

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Sử dụng biểu thức EL để truy cập các attribute
- ✓ Sử dụng JSTL để lập trình giao diện trong JSP

PHẦN I

Bài 1 (2 điểm)

Thực hiện theo hướng dẫn sau đây để sử dụng biểu thức EL kết xuất thông tin từ attribute trong ServletContext, HttpSession và HttpServletRequest

Bước 1: Tạo StudentController như sau

```
@Controller
@RequestMapping("/student/")
public class StudentController {
    @Autowired
    ServletContext application;

    @RequestMapping("index")
    public String index(HttpServletRequest request, HttpSession session) {
        application.setAttribute("name", "Phạm Minh Tuấn");
        application.setAttribute("level", 2);

        session.setAttribute("name", "Nguyễn Thị Kiều Oanh");
        session.setAttribute("level", 4);

        request.setAttribute("name", "Lê Phạm Tuấn Kiệt");
        request.setAttribute("level", 3);

        session.setAttribute("salary", 1000);
        request.setAttribute("photo", "images/poly-sv.png");

        return "student/index";
    }
}
```

}

Bước 2: Tạo view index.jsp với nội dung như sau

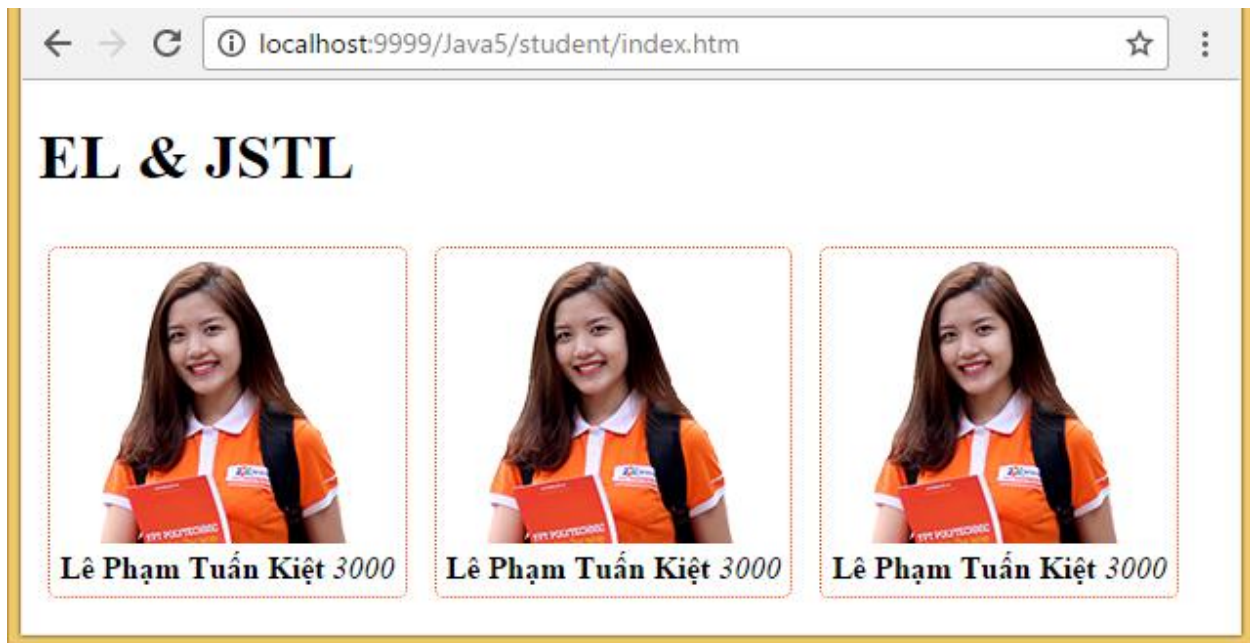
```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <title>Spring MVC</title>
    <base href="${pageContext.servletContext.contextPath}/">
    <style>
        div{
            display: inline-block;
            text-align: center;
            margin: 5px;
            padding: 5px;
            border: 1px dotted orangered;
            border-radius: 5px;
        }
    </style>
</head>
<body>
    <h1>EL & JSTL</h1>
    <div>
        
        <br>
        <strong>${name}</strong>
        <em>${salary*level}</em>
    </div>
    <div>
        
        <br>
        <strong>${name}</strong>
        <em>${salary*level}</em>
    </div>
    <div>
```

```

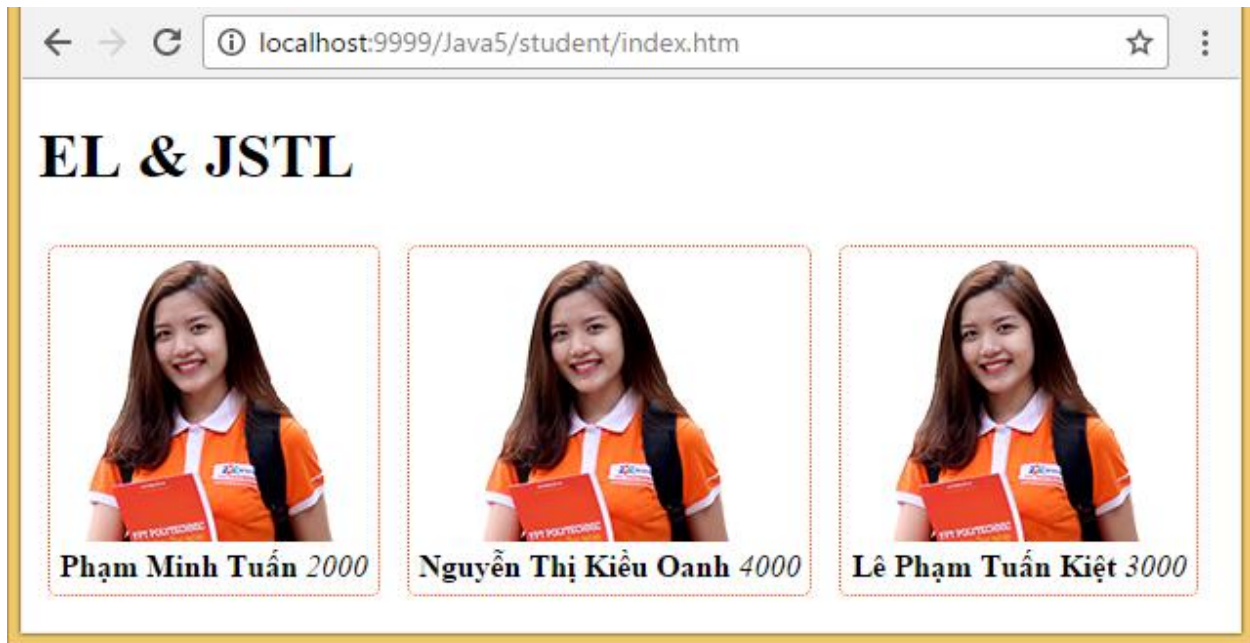

<br>
<strong>${name}</strong>
<em>${salary*level}</em>
</div>
</body>
</html>

```

Bước 3: Chạy student/index.htm sẽ có kết quả như sau



Yêu cầu: Hiệu chỉnh các biểu thức EL để trang web hiển thị với kết quả như sau



Bài 2 (2 điểm)

Thực hiện theo hướng dẫn sau đây để sử dụng EL truy xuất các thuộc tính bean, Map và Collection

Bước 1: Tạo lớp Student.java có mã như sau

```
package poly.bean;

public class Student {
    private String name;
    private Double mark;
    private String major;

    public Student(){}

    public Student(String name, Double mark, String major) {
        this.name = name;
        this.mark = mark;
        this.major = major;
    }

    public String getName() {
        return name;
    }
}
```

```

public void setName(String name) {
    this.name = name;
}
public Double getMark() {
    return mark;
}
public void setMark(Double mark) {
    this.mark = mark;
}
public String getMajor() {
    return major;
}
public void setMajor(String major) {
    this.major = major;
}
}

```

Bước 2: Bổ sung phương thức action vào StudentController ở bài 1 có mã nguồn như sau:

```

@RequestMapping("index2")
public String index2(ModelMap model) {
    Student sv1 = new Student("Phạm Minh Tuấn", 5.5, "Ứng dụng phần mềm");
    Student sv2 = new Student("Nguyễn Thị Kiều Oanh", 9.5, "Thiết kế trang web");
    Student sv3 = new Student("Lê Phạm Tuấn Kiệt", 3.5, "Thiết kế trang web");

    List<Student> list = new ArrayList<>();
    list.add(sv2);
    list.add(sv3);

    Map<String, Student> map = new HashMap<>();
    map.put("OanhNTK", sv2);
    map.put("KietLPT", sv3);

    model.addAttribute("bean", sv1);
    model.addAttribute("list", list);
    model.addAttribute("map", map);

    return "student/index2";
}

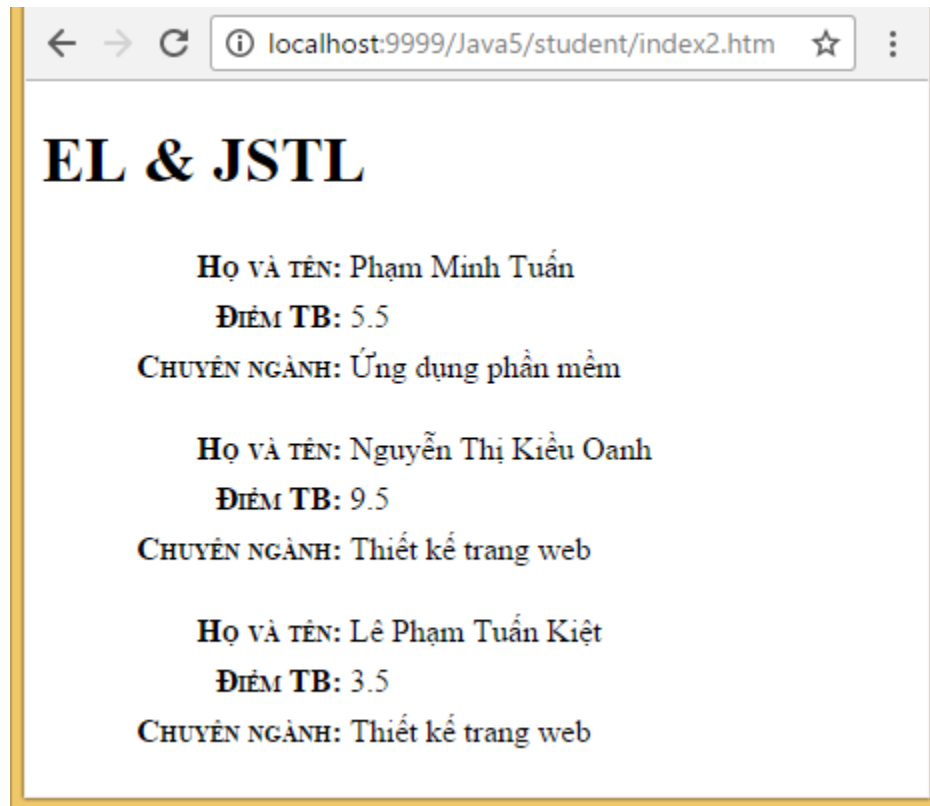
```

Bước 3: Tạo view index2.jsp có mã như sau

```

<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <title>Spring MVC</title>
    <base href="${pageContext.servletContext.contextPath}/">
    <style>
        li{
            list-style: none;
            line-height: 25px;
        }
        li>label{
            display: inline-block;
            text-align: right;
            width: 110px;
            font-variant: small-caps;
            font-weight: bold;
        }
    </style>
</head>
<body>
    <h1>EL & JSTL</h1>
    <ul>
        <li><label>Họ và tên:</label> ${??}</li>
        <li><label>Điểm TB:</label> ${??}</li>
        <li><label>Chuyên ngành:</label> ${??}</li>
    </ul>
    <ul>
        <li><label>Họ và tên:</label> ${??}</li>
        <li><label>Điểm TB:</label> ${??}</li>
        <li><label>Chuyên ngành:</label> ${??}</li>
    </ul>
    <ul>
        <li><label>Họ và tên:</label> ${??}</li>
        <li><label>Điểm TB:</label> ${??}</li>
        <li><label>Chuyên ngành:</label> ${??}</li>
    </ul>
</body>
</html>
    
```

Yêu cầu: Hiệu chỉnh EL trong index2.jsp để khi chạy student/index2.htm có kết quả hiển thị như sau



Trong đó:

- ✓ Thông tin sinh viên thứ nhất lấy từ **bean**
- ✓ Thông tin sinh viên thứ nhất lấy từ phần tử **thứ nhất của list**
- ✓ Thông tin sinh viên thứ nhất lấy từ phần tử có **key là KietLPT**

PHẦN II

Bài 3 (2 điểm)

Thực hiện theo hướng dẫn sau đây để hiển thị thêm thông tin danh hiệu của sinh viên nếu điểm từ 9 trở lên trong bài 2 ở trên.

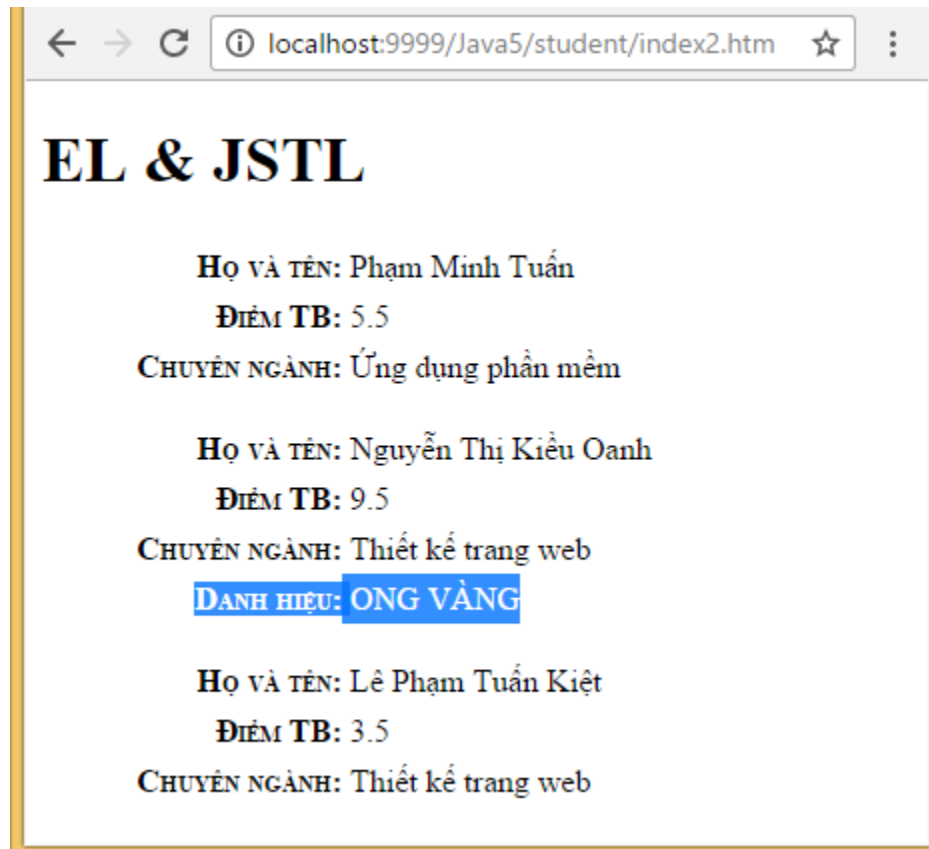
Bước 1: Khai báo thư viện core của JSTL ở đầu trang index2.jsp

```
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
```

Bước 2: Bổ sung mã HTML sau đây vào cuối mỗi thẻ . Chú ý điền vào phần ??? thích hợp để truy xuất điểm của sinh viên đang hiển thị trong

```
<c:if test="${???}.mark >= 9}">
    <li><label> Danh hiệu:</label> ONG VÀNG</li>
</c:if>
```

Bước 3: Chạy lại student/index2.jsp bạn sẽ có được thông tin sau



Bài 4 (2 điểm)

Bổ sung thông tin xếp loại cho các sinh viên trong bài 3 bằng cách thêm đoạn mã sau vào cuối mỗi

```
<li>
    <label>Xếp loại:</label>
    <c:choose>
        <c:when test="${???}.mark < 5}">YẾU</c:when>
        <c:when test="${???}.mark < 6.5}">TRUNG BÌNH</c:when>
        <c:when test="${???}.mark < 7.5}">KHÁ</c:when>
```

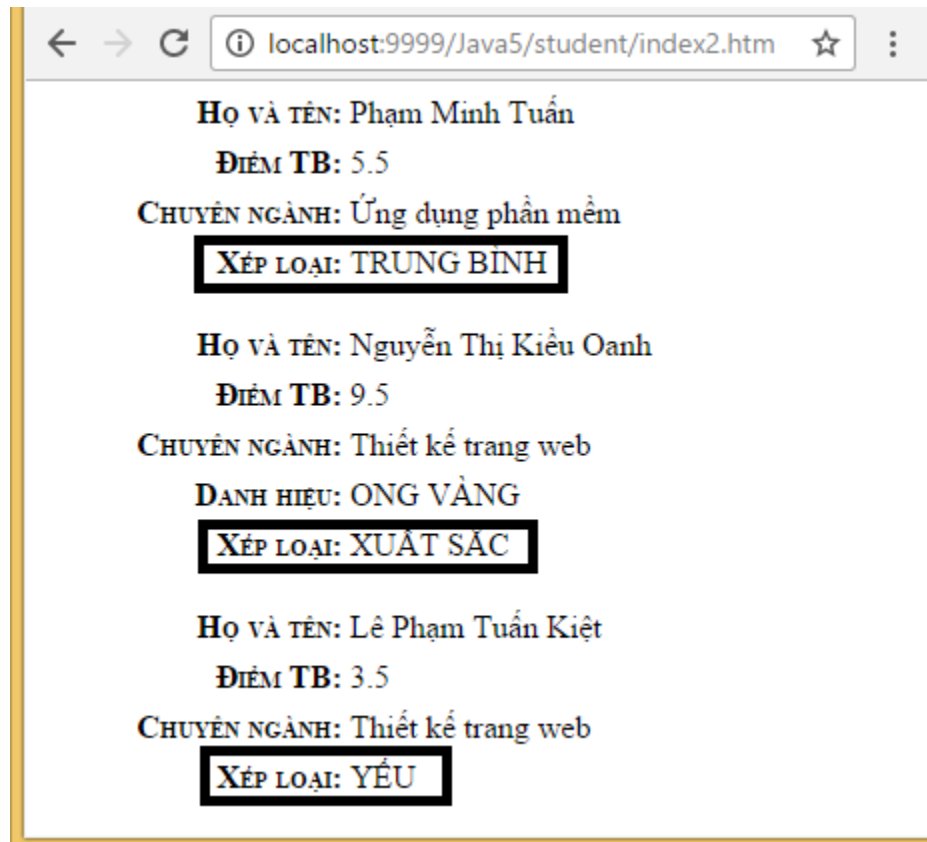


```

    <c:when test="${???}.mark < 9}">GIỎI</c:when>
    <c:otherwise>XUẤT SẮC</c:otherwise>
  </c:choose>
</li>

```

Chạy student/index2.jsp bạn sẽ nhận được kết quả hiển thị như sau



Bài 5 (2 điểm)

Hiển thị List<Product> như hình sau



Tên SP	Giá cũ	Giảm giá	Giá mới
Nokia Star	\$1,000.00	5%	\$950.00
iPhone 9	\$1,500.00	10%	\$1,350.00
Samsung Galaxy N10	\$750.00	15%	\$637.50
Sony Experia	\$500.00	0%	\$500.00

Thực hiện theo hướng dẫn sau

Bước 1: Xây dựng bean Product

```
package poly.bean;

public class Product {
    private String name;
    private Double unitPrice;
    private Double discount;

    public Product(){}
    public Product(String name, Double unitPrice, Double discount) {
        this.name = name;
        this.unitPrice = unitPrice;
        this.discount = discount;
    }

    public Double getNewPrice() {
        return unitPrice*(1-discount);
    }

    public String getName() {
        return name;
    }
}
```

```

    }
    public void setName(String name) {
        this.name = name;
    }
    public Double getUnitPrice() {
        return unitPrice;
    }
    public void setUnitPrice(Double unitPrice) {
        this.unitPrice = unitPrice;
    }
    public Double getDiscount() {
        return discount;
    }
    public void setDiscount(Double discount) {
        this.discount = discount;
    }
}
    
```

Chú ý: Lớp bean này có thuộc tính **newPrice** chứa giá mới (sau khi đã giảm giá)

Bước 2: Xây dựng ProductController

```

@Controller
@RequestMapping("/product/")
public class ProductController {
    @RequestMapping("list")
    public String list(ModelMap model) {
        List<Product> list = new ArrayList<>();
        list.add(new Product("Nokia Star", 1000.0, 0.05));
        list.add(new Product("iPhone 9", 1500.0, 0.1));
        list.add(new Product("Samsung Galaxy N10", 750.0, 0.15));
        list.add(new Product("Sony Experia", 500.0, 0.0));

        model.addAttribute("prods", list);
        return "product/list";
    }
}
    
```

Phương thức action chia sẻ List<Product> trong model với tên là prods.

Bước 3: Xây dựng view list.jsp

```
<%@ page pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jstl/fmt_rt" prefix="f" %>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8"/>
    <title>Spring MVC</title>
    <base href="${pageContext.servletContext.contextPath}/">
    <style>
        table{
            border-collapse: collapse;
            width: 100%;
        }
        th, td{
            line-height: 25px;
            border: 1px solid black;
            padding: 5px;
        }
        th{
            background-color: gray;
        }
    </style>
</head>
<body>
    <h1>EL & JSTL</h1>
    <table>
    <tr>
        <th>Tên SP</th>
        <th>Giá cũ</th>
        <th>Giảm giá</th>
        <th>Giá mới</th>
    </tr>
    <!--Dữ liệu trong prods sẽ được hiển thị ở vị trí này-->
    </table>
</body>
</html>
```

Trong view này cần duyệt List<Product> nên cần thư viện **core**, đồng thời dữ liệu được hiển thị cần được định dạng phù hợp nên cần thư viện **format**.

Để duyệt qua dữ liệu trong List<Product> prods bạn sử dụng <c:forEach>

```
<c:forEach var="p" items="${prods}">
<tr>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
</tr>
</c:forEach>
```

Mỗi cột cần phải hiển thị dữ liệu phù hợp. Ví dụ

- ✓ \${p.name} hiển thị tên
- ✓ \${p.newPrice} hiển thị giá mới. Chú ý trong Product có định nghĩa phương thức **getNewPrice()**
- ✓ ...

Muốn định dạng dữ liệu bạn cần sử dụng <f:formatNumber>

- ✓ <f:formatNumber value="\${p.newPrice}" type="currency"/> định dạng tiền tệ
- ✓ <f:formatNumber value="\${p.discount}" type="percent"/> định dạng %
- ✓ ...

Bước 4: Chạy product/list.htm bạn sẽ có kết quả như phần giới thiệu

Chú ý:

- ✓ Phần I và Phần II chỉ áp dụng cho dạy tích hợp. Sinh viên làm phần 1 và phần 2 theo 2 bài khác nhau tương ứng với 2 phần lý thuyết đã dạy trong bài học.
- ✓ Nếu giảng dạy theo phương pháp truyền thống thì sinh viên phải thực hiện tất cả các bài trong một buổi.