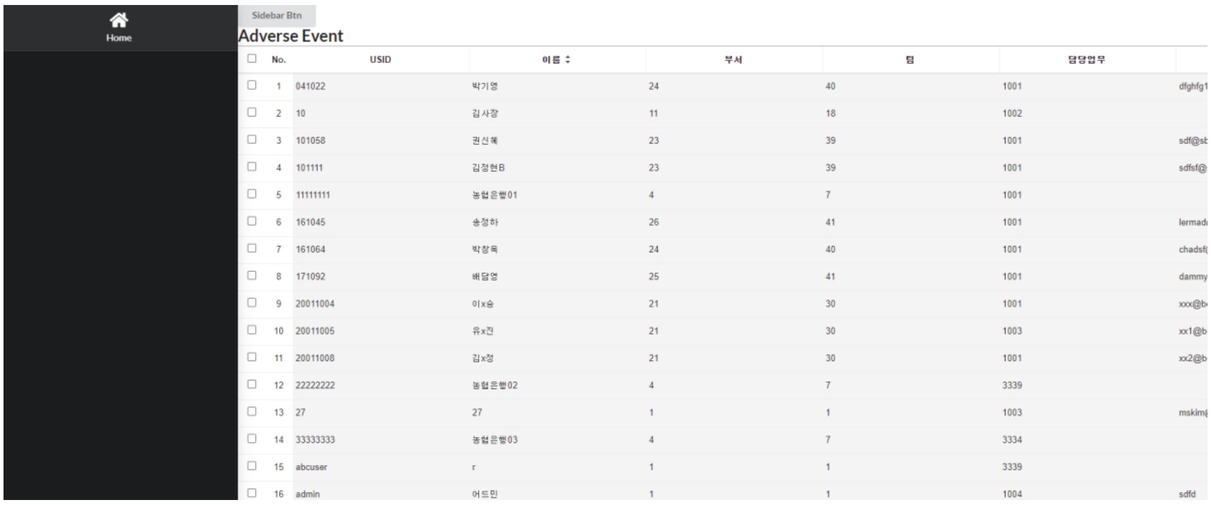
특정한 json 타입으로 넘겨줘야 리스트가 출력됨



USID	이름	부서	팀	담당업무	이메일	전화번호
041022	박기영	24	40	1001	dfghfg123@sbisb.co.kr	01023425476
10	김사장	11	18	1002		
101058	권신혜	23	39	1001	sdf@sbisb.co.kr	01201233234
101111	김정현B	23	39	1001	sdfsf@sbisb.co.kr	0345204234
11111111	농협은행01	4	7	1001		
161045	송정하	26	41	1001	lermad@sbisb.co.kr	01042353452
161064	박창욱	24	40	1001	chadsf@sbisb.co.kr	01051382341
171092	배담영	25	41	1001	dammy@sbisb.co.kr	01079032134
20011004	이x승	21	30	1001	xxx@bccard.com	
20011005	유x진	21	30	1003	xx1@bccard.com	
20011008	김x정	21	30	1001	xx2@bccard.com	
22222222	농협은행02	4	7	3339		
27	27	1	1	1003	mskim@banet.co.kr	
33333333	농협은행03	4	7	3334		
abcuser	r	1	1	3339		
admin	어드민	1	1	1004	sdfd	1234321
circleUser	서클유저	5	12	3333		

https://forward.nhn.com/2020/seoul/hands-on-labs/toastui.grid-account-book/04.html

6-6. 그외 컴퍼넌트 추가



Adverse Event

☐	No.	USID	이름 ↕	부서	팀	담당업무	
☐	1	041022	박기영	24	40	1001	dfghfg1
☐	2	10	김사장	11	18	1002	
☐	3	101058	권신혜	23	39	1001	sdf@sb
☐	4	101111	김정현B	23	39	1001	sdfsf@
☐	5	11111111	농협은행01	4	7	1001	
☐	6	161045	송정하	26	41	1001	lermad
☐	7	161064	박창욱	24	40	1001	chadsf
☐	8	171092	배담영	25	41	1001	dammy
☐	9	20011004	이x승	21	30	1001	xxx@b
☐	10	20011005	유x진	21	30	1003	xx1@b
☐	11	20011008	김x정	21	30	1001	xx2@b
☐	12	22222222	농협은행02	4	7	3339	
☐	13	27	27	1	1	1003	mskim
☐	14	33333333	농협은행03	4	7	3334	
☐	15	abcuser	r	1	1	3339	
☐	16	admin	어드민	1	1	1004	sdfd