JSP 내장객체

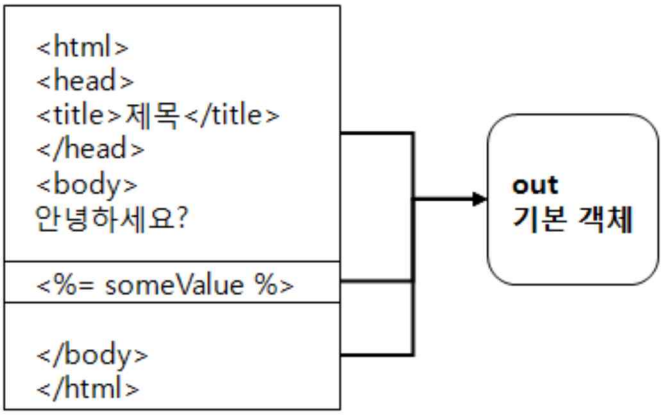
CONTENTS

■ 주요 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.

■ out 기본 객체

- **JSP 페이지가 생성하는 모든 내용은 out 기본 객체를 통해서 전송**



- **복잡한 if-else 사용시 out 기본 객체 사용하면 편리**

```
<%
    if (grade > 10) {
        out.println(gradeStringA);
    } else if (grade > 5) {
        out.println(gradeStringB);
    }
%>
```

■ out 기본 객체

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

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

</body>
</html>
```

■ out 기본 객체 주요 메서드

- 출력 메서드
  - print() - 데이터를 출력한다.
  - println() - 데이터를 출력하고, ₩r₩n(또는 ₩n)을 출력한다.
  - newLine() - ₩r₩n(또는 ₩n)을 출력한다.
- 버퍼 관련 메서드
  - 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.

■ out 기본 객체 주요 메서드

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

버퍼 크기: <%= out.getBufferSize() %> <br>
남은 크기: <%= out.getRemaining() %> <br>
auto flush: <%= out.isAutoFlush() %> <br>

</body>
</html>
```

■ pageContext 기본 객체

- **다른 기본 객체에 대한 접근 메서드 제공**

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.

■ pageContext 기본 객체

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<html>
<head><title>pageContext 기본 객체</title></head>
<body>
<%
        HttpServletRequest httpRequest =
                (HttpServletRequest)pageContext.getRequest();
%>
request 기본 객체와 pageContext.getRequest()의 동일여부:
<%= request == httpRequest %>
<br>
pageContext.getOut() 메서드를 사용한 데이터 출력:
<% pageContext.getOut().println("안녕하세요!"); %>
</body>
</html>
```

■ application 기본 객체: 초기화 파라미터

- **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.

■ application 기본 객체: 초기화 파라미터

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

<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
                      http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0" metadata-complete="true">

        <context-param>
                <description>?? ??</description>
                <param-name>logEnabled</param-name>
                <param-value>true</param-value>
        </context-param>
        <context-param>
                <description>??? ??</description>
                <param-name>debugLevel</param-name>
                <param-value>5</param-value>
        </context-param>

</web-app>
```

■ application 기본 객체: 초기화 파라미터

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import = "java.util.Enumeration" %>
<html>
<head><title>초기화 파라미터 읽어오기</title></head>
<body>
초기화 파라미터 목록:
<ul>
<%
        Enumeration initParamEnum = application.getInitParameterNames();
        while (initParamEnum.hasMoreElements()) {
                String initParamName = (String)initParamEnum.nextElement();
%>
<li><%= initParamName %> =
        <%= application.getInitParameter(initParamName) %>
<%
        }
%>
</ul>
</body>
</html>
```

■ application 기본 객체: 자원 구하기

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import = "java.io.*" %>
<html>
<head><title>절대 경로 사용하여 자원 읽기</title></head>
<body>
<%      FileReader fr = null;
        char[] buff = new char[512];
        int len = -1;
        try {       fr = new FileReader(
                            "C:₩₩apache-tomcat-7.0.27₩₩webapps₩₩chap06"+
                            "₩₩message₩₩notice₩₩notice.txt");
                    while ( (len = fr.read(buff)) != -1) {
                            out.print(new String(buff, 0, len));
                    }
        } catch(IOException ex) {
                    out.println("익셉션 발생: "+ex.getMessage());
        } finally {
                    if (fr != null) try { fr.close(); } catch(IOException ex) {}
        }
%></body></html>
```

■ application 기본 객체: 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.

■ application 기본 객체: 자원 구하기

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import = "java.io.*" %>
<html>
<head><title>application 기본 객체 사용하여 자원 읽기</title></head>
<body>

<%
        String resourcePath = "/message/notice/notice.txt";
%>
자원의 실제 경로:<br>
<%= application.getRealPath(resourcePath) %>
<br>
----------<br>
<%= resourcePath %>의 내용<br>
----------<br>
<%
        BufferedReader br = null;
        char[] buff = new char[512];
        int len = -1;
```

■ application 기본 객체: 자원 구하기

```
        try {
                br = new BufferedReader(
                        new InputStreamReader(
                                application.getResourceAsStream(resourcePath) ));
                while ( (len = br.read(buff)) != -1) {
                        out.print(new String(buff, 0, len));
                }
        } catch(IOException ex) {
                out.println("익셉션 발생: "+ex.getMessage());
        } finally {
                if (br != null) try { br.close(); } catch(IOException ex) {}
        }
%>

</body>
</html>
```

■ application 기본 객체: 자원 구하기

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import = "java.io.*" %>
<%@ page import = "java.net.URL" %>
<html>
<head><title>application 기본 객체 사용하여 자원 읽기2</title></head>
<body>

<%
        String resourcePath = "/message/notice/notice.txt";

        BufferedReader br = null;
        char[] buff = new char[512];
        int len = -1;

        try {
                URL url = application.getResource(resourcePath);

                br = new BufferedReader(
                        new InputStreamReader(
                                url.openStream() ));
```

■ application 기본 객체: 자원 구하기

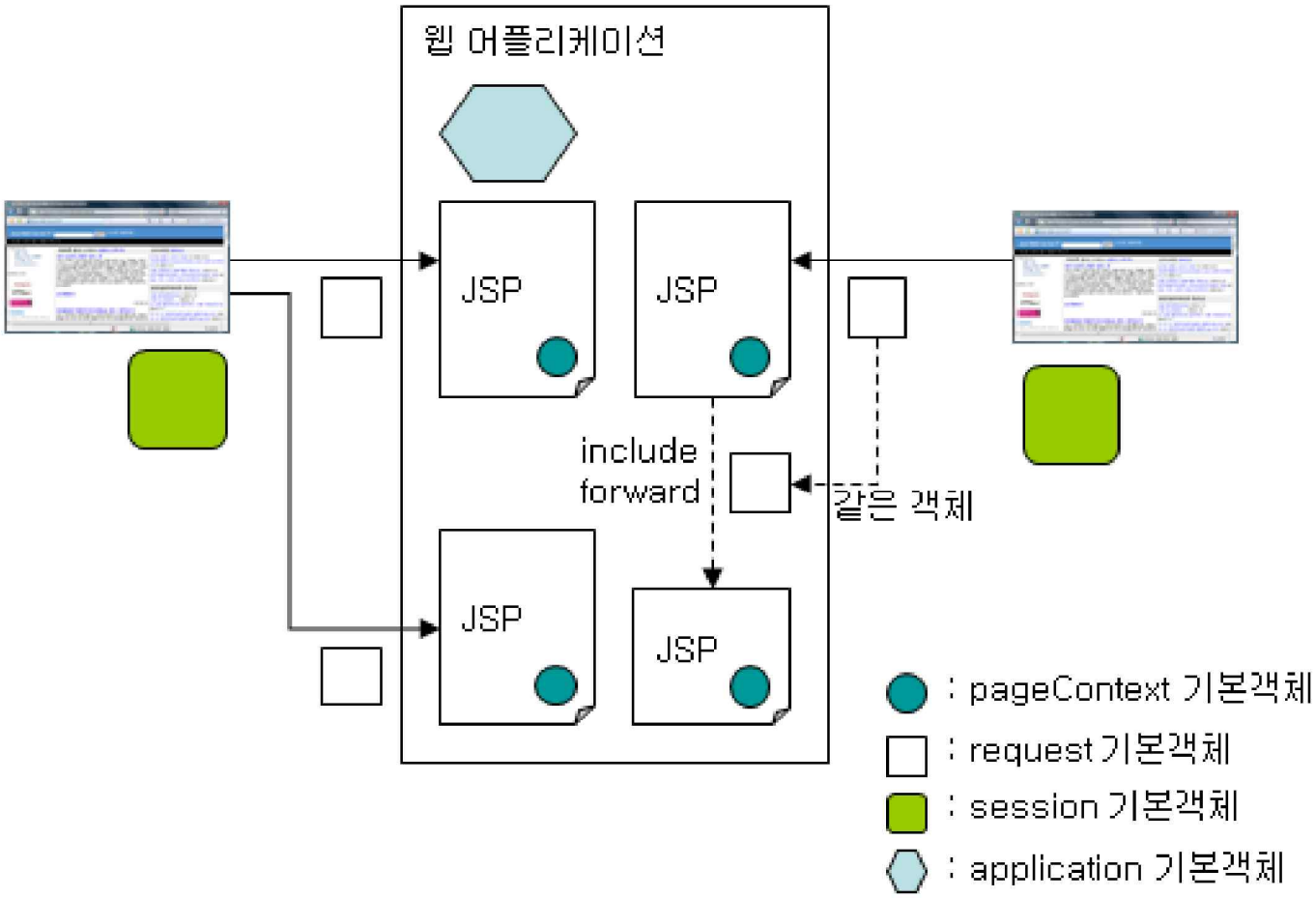
```
                while ( (len = br.read(buff)) != -1) {
                        out.print(new String(buff, 0, len));
                }
        } catch(IOException ex) {
                out.println("익셉션 발생: "+ex.getMessage());
        } finally {
                if (br != null) try { br.close(); } catch(IOException ex) {}
        }
%>

</body>
</html>
```

■ 기본 객체와 영역

- 네 영역
  - 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

■ 기본 객체와 영역


웹 어플리케이션
JSP
JSP
JSP
JSP
include
forward
같은 객체
: pageContext 기본객체
: request 기본객체
: session 기본객체
: application 기본객체

■ 속성(Attribute)

- **속성 기능 제공 기본 객체**
  - **pageContext, request, session, application**
- **속성 관련 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.)

■ 속성(Attribute)

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%      String name = request.getParameter("name");
        String value = request.getParameter("value");

        if (name != null && value != null) {
                application.setAttribute(name, value);
        }
%>
<html>
<head><title>application 속성 지정</title></head>
<body>
<%      if (name != null && value != null) {   %>
application 기본 객체의 속성 설정:
 <%= name %> = <%= value %>
<%      } else {    %>
application 기본 객체의 속성 설정 안 함
<%      }           %>
</body>
</html>
```

■ 속성(Attribute)

```
<%@ page contentType = "text/html; charset=euc-kr" %>
<%@ page import = "java.util.Enumeration" %>
<html>
<head><title>application 기본 객체 속성 보기</title></head>
<body>
<%
        Enumeration attrEnum = application.getAttributeNames();
        while(attrEnum.hasMoreElements() ) {
                String name = (String)attrEnum.nextElement();
                Object value = application.getAttribute(name);
%>
application 속성 : <b><%= name %></b> = <%= value %> <br>
<%
        }
%>
</body>
</html>
```

■ 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.