



2 서블릿의 기초

뇌를 자극하는 JSP & Servlet

❖ 학습목표

- 서블릿 클래스는 자바 클래스 형태로 구현되는 웹 애플리케이션 프로그램이며, 일반적인 자바 클래스를 작성 할 때보다 지켜야 할 규칙이 많다. 이 장에서는 그 규칙들을 배워보자.

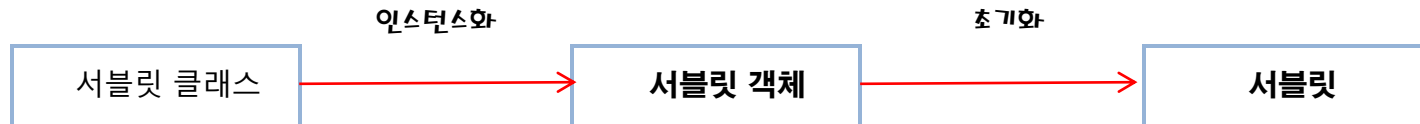
❖ 내용

- 서블릿이란?
- 서블릿 클래스의 작성, 컴파일, 설치, 등록
- 톰캣 관리자 프로그램 사용하기
- 웹 브라우저로부터 데이터 입력받기



1. 서블릿이란?

- 서블릿이란 서블릿 클래스로부터 만들어진 객체이다.
- 웹 컨테이너는 서블릿 클래스를 가지고 서블릿 객체를 만든 다음 그 객체를 초기화해서 웹 서비스를 할 수 있는 상태로 만드는데, 이 작업을 거친 서블릿 객체만 서블릿이라고 할 수 있다.



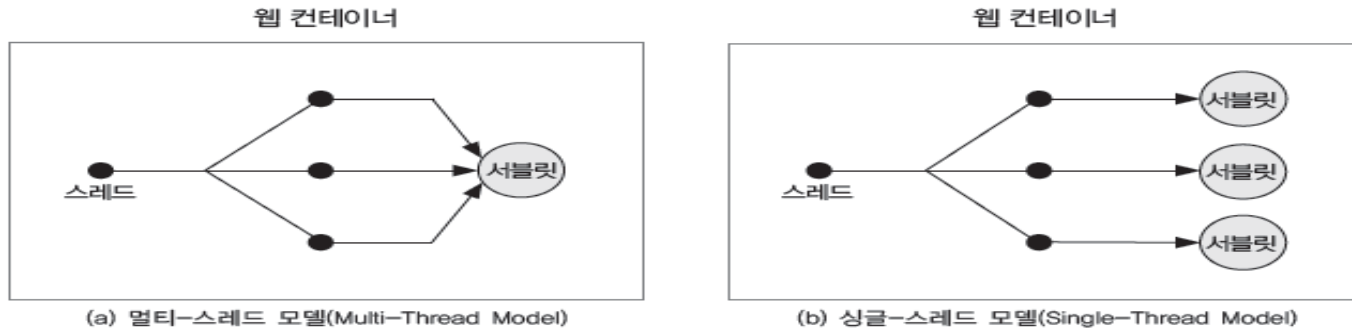
[그림 2-1] 서블릿 클래스, 서블릿 객체, 서블릿

- 인스턴스화(instantiation)란 클래스를 가지고 객체를 만드는 행위를 말한다.
- 멀티스레드(multithread)란 프로그램의 실행 흐름이 여러 갈래(thread)로 나뉘어져서 동시에 실행되는 것을 말한다.



1. 서블릿이란?

- 멀티스레드 모델의 장점: 필요한 서블릿의 수가 적기 때문에 서블릿을 만들기 위해 필요한 시스템 자원과 서블릿이 차지하는 메모리를 절약할 수 있다. 단점: 여러 스레드가 동시에 한 서블릿을 사용하기 때문에 데이터 공유 문제에 신경을 써야 한다.



[그림 2-2] 멀티-스레드 모델과 싱글-스레드 모델

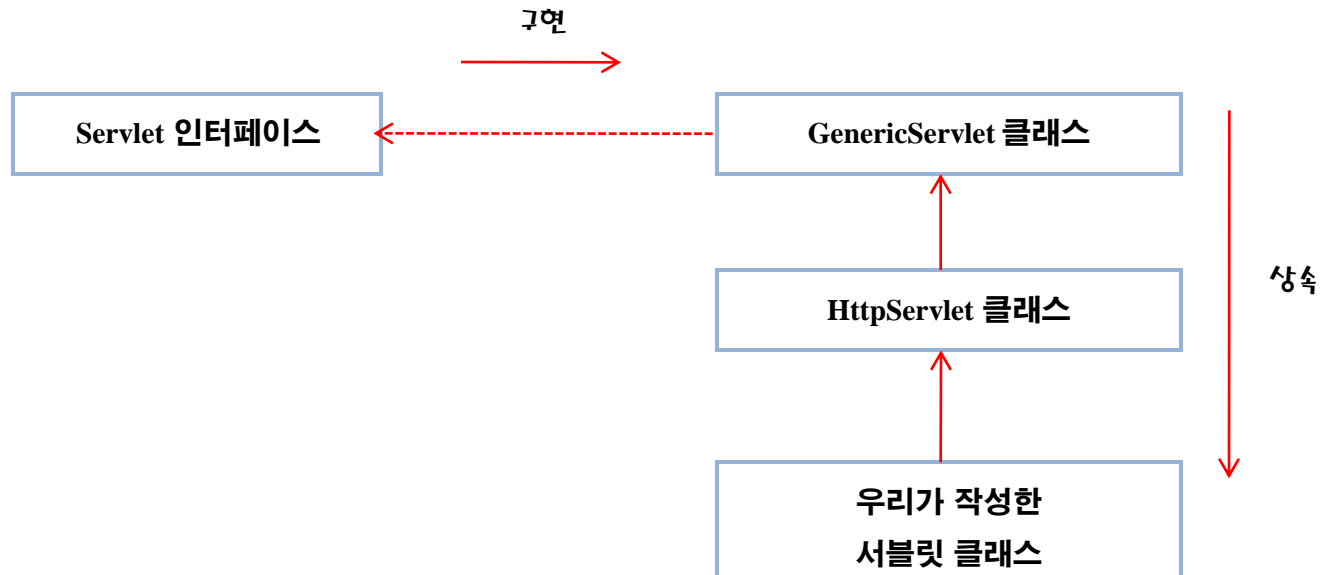
- 싱글-스레드 모델에서는 데이터 공유 문제를 걱정할 필요가 없지만 시스템 자원과 메모리가 더 많이 소모된다.



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스의 작성을 위한 준비

- 서블릿 클래스를 작성할 때 지켜야 할 규칙 세 가지
 - 서블릿 클래스는 javax.servlet.http.HttpServlet 클래스를 상속하도록 만들어야 한다
 - doGet 또는 doPost 메서드 안에 웹 브라우저로부터 요청이 왔을 때 해야 할 일을 기술해야 한다
 - HTML 문서는 doGet, doPost 메서드의 두 번째 파라미터를 이용해서 출력해야 한다



[그림 2-3] 서블릿 클래스의 상속/구현 관계



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스의 작성을 위한 준비

1. 자바 사이트의 URL (<http://java.sun.com>)을 입력합니다.

2. API 메뉴를 선택합니다.

3. Java EE 5를 선택합니다.

4. 알파벳 순서로 나열되어 있는 인터페이스/클래스 이름 중에서 Servlet, GenericServlet, HttpServlet을 선택합니다.

The screenshots show the following steps:

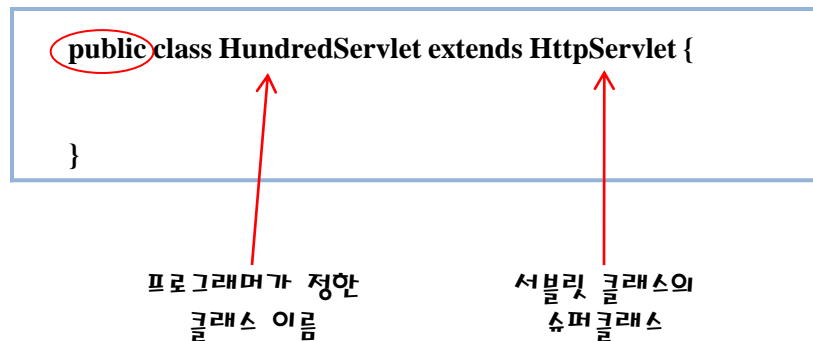
- Step 1: The Java website (<http://java.sun.com>) is displayed. The 'API Specifications' link is highlighted.
- Step 2: The 'API Specifications' page is displayed. The 'Servlet' link is highlighted.
- Step 3: The 'Servlet' page is displayed. The 'GenericServlet' link is highlighted.
- Step 4: The 'GenericServlet' page is displayed. The 'HttpServlet' link is highlighted.
- Step 5: The 'HttpServlet' page is displayed. The 'HttpServlet' link is highlighted.

[그림 2-4] Servlet 인터페이스와 GenericServlet, HttpServlet 클래스의 API 규격서

2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- 서블릿 클래스를 작성할 때 지켜야 할 첫 번째 규칙: `javax.servlet.http.HttpServlet` 클래스를 상속받도록 만들어야 한다. 그리고 `public` 클래스로 만들어야 한다.



- 서블릿 클래스 안에 `doGet` 또는 `doPost` 메서드를 선언해야 하며, 이 두 메서드는 `javax.servlet.http.HttpServletRequest`와 `javax.servlet.http.HttpServletResponse` 타입의 파라미터를 받고, 메서드 밖으로 `javax.servlet.ServletException`과 `java.io.IOException`을 던질 수 있도록 선언해야 한다.



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

HttpServlet 클래스의 API 규격서

1 HttpServlet 클래스의 API 규격서에서 트크를 바를 내리세요.

2 doGet 메서드의 이름을 클릭하면 메서드의 상세 정보가 나옵니다.

3 doGet 메서드의 리턴 타입, 파라미터 변수, 익셉션 타입을 확인하세요.

```
void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException
```

HttpServlet 클래스의 API 규격서

1 HttpServlet 클래스의 API 규격서에서 트크를 바를 내리세요.

2 doGet 메서드의 이름을 클릭하면 메서드의 상세 정보가 나옵니다.

3 doGet 메서드의 리턴 타입, 파라미터 변수, 익셉션 타입을 확인하세요.

```
void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException
```

[그림 2-5] HttpServlet 클래스의 doGet 메서드

2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- doGet 메서드를 작성할 때는 다음과 같은 골격을 만드는 것으로 시작한다.

```
public class HundredServlet extends HttpServlet {  
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
    }  
}
```

↑
HttpServlet 클래스의 doGet 메서드와
리턴 타입, 파라미터 변수, 익셉션 타입이 동일해야 한다.

- doGet 메서드를 public으로 선언해야 하는 이유는 웹 컨테이너가 웹 브라우저로부터 요청을 받아서 메서드를 호출할 때 필요하기 때문이다.
- doGet 메서드의 throws 절에 있는 ServletException과 IOException이 필요치 않으면 생략할 수도 있다. 하지만 다른 익셉션을 추가할 수는 없다.



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- doGet 메서드의 골격을 만든 다음에는 안에 내용을 채워 넣는다.

```
public class HundredServlet extends HttpServlet {  
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
        int total = 0;  
        for (int cnt = 1; cnt < 101; cnt++)  
            total += cnt;  
    }  
}
```

↑
1부터 100까지의 합을 구하는 명령문

- 실행 결과를 출력하는 코드는 doGet 메서드의 두 번째 파라미터를 이용해서 작성한다.
- 두 번째 파라미터는 javax.servlet.http.HttpServletResponse 인터페이스 타입이며, 여기에 getWriter라는 메서드를 호출해서 PrintWriter 객체를 구한다.

```
PrintWriter writer = response.getWriter();
```

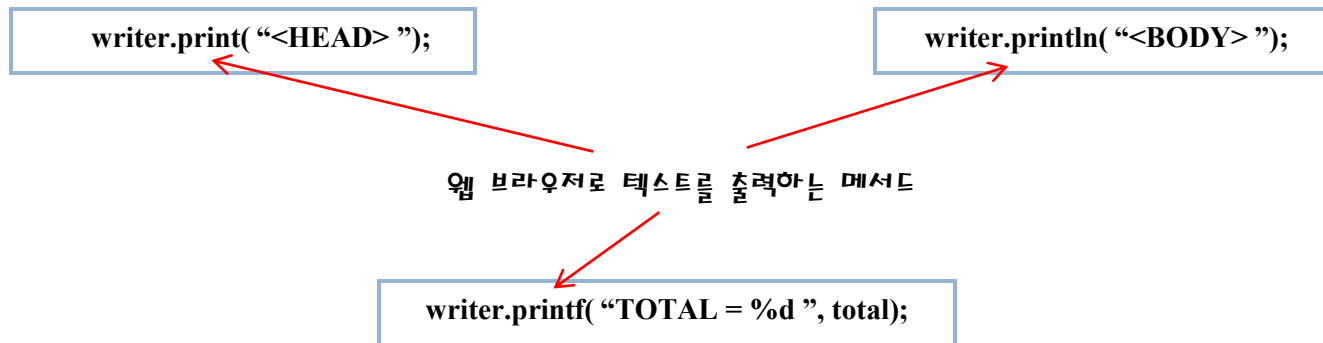
↑
PrintWriter 객체를 리턴하는 메서드



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- `PrintWriter`는 본래 자바 프로그램에서 파일로 텍스트를 출력할 때 사용하는 `java.io` 패키지의 `PrintWriter` 클래스이다.
- `Response.getWriter`메서드가 리턴하는 `PrintWriter` 객체는 파일이 아니라 웹 브라우저로 데이터를 출력한다.



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- 계산 결과를 웹 브라우저로 출력하는 코드

```
public class HundredServlet extends HttpServlet {  
    public void doGet(HttpServletRequest request, HttpServletResponse response)  
        throws ServletException, IOException {  
        int total = 0;  
        for (int cnt = 1; cnt < 101; cnt++)  
            total += cnt;  
        PrintWriter out = response.getWriter();  
        out.println( "<HTML> ");  
        out.println( "<HEAD><TITLE>Hundred Servlet</TITLE></HEAD> ");  
        out.println( "<BODY> ");  
        out.printf( "1 + 2 + 3 + ... + 100 = %d ", total);  
        out.println( "</BODY> ");  
        out.println( "</HTML> ");  
    }  
}
```

계산 결과를 HTML로 만들어서
웹 브라우저로 출력하는 명령문



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 작성하기

- 서블릿 클래스가 완성 되었으면, 코드에서 사용한 여러 가지 클래스와 인터페이스를 가져오는 import 문을 추가한다.

[예제2-1] 1부터 100까지 더하는 서블릿 클래스

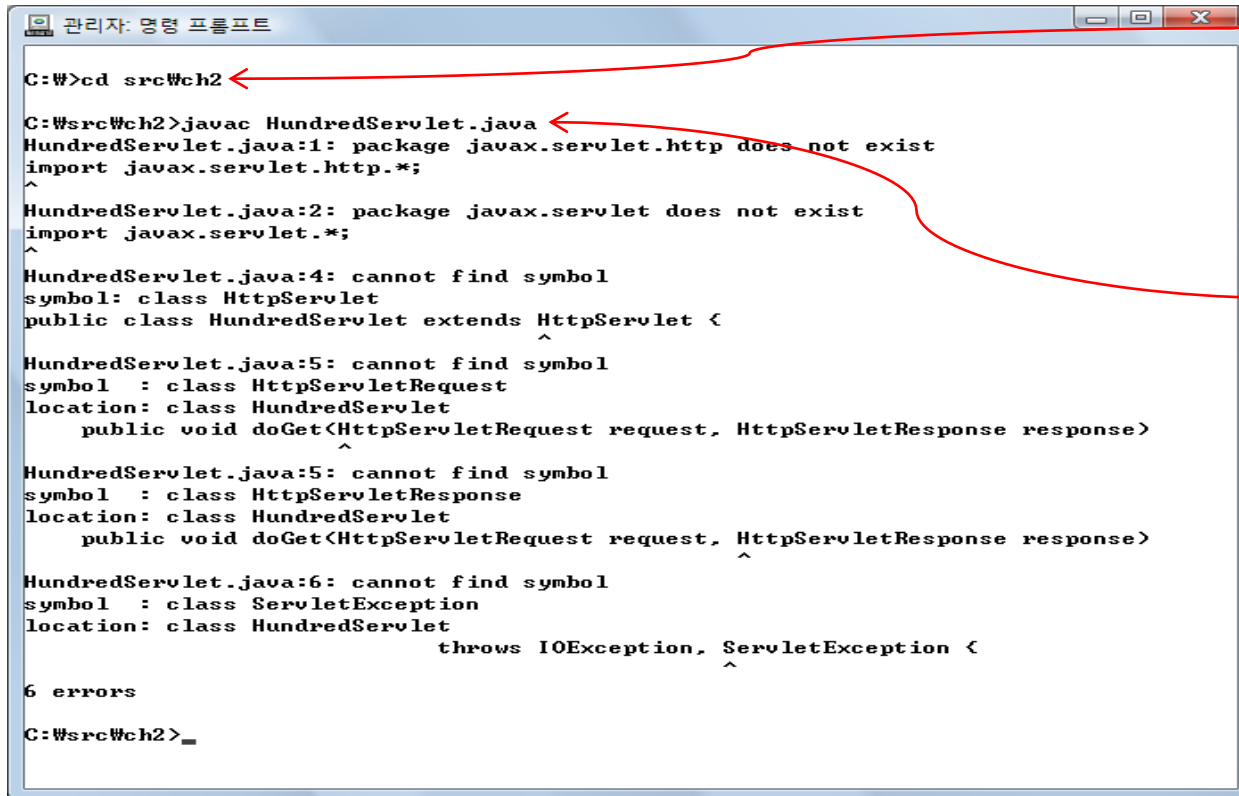
```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class HundredServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        int total = 0;
        for (int cnt = 1; cnt < 101; cnt++)
            total += cnt;
        PrintWriter out = response.getWriter();
        out.println( "<HTML> ");
        out.println( "<HEAD><TITLE>Hundred Servlet</TITLE></HEAD> ");
        out.println( "<BODY> ");
        out.printf( "1 + 2 + 3 + ... + 100 = %d ", total);
        out.println( "</BODY> ");
        out.println( "</HTML> ");
    }
}
```



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 컴파일하기

- 서블릿 클래스도 자바 클래스와 마찬가지로 `javac` 명령을 이용해서 컴파일할 수 있다.
- [예제 2-1]의 소스 코드를 저장해 놓는 디렉터리로 가서 `javac` 명령으로 컴파일하면 처음에는 다음과 같은 에러 메시지가 나온다.



```
C:\>cd srcWch2

C:\srcWch2>javac HundredServlet.java
HundredServlet.java:1: package javax.servlet.http does not exist
import javax.servlet.http.*;
^
HundredServlet.java:2: package javax.servlet does not exist
import javax.servlet.*;
^
HundredServlet.java:4: cannot find symbol
symbol: class HttpServlet
public class HundredServlet extends HttpServlet {
^
HundredServlet.java:5: cannot find symbol
symbol : class HttpServletRequest
location: class HundredServlet
    public void doGet(HttpServletRequest request, HttpServletResponse response)
^
HundredServlet.java:5: cannot find symbol
symbol : class HttpServletResponse
location: class HundredServlet
    public void doGet(HttpServletRequest request, HttpServletResponse response)
^
HundredServlet.java:6: cannot find symbol
symbol : class ServletException
location: class HundredServlet
    throws IOException, ServletException {
^

6 errors

C:\srcWch2>
```

소스 코드가 있는 디렉터리로 이동하는 명령

서블릿 클래스를 컴파일하는 명령

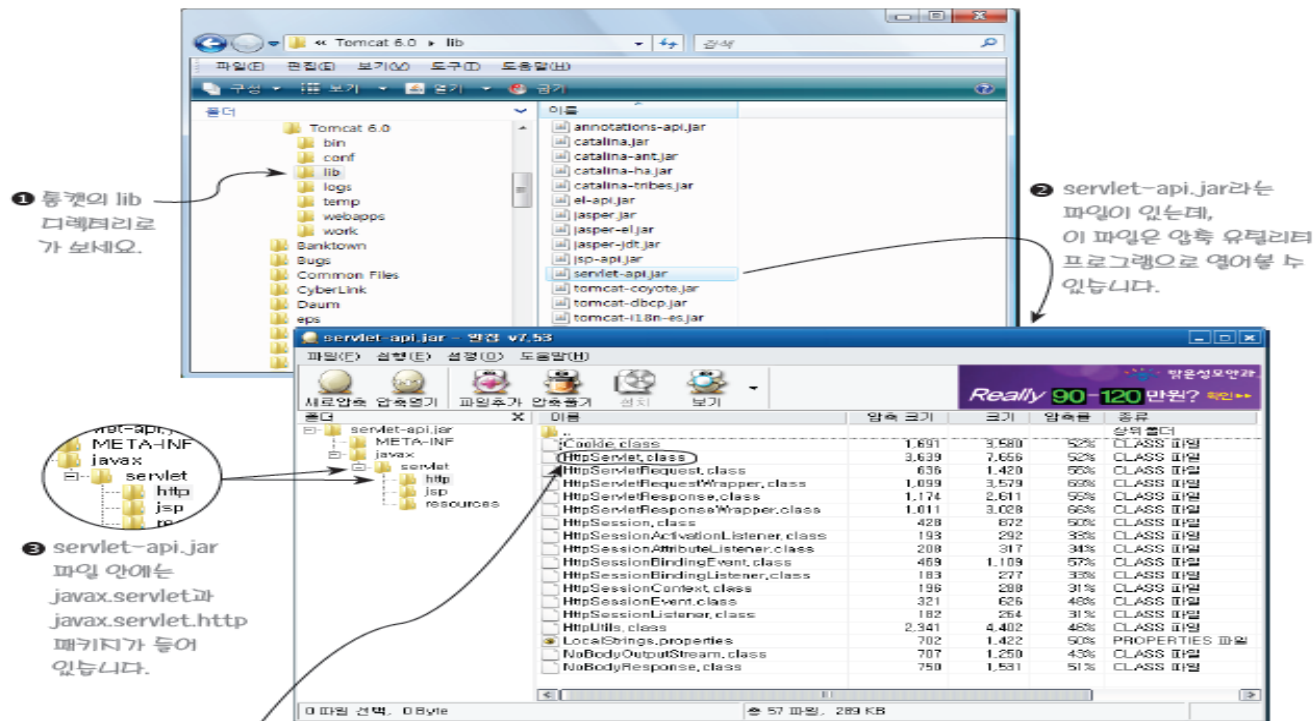
[그림 2-6] 아무 옵션도 사용하지 않고 서블릿 클래스를 컴파일했을 때 나오는 에러 메시지



2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 컴파일하기

- 에러 메시지가 나오는 이유는 import한 javax.servlet과 javax.servlet.http 패키지가 JDK의 표준 라이브러리 안에 없기 때문이다.
- 서블릿 클래스를 컴파일할 때는 -cp 옵션을 이용해서 이 두 패키지가 속하는 라이브러리의 경로명을 명시해 주어야 한다.




javax.servlet.http 패키지에 속하는
HttpServlet 클래스의 파일입니다

[그림 2-7] javax.servlet과 javax.servlet.http 패키지가
들어 있는 servlet-api.jar 파일

2. 서블릿 클래스의 작성, 컴파일, 설치, 등록

❖ 서블릿 클래스 컴파일하기

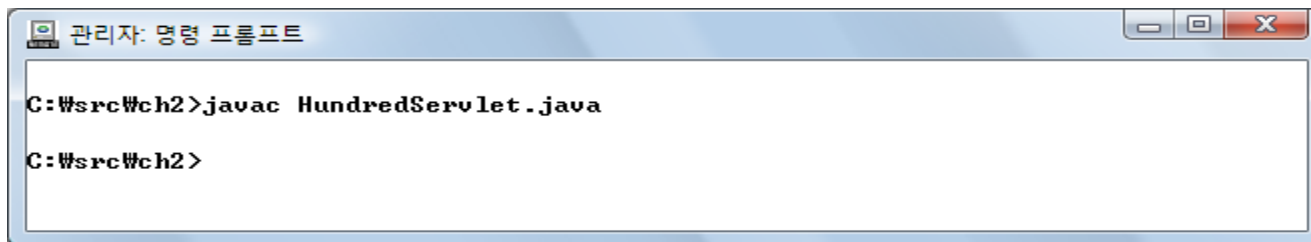
- 서블릿 클래스를 컴파일할 때 `-cp` 옵션으로 `javax.servlet-api.jar` 파일 경로명을 지정하면 컴파일 에러가 발생하지 않는다.



```
관리자: 명령 프롬프트
C:\wsrc\ch2>javac -cp "C:\Program Files\Apache Software Foundation\Tomcat 6.0\lib\servlet-api.jar" HundredServlet.java
C:\wsrc\ch2>
```

[그림 2-8] 서블릿을 컴파일하는 방법(1)

- 컴파일에 실패한다면 경로명을 입력 과정에서 실수일 수 있다.
- 톰캣의 lib 서브디렉터리에 있는 `javax.servlet-api.jar` 파일을 JDK 설치 디렉터리 아래의 `jre\lib\ext` 서브디렉터리로 복사한다.



```
관리자: 명령 프롬프트
C:\wsrc\ch2>javac HundredServlet.java
C:\wsrc\ch2>
```

[그림 2-10] 서블릿을 컴파일하는 방법(2)

