Universidad Nacional Autónoma de México

Facultad de Ciencias

Práctica 01:

Compilador

Pablo A. Trinidad Paz

Trabajo presentado como parte del curso de **Introducción a Ciencias de la Computación** impartido por la profesora **Verónica Esther Arriola Ríos**.

24 de agosto de 2018

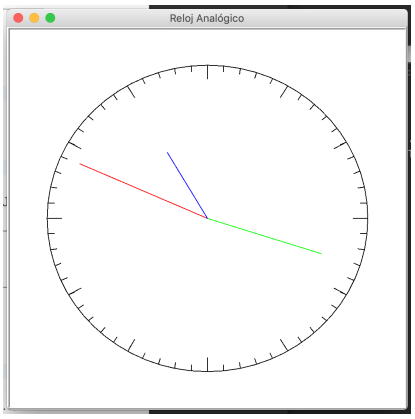
Los programas de la JDK

Actividad 1.2 **Escribe exactamente que archivos fueron creados y donde:**

Dentro de `Reloj/src/icc/practica1` los siguientes archivos fueron creados:

1. `ClaseReloj.class`
2. `PanelDeReloj.class`
3. `Reloj.class`
4. `TiempoSistema.class`
5. `UsoReloj.class`
6. `VistaReloj.class`
7. `VistaRelojAnalogico.class`
8. `VistaRelojAnalogico$1.class`

Actividad 1.3 **Ejecuta** `UsoReloj`



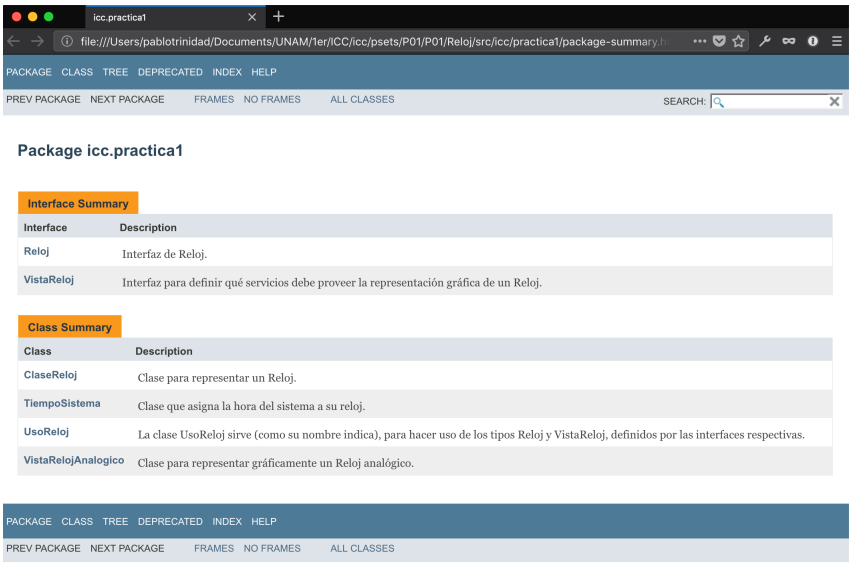
Actividad 1.4 **Intenta invocar a la máquina virtual con los nombres de otros archivos** `.class`**. ¿Qué sucede? Lee lo que devuelve la consola y abre los archivos** `.java` **correspondientes que necesites. ¿Qué tiene el archivo** `UsoReloj.java` **que permite invocar su** `.class` **con java?**

Al tratar de ejecutar otros archivos diferentes a `UsoReloj.class`, Java busca un método principal, `Main method`, que le especifique qué operaciones tiene que ejecutar para iniciar. Lo que tiene diferente el archivo `UsdoReloj.java` que permite que la ejecución de la aplicación suceda es la implementación del método `main` en la línea 16, que además está definido como pú blico, estático y que no regresa ningún valor al terminar la ejecución.

Actividad 1.6 **Generar la documentación de `practica1`. ¿Qué observas?**

Se genearon archivos para servir un sitio web estático como: HTMLs, CSSs y archivos de JavaScript. Al acceder al website de los documentos podemos ver el listado de clases, descripción de cada una de ellas y dentro de cada una sus métodos, atributos, etc.

El website luce así:



Usando una herammienta auxiliar: ant

Las siguientes respuestas corresponden a las actividades de la **Práctica 1** del Manual de Prácticas de Introducción a las Ciencias de la Computación escrito por **Canek Peláez V.** y **Elisa**.

Actividad 1.1 **Al invocar el comando `javac` Anota todas las opciones que se pueden pasar al compilador.**

- `@<filename>`: Read options and filenames from file
- `-Akey[=value]`: Options to pass to annotation processors
- `--add-modules <module>(,<module>)*`: Root modules to resolve in addition to the initial modules, or all modules on the module path if ¡module¿is ALL-MODULE-PATH.
- `--boot-class-path <path>, -bootclasspath <path>`: Override location of bootstrap class files
- `--class-path <path>, -classpath <path>, -cp <path>`: Specify where to find user class files and annotation processors
- `-d <directory>`: Specify where to place generated class files

- -deprecation: Output source locations where deprecated APIs are used
- -encoding <encoding>: Specify character encoding used by source files
- -endorseddirs <dirs>: Override location of endorsed standards path
- -extdirs <dirs>: Override location of installed extensions
- -g: Generate all debugging info
- -g: lines,vars,source: Generate only some debugging info
- -g:none: Generate no debugging info
- -h <directory>: Specify where to place generated native header files
- --help, -help: Print this help message
- --help-extra, -X: Print help on extra options -implicit: none,class: Specify whether or not to generate class files for implicitly referenced
- files
- -J<flag>: Pass ¡flag¿directly to the runtime system
- --limit-modules <module>(,<module>)*: Limit the universe of observable modules
- --module <module-name>, -m <module-name>: Compile only the specified module, check timestamps
- --module-path <path>, -p <path>: Specify where to find application modules --module-source-pat <module-source-path>: Specify where to find input source files for multiple
- modules
- --module-version <version>: Specify version of modules that are being compiled
- -nowarn: Generate no warnings
- -parameters: Generate metadata for reflection on method parameters
- -proc: none,only: Control whether annotation processing and/or compilation is done. -processor <class1>[,<class2>,<class3>...]: Names of the annotation processors to run; bypasses
- default discovery process
- --processor-module-path <path>: Specify a module path where to find annotation processors
- --processor-path <path>, -processorpath <path>: Specify where to find annotation processors
- -profile <profile>: Check that API used is available in the specified profile
- --release <release>: Compile for a specific VM version. Supported targets: 10, 6, 7, 8, 9
- -s <directory>: Specify where to place generated source files
- -source <release>: Provide source compatibility with specified release
- --source-path <path>, -sourcepath <path>: Specify where to find input source files
- --system <jdk>|none: Override location of system modules
- -target <release>: Generate class files for specific VM version
- --upgrade-module-path <path>: Override location of upgradeable modules

- `-verbose`: Output messages about what the compiler is doing
- `--version, -version`: Version information
- `-Werror`: Terminate compilation if warnings occur

Actividad 1.2 **Hola**