2012년 10월

웹스피어 애플리케이션 서버 v8.5 - 리버티 프로파일과 Tomcat 7 비교

애플리케이션 서비스 환경을 선택할 때 정보를 기반으로 의사 결정

내용

개요

이 문서는 애플리케이션 서비스 환경으로 IBM® WebSphere® Application Server 버전 8.5 - 리버티 프로파일과 Apache Tomcat 7 중 하나를 선택해야 할 때 충분한 정보를 바탕으로 결정을 내릴 수 있도록 돕기 위해 작성되었습니다. 두 서버 모두 규격 사양의 서블릿 컨테이너를 제공하지만 이 문서에서 설명하듯이 커다란 차이점도 있습니다.

리버티는 무엇입니까?

리버티 프로파일은 웹스피어 애플리케이션 서버 버전 8.5용으로 개발된 최신 동적 프로파일입니다. 이 프로파일은 경량화되고 표준화된 런타임으로 웹 애플리케이션을 신속하게 개발하고 배포할 수 있습니다. 리버티 프로파일은 웹스피어 애플리케이션 서버의 풀 프로파일을 충실하게 지원하므로 런타임 차이 없이 웹스피어 애플리케이션 서버 v8.5 - 리버티 프로파일에 맞춰 애플리케이션을 개발하고 테스트함은 물론, 웹스피어 애플리케이션 서버의 풀 프로파일에 맞춰 애플리케이션을 배포할 수 있습니다. 또한 웹스피어 애플리케이션 서버 v8.5 - 리버티 프로파일 서버에 운영 환경용으로 애플리케이션을 배포할 수 있으며 이 경우 풀 프로파일을 사용하는 웹스피어 애플리케이션 서버와 비슷한 성능을 발휘합니다.

리버티 아키텍처는 원천적으로 컴포넌트 기반으로 조정이 가능합니다. 애플리케이션에 필요한 구성요소를 사용자가 세밀한 단위로 구성할 수 있으므로 서버 재시작 시간이 대단히 빠르고 설치 공간은 최소한으로

유지됩니다. 선택 구성이 가능한 기능으로는 서블릿, JSP(JavaServer Pages) 및 JSF(JavaServer Faces) 등 여러 가지 Java 엔터프라이즈 서비스가 있습니다. 이와 함께 보안 서비스 및 OSGi 애플리케이션 지원도 선택적으로 구성할 수 있습니다.

Tomcat은 무엇입니까?

Tomcat은 Apache Foundation의 오픈소스 웹 애플리케이션 서버입니다. Tomcat은 Java 서블릿 및 JSP 기술을 위한 공식적인 기준 서버로 출발했습니다. Tomcat은 서블릿 컨테이너(Catalina), JSP 엔진(Jasper) 및 HTTP 전송(Coyote)의 세 가지 오픈소스 구성요소로 구성되어 있습니다. 현재 Sun Microsystems(현 Oracle)에서 공식적인 Java EE 기준 구현(GlassFish)을 제시하고 있지만 Tomcat은 여전히 서블릿 3.0 사양 및 JSP 2.2 사양과 호환됩니다.

중요한 것은 이 서버의 범위에는 변화가 없다는 점입니다. 서블릿 및 JSP 이외의 애플리케이션 유형이나 다른 서비스는 지원하지 않습니다.[1] 이 범위를 넘어서는 기능, 즉 웹스피어 애플리케이션 서버 등의 Java EE 컨테이너 기능을 Tomcat에서 동등한 수준으로 사용하려면 타사 라이브러리가 필요합니다. 이러한 타사 라이브러리를 통합하는 것은 사용자의 몫이며 호환성 및 통합 작업도 개발자가 책임져야 합니다. Tomcat 자체에 대한 지원은 Tomcat 커뮤니티의 인터넷 포럼을 통해 받을 수 있습니다. 한편 VMware Spring TCserver, Mule TCat 및 기타 업체 등 타사 공급업체 여러 곳에서 Tomcat 및 다양한 변형 제품에 대한 지원을 유료로 받을 수 있습니다.

"도시 전설"

Tomcat을 Apache HTTP 웹 서버와 혼동하는 경우가 많습니다. Tomcat은 Java 애플리케이션을 실행할 수 있는 웹 서버 및 웹 컨테이너를 순수하게 Java로 구현한 것이고, Apache HTTP 웹 서버는 C 언어로 구현한 웹 서버입니다.

Tomcat이 웹스피어 애플리케이션 서버 안에 통합되어 있다고 생각하는 사람들도 있습니다. 웹스피어 애플리케이션 서버 커뮤니티 에디션은 Tomcat이 포함된 Apache Geronimo를 기반으로 한 제품이지만, 리버티 프로파일을 비롯한 웹스피어 애플리케이션 서버의 다른 에디션은 OSGi 애플리케이션을 지원하는 Java EE 규격의 자체 웹 컨테이너를 포함하고 있습니다.

웹스피어 애플리케이션 서버 도입을 고려할 때는 커뮤니티 에디션과 상업용 에디션의 차이점을 이해해야 합니다.

- 웹스피어 애플리케이션 서버 커뮤니티 에디션은 Apache Geronimo 기반의 오픈소스 애플리케이션 서버입니다. IBM에서 제공하는 커뮤니티 에디션은 개발용인 경우 무료로 사용할 수 있습니다. 이를 위해 IBM에서 유료 지원 서비스를 제공합니다.

(참조: http://www-01.ibm.com/software/webservers/appserv/community/#)

- 웹스피어 애플리케이션 서버의 상업용 에디션은 오픈소스가 아닙니다. 상업용 에디션 간에는 공통의 코드 기반을 공유하지만 웹스피어 애플리케이션 서버 커뮤니티 에디션과는 코드가 다릅니다. 상업적 웹스피어 애플리케이션 서버 상업용 에디션(Commercial Edition)에 속하는 개발자용 웹스피어 애플리케이션 서버는 개발 시스템에서 무료로 사용할 수 있으며 지원 옵션은 유료로 제공됩니다. 나머지 웹스피어 애플리케이션 서버 에디션은 PVU 또는 소켓당 가격으로 라이센스를 구입해야 합니다.

(참조: http://www-01.ibm.com/software/webservers/appserv/was/)

범위

이 문서는 개발자용으로 작성되었습니다. 여기서는 IBM 웹스피어 애플리케이션 서버 리버티 프로파일과 Tomcat의 기능을 비교해 보겠습니다.

빠른 비교

1. 리버티 및 Tomcat 설치

웹스피어 애플리케이션 서버 리버티 프로파일(리버티)과 Tomcat은 모두 구하기도 쉽고 설치도 간단합니다.

1.1 제품 구하기

리버티 및 Tomcat은 모두 웹 사이트에서 다운로드할 수 있습니다. 리버티는 46MB의 jar 설치 파일이고 Tomcat은 8MB입니다. 리버티는 대부분의 웹 애플리케이션 및 OSGi 애플리케이션에서 요구하는 기능을 모두 제공합니다.

Tomcat은 웹 서블릿과 JSP 컨테이너에 불과하며, 리버티 수준의 기능을 얻기 위해 타사 라이브러리를 추가할 경우 설치 공간이 늘어납니다[2](섹션 3. 기능 비교 참조). 리버티 프로파일을 사용하면 동일한 설치 디렉토리에 여러 개의 서버를 설치하고 리버티 런타임 바이너리를 공유하면서 각 서버를 실행할 수 있습니다. Tomcat은 서버를 여러 개 사용하려면 설치된 바이너리의 사본을 여러 개 만들거나 시작 및 종료를 위한 신규 스크립트를 직접 제작하고 $CATALINA_HOME을 변경하여 다른 server.xml 파일을 가리키도록 해야 합니다.[3]

1.2 아카이브 설치

1.2.1. 리버티 및 Tomcat

리버티 또는 Tomcat을 설치하려면 다운로드한 아카이브를 원하는 디렉토리에 풀기만 하면 됩니다. Extraction will take only a few seconds. Tomcat 설치에는 기본 서버 구성이 포함되어 있습니다.[4] 리버티는 bin 디렉토리에서 "server" 명령을 처음 사용할 때 기본 서버가 생성됩니다.

1.3 관리 설치

1.3.2 리버티

IBM 설치 관리자로 리버티를 설치하고 리버티 프로파일을 다운로드하며, 기능 픽스 및 업그레이드를 적용할 수 있습니다. 필요에 따라 변경 후 롤백이 가능하며 원한다면 리버티를 삭제할 수 있습니다. 웹스피어 애플리케이션 서버 Network Deployment v8.5의 작업 관리자를 사용하여 원격 시스템에 리버티를 설치 또는 삭제(시작 또는 중지)할 수도 있습니다. 자세한 내용은 기능 비교 부분을 참조하십시오.

1.3.2 Tomcat

Tomcat에는 설치 관리자 기능이 없습니다.[5] 자동 설치하려면 SSH(Secure Shell)로 설치 스크립트를 작성하고 백업 및 복원을 수작업으로 해야 합니다. Mulesoft의 Tcat[6], Puppet Labs의 Puppet[7], Opscode의 Chef[8] 등에서 이러한 기능 중 일부를 이용할 수 있습니다.

1.4 업그레이드

1.4.1 리버티

설치 유형에 따라 IBM 설치 관리자를 사용하여 리버티를 자동 업그레이드하거나 신규 배포된 아카이브를 받아 수동으로 업그레이드할 수 있습니다. 기존의 서버 구성을 수정하지 않고 업데이트된 런타임으로 사용할 수 있습니다.

1.4.2 Tomcat

Tomcat은 zip 파일로만 제공되므로 모든 업데이트는 수동으로 해야 합니다.[9] 추가한 타사 라이브러리는 개별적으로 업데이트해야 하며, 따라서 Tomcat과의 호환성은 물론 사용 중인 다른 타사 라이브러리와의 호환성을 직접 확인해야 합니다. 이 작업을 위해 구성 파일 역시 필요한 부분을 수작업으로 고쳐야 합니다.

1.5 필수 선행조건

리버티 및 Tomcat 모두 Java 버전 6 이상이 필요합니다. 리버티 및 Tomcat 모두 규격 J2SE(Java 2 Platform Standard Edition)로 실행되며 따라서 다양한 운영체제를 선택할 수 있습니다.

1.6 요약 표

동작	리버티	Tomcat	참고
아카이브 추출 설치	예	예	
관리 설치	예(1)	아니오(2)	1. 웹스피어 애플리케이션 서버 리버티 프로파일은 IBM 설치 관리자로 설치 가능 2. Apache Tomcat은 Mulesoft의 Tcat, Puppet Labs의 Puppet 또는 Opscode의 Chef 같은 제품으로 설치 가능
업그레이드	예(3)	예(4)	3. 웹스피어 애플리케이션 서버 리버티 프로파일은 IBM 설치 관리자를 사용한 자동 업그레이드 또는 수동 업그레이드 가능 4. 타사 라이브러리의 장애 방지에 주의

2. 개발 환경

리버티 및 Tomcat 모두 Eclipse JavaEE에서 서버로 사용할 수 있습니다. 여기서는 이러한 접근 방식별로 설치 절차를 비교해 보겠습니다.

2.1 리버티

IBM 웹스피어 애플리케이션 서버 v8.5 - 이클립스용 리버티 프로파일 개발자 툴(Liberty Profile Developer Tools for Eclipse)은 Eclipse Marketplace 또는 웹스피어 애플리케이션 서버 개발자 커뮤니티 웹사이트(wasdev.net)에서 구할 수 있습니다. 표준 Eclipse 업데이트 기능을 사용하여 Java EE Developers 3.7.1(Indigo) 및 이후 버전용의 Eclipse 통합 개발 환경에 이러한 도구를 설치할 수 있습니다. 이 플러그인을 간단한 도구 모음으로 사용하여 리버티용 애플리케이션을 개발, 구성 및 배포할 수 있습니다. 필요한 경우, Eclipse에서 리버티 프로파일 서버 정의를 처음 생성할 때 이 도구 플러그인을 함께 다운로드하고 런타임도 설치할 수 있습니다.

리버티 도구의 기능은 다음과 같습니다.

1. 도구와 런타임 사이의 광범위한 통합. 서버 보기에 직접 구성 표시, 문제점 해결을 위한 콘솔 링크 제시 등
2. 변수 및 참조 지원을 포함한 양식 기반의 구성 편집기
3. 패키지 등 런타임 유틸리티 지원, plugin-cfg.xml, SSL 인증 생성 및 덤프
4. 공유 라이브러리 지원
5. 여러 가지 런타임, 서버 및 공유 구성 스니펫에 대한 관리 지원 (런타임 탐색기)

나머지 이클립스용 웹스피어 개발자 툴의 기능은 다음과 같습니다.

1. 데이터 정의 및 JPA(Java Persistence API)를 위한 양식 기반의 편집기, 엔터프라이즈 탐색기, 검증 및 "빠른 픽스" 등 향상된 JavaEE 도구
2. 리치 페이지 편집기(WYSIWYG HTML 및 JSP 편집기), Dojo 지원, 향상된 JavaScript 도구, 디버깅 기능 등 웹 및 모바일 도구
3. 블루프린트 및 애플리케이션 편집기 등 완벽한 OSGi 개발자 도구와 번들 종속관계 뷰어
4. 웹스피어 애플리케이션 서버 프로그래밍 모델 지원: 웹스피어 애플리케이션 서버 JPA 플랫폼, 양식 기반의 바인딩 및 확장 파일 편집기

2.2 Tomcat

Java EE 개발자를 위한 Eclipse 통합 개발 환경을 설치하면 Eclipse에서 Tomcat을 사용할 수 있습니다. Tomcat은 Eclipse JEE에서 사용 가능한 기본 서버 목록에 포함되어 있습니다.[10] 서버 설정 과정에서 시스템에 설치된 기존의 Tomcat을 지정해야 합니다. Tomcat 런타임을 다운로드하기 전이라면 Eclipse 서버 마법사의 버튼을 눌러 Tomcat 런타임을 다운로드 및 설치할 수 있습니다.

2.3 요약 표

기능	리버티	Tomcat	참고
Eclipse JavaEE 서버	예(1) (2)	예	1. 플러그인은 Eclipse Marketplace에서 받을 수 있음 2. 웹스피어 애플리케이션 서버 v8.5 - 이클립스용 리버티 프로파일 개발자 툴 및 IBM Rational® Application Developer Tools 도구 모음으로 기능 향상

3. 구성

리버티 및 Tomcat은 둘 다 server.xml 파일을 사용하여 구성합니다. 두 환경 모두 server.xml에 다른 xml 파일을 포함시킬 수 있으므로 모듈 방식으로 구성할 수 있습니다. 리버티 및 Tomcat 환경 모두 Eclipse를 사용한 서버 편집 및 구성이 가능하며, 일단 설치한 뒤에는 두 환경 모두 일반적인 Eclipse "서버에서 실행" 작업을 지원합니다.[11]

3.1 리버티

리버티는 구성이 단순하며 전체 구성 기본값 세트는 런타임에 내장되어 있습니다. 이 기본값은 개발 환경에서 원활한 서버 작동을 보장하도록 설계되어 있습니다. 리버티 런타임에 의해 생성되는 서버 기본값은 서블릿 및 JSP 지원 등이며, 기본 HTTP 포트 구성도 포함되므로 변경하기 쉽습니다. 추가 기능은 대개 server.xml에 한 행으로 추가할 수 있습니다. server.xml 파일을 이용한 리버티 구성은 최대한 간결하고 단순하게 실행되도록 설계되어 있습니다. 리버티 런타임에서 구성 파일을 모니터링하며, 변경사항은 자동으로 런타임에 적용됩니다. 따라서 서버를 다시 시작할 필요가 없습니다.

3.2 Tomcat

Tomcat의 server.xml은 거의 모든 Tomcat 기능을 사용할 수 있도록 구성된 상태로 제공되며 길이가 142행에 이릅니다.[12] Tomcat에 애플리케이션을 배포하기 전에 server.xml을 검토하는 것이 좋습니다. 필요 없는 기능은 사용자가 비활성화해야 합니다. server.xml을 변경하려면 Tomcat 서버를 다시 시작해야 합니다.[13]

3.3 요약 표

특징	리버티	Tomcat	참고
최소한의 간결한 구성	예	아니오	
유연한 모듈식 구성	예	예	
server.xml 동적 업데이트	예	아니오(1)	1. server.xml 변경 후 재시작 필요

4. 애플리케이션 배포

Eclipse의 애플리케이션 배포 절차는 리버티와 Tomcat 모두 유사합니다. 웹스피어 애플리케이션 서버(리버티 프로파일 포함) 및 Tomcat은 표준 JavaEE 오퍼링을 넘어서는 서버별 애플리케이션 프로그래밍 인터페이스(API)를 제공합니다.[14] 리버티는 웹스피어 애플리케이션 서버의 프로파일이기 때문에 리버티 프로파일에서 웹스피어 애플리케이션 서버 API를 사용하는 애플리케이션은 웹스피어 애플리케이션 서버 풀 프로파일에서도 동일하게 실행됩니다.

4.1 웹 애플리케이션

개발자는 Eclipse JavaEE의 관점에서 웹-애플리케이션 레이아웃을 파악할 수 있습니다. 이렇게 해서 필요한 기본값 파일을 생성하면 순식간에 애플리케이션을 서버에 배포 및 실행할 수 있게 됩니다.

4.2 배포

리버티 및 Tomcat은 서버 보기에 표시되며, 서버는 대단히 손쉽게 구성, 시작 및 중지할 수 있습니다. 배포 측면에서는 리버티가 약간 더 유리합니다. 앞서 설명한 바와 같이, 리버티 서버는 server.xml을 변경한 후에 다시 시작할 필요가 없습니다. 다시 말해, 애플리케이션을 개발, 수정 및 재배포하는 동안 서버를 계속 실행할 수 있다는 뜻입니다. 개발자는 .war 파일을 BHAGAYAtomcat-installSACHIN/webapps 디렉토리에 복사함으로써 Eclipse 외부에서 Tomcat에 애플리케이션을 배포할 수 있습니다.

리버티 런타임에서 .war, .ear, .eba 및 .wab 애플리케이션 파일을 서버의 <liberty-install>/usr/servers/<servername>/dropins 디렉토리에 복사하여 배포하거나, server.xml 구성 파일에서 이러한 파일 유형을 직접 애플리케이션으로 지정할 수 있습니다.

Tomcat 구성 파일인 server.xml에서 하위 컨텍스트 요소를 host에 추가하여 애플리케이션을 정의할 수도 있지만, 이 애플리케이션을 실행하려면 사용자가 서버를 다시 시작해야 합니다.[15]

4.3 요약 표

기능	리버티	Tomcat	참고
애플리케이션 개발	예	예	
손쉬운 배포	예(1)	예(1)	1. 동시 배포 디렉토리 사용 가능
구성 내 애플리케이션	예	예	
server.xml 동적 업데이트	예	아니오	

기능 비교

앞서 설명했듯이 리버티 프로파일에는 여러 가지 JavaEE 기능이 있지만 Tomcat은 본질적으로 서블릿 컨테이너에 불과합니다. 아래 표에서 양쪽 환경의 여러 가지 기능을 비교해 볼 수 있습니다.

1. 기본 기능

기능성	리버티	Tomcat	참고
서블릿 3.0	예	예	
JSP 2.2	예	예	
SSL(Secure Socket Layer)	예	예	
FIPS(Federal Information Processing Standard) 140-2	예	예	
FIPS 800-331	예	아니오	
JDBC(Java Database Connectivity)	예	예	
LDAP(Lightweight Directory Access Protocol)	예	예	
JNDI(Java Naming and Directory Interface)	예	예	
JMX(Java Management Extensions)	예(1)	예	1. 로컬 및 원격 REST(Representational State Transfer) 커넥터 있음
공유 라이브러리	예	예	다양한 개념(다음 섹션 참조)
서블릿 보안	예	예	
JSF(JavaServer Faces)	예	아니오(2)	2. Apache MyFaces
JPA(Java Persistence API)	예	아니오(3)	3. Apache OpenJPA
JTA(Java Transaction API)	예	아니오(4)	4. Apache Geronimo 트랜잭션, JOTM
WAR(Web Archive) 애플리케이션	예	예	
EAR(Enterprise Archive)	예(5)	아니오(5a)	5. EJB(Enterprise JavaBeans) 지원(5a) 없는 Apache OpenEJB
WAB(Web Application Bundle)	예	아니오(6)	6. Apache Geronimo에 한함
EBA(Enterprise Bundle Archive)	예	아니오(7)	7. Apache Geronimo에 한함
Bean 검증	예	아니오(8)	8. Apache Bean 검증
IBM z/OS® 지원	예	아니오(9)	9. Dovetailed Technologies T:Z
iSeries 지원	예	예	

리버티에는 JAX-RS, OSGi-JPA, 웹 보안 기능 및 다수의 IBM z/OS 활용 기능 등 런타임 기능도 많이 있습니다.

2. JVM(Java Virtual Machine) 비교

다음은 JVM 간의 몇 가지 차이점입니다(지원 요소 제외).

기능	리버티 프로파일 (IBM J9)	Tomcat(Hotspot Open JDK)	참고
크기가 큰 힙(>4GB)에서 빠른 가비지 수집 -xgcpolicy: 밸런스 유지	예	아니오	
메모리 사용량 감소 및 빠른 시작을 위한 시스템-클래스 데이터 공유	예	예(1)	1. 클라이언트에 한함
메모리 사용량 감소 및 빠른 시작을 위한 애플리케이션-클래스 데이터 공유	예	아니오	
PermGen이 가득 차면 JVM 재시작	예	아니오	
빠른 런타임 및 메모리 사용량 감소를 위한 압축 64비트 참조	예	예(2)	2. 최근 추가
지연(hang), 중단 및 메모리 관리를 위한 덤프 분석기	예	예	IBM의 덤프 분석기가 더 우수
메모리 사용량 및 성능 모니터링을 위한 메모리 표시기 및 가비지 수집	예	아니오	
메모리 누출 및 과도한 힙 사용 문제 해결을 위한 메모리 분석기	예	예	
가상 시스템 실행을 거의 실시간으로 모니터링하는 상태 센터	예	아니오	

3. FIPS 규격의 JDK 1.6 및 1.7

FIPS(Federal Information Processing Standard) 공고문 제140-2호, 즉 FIPS PUB 140-2는 미국 정부의 컴퓨터 보안 표준으로 암호화 모듈 인증에 사용됩니다. 제목은 Security Requirements for Cryptographic Modules(암호화 모듈에 대한 보안 요구사항)입니다. 첫 번째 공고는 2001년 5월 25일 발표되었으며 마지막 개정은 2002년 12월 3일에 이루어졌습니다.[16] 더 강력한 암호화 키와 더 견고한 알고리즘으로 바꾸기 위한 구체적인 전환 지침을 담은 최신 FIPS 공고문 초안은 특별 공고(SP) 제800-131호 A Framework for Designing Cryptographic Key Management Systems(암호화 키 관리 시스템의 설계 프레임워크)입니다.[17]

3.1 리버티

리버티의 기능 또는 배포된 다른 애플리케이션에 영향을 주지 않고 FIPS 규격의 JDK로 리버티를 실행할 수 있습니다. 또한 리버티는 개정된 FIP 800-131을 지원합니다.

3.2 Tomcat

Tomcat FIPS 규격은 버전 7.0.23 이상을 기준으로 개발되었으며 이후 6.0.36 버전에 맞게 조정되었습니다.[18] FIPS 규격 모드에서 Tomcat을 사용할 때는 이 최소 버전을 사용하고, 타사 라이브러리는 이러한 Tomcat 중 비교적 최신 버전과 호환되는 것으로 선택해야 합니다.

3.3 요약 표

특징	리버티	Tomcat	참고
FIPS 140-2	예	예	
FIPS 800-131	예	아니오	

4. JTA(Java Transaction API)

JTA(Java Transaction API)는 분산 트랜잭션 시스템의 구성요소, 즉 리소스 관리자, 애플리케이션 서버 및 트랜잭션 어플리케이션 등과 트랜잭션 관리자 사이의 표준 Java 인터페이스를 지정합니다.

4.1 리버티

웹스피어 애플리케이션 서버 트랜잭션 관리자로 복구 가능한 "XA(eXtended Architecture)" 트랜잭션을 지원합니다.

4.2 Tomcat

Tomcat에는 JTA 기본 지원이 없습니다.[19] 그에 상응하는 기능을 제공하려면 타사 라이브러리를 사용해야 합니다.

4.3 요약 표

특징	리버티	Tomcat	참고
JTA	예	아니오(1)	1. Apache Geronimo 트랜잭션, JOTM 사용

5. JPA(Java Persistence API)

JPA(Java Persistence API)는 관계형 데이터베이스에 접근하기 위한 표준 ORM(Object and Relational Mapping) 기술입니다. Java 개발자는 이 매핑 기술을 이용하여 Java 애플리케이션의 관계형 데이터베이스를 관리합니다.

5.1 리버티

리버티 프로파일에서 JPA(Java Persistence API)를 사용하는 애플리케이션을 지원하려면 server.xml 파일에 jpa-2.0 기능을 추가해야 합니다. 이와 함께 지속성 컨텍스트 및 지속성 단위도 정의하고, 개체 관리자 및 개체 관리자 팩토리에 대한 액세스 권한도 설정해야 합니다. jpa-2.0 기능은 JPA 2.0 사양으로 작성된 애플리케이션 관리 JPA 및 컨테이너 관리 JPA를 사용하는 애플리케이션을 지원합니다. 웹스피어 애플리케이션 서버 풀 프로파일과 같이 Apache OpenJPA 위에 지원 기능이 구축되어 있으며, 확장 팩으로 컨테이너 관리 프로그래밍 모델을 지원합니다.

5.2 Tomcat

Tomcat에는 JPA가 없습니다. 이 기능을 지원하려면 Apache OpenJPA 같은 제품을 사용하고 이를 Tomcat 설치에 통합해야 합니다.

5.3 요약 표

특징	리버티	Tomcat	참고
JPA	예	아니오(1)	1. Apache OpenJPA 사용

6. JDBC(Java Database connectivity)

JDBC 애플리케이션 프로그래밍 인터페이스는 단순한 것을 단순하게 유지하도록 설계되어 있습니다. 즉, JDBC는 일상적인 데이터베이스 작업을 편리하게 만들어 줍니다. JDBC를 사용하여 일반 SQL(Structured Query Language) 문을 실행하거나 데이터베이스 애플리케이션의 기타 일반적인 작업을 수행할 수 있습니다.

6.1 리버티

리버티 프로파일에서 JDBC 사용 애플리케이션을 지원하도록 하려면 server.xml 파일에 jdbc-4.0 기능을 추가하는 한편 구성 파일을 통해 JDBC 드라이버의 위치를 서버에 알리고 JDBC 드라이버가 데이터베이스 연결에 사용할 속성을 지정해야 합니다. 웹스피어 애플리케이션 서버(리버티 프로파일의 프로비저닝 포함)는 확실한 연결 관리를 보장합니다.

6.2 Tomcat

Tomcat의 기본 데이터 소스 지원은 JDBC 연결 풀인 org.apache.tomcat.jdbc.pool을 기반으로 합니다.[20] 이 풀은 과거 Tomcat이 사용하던 Apache Commons DBCP 프로젝트의 대체물 또는 대안입니다. Tomcat 공식 설명서에는 아직 이러한 변경사항이 반영되지 않았습니다.[21] 아직 공식화되지는 않았지만 javax.sql.DataSource를 구현하는 다른 연결 풀을 사용할 수 있습니다. 그러려면 맞춤형 리소스 팩토리를 직접 작성하면 됩니다. 이때 웹 애플리케이션 배포 설명(/WEB-INF/web.xml)

을 수정하여 사전 구성된 데이터 소스를 조회할 JNDI 이름을 선언해야 합니다. 그런 다음 server.xml에 java.sql.DataSource 유형의 리소스 팩토리를 추가하십시오.

6.3 요약 표

특징	리버티	Tomcat	참고
JDBC	예	예	

7. LDAP

LDAP(Lightweight Directory Access Protocol)은 IP(Internet Protocol) 네트워크를 통해 분산 디렉토리 정보 서비스에 액세스하고 이를 유지 관리하기 위한 애플리케이션 프로토콜입니다.

LDAP은 흔히 인증에 사용됩니다.

7.1 리버티

인증을 위해 리버티 프로파일과 함께 LDAP 서버를 구성할 수 있습니다. 그러려면 server.xml 파일에 appSecurity-1.0 서버 기능을 추가하고, server.xml 파일에서 LDAP 서버 연결을 위한 구성 정보를 지정해야 합니다.

7.2 Tomcat

Tomcat Realm 인터페이스 구현인 JNDIRealm을 사용하여 Tomcat에서 LDAP을 구성할 수 있습니다[22]. 이 Tomcat Realm 인터페이스는 JDNI 공급자(일반적으로 JNDI API 클래스에 사용하는 표준 LDAP 공급자)가 액세스하는 LDAP 디렉토리 서버에서 사용자를 조회합니다. 이 유형은 디렉토리를 통한 인증 방법을 광범위하게 지원합니다.

7.3 요약 표

특징	리버티	Tomcat	참고
LDAP	예	예	

LDAP에 대한 자세한 내용은 이 설명서의 관리 기능 부분을 참조하십시오.

8. 공유 라이브러리

공유 라이브러리란 JAR(Java 아카이브) 파일 또는 여러 웹 애플리케이션에서 사용할 수 있는 파일 집합을 말합니다. 서블릿 사양에 따라 필요한 경우, 리버티 및 Tomcat 모두 "글로벌" 공유 라이브러리를 지원합니다.

특정 디렉토리에 공유 라이브러리를 놓을 수 있습니다. 리버티의 경우 <liberty-install-dir>/usr/shared/lib/global , Tomcat의 경우 <tomcat-install-dir>/common/lib 디렉토리를 사용합니다. 서버에 배치된 모든 애플리케이션에서 이 라이브러리를 볼 수 있으며 모든 애플리케이션은 해당 클래스의 동일한 인스턴스에 액세스하게 됩니다.

글로벌 공유 라이브러리 기능은 일부 시나리오에서 유용한 기능이지만 두 가지 주된 제한사항이 있습니다.

- 모든 애플리케이션이 라이브러리 클래스의 동일한 정적 변수 파악 가능(애플리케이션이 서로 분리되지 않음)
- 서로 다른 애플리케이션에 사용하기 위해 동일한 라이브러리를 여러 버전으로 제공하는 것이 불가능합니다.[23]

Tomcat에서 이러한 제한을 피하려면 라이브러리가 필요한 애플리케이션별로 해당 라이브러리의 사본을 별도로 패키징해야 합니다. 반면 리버티는 공유 라이브러리 옵션을 추가로 제공하므로 공유 라이브러리의 사용 방식에 따라 유연성을 발휘할 수 있습니다.

8.1 리버티

리버티의 공유 라이브러리는 다음 세 종류입니다.

1. 모든 애플리케이션이 공통의 라이브러리 인스턴스를 공유하는 글로벌 공유 라이브러리(이 문서 앞부분에 설명)
2. 특정 애플리케이션끼리 공통의 라이브러리 인스턴스를 공유하는 공용 라이브러리
3. 특정 애플리케이션끼리 비공개 라이브러리 인스턴스를 사용하는 비공개 라이브러리

server.xml에서 JAR 파일 및 클래스 컬렉션을 공유 라이브러리로 지정할 수 있습니다. 그런 다음 해당 애플리케이션의 구성 항목을 통해 애플리케이션 클래스 경로에 이 라이브러리를 추가하고, 해당 애플리케이션의 라이브러리 사용 유형도 "공용" 또는 "비공개"로 정의하면 됩니다. 일반적으로, 설치된 리버티의 특정 디렉토리에 공유 라이브러리를 복사해 넣거나 자체 설치 경로에서 직접 공유 라이브러리를 참조할 수 있습니다.

다음은 서버 구성 파일(server.xml)에 추가된 예제 항목입니다.

```
<library id="AppSharedLibrary">
<fileset dir="${shared.resource.dir}/
lib" includes="*.jar" />
</library>
```

애플리케이션 요소의 classloader 하위 요소에서 라이브러리에 액세스할 수 있도록 하려면 다음과 같이 입력하십시오.

```
<application location="MyFirstApp.war">
<classloader commonLibraryRef=
"AppSharedLibrary"/>
</application>
<application location="MySecondApp.war">
<classloader privateLibraryRef=
"AppSharedLibrary"/>
</application>
```

MyFirstApp은 MyFirstApp이 commonLibraryRef로 지정된 다른 모든 애플리케이션과 AppSharedLibrary 클래스를 공유하게 됩니다. MySecondApp은 AppSharedLibrary 클래스의 자체 비공개 사본을 가지게 됩니다. 공용 라이브러리는 여러 애플리케이션에서 여러 버전의 라이브러리를 볼 수 있도록 해야 하는 경우에 유용하지만, 해당 애플리케이션에서 동일한 인스턴스를 버전별로 안전하게 공유할 수 있으므로 라이브러리 클래스의 메모리 사용량 절감에도 도움이 됩니다.

비공개 라이브러리는 공유 라이브러리 인스턴스를 애플리케이션별로 분리해야 하거나, 라이브러리에서 애플리케이션 클래스에 액세스해야 하는 경우에 유용합니다.

8.2 요약 표

특징	리버티	Tomcat	참고
글로벌 공유 라이브러리	예	예	
여러 버전의 동일 라이브러리	예	아니오	
라이브러리 클래스의 비공개 인스턴스	예	아니오	

9. OSGi 애플리케이션

WAB(Web Application Bundle)는 웹 애플리케이션 번들로서, OSGi 동적 모듈 체계를 지원하는 웹 컨테이너에 배포할 수 있습니다. WAB는 OSGi 엔터프라이즈 사양에 포함시켜 정의합니다. 아주 간단하게 말하자면 WAB는 웹 애플리케이션 아카이브(WAR) 파일의 OSGi 번들 버전이라고 할 수 있습니다. WAB는 기존의 JavaEE 순서인 WEB-INF 또는 클래스, WEB-INF의 jar 또는 lib가 아니라 OSGi 번들 클래스 경로를 사용합니다. WAB는 서블릿, JSF, 정적 컨텐트 또는 JSP(JavaServer Pages)를 포함할 수 있습니다.

EBA(Enterprise Bundle Aarchive) 파일에는 다른 OSGi 애플리케이션과 분리하여 단일 OSGi 애플리케이션으로 배포할 수 있는 일련의 OSGi 번들이 들어 있습니다. 각각의 OSGi

애플리케이션은 분리된 자체 OSGi 프레임워크 인스턴스에서 실행되며 자체 OSGi 서비스 레지스트리를 가집니다. 양쪽 OSGi 애플리케이션에서 번들, 서비스 또는 패키지를 명시적으로 공유하지 않는 한, 특정 OSGi 애플리케이션의 번들은 다른 OSGi 애플리케이션에 정의되어 있는 패키지나 서비스 또는 번들을 볼 수 없습니다.

OSGi 애플리케이션은 공유된 번들 공간, 즉 OSGi 애플리케이션의 모든 분리된 프레임워크 인스턴스의 상위 격인 OSGi 프레임워크 인스턴스에서 OSGi 서비스를 사용하거나 패키지를 로드할 수도 있습니다.

9.1. 리버티

리버티는 OSGi 프레임워크에 구축되며 OSGi 애플리케이션 배포를 지원합니다. WAB(Web Application Bundle) 및 EBA(Enterprise Bundle Archive)는 OSGi 애플리케이션 유형으로 지원됩니다.

9.2. Tomcat

Tomcat은 OSGi 컨테이너가 아니며 OSGi 애플리케이션을 지원하지 않습니다.[24]

9.3. 요약 표

특징	리버티	Tomcat	참고
OSGi 애플리케이션	예	아니오	

관리 기능

1. 보안 고려사항

1.1. 리버티

리버티 프로파일에서는 보안 유지를 위해 appSecurity-1.0 서버 기능을 사용합니다. 이 기능으로 사용자 레지스트리, 인증 및 권한 부여를 지원할 수 있습니다. 사용 가능한 사용자 레지스트리 유형은 기본 사용자 레지스트리 및 LDAP 사용자 레지스트리입니다. appSecurity-1.0 기능으로 웹 애플리케이션의 보안 수준을 높일 수 있습니다. 보안 구성에 따라서는 보안 HTTPS 리스너를 통한 SSL(Secure Socket Layer) 연결을 지원하는 ssl-1.0 기능 등 다음과 같은 추가 서버 기능을 하나 이상 추가해야 할 수도 있습니다.

웹스피어 애플리케이션 서버 리버티 프로파일의 요약

a. 인증

a.1. 기본

인증을 위해 리버티 프로파일의 기본 사용자 레지스트리를 구성할 수 있습니다.
비밀번호는 평서문으로 사용하거나 base64로 암호화합니다.

a.2. LDAP

인증을 위해 리버티 프로파일과 함께 LDAP(Lightweight Directory Access Protocol) 서버를 구성할 수 있습니다.

지원되는 LDAP 서버 유형은 다음과 같습니다.

- IBM Directory Server
- Microsoft Active Directory Server

a.3. JAAS

리버티 프로파일 서버 로그인 모듈을 추가하기 이전 또는 이후에 JAAS(Java Authentication and Authorization Service) 로그인 모듈을 커스터마이즈할 수 있습니다.

a.4 SSO(Single Sign-On)
SSO(Single Sign-On) 지원 덕분에 웹 사용자가 일단 리버티 프로파일 리소스(HTML, JSP 파일, 서블릿 등)에 액세스하거나 동일한 LTPA(Lightweight Third Party Authentication) 키를 공유하는 리버티 프로파일 서버 여러 대의 리소스에 액세스하면 인증됩니다.

a.5 TAI
TAI(Trust Association Interceptors)를 사용하여 리버티 프로파일과 타사 보안 서비스를 통합할 수 있습니다. SSO(Single Sign-On)를 호출하기 이전 또는 이후에 TAI를 호출할 수 있습니다. TAI는 타사 보안 서버와 리버티 프로파일 서버 사이의 HTTP 요청 검증에 사용됩니다. TAI는 타사 보안 서버의 HTTP 요청을 검사하여 보안 특성이 있는지 확인합니다. Interceptor에서 요청 검증이 성공적으로 이루어지면 리버티 프로파일 서버는 클라이언트 측 사용자에게 그 리소스에 액세스할 권한이 있는지 여부를 확인한 다음 해당 요청에 권한을 부여합니다.

b. 애플리케이션 보안
server.xml 파일에서 appSecurity-1.0 서버 기능을 활성화하여 리버티 프로파일 서버 인증을 위한 사용자 레지스트리를 구성합니다. 인증 테이블을 구성하려면 해당하는 애플리케이션 요소 아래의 server.xml 파일에서 구성하거나, 웹스피어 애플리케이션 서버 풀 프로파일과의 호환성을 유지하기 위해 ibm-application-bnd.xml 또는 ibm-application-bnd.xmi 파일에 인증 테이블 정의를 추가할 수 있습니다.

1.2. Tomcat
Tomcat은 보안 유지를 위해 여러 가지 인터페이스를 제공하지만 운영환경 용도로 권장되는 인터페이스는 DataSourceRealm[25] 하나뿐입니다. Tomcat 설명서에 나머지 인터페이스에 대한 경고와 함께 매우 구체적인 내용이 나와 있습니다. 따라서 보안을 생각할 때 Tomcat은 상당히 불리한 솔루션입니다.

Tomcat 공식 설명서의 요약 정보[26]:

a. Realms
a.1. MemoryRealm
MemoryRealm은 Tomcat Realm 인터페이스를 시연 목적으로 간단히 구현한 것입니다. 운영환경용으로 설계된 것이 아닙니다. MemoryRealm은 시동 중 모든 사용자 및 해당 역할에 대한 정보를 XML 문서(기본적으로 tomcat-user.xml)에서 로드합니다. tomcat-users.xml의 변경사항을 적용하려면 Tomcat을 다시 시작해야 하기 때문에 MemoryRealm은 운영환경용으로 적합하지 않습니다.

a.2. JDBCRealm
JDBCRealm은 JDBC 드라이버를 통해 관계형 데이터베이스에 액세스하여 사용자를 조회하는 Tomcat Realm 인터페이스 구현입니다. JDBCRealm은 모든 인증 및 권한 부여 옵션에 단일 스레드를 사용하므로 운영환경용으로 권장하지 않습니다. 그 대신 DataSourceRealm을 사용하십시오.

a.3. DataSourceRealm
DataSourceRealm은 JNDI로 명명된 JDBC DataSource를 통해 관계형 데이터베이스에 액세스하여 사용자를 조회하는 Tomcat Realm 인터페이스 구현입니다. UserDatabaseRealm은 대규모 설치에 적합하지 않습니다. 비교적 정적인 소규모 환경용입니다.

a.4. UserDatabaseRealm
UserDatabaseRealm은 JNDI 리소스를 사용하여 사용자 정보를 저장하는 Tomcat Realm 인터페이스 구현입니다. 기본적으로 XML 파일로 이 JNDI 리소스를 지원합니다. 대규모 운영환경용으로 설계된 것이 아닙니다. UserDatabaseRealm은

시동 중 모든 사용자 및 해당 역할에 대한 정보를 XML 문서(기본적으로 tomcat-users.xml)에서 로드합니다. 일반적으로 사용자, 암호 및 역할은 모두 JMX를 통해 동적으로 편집할 수 있습니다. 변경사항은 저장 후 XML 파일에 적용됩니다.

a.5. JAASRealm
JAASRealm은 현재 표준 Java SE API의 구성요소로 제공되는 JAAS(Java Authentication & Authorization Service)를 통해 사용자를 인증하는 Tomcat 6 Realm 인터페이스 구현입니다, JAASRealm은 널리 사용되지 않기 때문에 다른 유형의 인터페이스에 비해 코드가 완벽하지 않습니다. 이 유형은 사용하기 전에 추가로 테스트하는 것이 좋습니다.

a.6. JNDIRealm
JNDIRealm은 Tomcat Realm 인터페이스의 구현 유형으로, JNDI 공급자(일반적으로 JNDI API 클래스에 사용하는 표준 LDAP 공급자)가 액세스하는 LDAP 디렉토리 서버에서 사용자를 조회합니다. 이 유형은 디렉토리를 통한 인증 방법을 광범위하게 지원합니다. 기본적으로 이 유형에서는 어떤 형태의 계정 폐쇄도 실행하지 않습니다. 즉, 폭력적인 공격이 성공할 수 있습니다. 폭력적인 공격을 방지하려면 선택한 유형을 LockOutRealm으로 보호해야 합니다.

b. 애플리케이션 보안
보호할 애플리케이션에 대해 server.xml은 물론 WEBINF/web.xml을 구성하여 웹 애플리케이션의 보안을 유지할 수 있습니다. 보안 및 비보안 애플리케이션을 혼합하여 사용할 수 있습니다. 자세한 내용은 http://tomcat.apache.org/tomcat-7.0-doc/config/http.html#SSL_Support를 참조하십시오.

1.3. 요약 표

특징	리버티	Tomcat	참고
기본 보안	예	예	
LDAP	예	예	
JAAS	예	예	
SSO	예	아니오(1)	1. 직접 구현
TAI	예	아니오	
애플리케이션 보안	예	예	

2. IBM z/OS 통합
최고의 가용성, 보안 강화, 엔터프라이즈 시스템과의 로컬 통합 최적화를 원한다면 IBM z/OS에 웹 애플리케이션을 구축하십시오.

2.1. 리버티
리버티 프로파일은 IBM z/OS 시스템에서 작업 환경을 제공합니다. IBM MVS™ 작업자 명령을 사용하여 리버티 프로파일을 시작, 정지 또는 수정할 수 있습니다. 리버티는 RACF(Resource Access Control Facility) 사용자 레지스트리, SAF(System Authorization Facility) 키링, SAF 권한 부여를 지원합니다. SAF 레지스트리에는 사용자 인증 및 사용자, 그룹 또는 사용자의 연관 그룹에 대한 정보 검색 등 보안 관련 기능을 수행하는 데 필요한 정보가 들어 있습니다. 리버티 server.xml을 구성하여 SAF 레지스트리를 활성화 및 구성합니다. JDBC 드라이버를 사용하여 z/OS의 IBM DB2®에 연결할 수 있으며 IBM z/OS의 기본 트랜잭션 구성 및 문제 해결을 위한 덤프 생성이 가능합니다. z/OS 기본 기능을 사용하여 워크로드 관리에 사용할 애플리케이션 엔드포인트 및 메서드를 분류할 수 있습니다.

2.2 Tomcat

Tomcat은 IBM z/OS를 기본으로 지원하지 않습니다. 다만, Dovetailed Technologies가 IBM z/OS® 운영체제에서 설치, 구성 및 작업하기 쉽도록 특별히 제작한 Apache Tomcat 버전인 "T:Z"라는 제품을 판매하고 있습니다.[27]

2.3 요약 표

특징	리버티	Tomcat	참고
IBM z/OS 작업	예	예(1)	1. 타사의 상업적 Tomcat 버전 사용
IBM z/OS 서비스 품질 사용	예(2)	아니오	2. z/OS 보안, 트랜잭션 및 워크로드 관리

3. 문제 해결

문제 해결은 개발 단계 및 운영 환경에서 중요한 요소입니다. 서버의 상황을 빠르게 파악하기 위한 설비를 갖춰야 합니다.

3.1. 리버티 프로파일

3.1.1. 추적 및 로깅

이 제품에는 통합된 로깅 구성요소가 있습니다. 리버티 프로파일에는 런타임 코드 및 애플리케이션 코드를 위한 FFDC(First Failure Data Capture) 서비스 및 추적 기능이 기본적으로 구현되어 있으므로 디버깅 정보를 수집할 수 있습니다. 이 추적 및 FFDC 기능은 정적 초기화 과정에서 초기 구성을 적용합니다. 서버 구성 파일(리버티 프로파일 런타임 환경 구성 참조) 또는 bootstrap.properties(리버티 프로파일 부트스트랩 속성 지정 참조) 파일에 속성을 지정하여 이 기본 초기화 구성을 수정할 수 있습니다.

메시지는 표준 출력 형식으로 작성되어 정해진 추적 대상 폴더에 기록됩니다. OSGi 로깅 출력을 가로채서 추적 기능으로 출력할 수 있습니다. java.util.logging 출력도 가로챌 수 있습니다. 각각의 리버티 서버는 <ibertyinstall>/usr/servers/<servername>/logs 디렉토리에 자체 로깅, 추적 및 FFDC 정보를 생성합니다. 리버티 프로파일 기반으로 실행되는 웹 애플리케이션은 다음 작업을 할 수 있습니다.

- 시스템 로깅 API, java.util.logging 사용
- Java 서블릿 사양, javax.servlet.ServletContext.log(...)의 로깅 API 사용
- 거의 모든 로깅 프레임워크를 자유롭게 사용

3.1.2. 리버티 프로파일 서버의 상태 포착

명령 프롬프트에서 서버 덤프 명령을 사용하여 리버티 프로파일 서버에 대한 상태 정보를 확인할 수 있습니다. 이 명령은 실행 중이거나 정지된 서버에 사용할 수 있습니다. 이 서버 덤프 명령의 결과 파일에는 작업 영역 디렉토리에 설치된 애플리케이션에 대한 상세 정보, 로그 정보 및 서버 구성이 포함되므로 리버티 프로파일 서버의 문제 진단에 유용합니다. 결과는 "사람이 읽기 쉬운" 텍스트 형식으로 출력됩니다. 서버가 실행 중인 경우 다음 정보도 포함됩니다.

- 서버에 있는 각 OSGi 번들 상태
- 서버에 있는 각 OSGi 번들의 배선 정보
- SCR(Service Component Runtime)로 관리하는 구성요소 목록
- SCR의 각 구성요소에 대한 상세 정보
- 스레드 덤프

3.2. Tomcat

3.2.1. 로깅

Tomcat 서버는 <tomcat-installation>/logs 디렉토리에 몇 가지 로깅 출력을 생성합니다. 자체 로깅 출력을 사용하는 타사 라이브러리를 설치한 경우, 로그 파일을 추가로 생성할 수 있습니다. 예를 들어, Apache OpenJPA는 <tomcat-install>/logs 디렉토리의 openjpa.Log 파일에 출력을 기록합니다.

Apache Tomcat 기반으로 실행되는 웹 애플리케이션은 다음 기능을 수행할 수 있습니다[28].

- 시스템 로깅 API, java.util.logging 사용
- Java 서블릿 사양, javax.servlet.ServletContext.log(...)의 로깅 API 사용
- 모든 로깅 프레임워크를 자유롭게 사용

server.xml의 로깅 수준을 구성하려면

org.apache.catalina.level 속성을 java.util.logging의 수준 중 하나로 설정하면 됩니다. 타사 라이브러리의 경우 로깅 구성을 위해 더 많은 파일을 편집해야 할 수 있습니다. 라이브러리마다 자체 구문이 있습니다. 로그가 통합되어 있지 않으면 여러 출처로부터 정보를 취합해야 하기 때문에 애플리케이션 처리 일정을 평가하기가 어려워집니다.

3.2.2. 서버 덤프

서버 덤프(로그, 추적 파일, JVM 헤드덤프)에 해당하는 작업을 위한 기능은 내장되어 있지 않습니다. Tomcat의 로그 수준을 가장 세밀한 설정으로 변경하고 Tomcat 서버를 실행하는 JVM에 SIGBREAK 신호를 전송하여 유사한 정보를 수작업으로 수집할 수 있습니다.

3.3. 요약 표

특징	리버티	Tomcat	참고
추적 및 로깅	예	예	
FFDC	예	아니오	
서버 덤프	예	예(1)	1. 내장되지 않음, SIGBREAK 사용, 최고 수준의 로그 세밀도 설정

4. 모니터링 및 관리

리버티와 Tomcat은 모두 JMX MBeans가 사전 구현되어 있으며 jconsole을 통한 모니터링 및 관리를 지원합니다.

4.1. 리버티

4.1.1. JMX MBeans

리버티 서버에는 다양한 MBeans가 기본적으로 설치되어 있습니다. 로컬 커넥터에는 보안 조치가 내재되어 있습니다 (서버를 실행하는 운영체제와 동일한 운영체제의 사용자 ID로만 사용 가능). 한편 원격(REST) 커넥터는 server.xml의 단순 구성을 이용하여 항상 철저한 보안 상태를 유지합니다. 사용자는 맞춤형 Mbeans를 개발하고 이를 리버티 서버에 배포할 수 있습니다.

4.1.2. 웹스피어 애플리케이션 서버 Network Deployment 작업 관리자

작업 관리자 콘솔 또는 배포 관리자 콘솔에서 작업을 실행하여 리버티 프로파일 서버를 관리할 수 있습니다. 작업 관리자 구성요소는 웹스피어 애플리케이션 서버 Network Deployment 웹스피어 애플리케이션 서버 에디션에 포함되어 있습니다. 이 구성요소로 호스트 시스템 그룹에 패키지 서버를 설치하기

위한 반복적인 압축 해제-설치 작업을 자동화하고 이후 해당 서버의 라이프사이클을 관리할 수 있습니다. 작업 관리자를 통해 원격 호스트 그룹을 정의하고 이 그룹에 대해 다양한 작업을 실행할 수 있습니다. 예를 들면 웹스피어 애플리케이션 서버 리버티 프로파일 인스턴스의 설치, 제거, 시작 및 정지 작업 등입니다.

기존의 서버 설치에 구성 및 애플리케이션을 추가하여 "임베디드 서버" zip 파일을 만들 수 있습니다. 임베디드 서버 zip 파일을 만들려면 리버티의 패키지 명령을 사용합니다. 이를 통해 동일한 서버 구성의 인스턴스 여러 개를 호스트 시스템 전체에 배포할 수 있습니다. 작업 관리자에 정의된 각 호스트에서 선택적으로 호스트별 변수를 지정하고, 관리 대상인 웹스피어 애플리케이션 서버 리버티 프로파일 인스턴스 각각의 bootstrap.properties 속성에 이를 캡처할 수 있습니다. 임베디드 서버 zip 파일을 호스트 시스템에 복사 및 압축 해제한 다음 구성 파일을 수정하여 로컬 값(이 값은 bootstrap.properties 파일의 변수로 분리 가능)을 업데이트하는 방법으로 작업 관리자를 사용하지 않고 임베디드 서버 zip을 수동 배포할 수도 있습니다.

4.2. Tomcat

4.2.1. JMX MBeans

JAVA_OPTS 환경 변수를 편집하여 로컬 및 원격 커넥터의 보안을 유지할 수 있습니다. Catalina에는 기본적으로 광범위한 MBeans가 배포되어 있습니다. Catalina에 배포된 전체 MBeans 목록은 http://tomcat.apache.org/tomcat-7.0-doc/funcspecs/mbeannames.html을 참조하십시오. 맞춤형 MBeans를 개발하여 JMX API의 범위를 넘어서는 작업도 수행할 수 있습니다. Tomcat 설명서에서는 맞춤형 MBeans 개발 방법에 대해 자세히 다루지 않으므로 인터넷에서 예제를 검색해 보아야 합니다. http://oss.wxnet.org/mbeans.html에서 그러한 예제 중 하나를 볼 수 있습니다.

4.2.2. Mulesoft Tcat

Mulesoft Tcat은 Tomcat 서버 관리를 위한 상업용 버전으로, 사용하려면 운영환경용 용도로 구매해야 합니다. Mulesoft Tcat은 운영환경-배포 위주의 애플리케이션 관리(애플리케이션 배포 및 롤백 포함), 보안, 진단 및 구성 관리 등 추가 기능도 제공합니다.[29]

4.3 요약 표

특징	리버티	Tomcat	참고
JMX MBeans	예	예	
관리	작업 관리자	Mulesoft Tcat	VMware TCserver는 또 하나의 TCat 관리 옵션

5. 확장

리버티 및 Tomcat을 운영 환경에 사용하는 경우도 있기 때문에 확장성은 리버티 및 Tomcat 모두 중요한 기능입니다.

5.1. HTTP 클러스터링 및 로드 밸런싱

5.1.1 리버티

5.1.1.1. 웹 서버 플러그인

웹 서버 플러그인은 지원되는 웹 서버에서 하나 이상의 애플리케이션 서버로 HTTP 요청을 전달할 때 사용됩니다. 이 플러그인은 요청을 받아서 plugin-cfg.xml 파일의 구성 데이터를 기준으로 요청을 점검합니다. 구성 데이터에 따라 HTTP 요청의 URI(Uniform Resource Identifier)를 애플리케이션 서버의 호스트 이름에 매핑합니다. 그런 다음 웹 서버 플러그인은 이 정보를 사용하여 요청을 애플리케이션 서버로 보냅니다.

5.1.1.1.1. 플러그인 생성
defaultPluginConfig 생성 MBean을 통해 리버티 구성을 위한 웹 서버 플러그인 plugin-cfg.xml 파일을 생성할 수 있습니다.

5.1.1.1.2. 플러그인 설치
생성된 병합 plugin-cfg.xml 파일을 웹 서버에 설치합니다. 일반적으로, LoadModule 구문을 사용하여 웹 서버의 httpd.conf 파일 안에 플러그인을 활성화하고, WebSpherePluginConfig 구문을 사용하여 plugin-cfg.xml 파일의 위치를 지정해야 합니다.

5.1.1.1.3. 워크로드 밸런싱 및 페일오버
리버티 서버 여러 개의 플러그인 정보를 병합하고, 해당 서버의 워크로드 밸런싱 및 HTTP 세션의 연관성 유지에 그 플러그인을 사용할 수 있습니다. HTTP 세션 상태를 데이터베이스에 계속 기록해 두면 장애가 발생하여 플러그인이 다른 서버로 트래픽을 보내는 경우에도 그 서버에서 해당 세션의 상태를 확인할 수 있습니다.

5.1.2. Tomcat
5.1.2.1. HTTP 클러스터링
Tomcat에 손쉽게 클러스터를 구성할 수 있습니다. 이렇게 구성된 클러스터는 기본적으로 DeltaManager(Tomcat에 포함)를 사용하는 완벽한 세션 복제를 지원하며, 따라서 세션 델타를 복제할 수 있습니다. 다시 말해, 해당 세션이 클러스터의 다른 모든 노드에 복제됩니다. 단, 소규모 클러스터에서는 이 방법이 통하지만, Tomcat 설명서에 따르면 Tomcat은 노드 수가 많은 대규모 클러스터에서는 사용을 권장하지 않습니다. 또한 DeltaManager를 사용하면 DeltaManager가 해당 애플리케이션이 배포되지 않은 노드를 비롯하여 모든 노드에 복제하게 됩니다. 이 문제를 피하려면 BackupManager를 사용해야 합니다. 이 관리자는 세션 데이터를 백업 노드 하나에만 복제하고, 애플리케이션이 배포된 노드에만 복제합니다. BackupManager의 단점은 DeltaManager처럼 "철저한 현장 테스트"를 거치지 않았다는 점입니다.[30]

5.1.2.2. 로드 밸런싱
Tomcat 서버의 로드 밸런싱에 mod_proxy 모듈로 구성된 Apache HTTP 서버를 사용할 수 있습니다.[31]

5.2. IBM WebSphere eXtreme Scale
WebSphere eXtreme Scale v8.5는 리버티 및 Tomcat 서버 구성을 지원합니다. 특히 Tomcat은 이전 버전에서도 지원됩니다. 웹스피어 애플리케이션 서버 v8.5에서 일반적으로 사용할 수 있는 WebSphere eXtreme Scale은 리버티 프로파일에 향상된 기능을 제공합니다.

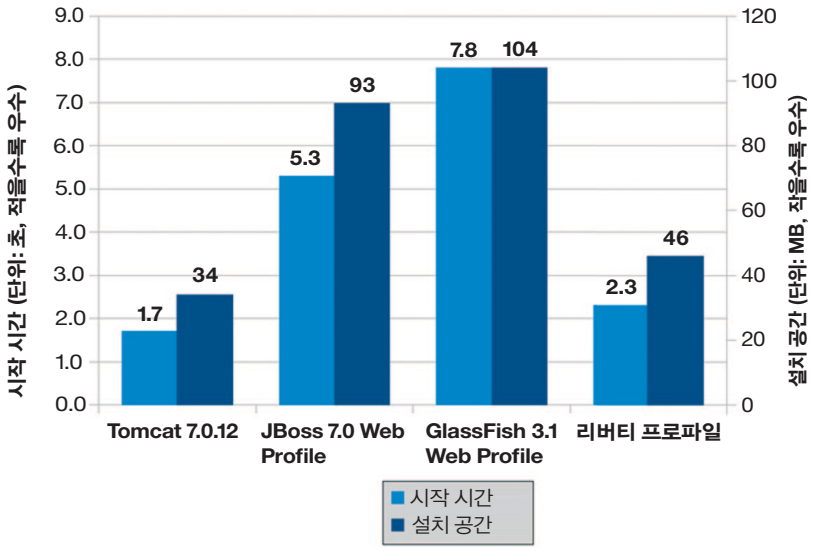
5.3 요약 표

특징	리버티	Tomcat	참고
HTTP 클러스터링	예	예	
로드 밸런싱	예	예	
eXtreme Scale	예(1)	예	1. 향상된 기능, HTTP 세션 지속성

성능

WebSphere는 리버티 출시로 경량화된 개발환경에 대한 이슈를 한 번에 해결하였습니다. 리버티 프로파일의 시동 및 설치 공간은 Tomcat과 거의 비슷한 수준입니다. 그리고 리버티 프로파일은 JBoss 웹 프로파일에 비해 시작 시간이 절반도 안 걸립니다.

다양한 경량 서버의 시작 시간 및 설치 공간 비교



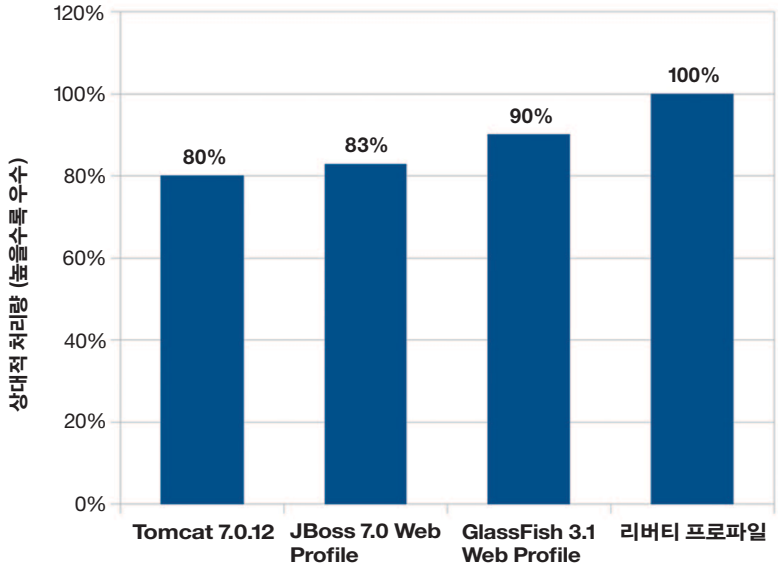
참고: Tomcat, JBoss 및 GlassFish는 HotSpot JDK로, 리버티는 IBM JDK로 측정했습니다.

시스템 정보:
Lenovo T60p - 2.16GHz Intel Core Duo T2600 2개
2GB RAM, Windows XP 32-bit
Apache Tomcat 7.0.12
JBoss Community Edition 7.0 웹 프로파일 서버
GlassFish Server 3.1 Open Source Edition 웹 프로파일
웹스피어 애플리케이션 서버 v8.5 - 리버티 프로파일
(모든 서버에 TradeLite 벤치마크 애플리케이션을 설치했습니다.)
IBM 내부 테스트를 기준으로 합니다. 일반적인 결과가 아닐 수 있으며 운영 환경의 실제 구성, 애플리케이션 및 기타 변수에 따라 달라집니다.

리버티는 완전한 운영 환경 서버의 속도로 서비스 요청을 처리할 수 있는 경량 서버입니다.

리버티 프로파일은 JBoss에 비해 최대 20%, Tomcat에 비해 25% 더 높은 런타임 성능을 발휘합니다.[32]

다양한 경량 서버의 처리량 비교



참고: Tomcat, JBoss, 및 GlassFish는 HotSpot JDK로, 리버티는 IBM JDK로 측정했습니다.

시스템 정보:
IBM x3550 - 1.86GHz Intel Xeon E5320 4개, 8GB RAM
RedHat Linux 5.3 32-bit
Apache Tomcat 7.0.12
JBoss Community Edition 7.0 웹 프로파일 서버
GlassFish Server 3.1 Open Source Edition 웹 프로파일
웹스피어 애플리케이션 서버 V8.5 - 리버티 프로파일

(모든 서버에 TradeLite 벤치마크 애플리케이션을 설치했습니다.)
IBM 내부 테스트를 기준으로 합니다. 일반적인 결과가 아닐 수 있으며 운영 환경의 실제 구성, 애플리케이션 및 기타 변수에 따라 달라집니다.

요약

리버티 및 Tomcat은 기본 서블릿 엔진을 제공하지만 확장을 통해 Java 엔터프라이즈 서비스 및 관련 서비스를 추가로 제공할 수도 있습니다. 리버티에서는 그러한 서비스가 이미 리버티 커널과 통합되어 있으며 공통의 서비스 구성 파일 및 로그 출력을 사용합니다. Tomcat의 경우, 사용자가 타사 라이브러리의 통합을 수행하게 됩니다.

리버티 프로파일 기본 구성은 크기가 매우 작고 더 단순한 반면 Tomcat은 대부분의 기능이 기본적으로 활성화되어 있습니다. 리버티 기능은 필요에 따라 배포된 애플리케이션으로 활성화되며, 따라서 필요한 경우 서버 설치 공간을 최소한으로 줄일 수 있습니다.

리버티는 테스트 및 운영 환경 모두에서 포괄적 테스트를 거친 여러 가지 강화된 보안 기능을 지원하는 반면, Tomcat의 각종 보안 기능은 있는 그대로 제공되며 철저한 테스트를 거치지 않았습니다. 그러므로 운영 환경에는 Tomcat 보안을 권장하지 않습니다.

두 서버 모두 3초 안에 시작되지만 재시작 시간은 Tomcat이 약간 더 빠릅니다. 리버티는 구성이 동적으로 업데이트되기 때문에 재시작해야 하는 횟수가 더 적습니다. 메모리 공간은 비슷하고 양쪽 모두 구성된 기능(리버티) 또는 추가된 라이브러리(Tomcat)에 따라 달라집니다. 리버티 공유 라이브러리 지원의 경우, 애플리케이션에서 동일한 라이브러리를 여러 버전으로 사용해야 할 때 설치 공간을 대폭 줄일 수 있습니다.

리버티는 JSP/JDBC 워크로드의 프로덕션 처리량이 30% 가량 더 우수하며[33], 따라서 대용량 성능이 중요한 경우 리버티를 선택하는 것이 더 좋습니다.

추가 정보

IBM 웹스피어 애플리케이션 서버 리버티 프로파일에 대한 자세한 내용은 IBM 담당자 또는 IBM 비즈니스 파트너에게 문의하거나 **ibm.com**/software/webservers/appserv/was/ 웹사이트 또는 리버티 프로파일에 대한 커뮤니티 포럼인 www.wasdev.net을 방문하시기 바랍니다.

[1] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[2] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[3] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[4] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[5] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[6] http://www.mulesoft.com/understanding-apache-tomcat

[7] http://puppetlabs.com/

[8] http://www.opscode.com/chef/

[9] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[10] http://www.eclipse.org/webtools/

[11] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[12] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[13] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[14] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[15] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[16] http://en.wikipedia.org/wiki/FIPS_140-2

[17] http://csrc.nist.gov/publications/PubsDrafts.html#800-131

[18] https://issues.apache.org/bugzilla/show_bug.cgi?id=50570

[19] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[20] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[21] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[22] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[23] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[24] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[25] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[26] http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[27] http://www.dovetail.com/products/tomcat.html

[28] Tomcat 문서, http://tomcat.apache.org/tomcat-7.0-doc/setup.html

[29] http://www.mulesoft.com/tcat-leading-enterprise-apache-tomcat-application-server

[30] http://tomcat.apache.org/tomcat-7.0-doc/cluster-howto.html

[31] http://tomcat.apache.org/tomcat-7.0-doc/balancerhowto.html

[32] IBM 내부 테스트를 기준으로 합니다. 일반적인 결과가 아닐 수 있으며 프로덕션 환경의 실제 구성, 애플리케이션 및 기타 변수에 따라 달라집니다.

[33] IBM 내부 테스트를 기준으로 합니다. 일반적인 결과가 아닐 수 있으며 프로덕션 환경의 실제 구성, 애플리케이션 및 기타 변수에 따라 달라집니다.

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
2012년 10월

재활용하십시오

WSW14199-KRKO-00