Tomcat 10과 Tomcat 9는 주요 차이점이 있으며, 이 중 가장 큰 변화는 Java EE에서 Jakarta EE로의 네임스페이스 전환이다. 두 버전의 비교를 통해 주요 차이점과 성능, 기능적 차이를 자세히 설명한다.

**Tomcat 10 vs Tomcat 9 비교 표**

|  |  |  |
| --- | --- | --- |
| **항목** | **Tomcat 9** | **Tomcat 10** |
| Java EE/Jakarta EE 버전 | Java EE 8 (Servlet 4.0, JSP 2.3) | Jakarta EE 9 (Servlet 5.0, JSP 3.0) |
| 네임스페이스 | javax.\* 패키지를 사용 | jakarta.\* 패키지를 사용 |
| Servlet 버전 | Servlet 4.0 | Servlet 5.0 |
| JSP 버전 | JSP 2.3 | JSP 3.0 |
| WebSocket 버전 | WebSocket 1.1 | WebSocket 2.0 |
| HTTP/2 지원 | 기본 지원 | 기본 지원, 성능 최적화 및 최신 프로토콜 개선 |
| TLS(SSL) 버전 | TLS 1.2 | TLS 1.3 |
| 비동기 I/O 성능 | 개선된 비동기 I/O, 기본 제공 | 비동기 I/O 성능이 더욱 향상됨 |
| 마이그레이션 | Java EE 기반 프로젝트와의 완벽한 호환성 | javax.\*에서 jakarta.\*로 패키지 네임 변경 필요 |
| 컨테이너화 지원 | 기본적인 Docker 및 Kubernetes 배포 지원 | 향상된 Docker 및 Kubernetes 지원 |
| 호환성 | Java 8, Java 11 | Java 11 이상에서 사용 가능 |
| 메모리 및 CPU 최적화 | 전통적인 메모리 관리 및 CPU 최적화 | 더 향상된 메모리 효율성과 CPU 리소스 관리 |

**주요 차이점 설명**

1. **Java EE vs Jakarta EE (네임스페이스 변화)**
   * Tomcat 9는 Java EE 8을 지원하며, 주요 API들이 javax.\* 네임스페이스를 사용.
   * Tomcat 10은 Jakarta EE 9를 기반으로 동작하며, javax.\*에서 jakarta.\*로 모든 API 네임스페이스가 변경됨. 이는 코드 호환성에 중요한 영향을 미치며, Tomcat 9에서 Tomcat 10으로 마이그레이션 시 패키지 이름 변경 작업이 필수적임.
     + 예: javax.servlet.\* → jakarta.servlet.\*
2. **Servlet 버전**
   * Tomcat 9는 Servlet 4.0을 지원하며, 이는 HTTP/2 지원과 비동기 HTTP 요청의 성능 향상이 이루어짐.
   * Tomcat 10은 Servlet 5.0을 지원하며, Jakarta EE 9의 새로운 네임스페이스 외에도 더욱 향상된 비동기 요청 처리 및 성능 개선이 이루어짐.
3. **JSP 버전**
   * Tomcat 9는 JSP 2.3을 지원하며, 기존의 Java EE API를 기반으로 한 JSP 동작을 유지.
   * Tomcat 10은 JSP 3.0을 지원하며, 더 효율적인 태그 라이브러리 및 표현 언어(EL)가 제공됨.
4. **WebSocket 지원**
   * Tomcat 9는 WebSocket 1.1을 지원하여 실시간 양방향 통신을 처리.
   * Tomcat 10은 WebSocket 2.0을 지원하며, 성능과 안정성이 개선된 실시간 통신 프로토콜을 제공.
5. **HTTP/2와 TLS 지원**
   * Tomcat 9와 Tomcat 10 모두 HTTP/2를 기본 지원하며, 비동기식 요청 처리 성능을 크게 향상.
   * Tomcat 10은 HTTP/2 관련 성능 최적화가 추가되었고, 더 빠른 프로토콜 사용이 가능.
   * 또한, Tomcat 10은 최신 보안 프로토콜인 TLS 1.3을 지원하여 보안성을 강화. Tomcat 9은 TLS 1.2를 기본적으로 지원.
6. **비동기 I/O 성능**
   * Tomcat 9에서는 비동기 I/O 처리가 기본적으로 제공되며 성능이 개선됨.
   * Tomcat 10에서는 비동기 I/O 처리 성능이 더욱 최적화되었으며, 고성능 웹 애플리케이션을 위한 NIO(Non-blocking I/O)와 APR(Apache Portable Runtime) 기반의 성능 개선이 이루어짐.
7. **컨테이너화 및 클라우드 환경**
   * Tomcat 9은 Docker 및 Kubernetes 같은 컨테이너 환경에 배포가 가능하며, 클라우드 상에서 사용이 가능하지만 특별한 최적화가 이루어지지 않음.
   * Tomcat 10은 이러한 컨테이너 환경에서의 배포 및 실행 성능이 더욱 향상됨. Docker 및 Kubernetes에 적합하게 최적화되어, 클라우드 상에서의 실행 성능이 더욱 개선됨.
8. **호환성 및 마이그레이션**
   * Tomcat 9는 Java EE 8을 사용하고 있기 때문에 Java 8과의 호환성이 유지됨.
   * Tomcat 10은 Java 11 이상에서만 동작하며, Jakarta EE 9를 사용하기 때문에 이전 Java EE 기반 애플리케이션은 코드 수정이 필요함. 이를 위해 Eclipse Transformer 같은 도구를 사용해 자동으로 네임스페이스를 변환하는 작업을 수행할 수 있음.
9. **메모리 및 CPU 최적화**
   * Tomcat 9는 기존의 메모리 관리와 CPU 최적화를 제공.
   * Tomcat 10에서는 비동기 처리 성능과 HTTP/2 관련 최적화로 인해 더 나은 메모리 및 CPU 효율성을 기대할 수 있음.

**Tomcat 9 vs Tomcat 10 선택 기준**

1. **애플리케이션 호환성**
   * 기존의 Java EE 8 기반 애플리케이션이 있으며, 이를 변경 없이 그대로 유지해야 하는 경우 Tomcat 9이 더 적합.
   * 반면, 최신 Jakarta EE 9 기술을 사용하고 싶거나 새로운 프로젝트를 시작하는 경우 Tomcat 10을 사용하는 것이 유리.
2. **마이그레이션 비용**
   * Tomcat 9에서 10으로 마이그레이션하려면 javax.\*에서 jakarta.\*로의 패키지 변경 작업이 필수적임. 이 작업은 프로젝트 규모에 따라 상당한 비용이 발생할 수 있으므로 신중히 고려해야 함.
3. **최신 기능 활용**
   * WebSocket 2.0, HTTP/2 최적화, TLS 1.3 등 최신 웹 기술 및 보안 요구 사항을 만족시키기 위해서는 Tomcat 10을 선택하는 것이 좋음.
   * 최신 기술을 적용하는 것이 필수적이지 않다면, Tomcat 9도 여전히 안정적이고 널리 사용되는 옵션임.

**결론** Tomcat 9은 Java EE 8을 기반으로 안정적이며 기존 애플리케이션과의 호환성이 뛰어난 반면, Tomcat 10은 Jakarta EE 9를 지원하며, 최신 웹 표준과 성능 최적화를 제공한다. 새로운 프로젝트나 클라우드 환경을 목표로 하는 애플리케이션이라면 Tomcat 10이 적합할 수 있고, 기존 Java EE 애플리케이션을 유지하고자 한다면 Tomcat 9이 더 나은 선택일 수 있다.