SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

Web-BackEnd

<알림>

본 강의는 삼성 청년 SW아카데미의 컨텐츠로
보안서약서에 의거하여
강의 내용을 어떠한 사유로도 임의로 복사,
촬영, 녹음, 복제, 보관, 전송하거나
허가 받지 않은 저장매체를
이용한 보관, 제3자에게 누설, 공개,
또는 사용하는 등의 행위를 금합니다.

SAMSUNG SW ACADEMY FOR YOUTH

목차

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Model1

삼성 청년 SW 아카데미

Web Application Architecture - MVC

✓ **Web Application Architecture.**

- JSP를 이용하여 구성할 수 있는 Web Application Architecture는 크게 model1과 model2로 나뉜다.
- JSP가 client의 요청에 대한 Logic 처리와 response page(view)에 대한 처리를 모두 하느냐, 아니면 response page(view)에 대한 처리만 하는지가 가장 큰 차이점이다.
- Model2구조는 MVC(Model-View-Controller) Pattern을 web 개발에 도입한 구조를 말한다.

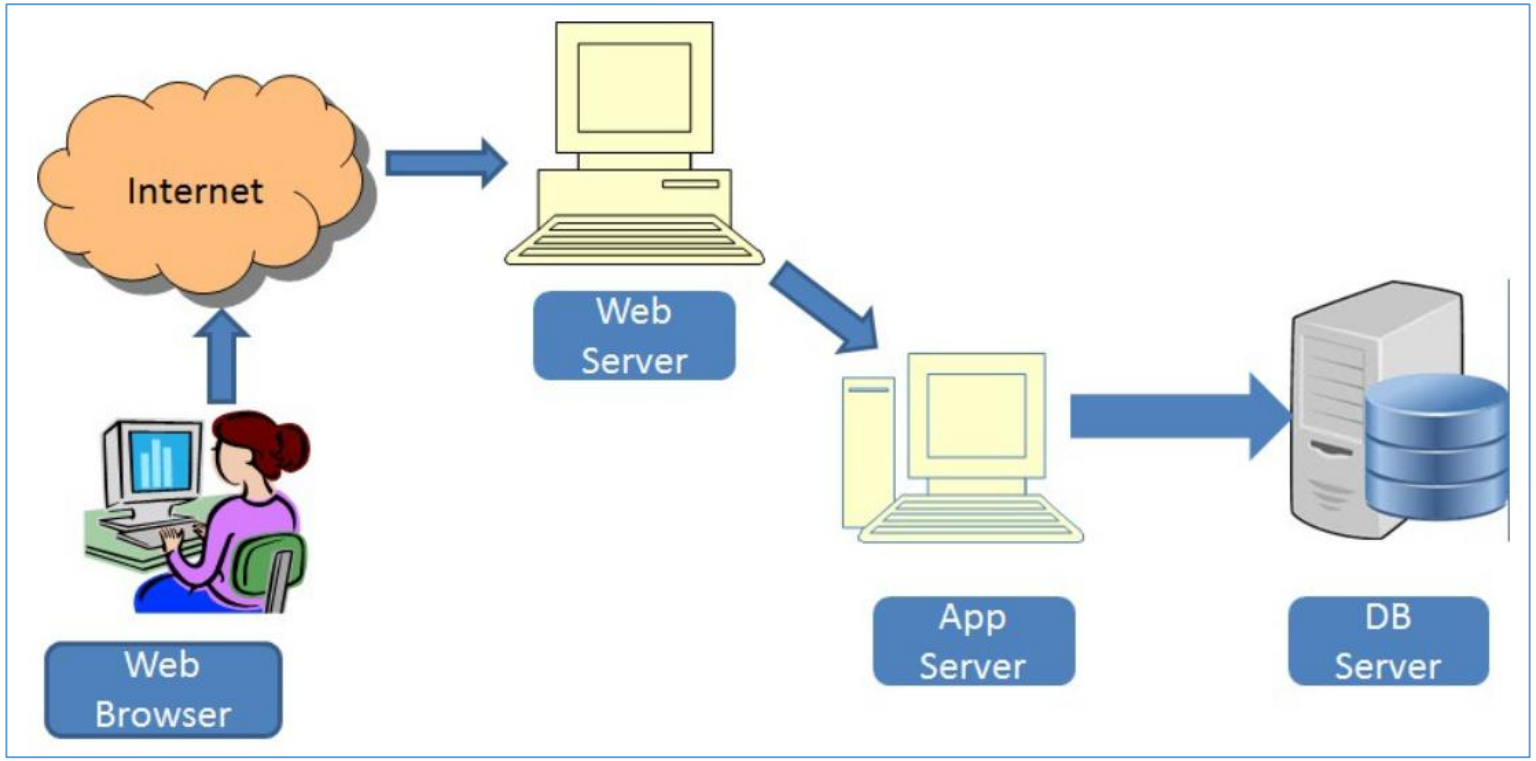
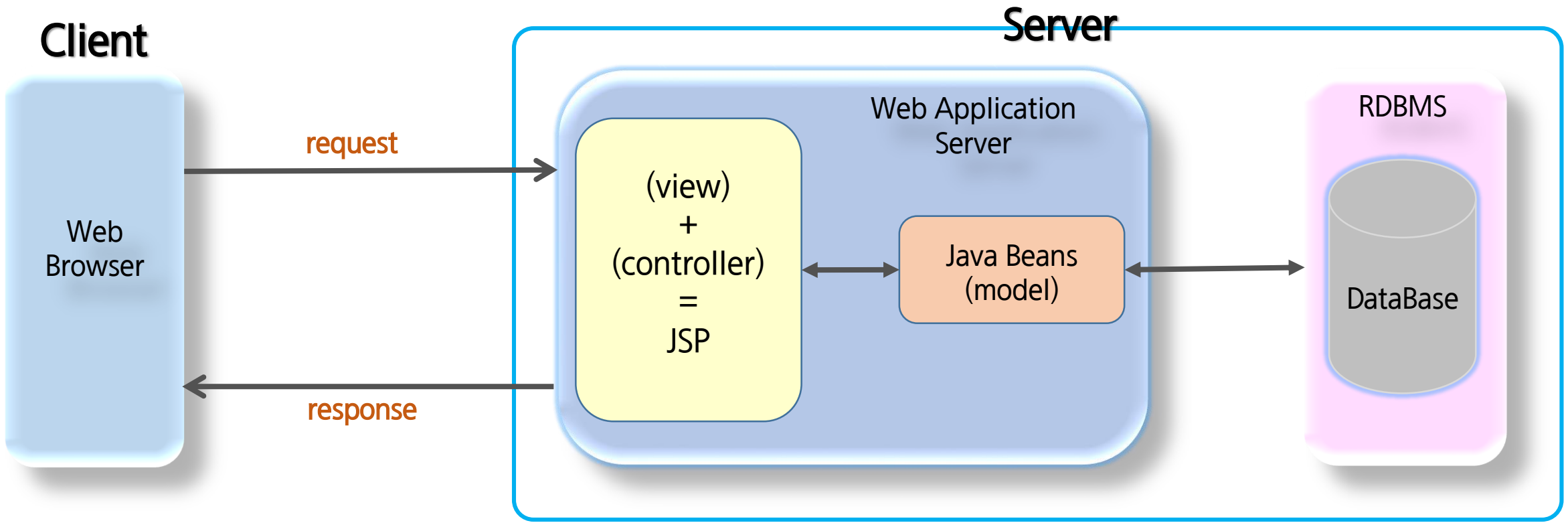


그림 출처 : leafcats

Web Application Architecture - MVC

✓ Model1 구조.

- model1은 view와 logic을 JSP 페이지 하나에서 처리하는 구조를 말한다.
- client로부터 요청이 들어오게 되면 JSP 페이지는 java beans나 별도의 service class를 이용하여 작업을 처리, 결과를 client에 출력한다.



[JSP Model1 Architecture]

Web Application Architecture - MVC

✓ **Model1 구조의 장단점.**

- **간단한 page를 구성하기 위해 과거에 가장 많이 사용되었던 architecture.**

장점	단점
구조가 단순하며 직관적이기 때문에 배우기 쉽다.	출력을 위한 view(html) 코드와 로직 처리를 위한 java 코드가 섞여 있기 때문에 JSP 코드 자체가 복잡해 진다.
개발 시간이 비교적 짧기 때문에 개발 비용이 감소.	JSP코드에 Back-End(Developer)와 Front-End(Designer)가 혼재되기 때문에 분업이 힘들어진다.
	project의 규모가 커지게 되면 코드가 복잡해 지므로 유지보수 하기가 어려워 진다.
	확장성(신기술의 도입, framework등..)이 나쁘다.

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Model2 MVC Pattern
(Model-View-Controller)

삼성 청년 SW 아카데미

Web Application Architecture - MVC

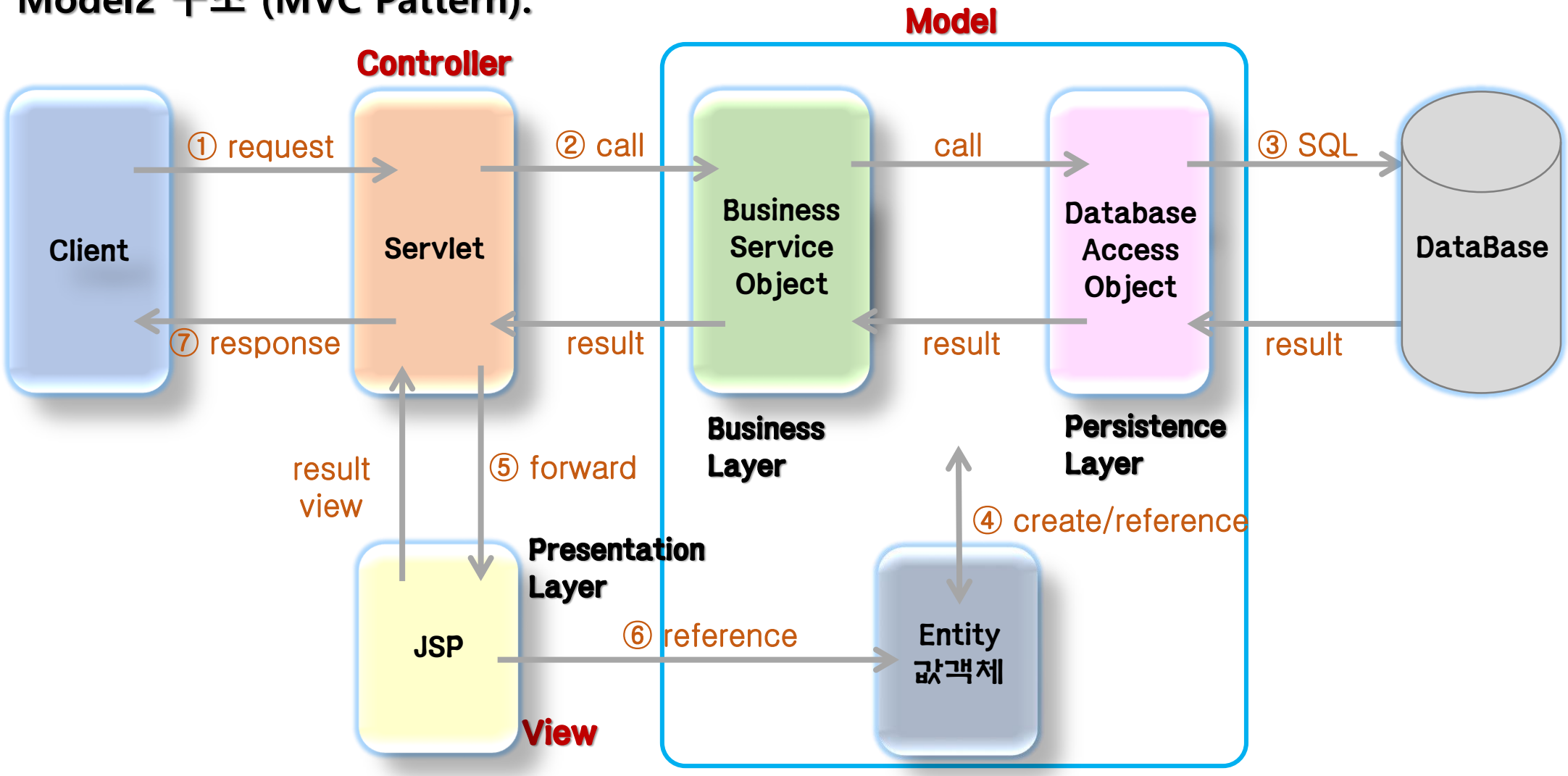
✓ **Model2 구조.**

- model2는 모든 처리를 JSP 페이지에서 하는 것이 아니라, client 요청에 대한 처리는 servlet이, logic처리는 java class(Service, Dao, ..), client에게 출력하는 response page를 JSP가 담당한다.
- model2 구조는 MVC(Model-View-Controller) pattern을 웹개발에 도입한 구조이며 완전히 같은 형태를 보인다.

Model2	MVC Pattern	설명
Service, Dao or Java Beans	Model	Logic(Business & DB Logic)을 처리하는 모든 것. controller로 부터 넘어온 data를 이용하여 이를 수행하고 그에 대한 결과를 다시 controller에 return한다.
JSP	View	모든 화면 처리를 담당. Client의 요청에 대한 결과 뿐 아니라 controller에 요청을 보내는 화면단도 jsp에서 처리한다. Logic처리를 위한 java code는 사라지고 결과 출력을 위한 code만 존재.
Servlet	Controller	Client의 요청을 분석하여 Logic처리를 위한 Model단을 호출한다. return 받은 결과 data를 필요에 따라 request, session등에 저장하고, redirect 또는 forward 방식으로 jsp(view) page를 이용하여 출력한다.

Web Application Architecture - MVC

✓ Model2 구조 (MVC Pattern).



[JSP Model2 Architecture]

Web Application Architecture - MVC

✓ **Model2 구조의 장단점.**

- Model2는 Model1의 단점을 보완하기 위해 만들어 졌으나, 다루기 어렵다는 단점이 있다.

장점	단점
출력을 위한 view(html) 코드와 로직 처리를 위한 java 코드가 분리 되었기 때문에 JSP는 Model1에 비해 코드가 복잡하지 않다.	구조가 복잡하여 초기 진입이 어렵다.
화면단과 Logic단이 분리 되었기에 분업이 용이해 졌다.	개발 시간의 증가로 개발 비용 증가.
기능에 따라 code가 분리 되었기 때문에 유지 보수가 쉬워졌다.	
확장성이 뛰어나다.	

※ 모든 web site는 Model2로만 만드는가?

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

Cookie

삼성 청년 SW 아카데미

Cookie & HttpSession.

- ✓ Session & Cookie.
  - ▪ http protocol의 특징.
    - client가 server에 요청.
    - server는 요청에 대한 처리를 한 후 client에 응답.
    - 응답 후 연결을 해제. >> stateless
      - ❖ 지속적인 연결로 인한 자원낭비를 줄이기 위해 연결을 해제한다.
      - ❖ 그러나 client와 server가 연결 상태를 유지해야 하는 경우 문제가 발생(로그인정보등.).
      - ❖ 즉 client단위로 상태 정보를 유지해야 하는 경우 Cookie와 Session이 사용된다.
  - ▪ HTTP protocol의 특징(약점)을 보완하기 위해 사용.

Cookie & HttpSession.

- ✓ Cookie? : javax.servlet.http.Cookie
  - 서버에서 사용자의 컴퓨터에 저장하는 정보파일.
  - 사용자가 별도의 요청을 하지 않아도 브라우저는 request시 Request Header를 넣어 자동으로 서버에 전송.
  - key와 value로 구성되고 String 형태로 이루어져 있음.
  - Browser마다 저장되는 쿠키는 다르다. (서버에서는 Browser가 다르면 다른 사용자로 인식).

- ✓ Cookie의 사용 목적.
  - 세션관리 : 사용자 아이디, 접속시간, 장바구니 등의 서버가 알아야 할 정보 저장.
  - 개인화 : 사용자마다 다르게 그 사람에 적절한 페이지를 보여줄 수 있다.
  - 트래킹 : 사용자의 행동과 패턴을 분석하고 기록.

Cookie & HttpSession.

✓ Cookie의 사용 예.

- ID 저장 (자동로그인).
- 일주일간 다시 보지 않기.
- 최근 검색한 상품들을 광고에 추천.
- 쇼핑몰 장바구니 기능.

✓ Cookie의 구성요소.

- 이름 : 여러 개의 쿠키가 client의 컴퓨터에 저장되므로 각 쿠키를 구별하는 데 사용되는 이름.
- 값 : 쿠키의 이름과 매핑되는 값.
- 유효기간 : 쿠키의 유효기간.
- 도메인 : 쿠키를 전송할 도메인
- 경로(path) : 쿠키를 전송할 요청 경로.

Cookie & HttpSession.

✓ Cookie의 동작 순서.

- Client가 페이지를 요청.
- WAS는 Cookie를 생성.
- HTTP Header에 Cookie를 넣어 응답.
- Browser는 넘겨받은 Cookie를 PC에 저장하고, 다시 WAS가 요청할 때 요청과 함께 Cookie를 전송.
- Browser가 종료되어도 Cookie의 만료 기간이 남아 있다면 Client는 계속 보관.
- 동일 사이트 재방문시 Client의 PC에 해당 Cookie가 있는 경우, 요청 페이지와 함께 Cookie를 전송.

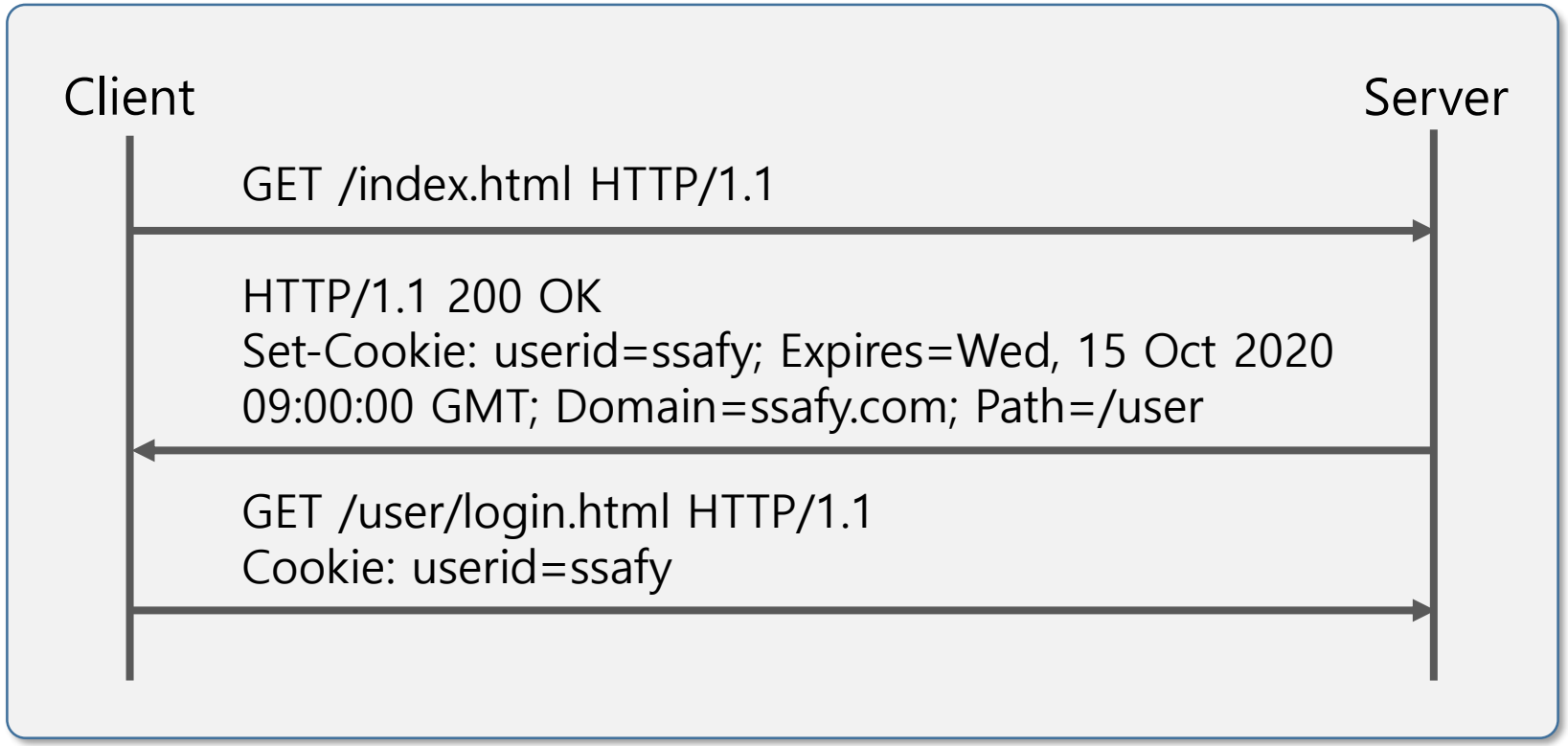
✓ Cookie의 특징.

- 이름, 값, 만료일(저장 기간 설정), 경로 정보로 구성되어 있다.
- 클라이언트에 총 300개의 쿠키를 저장할 수 있다.
- 하나의 도메인 당 20개의 쿠키를 가질 수 있다.
- 하나의 쿠키는 4KB(=4096byte)까지 저장 가능하다.

Cookie & HttpSession.

✓ Cookie의 설정.

- name이 userid인 Cookie의 값은 ssafy이고, 유효기간은 2020년 10월 15일이다.
- 유효도메인 및 경로는 ssafy.com Domain의 /user Path가 된다.



Cookie & HttpSession.

✓ Cookie의 주요 기능.

기능	method
생성	Cookie cookie = new Cookie(String name, String value);
값 변경/얻기	cookie.setValue(String value); / String value = cookie.getValue();
사용 도메인지정/얻기	cookie.setDomain(String domain); / String domain = cookie.getDomain();
값 범위지정/얻기	cookie.setPath(String path); / String path = cookie.getPath();
cookie의 유효기간지정/얻기	cookie.setMaxAge(int expiry); / int expiry = cookie.getMaxAge(); cookie 삭제 : cookie.setMaxAge(0);
생성된 cookie를 client에 전송.	response.addCookie(cookie);
client에 저장된 cookie 얻기	Cookie cookies[] = request.getCookies();

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

HttpSession

삼성 청년 SW 아카데미

Cookie & HttpSession.

- ✓ session? : javax.servlet.http.HttpSession
  - 방문자가 웹 서버에 접속해 있는 상태를 하나의 단위로 보고 그것을 세션이라 한다.
  - WAS의 memory에 Object의 형태로 저장.
  - memory가 허용하는 용량까지 제한 없이 저장 가능.

- ✓ session의 사용 예.
  - site내에서 화면을 이동해도 로그인(사용자 정보)이 풀리지 않고 유지.
  - 장바구니.

Cookie & HttpSession.

✓ session의 동작 순서.

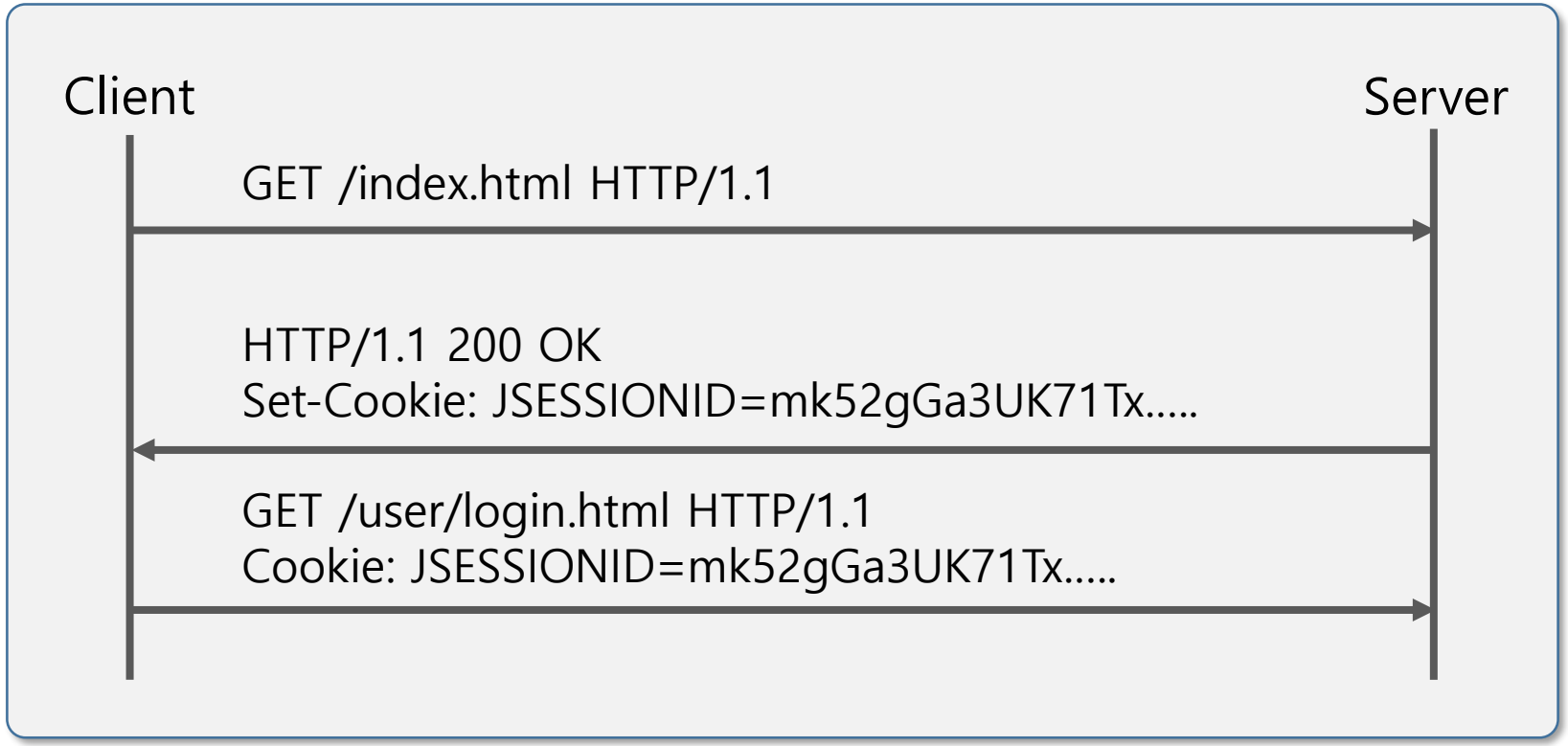
- 클라이언트가 페이지를 요청.
- 서버는 접근한 클라이언트의 Request-Header 필드인 Cookie를 확인하여, 클라이언트가 해당 session-id를 보냈는지 확인.
- session-id가 존재하지 않는다면, 서버는 session-id를 생성해 클라이언트에게 돌려준다.
- 서버에서 클라이언트로 돌려준 session-id를 쿠키를 사용해 서버에 저장. 쿠키 이름 : JSESSIONID
- 클라이언트는 재 접속 시, 이 쿠키(JSESSIONID)를 이용하여 session-id 값을 서버에 전달.

✓ session의 특징.

- 웹 서버에 웹 컨테이너의 상태를 유지하기 위한 정보를 저장.
- 웹 서버에 저장되는 쿠키(=세션 쿠키)
- 브라우저를 닫거나, 서버에서 세션을 삭제 했을 때만 삭제가 되므로, 쿠키보다 비교적 보안이 좋다.
- 저장 데이터에 제한이 없다.
- 각 클라이언트 고유 Session ID를 부여한다.
- Session ID로 클라이언트를 구분하여 각 클라이언트 요구에 맞는 서비스 제공.

Cookie & HttpSession.

- ✓ session의 설정.
  - Browser당 하나의 JSESSIONID를 할당 받음.
  - 아이디 또는 닉네임과 같이 로그인을 했을 경우 자주 사용되는 정보를 session에 저장하면 DB를 접근할 필요가 없으므로 효율적이다.



Cookie & HttpSession.

✓ HttpSession의 주요 기능.

기능	method
생성	HttpSession session = request.getSession(); HttpSession session = request.getSession(false);
값 저장	session.setAttribute(String name, Object value);
값 얻기	Object obj = session.getAttribute(String name);

값 제거	session.removeAttribute(String name); // 특정 이름의 속성제거
	session.invalidate(); // binding되어 있는 모든 속성 제거

생성시간	long ct = session.getCreationTime();
마지막 접근 시간	long lat = session.getLastAccessedTime();

Cookie & HttpSession.

✓ Session & Cookie 정리.

	Session	Cookie
Type	javax.servlet.http.HttpSession (Interface)	javax.servlet.http.Cookie (Class)
저장 위치	Server의 memory에 Object로 저장.	Client 컴퓨터에 file로 저장.
저장 형식	Object는 모두 가능.(일반적으로 Dto, List등 저장.)	file에 저장되기 때문에 String형태.
사용 예	로그인 시 사용자 정보, 장바구니 등.	최근 본 상품 목록, 아이디저장(자동로그인), 팝업메뉴에서 '오늘은 그만 열기'등.
용량제한	제한 없음.	도메인당 20개, 1쿠키당 4KB
만료시점	알 수 없음(Client가 로그아웃 하거나, 이정 시간동안 session에 접근 하지 않을 경우 [만료 시간은 web.xml에 설정.])	쿠키 저장 시 설정(설정이 없을 경우 Browser 종료 시 만료)
공통	전역에 저장하기 때문에 project내의 모든 JSP에서 사용가능.	
	Map형식으로 관리하기 때문에 key값의 중복을 허용하지 않는다.	

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

EL
(Expression Language)

삼성 청년 SW 아카데미

EL (Expression Language)

- ✓ EL (Expression Language).
  - EL은 표현을 위한 언어로 JSP 스크립트의 표현식을 대신하여 속성 값을 쉽게 출력하도록 고안된 language 이다.
  - 즉 표현식 (<%= %>)를 대체할 수 있다.
  - EL 표현식에서 도트 연산자 왼쪽은 반드시 java.util.Map 객체 또는 Java Bean 객체여야 한다.
  - EL 표현식에서 도트 연산자 오른쪽은 반드시 맵의 키이거나 Bean 프로퍼티여야 한다.

  - EL에서 제공하는 기능.
    - JSP의 네가지 기본 객체가 제공하는 영역의 속성 사용.
    - 자바 클래스 메소드 호출 기능.
    - 표현 언어만의 기본 객체 제공.
    - 수치, 관계, 논리 연산 제공.

EL (Expression Language)

✓ EL 문법.

스크립트릿

```
<%=  ((com.ssafy.model.MemberDto)
request.getAttribute(“userinfo")).getZipDto().getAddress() %>
```

Expression Language

```
${userinfo.zipDto.address}
```

[EL 문법] Map을 사용하는 경우

${ Map . Map의 키 }

[EL 문법] Java Bean을 사용하는 경우

${ Java Bean . Bean 프로퍼티 }

EL (Expression Language)

✓ **EL 문법 : [ ] 연산자.**

- EL에는 Dot 표기법 외에 [ ] 연산자를 사용하여 객체의 값에 접근할 수 있다.
- [ ] 연산자 안의 값이 문자열인 경우, 이것은 맵의 키가 될 수도 있고, Bean 프로퍼티나 리스트 및 배열의 인덱스가 될 수 있다.
- 배열과 리스트인 경우, 문자로 된 인덱스 값은 숫자로 변경하여 처리합니다.

[] 연산자를 이용한 객체 프로퍼티 접근	Dot 표기법을 이용한 객체 프로퍼티 접근
`${userinfo["name"]}`	`${userinfo.name}`

리스트나 배열 요소에 접근

```
// Servlet
String[] names = {"홍길동", "이순신", "임꺽정"};
request.setAttribute("userNames", names);

// JSP
${userNames[0]}   // 홍길동 출력.
${userNames["1"]} // 문자열인 인덱스 값이 숫자로 바뀌어 userNames[1]의 결과인 이순신 출력.
```

EL (Expression Language)

✓ **EL 내장객체.**

category	identifier	Type	description
JSP	pageContext	Java Bean	현재 페이지의 프로세싱과 상응하는 PageContext instance.
범위 (scope)	pageScope	Map	page scope에 저장된 객체를 추출.
	requestScope	Map	request scope에 저장된 객체를 추출.
	sessionScope	Map	session scope에 저장된 객체를 추출.
	applicationScope	Map	application scope에 저장된 객체를 추출.

EL (Expression Language)

✓ **EL 내장객체** : EL 내장객체는 JSP 페이지의 EL 표현식에서 사용할 수 있는 객체.

category	identifier	Type	description
요청 매개변수	param	Map	ServletRequest.getParameter(String)을 통해 요청 정보를 추출.
	paramValues	Map	ServletRequest.getParameterValues(String)을 통해 요청 정보를 추출.
요청 헤더	header	Map	HttpServletRequest.getHeader(String)을 통해 헤더 정보를 추출.
	headerValues	Map	HttpServletRequest.getHeaders(String)을 통해 헤더 정보를 추출.
쿠키	cookie	Map	HttpServletRequest.getCookies()를 통해 쿠키 정보를 추출.
초기화 매개변수	initParam	Map	ServletContext.getInitParameter(String)를 통해 초기화 파라미터를 추출.

EL (Expression Language)

- ✓ EL 사용.
  - pageContext를 제외한 모든 EL 내장 객체는 Map이다.
  - 그러므로 key와 value의 쌍으로 값을 저장하고 있다.
  - 기본 문법

```
${ expr }
```

EL (Expression Language)

✓ EL에서 객체 접근.

- request.setAttribute("userinfo", "안효인");
  1. ${requestScope.userinfo}
  2. ${pageContext.request.userinfo} , ${userinfo}

property이름만 사용 할 경우 자동으로 pageScope > requestScope > sessionScope > applicationScope 순으로 객체를 찾음.

- url?name=안효인&fruit=사과&fruit=바나나
  1. ${param.name}
  2. ${paramValues.fruit[0]}. ${paramValues.fruit[1]}

EL에서 request 객체 접근

```
Method is : ${pageContext.request.method}

// Servlet
request.setAttribute("ssafy.user", memberDto);

// Case #1 : 에러
${ssafy.user.name} // ssafy라는 속성은 존재하지 않음

// Case #2 : request 내장객체에서 []연산자를 통해 속성 접근
${requestScope["ssafy.user"].name}
```

EL (Expression Language)

✓ **EL에서 객체 접근.**

- ${cookie.id.value}
  1. Cookie가 null이라면 null return.
  2. null이 아니라면 id를 검사 후 null이라면 null return.
  3. null이 아니라면 value값 검사.

※ EL은 값이 null이라도 null을 출력 하지 않는다. (공백)

스크립트릿을 통한 쿠키 값 출력

```
Cookie[] cookies = request.getCookies();
for(Cookie cookie : cookies) {
if(cookie.getName().equals("userId")) {
    out.println(cookie.getValue());
}
```

EL 내장객체를 통한 쿠키 값 출력

```
${cookie.userId.value}
```

EL (Expression Language)

✓ **EL Operator(연산자).**

- 대부분 java와 동일.

	description
산술	+, -, *, / (div), % (mod)
관계형	== (eq), != (ne), < (lt), > (gt), <= (le), >= (ge)
3항 연산	조건 ? 값1 : 값2
논리	&& (and), || (or), ! (not)
타당성검사	empty

※ empty 연산자에서 true를 return 하는 경우. >> ${empty var}

1. 값이 null이면 true
2. 값이 빈 문자열("") 이면 true
3. 길이가 0인 배열([]) 이면 true
4. 빈 Map 객체는 true
5. 빈 Collection 객체이면 true

EL (Expression Language)

✓ EL에서 객체 method 호출.

```
<%
List<MemberDto> list = dao.getMembers();
request.setAttribute("users", list);
%>
```

- 회원 수 : ${ requestScope.users.size() }, ${ users.size() }

※ 주의 >> ${ users.size } == <%= request.getAttribute("users").getSize() %>

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

이어서..

삼성 청년 SW 아카데미

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

JSTL

(Jsp Standard Tag Library)

삼성 청년 SW 아카데미

JSTL (Jsp Standard Tag Library)

✓ JSTL (JSP Standard Tag Library).

자바서버 페이지 표준 태그 라이브러리(JavaServer Pages Standard Tag Library, 약칭 JSTL)은 Java EE 기반의 웹 애플리케이션 개발 플랫폼을 위한 컴포넌트 모음이다. JSTL은 XML 데이터 처리와 조건문, 반복문, 국제화와 지역화 같은 일을 처리하기 위한 JSP 태그 라이브러리를 추가하여 JSP 사양을 확장했다. JSTL은 JSR 52로서 JCP 하에서 개발되었으며, 2006년 5월 8일에 JSTL 1.2가 출시되었다.

JSTL은 JSP 페이지 내에서 자바 코드를 바로 사용하지 않고 로직을 내장하는 효율적인 방법을 제공한다. 표준화된 태그 셋을 사용하여 자바 코드가 들락거리는 것보다 더 코드의 유지보수와 응용 소프트웨어 코드와 사용자 인터페이스 간의 관심사의 분리로 이어지게 한다.

[출처 : 위키백과]

JSTL (Jsp Standard Tag Library)

- ✓ JSTL.
  - custom tag : 개발자가 직접 태그를 작성할 수 있는 기능을 제공.
  - custom tag 중에서 많이 사용되는 것들을 모아서 JSTL이라는 규약을 만듦.
  - 논리적인 판단, 반복문의 처리, 데이터베이스 등의 처리를 할 수 있다.
  - JSP 2.1 ~ JSP 2.2 와 호환되는 JSTL 버전은 1.2 이다.
  - JSTL은 JSP 페이지에서 스크립트릿을 사용하지 않고 액션을 통해 간단하게 처리할 수 있는 방법을 제공.
  - JSTL에는 다양한 액션이 있으며, EL과 함께 사용하여 코드를 간결하게 작성할 수 있다.
  - https://mvnrepository.com/artifact/javax.servlet/jstl >> jstl-1.2.jar download

JSTL (Jsp Standard Tag Library)

- ✓ JSTL Tag.
  - directive 선언 형식 : <%@ taglib prefix="prefix" uri="uri" %>

library	prefix	function	URI
core	c	변수 지원, 흐름제어, URL처리	http://java.sun.com/jsp/jstl/core
XML	x	XML 코어, 흐름제어, XML변환	http://java.sun.com/jsp/jstl/xml
국제화	fmt	지역, 메시지 형식, 숫자 및 날짜 형식	http://java.sun.com/jsp/jstl/fmt
database	sql	SQL	http://java.sun.com/jsp/jstl/sql
함수		Collection, String 처리	http://java.sun.com/jsp/jstl/functions

JSTL (Jsp Standard Tag Library)

✓ JSTL - core tag.

- 선언 : <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

function	tag	description
변수지원	set	jsp page에서 사용 할 변수 설정.
	remove	설정한 변수를 제거.
흐름제어	if	조건에 따른 코드 실행.
	choose, when, otherwise	다중 조건을 처리할 때 사용.(if ~ else if ~ else)
	forEach	array나 collection의 각 항목을 처리할 때 사용.
	forTokens	구분자로 분리된 각각의 토큰을 처리할 때 사용.(StringTokenizer)
URL처리	import	URL을 사용하여 다른 자원의 결과를 삽입.
	redirect	지정한 경로로 redirect.
	url	URL 작성.
기타태그	catch	Exception 처리에 사용.
	out	JspWriter에 내용을 처리한 후 출력.

JSTL (Jsp Standard Tag Library)

✓ 변수 선언 : <c:set>

- <c:set> 액션은 변수나 특정 객체의 프로퍼티에 값을 할당할 때 사용.
- value 속성의 값이나 액션의 Body content로 값을 설정.
- var 속성은 변수를 나타내며, 변수의 생존범위는 scope 속성으로 설정. (디폴트는 page)
- 특정 객체의 프로퍼티에 값을 할당할 때는 target 속성에 객체를 설정하고 property에 프로퍼티명을 설정.

value 속성을 이용하여 생존범위 변수 값 할당

```
<c:set value="value" var="varName" [scope="{page|request|session|application}"]/>
```

액션의 Body 컨텐츠를 사용하여 생존범위 변수 값 할당

```
<c:set var="varName" [scope="{page|request|session|application}"]>
body content
</c:set>
```

value 속성을 이용하여 대상 객체의 프로퍼티 값 할당

```
<c:set value="value" target="target" property="propertyName"/>
```

액션의 Body 컨텐츠를 사용하여 대상 객체의 프로퍼티 값 할당

```
<c:set target="target" property="propertyName">
body content
</c:set>
```

JSTL (Jsp Standard Tag Library)

✓ 예외 : <c:catch>

- 기본적으로 JSP 페이지는 예외가 발생하면 지정된 오류페이지를 통해 처리한다.
- <c:catch> 액션은 JSP 페이지에서 예외가 발생할 만한 코드를 오류페이지로 넘기지 않고 직접 처리할 때 사용.
- var 속성에는 발생한 예외를 담을 page 생존범위 변수를 지정.
- <c:catch>와 <c:if> 액션을 함께 사용하여 Java 코드의 try~catch 와 같은 기능을 구현할 수 있다.

try ~ catch 구문

```
try {
    String str = null;
    out.println("Length of string : " +
                str.length());
} catch(Throwable ex) {
    out.print(ex.getMessage());
}
```

<c:catch>와 <c:if> 사용하기

```
<%@ page contentType="text/html" pageEncoding="UTF-8"
         errorPage="error.jsp" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<c:catch var="ex">
<%
    String str = null;
    out.println("Length of string : " + str.length());// 예외 발생!!
%>
</c:catch>

<c:if test="${ex != null}">
    예외가 발생하였습니다. ${ex.message}
</c:if>
```

JSTL (Jsp Standard Tag Library)

✓ 조건문 : <c:if>, <c:choose><c:when><c:otherwise>

- <c:if> 액션은 test 속성에 지정된 표현식을 평가하여 결과가 true인 경우 액션의 Body 컨텐츠를 수행.
- <c:if> 액션의 var 속성은 표현식의 평가 결과인 Boolean 값을 담을 변수를 나타내며, 변수의 생존범위는 scope 속성으로 설정.
- <c:choose><c:when><c:otherwise> 액션을 사용하면 if, else if, else 와 같이 처리할 수 있다.

<c:if> 액션 사용 예

```
<c:if test="${userType eq 'admin'}">
    <jsp:include page="admin.jsp" />
</c:if>
```

<c:if> 액션의 var 속성

```
<c:if test="${userType eq 'admin'}" var="accessible">
    <jsp:include page="admin.jsp" />
</c:if>

...
<c:if test="${category == 'user' && menu == 'list'}">
   회원 목록.
</c:if>
```

<c:choose>, <c:when>, <c:otherwise> 액션 사용 예

```
<c:choose>
    <c:when test="${userType == 'admin'}">
        관리자 화면….
   </c:when>

   <c:when test="${userType == 'member'}">
        회원 사용자 화면…
   </c:when>

   <c:otherwise>
        일반 사용자 화면..
   </c:otherwise>
</c:choose>
```

JSTL (Jsp Standard Tag Library)

✓ 반복문 : <c:forEach>

- <c:forEach> 액션은 컬렉션에 있는 항목들에 대하여 액션의 Body 컨텐츠를 반복하여 수행.
- 컬렉션에는 Array, Collection, Map 또는 콤마로 분리된 문자열이 올 수 있다.
- var 속성에는 반복에 대한 현재 항목을 담는 변수를 지정하고 items 속성은 반복할 항목들을 갖는 컬렉션을 지정.
- varStatus 속성에 지정한 변수를 통해 현재 반복의 상태를 알 수 있다.

course

0	Java Fundamental
1	Java IO
2	Java Socket
3	JSP&Servlet
4	Spring MVC
5	JavaScript
6	jQuery Basic
7	MyBatis

<c:forEach> 액션의 다양한 활용

```
<b>1) courses 리스트를 반복하면서 과정명 출력하기</b><br/>
<c:forEach var="course" items="${courses}">
    ${course.name}<br/>
</c:forEach>

<b>2) courses 리스트를 반복하면서 순번과 과정명 출력하기</b><br/>
<c:forEach var="course" items="${courses}" varStatus="varStatus">
    ${varStatus.count}. ${course.name}<br/>
</c:forEach>

<b>3) 짝수번째 과정명만 출력하기</b><br/>
<c:forEach var="course" items="${courses}" begin="0" end="5" step="2">
    ${course.name}<br/>
</c:forEach>

<b>4) 숫자 1부터 5까지 출력하기</b><br/>
<c:forEach var="value" begin="1" end="5" step="1">
    ${value}<br/>
</c:forEach>
```

실행결과

1) courses 리스트를 반복하면서 과정명 출력하기
Java Fundamental
Java IO
Java Socket
JSP&Servlet
Spring MVC
JavaScript
jQuery Basic
MyBatis
2) courses 리스트를 반복하면서 순번과 과정명 출력하기
1. Java Fundamental
2. Java IO
3. Java Socket
4. JSP&Servlet
5. Spring MVC
6. JavaScript
7. jQuery Basic
8. MyBatis

실행결과

3) 짝수번째 과정명만 출력하기
Java Fundamental
Java Socket
Spring MVC
4) 숫자 1부터 5까지 출력하기
1
2
3
4
5

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

내일 방송에서 만나요!

삼성 청년 SW 아카데미