Contents

Guía 1: Instalar JDK

Paso 1: Descargar el JDK

De la siguiente URL descargar el instalador para su sistema operativo

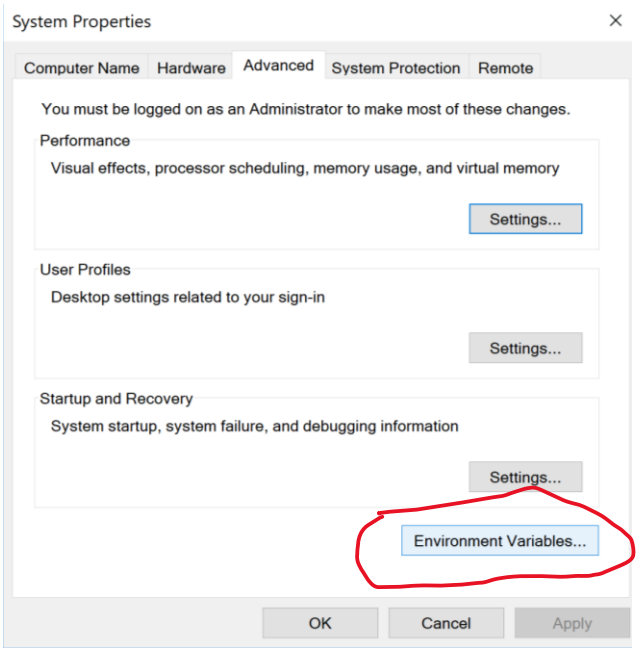
http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html?ssSourceSiteId=otnes

- *En caso de utilizar Linux, basado en Debian/ Ubuntu, puede instalarlo con apt-get siguiendo las siguientes instrucciones http://www.webupd8.org/2012/09/install-oracle-java-8-in-ubuntu-via-ppa.html*

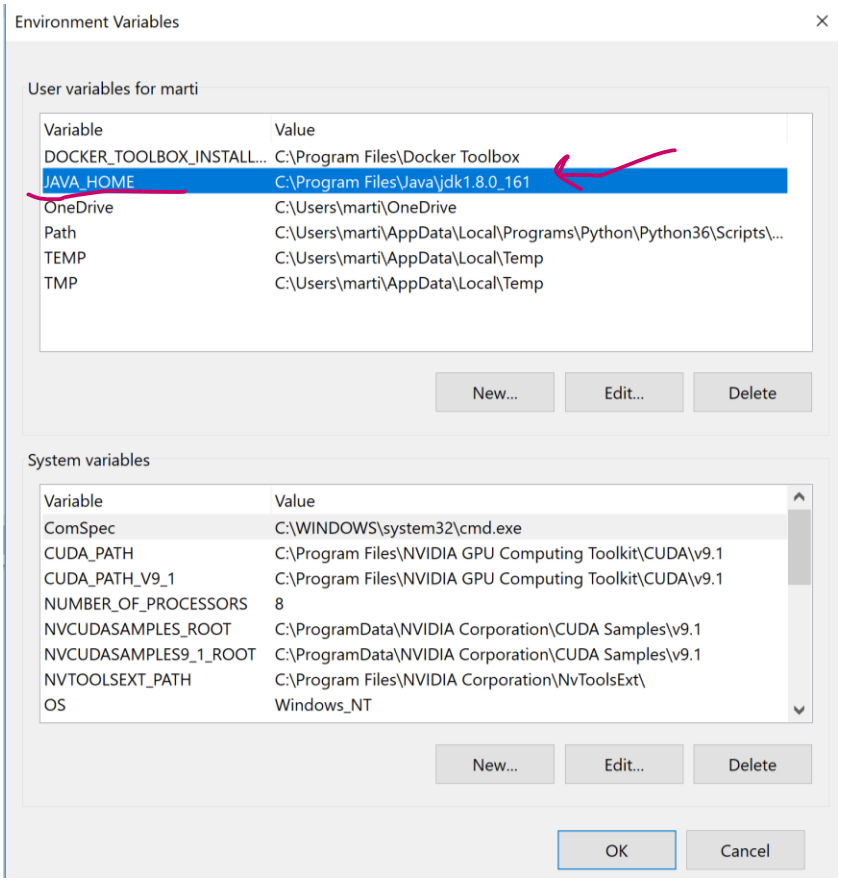
Paso 2: Configurar las variables de entorno

Luego de instalado Java, crearemos la variable de entorno del sistema operativo, JAVA_HOME, y también agregaremos al PATH los comandos del directorio jdk1.8.0_161\bin (entre los que se encuentra el compliador "javac", el interprete "java" o el generador de documentación "javadoc"

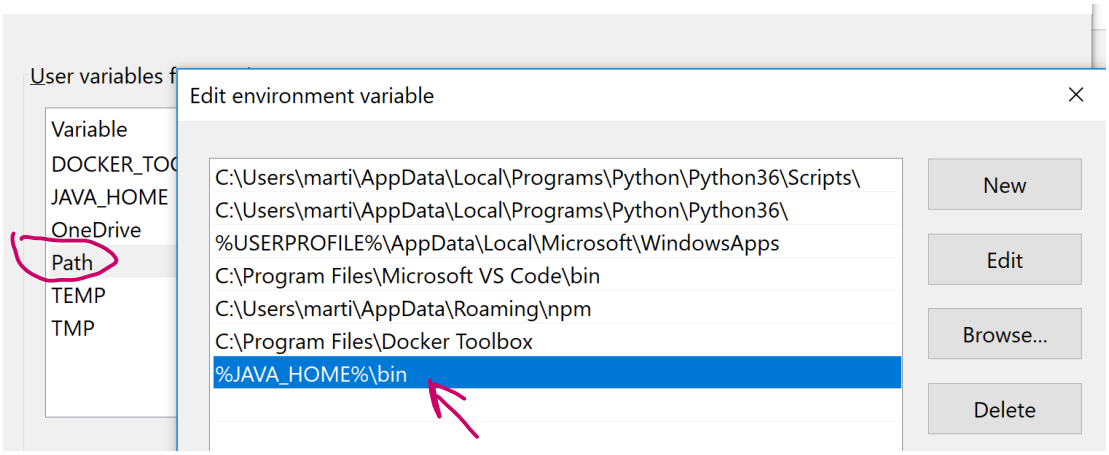
En Windows, seleccionar las propiedades del sistema y luego las variables de entorno



Crear la variable JAVA_HOME y asignarle como valor el directorio de instalación del JDK, típicamente es c:\Program Files\Java\jdk1.8.0_161 (donde 1.8.0_161 corresponde a la versión que tiene instalada, puede variar según la versión disponible)



Luego modificar la variable PATH y agregarle el valor %JAVA_HOME%\bin



Con este comando, estamos indicando al sistema operativo, que cuando alguien solicite el comando java, javac, o javadoc, lo busque en el directorio indicado. De esta manera podremos ejecutar estos comandos en cualquier ruta del sistema operativo.

Abrir una consola y ejecutar los siguientes comandos para verificar que la instalación fue correcta.

- java -version
- javac - version

Ejemplo:

```
C:\Users\marti>java -version
java version "1.8.0_161"
Java(TM) SE Runtime Environment (build 1.8.0_161-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.161-b12, mixed mode)

C:\Users\marti>javac -version
javac 1.8.0_161
```

Paso 3: Crear un programa, compilarlo y ejecutarlo.

En este paso, en un directorio arbitrario del sistema operativo crearemos una clase, la compilaremos, la ejecutaremos y generaremos la documentación.

Crear un directorio, por ejemplo “died-ejemplo01”

Dentro de dicho directorio crear un subdirectorio “app” , y dentro de este subdirecotrio un archivo **Hola.java** con el siguiente código de una clase Java

```java
package app;

/**
  Clase de prueba de instalacion
  @author martin
  @version 1.0.0
 */
public class Hola {

    /**
     * Metodo que ejecuta la aplicación. Si recibe un argumento o
     * más los imprime.
     * Si no recibe argumentos indica ese mensaje en consola
```

```
     */
    public static void main(String[] args){
        if(args.length==0) {
                System.out.println("Se ejecuto el programa sin argumentos");
        }else{
            for(String argumento : args){
                System.out.println("argumentos recibidos: "+ argumento);
            }
        }
    }
}
```

En este momento tenemos creada la siguiente estructura de archivos

```
C:\Users\marti\Documents\devs>cd died-ejemplo01

C:\Users\marti\Documents\devs\died-ejemplo01>dir
 Volume in drive C is OS
 Volume Serial Number is A651-73AB

 Directory of C:\Users\marti\Documents\devs\died-ejemplo01

04-Apr-18  10:29 PM    <DIR>          .
04-Apr-18  10:29 PM    <DIR>          ..
04-Apr-18  10:29 PM    <DIR>          app
               0 File(s)              0 bytes
               3 Dir(s)  376,687,534,080 bytes free

C:\Users\marti\Documents\devs\died-ejemplo01>dir app
 Volume in drive C is OS
 Volume Serial Number is A651-73AB

 Directory of C:\Users\marti\Documents\devs\died-ejemplo01\app

04-Apr-18  10:29 PM    <DIR>          .
04-Apr-18  10:29 PM    <DIR>          ..
04-Apr-18  10:29 PM               587 Hola.java
               1 File(s)            587 bytes
               2 Dir(s)  376,687,525,888 bytes free

C:\Users\marti\Documents\devs\died-ejemplo01>
```

Luego en una ventana de comando posicionarse en el directorio “died-ejemplo01” y compilar la clase java, con el comando “javac app\Hola.java”.

Luego ejecutar la clase con el comando “java app.Hola” y verificar que imprime que no se recibieron parámetros. Posteriormente ejecutarla con más de un parámetro y verificar que los imprime.

Finalmente generar la documentación con el comando “javadoc”. El comando javadoc, recibe dos parámentros

- d <directorio destino de la documentación>
- <nombre del paquete>

En nuestro caso el directorio destino será “html” y el nombre del paquete es “app”

C:\WINDOWS\system32\cmd.exe

```
C:\Users\marti\Documents\devs\died-ejemplo01>javac app\Hola.java

C:\Users\marti\Documents\devs\died-ejemplo01>java app.Hola
Se ejecuto el programa sin argumentos

C:\Users\marti\Documents\devs\died-ejemplo01>java app.Hola argumento1 1234 abcd
argumentos recibidos: argumento1
argumentos recibidos: 1234
argumentos recibidos: abcd

C:\Users\marti\Documents\devs\died-ejemplo01>javadoc -d html app
Loading source files for package app...
Constructing Javadoc information...
Creating destination directory: "html\"
Standard Doclet version 1.8.0_161
Building tree for all the packages and classes...
Generating html\app\Hola.html...
.\app\Hola.java:14: warning: no @param for args
    public static void main(String[] args){
                       ^
Generating html\app\package-frame.html...
Generating html\app\package-summary.html...
Generating html\app\package-tree.html...
Generating html\constant-values.html...
Building index for all the packages and classes...
Generating html\overview-tree.html...
Generating html\index-all.html...
Generating html\deprecated-list.html...
Building index for all classes...
Generating html\allclasses-frame.html...
Generating html\allclasses-noframe.html...
Generating html\index.html...
Generating html\help-doc.html...
1 warning
```

Abrir el archivo html\index.html y verfiicar que se ve el contenido del javadoc como se muestra a continuación