스프링 프레임웍 소개

|  |  |
| --- | --- |
| 1 | 다음 설명 중 틀린 것 은?  (1) 96년 발표된 자바 언어가 세상에 관심을 끌었던 이유는 웹브라우저에서 화려한 에니메이션을 구현할 수 있는 애플릿 때문이었다.  (2) Sun은 같은 해 말 자바 언어의 컴포넌트 기반 개발 방식의 장점을 내세우며 JavaBeans 컴포넌트 모델 에 대한 스팩을 발표하였다.  (3) 자바 빈즈 스팩이란 자바 객체를 활용하여 어플리케이션을 개발하는 방법을 명시한 코딩 정책을 말한다.  (4) 자바 빈즈 스팩이 발표된 이후, 자바는 실제적인 기업용 어플리케이션의 대세로 자리잡았다.  (5) Sun은 기업용 어플리케이션 시장 공략을 위해 기존 자바 빈즈에서는 지원하지 않았던 트랜잭션 , 보안, 분산 컴퓨팅 등의 서비스를 추가한 새로운 자바 빈즈 스팩을 발표하게 되었는데, 이것이 바로 기업용 자바 빈즈인 EJB(Enterprise JavaBeans) 스팩이었다. |
| 2 | 다음 설명 중 틀린 것은?  (1) EJB는 개발자가 직접 구현해야 할 업무를 다양한 서비스로 지원하였는데, 특히 트랜잭션, 보안 등을  프로그래밍적이 아닌 , XML과 같은 선언적 방법으로 지원하였다.  -> 선언적 방법 : xml설정에 의한 방법  (2) EJB의 배포디스크립터, 홈 인터페이스, 리모트 인터페이스, 로컬 인터페이스 등, 사용상의 복잡성은 개발자로 하여금 실제 핵심 비즈니스 로직에 집중할 수 없는 수준이었기 때문에 개발자들에게 점점 외면당하게 되었다.  (3) EJB는 이름에서 알 수 있 듯, 기존의 자바 빈즈를 엔터프라이즈급으로 확장한 기술이기 때문에 기존  자바 빈즈에 익숙한 개발자라면 누구나 EJB 사용에 큰 무리가 없었다.  (4) EJB는 이름으로 봐서는 자바 빈즈 스팩을 계승 및 유지한 것처럼 보이지만, 사실상 기존의 자바 빈즈  와의 유사성은 전혀 없는 기술로 봐야한다 |
| 3 | 다음 설명 중 틀린 것은?  (1) 관점 지향 프로그래밍(AOP)과 의존성 주입(DI)을 이용하면, 굳이 EJB를 이용하지 않더라도 선언적  트랜잭션 등 필요한 기술들을 구현할 수 있다는 사실을 알게됨으로서 EJB는 점점 외면받게 되었다.  (2) Road Johnson은 그의 저서인 Expert One-on-One: J2EE Design and Development 를 통해 스프링 프레임 워크를 소개하였는데, 주 내용은 기존의 평범한 자바 빈즈(POJO)로도 기업용 어플리케이션을 제작할 수 있다는 내용이었다.  (3) 복잡한 기술없이 평범한 자바빈즈(POJO)로도 선언적 트랜잭션 , 보안 등을 구현할 수 있다면 굳이 설정 이 복잡한 EJB를 개발자들이 사용할 이유가 없다.    (4) POJO 만로도 자바 개발자가 원하는 엔터프라즈 수준의 서비스를 제공할 수 있는 이유는 DI와 AOP가  적용되었기 때문이며, 스프링을 지배하는 핵심 개발 원리 이기도 하다  (5) DI는 객체 간 의존성을 약화시켜 결합도를 낮춰 주지만 여전히 의존성은 존재한다. 하지만 AOP를 이용 하면 객체 간 의존성을 완전히 탈피할 수 있다.  (6) 트랜잭션, 보안 등의 처리는 사용빈도 수가 높으며, 어플리케이션 전반적에 걸쳐 공통적으로 사용되는  로직이므로, DI로 처리하는 게 유리하다   * AOP가 적당함 |
| 4 | 다음 설명 중 틀린 것은? -> 틀린답 없음  (1) Java 언어는 사설 표준이므로, 업체마다 자신들의 자체 프레임워크를 구축하여 프로젝트에 적용하다 보니 개발의 수준, 방법 등의 차이로 인하여 여러 문제점들이 발생하고 있다.  (2) 국내에서는 공공부문 정보화사업에 표준화된 개발 프레임워크를 개발 및 도입하였는데, 이를 전자정부 프레임워크라 한다.  (3) 전자정부 프레임워크는 2008년 오픈 소스인 스프링 프레임워크를 기반으로 웹어플리케이션 개발에 필요한 개발 환경 및 여러 기능을 미리 구현해 놓았다.  (4) 전자정부 프레임워크를 도입하면 소프트웨어의 표준화, 품질 향상 및 컴포넌트의 중복 개발 등의 문제를 방지할 수 있으며 특정 업체의 종속성 심화 문제를 해결할 수 있다. |