Entornos de desarollo

Act.UT02_Instalación y uso de IDE

Francisco José García Cutillas | 1FPGS_DAM

Índice

Ejercicio 1

Realiza un estudio de los principales IDE del mercado (todo debe hacerse con tu vocabulario, nada de copia y pega).

a) Realiza un estudio de las características de cada uno de ellos.

b) IDEs más usados

Entre los principales IDE del mercado podemos encontrar:

1. BlueJ.
    - Se usa únicamente para programación con lenguaje Java.
    - Está exclusivamente diseñado para la enseñanza, aunque también es posible desarrollar con él aplicaciones a pequeña escala.
    - Posee una interfaz en la que se representan gráficamente la estructura de clases de la aplicación en desarrollo.

2. Eclipse.
    - Posee infinidad de complementos, mediante los cuales es posible ampliar la funcionalidad del IDE (nuevos lenguajes, sistema de control de versiones o UML).
    - Permite crear la interfaz de usuario de la aplicación en desarrollo de manera intuitiva, arrastrando y soltando objetos.
    - Tiene una función JavaDoc para crear automáticamente la documentación de nuestra aplicación.
    - Su interfaz de código te permite depurar fácilmente el código gracias a los resaltados de colores.

3. Slickedit.
    - Proporciona un front-end fácil de usar.
    - Se integra con sistemas de control de versiones como Git, Mercurial, Subversion, CVS, Perforce, PVCS y ClearCase.
    - Reduce al máximo las pulsaciones de teclas gracias a que conforme vas escribiendo, el IDE te va sugiriendo lo que sigue.

4. Code:: Blocks.
    - Es de código abierto y multiplataforma.
    - Se le pueden añadir nuevas funcionalidades gracias al amplio catálogo de plugins que posee.
    - No soporta muy bien grandes proyectos.
    - Es compatible con veinte compiladores distintos como GCC, Microsoft Visual C++, Tiny C, Digital Mars entre los más conocidos.

5. Gnat Studio.
    - Es un IDE multiplataforma libre para el lenguaje Ada, aunque también admite C, JavaScript, Pascal y Pyton.
    - Posee características como autocompletar, edición y depuración remota y compilación para plataformas no nativas.

6. AWS Cloud9
    - Se puede acceder a él a través de un navegador web.
    - Se puede colaborar con otros usuarios en tiempo real.
    - Utiliza repositorios de código online.
    - Trabaja con varios lenguajes.

7. CodeLite.
    - Multiplataforma, libre y de código abierto.
    - Útil para el lenguaje C/C++ usando wxWidgets para su interfaz gráfica.
    - Ofrece gestión de proyectos.

8. NetBeans.
    - Gratuito y de código abierto.
    - Posee el autocompletado de código inteligente.
    - Ofrece un compilador y un intérprete.
    - Está orientado al lenguaje Java, aunque también soporta otros lenguajes.

9. Intellij IDEA.
    - Su lenguaje principal es Java, aunque también soporta otros lenguajes JVM como Kotlin, Scada y Groovy.
    - Su interfaz gráfica es prácticamente configurable en su totalidad.
    - Permite ampliar su funcionalidad, así como sus lenguajes, añadiendo complementos.

10. QC-Qt Creator.
    - Está diseñado para la programación en lenguaje C++, aunque también soporta C#, .NET, Python, Ada, Pascal, Perl, PHP y Ruby.
    - Es multiplataforma.
    - Posee un depurador visual.
    - Tiene soporte para refactorización de código.

11. Eclipse Theia.
    - Es una plataforma para desarrollar tanto aplicaciones de escritorio, como para lanzarse en la nube.
    - Está construido sobre la base de la arquitectura backend/frontend.
    - Su lenguaje principal es Java, aunque también soporta JavaScript y Python entre otros.

12. RStudio.
    - Utilizado para programar con lenguaje R, dedicado a la computación estadística y gráficos.
    - Está disponible en configuraciones de escritorio y servidor.
    - Depurador interactivo para diagnosticar y corregir errores rápidamente.

13. XCode.
    - IDE exclusivo para sistemas operativos de Apple.
    - Sus lenguajes son Objective-C o Swift, aunque también admite C, C++, JavaScript y Ruby.
    - Está disponible para desarrolladores de manera gratuita.

Fuentes:

https://geekflare.com/es/ide-for-programmer/

https://laboratoriolinux.es/index.php/-noticias-mundo-linux-/software/23243-bluej-un-ide-para-aprender-java-de-manera-interactiva-y-visual.html

https://spa.myservername.com/java-development-using-eclipse-ide

https://www.comparasoftware.es/slickedit#:~:text=%C2%BFQu%C3%A9%20es%20SlickEdit%3F,de%20programaci%C3%B3n%20que%20ahorran%20tiempo.

https://ubunlog.com/codeblocks-ide-desarrollo-ubuntu/

https://kripkit.com/estudio-de-programacin-gnat/

https://docs.aws.amazon.com/es_es/cloud9/latest/user-guide/welcome.html

https://sites.google.com/site/aprendizajebasadoendiseno/unidad-v-desarrollo-de-proyectos/entorno-de-desarrollo-c

https://www.crehana.com/blog/transformacion-digital/que-es-netbeans/

https://www.jetbrains.com/es-es/idea/features/

https://www.ecured.cu/Qt_Creator

https://blog.desdelinux.net/eclipse-theia-1-0-la-alternativa-open-source-a-visual-studio/

https://blog.desdelinux.net/rstudio-un-ide-para-el-lenguaje-de-programacion-r-en-linux/

https://tecno-simple.com/que-es-apple-xcode-ide/

Ejercicio 2

Realiza en NetBeans las siguientes tareas (deberás mostrar aquellos pantallazos más relevantes y explicar todos los pasos que se vayan dando).

a. **Importar el proyecto que se adjunta a la tarea llamado Tarea2 (Se da una carpeta comprimida en un fichero .zip que habrá que descomprimir primero). Se recomienda ubicar dicha carpeta en la carpeta de proyectos de NetBeans.**

b. **Como podrás ver este proyecto está lleno de errores. Tienes que conseguir que la consola muestre todos los mensajes que están indicados mediante comentarios en el método main de la clase principal (Tarea2.java).**

c. **Indica para cada error el cambio hecho para corregirlo y el motivo de que diera cada error. Puedes hacerlo mientras comentario en el código.**

d. **Además de corregir los errores deberás eliminar también aquellos avisos que nos pueda indicar NetBeans.**

El proyecto de este ejercicio se encuentra adjunto en el comprimido de la tarea.

Ejercicio 3

Joda Time es un API Java que permite trabajar con fechas de una forma más sencilla, potente y eficiente que el API estándar de fechas de Java. Especialmente a la hora de trabajar con intervalos. Vamos a crear un nuevo proyecto para usar esta librería (adjunta a la tarea). Tenemos que añadir la API al repositorio local del proyecto y añade el siguiente código (se puede añadir directamente en el main):

```
DateTime date1 = new DateTime();
System.out.println(date1);
// Imprime tu fecha de cumpleaños
DateTime date2 = new DateTime();
date2 = new DateTime("2000-12-30");
System.out.println(date2);
// Imprime la fecha actual pero metida por ti a mano
DateTime date3 = new DateTime();
date3 = new DateTime("2018-11-30");
System.out.println(date3);
```

a. Ejecuta ese código y muestra un pantallazo de lo que muestra por consola

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Ej03_Tarea2_Entornos ---
2022-11-09T17:55:02.286+01:00
1993-04-09T00:00:00.000+02:00
2022-11-09T00:00:00.000+01:00
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.032 s
Finished at: 2022-11-09T17:55:02+01:00
------------------------------------------------------------------------
```

b. ¿Has tenido que añadir algo? ¿Por qué?

He tenido que importar la librería para poder hacer uso de ella.

```
package com.mycompany.ej03_tarea2_entornos;
import org.joda.time.*;

/**
```

c. Debes entregar este proyecto de dos formas diferentes: empaquetado (.JAR) y el proyecto completo (.ZIP)