**Versiones para “Herramientas”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 27/05/15 | 1.0 | El documento describe las herramientas que se utilizaran para la elaboración del software. | Hernández Chávez Celia |

**Herramientas**

**PostgreSQL**

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

**Componentes más importantes en un sistema PostgreSQL**

* **Aplicación cliente**: Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP o sockets locales.
* **Demonio postmaster**: Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.
* **Ficheros de configuración**: Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg\_hba.conf y pg\_ident.conf
* **Procesos hijos postgres**: Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
* **PostgreSQL share buffer cache**: Memoria compartida usada por PostgreSQL para almacenar datos en caché.
* **Write-Ahead Log (WAL)**: Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO)
* **Kernel disk buffer cache**: Caché de disco del sistema operativo
* **Disco**: Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

**Características**

La última serie de producción es la 9.3. Sus características técnicas la hacen una de las bases de datos más potentes y robustas del mercado. Su desarrollo comenzó hace más de 16 años y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

**Características Generales**

* Es una base de datos 100%
* Integridad referencial
* Tablespaces
* Nested transactions (savepoints)
* Replicación asincrónica/sincrónica / Streaming replication - Hot Standby
* Two-phase commit
* PITR - point in time recovery
* Copias de seguridad en caliente (Online/hot backups)
* Unicode
* Juegos de caracteres internacionales
* Regionalización por columna
* Multi-Version Concurrency Control (MVCC)
* Multiples métodos de autentificación
* Acceso encriptado via SSL
* Actualización in-situ integrada (pg\_upgrade)
* SE-postgres
* Completa documentación
* Licencia BSD
* Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.

**Desarrollo**

* Funciones/procedimientos almacenados (stored procedures) en numerosos lenguajes de programación, entre otros PL/pgSQL (similar al PL/SQL de Oracle), PL/Perl, PL/Python y PL/Tcl
* Bloques anónimos de código de procedimientos (sentencias DO)
* Numerosos tipos de datos y posibilidad de definir nuevos tipos. Además de los tipos estándares en cualquier base de datos, tenemos disponibles, entre otros, tipos geométricos, de direcciones de red, de cadenas binarias, UUID, XML, matrices, etcétera.
* Soporta el almacenamiento de objetos binarios grandes (gráficos, videos, sonido).
* APIs para programar en C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, PHP, Lisp, Scheme, Qt y muchos otros.

**SQL**

* SQL92,SQL99,SQL2003,SQL2008
* Llaves primarias (primary keys) y foráneas (foreign keys)
* Check, Unique y Not null constraints
* Restricciones de unicidad postergables (deferrable constraints)
* Columnas auto-incrementales
* Indices compuestos, únicos, parciales y funcionales en cualquiera de los metodos de almacenamiento disponibles, B-tree, R-tree, hash ó GiST
* Sub-selects
* Consultas recursivas
* Funciones 'Windows'
* Joins
* Vistas (views)
* Disparadores (triggers) comunes, por columna, condicionales.
* Reglas (Rules)
* Herencia de tablas (Inheritance)
* Eventos LISTEN/NOTIFY

**JAVA**

Java es uno de los lenguajes más difíciles de aprender, quizás no tanto por su sintaxis o por el hecho de que está orientado a objetos, sino más bien por su amplitud. Dentro de Java hay una biblioteca de clases (librerías) para hacer cualquier tipo de cosa. Es muy difícil que se desarrolle cualquier tipo de aplicación y comience desde cero en Java, al contrario, siempre se apoyará en algo ya escrito, que implementa las bases de aquello que quiere construir.

Java es muy grande y aunque represente una dificultad en el aprendizaje, es también una de sus ventajas. De hecho, se podría decir que, aunque un desarrollador tenga muchos años de experiencia programando, es prácticamente imposible que conozca todas las áreas para las que Java se puede utilizar, ya que son muchas y muy diferentes. Normalmente, las clases (librerías) que se utilizan cuando se realiza un tipo de programa son diferentes a las que se utilizan para otro programa y lo habitual es que los proyectos solo requieran trabajar con un pequeño conjunto de los recursos que existen en Java.

## **Java Virtual Machine (JVM)**

Java es un lenguaje multiplataforma, que se ejecuta en cualquier máquina. Esto es gracias a la JVM (Java Virtual Machine) que nos permite ejecutar el código de Java en cualquier lugar para el que se haya creado dicha máquina virtual. JVM es el secreto (no tan secreto porque todo el mundo lo sabe) y la clave de Java como lenguaje multiplataforma.

La JVM hace de puente entre el código compilado de Java y la máquina donde se pretende ejecutar. Podemos verlo como un traductor, que se encarga de interpretar el código de Java, de manera que sea correctamente ejecutado en el sistema objetivo. Es decir, con Java compilamos el código y el resultado de la compilación se ejecuta en la JVM que tenga el sistema operativo donde quieres poner en marcha el programa.

Casi en cualquier hardware se puede montar un JVM y gracias a ello ser compatible con Java. De hecho, los primeros teléfonos móviles, antes de la llegada de los Smartphone, tenían la posibilidad de ejecutar juegos en Java y eso es porque simplemente se había creado una JVM para ellos.

## **Qué se necesita para programar en Java**

**1) Java JDK**   
Lo primero que se necesita para poder desarrollar en Java es el "Java Development Kit". Es un software gratuito que contiene todo aquello que requiere tu máquina para trabajar con el lenguaje, tanto la JVM como las librerías para realizar programas de Java, desde los más básicos hasta los más complejos y específicos.

Para obtener el JDK necesitas descargarlo desde la página de Oracle.

Hay diferentes "entregas", "sabores" o "paquetes", de Java. Cada una tiene un nombre y a veces resulta un poco lioso por tener tantas siglas que aprenderse.

Cuando se descarga el JDK, en realidad se está descargando Java por completo, incluyendo todas las librerías y también por supuesto el Java VM (JVM). Si te descargas el JRE, en realidad estás trayéndote un subconjunto de Java. Así pues, simplemente tienes que fijarte qué es lo que estás descargando para traerte aquello que necesites.

**2) JVM**   
Si tienes o no Java realmente no debemos preocuparnos, la JMV se instala también cuando se instala el JDK para Java.

**3) IDE para desarrollo**   
El tercer paso sería contar con un IDE para el desarrollo. IDE son las siglas de "Integrated Development Environment" o en español, Entorno de Desarrollo Integrado.

## **Alternativas de IDE para Java**

Sobre IDE tenemos varias opciones para Java, entre las más conocidas y usadas tenemos Eclipse, Netbeans, Jdeveloper, IntelliJ IDEA. Cualquiera de ellos puedes usarlo perfectamente, así como cualquier otro que encuentres y te sientas cómodo.

El IDE Eclipse es con diferencia el más usado (tiene en torno al 60% o 70% de tasa de uso entre los desarrolladores de Java) porque es el primero que se popularizó, gracias a que fue el que primero introdujo las extensiones por medio de plugins. Java es atractivo también porque es abierto y gratuito para cualquier tipo de uso.

## **Eclipse para programar en Java**

Eclipse es multiplataforma, por lo que también tendrás que asegurarte de estar descargando exactamente la versión para tu sistema operativo, Linux, Mac o si estás en Windows, la versión de 32 Bit o 64 Bit que corresponda con el tipo de procesador de tu PC.

Eclipse tiene una particularidad y es que no requiere instalación, al contrario de lo que estamos acostumbrados en muchos sistemas. Realmente lo que descargas es el programa entero, ya "paquetizado" para que puedas usarlo. Simplemente tienes que descomprimir el contenido de tu fichero y dejarlo en cualquier localización de tu disco duro, el escritorio, la raíz del disco, un disco externo o donde desees. Una vez descomprimido, simplemente tendrás que hacer un doble clic en el ejecutable (eclipse.exe) para poner en marca el IDE.

Como puedes ver, Eclipse se instala como lo que podríamos decir una versión "portable" del software, porque todo lo que necesita el programa para funcionar está en la carpeta que has descomprimido. Si te llevas la carpeta a cualquier otra localización, seguirá funcionando de la misma manera. Incluso si tienes que cambiar de ordenador, simplemente te llevas la carpeta a la nueva máquina y funcionará perfectamente, junto con todos tus proyectos.

Otra de las ventajas que nos proporciona el estilo de instalación de Eclipse, sin el típico asistente, es que puedes tener varias distribuciones de Eclipse, configuradas de distinta manera, para varios tipos de lenguajes o varios proyectos, sin que se peguen la una con la otra.

**SCENE BUILDER**

Hay dos formas de crear la interfaz de usuario. Programándolo en Java o mediante un archivo XML. Aunque existe información relativa a ambos métodos. Si usamos XML (archivo con la extensión .fxml) puede que se encuentre más claro mantener el controlador y la vista separados entre sí.

Además, se puede usar la herramienta de edición visual Scene Builder, la cual nos evita tener que trabajar directamente con el XML.

Una nueva herramienta para diseñar y crear contenidos en JavaFX, que básicamente es un editor de archivos FXML.

Es el comienzo de una herramienta más completa de tipo RAD (desarrollo rápido de aplicaciones) para JavaFX, con capacidades de construir un GUI mediante arrastrar y soltar y eventualmente conexión de datos mediante el diseño en Scene Builder.

**ESPECIFICACIONES DEL EQUIPO DE COMPUTO DONDE SE EJECUTARA LA APLICACIÓN**

La aplicación se podrá ejecutar en algún equipo que cuente con el sistema operativo Windows 7 u Windows 8 y el equipo debe contar mínimo con:

* RAM: 128 MB
* Espacio en disco: 124 MB para JRE; 2 MB para Java Update
* Procesador: Mínimo Pentium 2 a 266 MHz
* Exploradores: Firefox, Chrome

Estas especificaciones del equipo de cómputo servirán para el buen funcionamiento de la aplicación y para mantener la eficiencia del software que se pretende instalar.