Spring-boot thymeleaf

강병준

Thymeleaf

- ✓ 화면 출력을 위한 템플릿 엔진 중의 하나
- ✓ 장점
  - 데이터를 JSP와 유사하게 ${}를 이용해서 출력
  - Model에 담긴 객체를 화면에서 JavaScript로 처리하기 편리
  - 연산이나 포맷과 관련된 기능을 추가적인 개발없이 지원
  - natural templates - html 파일로 출력 가능한데 서버 사이드 랜더링을 하지 않고 브라우저에 띄워도 정상적인 화면을 출력을 할 수 있음
- ✓ https://www.thymeleaf.org/
- ✓ Request, HttpSession, Application 에 저장된 데이터를 태그에 출력하고자 할 때는 태그 안에 th:text 속성을 추가한 후 데이터를 EL 형태로 설정하면 됨

Spring Boot Devtools 의 주요 기능

- ✓ Automatic Restart: classpath에 있는 파일이 변경될 때마다 애플리케이션을 자동으로 재시 작해서 개발자가 소스 수정 후 애플리케이션을 재실행하는 과정을 줄일 수 있으므로 생산성을 향상시킬 수 있음
- ✓ Live Reload: 정적 자원(html, css, js) 수정 시 새로 고침 없이 바로 적용
- ✓ Property Defaults: Thymeleaf는 기본적으로 성능을 향상시키기 위해서 캐싱 기능을 사용하는데 개발하는 과정에서 캐싱 기능을 사용한다면 수정한 소스가 제대로 반영되지 않을 수 있기 때문에 cache의 기본값을 false로 설정할 수 있음

❖ Spring Boot Devtools 의 주요 기능

- ✓ Live Reload 적용
  - application.properties 파일에 설정

```
#Live Reload 기능 활성화
spring.devtools.livereload.enabled=true
```

Thymeleaf

- Thymeleaf 프로젝트
  - ✓ application.properties 파일 수정

```
#spring.mvc.view.prefix=/WEB-INF/views/
#spring.mvc.view.suffix=.jsp

#spring.thymeleaf.prefix=classpath:/templates/
#spring.thymeleaf.suffix=.html
#spring.thymeleaf.cache=false

#spring.thymeleaf.view-names=thymeleaf/*



spring.thymeleaf.cache=false

spring.devtools.livereload.enabled=true
```

Thymeleaf

❖ Thymeleaf 프로젝트

✓ 프로젝트에 Thymeleaf 라는 템플릿의 의존성을 설정해야 함

✓ PageController 클래스에 작성

```
@GetMapping("/ex1")
public String ex1(){
        return “ex1”;
}
```

Thymeleaf

- Thymeleaf 프로젝트
  - ✓ src/main/resources/templates 디렉토리에 ex1.html 파일을 만들고 작성 – 기존의 속성 앞에 th:를 붙이고 사용

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>ex1</title>
</head>
<body>
    <h1 th:text="${'Hello Thymeleaf'}"></h1>
</body>
</html>
```

Thymeleaf

- Thymeleaf 프로젝트
  - ✓ 실행 후 브라우저에 localhost:8080/ex1 에 입력하고 확인

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ 속성이 아닌 곳에서의 데이터 출력

[[${데이터}]]

✓ 반복문

th:each = "변수: ${목록} "

- 반복문을 사용하면 state 객체가 같이 생성되는데 이를 이용하면 순번이나 인덱스 번호, 카운트, 홀수/짝수 등을 지정할 수 있음

✓ 분기문

- th:if ~ unless 를 이용하면 조건문을 사용할 수 있는데 th:if 와 th:unless는 별도로 작성 가능
- th:switch 와 th:case 사용 가능
- 삼항 연산자 사용이 가능한데 마지막 항은 생략이 가능

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ templates 디렉토리에 main.html 파일을 생성하고 작성

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        table, tr, td, th {
            border: 1px solid #444444;
        }
    </style>
</head>
```

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ templates 디렉토리에 main.html 파일을 생성하고 작성

```
<div>
    <table>
        <tr th:each="task : ${list}">
            <td>[[${task}]]</td>
        </tr>
    </table>
</div>

</body>
</html>
```

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ template 디렉토리에 ex2.html 파일을 생성하고 작성

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>List</title>
</head>
<body>
    <ul>
        <li th:each="vo : ${list}">
            [[${vo}]]
        </li>
    </ul>
</body>
</html>
```

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ template 디렉토리에 ex2.html 파일을 수정

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>List</title>
</head>
<body>
    <ul>
        <li th:each="vo, state : ${list}">
            [[${state.index}]] --- [[${vo}]]
        </li>
    </ul>
</body>
</html>
```

Thymeleaf

- Thymeleaf 데이터 출력
  - ✓ template 디렉토리에 ex2.html 파일을 수정

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>List</title>
</head>
<body>
    <ul>
        <li th:each="vo, state : ${list}" th:if="${vo.sno % 5 == 0}">
            [[${state.index}]] --- [[${vo}]]
        </li>
    </ul>
</body>
</html>
```

Thymeleaf

- ✓ th:block
  - th:block은 별도의 태그가 필요하지 않기 때문에 반드시 태그에 붙어서 th:text나 th:value 등을 써야 하는 제약이 없음

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>List</title>
    <style>
        .target{
            background-color: red;
        }
    </style>
</head>
<body>
    <ul>
        <th:block th:each="vo:${list}">
            <li th:text="${vo.sno % 5 == 0}?${vo.sno}:${vo.first}"></li>
        </th:block>
    </ul>
</body>
</html>
```

Thymeleaf

- Thymeleaf 데이터 출력
  - ✓ th:block
    - ex2.html 파일 수정

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>List</title>
</head>
<body>
<ul>
    <li th:each="vo, state : ${list}">
        <th:block th:switch="${vo.sno % 5 == 0}">
            <span th:case=true th:text="${'-------------------------' +
  vo.sno}"></span>
            <span th:case=false th:text="${vo.first}"></span>
        </th:block>
    </li>
</ul>
</body></html>
```

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ inline

● PageController 클래스에 새로운 요청을 처리하는 메서드를 생성

```
@GetMapping({"/inline"})
public String exInline(RedirectAttributes redirectAttributes){
        SampleVO vo =
    SampleVO.builder().sno(100L).first("First..100").last("Last..100").regTi
    me(LocalDateTime.now()).build();
        redirectAttributes.addFlashAttribute("result", "success");
        redirectAttributes.addFlashAttribute("vo", vo);
        return "redirect:/ex3";
}

@GetMapping("/ex3")
public void ex3(){}
```

Thymeleaf

❖ Thymeleaf 데이터 출력

✓ 링크 처리

- href 속성에 @{ }를 이용해서 설정
- 파라미터를 전달할 때는 (파라미터이름=${출력할 데이터})
- path 로 전달하고자 하는 경우에는 /뒤에 {임시변수}(임시변수 = ${데이터} 형태로 작성

Thymeleaf

- Thymeleaf 데이터 출력
  - ✓ 링크 처리
    - template 디렉토리에 exlink.html 파일을 생성하고 작성

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8">
    <title>Link</title>
</head>
<body>
<ul>
    <li th:each="vo:${list}">
        <a th:href="@{/exview}">[[${vo.first}]]</a>
        <a th:href="@{/exview(sno=${vo.sno})}">[[${vo.first}]]</a>
        <a th:href="@{/exview/{sno}(sno =
    ${vo.sno})}">[[${vo.first}]]</a>
    </li>
</ul>
</body></html>
```

Thymeleaf

- Thymeleaf 데이터 포맷
  - ✓ 데이터의 포맷 설정
    - 숫자의 경우는 #numbers를 이용해서 숫자의 포맷을 설정
    - 날짜의 경우는 #temporals를 이용해서 날짜의 포맷을 설정하는데 build.gradle 파일에 아래 의존성이 추가되어야 함

```
implementation group: 'org.thymeleaf.extras', name: 'thymeleaf-extras-java8time'
```

```
<!DOCTYPE html><html lang="en" xmlns:th="http://www.thymeleaf.org">
    <head>  <meta charset="UTF-8">  <title>List</title>
  <style>
     .target{
        background-color: red;
     }
  </style></head><body>
    <ul>
        <li th:each="vo : ${list}">
            [[${#numbers.formatInteger(vo.sno, 5)}]] --- [[${#temporals.format(vo.regTime, 'yyyy/MM/dd')}]]
        </li>
          </ul>
</body>
</html>
```

Thymeleaf

❖ Thymeleaf 데이터 포맷

✓ 데이터의 포맷 설정

● 프로젝트를 실행하고 exformat 라고 입력하고 확인

- 00000 --- 2022/01/10
- 00001 --- 2022/01/10
- 00002 --- 2022/01/10
- 00003 --- 2022/01/10
- 00004 --- 2022/01/10
- 00005 --- 2022/01/10
- 00006 --- 2022/01/10
- 00007 --- 2022/01/10
- 00008 --- 2022/01/10
- 00009 --- 2022/01/10

Thymeleaf에서 include

```
<header>
    <link rel="stylesheet" th:href="@{/css/bootstrap.min.css}">
    <script type="text/javascript" th:src="@{/js/jquery.js}"></script>
    <script type="text/javascript" th:src="@{/js/bootstrap.min.js}"></script>
    <style type="text/css">
    .err { color: red; font-weight: bold; }
    </style>
</header>
<span th:replace="/dept/header.html::header"></span>
<span th:insert="/dept/header.html::header"></span>
    <script type="text/javascript">
    $(function() {
        $('#deptList').load("/dept/deptList.do");
    });
    </script>
```

✓ 레이아웃 설정 방법

- JSP의 include 와 같이 특정 부분을 외부 혹은 내부에서 가져와서 포함하는 형태
- 특정한 부분을 파라미터로 전달해서 내용에 포함하는 형태

✓ include 방식의 처리

- 특정한 부분을 다른 내용으로 변경할 수 있는 th:insert나 th:replace 를 이용
- th:replace를 이용하는 경우에는 기존의 내용을 완전히 대체하는 방식
- th:insert의 경우에는 내용의 바깥쪽 태그는 그대로 유지하면서 추가되는 방식

th:replace & th:insert

fragment와 함께 쓰이며, 각 화면에 분리해 놓은 fragment를 붙여넣을 때 사용한다.

th:replace는 태그 전체를 교체해주는 것이다.
(아래 예시 경우, head 태그 자체가 fragments.html의 head로 바뀐다.)

```
// index.html
<head th:replace="fragments.html :: head"></head>
```

th:insert는 해당 태그 내부에 fragment를 삽입해주는 것이다.(아래 예시 경우, div 태그 내부에 fragments.html의 content가 삽입된다.)

```
// index.html
<div th:insert="fragments.html :: content">

</div>
```

Thymeleaf

- Thymeleaf 레이아웃
  - ✓ include 방식의 처리
    - template 디렉토리에 레이아웃에 사용될 파일들을 저장할 fragments 라는 디렉토리를 생성
    - fragments 디렉토리에 fragment1.html 파일을 생성하고 작성

```
<!DOCTYPE html>
<html lang="en"   xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<div th:fragment="part1">
  <h2>Part 1</h2>
</div>
<div th:fragment="part2">
  <h2>Part 2</h2>
</div>
<div th:fragment="part3">
  <h2>Part 3</h2>
</div>
</body></html>
```

Thymeleaf

- Thymeleaf 레이아웃
  - ✓ include 방식의 처리
    - templates 디렉토리에 exlayout1.html 파일을 생성하고 작성

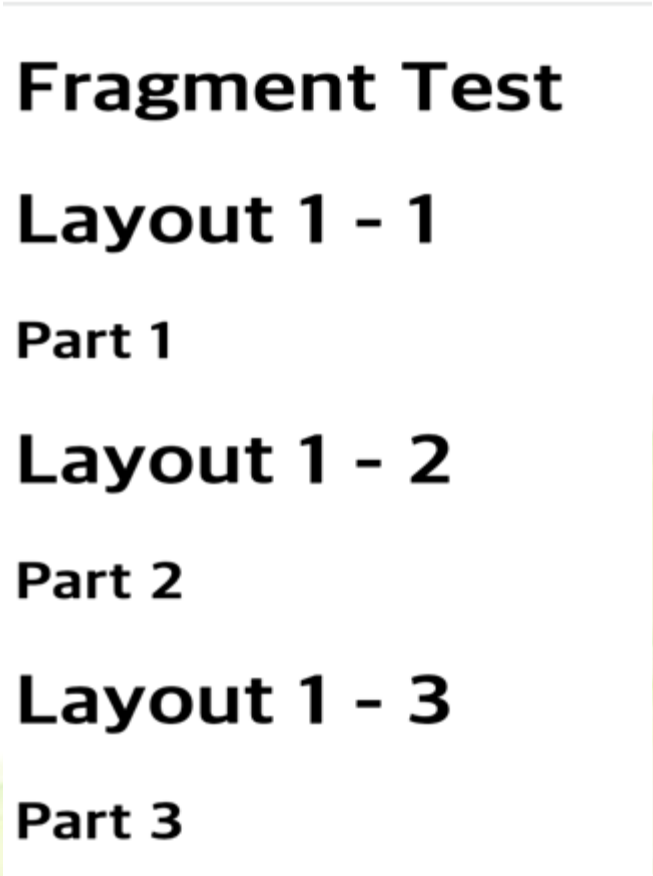
```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>
<h1>Fragment Test</h1>

<h1>Layout 1 - 1</h1>
<div th:replace="~{/fragments/fragment1 :: part1}" ></div>

<h1>Layout 1 - 2</h1>
<div th:replace="~{/fragments/fragment1 :: part2}" ></div>

<h1>Layout 1 - 3</h1>
<th:block th:replace="~{/fragments/fragment1 :: part3}" ></th:block>

</body>
</html>
```

Thymeleaf

❖ Thymeleaf 레이아웃

✓ include 방식의 처리

● 애플리케이션을 실행하고 브라우저에 exlayout1 을 입력하고 확인



Thymeleaf

❖ Thymeleaf 레이아웃

✓ include 방식의 처리

● fragments 디렉토리에 fragment2.html 파일을 생성하고 작성

```
<div>
    <hr/>
    <h2>Fragment2 File</h2>
    <h2>Fragment2 File</h2>
    <h2>Fragment2 File</h2>
    <hr/>
</div>
```

Thymeleaf

- Thymeleaf 레이아웃
  - ✓ include 방식의 처리
    - templates 디렉토리에 exlayout1.html 파일에 내용을 추가

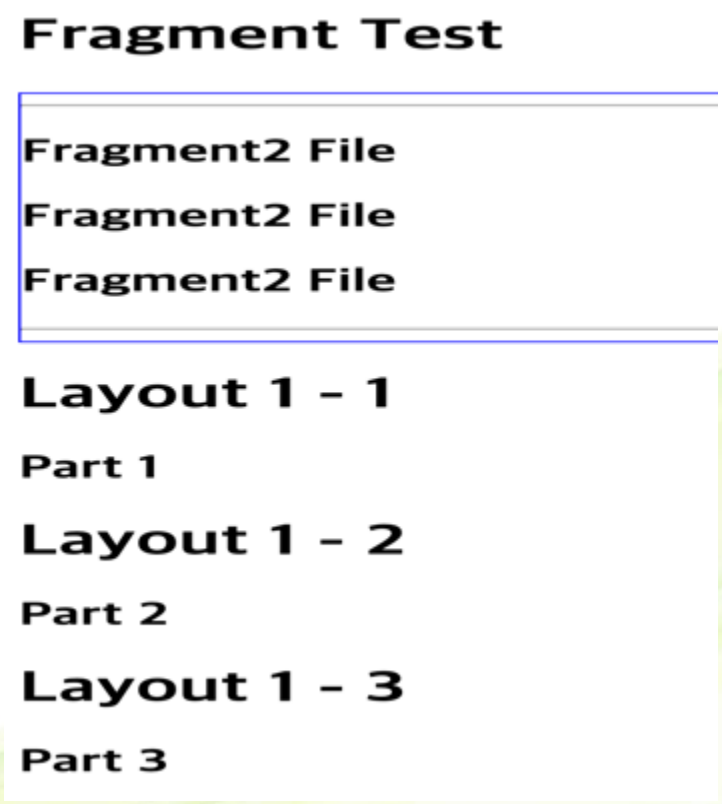
```
<div style="border: 1px solid blue">
  <th:block th:replace="~{/fragments/fragment2}"></th:block>
</div>
```

Thymeleaf

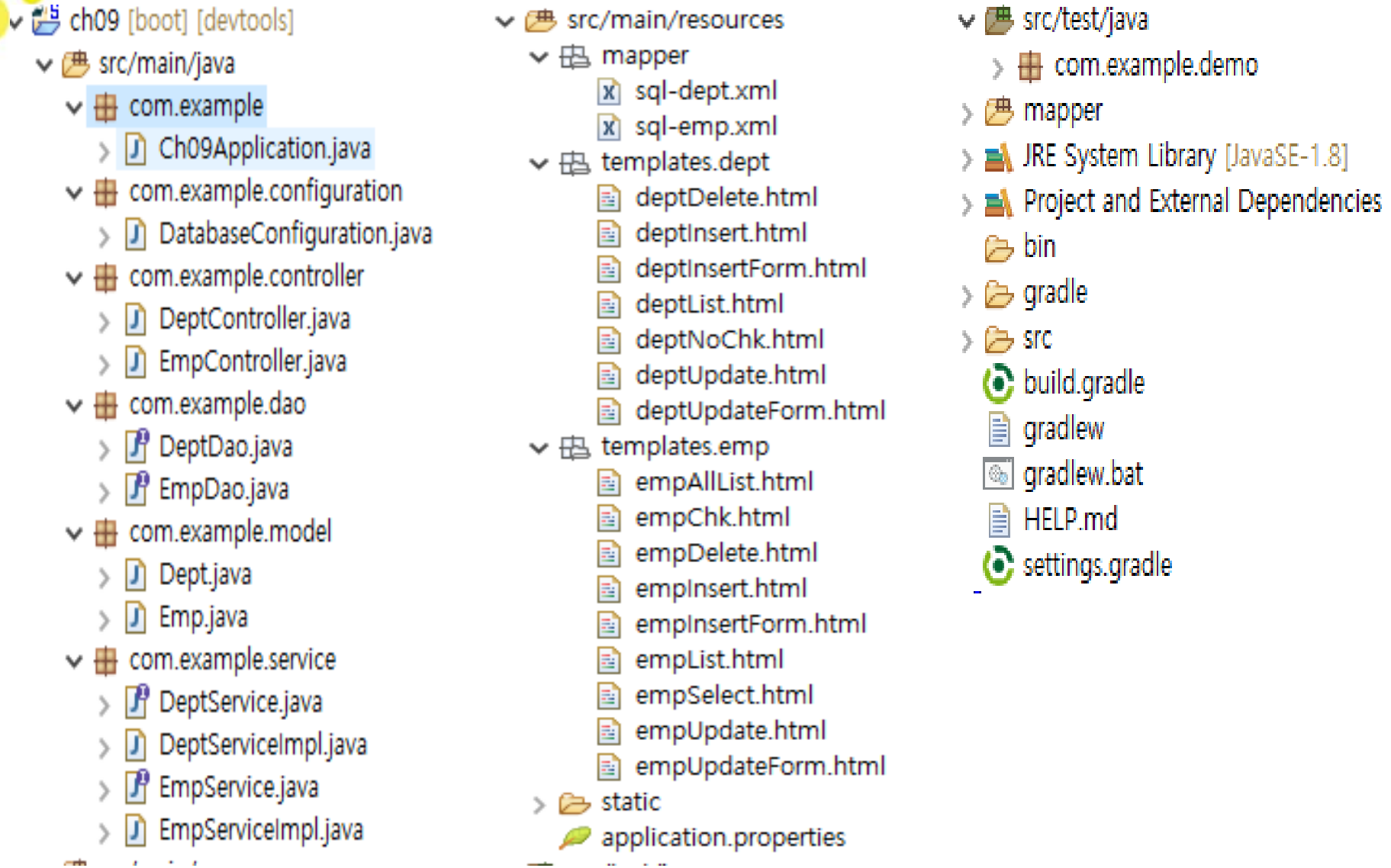
❖ Thymeleaf 레이아웃

✓ include 방식의 처리

● 애플리케이션을 실행하고 브라우저에 exlayout1 을 입력하고 확인



오라클 thymeleaf사용



http://download.eclipse.org/releases/2020-06 (버전은 변경)

application.properties

```
spring.datasource.hikari.driver-class-name=oracle.jdbc.driver.OracleDriver

spring.datasource.hikari.jdbc-url=jdbc:oracle:thin:@127.0.0.1:1521:xe

spring.datasource.hikari.username=scott

spring.datasource.hikari.password=tiger


mybatis.configuration.map-underscore-to-camel-case=true

spring.devtools.livereload.enabled=true

spring.thymeleaf.cache=false

spring.devtools.restart.enabled=false
```

build.gradle

```
plugins {
	id 'org.springframework.boot' version '2.2.6.RELEASE'
	id 'io.spring.dependency-management' version '1.0.9.RELEASE'
	id 'java'
}
group = 'com.example'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '1.8'
configurations {
	developmentOnly
	runtimeClasspath {
		extendsFrom developmentOnly
	}
	compileOnly {
		extendsFrom annotationProcessor
	}
}
repositories {
	mavenCentral()
}
```

build.gradle

```
dependencies {
	implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
	implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
	implementation 'org.springframework.boot:spring-boot-starter-web'
	implementation 'org.mybatis.spring.boot:mybatis-spring-boot-starter:2.1.2'
	implementation 'org.springframework.boot:spring-boot-devtools'
	compileOnly 'org.projectlombok:lombok'
	developmentOnly 'org.springframework.boot:spring-boot-devtools'
	runtimeOnly 'com.oracle.ojdbc:ojdbc8'
	runtimeOnly 'mysql:mysql-connector-java'
	annotationProcessor 'org.springframework.boot:spring-boot-configuration-
processor'
	annotationProcessor 'org.projectlombok:lombok'
	testImplementation('org.springframework.boot:spring-boot-starter-test') {
		exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
	}
}
test {
	useJUnitPlatform()
}
```

Application.java

```
package com.example;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class Ch09Application {

        public static void main(String[] args) {

                SpringApplication.run(Ch09Application.class, args);

        }

}
```

DatabaseConfiguration.java

```
import org.apache.ibatis.session.SqlSessionFactory;
import org.mybatis.spring.SqlSessionFactoryBean;
import org.mybatis.spring.SqlSessionTemplate;
import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.PropertySource;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;
@Configuration
@PropertySource("classpath:/application.properties")
public class DatabaseConfiguration {
        @Autowired
        private ApplicationContext applicationContext;
        @Bean
        @ConfigurationProperties(prefix="spring.datasource.hikari")
        public HikariConfig hikariConfig() {
                return new HikariConfig();
        }
        @Bean
        @ConfigurationProperties(prefix="mybatis.configuration")
        public org.apache.ibatis.session.Configuration mybatisConfig(){
                return new org.apache.ibatis.session.Configuration();
        }
```

DatabaseConfiguration.java

```
	@Bean
	public DataSource dataSource() throws Exception{
		DataSource dataSource = new HikariDataSource(hikariConfig());
		return dataSource;
	}
	@Bean
	public SqlSessionFactory sqlSessionFactory(DataSource dataSource) throws
Exception{
		SqlSessionFactoryBean sqlSessionFactoryBean = new
SqlSessionFactoryBean();
		sqlSessionFactoryBean.setDataSource(dataSource);

	sqlSessionFactoryBean.setMapperLocations(applicationContext.getResources("classp
ath:/mapper/**/sql-*.xml"));
		sqlSessionFactoryBean.setConfiguration(mybatisConfig());
		return sqlSessionFactoryBean.getObject();
	}
	@Bean
	public SqlSessionTemplate sqlSessionTemplate(SqlSessionFactory
sqlSessionFactory){
		return new SqlSessionTemplate(sqlSessionFactory);
	}
}
```

DeptController.java

```
package com.example.controller;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import com.example.model.Dept;
import com.example.service.DeptService;
@Controller
public class DeptController {
		@Autowired
		private DeptService ds;
		@RequestMapping("/dept/deptList.do")
		public String deptList(Model model) {
				List<Dept> list = ds.selectDeptList();
				model.addAttribute("list", list);
				return "/dept/deptList";
		}
		@RequestMapping("/dept/deptInsertForm.do")
		public String deptInsertForm() {
				return "/dept/deptInsertForm";
		}
```

DeptController.java

```
@RequestMapping("/dept/deptInsert.do")
public String deptInsert(Dept dept, Model model) {
		Dept dt = ds.selectDept(dept.getDeptno());
		int result = 0;
		if (dt == null)
			result = ds.deptInsert(dept);
		else
			result = -1;
		model.addAttribute("result", result);
		return "/dept/deptInsert";
}
@RequestMapping("/dept/deptUpdateForm.do")
public String deptUpdateForm(int deptno, Model model) {
		Dept dept = ds.selectDept(deptno);
		model.addAttribute("dept", dept);
		return "/dept/deptUpdateForm";
}
@RequestMapping("/dept/deptUpdate.do")
public String deptUpdate(Dept dept, Model model) {
		int result = ds.deptUpdate(dept);
		model.addAttribute("result", result);
		return "/dept/deptUpdate";
}
```

DeptController.java

```
	@RequestMapping("/dept/deptDelete.do")
	public String deptDelete(int deptno, Model model) {
		int result = ds.deptDelete(deptno);
		model.addAttribute("result", result);
		return "/dept/deptDelete";
	}
//	@RequestMapping("/dept/deptNoChk.do" )
//	public String deptNoChk(int deptno,	Model model) {
//		String msg = "";
//		Dept dept = ds.selectDept(deptno);
//		if (dept == null) msg = "사용가능합니다";
//		else msg="다른 부서코드를 사용하세요";
//		model.addAttribute("msg", msg);
//		return "/dept/deptNoChk";
//	}
	@RequestMapping(value="/dept/deptNoChk.do", produces="text/html;charset=utf-8")
	 @ResponseBody public String deptNoChk(int deptno) {
		String msg = ""; Dept
		dept = ds.selectDept(deptno);
		if (dept == null) msg = "사용가능합니다";
		else msg="다른 부서코드를 사용하세요";
		return msg;
	 }
}
```

EmpController.java

```
import java.util.Collection;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;
import com.example.model.Dept;
import com.example.model.Emp;
import com.example.service.DeptService;
import com.example.service.EmpService;
@Controller
public class EmpController {
	@Autowired
	private EmpService es;
	@Autowired
	private DeptService ds;
	@RequestMapping("/emp/empList.do")
	public String empList(int deptno, Model model) {
		Dept dept = ds.selectDept(deptno);
		Collection<Emp> empList = es.selectEmplist(deptno);
		model.addAttribute("dept", dept);
		model.addAttribute("empList", empList);
		return "emp/empList";
	}
```

EmpController.java

```
@RequestMapping("/emp/empSelect.do")
public String empSelect(int empno, Model model) {
        Emp emp = es.select(empno);  model.addAttribute("emp", emp);
        return "/emp/empSelect";
}
@RequestMapping("/emp/empInsertForm.do")
public String empInsertForm(int deptno, Model model) {
        Collection<Emp> empList = es.selectEmplist(0);
        Collection<Dept> deptList = ds.selectDeptList();
        model.addAttribute("deptno",   deptno);
        model.addAttribute("empList",  empList);
        model.addAttribute("deptList", deptList);
        return "/emp/empInsertForm";
}
@RequestMapping("/emp/empInsert.do")
public String empInsert(Emp emp, Model model) {
        int result = 0;
        Emp emp2 = es.select(emp.getEmpno());
        if (emp2 == null) result = es.insert(emp);
        else result = -1;
        model.addAttribute("result", result);
        model.addAttribute("emp", emp);
        return "/emp/empInsert";
}
```

EmpController.java

```
@RequestMapping("/emp/empUpdateForm.do")
public String empUpdateForm(int empno, Model model) {
        Emp emp = es.select(empno);
        Collection<Emp> empList = es.selectEmplist(0);
        Collection<Dept> deptList = ds.selectDeptList();
        model.addAttribute("emp", emp);
        model.addAttribute("empList",  empList);
        model.addAttribute("deptList", deptList);
        return "/emp/empUpdateForm";
}
@RequestMapping("/emp/empUpdate.do")
public String empUpdate(Emp emp, Model model) {
        int result = es.update(emp);     model.addAttribute("result", result);
        model.addAttribute("emp", emp);
        return "/emp/empUpdate";
}
@RequestMapping("/emp/empDelete.do")
public String empDelete(int empno, Model model) {
        Emp emp = es.select(empno);
        int result = es.delete(empno);
        model.addAttribute("result", result);
        model.addAttribute("emp", emp);
        return "/emp/empDelete";
}
```

EmpController.java

```
	@RequestMapping("/emp/empAllList.do")
	public String empAllList(Model model) {
		Collection<Emp> list = es.empAllList();
		model.addAttribute("list", list);
		return "/emp/empAllList";
	}
//	@RequestMapping("/emp/empChk.do")
//	public String empChk(int empno, Model model) {
//		String msg = "";
//		Emp emp =		es.select(empno);
//		if (emp == null) msg = "사용가능한 사번입니다";
//		else msg = "이미 사용중인니 다른 사번을 사용하시오";
//		model.addAttribute("msg", msg);
//		return "/emp/empChk";
//	}
	 @RequestMapping(value="/emp/empChk.do", produces="text/html;charset=utf-8")
	 @ResponseBody
	 public String empChk(int empno) {
		String msg = "";			Emp emp = es.select(empno);
		if (emp == null) msg = "사용가능한 사번입니다";
		else msg = "이미 사용중인니 다른 사번을 사용하시오";
		return msg;
	}
}
```

DeptDao.java

```
package com.example.dao;
import java.util.List;
import org.apache.ibatis.annotations.Mapper;
import com.example.model.Dept;
@Mapper
public interface DeptDao {
        List<Dept> selectDeptList();
        Dept selectDept(int deptno);
        int deptInsert(Dept dept);
        int deptUpdate(Dept dept);
        int deptDelete(int deptno);

}
```

EmpDao.java

```
package com.example.dao;
import java.util.Collection;
import org.apache.ibatis.annotations.Mapper;
import com.example.model.Emp;
@Mapper
public interface EmpDao {
        Collection<Emp> selectEmplist(int deptno);
        Emp select(int empno);
        int insert(Emp emp);
        int update(Emp emp);
        int delete(int empno);
        Collection<Emp> empAllList();

}
```

model

```
import lombok.Data;
@Data
public class Dept {
		private int deptno;
		private String dname;
		private String loc;
}

import java.sql.Date;
@Data
public class Emp {
		private int empno;
		private String ename;
		private String job;
		private int mgr;
		private Date hiredate;
		private int sal;
		private int comm;
		private int deptno;
		// join
		private String dname;
		private String loc;
}
```

Service

```
import java.util.List;
import com.example.model.Dept;
public interface DeptService {
        List<Dept> selectDeptList();
        Dept selectDept(int deptno);
        int deptInsert(Dept dept);
        int deptUpdate(Dept dept);
        int deptDelete(int deptno);
}

package com.example.service;
import java.util.Collection;
import com.example.model.Emp;
public interface EmpService {
        Collection<Emp> selectEmplist(int deptno);
        Emp select(int empno);
        int insert(Emp emp);
        int update(Emp emp);
        int delete(int empno);
        Collection<Emp> empAllList();
}
```

DeptServiceImpl.java

```
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.example.dao.DeptDao;
import com.example.model.Dept;
@Service
public class DeptServiceImpl implements DeptService {
		@Autowired
		private DeptDao dd;
		public List<Dept> selectDeptList() {		return dd.selectDeptList();	}
		public Dept selectDept(int deptno) {
			return dd.selectDept(deptno);
		}
		public int deptInsert(Dept dept) {
			return dd.deptInsert(dept);
		}
		public int deptUpdate(Dept dept) {
			return dd.deptUpdate(dept);
		}
		public int deptDelete(int deptno) {
			return dd.deptDelete(deptno);
		}
}
```

EmpServiceImpl.java

```
import java.util.Collection;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
@Service
public class EmpServiceImpl implements EmpService {
		@Autowired
		private EmpDao ed;
		public Collection<Emp> selectEmplist(int deptno) {
			return ed.selectEmplist(deptno);
		}
		public Emp select(int empno) {		return ed.select(empno);	}
		public int insert(Emp emp) {
			return ed.insert(emp);
		}
		public int update(Emp emp) {
			return ed.update(emp);
		}
		public int delete(int empno) {
			return ed.delete(empno);
		}
		public Collection<Emp> empAllList() {
			return ed.empAllList();
		}
}
```

resources/mapper/ sql-dept.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.example.dao.DeptDao">
        <select id="selectDeptList" resultType="com.example.model.Dept">
                select * from dept order by deptno
        </select>
        <select id="selectDept" parameterType="int" resultType="com.example.model.Dept">
                select * from dept where deptno=#{deptno}
        </select>
        <insert id="deptInsert" parameterType="com.example.model.Dept">
                insert into dept values (#{deptno},#{dname},#{loc})
        </insert>
        <update id="deptUpdate" parameterType="com.example.model.Dept">
                update dept set dname=#{dname}, loc=#{loc}
                        where deptno=#{deptno}
        </update>
        <delete id="deptDelete" parameterType="int">
                delete from dept where deptno=#{deptno}
        </delete>
</mapper>
```

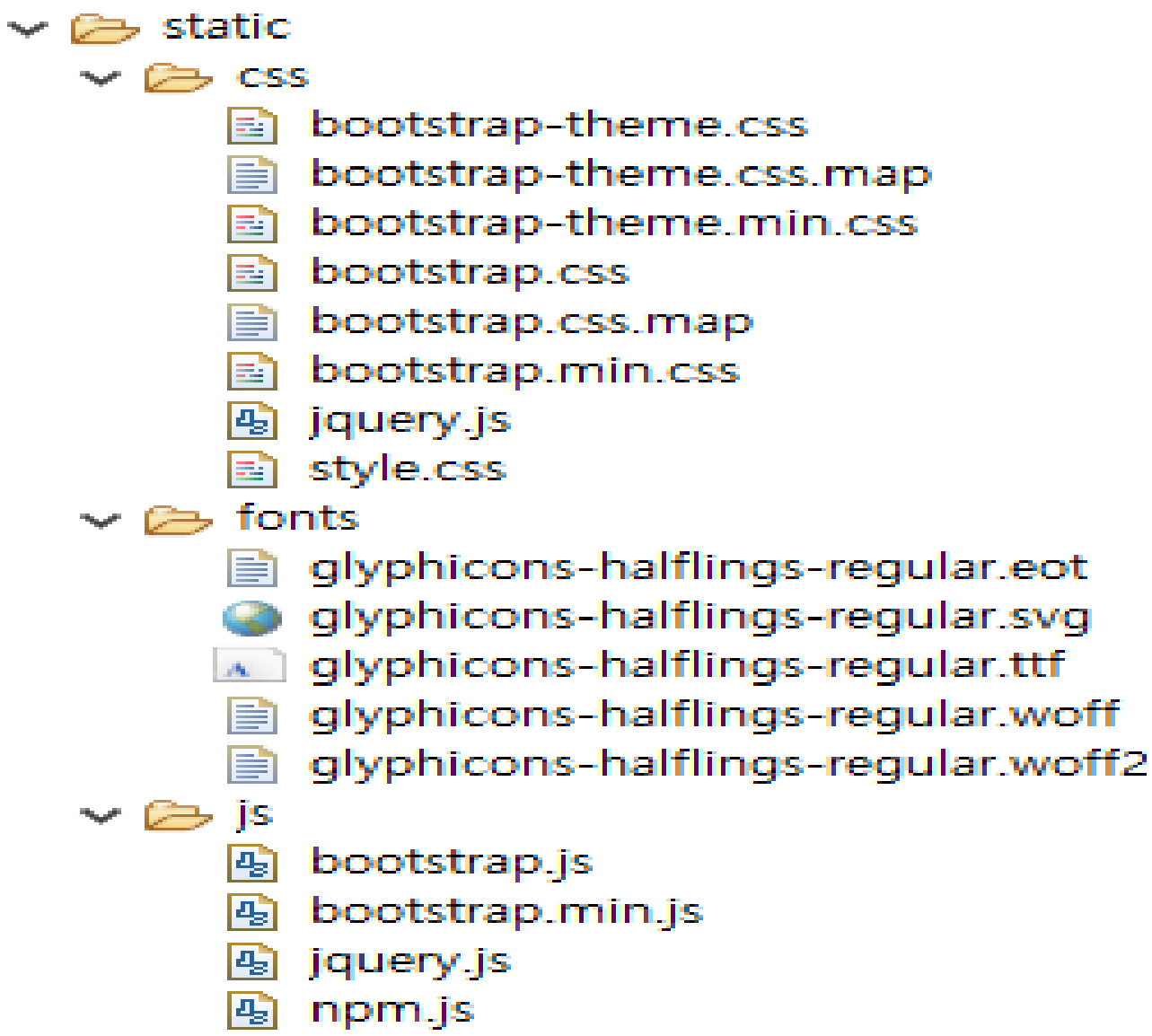
resources/mapper/ sql-emp.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.example.dao.EmpDao">
        <select id="selectEmplist" parameterType="hashMap"
resultType="com.example.model.Emp">
                select * from emp
                <if test="deptno!=0"> where deptno=#{deptno}      </if>
                order by empno
        </select>
        <select id="empAllList" resultType="com.example.model.Emp">
                select e.*,dname,loc from emp e, dept d
                        where e.deptno=d.deptno order by empno
        </select>
        <select id="select" parameterType="int" resultType="com.example.model.Emp">
                select * from emp where empno=#{empno}
        </select>
        <insert id="insert" parameterType="com.example.model.Emp">
                insert into emp values (#{empno}, #{ename},#{job},#{mgr},
                        #{hiredate}, #{sal}, #{comm}, #{deptno})
        </insert>
```

resources/mapper/sql-emp.xml

```
        <update id="update"  parameterType="com.example.model.Emp">
                update emp set ename=#{ename}, job=#{job}, mgr=#{mgr},
                        hiredate=#{hiredate}, sal=#{sal}, comm=#{comm},
                        deptno=#{deptno} where empno=#{empno}
        </update>
        <delete id="delete" parameterType="int">
                delete from emp where empno=#{empno}
        </delete>
</mapper>
```

resources/static/



resources/templates/dept/deptDelete

```
<!DOCTYPE html><html lang="ko"
xmlns:th="http://www.thymeleaf.org"><head><meta charset="UTF-8">
<title>Insert title here</title></head><body>
<th:if test="${result > 0  }">
		<script type="text/javascript">
			alert("삭제되었습니다");
			location.href="/dept/deptList.do";
		</script>
</th:if>
<th:if test="${result == 0  }">
		<script type="text/javascript">
			alert("삭제 실패");
			history.go(-1);
		</script>
</th:if>
</body>
</html>
```

resources/templates/dept/deptInsertForm

```
<!DOCTYPE html><html lang="ko"
xmlns:th="http://www.thymeleaf.org"><head><meta charset="UTF-8">
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style>
<title>Insert title here</title>
<script type="text/javascript">
		function deptNoChk() {
			if (!frm.deptno.value) {
				alert("부서코드 입력후에 체크하시오");
				frm.deptno.focus();			return false;		}
			$.post("/dept/deptNoChk.do", "deptno="+frm.deptno.value,
					function(data) {
				$('#disp').html(data);
			});
		}
</script></head><body>
```

resources/templates/dept/deptInsertForm

```
<div class="container" align="center">
        <h2 class="text-primary">부서정보 입력</h2>
<form action="/dept/deptInsert.do" method="post" name="frm">
<table class="table table-bordered">
        <tr><td>부서코드</td><td><input type="number" name="deptno"
                required="required" autofocus="autofocus">
                <input type="button" value="중복체크" onclick="deptNoChk()"
                        class="btn btn-success btn-sm">
                        <div id="disp" class="err"></div></td></tr>
        <tr><td>부서명</td><td><input type="text" name="dname"
                required="required"></td></tr>
        <tr><td>근무지</td><td><input type="text" name="loc"
                required="required"></td></tr>
        <tr><td colspan="2"><input type="submit" value="확인"></td></tr>
</table>
</form>
<a class="btn btn-info" href="deptList.do">부서목록</a>
</div>
</body>
</html>
```

resources/templates/dept/deptInsert

```
<!DOCTYPE html><html lang="ko" xmlns:th="http://www.thymeleaf.org"><head><meta
charset="UTF-8">
<title>Insert title here</title></head><body>
<th:if test="${result > 0 }">
		<script type="text/javascript">
			alert("입력 성공");
			location.href="deptList.do";
		</script>
</th:if>
<th:if test="${result == 0 }">
		<script type="text/javascript">
			alert("에쿵 실패네");
			history.go(-1);
		</script>
</th:if>
<th:if test="${result == -1 }">
		<script type="text/javascript">
			alert("요놈 !! 중복됐다는 왜 입력해");
			history.go(-1);
		</script>
</th:if>
</body>
</html>
```

resources/templates/dept/deptList

```
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style></head><body>
<div class="container" align="center">    <h2 class="text-primary">부서 목록</h2>
<table class="table table-hover"><thead>
            <tr><td>부서코드</td><td>부서명</td><td>근무지</td>
                        <td>수정 삭제</td></tr></thead> <tbody>
<tr th:if="${#lists.size(list)} > 0" th:each="dept : ${list}">
            <td th:text="${dept.deptno}"></td>
            <td class="btn btn-info btn-sm">
                        <a href="/emp/empList.do?deptno=" th:attrappend="href=${dept.deptno}"
            th:text="${dept.dname}"></a></td>
            <td th:text="${dept.loc}"></td>
            <td class="btn btn-warning btn-sm"><a href="/dept/deptUpdateForm.do?deptno="
th:attrappend="href=${dept.deptno}">수정</a></td>
            <td class="btn btn-danger btn-sm"><a href="/dept/deptDelete.do?deptno="
th:attrappend="href=${dept.deptno}">삭제</a></td>
</tr></tbody></table>
            <a class="btn btn-default" href="/dept/deptInsertForm.do">부서입력</a>
            <a class="btn btn-success" href="/emp/empAllList.do">전직원 조회</a>
</div>
</body></html>
```

resources/templates/dept/updateForm

```
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style>
</head><body>
<div class="container" align="center">
        <h2 class="text-primary">부서정보 수정</h2>
<form action="/dept/deptUpdate.do" method="post" name="frm">
        <input type="hidden" name="deptno" th:value="${dept.deptno}">
<table class="table table-bordered">
        <tr><td>부서코드</td><td th:text="${dept.deptno}"></td></tr>
        <tr><td>부서명</td><td><input type="text" name="dname"
                required="required" autofocus="autofocus"
                th:value="${dept.dname}"></td></tr>
        <tr><td>근무지</td><td><input type="text" name="loc"
                required="required" th:value="${dept.loc}"></td></tr>
        <tr><td colspan="2"><input type="submit" value="확인"></td></tr>
</table>
</form>
<a class="btn btn-info" href="/dept/deptList.do">부서목록</a>
</div>
</body></html>
```

resources/templates/dept/update

```
<!DOCTYPE html><html lang="ko"
xmlns:th="http://www.thymeleaf.org"><head><meta charset="UTF-8">
<title>Insert title here</title></head><body>
<span th:if="${result} > 0“>
        <script type="text/javascript">
                alert("수정성공");
                location.href="/dept/deptList.do";
        </script>
</span>
<span th:if="${result} == 0">
        <script type="text/javascript">
                alert("수정 실패 □□");
                history.go(-1);
        </script>
</span>
</body>
</html>
```

resources/templates/emp/empAllList

```
<!DOCTYPE html><html lang="ko" xmlns:th="http://www.thymeleaf.org"><head><meta
charset="UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style></head><body>
<div class="container" align="center">
            <h2 class="text-primary">전직원 목록</h2>
<table class="table table-bordered">
            <tr><td>사번</td><td>이름</td><td>업무</td><td>입사일</td>
                        <td>급여</td><td>부서명</td><td>근무지</td></tr>
<tr th:if="${#lists.size(list)} > 0" th:each="emp : ${list}">
                        <td th:text="${emp.empno}"></td><td th:text="${emp.ename}"></td>
                        <td th:text="${emp.job}"></td><td th:text="${emp.hiredate}"></td>
                        <td th:text="${emp.sal}"></td><td th:text="${emp.dname}"></td>
                        <td th:text="${emp.loc }"></td>
</tr>
</table>
<a href="/dept/deptList.do" class="btn btn-info">부서목록</a>
</div>
</body>
</html>
```

resources/templates/emp/empChk

```
<!DOCTYPE html>
<html lang="ko" xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<div th:text="${msg}"></div>
</body>
</html>
```

resources/templates/emp/empDelete

```
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style></head><body>
<span th:if="${result} > 0">
        <script type="text/javascript">
                alert("삭제댔다 쭈아쭈아 !!");
                location.href="empList.do?deptno="+[[${emp.deptno}]];
        </script>
</span>
<span th:if"${result} == 0">
        <script type="text/javascript">
                alert("헐 ! 에러 !!");
                history.go(-1);
        </script>
</span>
</body>
</html>
```

resources/templates/emp/empInsertForm

```
<style> .err { color: red; font-weight: bold; }</style>
<script type="text/javascript">
    function empChk() {
        if (!frm.empno.value) { alert("사번 입력후에 체크하시오");
            frm.empno.focus();      return false;                    }
        $.post('/emp/empChk.do',"empno="+frm.empno.value,  function(data) {
            $('#empCk').html(data);
        });
    }
</script></head><body><div class="container" align="center">
        <h2 class="text-primary">직원정보 입력</h2>
<form action="/emp/empInsert.do" method="post" name="frm">
<table class="table table-bordered">
        <tr><td>사번</td><td><input type="number" name="empno"
            required="required" autofocus="autofocus">
            <input type="button" value="중복체크" onclick="empChk()">
            <div class="err" id="empCk"></div></td></tr>
        <tr><td>이름</td><td><input type="text" name="ename"
            required="required"></td></tr>
        <tr><td>업무</td><td><input type="text" name="job"
            required="required"></td></tr>
        <tr><td>관리자사번</td><td><select name="mgr">
            <option th:each="e:${empList}"  th:value="${e.empno}"
                                th:text="${e.ename}+'('+${e.empno}+')'">

                            </option>
        </select>
```

resources/templates/emp/empInsertForm

```
            </td></tr>
            <tr><td>입사일</td><td><input type="date" name="hiredate"
                    required="required"></td>
            <tr><td>급여</td><td><input type="number" name="sal"
                    required="required"></td></tr>
            <tr><td>COMM</td><td><input type="number" name="comm"
                    required="required"></td></tr>
            <tr><td>부서코드</td><td><select name="deptno">
                                    <option th:each="dept:${deptList}"
th:if="${deptno==dept.deptno}" th:value="${dept.deptno}"
                                     selected="selected"
th:text="${dept.dname}+'('+${dept.deptno}+')'"></option>
                    <th:if test="${emp.deptno!=dept.deptno}">
                    <option th:each="dept:${deptList}" th:if="${deptno!=dept.deptno}"
th:value="${dept.deptno}"

th:text="${dept.dname}+'('+${dept.deptno}+')'"></option></select> </td></tr>
            <tr><td colspan="2"><input type="submit" value="확인"></td></tr>
</table>
</form>
</div></body></html>
```

resources/templates/emp/empInsert

```
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style></head><body>
<span th:if="${result} > 0">
        <script type="text/javascript">
                alert("입력 성공 !! 대박");
                location.href = "empSelect.do?empno"+[${emp.empno}]]";
        </script>
</span>
<th:if="${result} == 0">
        <script type="text/javascript">
                alert("입력 실패 !! 쪽박");
                history.go(-1);
        </script>
</span>
<span th:if="${result} == -1">
        <script type="text/javascript">
                alert("있는 사번인데 !! 돌아이 ");
                history.back();
        </script>
</span>
</body></html>
```

resources/templates/emp/empList

```
<script type="text/javascript">
		$(function() {		$('#disp').load('/dept/deptList.do');		});
</script></head><body>
<div class="container" align="center" >
		<h2><span th:text="${dept.dname}"></span> 직원목록</h2>
<table class="table table-striped" >
		<tr><td>사번</td><td>이름</td><td>업무</td><td>입사일</td></tr>
<tr th:if="${#lists.size(empList)} == 0">
		<th colspan="4">직원이 없습니다</th></tr>
</th:if>
<tr th:if="${#lists.size(empList)} > 0" th:each="emp:${empList}">
				<td th:text="${emp.empno}"></td>
				<td><a class="btn btn-sm btn-success"
		href="/emp/empSelect.do?empno=" th:attrappend="href=${emp.empno}"
th:text="${emp.ename}"></a></td>
					<td th:text="${emp.job }"></td><td
th:text="${emp.hiredate }"></td></tr>
</table>
		<a class="btn btn-info" href="/dept/deptList.do">부서목록</a>
		<a class="btn btn-default"
				href="/emp/empInsertForm.do?deptno="
th:attrappend="href=${dept.deptno}">직원정보 입력</a>
		<div id="disp"></div>
</div></body></html>
```

resources/templates/emp/empSelect

```
<script type="text/javascript">
     $(function() {      $('#empDisp').load('empList.do?deptno='+[[${emp.deptno}]]);  });
    function delConfirm() {
            var cf = confirm("정말로 삭제하겠습니까?");
            if (cf) location.href="empDelete.do?empno="+[[${emp.empno}]];
            else alert("삭제 취소 되었습니다");          }
</script></head><body><div class="container" align="center"><h2>직원 상세정보</h2>
<table class="table table-bordered table-striped">
            <tr><td>사번</td><td th:text="${emp.empno}"></td>
                        <td>이름</td><td th:text="${emp.ename}"></td></tr>
            <tr><td>업무</td><td th:text="${emp.job }"></td>
                        <td>관리자사번</td><td th:text="${emp.mgr }"></td></tr>
            <tr><td>입사일</td><td th:text="${emp.hiredate }"></td>
                        <td>급여</td><td th:text="${emp.sal }"></td></tr>
            <tr><td>COMM</td><td th:text="${emp.comm }"></td>
                        <td>부서코드</td><td th:text="${emp.deptno }"></td></tr>
            <tr><td><a href="/emp/empList.do?deptno=" th:attrappend="href=${emp.deptno}"
                                    class="btn btn-info">직원목록</a></td>
        <td><a href="/emp/empUpdateForm.do?empno=" th:attrappend="href=${emp.empno}"
                                    class="btn btn-warning">수정</a></td>
            <td><button onclick="delConfirm() "   class="btn btn-danger">삭제</button></td>
            <td><a href="/dept/deptList.do" class="btn btn-default">부서목록</a></td></tr>
</table>   <div id="empDisp"></div>
</div></body></html>
```

resources/templates/emp/empUpdateForm

```
</head><body>
<div class="container" align="center">
		<h2 class="text-primary">직원정보 수정</h2>
<form action="/emp/empUpdate.do" method="post" name="frm">
		<input type="hidden" name="empno" th:value="${emp.empno}">
<table class="table table-bordered">
		<tr><td>사번</td><td th:text="${emp.empno}"></td></tr>
		<tr><td>이름</td><td><input type="text" name="ename"
			required="required" autofocus="autofocus"
			th:value="${emp.ename}"></td></tr>
		<tr><td>업무</td><td><input type="text" name="job"
			required="required" th:value="${emp.job}"></td></tr>
		<tr><td>관리자사번</td><td><!-- <select name="mgr"> -->
			<select name="mgr">
				<option th:each="e:${empList}"  th:if="${emp.mgr==e.empno}"
th:value="${e.empno}" selected="selected"
				 th:text="${e.ename}+'('+${e.empno}+')'">
				</option>
				<option th:each="e:${empList}"  th:if="${emp.mgr != e.empno}"
th:value="${e.empno}"
		th:text="${e.ename}+'('+${e.empno}+')'">
				</option>
		</select>
		</td></tr>
```

resources/templates/emp/empUpdate

```
<!DOCTYPE html><html lang="ko"
xmlns:th="http://www.thymeleaf.org"><head><meta charset="UTF-8">
<title>Insert title here</title>
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}"/>
<link th:href="@{/css/bootstrap.min.css}" rel="stylesheet">
<script th:src="@{/js/jquery.js}"></script>
<script th:src="@{/js/bootstrap.min.js}"></script>
<style> .err { color: red; font-weight: bold; }</style></head><body>
<th:if test="${result > 0 }">
		<script type="text/javascript">
			alert("수정성공");
			location.href="empSelect.do?empno="+[[${emp.empno}]];
		</script>
</th:if>
<th:if test="${result == 0 }">
		<script type="text/javascript">
			alert("애고고！ 실패네");
			history.go(-1);
		</script>
</th:if>
<body>
```