JSP 2-2

JSP 2-1

📚 Enumeration & 초기화 파라미터

🔢 Enumeration 이란?

> 넘어온 데이터의 이름 목록을 순차적으로 읽기 위한 타입 (`java.util.Enumeration`)

```java
@WebServlet("/student2")
public class Servlet05 extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
    {
        request.setCharacterEncoding("UTF-8");

        Enumeration<String> e = request.getParameterNames();
        //넘어온 데이터의 이름의 목록을
        //Enumeration 타입으로 반환

        while(e.hasMoreElements())//hasMoreElements: 읽어올 요소가 있는 지 확인후, true / false 반환
        {
            String name = e.nextElement();//다음 요소를 읽어온다

            String [] values = request.getParameterValues(name);
            for(String value : values)
            {
                System.out.println("name: " + name + ", value: " + value);
            }
        }
    }
}
```

```
name: studentID, value: 2020
name: name, value: 곱창김
name: age, value: 24
name: major, value: 컴퓨터공학
name: circle, value: 밴드
name: circle, value: 축구
```

사용 목적

- 이름을 모를 때 사용 (예: 요청 파라미터 목록 전체 순회)

주요 메소드

메소드	설명
`hasMoreElements()`	읽어올 요소가 있는지 여부 반환 (true/false)
`nextElement()`	다음 요소를 읽어온다

🛠️ 초기화 파라미터 (Initialization Parameter)

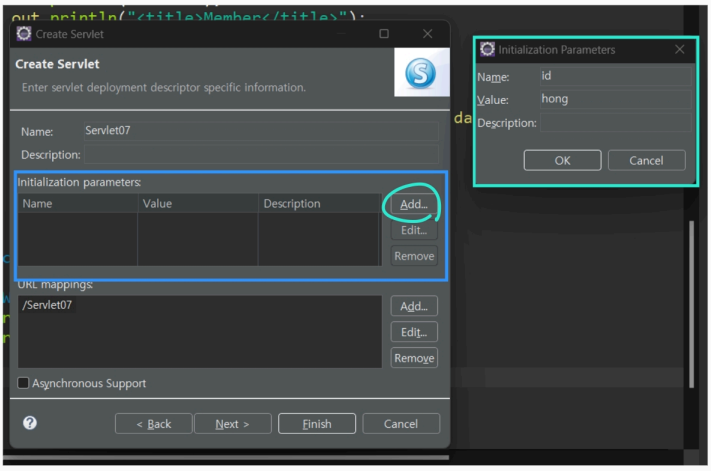
> Servlet이 생성될 때 필요한 설정 값을 미리 정의한 파라미터

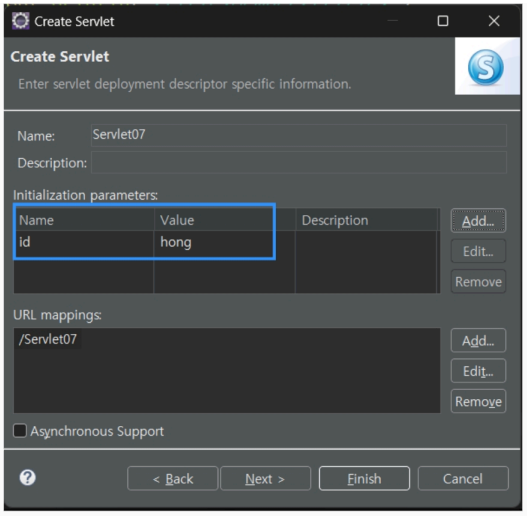
- 예: 관리자 ID, 서버 설정 값 등
- `web.xml` 에 정의하고 **ServletConfig** 객체를 통해 접근
- 또는 어노테이션으로 직접 정의할 수 있음

💡 ServletConfig 사용 구조

```
<!-- web.xml 예시 -->
<servlet>
```

```
    <servlet-name>SampleServlet</servlet-name>
    <servlet-class>com.example.SampleServlet</servlet-class>
    <init-param>
        <param-name>adminId</param-name>
        <param-value>admin123</param-value>
    </init-param>
</servlet>
```


Create Servlet
Create Servlet
Enter servlet deployment descriptor specific information.
Name: Servlet07
Description:
Initialization parameters:
Name
Value
Description
Add...
Edit...
Remove
URL mappings:
/Servlet07
Add...
Edit...
Remove
Asynchronous Support
< Back
Next >
Finish
Cancel
Initialization Parameters
Name: id
Value: hong
Description:
OK
Cancel


Create Servlet
Create Servlet
Enter servlet deployment descriptor specific information.
Name: Servlet07
Description:
Initialization parameters:
Name
Value
Description
id
hong
Add...
Edit...
Remove
URL mappings:
/Servlet07
Add...
Edit...
Remove
Asynchronous Support
< Back
Next >
Finish
Cancel

```java
@WebServlet(
        urlPatterns = { "/Servlet07" },
        initParams = {
                @WebInitParam(name = "id", value = "hong"),
                @WebInitParam(name = "pw", value = "1234")
        })
public class Servlet07 extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

}
```

···· 저장

```java
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
    ServletConfig config = super.getServletConfig();
    //ServletConfig 객체는 서블릿이 초기화 되는동안
    //참조해야할 정보를 가지고 있다가 전달해주는 역할을 한다
}
```

GenericServlet 추상 클래스 안에 있음

```java
String id = config.getInitParameter("id");
String pw = config.getInitParameter("pw");
System.out.println("id: " + id + ", pw: " + pw);
```

값 넣을 때

```xml
<servlet>
  <servlet-name>myServlet2</servlet-name>
  <servlet-class>com.servlet.Servlet07</servlet-class>
  <!-- 초기화 파라미터는 servlet태그안에 위치해야한다 -->
  <init-param>
      <param-name>id</param-name>
      <param-value>sung</param-value>
  </init-param>

      <init-param>
      <param-name>pw</param-name>
      <param-value>7777</param-value>
  </init-param>
</servlet>

<servlet-mapping>
  <servlet-name>myServlet2</servlet-name>
  <url-pattern>/initParam</url-pattern>
</servlet-mapping>
```

← web.xml에 초기 파라미터 넣는 방법

web.xml로하면 java 파일 건들지 않고도 수정이 가능하다

```
Problems  Terminal  Data Source Explorer  Properties  Console
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jdk-11\bin\javaw.exe (2024. 10. 29. 오후 2:27:00) [pid: 21776]
INFO: 웹 애플리케이션 디렉토리 [C:\JSP_class\JSP\apache-tomcat-9.0.95\webapps\manager]에 대한 배치가 [51] 밀리초에 완료되었습니다.
10월 29, 2024 2:27:03 오후 org.apache.catalina.startup.HostConfig deployDirectory
INFO: 웹 애플리케이션 디렉토리 [C:\JSP_class\JSP\apache-tomcat-9.0.95\webapps\ROOT]을(를) 배치합니다.
10월 29, 2024 2:27:03 오후 org.apache.catalina.startup.HostConfig deployDirectory
INFO: 웹 애플리케이션 디렉토리 [C:\JSP_class\JSP\apache-tomcat-9.0.95\webapps\ROOT]에 대한 배치가 [26] 밀리초에 완료되었습니다.
10월 29, 2024 2:27:03 오후 org.apache.coyote.AbstractProtocol start
INFO: 프로토콜 핸들러 ["http-nio-8090"]을(를) 시작합니다.
10월 29, 2024 2:27:03 오후 org.apache.catalina.startup.Catalina start
INFO: 서버가 [2205] 밀리초 내에 시작되었습니다.
id: sung, pw: 7777
id: sung, pw: 7777
id: sung, pw: 7777
id: sung, pw: 7777
id: sung, pw: 7777
```

🔗 데이터 공유 (ServletContext)

> 특정 데이터를 여러 서블릿에서 공유해야 할 때 사용

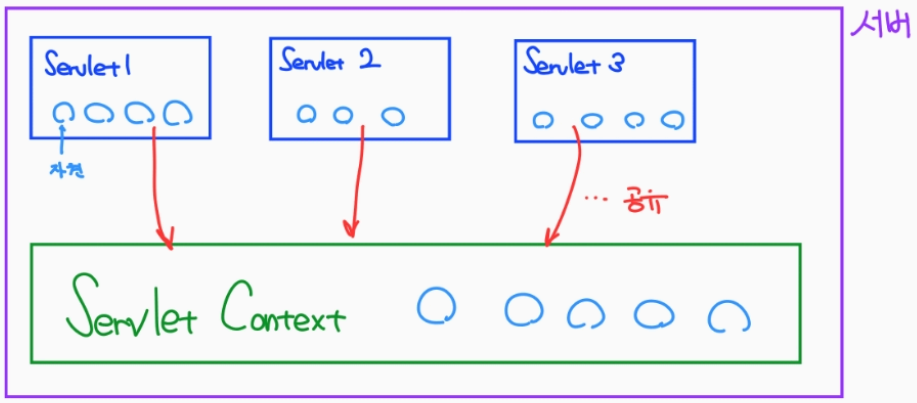
- `web.xml`에 파라미터 기술
- Servlet에서는 `ServletContext` 객체로 접근

✅ 구조 요약

```
<context-param>
    <param-name>sharedData</param-name>
    <param-value>value123</param-value>
</context-param>
```

> 📦 ServletContext를 통해 데이터 불러오기

```
ServletContext context = getServletContext();
String value = context.getInitParameter("sharedData");
```




```xml
<!-- 여러 서블릿에서 공유해서 쓰는 context-param 데이터 -->

<context-param>
  <param-name>name</param-name>
  <param-value>홍길동</param-value>
</context-param>

 <context-param>
  <param-name>age</param-name>
  <param-value>20</param-value>
</context-param>
```

: web.xml

… 저장

```java
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
    //웹 어플리케이션마다 하나의 ServletContext 객체가 생성이된다
    //서블릿끼리 자원(데이터)을 공유하는데 사용된다
    ServletContext application = super.getServletContext();

}
```

여러 서블릿들과 공유해서 사용가능

```java
protected void doGet(HttpServletRequest request, HttpServletResponse respon
{
    //웹 어플리케이션마다 하나의 ServletContext 객체가 생성이된다
    //서블릿끼리 자원(데이터)을 공유하는데 사용된다
    ServletContext application = super.getServletContext();

    String name = application.getInitParameter("name");
    String age = application.getInitParameter("age");

    System.out.println("name: " + name + ", age: " + age);

}
```