

Ayudas y Sugerencias.

Elo329

Fecha: 08-04-09

Revisión 14-04-10

Versión del Documento: 1.1

1.Introducción

El presente documento consiste en un conjunto de instrucciones y sugerencias para hacer más fácil el desarrollo de las tareas del ramo.

No es necesario que lea todo este documento. Si quiere puede revisar solo aquellos puntos donde tenga dudas o que le interesen.

Nota: en el presente documento se utiliza '\$' para indicar el ingreso de un comando por consola, no debe incluirlo dentro del comando.

Puede que usted ya tenga instalado el JRE y/o el JDK, pero, de todas formas puede ser buena idea seguir los pasos siguientes para asegurarse de que tiene la última versión y que está instalada correctamente.

Si ya tiene instalado el JRE, puede determinar su versión con:

```
$ java -version
```

Si ya tiene instalado el JDK, puede determinar su versión con:

```
$ javac -version
```

Y retornará su versión:

```
java version "1.6.0_19"
```

```
Java(TM) SE Runtime Environment (build 1.6.0_19-b04)
```

En el sitio web de Oracle podrá encontrar la última versión.

2. Instalación del JRE (java runtime environment) y del JDK (java development kit).

Hay diversos modos de instalarlos, pero, sugiero instalar la última versión estable de java desde:

<http://java.sun.com/javase/downloads/index.jsp>

Aquí encontrará la última versión del JDK para Windows y Linux.

También está disponible un paquete con Netbeans+JDK que le podría ser de gran utilidad para el desarrollo de las tareas, ya que ese le permite la manipulación y creación en forma gráfica de interfaces graficas para sus aplicaciones en java.

Después de descargar ejecute el instalador.

Si utiliza linux, el instalador colocará los archivos necesarios en una carpeta (jre6 o jdk1.6.0_13) dentro de la carpeta donde se ejecuta el instalador. Esto puede no ser lo más conveniente; tal vez desee colocar esta carpeta en un lugar donde otros usuarios la encuentren y puedan tener acceso (por ejemplo, en /usr/local/java, donde muchos instaladores suelen colocarla). Recuerde que puede necesitar privilegios de administrador.

Otro método en Linux, el más simple de todos, es a través de algún sistema gestor de paquetes. Consulte la documentación de su distribución para obtener más detalles.

3. Verificación de que el PATH esté seteado.

Abra una consola en windows o linux y ejecute el comando:

```
$javac -version
```

Debería indicarle que la versión. Si la versión no es la última o no encuentra el comando, con seguridad no tiene la ruta a las carpetas instaladas dentro de la variable de entorno PATH.

En Linux puede revisar su PATH actual con el comando "echo":

```
> echo $PATH
```

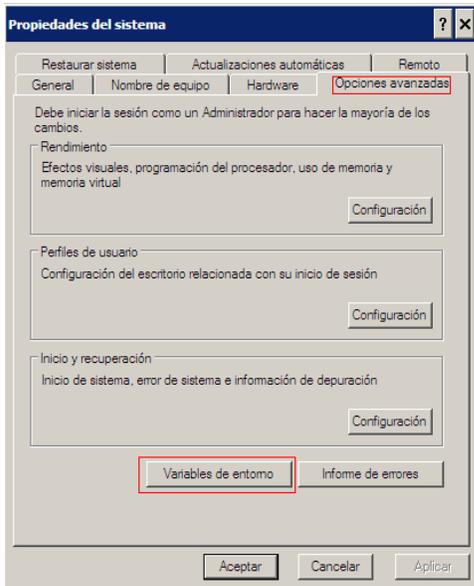
3.1. Como setear la variable PATH.

3.1.1. En Windows.

Para hacerlo de forma gráfica, haga click derecho sobre "Mi PC" y vaya al submenu "Propiedades".

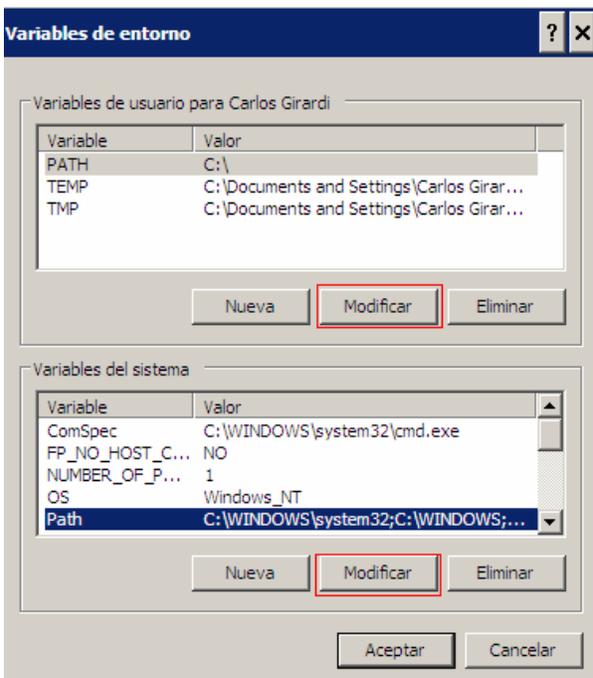


Vaya a la pestaña "Opciones avanzadas" y apriete el botón "Variables de Entorno".



Podrá ver que hay una variable PATH que es inherente al entorno del usuario (solo válida para el usuario actual) y otra al sistema (válida para todos los usuarios). Es más cómodo si hace las modificaciones para el sistema, pero requiere permisos de administrador.

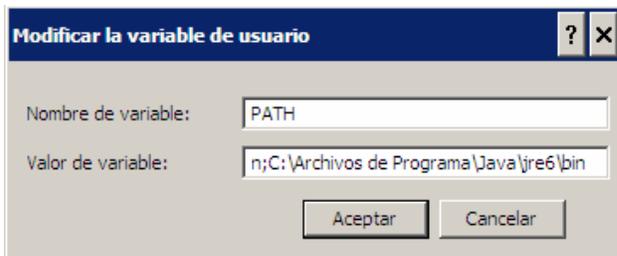
Seleccione la variable PATH (en entorno de usuario o en entorno de sistema según corresponda) y presione en el botón "Modificar".



Agregue al valor de la variable la ruta a ambas carpetas "bin", dentro de la carpeta del JRE y del JDK, separe cada ruta con ';'. Si ha usado los instaladores

sugeridos, las rutas posiblemente sean:

";C:\Archivos de Programa\Java\jdk1.6.0_13\bin;C:\Archivos de Programa\Java\jre6\bin" o similar dependiendo de la carpeta donde instaló Java.



IMPORTANTE: ¡Borre cualquier ruta a una versión anterior!

3.1.2. En Linux:

MÉTODO #1

Se puede hacer a través la asignación dada en "/etc/profile". Este método cambia PATH para todos los usuarios, aunque en algunas distribuciones no lo hace para el usuario root.

Al inicio de /etc/profile se hace el seteo de PATH. Agregue las rutas a las carpetas "bin" de las carpetas del JRE y del JDK instalado. Separe cada ruta por ':'. Si ha instalado manualmente los paquetes a través el sitio web de Oracle, los paths correspondientes serán similares a:

"/usr/local/java/jdk1.6.0_13/bin:/usr/local/java/jre6/bin".

Borre las rutas a versiones anteriores.

Es posible que deba resetear el computador para que funcione.

MÉTODO #2

Un método que no necesita de privilegios de administrador:

En su archivo /home/usuario/.bashrc (Para sistemas con interpretador de comandos BASH, por ejemplo Ubuntu) agregue los directorios correspondientes al final de la variable PATH. Así estos cambios serán permanentes para ese usuario.

En resumen, para Ubuntu:

```
$gedit /home/usuario/.bashrc
```

Agregue:

```
PATH=$PATH:/directorio_donde_esta_java/bin
export PATH
```

Guarde y cierre. Finalmente:

```
$source ~/.bashrc
```

Compruebe con:

```
$echo $PATH
```

4. Recordatorio sobre Compilación.

Inicie una consola en linux o windows, navegue hasta la carpeta donde tiene los archivos fuente y ejecute el comando:

```
$javac <nombre_codigo_fuente>.java
```

Recuerde que debe incluir la extensión ".java" y que puede incluir más de un archivo fuente separando por un espacio cada nombre.

Puede ejecutar:

```
$javac -help
```

Para obtener un listado de opciones.

5. Recordatorio de invocación.

Inicie una consola en windows o linux, navegue hasta la carpeta donde está la clase o hasta la carpeta que contiene el paquete (más detalles más adelante).

Ejecute:

```
$java [opciones] <nombre-clase> [argumentos del programa]
```

Si tenemos una clase, por ejemplo, dentro de un paquete personas, que a la vez está en el paquete empresa, navegaríamos hasta la carpeta que contiene "empresa" y ejecutaríamos:

```
$java empresa.personas.<nombre-de-clase>
```

5.1. Ejecución de JAR.

5.1.1. Por consola.

Debe incluir la opción "-jar" dentro del comando de invocación del programa.

5.1.2. Gráficamente.

Algunos JAR tienen GUI y permiten ser ejecutados si hacemos doble click sobre estos en un entorno gráfico. Es necesario que la extensión ".jar" esté vinculada al ejecutable de la JVM.

6. Javadoc.

6.1. Como escribir la documentación de java.

El código para generar la documentación de java para un método puede lucir como el siguiente ejemplo:

```
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute {@link URL}. The name
 * argument is a specifier that is relative to the url argument.
 * <p>
 * This method always returns immediately, whether or not the
 * image exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives
 * that draw the image will incrementally paint on the screen.
 *
 * @param url an absolute URL giving the base location of the image
 * @param name the location of the image, relative to the url argument
 * @return the image at the specified URL
 * @see Image
 */
public Image getImage(URL url, String name) {
    try {
        return getImage(new URL(url, name));
    } catch (MalformedURLException e) {
        return null;
    }
}
```

Una vez compilado con javadoc, el resultado es algo así:

getImage

```
public Image getImage(URL url,
String name)
```

Returns an Image object that can then be painted on the screen. The url argument must specify an absolute [URL](#). The name argument is a specifier that is relative to the url argument.

This method always returns immediately, whether or not the image exists. When this applet attempts to draw the image on the screen, the data will be loaded. The graphics primitives that draw the image will incrementally paint on the screen.

Parameters:

url - an absolute URL giving the base location of the image
name - the location of the image, relative to the url argument

Returns:

the image at the specified URL

See Also:

[Image](#)

Como puede notarse, es necesario que todo el cuerpo para la descripción de la documentación este encerrado entre “/**” (nótese que deben haber dos asteriscos) y “*”. Además, cada línea intermedia debe partir con un asterisco.

Las líneas de texto iniciales (que no están precedidas por alguna marca descriptiva que parte con ‘@’, llamadas “blocktags”) corresponden a la descripción de lo que hace el método.

El ejemplo muestra dos de los blocktags más importantes: @param y @return. El primero indica la descripción de un argumento del método y tiene la sintaxis: @param <nombre_del_argumento> <descripción_del_argumento>.

Return indica que es lo que devuelve el método y tiene sintaxis:

```
@return <descripción_de_lo_retornado>
```

Otros blocktags que existen son:

```
* @param (para clases, interfaces, métodos y constructores)
* @return (solo métodos)
* @exception (o @throws en javadoc 1.2)
* @author (debe aparecer en clases e interfaces, exclusivamente)
* @version (debe aparecer en clases e interfaces, exclusivamente)
* @see
* @since
* @serial (o @serialField o @serialData)
* @deprecated
```

Y, de incluirse, deben ir en ese orden.

Para más información remitase a:

<http://java.sun.com/j2se/javadoc/writingdoccomments/>

Algunas personas no documentan aquellos elementos privados de las clases, ya que estos no aparecen en la documentación generada por Javadoc. Sin embargo, se sugiere fuertemente que se documenten incluso estos elementos privados, ya que esto ayuda al entendimiento y mantención del código y además, no es raro que un método pueda ir cambiando su declaración (de privado a público, de público a privado, de privado a protegido, etc.) en el transcurso del desarrollo de una clase.

No solo se pueden documentar métodos.

6.2. Uso de Javadoc para compilar documentación.

Una vez codificada la documentación de java, debe compilarla ejecutando en una consola.

```
$ javadoc <nombre_de_a_clase>
```

Al igual que javac, se compilan las documentaciones de aquellas clases de las cuales depende la clase ingresada en el argumento.

8. Makefile

El makefile es un archivo de texto plano, llamado por omisión Makefile, se ejecuta para compilar el código fuente, eliminar binarios (clear) o revisar dependencias. Se ejecuta través del comando make.

\$make

Ya que tiene seteado el PATH correctamente y entiende el comportamiento de javac y javadoc puede escribir su propio archivo makefile.

Por ejemplo, si decide que los .java irán dentro de la carpeta source podría usar a modo de ejemplo (A este makefile le falta un "clear" para eliminar carpetas y archivos)

```
##### Makefile #####

MAIN = simulador.java
OUTDIR = exe
INDIR = source
CPATH = source
DOC = docs
JAVACPARAM = -d $(OUTDIR) -classpath $(CPATH)
JAVADOCPARAM = -d $(DOC)

#####
#   Compilar. Crear Clases y Documentacion
#####

build:
    @if [ -d $(OUTDIR) ]; then echo; else mkdir $(OUTDIR); fi
    @if [ -d $(DOC) ]; then echo; else mkdir $(DOC); fi
    @javac $(JAVACPARAM) $(INDIR)/$(MAIN)
    @javadoc $(JAVADOCPARAM) $(INDIR)/*.java

hablame:
    @echo
    @echo "hola mundo"
    @echo

#####
```

Como verá, es posible utilizar comandos Shell dentro del makefile, como también podría ejecutar binarios en el.

IMPORTANTE: El makefile deberá funcionar en Aragorn

Nota: Para que este makefile hable, \$make hablame

Nota 2: el signo "@" ejecuta el comando sin este ser mostrado por pantalla.

8. Cómo trabajar remotamente en aragorn.

Para trabajar remotamente en aragorn, deben utilizarse programas de conexión SSH / SFTP. Dentro de los más recomendados están:

-Putty

<http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe>

- SSH Secure Shell Client.

<http://www.elo.utfsm.cl/~install/index.php?dir=Conectividad/Ssh/SSHClient/&file=SSHWin-3.2.0.exe>

- WinSCP

<http://winscp.net/download/winscp418setup.exe>

Si utiliza linux, posiblemente no necesite ninguna de estas herramientas. La mayoría de las distribuciones vienen con clientes ssh ya instalados y que se ejecutan desde consola con:

```
$ssh <nombre_usuario>@aragorn.elo.utfsm.cl
```

Si no se encuentra el comando, posiblemente deba instalar un paquete de SSH Client que seguramente estará en el repositorio.

Para subir y descargar archivos desde aragorn, muchas distribuciones incluyen herramientas que permiten ingresar a SFTP desde el explorador de archivos. Por ejemplo, en KDE basta con escribir

```
"sftp://<nombre_usuario>@aragorn.elo.utfsm.cl"
```

en la barra de direcciones de Konqueror (o Dolphin) para ingresar.

No me extenderé en los detalles de cómo utilizar cada programa, ya que son bastante intuitivos, si mencionaré que Putty se utiliza para ingresar a la consola, mientras que SSH Secure Shell Cliente y WinSCP suelen usarse para subir y descargar archivos desde aragorn (aunque también tienen herramientas para utilizar la consola de aragorn mientras están conectados).

Muchos alumnos suelen trabajar desde sus casas e ir subiendo los archivos que van desarrollando a aragorn (mediante WinSCP por ejemplo) para luego compilarlos y probarlos con aragorn (mediante Putty por ejemplo),

Un aspecto que si profundizaremos, será respecto a trabajar gráficamente desde aragorn.

8.1. Cómo trabajar gráficamente en aragorn.

8.1.1. En Linux.

Necesitará el paquete de cliente SSH mencionado en el punto anterior. Vaya a una consola y ejecútelo con la opción `-X` (debe ser en mayúscula). Es decir:

```
$ssh -X <nombre_usuario>@aragorn.elo.utfsm.cl
```

Esto hará un "X11 forwarding", es decir, la información para desplegar las imágenes por pantalla no serán enviadas al servidor X de aragorn, sino al del computador donde está trabajando. Por lo tanto, si tiene un servidor X correctamente

funcionando, las ventanas gráficas que ejecute en aragorn se verán en su computador.

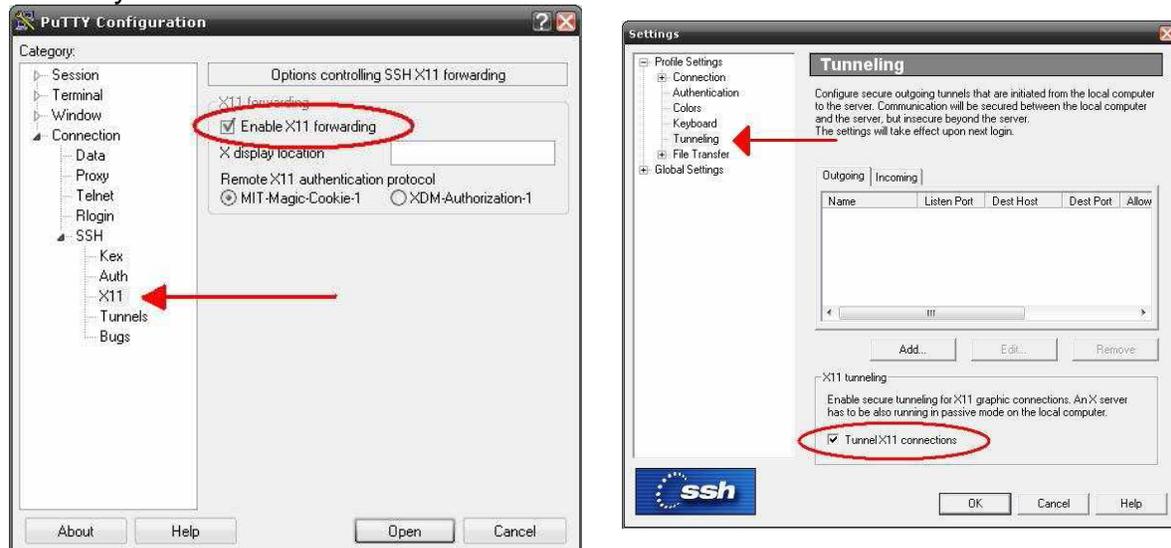
8.1.2. En Windows.

Windows no utiliza un servidor X como el de Linux, por lo que necesita instalar uno. Para esto se sugiere Xming:

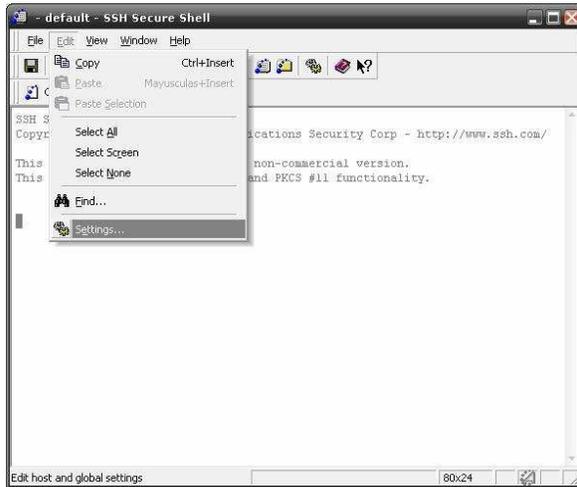
http://sourceforge.net/project/downloading.php?group_id=156984&use_mirror=ufpr&filename=Xming-6-9-0-31-setup.exe

Una vez instalado y funcionando, abra Putty o SSH Secure Shell Client y, antes de conectarse realice la siguiente configuración:

En Putty:



En SSH Secure Shell Client:



Luego conéctese y debería ser capaz de ver desplegadas todas las aplicaciones con GUI que ejecute (recuerde que xming debe estar corriendo).



9. Notas sobre las entregas de las tareas.

Todas las normas y evaluación sobre las tareas están publicadas en <http://profesores.elo.utfsm.cl/~agv/elo329/1s10/>

Además

1) Las tareas se revisan en aragorn y **SOLO** en aragorn. Los alumnos deben compilar y probar su código en aragorn antes de enviarlo.

Si el código no compila o no se ejecuta como debería, no se aceptará la excusa "Es que en mi casa corría bien".

2) Desde las 24:00 horas se considera como un nuevo día. Recuerden que no se les pide que entreguen la tarea el día XX antes de las 24:00, sino que la entreguen ANTES de las 24:00 del día XX. Hacer la tarea con anticipación y no el último día, les traerá beneficios (partiendo por la posibilidad de esclarecer mejor sus dudas).

No se aceptará la excusa de que aragorn estuvo funcionando mal el día de la entrega. Tome en consideración también que si hace una pregunta a última hora, hay posibilidades de que no obtenga una respuesta a tiempo para poder terminar.