서블릿

|  |  |
| --- | --- |
| 1 | 다음 설명 중 맞는 것을 모두 고르면?  (1) 웹서버에서 실행되는 javaEE 기반의 클래스를 서블릿이라 한다.  (2) ~~서블릿의 인스턴스는 프로그래머가 new 연산자를 통하여 메모리에 올려야 한다.~~  (3) ~~서블릿을 실행시키려면 웹브라우저의 URL에 클래스명을 기입하여 접속을 시도하면 된다~~  (4) ~~서블릿 코드 내에 HTML 코드는 포함할 수 없다~~ |
| 2 | 다음 설명 중 틀린 것을 고르면?  (1) ~~JSP와 서블릿 중 HTML 코드를 보다 쉽게 작성할 수 있는 기술은 서블릿이다.~~  (2) ~~JSP와 서블릿은 서로 관련성이 전혀 없는 기술이며, 각각 독자적인 방법으로 실행되어 진다~~  (3) JSP와 서블릿은 별도의 존재가 아니며 JSP로 작성된 코드는 실행 시 자동으로 서블릿 코드로 변경되어진다.  (4) 서블릿 클래스내에 HTML 코드를 포함하여 디자인을 표현하는 방식은 효율성이 너무 떨어지므로 ,  즉 결과 페이지를 보다 쉽고 빠르게 작성하기 위해서 고안된 기술이 바로 JSP 인 것이다..  (5) ~~JSP를 서블릿 클래스로 변환시키는 주체는 개발자이다.~~ |
| 3 | 서블릿에 대한 설명 중 틀린 것은?  (1) ~~서블릿은 컨테이너에 의해서만 자동으로 생성되어지며 프로그래머가 작성할 수는 없다.~~  (2) 서블릿은 JSP와 달리 웹브라우저 URL에 클래스명을 직접 입력하여 실행시킬 수는 없다  (3) ~~개발자가 서블릿의 매핑을 작성하려면 server.xml 에서 작성한다~~  (4) JSP의 <%! 선언부 %> 영역에 코드를 작성하는 것은 서블릿의 멤버 영역에 코딩한 것과 같다.  (5) JSP의 <% 서비스 %> 영역에 코드를 작성하는 것은 서블릿의 service() 메서드에 코딩한 것과 같다. |
| 4 | 다음 설명 중 틀린 것을 고르면?  (1) ~~서블릿의 인스턴스는 클라이언트의 요청이 있을 때마다 메모리에 생성되어진다.~~  (2) 클라이언트의 요청이 있을때마다 서버 측에서는 요청을 처리하기 위한 쓰레드가 생성되며, 이 쓰레드  의해 서블릿의 service() 메서드가 실행되어 진다.  (3) 위(2)번에 의하면, 결국 요청을 처리하는 메서드는 service() 메서드 이므로 service() 메서드의 매개변  수로 request 객체와 response 객체가 전달되어야 한다.  (4) service() 메서드는 클라이언트의 요청 방법이 GET이면 doGet() 메서드를 , POST 이면 doPost() 메서드  를 자동 호출하여 준다.  (5) 결국 개발자가 서블릿을 재 정의할때, 요청 처리 로직을 작성해야 할 메서드는 service()가 아니라,  doXXX() 형 메서드이다.  (6) JSP는 클라이언트의 최초 요청이 있을때 서블릿으로 변경되어 인스턴스가 생성되며, 이 인스턴스 생성  이후 부터는 요청이 있을 때마다 쓰레드가 생성되어 요청을 처리하게 된다. |
| 5 | javaEE ApI에 대한 설명 중 틀린 것은?  (1) 서블릿 클래스의 자료형은 HttpServlet 이며 바로 직계 부모는 GenericServlet 이고, 최상위 객체는  Servlet 인터페이스이다.  (2) Servlet 인터페이스는 서블릿의 최상위 객체이므로 서블릿으로써 갖추어쟈 할 가장 기본이 되는 메서드  가 정의되어 있다  (3) ~~JavaEE 기반의 서버 제품마다 각각 기술력이 틀리므로 Tomcat에서는 웹요청을 Servlet 이 처리하지만,~~  ~~JBoss에서는 전혀 다른 명칭의 객체와 메서드가 웹요청을 처리한다~~  (4) ~~개발자가 서블릿 클래스를 정의한 후에 new 연산자로 인스턴스를 미리 생성해 놓아야 클라이언트의~~  ~~요청을 받을 수 있다.~~  (5) ~~HttpServlet, GenericServlet, Servlet 인터페이스 등을 포함한 JavaEE 관련 API 객체들은 jar파일 형태로~~  ~~존재하는데 이 jar 파일은 오라클사에서 다운로드 받아야 한다.~~  (6) Tomcat, JBoss, WebLogic, JEUS 등 WAS 제품은 다양하지만 각 제품의 종류와 상관없이 개발자가 항상  동일한 API를 사용할 수 있는 이유는 JavaEE 기반의 WAS들은 자바의 기준 스팩을 준수하여 개발되었  기 때문이다. |
| 6 | HttpServlet 클래스의 생명주기에 대한 설명 중 맞는 것은?  (1) 개발자에 의해 작성된 서블릿을 서버에 배포해 놓으면 최초의 요청을 시도하는 클라이언트에 의해  인스턴스가 생성되며, 이때 서블릿은 초기화가 진행되는데~~, 이때 초기화를 담당하는 생명주기 메서드가~~  ~~service() 이다~~  (2)일단 서블릿의 인스턴스가 생성된 후에는 클라이언트의 요청이 추가로 발생하더라도 인스턴스는 중복  생성되지 않으며, 대신 쓰레드가 생성되어 요청을 처리하게 된다, ~~이때 이 요청을 처리하기 위한 생명주~~  ~~기 메서드가 바로 init() 메서드 이다~~  (3) service() 메서드가 요청을 처리하기 위해서는 클라이언트의 요청 정보도 알아야 하고, 응답 내용을  담아 놓을 수 있는 객체도 필요하므로 service() 메서드의 API를 확인해보면 Request 객체와 Response  객체를 매개변수로 받고 있는 것을 확인할 수 있다.  (4) 서블릿의 service() 메서드의 수행이 완료되면 WAS인 Tomcat은 Response 정보를 확인하여 최종적으  응답 정보를 HTML 형식으로 구성하여 클라이언트에게 응답한다. |
| 7 | 다음 설명 중 맞는 것은?  (1) 서블릿의 생명주기 메서드 중 가장 먼저 호출되어 지는 메서드는 init() 메서드이며~~, 클라이언트의~~  ~~요청이 있을 때마다 호출되어 진다.~~  (2) init() 메서드는 서블릿의 초기화를 담당하므로 ~~서블릿의 생성자로 볼 수 있다.~~  (3) ~~서블릿 추상클래스이며 컨테이너가 생성하므로 생성자가 존재할 수 없다.~~  (4) 서블릿의 생명주기와 관련된 메서드에는 init(), service(), destroy() 가 있다. |
| 8 | 다음 설명 중 틀린 것은?  (1) **서블릿 객체가 생성되어 질 때 자신이 속한 웹어플리케이션의 정보를 컨테이너로부터 얻게 되는데**  **이 정보를 가진 객체가 바로 ServletConfig 인터페이스이다.**  (2) ~~서블릿 객체 생성시 생성자 보다 앞서 호출되는 메서드가 바로 init() 메서드이다.~~  (3) 웹어플리케이션의 서블릿 환경 정보를 담는 ServletConfig 객체는 init() 메서드의 인수로 전달되어  진다.  (4) **위(3)번의 초기화 정보에 개발자가 원하는 데이터를 추가하려면 web.xml 파일을 이용할 수 있다.** |
| 9 | 서블릿에 대한 설명 중 틀린 것은?  (1) HttpServletRequest는 사실 JSP에서 사용하였던 request 내장 객체의 데이터 타입이었다.  (2) HttpServletResponse는 사실 JSP에서 사용하였던 response 내장 객체의 데이터 타입이었다.  (3) HttpServletRequest, HttpServletResonse 객체는 인터페이스이므로 프로그래머가 new 할 수 없으며  요청이 있을 때마다 컨테이너에 의해 메모리에 올라간다.  (4) 하나의 요청이 끝나는 시점은 결국 doXXX형 메서드의 닫는 괄호를 만나는 시점이다  (5) 하나의 요청이 끝나는 시점은 doXXX 형 메서드에서 요청에 대한 응답 정보가 완성된 후, 이때 저장된  응답 정보를 통해 WAS가 클라이언트에게 응답해 주는 시점이다  (6) WAS가 클라이언트에게 응답을 한 후, 요청 처리에 관여했던 객체인 쓰레드, request, response 객체는  모두 소멸된다. |
| 10 | 다음 설명 중 틀린 것은?  (1) 서블릿의 API는 클라이언트의 “요청” 에 대한 개념을 HttpServletRequest 인터페이스로 정의해 놓았  으며 이 객체를 이용하여 클라이언트의 요청과 관련된 많은 업무를 처리할 수 있다.  (2) HttpServletRequest 인터페이스에 대한 API 문서는 Sun사에서 제공한다.  (3) ~~개발자는 doXXX()형 메서드보다는 주로 service() 메서드를 재정의함 으로서 웹서비스를 구현한다.~~  (4) ~~클라이언트와 서버의 개념은 HTTP 프로토콜에서만 사용되는 개념이다.~~ |