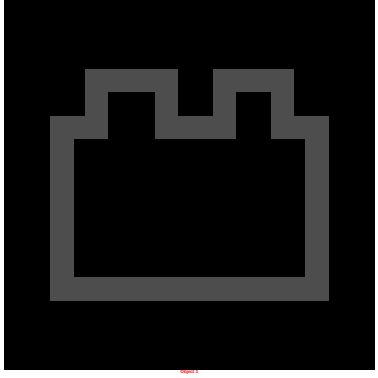


Sumario

Java Enterprise Edition

¿Qué es J2EE?

J2EE es una **plataforma para el desarrollo empresarial** a partir de la cual es posible el desarrollo profesional de aplicaciones empresariales distribuidas sobre una arquitectura multicapa, que son escritas con el lenguaje de programación Java y son ejecutadas desde un servidor de aplicaciones.

J2EE es, por tanto, una plataforma de programación cuya especificación original fue desarrollada por la empresa Sun Microsystems, si bien en el año 2000 la compañía Oracle se hizo con su control.

La versión J2EE tiene su origen en el lenguaje de programación Java correspondiendo sus siglas "EE" a "**Enterprise Edition**". Esta tecnología Java permite a los desarrolladores y programadores crear (escribir) aplicaciones una única vez vez y que sea compatibles sobre cualquier equipo ya que se comunican directamente con la máquina virtual, y no con el sistema operativo.

Más tarde, las necesidad de medios y herramientas que presentaba el sector del desarrollo del software para desarrollar aplicaciones empresariales dio paso a esta plataforma denominada **Java 2 Enterprise Edition**, capaz de proporcionar las especificaciones técnicas que describen el lenguaje a la vez que facilita o suministra las herramientas necesarias que permiten **implementar aplicaciones** (productos de software) basados en dichas especificaciones. Exactamente esto es lo que aprendemos en el **curso de java J2EE**, es decir, enfocado hacia las empresas.

Así, J2EE no es más que un **conjunto de especificaciones**, es decir, siendo cada una de estas especificaciones el detalle de las tecnologías dentro de la plataforma J2EE; el conjunto de normas o directrices bajo las cuales debe desarrollarse esa aplicación de tal manera que pueda desplegarse y ejecutarse cualquier servidor que cumpla con la especificación J2EE.

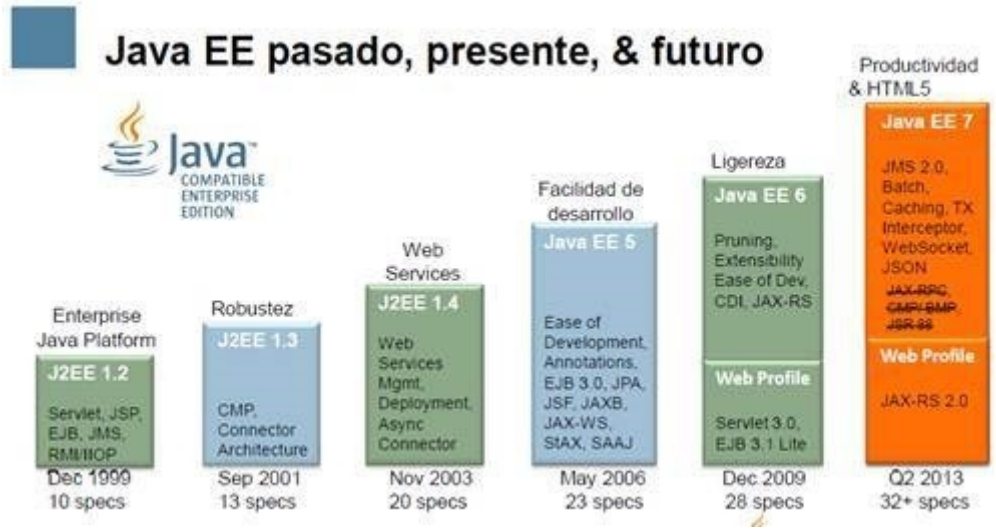
¿Y qué tecnologías incluye Java Enterprise Edition?

- Enterprise JavaBeans (EJB).
- Java Servlet
- JavaServer Page (JSP)
- JavaServer Pages Standard Tag Library (JSTL).
- JavaServer Faces (JSF)
- Java Message Service (JMS).
- Java Transaction API (JTA).
- JavaMail API y JavaBeans Activation Framework (JAF).
- Tecnologías XML (JAXP, JAX-RPC, JAX-WS, JAXB, SAAJ, JAXR) JPA, JDBC API

- Java Naming and Directory Interface (JNDI)
- Java Authentication and Authorization Service (JAAS)

Evolución de Java EE.

JAVA EE ha evolucionado relativamente bien en el tiempo. Es cierto que en sus inicios carecía de numerosas funcionalidades que condicionaron la aparición de Spring Framework, pero ahora con la liberación de JAVA EE7 y próximamente JAVA EE8 estas carencias están resueltas en su mayoría. En la siguiente imagen pueden verla evolución hasta el momento.



Características de Java EE 5

Mediante Java EE 5 Platform Enterprise Edition (Java EE), puede desarrollar aplicaciones de forma más rápida y conveniente que en versiones anteriores. La plataforma Java EE 5 sustituye Java 2 Enterprise Edition (J2EE), versión 1.4. Las herramientas del producto soportan ambas versiones. Java EE 5 aumenta significativamente la facilidad de uso al

- Reducir el tiempo de desarrollo
- Reducir la complejidad de la aplicación
- Mejorar el rendimiento de la aplicación

Java EE 5 proporciona un modelo de programación simplificado, incluyendo las herramientas siguientes:

- Configuración incorporada en línea con anotaciones, lo que convierte ahora en opcionales los descriptores de despliegue
- Inyección de dependencias, lo que permite ocultar la creación y la búsqueda de recursos del código de la aplicación
- La API de persistencia Java (JPA) permite gestionar los datos sin SQL o JDBC explícito
- Utilización de objetos POJO (Plain old Java object) para Enterprise JavaBeans y servicios Web

Java EE 5 proporciona reglas de empaquetado simplificadas para aplicaciones de empresa:

- Las aplicaciones web utilizan archivos .WAR
- Los adaptadores de recursos utilizan archivos .RAR
- Las aplicaciones de empresa utilizan archivos .EAR
- El directorio `lib` contiene archivos .JAR compartidos
- Un archivo .JAR con `Main-Class` implica un cliente de aplicación
- Un archivo .JAR con la anotación @Stateless implica una aplicación EJB
- Muchas aplicaciones simples ya no necesitan descriptores de despliegue, entre las que se incluyen
    - Aplicaciones EJB (archivos .JAR)
    - Aplicaciones web que utilizan solamente la tecnología JSP
    - Clientes de aplicación
    - Aplicaciones de empresa (archivos .EAR)

Java EE 5 proporciona una acceso simplificado a los recursos mediante la inyección de dependencias:

- En el patrón de Inyección de dependencias, una entidad externa proporciona automáticamente las dependencias de un objeto.
    - No es necesario que el objeto solicite estos recursos explícitamente

- En Java EE 5, la inyección de dependencias puede aplicarse a todos los recursos que necesita un componente
    - La creación y la búsqueda de recursos queda oculta del código de la aplicación

- La inyección de dependencias puede aplicarse por toda la tecnología Java EE 5:
    - Contenedores EJB
    - Contenedores web
    - Clientes
    - Servicios Web

Características de Java EE 6

Entre las principales novedades que tiene la siguiente versión de Java EE está la especificación de los Servlets 3.0, en la cual no se requerirán descriptores de despliegue para una aplicación web, sino que será posible configurar lo todo mediante anotaciones.
También se introduce un mecanismo ("web fragments") para que librerías de terceras partes puedan especificar sus descriptores de un modo transparente para el usuario.
Otra de las grandes novedades será Web Beans, que pretende simplificar la integración de la capa de presentación y persistencia de una aplicación web. JPA 2.0, JSF 2.0, y el JSR-311 JAX-RS: Java API for RESTful Web Service son otras novedades.
Tampoco podemos olvidarnos de la inclusión de los perfiles, cuyo propósito es definir subconjuntos estándar del conjunto global de especificaciones que constituyen Java EE. En la versión 6 parece que sólo habrá perfiles orientados a la web. Todavía no se ha alcanzado una decisión definitiva al respecto, pero parece que habrá un perfil que incluirá Servlet 3.0, JSP 2.2, JSR-45, EL 1.2, JSTL 1.2

y JSR-250, y otro que incluirá además de las especificaciones anteriores EJB 3.1 (Lite), JTA 1.1, JPA 2.0, JSF 2.0 y Web Beans 1.0 (así parece que se soluciona la polémica de si se debería incluir EJB litle en el perfil web.

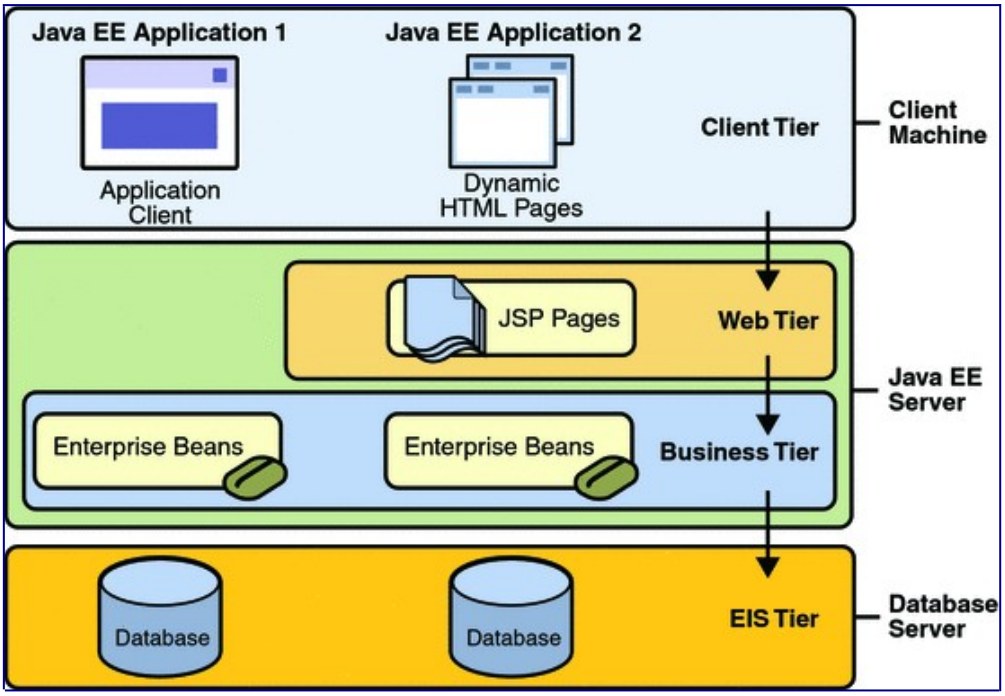
Características de Java EE 7

Java EE 7 es el conjunto de especificaciones a disposición de las aplicaciones empresariales que son implementadas por cada servidor de aplicaciones donde se ejecutan las *apps* Java, estas especificaciones describen la funcionalidad y permiten que una aplicación pueda ser ejecutada en cualquier servidor que las implemente ya sea WildFly, WebLogic o WebSphere sin necesidad de grandes cambios o ninguno en absoluto al proporcionar interoperabilidad entre diferentes implementaciones con el objetivo de no estar encadenado a un determinado vendedor.

Java EE usa como lenguaje de programación Java que incorporando numerosas novedades en la versión 8 lo hacen seguir siendo uno de los mejores lenguajes y más usado para el desarrollo de aplicaciones empresariales. Estos son mis 10 razones para seguir usando Java.

El modelo clásico de capas en la arquitectura de las aplicaciones Java EE se divide en las siguientes:

- Cliente: normalmente se trata de un navegador pero puede ser un teléfono inteligente o una computadora de escritorio, incluso otra aplicación. Es la que presenta la información al usuario.
- Capa web: se comunica con el cliente y la capa de negocio. Obtiene los datos y los transforman al formato adecuado al cliente generalmente o .
- Capa de negocio: proporciona y persiste los datos de la capa cliente y contiene la lógica de negocio de la aplicación. Se ejecuta en un servidor de aplicaciones o contenedor de *servlets*.
- Sistemas de información: donde se persisten los datos de la aplicación, puede ser una base de datos relacional como Oracle, MySQL o PostgreSQL o una base de datos NoSQL como Redis o MongoDB u otros sistemas como Elasticsearch.



En el listado de especificaciones encontramos algunas dedicadas a persistencia en base de datos (JDBC, JPA), transaccionalidad (JTA), procesamiento de peticiones HTTP (Servlets, JSF, REST), generación de HTML (Servlets, JSP, JSF), servicios web basados en REST y SOAP (JAX-RS, JAX-WS), soporte para websockets en el lado del servidor y cliente, generar, procesar y modificar documentos JSON, validación de objetos, comunicación entre aplicaciones desacoplada con mensajes (JMS), concurrencia, servicios de nombres y descubrimiento o trabajos en lotes entre otras. Las especificaciones y versiones que componen Java EE 7 son:

Tecnologías aplicaciones Web (websockets, html)

- **Java API for WebSocket** 1.0 (nueva)
- **Java API for JSON Processing** 1.0 (nueva)
- Java Servlet 3.1
- JavaServer Faces (JSF) 2.2
- Expression Language (EL) 3.0
- JavaServer Pages (JSP) 2.3

Tecnologías aplicaciones empresariales (persistencia, transaccionalidad, inyección de dependencias, concurrencia, validación, mensajería, correos electrónicos)

- **Batch Applications for the Java Platform** 1.0 (nueva)
- **Concurrency Utilities for Java EE** 1.0 (nueva)
- Contexts and Dependency Injection for Java (CDI) 1.1
- Dependency Injection for Java 1.0
- Java Persistence (JPA) 2.1
- Java Transaction API (JTA) 1.2
- Java Message Service API (JMS) 2.0
- Enterprise JavaBeans (EJB) 3.2
- Bean Validation 1.1

- JavaMail 1.5
- Interceptors 1.2
- Java EE Connector Architecture (JCA) 1.7
- Common Annotations for the Java Platform 1.2

Tecnologías servicios web (servicios web REST, SOAP)

- Java API for RESTful Web Services (JAX-RS) 2.0
- Java API for XML-Based Web Services (JAX-WS) 2.2
- Implementing Enterprise Web Services 1.3

Metadatos de servicios web para la plataforma Java

- Java API for XML-Based RPC (JAX-RPC) 1.1 (Opcional)
- Java API for XML Messaging 1.3
- Java API for XML Registries (JAXR) 1.0

Tecnologías de gestión y seguridad (autenticación, autorización)

- Java Authentication Service Provider Interface for Containers (JASPIC) 1.1
- Java Authorization Contract for Containers (JACC) 1.5
- Java EE Application Deployment 1.2 (Opcional)
- J2EE Management 1.1
- Debugging Support for Other Languages 1.0

Especificaciones relacionadas con Java EE en Java SE (bases de datos, XML, gestión)

- Java Database Connectivity (JDBC) 4.0
- Java Architecture for XML Binding (JAXB) 2.2
- Java Management Extensions (JMX) 2.0
- JavaBeans Activation *framework* (JAF) 1.1
- Java API for XML Processing (JAXP) 1.3
- Streaming API for XML (StAX) 1.0

Además de incorporar nuevas especificaciones y las existentes recibir mejoras entre las novedades se encuentran varias que facilitan el desarrollo posibilitando en gran medida prescindir de configuración en archivos propensos a errores, ya se inició en Java EE 6, sustituyéndose por definiciones declarativas con anotaciones. Entre las mejoras están:

- Usando la nueva funcionalidad de entrada y salida (NIO) de Java SE los *servlets* pueden manejar comunicación asíncrona. Se permite *upgrade* del protocolo lo que permite por ejemplo empezar una petición como HTTP/1.1 y pasar a usar HTTP/2, en WildFly esto es usado para reducir el número de puertos abiertos.
- JPA ahora puede invocar procedimientos almacenados, ejecutar sentencias SQL de *update* y *delete* masivas y controlar que entidades son cargadas de forma ansiosa (*eager*) o vaga (*lazy*).
- Se continúa avanzando en lo iniciado en versiones anteriores permitiendo usar POJO con anotaciones para facilitar el desarrollo.
- JTA define una nueva anotación que permite a cualquier *bean* CDI usar transacciones.

- En JAX-RS se añade una API cliente para invocar *enpoints* REST, se añade soporte para E/S asíncrona tanto para el cliente como para el servidor y *hypermedia linking*.
- Bean Validation permite validación a nivel de método con mejor integración en el resto de la plataforma Java EE.

Los servidores de aplicaciones Java pueden implementar todas las especificaciones denominándose *full-profile* como JBoss/WildFly, WebLogic o WebSphere. Sin embargo, algunas aplicaciones no necesitan todas las funcionalidades definidas en Java EE por lo que algunos servidores como TomEE también WildFly pueden implementar únicamente un subconjunto para la generación de contenido web, denominándose así *web-profile*. Otros servidores como Tomcat y Jetty son contenedores de *servlets* que soportan un grupo más reducido de especificaciones de las tecnologías web (únicamente *servlet*, JSP, EL y WebSocket) pero que siguen siendo suficientes para algunas aplicaciones o usando algunas equivalentes proporcionadas por Spring.

Tecnologías *web-profile*

- Java Servlet API
- Enterprise Java Bean Lite
- Context and Dependency Injection
- Java Server Faces
- Java Transaction API
- Java Persistence API
- Web Socket
- Bean Validation
- JAX-RS
- JSON-P

Java EE 7 fué publicada hace ya varios años y muy posiblemente muchas entidades públicas y empresas privadas seguirán usando versiones más antiguas para sus aplicaciones. En el mundo empresarial las aplicaciones se han de mantener funcionando algunos lustros o décadas, para atender este requisito Java en sus 20 años se ha caracterizado por ofrecer compatibilidad hacia atrás en cada nueva versión e incorporar en el lenguaje solo aquellas mejoras que se han mostrado útiles. Esto puede hacer parecer que avanza lentamente, al menos más que otras tecnologías, y que usando las versiones anteriores hoy parezcan obsoletas, en su momento XML era uno de los formatos más empleados para realizar configuración y se usaba profusamente, hoy se están prefiriendo formatos menos verbosos y legibles como JSON y YAML. Java 8 ofrece varias novedades en el lenguaje y Java EE 7 usando anotaciones en gran medida hace innecesarios los ficheros XML ambos siguiendo las nuevas tendencias del desarrollo y programación. Algunas personas comparan las tecnologías que prefieren con lo que recuerdan de versiones antiguas de Java o Java EE.

Java EE ofrece a los desarrolladores un conjunto de especificaciones que cubren las necesidades de un gran número de aplicaciones empresariales y la plataforma ofrece garantías a largo plazo como ha demostrado que le hacen seguir siendo una de opciones preferidas para el desarrollo. La alternativa a Java EE más usada es Spring y sus numerosos proyectos que proporcionan funcionalidades similares sin necesidad de un contenedor *full-profile*. En cualquiera de estas dos opciones la base de procesamiento y generación de HTML en el lado del servidor son los *servlets* pero estos trabajan a bajo nivel, no se suelen usar directamente prefiriéndose *frameworks* de más

alto nivel como Spring MVC, Apache Tapestry, Grails, Struts u otros. Del mismo modo para el acceso a la base de datos está JDBC pero tampoco se suele usar directamente por ofrecer una API a bajo nivel prefiriendo Hibernate, JPA o jOOQ como alternativa a JDBC a los ORM anteriores. En el artículo Nueva visita a Herramientas para un proyecto Java comento algunas herramientas alternativas o complementarias a algunas de las especificaciones de Java EE.

Características de Java EE 8

Oracle anunció su intención de trasladar las tecnologías de Java EE a la Eclipse Foundation en colaboración con otros proveedores y la comunidad. Oracle, Eclipse y otros miembros de la comunidad actualmente están afinando los detalles de la transferencia de tecnología y la gobernanza y proceso continuo dentro de la comunidad de Eclipse.

“El importante lanzamiento de Java Platform Enterprise Edition es algo que, a nuestro parecer, los desarrolladores estarán emocionados de utilizar y al abrir el código de las tecnologías de Java EE a la Eclipse Foundation, hemos sentado las bases para un éxito continuo en el futuro”, comentó Mike Lehmann, vicepresidente de gestión de productos de Oracle. “Oracle está comprometido a trabajar con la comunidad de Java EE y la Eclipse Foundation para continuar la innovación, soporte y evolución de Java de empresas”.

Las características clave de Java EE 8 incluyen:

- Soporte HTTP/2 en Servlet 4.0
- Nuevo API de enlace con JSON y distintas mejoras en JSON-P 1.1
- Expansión de JAX-RS para soportar Server-SentíEvento (eventos enviados por servidor) y un nuevo API de cliente reactivo
- Nueva API de seguridad para aplicaciones basadas en PaaS
- Múltiples mejoras de CDI que incluyen soporte para eventos asíncronos

Resumen del modelo Java EE

J2EE no es un producto es una especificación, en base a esta especificación existen muchas implementaciones.

Las implementaciones son proporcionadas por fabricantes a terceros (IBM, BEA…), estas implementaciones deben de cumplir la especificación J2EE.

Elementos de la especificación Java EE

- Java EE Platform: estándar representado por un conjunto de APIs y directivas, soportadas por un servidor de aplicación.
- Java EE Server: implementación de referencia de un servidor de aplicaciones para Java EE
- Java EE Testsuite: Java EE Compatibility Testsuite (CTS), certificación de implementaciones de Java EE etc.
- Java EE Blueprints: consejos para el desarrollo de aplicaciones Java EE, patrones de diseño y un ejemplo de aplicación.

Existen multitud de entornos de desarrollo para Java, cada compañía con cierto peso en la tecnología Java dispone del suyo propio, JDeveloper (Oracle), NetBeans (Sun), Eclipse (IBM), etc.

Para simplificar el desarrollo web y clarificar las responsabilidades de cada uno de los componentes de un proyecto, las aplicaciones web se dividen en capas.

Inicialmente en un entorno sin capas el desarrollo de aplicaciones mezclaba todas las tareas del lado servidor.

Posteriormente surge el modelo de 3 capas que separa capa cliente, servidor y de datos, y con la evolución de los servidores de aplicaciones surge la necesidad de crear una cuarta capa, para separar la lógica de negocio de la transaccional.

Este tipo de arquitectura proporciona mayor control sobre los elementos que entran en juego en el desarrollo de aplicaciones web.

Dentro de la plataforma Java EE disponemos de varios contenedores. Un contenedor es un entorno de ejecución específico para un conjunto de objetos de un determinado tipo y con unos fines concretos.

Por ejemplo, el contenedor Web es un contenedor específico para Servlets y páginas JSP, y que por tanto provee ciertos servicios útiles para este tipo de objetos, como son la librería JSTL de etiquetas JSP, o la infraestructura de JSF. En el contenedor de EJB no se proveen estos servicios ya que no son necesarios para los objetos que albergan.

Las aplicaciones Java EE están establecidas para Internet. Toda aplicación JEE debe tener un servidor de aplicaciones en el que se incluyan los diversos elementos que tengamos ubicados en nuestro proyecto.

Dentro de las aplicaciones JEE existen tres niveles:

- Maquina Cliente: Es lo que el cliente visualizará al acceder a nuestra aplicación.
- Servidor J2EE: Es el servidor de aplicaciones que contendrá en su interior toda la lógica que hayamos incluido en el proyecto, elementos tales como páginas JSP, servlets o Enterprise JavaBeans.
- Servidor BBDD. Es el sistema que ofrecerá información a nuestro proyecto y que podremos representar en el cliente mediante los elementos incluidos en el servidor de aplicaciones.

Diferentes versiones y plataformas Java existentes

Existen varias Ediciones de Java, cada una de ellas diseñada para cierto entorno en particular.

Las ediciones de Java más importantes son:

* Java Standard Edition (Java SE)

* Java Micro Edition (Java ME)

* Java Entreprise Edition (Java EE)

* Java Card

Java Standard Edition es la edición que se emplea en computadoras personales (desktops y laptops). Se le conoce también como Java Desktop (escritorio) y es la versión que necesitamos instalar para poder programar en Java en el ordenador.

A pesar de que su utilidad sirva para crear aplicaciones de escritorio, su enfoque está destinado a comprender el lenguaje Java y los conceptos básicos del lenguaje.

J2SE es una colección de APIs del lenguaje de programación Java útiles para muchos programas de la Plataforma Java.

La Plataforma Java Enterprise Edition incluye todas las clases en el entorno Java SE, además de librerías que son utilizados en cualquier desarrollo Java, independientemente del tipo de desarrollo.

Java Micro Edition es la edición de Java que se emplea en dispositivos móviles, tales como los teléfonos móviles.

Es una versión reducida de Java SE standard con ciertas extensiones enfocadas a las necesidades particulares de este tipo de dispositivos.

La plataforma Java Micro Edition, o Java ME (anteriormente J2ME), es una colección de APIs en Java orientadas a productos de consumo móviles, tales como PDAs, teléfonos móviles o electrodomésticos.

Java ME se ha convertido en una gran opción en el momento de crear un desarrollo para dispositivos móviles, como pueden ser juegos en PDAs.

Ofrece un entorno de desarrollo en el que podemos emular en un PC todo el entorno gráfico durante la fase debug del proyecto y, posteriormente, implementar el proyecto ya creado en el dispositivo móvil.

Al utilizar tecnologías Java en el desarrollo de aplicaciones o juegos con estas APIs, resulta bastante sencillo de migrar las aplicaciones a otros dispositivos, debido a que J2ME es una “mini” edición de Java Enterprise Edition.

J2EE es un grupo de especificaciones diseñadas por Sun que permiten la creación de aplicaciones empresariales enfocadas al entorno web.

J2EE es solamente una especificación sobre un conjunto de “reglas” que deben seguir las páginas web de servidor para crear aplicaciones empresariales. Algunos productos siguen todas las especificaciones, pero no todos cumplen es estándar completo.

Java Card es la versión de Java enfocada a aplicaciones que se ejecutan en tarjetas de crédito con chip.

Es una versión muy recortada de Java. Un Java Card es una tarjeta capaz de ejecutar mini-aplicaciones Java. En este tipo de tarjetas el sistema operativo es una pequeña máquina virtual Java (JVM) y en ellas se pueden cargar dinámicamente aplicaciones desarrolladas específicamente para este entorno.

Existen diferentes versiones de Java que han ido apareciendo a lo largo del tiempo y dónde el entorno de desarrollo ha ido evolucionando, incluyendo mejoras en cada uno de las plataformas existentes.

Jakarta EE

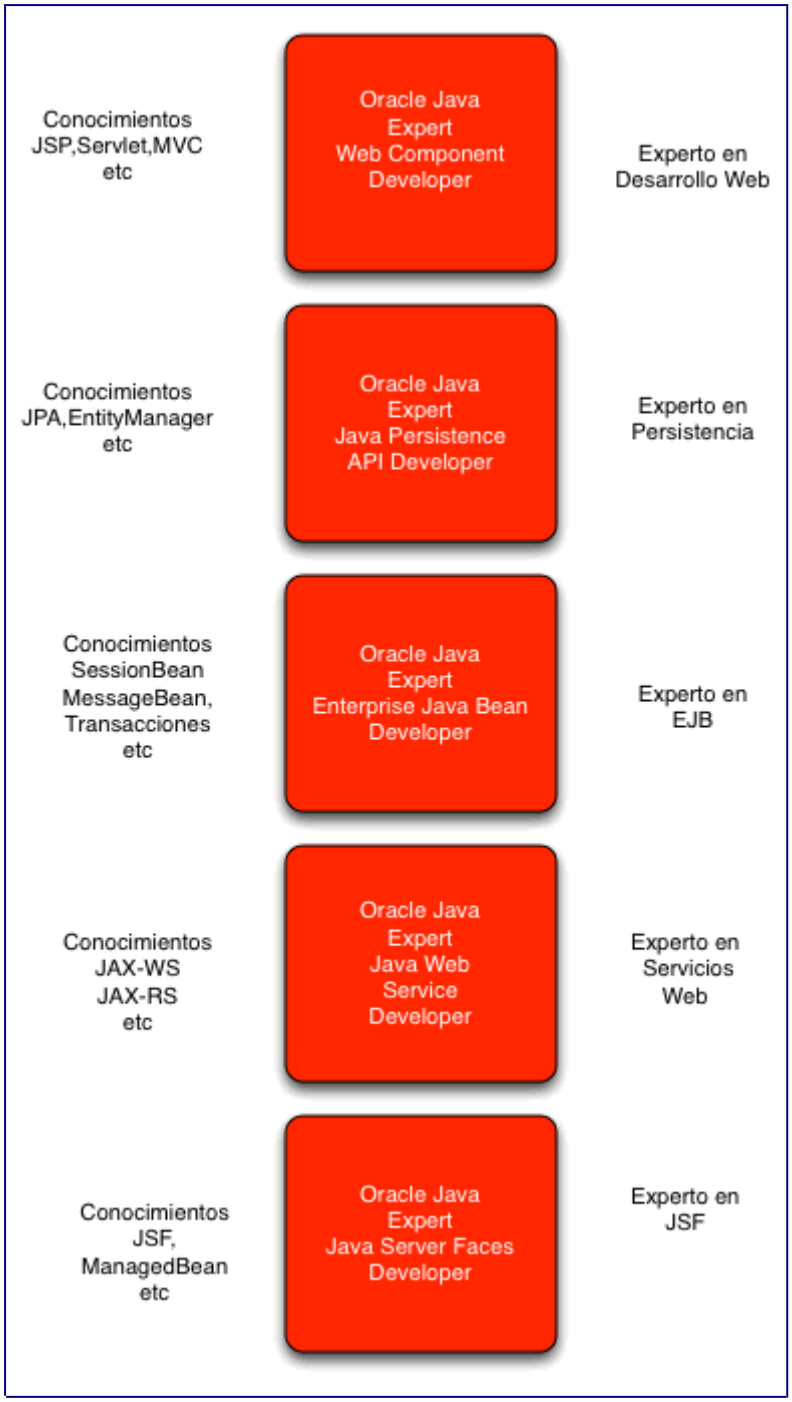
En 2017 Oracle decidió dejar de controlar el desarrollo de Java EE y empezarían por buscar una organización open source a la que pasar la batuta. Esa organización terminó siendo la Eclipse Foundation, la misma comunidad detrás de la plataforma Eclipse que consiste en el IDE y el ambiente de desarrollo.

Oracle dijo que la razón de esto era mejorar el proceso de desarrollo de Java EE que a pesar de ya desarrollarse de forma abierta con la ayuda de la comunidad, debería tener un proceso más flexible y abierto. Pero flexible es algo que Oracle no es a la hora de usar la marca Java, y por ello **Java EE ha tenido que cambiar de nombre a Jakarta EE**.

Las pautas de uso de la marca registrada de Oracle no permiten que el nombre empiece por "Java", y una vez que el proceso de migrar Java EE a la Eclipse Foundation comenzó, también empezó un proceso para renombrar la tecnología. Aunque algunos miembros de la comunidad querían que el nombre "java" formase parte de alguna forma, al final se han inclinado por Jakarta.

Examenes de Certificación Oracle Java EE

Existen exámenes oficiales de Oracle para certificar destrezas en la plataforma EE.



Como podemos ver los título de expertos están divididos en cinco:

Web Component Developer : Identifica al experto en desarrollo web que tiene fuerte conocimiento en JSP, Servlets, Modelo MVC, Deployment descriptor etc.

EJB Component Developer: Identifica al experto en desarrollo de EJBs y tiene un fuerte conocimiento sobre el ciclo de vida de los EJBs, sus tipologías así como aspectos complementarios como gestión de transaccionalidad y seguridad.

Persistence API Developer : Identifica al experto en capa de persistencia capaz de desarrollar modelos de dominio complejos , usar JPA Query Language y gestionar las transacciones.

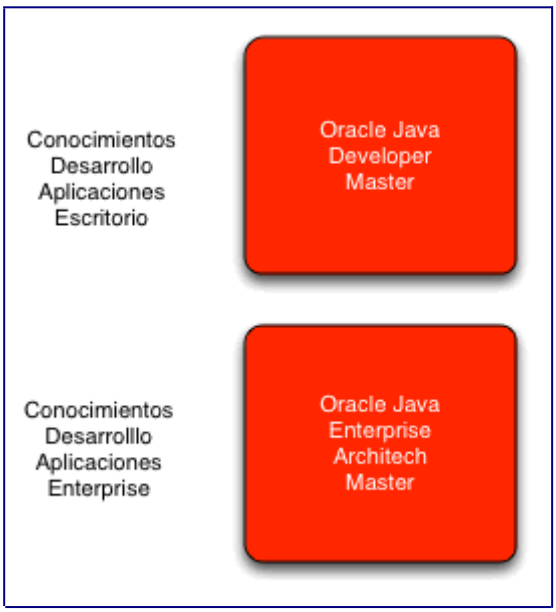
Web Service Developer: Identifica al experto en construcción de servicios web tanto SOAP como REST. Permitiendo una mejor integración de nuestras aplicaciones con aplicaciones externas.

Java Server Faces Developer: Identifica al experto en construcción de capa de presentación a partir de componentes . Maneja el ciclo de vida y los patrones de diseño complejos que esta capa soporta.

Estas cinco certificaciones cada una de ellas es un examen tipo test bastante complejo.

Certificaciones Master

Apartadas del resto de certificaciones están las certificaciones Master. Estas se diferencian del resto en que cada una de ellas se compone de varias partes y **es obligatorio realizar un curso oficial de Oracle antes de abordarlas ademas de tener la certificación de programmer.**



Estas certificaciones **son especiales porque hay que realizar un "trabajo de desarrollo" o tarea a partir de una asignación que te encargan y entregarla.** De esta forma Oracle se asegura de que la persona tiene una experiencia real en desarrollar aplicativos. La tarea o asignación puede tardar en desarrollarse d**e varios meses a un año mas o menos depende del tiempo que tengas disponible.** La certificación de Arquitecto obliga también a realizar un examen previo sobre Patrones de Diseño, UML , EJBs y conceptos generales de Arquitectura antes de poder abordar el trabajo.

Implementaciones de Java EE

WidFly – JBoss Application Server

WidFly inicialmente conocida como JBoss Application Server, es un software de aplicación creado por JBoss, ahora desarrollado por Red Hat. WidFly esta escrito en Java y se puede ejecutar en las mejoras plataformas Java EE, se encuentra disponible en multi plataformas.

En terminos de API, WidFly soporta el último estándar en Java Enterprise (Java EE 8) y esta basado en los mejores proyectos de software abierto tales como:

- Undertow (servidor web)
- Apache CXF (web services basado en SOAP)
- RestEasy (web services basado en REST)
- Infinispan (plataforma de Data Grid y Cluster)
- HornetQ (proveedor de JMS)
- JGroups (framework de comunicación Multicast)

En términos de arquitectura interna, WidFly (tal cual JBoss AS 7) esta diseñado en una arquitectura modular basado en dos proyectos principales:

- JBoss Module, maneja la carga de los recursos de clase en el contenedor
- Modular Service Container (MSC), que proporciona una forma de instalar, des-instalar y administrar los servicios utilizados por el contenedor.

WidFly se puede ejecutar de dos formas: con un simple JVM y multi-JVM o modo domain.

GlassFish

GlassFish es un servidor de aplicaciones de código abierto iniciado por Sun Microsystems para la plataforma Java EE y ahora patrocinado por la corporación Oracle.

El servidor GlassFish 4.0 esta desarrollado a través de un proyecto de la comunidad de sofware en http://glassfish.java.net/. El proyecto GlassFish proporciona un proyecto estructurado para el desarrollo el cual permite tener disponibles las nuevas características de la plataforma Java EE en forma más rápido que en otros servidores.

GlassFish esta construido en un núcleo modular basado en OSGi, GlassFish se ejecuta sobre la aplicación de Apache Felix. También se ejecuta con Equinox OSGi o Knopflerfish OSGi.

Apache Geronimo

Apache Geronimo es un servidor de software abierto que integra los mejores proyectos de software abierto creados para Java/OSGi el cual conoce las necesidades de programadores y administradores de sistema. Entonces la característica más importante de este servidor es que contiene una colección de los mejores proyectos de software abierto para hacer un servidor complementa funcional. Los componentes del núcleo de Jeronimo incluyen:

- Apache Tomcat (Servidor HTTP y contenedor de servlet)
- Jetty (Servidor HTTP y contenedor de servlet, alternativo a Tomcat)
- Apache ActiveMQ (proveedor de JMS 1.1)
- Apache OpenEJB (contenedor EJB)
- Apache OpenJPA (Implementación de Java Persistence API – JPA 1.0)
- Apache ServiceMix (Enterprise Services Bus)
- Apache CXF (framework de web services con una variedad de protocolos)
- Apache Derby (administrador de base de datos relacional con un soporte nativo para JDBC)
- Apache WADI (solución cluster y de balanceo de carga)

El núcleo de Geronimo es un framework para los servicios del núcleo y control de los componentes básicos del servidor. Todo en Geronimo es un GBean: contenedor, conector, adaptador, aplicación y más. Un GBean es la unidad mínima de Geronimo, el cual consiste en un conjunto de clases (.jar) y el plan de implementación.

JOnAS

JOnAs es una implementación de software libre de la especificación de JEE para servidores de aplicación, desarrollada y alojada el consorcio ObjectWeb.

Las características de JOnAS lo certifican como un contenedor Java EE 6. Proporciona un contenedor de EJB a través de EasyBeans y esta disponible con un servidor Tomcat y un servidor Jetty. JVM 6 esta soportado y los intentos de ejecutar en una pila libre con el classpath GNU son prometedores. Además de la pila de Java EE, JOnAS proporciona mecanismos de alto nivel para cluster y así proporcionar escalabilidad y alta disponibilidad.

Apache TomEE

Apache TomEE es la edición Java EE de Apache Tomcat, el cual combina varios proyectos Java incluyendo Apache OpenEJB, ApacheOpenWebBeans, Apache OpenJPA, ApacheMyFaces y otros. El proyecto tiene una certificación de la corporación Oracle como un servidor compatible del perfil web Java EE 6 , que es una parte de toda la especificación Java EE.

Básicamente ApacheTomEE es ensamblada de vanilla Apache Tomcat.

Algunas otras implementaciones o "Application Servers" **"Fully J2EE Compliant"** son:

- WebLogic
- Websphere
- Oracle 9i Application Server
- Sun Application Server (Previamente *Netscape Enterprise* o Kiva)
- Jetty
- Jenkins
- Kassandra