

# 04장. 내부객체

강사 강태광

# 내부 객체의 개요

1. 내부 객체는 JSP 페이지 내에서 특정한 변수로 호출하고자 하는 변수와 메서드로 접근을 한다.
2. JSP 페이지에서 사용하게 되는 특정한 변수가 아무런 선언과 객체 생성 없이 사용할 수 있는 이유는 JSP 페이지가 서블릿으로 변환이 될 때 JSP 컨테이너가 자동적으로 제공을 하기 때문이다.
3. 9개의 내부 객체의 사용되는 범주에 따라 4가지 형태로 분류.
  - JSP 페이지 입출력 관련 기본 객체
  - JSP 페이지 외부 환경 정보 제공 기본 객체
  - JSP 페이지 서블릿 관련 기본 객체
  - JSP 페이지 예외 관련 기본 객체
4. 내부 객체의 종류  
request, response, out, pageContext, session, application, config, page, exception

# 내부 객체가 제공하는 메서드

request, session, application, pageContext 내부 객체는 임의 속성(attribute) 값을 저장하고 읽을 수 있는 메서드를 제공

메서드	리턴 타입	설명
setAttribute (String key, Object value)	void	주어진 key속성의 값을 value로 지정한다.
getAttributeNames()	java.util.Enumeration	모든 속성의 이름을 구한다.
getAttribute (String key)	Object	주어진 key속성의 값을 얻어낸다.
removeAttribute (String key)	void	주어진 key속성의 값을 제거한다.

# request 내부 객체

- 1.request 객체는 웹 브라우저에서 JSP 페이지로 전달되는 정보의 모임으로 HTTP 헤더와 HTTP 바디로 구성
2. 웹 컨테이너는 요청된 HTTP 메시지를 통해 HttpServletRequest 객체 타입으로 사용되고 request 객체 명으로 사용

메서드	설명
<code>String getParameter(name)</code>	이름이 name인 파라미터에 할당된 값을 리턴 하며 지정된 파라미터 값이 없으면 null값을 리턴 한다.
<code>String[] getParameterValues(name)</code>	이름이 name인 파라미터의 모든 값을 String 배열로 리턴 한다. checkbox에서 주로 사용된다.
<code>Enumeration getParameterNames()</code>	요청에 사용된 모든 파라미터 이름을 <code>java.util.Enumeration</code> 타입으로 리턴 한다.

# request 내부객체 클라이언트 정보 메서드

1. request 객체는 또한 웹 브라우저와 웹 서버의 정보도 가져 올 수가 있다
2. request 내부 객체의 클라이언트 정보 메서드

메서드	설명
String getMethod()	요청에 사용된 요청 방식(GET, POST, PUT)을 리턴 한다.
String getRequestURI()	요청에 사용된 URL로부터 URI을 리턴 한다.
String getQueryString()	요청에 사용된 Query 문장을 리턴 한다.
String getRemoteHost()	클라이언트의 호스트 이름을 리턴 한다.
String getRemoteAddr()	클라이언트의 주소를 리턴 한다.
String getProtocol()	사용 중인 프로토콜을 리턴 한다.
String getServerName()	서버의 도메인 이름을 리턴 한다.
int getServerPort()	서버의 port번호를 리턴 한다.
String getHeader(name)	HTTP 요청 헤더에 지정된 name의 값을 리턴 한다.
String getContextPath()	해당 JSP페이지가 속한 웹 어플리케이션의 컨텍스트 경로를 리턴 한다.

# request 내부객체 클라이언트 정보 메서드

1. request 객체는 또한 웹 브라우저와 웹 서버의 정보도 가져 올 수가 있다
2. request 내부 객체의 클라이언트 정보 메서드

메서드	설명
String getMethod()	요청에 사용된 요청 방식(GET, POST, PUT)을 리턴 한다.
String getRequestURI()	요청에 사용된 URL로부터 URI을 리턴 한다.
String getQueryString()	요청에 사용된 Query 문장을 리턴 한다.
String getRemoteHost()	클라이언트의 호스트 이름을 리턴 한다.
String getRemoteAddr()	클라이언트의 주소를 리턴 한다.
String getProtocol()	사용 중인 프로토콜을 리턴 한다.
String getServerName()	서버의 도메인 이름을 리턴 한다.
int getServerPort()	서버의 port번호를 리턴 한다.
String getHeader(name)	HTTP 요청 헤더에 지정된 name의 값을 리턴 한다.
String getContextPath()	해당 JSP페이지가 속한 웹 어플리케이션의 컨텍스트 경로를 리턴 한다.

# request 내부객체 클라이언트 정보 메서드

## RequestTest2.jsp(예시)

```
<%@ page contentType="text/html; charset=euc-kr"%>
<%
```

```
String protocol = request.getProtocol();
String server = request.getServerName();
int port = request.getServerPort();
String clientIp = request.getRemoteAddr();
String clientHost = request.getRemoteHost();
String methodType = request.getMethod();
String url = new String(request.getRequestURL());
String uri = request.getRequestURI();
String contextPath = request.getContextPath();
String browser = request.getHeader("User-Agent");
String mediaType = request.getHeader("Accept");
```

```
%>
```

```
<h2>Request내장 객체 예제2</h2>
```

```
프로토콜명: <%=protocol%> <p>
```

```
접속한 서버명 : <%=server%> <p>
```

```
접속한 서버의 포트 번호 : <%=port%> <p>
```

```
클라이언트의 IP : <%=clientIp%> <p>
```

```
클라이언트의 호스트명 : <%=clientHost%> <p>
```

```
현재 페이지의 method방식 : <%=methodType%> <p>
```

```
요청한 현재 페이지의 경로(URL) : <%=url%> <p>
```

```
요청한 현재 페이지의 경로(URI) : <%=uri%> <p>
```

```
웹어플리케이션에서의 컨텍스트 경로 : <%=contextPath%> <p>
```

```
사용한 웹 브라우저 : <%=browser%> <p>
```

```
웹 브라우저가 지원하는 매체(media)의 타입 : <%=mediaType%> <p>
```

# response 내부 객체

1. response 객체는 웹 브라우저로 응답할 응답 정보를 가지고 있음.
2. 웹 컨테이너는 요청된 HTTP 메시지를 통해 HttpServletResponse 객체 타입으로 사용되고 response 객체명으로 사용.
3. response 객체는 응답정보와 관련하여 주로 헤더 정보 입력, 리다이렉트 하기등의 기능을 제공

메서드	설명
<code>void setHeader(name, value)</code>	응답에 포함될 Header를 설정한다.
<code>void setContentType(type)</code>	출력되는 페이지의 contentType 설정한다.
<code>void sendRedirect(url)</code>	지정된 URL로 요청을 재전송한다



# pageContext 내부 객체

1. PageContext 객체는 현재 JSP 페이지의 컨텍스트를 나타내며 PageContext 내부 객체를 통해서 다른 내부 객체에 접근을 할 수가 있음.
2. pageContext 객체는 javax.servlet.jsp.PageContext 클래스 타입으로 제공 되는 JSP 기본 객체.

메서드	설명
<code>ServletRequest getRequest()</code>	페이지 요청 정보를 가지고 있는 request 기본 객체를 리턴 한다.
<code>ServletResponse getResponse()</code>	페이지 요청에 대한 응답 정보를 가지고 있는 response 기본 객체를 리턴 한다.
<code>JspWriter getOut()</code>	페이지 요청에 대한 응답 출력 스트림인 out 기본 객체를 리턴 한다.
<code>HttpSession getSession()</code>	요청한 클라이언트의 세션 정보를 담고 있는 session 기본 객체를 리턴 한다.
<code>ServletContext getServletContext()</code>	페이지에 대한 서블릿 실행 환경 정보를 담고 있는 application 기본 객체를 리턴 한다.
<code>Object getPage()</code>	page 기본 객체를 리턴 한다.
<code>ServletConfig getServletConfig()</code>	페이지의 서블릿 초기 정보 설정 정보를 담고 있는 config 기본 객체를 리턴 한다.
<code>Exception getException()</code>	페이지 실행 중에 발생하는 예외 페이지에 대한 예외정보를 갖고 있는 exception 기본 객체를 리턴 한다.

# 내부객체의 영역(scope)

1. 웹 어플리케이션은 page, request, session, application 이라는 4개의 영역을 가지고 있음.
2. 기본객체의 영역은 객체의 유효기간이라고도 불리며, 객체를 누구와 공유할 것인가를 나타냄.
  - 1) page영역
    - 한번의 웹 브라우저(클라이언트)의 요청에 대해 하나의 JSP페이지가 호출.
    - 웹 브라우저의 요청이 들어오면 이때 단 한 개의 페이지만 대응.
  - 2) request영역
    - 한번의 웹 브라우저(클라이언트)의 요청에 대해 같은 요청을 공유하는 페이지가 대응.
    - 웹 브라우저의 한번의 요청에 단지 한 개의 페이지만 요청될 수 있고, 같은 request영역이면 두개의 페이지가 같은 요청을 공유 . include 액션 태그, forward 액션 태그를 사용시
  - 3) session영역
    - 하나의 웹 브라우저당 1개의 session객체가 생성.
    - 같은 웹 브라우저 내에서는 요청되는 페이지 들은 같은 객체를 공유.
  - 4) application영역
    - 하나의 웹 어플리케이션당 1개의 application 객체가 생성. 같은 웹 어플리케이션에 요청되는 페이지들은 같은 객체를 공유.

# JSP 페이지의 연산자 |

1. 자바의 기본적인 문법을 그대로 사용
2. 식별자(identifier)규칙
  - ① JSP는 자바 식별자 규칙을 따르는데, 식별자(identifier)란 클래스명, 메소드명, 멤버변수명, 자동변수명등을 일컫는다.
  - ② 자바 식별자는 길이에겐 제한이 없고 첫 글자는 반드시 영문자, \_,\$로 시작해야 한다.  
자바의 클래스명, 메소드명, 인스턴스변수, 자동변수등에 자바 식별자 규칙이 적용
  - ③ 자바는 대소문자를 구별하므로 주의해야한다.
3. 기본데이터타입(primitive data type)
  - 자바 및 JSP는 byte, short, int, long, float, double, char, boolean등의 기본 데이터타입(primitive data type)을 제공한다

타입	크기(byte)	자료범위	기본값
byte	1byte	-128 ~ +127	0
short	2byte	-32,768 ~ +32,767	0
int	4byte	-2,147,243,648 ~ +2,147,243,647	0
long	8byte	-9,223,372,036,854,775,808 ~ +9,223,372,036,854,775,807	0
float	4byte	-3.40292347E+38 ~ +3.40292347E+38	0
double	8byte	-1.79769313486231570E+308 ~ +1.79769313486231570E+308	0
char	2byte	'\u0000' ~ '\uFFFF'	0
boolean	1bit	true or false	false

# JSP 페이지의 연산자2

1. JSP에서는 자바와 마찬가지로 산술연산자, 관계연산자, 논리연산자, 비트연산자, shift연산자, 증감연산자, 조건연산자, 대입연산자들을 사용할 수 있다.

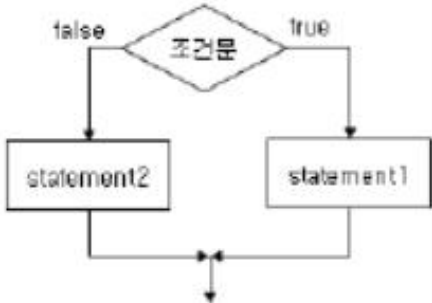
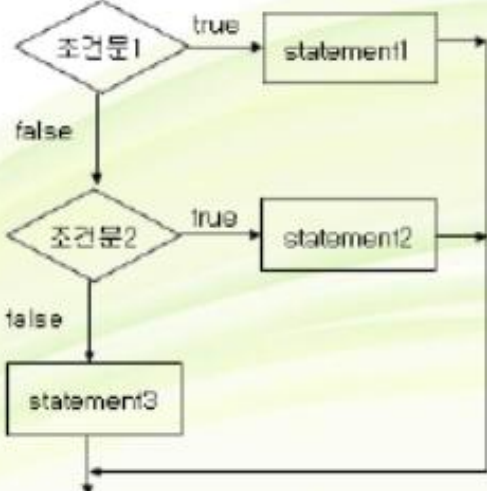
## 2. 연산자의 종류

- 산술연산자: `*`, `/`, `%`, `+`, `-`
- 관계연산자: `<`, `>`, `<=`, `>=`
- 논리연산자: `&&`, `||`, `!`
- 비트연산자: `&`, `|`, `^`
- shift연산자: `<<`, `>>`, `>>>`
- 증감연산자: `++`, `--`
- 조건연산자: `?:`
- 대입연산자: `=`, `+=`, `-=`, `*=`, `/=`, `%=`

# JSP페이지의 제어문

## 1. if문

- ① if문은 조건비교 분기문의 하나로 주어진 조건을 비교해서 그 결과에 따라 여러 대안들 중에서 하나를 선택할 때 사용된다.
- ② if문의 조건에 들어갈 수 있는 타입은 리턴타입 또는 결과 값이 boolean 값일 경우만 가능하다

문법	순서도(Flowchart)	문법	순서도(Flowchart)
<pre>if(조건){     statement1; }else{     statement2; }</pre>		<pre>if(조건1){     statement1; }else if(조건2){     statement2; }else{     statement3; }</pre>	

# 웹 애플리케이션 배포

1. 모든 파일을 직접 배포 가능
2. war(Web Application Archive) 파일로 압축해서 배포
  - Eclipse에서는 [File] – [Export] – [WAR file]을 선택해서 생성이 가능

