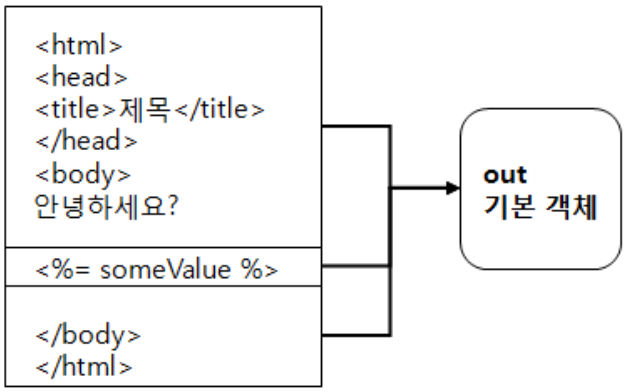
5장 기본 객체와 영역

5.1 기본 객체

기본 객체	실제 타입	설명
request	javax.servlet.http.HttpServletRequest 또는 javax.servlet.ServletRequest	클라이언트의 요청 정보를 저장한다.
response	javax.servlet.http.HttpServletResponse 또는 javax.servlet.ServletResponse	응답 정보를 저장한다.
pageContext	javax.servlet.jsp.PageContext	JSP 페이지에 대한 정보를 저장한다.
session	javax.servlet.http.HttpSession	HTTP 세션 정보를 저장한다.
application	javax.servlet.ServletContext	웹 어플리케이션에 대한 정보를 저장한다.
out	javax.servlet.jsp.JspWriter	JSP 페이지가 생성하는 결과를 출력할 때 사용되는 출력 스트림이다.
config	javax.servlet.ServletConfig	JSP 페이지에 대한 설정 정보를 저장한다.
page	java.lang.Object	JSP 페이지를 구현한 자바 클래스 인스턴스이다.
exception	java.lang.Throwable	예외 객체. 에러 페이지에서만 사용된다.

5.2 out 기본 객체

■ JSP 페이지가 생성하는 모든 내용은 out 기본 객체를 통해 전송된다.



■ 복잡한 if-else 사용시 out 기본 객체 사용하면 편리

```
out.println("<html>");
out.println("<head>");
...

// 스크립트릿 + 표현식 사용
<% if (grade > 10 ) { %>
<%= gradeStringA %>
<% } else if (grade > 5 ) { %>
<%= gradeStringB %>
<% } %>

// 스크립트릿 + out객체 사용
<%
    if (grade > 10) {
        out.println(gradeStringA);
    } else if (grade > 5) {
        out.println(gradeStringB);
```

```
    }
%>
```

[chap05\useOutObject.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<html>
<head><title>out 기본 객체 사용</title></head>
<body>

<%
          out.println("안녕하세요?");
%>
<br>
out 기본 객체를 사용하여
<%
          out.println("출력한 결과입니다.");
%>

</body>
</html>
```

5.2.1 out 기본 객체의 출력 메서드

- print() - 데이터를 출력한다.
- println() - 데이터를 출력하고, \r\n(또는 \n)을 출력한다.
- newLine() - \r\n(또는 \n)을 출력한다.

5.2.2 out 기본 객체와 버퍼의 관계

- int getBufferSize() - 버퍼의 크기를 구한다.
- int getRemaining() - 현재 버퍼의 남은 크기를 구한다.
- clear() - 버퍼의 내용을 비운다. 만약 버퍼가 이미 플러시 되었다면 IOException을 발생시킨다.
- clearBuffer() - 버퍼의 내용을 비운다.
- flush() - 버퍼를 플러시 한다.
- boolean isAutoFlush() - 버퍼가 다 찼을 때 자동으로 플러시 할 경우 true를 리턴한다.

[chap05\bufferInfo.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ page buffer="8kb" autoFlush="false" %>
<html>
<head><title>버퍼 정보</title></head>
<body>

버퍼 크기: <%= out.getBufferSize() %> <br>
```

```
08    남은 크기: <%= out.getRemaining() %> <br>
09    auto flush: <%= out.isAutoFlush() %> <br>
10
11    </body>
12    </html>
```

5.3 pageContext 기본 객체

■ pageContext 기본 객체는 JSP 페이지와 일대일로 연결된 객체로 다음의 기능을 제공한다.
- 기본 객체 구하기
- 속성 처리하기
- 페이지의 흐름 제어하기
- 에러 데이터 구하기

5.3.1 기본 객체 접근 메서드

메서드	리턴타입	설명
getRequest()	ServletRequest	request 기본 객체를 구한다.
getResponse()	ServletResponse	response 기본 객체를 구한다.
getSession()	HttpSession	session 기본 객체를 구한다.
getServletContext()	ServletContext	application 기본 객체를 구한다.
getServletConfig()	ServletConfig	config 기본 객체를 구한다.
getOut()	JspWriter	out 기본 객체를 구한다.
getException()	Exception	exception 기본 객체를 구한다.
getPage()	Object	page 기본 객체를 구한다.

[chap05\usePageContext.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <html>
03    <head><title>pageContext 기본 객체</title></head>
04    <body>
05
06    <%
07              HttpServletRequest httpRequest =
08                        (HttpServletRequest)pageContext.getRequest();
09    %>
10
11    request 기본 객체와 pageContext.getRequest()의 동일여부:
12
13    <%= request == httpRequest %>
14
15    <br>
16
17    pageContext.getOut() 메서드를 사용한 데이터 출력:
18
19    <% pageContext.getOut().println("안녕하세요!"); %>
20    </body>
21    </html>
```

5.4 application 기본 객체

■ application 기본 객체는 웹 어플리케이션 전반에 걸쳐서 사용되는 정보를 담고 있다.

5.4.1 웹 어플리케이션 초기화 파라미터 읽어오기

■ 초기화 파라미터 설정: web.xml 파일 이용

```
<context-param>
    <description>파라미터 설명(필수 아님)</description>
    <param-name>파라미터이름</param-name>
    <param-value>파라미터값</param-value>
</context-param>
```

■ application 기본 객체의 초기화 파라미터 관련 기능

메서드	리턴타입	설명
getInitParameter(String name)	String	이름이 name인 웹 어플리케이션 초기화 파라미터의 값을 읽어온다. 존재하지 않을 경우 null을 리턴한다.
getInitParameterNames()	Enumeration	웹 어플리케이션 초기화 파라미터의 이름 목록을 리턴한다.

[chap05\WEB-INF\web.xml]

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
             http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         version="3.1">

          <context-param>
                    <description>로깅 여부</description>
                    <param-name>logEnabled</param-name>
                    <param-value>true</param-value>
          </context-param>

          <context-param>
                    <description>디버깅 레벨</description>
                    <param-name>debugLevel</param-name>
                    <param-value>5</param-value>
          </context-param>

</web-app>
```

[chap05\readInitParameter.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ page import = "java.util.Enumeration" %>
<html>
<head><title>초기화 파라미터 읽어오기</title></head>
<body>

초기화 파라미터 목록:
<ul>
<%
    Enumeration<String> initParamEnum = application.getInitParameterNames();
          while (initParamEnum.hasMoreElements()) {
                    String initParamName = initParamEnum.nextElement();
%>
<li><%= initParamName %> =
```

```
15        <%= application.getInitParameter(initParamName) %>
16    <%
17              }
18    %>
19    </ul>
20    </body>
21    </html>
```

5.4.2 서버 정보 읽어오기

■ 서버 정보 관련 메서드

메서드	리턴타입	설명
getServerInfo()	String	서버 정보를 구한다.
getMajorVersion()	String	서버가 지원하는 서블릿 규약의 메이저 버젼을 리턴한다. 버전의 정수 부분을 리턴한다.
getMinorVersion()	String	서버가 지원하는 서블릿 규약의 마이너 버젼을 리턴한다. 버전의 소수 부분을 리턴한다.

[chap05\viewServerInfo.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <html>
03    <head><title>서버 정보 출력</title></head>
04    <body>
05
06    서버정보: <%= application.getServerInfo() %> <br>
07    서블릿 규약 메이저 버전: <%= application.getMajorVersion() %> <br>
08    서블릿 규약 마이너 버전: <%= application.getMinorVersion() %>
09
10    </body>
11    </html>
```

5.4.3 로그 메시지 기록하기

■ 로그 기록 메서드

메서드	리턴타입	설명
log(String name)	void	msg를 로그로 남긴다.
log(String name, Throwable throwable)	void	msg를 로그로 남긴다. 익셉션 정보도 함께 로그에 기록한다.

[chap05\useApplicationLog.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <html>
03    <head><title>로그 메시지 기록</title></head>
04    <body>
05
06    <%
07              application.log("로그 메시지 기록");
08    %>
09    로그 메시지를 기록합니다.
10
11    </body>
12    </html>
```

[chap05\useJspLog.jsp] JSP 페이지가 제공하는 log 메서드를 이용해도 로그 메세지를 기록할 수 있다.

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <html>
03    <head><title>로그 메시지 기록2</title></head>
04    <body>
05
06    <%
07              log("로그 메시지 기록2");
08    %>
09    로그 메시지를 기록합니다.
10
11    </body>
12    </html>
```

5.4.4 웹 어플리케이션의 자원 구하기

■ 자원 접근 메서드

메서드	리턴타입	설명
getRealPath(String path)	String	웹 어플리케이션 내에서 지정한 경로에 해당하는 자원의 시스템상에서의 자원 경로를 리턴한다.
getResource(String path)	URL	웹 어플리케이션 내에서 지정한 경로에 해당하는 자원에 접근할 수 있는 URL 객체를 리턴한다.
getResourceAsStream(String path)	InputStream	웹 어플리케이션 내에서 지정한 경로에 해당하는 자원으로부터 데이터를 읽어올 수 있는 InputStream을 리턴한다.

[chap05\readFileDirectly.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <%@ page import = "java.io.*" %>
03    <html>
04    <head><title>절대 경로 사용하여 자원 읽기</title></head>
05    <body>
06
07    <%
08              char[] buff = new char[128];
09              int len = -1;
10
11              String filePath = "D:\\dev\\workspace\\jsp\\chap05\\WebContent\\message\\notice.txt";
12
13              try(InputStreamReader fr = new InputStreamReader(new FileInputStream(filePath), "UTF-8")) {
14                        while ( (len = fr.read(buff)) != -1) {
15                                  out.print(new String(buff, 0, len));
16                        }
17              } catch(IOException ex) {
18                        out.println("익셉션 발생: "+ex.getMessage());
19              }
20    %>
21
22    </body>
23    </html>
```

[chap05\readFileUsingApplication.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <%@ page import = "java.io.*" %>
03    <html>
04    <head><title>application 기본 객체 사용하여 자원 읽기</title></head>
```

```
05    <body>
06
07    <%
08              String resourcePath = "/message/notice.txt";
09    %>
10    자원의 실제 경로:<br>
11    <%= application.getRealPath(resourcePath) %>
12    <br>
13    ----------<br>
14    <%= resourcePath %>의 내용<br>
15    ----------<br>
16    <%
17              char[] buff = new char[128];
18              int len = -1;
19
20              try(InputStreamReader br = new InputStreamReader(
21                                        application.getResourceAsStream(resourcePath), "UTF-8")) {
22                        while ( (len = br.read(buff)) != -1) {
23                                  out.print(new String(buff, 0, len));
24                        }
25              } catch(IOException ex) {
26                        out.println("익셉션 발생: "+ex.getMessage());
27              }
28    %>
29
30    </body>
31    </html>
```

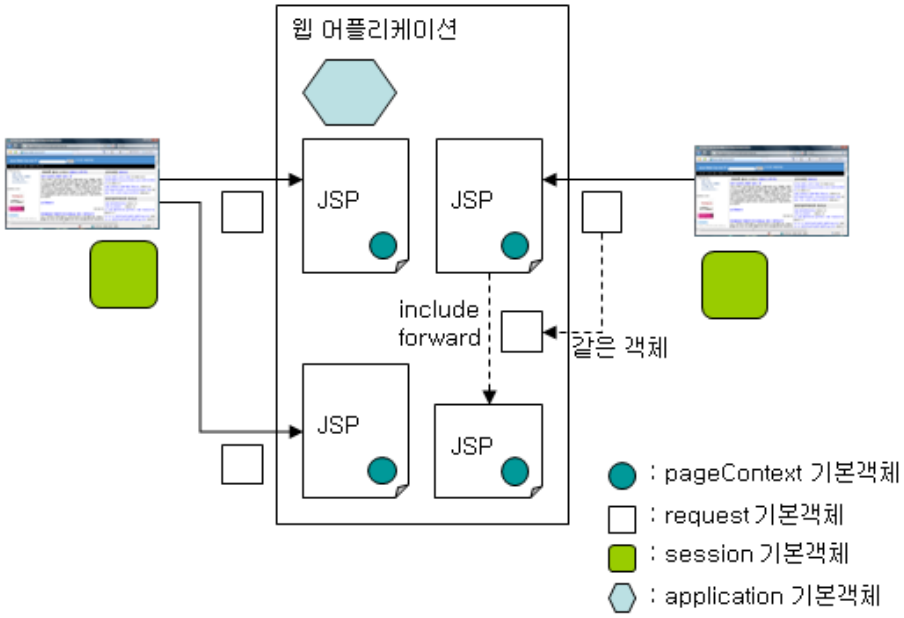
[chap05\readFileUsingURL.jsp] URL 객체를 리턴하는 application.getResource() 메서드를 사용

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <%@ page import = "java.io.*" %>
03    <%@ page import = "java.net.URL" %>
04    <html>
05    <head><title>application 기본 객체 사용하여 자원 읽기2</title></head>
06    <body>
07
08    <%
09              String resourcePath = "/message/notice.txt";
10              char[] buff = new char[128];
11              int len = -1;
12              URL url = application.getResource(resourcePath);
13              try (InputStreamReader br = new InputStreamReader(url.openStream(), "UTF-8")) {
14                        while ( (len = br.read(buff)) != -1) {
15                                  out.print(new String(buff, 0, len));
16                        }
17              } catch(IOException ex) {
18                        out.println("익셉션 발생: "+ex.getMessage());
19              }
20    %>
21
22    </body>
23    </html>
```

5.5 JSP 기본 객체와 영역

■ 웹 어플리케이션은 다음의 네 가지 영역(scope)으로 구성된다.

- PAGE 영역 : 하나의 JSP 페이지를 처리할 때 사용되는 영역
- REQUEST 영역 : 하나의 HTTP 요청을 처리할 때 사용되는 영역
- SESSION 영역 : 하나의 웹 브라우저와 관련된 영역
- APPLICATION 영역 : 하나의 웹 어플리케이션과 관련된 영역



5.6 기본 객체의 속성(Attribute) 사용하기

■ 속성 처리 메서드

메서드	리턴타입	설명
setAttribute(String name, Object value)	void	이름이 name인 속성의 값을 value로 지정한다.
getAttribute(String name)	Object	이름이 name인 속성의 값을 구한다. 지정한 이름의 속성이 존재하지 않을 경우 null을 리턴한다.
removeAttribute(String name)	void	이름이 name인 속성을 삭제한다.
getAttributeNames()	Enumeration	속성의 이름 목록을 구한다. (pageContext 기본 객체는 이 메서드를 제공하지 않는다.)

[chap05\setApplicationAttribute.jsp]

```
01   <%@ page contentType = "text/html; charset=utf-8" %>
02   <%
03            String name = request.getParameter("name");
04            String value = request.getParameter("value");
05
06            if (name != null && value != null) {
07                     application.setAttribute(name, value);
08            }
09   %>
10
11   <html>
12   <head><title>application 속성 지정</title></head>
13   <body>
14   <%
15            if (name != null && value != null) {
16   %>
17   application 기본 객체의 속성 설정: <br>
18    <%= name %> = <%= value %> <br>
19    <%= name %> = <%= application.getAttribute("userid") %>
20   <%
21            } else {
22   %>
```

```
23    application 기본 객체의 속성 설정 안 함
24    <%
25              }
26    %>
27    </body>
28    </html>
29
30
31
```

[chap05\viewApplicationAttribute.jsp]

```
01    <%@ page contentType = "text/html; charset=utf-8" %>
02    <%@ page import = "java.util.Enumeration" %>
03    <html>
04    <head><title>application 기본 객체 속성 보기</title></head>
05    <body>
06    <%
07              Enumeration<String> attrEnum = application.getAttributeNames();
08              while(attrEnum.hasMoreElements() ) {
09                        String name = attrEnum.nextElement();
10                        Object value = application.getAttribute(name);
11    %>
12    application 속성 : <b><%= name %></b> = <%= value %> <br>
13    <%
14              }
15    %>
16    </body>
17    </html>
```

5.6.1 속성의 값 타입

■ 속성의 이름은 문자열을 나타내는 String 타입이지만, 값은 모든 클래스 타입이 올 수 있다.

```
public void setAttribute(String name, Object value)
public Object getAttribute(String name)

session.setAttribute("session_start", new java.util.Data());
Date date = (Data) session.getAttribute("session_start");

application.setAttribute("application_temp", new File("C:\\temp"));
File tempDir = (File) application.getAttribute("application_temp");

request.setAttribute("age", 20);
int age = (Integer) request.getAttribute("age");
```

5.6.2 속성의 활용

기본 객체	영역	쓰임새
pageContext	PAGE	(한번의 요청을 처리하는) 하나의 JSP 페이지 내에서 공유될 값을 저장한다.
request	REQUEST	한번의 요청을 처리하는 데 사용되는 모든 JSP 페이지에서 공유될 값을 저장한다.
session	SESSION	한 사용자와 관련된 정보를 JSP 들이 공유하기 위해서 사용된다.
application	APPLICATION	모든 사용자와 관련해서 공유할 정보를 저장한다.