학생 조회 기능 조합

2021-11-15

이승진

학습목표

학생 목록 조회 페이지에서, 학생 한 명 조회 페이지로 넘어가는 링크를 구현하자.

목차

[1. student/detail 실행 2](#_Toc87855464)

[2. student/test1 3](#_Toc87855465)

[1) StudentController.java 수정 3](#_Toc87855466)

[2) student/test1.jsp 4](#_Toc87855467)

[3) 실행 4](#_Toc87855468)

[4) 절대 주소 URL, 상대 주소 URL 6](#_Toc87855469)

[3. student/list2 7](#_Toc87855470)

[1) StudentController 수정 7](#_Toc87855471)

[2) student/list2.jsp 8](#_Toc87855472)

[4. student/list3 10](#_Toc87855473)

[1) Javascript 10](#_Toc87855474)

[2) location.href 10](#_Toc87855475)

[3) jquery 자바스크립트 라이브러리 11](#_Toc87855476)

[4) StudentController 수정 12](#_Toc87855477)

[5) student/list3.jsp 13](#_Toc87855478)

[6) student/detail.jsp 수정 - 되돌아 가기 기능 15](#_Toc87855479)

[5. student/list0 16](#_Toc87855480)

[1) StudentController 수정 16](#_Toc87855481)

[2) student/list0.jsp 17](#_Toc87855482)

[3) 실행 18](#_Toc87855483)

[6. 연습문제 19](#_Toc87855484)

# student/detail 실행

지난 시간에 구현한 student/detail 액션 메소드를 실행하자.

### http://localhost:8088/student/detail?id=1

Table

Description automatically generated

### http://localhost:8088/student/detail?id=2

Table

Description automatically generated

# student/test1

## StudentController.java 수정

### src/main/net/skhu/controller/StudentController.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | package net.skhu.controller;  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 net.skhu.mapper.StudentMapper;  @Controller  public class StudentController {  @Autowired StudentMapper studentMapper;  @RequestMapping("student/list")  public String list(Model model) {  model.addAttribute("students", studentMapper.findAll());  return "student/list";  }  @RequestMapping("student/list1")  public String list1(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list1";  }  @RequestMapping("student/detail")  public String detail(Model model, Integer id) {  if (id == null) id = 5;  model.addAttribute("student", studentMapper.findById(id));  return "student/detail";  }  @RequestMapping("student/test1")  public String test1(Model model) {  return "student/test1";  }  } |

## student/test1.jsp

src/main/webapp/WEB-INF/views/student/test1.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <style>  div.container { width: 600px; margin: 50px auto; }  a { display: block; margin-top: 20px; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <a href="http://localhost:8088/student/detail?id=1">최하은</a>  <a href="http://localhost:8088/student/detail?id=2">김진영</a>  </div>  </body>  </html> |

## 실행

### http://localhost:8088/student/test1

Graphical user interface, text, application

Description automatically generated

위 화면에서 '최하은' 항목을 클릭하면, 아래와 같은 화면이 출력된다.

Table

Description automatically generated

### 실행과정

(1) <a href="http://localhost:8088/student/detail?id=1">최하은</a> 태그가 클릭되면

웹브라우저는 서버에 이 URL을 요청한다.

(2) 요청하자마자 웹브라우저 주소칸 값은 방금 요청한 URL로 바뀐다.

즉 웹브라우저 주소칸 값은 "http://localhost:8088/student/detail?id=1" 으로 바뀐다.

(3) 서버에서 StudentController의 detail 액션 메소드가 실행된다.

이때 request parameter 에 id=1 값이 들어있다.

(4) student/detail.jsp 뷰 파일이 실행된다.

이 뷰의 실행 결과 출력이 서버로부터 웹브라우저에 전송된다.

전송된 html 태그들이 웹브라우저 창에 출력된다.

## 절대 주소 URL, 상대 주소 URL

Graphical user interface, text, application

Description automatically generated

위 화면에서 웹브라우저 주소칸의 URL은

"http://localhost:8088/student/test1" 이다.

위 화면에서 최하은 링크를 클릭했을 때 서버에 요청되는 URL은

"http://localhost:8088/student/detail?id=1" 이다.

위 두 URL의 앞부분이 일치하고, 파일명 부분만 다르다.

### 절대 주소 URL

<a href="http://localhost:8088**/**student/detail?id=1">최하은</a>

이 a 태그를 클릭하면, "http://localhost:8088**/**student/detail?id=1" URL 이 서버에 요청된다.

이 URL은 절대 주소 URL 이다.

절대 주소 URL은 http 프로토콜 부분부터 시작하거나, 포트 번호 뒤 **/** 부분부터 시작한다.

http://localhost:8088**/**student/detail?id=1

**/**student/detail?id=1

<a href="detail?id=1">최하은</a>

이 URL은 상대 주소 URL 이다. (절대 주소 URL 형태가 아니면, 상대 주소 URL 이다)

이 a 태그를 클릭하면, 현재 웹브라우저 주소칸의 URL 에서

파일명 부분만 "detail?id=1" 으로 바뀐 URL이 서버에 요청된다.

현재 웹브라우저 주소칸의 URL이 http://localhost:8088/student/test1 이므로

http://localhost:8088/student/detail?id=1 URL이 서버에 요청된다.

<a href="http://localhost:8088/student/detail?id=1">최하은</a>

<a href="detail?id=1">최하은</a>

위 두 a 태그는 클릭했을 때, 서버에 요청되는 URL이 동일하다.

따라서 좀 더 간결한 상대 주소 URL을 사용하여 구현하는 것이 더 바람직하다.

절대 주소 URL을 사용하여 구현한 경우에는,

서버 IP 주소가 바뀌면, URL에서 서버 주소를 수정해 주어야 하니, 매우 번거롭다.

상대 주소 URL을 사용하여 구현하면 이런 번거로움이 없다.

# student/list2

## StudentController 수정

### src/main/net/skhu/controller/StudentController.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49 | package net.skhu.controller;  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 net.skhu.mapper.StudentMapper;  @Controller  public class StudentController {  @Autowired StudentMapper studentMapper;  @RequestMapping("student/list")  public String list(Model model) {  model.addAttribute("students", studentMapper.findAll());  return "student/list";  }  @RequestMapping("student/list1")  public String list1(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list1";  }  @RequestMapping("student/detail")  public String detail(Model model, Integer id) {  if (id == null) id = 4;  model.addAttribute("student", studentMapper.findById(id));  return "student/detail";  }  @RequestMapping("student/test1")  public String test1(Model model) {  return "student/test1";  }  @RequestMapping("student/list2")  public String list2(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list2";  }  } |

student/list1 액션 메소드와 student/list2 액션 메소드 구현은 동일하다.

리턴하는 뷰의 이름만 다르다.

## student/list2.jsp

### src/main/webapp/WEB-INF/views/student/list2.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <style>  div.container { width: 900px; margin: 50px auto; }  thead th { background-color: #eee; }  table{ border-collapse: collapse; width: 100%; }  td, th { padding: 4px; border: 1px solid lightgray; }  td:nth-child(4) { text-align: center; }  input { padding: 5px; font-size: 10pt; }  button { margin: 10px ; padding: 0.4em 2em; }  </style>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <form>  <label>이름</label>  <input type="text" name="srchText" value="${ srchText }" placeholder="검색조건" />  <button type="submit">조회</button>  </form>    <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>전화</th>  <th>성별</th>  <th>이메일</th>  </tr>  </thead>  <tbody>  <c:forEach var="student" items="${ students }">  <tr>  <td>${ student.studentNo }</td>  <td><a href="detail?id=${ student.id }">${ student.name }</a></td>  <td>${ student.departmentName }</td>  <td>${ student.phone }</td>  <td>${ student.sex }</td>  <td>${ student.email }</td>  </tr>  </c:forEach>  </tbody>  </table>  </div>  </body>  </html> |

<td><a href="detail?id=${ student.id }">${ student.name }</a></td>

위 태그의 실행 결과 출력은 다음과 같다.

<td><a href="detail?id=1">최하은</a></td>

<td><a href="detail?id=2">김진영</a></td>

<td><a href="detail?id=3">나철진</a></td>

<td><a href="detail?id=4">이익수</a></td>

...

### http://localhost:8088/student/list2Graphical user interface, table Description automatically generated

위 학생 목록에서 '최하은' 이름을 클릭하면, 아래 화면이 출력된다.

Table

Description automatically generated

시험범위는 여기까지

# student/list3

## Javascript

### JSP 뷰 파일이 서버에서 실행될 때

JSP 파일 내부의 Javascript 코드는 평범한 html 태그로 취급되어 그냥 그대로 출력되어 웹브라우저로 전송된다.

### 웹서버의 출력이 웹브라우저로 전송되면

전송된 출력에 포함된 Javascript 코드가 웹브라우저에서 실행된다.

### script 태그

|  |
| --- |
| <script>  자바스크립트 코드  </script> |

자바스크립트 코드는 script 태그 내부에 들어있어야 한다.

## location.href

|  |
| --- |
| var url = "http://www.naver.com";  location.href = url; |

위 코드는 웹브라우저에서 실행되는 자바스트립트 코드이다.

var 키워드는 javascript 변수를 선언하는 키워드이다.

location.href 속성에 URL을 대입하는 것은, 웹브라우저의 주소칸에 그 URL을 입력하고

엔터키를 누르는 것과 같다.

그 URL이 서버에 요청되고, 서버가 응답한 내용이 웹브라우저 창에 표시된다.

## jquery 자바스크립트 라이브러리

웹브라우저에서 실행되는 간단한 기능을 자바스크립트로 구현 할 때,

jquery 자바스크립트 라이브러리가 널리 사용된다.

이것을 사용하려면 웹페이지에 아래의 태그가 포함되어야 한다.

|  |
| --- |
| <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script> |

### jquery $ 함수

jquery 기능을 사용하는 자바스크립트 코드의 형태는 다음과 같다.

|  |
| --- |
| $("**CSS실렉터**").메소드(파라미터); |

위 코드에서 "**CSS실렉터**" 부분이 따옴표로 묶여 있음에 주목하자.

위 코드는 **CSS실렉터** 가 지정하는 태그들에 대해서 **메소드**를 실행한다.

예:

|  |
| --- |
| $("[data-url]").click(...); |

data-url 애트리뷰트를 가진 태그들에 대해서 click 메소드를 실행한다.

### click 메소드

|  |
| --- |
| $("CSS실렉터").click(function() {  실행될 자바스크립트 코드  }); |

"CSS실렉터" 가 지정하는 태그에, 그 태그가 클릭되면 실행될 자바스크립트 코드를 등록한다.

예:

|  |
| --- |
| $("span#hello").click(function() {  alert("안녕하세요");  }); |

<span id="hello"> 태그를 클릭하면 "안녕하세요" 메시지 박스가 실행된다.

### attr 메소드

|  |
| --- |
| $("CSS실렉터").attr("애트리뷰트명"); |

"CSS실렉터" 가 지정하는 태그의 애트리뷰트 값을 리턴한다.

예:

|  |
| --- |
| $("tr[data-url]").attr("data-url"); |

<tr data-url="...."> 태그의 data-url 애트리뷰트 값을 리턴한다.

|  |
| --- |
| <script>  $("[data-url]").click(function() {  var url = $(this).attr("data-url");  location.href = url;  })  </script> |

data-url 애트리뷰트를 가진 태그를 클릭하면 다음 코드가 실행된다.

var url = $(this).attr("data-url");

location.href = url;

위 코드는 클릭된 그 태그의 data-url 애트리뷰트 값을 location.href 속성에 대입한다.

location.href 속성에 대입된 값은 URL 이어야 한다.

그 URL이 서버에 요청되고, 서버가 응답한 내용이 웹브라우저 창에 표시된다.

## StudentController 수정

### src/main/net/skhu/controller/StudentController.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57 | package net.skhu.controller;  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 net.skhu.mapper.StudentMapper;  @Controller  public class StudentController {  @Autowired StudentMapper studentMapper;  @RequestMapping("student/list")  public String list(Model model) {  model.addAttribute("students", studentMapper.findAll());  return "student/list";  }  @RequestMapping("student/list1")  public String list1(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list1";  }  @RequestMapping("student/detail")  public String detail(Model model, Integer id) {  if (id == null) id = 5;  model.addAttribute("student", studentMapper.findById(id));  return "student/detail";  }  @RequestMapping("student/test1")  public String test1(Model model) {  return "student/test1";  }  @RequestMapping("student/list2")  public String list2(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list2";  }  @RequestMapping("student/list3")  public String list3(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list3";  }  } |

student/list1 액션 메소드, student/list2 액션 메소드, student/list3 액션 메소드 구현은 동일하다.

리턴하는 뷰의 이름만 다르다.

## student/list3.jsp

### src/main/webapp/WEB-INF/views/student/list3.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <style>  div.container { width: 900px; margin: 50px auto; }  thead th { background-color: #eee; }  table{ border-collapse: collapse; width: 100%; }  td, th { padding: 4px; border: 1px solid lightgray; }  td:nth-child(4) { text-align: center; }  input { padding: 5px; font-size: 10pt; }  button { margin: 10px ; padding: 0.4em 2em; }  tr[data-url]:hover { background-color: #ffb; cursor: pointer; }  </style>  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>  </head>  <body>  <div class="container">  <h1>학생목록</h1>  <form>  <label>이름</label>  <input type="text" name="srchText" value="${ srchText }" placeholder="검색조건" />  <button type="submit">조회</button>  </form>    <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>전화</th>  <th>성별</th>  <th>이메일</th>  </tr>  </thead>  <tbody>  <c:forEach var="student" items="${ students }">  <tr data-url="detail?id=${ student.id }">  <td>${ student.studentNo }</td>  <td><a href="detail?id=${ student.id }">${ student.name }</a></td>  <td>${ student.departmentName }</td>  <td>${ student.phone }</td>  <td>${ student.sex }</td>  <td>${ student.email }</td>  </tr>  </c:forEach>  </tbody>  </table>  </div>  <script>  $("[data-url]").click(**function**() {  **var** url = $(**this**).attr("data-url");  location.href = url;  })  </script>  </body>  </html> |

<tr data-url="detail?id=${ student.id }">

위 소스 코드의 실행 결과 출력은 다음과 같다.

<tr data-url="detail?id=1">

<tr data-url="detail?id=2">

<tr data-url="detail?id=3">

<tr data-url="detail?id=3">

이 태그를 클릭하면, 웹브라우저는 "detail?id=3" 상대 URL을 서버에 요청한다.

즉 웹브라우저 창은 "detail?id=3" 상대 URL로 넘아간다.

절대 URL "http://localhost:8088/student/detail?id=3"

tr[data-url]:hover { background-color: #ffb; cursor: pointer; }

tr[data-url]

data-url 애트리뷰트가 포함된 tr 태그들에 이 서식이 적용된다.

tr[data-url]:hover

data-url 애트리뷰트가 포함된 tr 태그들 중에서

현재 마우스 포인터가 위치한 태그에 이 서식이 적용된다.

{ background-color: #ffb; cursor: pointer; }

배경색은 밝은 노란색, 마우스 포인터는 손가락 모양이된다

## student/detail.jsp 수정 - 되돌아 가기 기능

### src/main/webapp/WEB-INF/views/student/detail.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <style>  div.container { width: 600px; margin: 50px auto; }  body { font-family: 굴림체; }  table { width: 500px; border-collapse: collapse; }  table td:nth-child(1) { background-color: #eee; }  td { padding: 5px; border: 1px solid lightgray; }  </style>  </head>  <body>  <div class="container">  <h1>학생</h1>  <table>  <tr>  <td>ID</td>  <td>${ student.id }</td>  </tr>  <tr>  <td>학번</td>  <td>${ student.studentNo }</td>  </tr> <tr>  <td>이름</td>  <td>${ student.name }</td>  </tr>  <tr>  <td>학과ID</td>  <td>${ student.departmentId }</td>  </tr>  <tr>  <td>학과명</td>  <td>${ student.departmentName }</td>  </tr>  <tr>  <td>전화</td>  <td>${ student.phone }</td>  </tr>  <tr>  <td>성별</td>  <td>${ student.sex }</td>  </tr>  <tr>  <td>이메일</td>  <td>${ student.email }</td>  </tr>  </table>  <a href="javascript:window.history.back()">돌아가기</a>  </div>  </body>  </html> |

<a href="javascript:window.history.back()">돌아가기</a>

이 태그를 클릭하면, 웹브라우저에서 백 버튼을 클릭한 것과 동일한 일이 발생한다.

# student/list0

## StudentController 수정

### src/main/java/net/skhu/controller/StudentController.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63 | package net.skhu.controller;  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 net.skhu.mapper.StudentMapper;  @Controller  public class StudentController {  @Autowired StudentMapper studentMapper;  @RequestMapping("student/list")  public String list(Model model) {  model.addAttribute("students", studentMapper.findAll());  return "student/list";  }  @RequestMapping("student/list0")  public String list0(Model model) {  model.addAttribute("students", studentMapper.findAll());  return "student/list0";  }  @RequestMapping("student/list1")  public String list1(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list1";  }  @RequestMapping("student/detail")  public String detail(Model model, Integer id) {  if (id == null) id = 5;  model.addAttribute("student", studentMapper.findById(id));  return "student/detail";  }  @RequestMapping("student/test1")  public String test1(Model model) {  return "student/test1";  }  @RequestMapping("student/list2")  public String list2(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list2";  }  @RequestMapping("student/list3")  public String list3(Model model, String srchText) {  if (srchText == null) srchText = "";  model.addAttribute("students", studentMapper.findByName(srchText + "%"));  model.addAttribute("srchText", srchText);  return "student/list3";  }  } |

student/list 액션 메소드와 student/list1 액션 메소드 구현은 동일하다.

리턴하는 뷰의 이름만 다르다.

## student/list0.jsp

### src/main/webapp/WEB-INF/views/student/list0.jsp

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54 | <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>  <!DOCTYPE html>  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">  <style>  div.container { width: 900px; margin: 50px auto; }  thead th { background-color: #eee; }  table{ border-collapse: collapse; width: 100%; }  td, th { padding: 4px; border: 1px solid lightgray; }  td:nth-child(4) { text-align: center; }  tr[data-url]:hover { background-color: #ffb; cursor: pointer; }  </style>  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>  </head>  <body>  <div class="container">  <h1>학생목록</h1>    <table class="table table-bordered table-condensed">  <thead>  <tr>  <th>학번</th>  <th>이름</th>  <th>학과</th>  <th>전화</th>  <th>성별</th>  <th>이메일</th>  </tr>  </thead>  <tbody>  <c:forEach var="student" items="${ students }">  <tr data-url="detail?id=${ student.id }">  <td>${ student.studentNo }</td>  <td>${ student.name }</td>  <td>${ student.departmentName }</td>  <td>${ student.phone }</td>  <td>${ student.sex }</td>  <td>${ student.email }</td>  </tr>  </c:forEach>  </tbody>  </table>  </div>  <script>  $("[data-url]").click(**function**() {  **var** url = $(**this**).attr("data-url");  location.href = url;  })  </script>  </body>  </html> |

## 실행

### http://localhost:8088/student/list0

Graphical user interface, table

Description automatically generated

# 연습문제

student/list0 페이지와 같은 구조로 book/list0 페이지를 구현하라.