

Instalación y uso de entornos de desarrollo.



Caso práctico



Tras el éxito del anterior proyecto, en BK están recibiendo más peticiones de creación de software que nunca.

Ana y Antonio, que ya hace unas semanas que están estudiando el Ciclo de Diseño de Aplicaciones Multiplataforma, piensan que este es un buen momento para participar activamente en los proyectos, pues a sus compañeros no les vendría nada mal un poco de ayuda.

Ada confía en ellos, pero aún es pronto. Por lo menos, ya conocen las fases por las que tiene que pasar todo el desarrollo de aplicaciones, pero eso no será suficiente.

María, sin embargo, no piensa lo mismo y decide darles una oportunidad trabajando en la fase de codificación de un nuevo proyecto de la empresa.

Ana se muestra muy ilusionada y no piensa desperdiciar esta gran oportunidad. Sabe que tiene a su disposición los llamados entornos de desarrollo que le facilitarán su futura tarea.

¿Cómo influirá el conocimiento de esta herramienta en el futuro de Ana y Antonio? A través de esta unidad, veremos si nuestros amigos van logrando ganarse un puesto en la empresa, y de paso, la confianza de Ada.

La fase de codificación es compleja, pero Ana y Antonio están aprendiendo a dominar los llamados entornos integrados de desarrollo de software.



GOBIERNO
DE ESPAÑA

MINISTERIO
DE EDUCACIÓN, CULTURA
Y DEPORTE

Materiales formativos de FP Online propiedad del Ministerio de Educación, Cultura y Deporte.

[Aviso Legal](#)

1.- Concepto de entorno de desarrollo. Evolución histórica.



Caso práctico



Todos en la empresa están sorprendidos del entusiasmo de Ana ante los nuevos proyectos que BK programación tiene por delante. Juan, que acabó el Ciclo Superior de Desarrollo de Aplicaciones Informáticas (DAI) hace algunos años, se muestra inquieto porque es consciente de que en sólo unos cuatro años han salido muchas herramientas nuevas en el mercado y necesita reciclarse. Escucha a Ana decir que está estudiando los entornos de desarrollo.

—Yo también debería ponerme al día —piensa Juan.

En la unidad anterior tratamos las fases en el proceso de desarrollo de software.

Una de ellas era la fase de **codificación**, en la cual se hacía uso de algún lenguaje de programación para pasar todas las acciones que debía llevar a cabo la aplicación a algún lenguaje que la máquina fuera capaz de entender y ejecutar.

También se hizo alusión a herramientas de apoyo al proceso de programación. En esta unidad vamos a analizar, instalar y ejecutar estas herramientas para entender su acción y efecto.

Muchas personas aprenden a programar utilizando un **editor de texto** simple, **compilador** y **depurador**. Pero la mayoría, finalmente, terminan haciendo uso de algún entorno de desarrollo integrado para crear aplicaciones.

Un **entorno integrado de desarrollo (IDE)**, es un tipo de software compuesto por un conjunto de herramientas de programación. En concreto, el IDE se compone de:

- Editor de código de programación.
- Compilador.
- Intérprete.
- Depurador.
- Constructor de interfaz gráfico.

Los primeros entornos de desarrollo integrados nacieron a principios de los años 70, y se popularizaron en la década de los 90. Tienen el objetivo de ganar fiabilidad y tiempo en los proyectos de software. Proporcionan al programador una serie de componentes con la misma interfaz gráfica, con la consiguiente comodidad, aumento de eficiencia y reducción de tiempo de codificación.

Normalmente, un IDE está dedicado a un determinado lenguaje de programación. No obstante, las últimas versiones de los IDE tienden a ser compatibles con varios lenguajes (por ejemplo, Eclipse, NetBeans, Microsoft Visual Studio) mediante la instalación de plugins adicionales.

En este tema, nuestro interés se centra en conocer los entornos de desarrollo, los tipos, en función de su licencia y del lenguaje de programación hacia el cual están enfocados. Instalaremos NetBeans bajo Ubuntu y veremos cómo se configura y cómo se generan ejecutables, haciendo uso de sus componentes y herramientas.



Reflexiona

¿Tan necesario es un entorno de desarrollo? Si el código fuente es texto interpretado por un compilador,

¿por qué no es suficiente con un editor de texto?

1.1.- Evolución Histórica.

En las décadas de utilización de la [tarjeta perforada](#) como sistema de almacenamiento el concepto de **Entorno de Desarrollo Integrado** sencillamente no tenía sentido.

Los programas estaban escritos con [diagramas de flujo](#) y entraban al sistema a través de las **tarjetas perforadas**. Posteriormente, eran compilados.

El primer lenguaje de programación que utilizó un IDE fue el [BASIC](#) (que fue el primero en abandonar también las tarjetas perforadas o las cintas de papel).

Este primer IDE estaba basado en consola de comandos exclusivamente (normal por otro lado, si tenemos en cuenta que hasta la década de los 90 no entran en el mercado los sistemas operativos con interfaz gráfica). Sin embargo, el uso que hace de la gestión de archivos, compilación y depuración; es perfectamente compatible con los IDE actuales.

A nivel popular, el primer IDE puede considerarse que fue el IDE llamado **Maestro**. Nació a principios de los 70 y fue instalado por unos 22.000 programadores en todo el mundo. Lideró este campo durante los años 70 y 80.

El uso de los entornos integrados de desarrollo se ratifica y afianza en los 90 y hoy en día contamos con infinidad de IDE, tanto de licencia libre como no.

Tipos de entornos de desarrollo más relevantes en la actualidad.

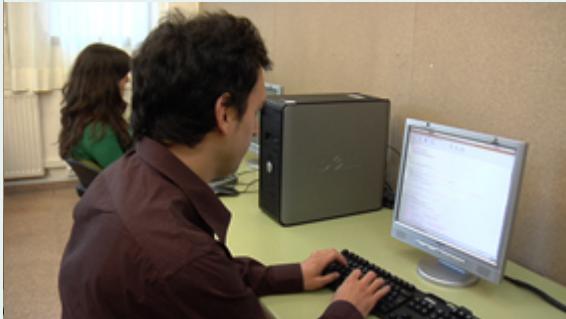
Entorno de desarrollo	Lenguajes que soporta	Tipo de licencia
NetBeans.	C/C++, Java, JavaScript, PHP, Python.	De uso público.
Eclipse.	Ada, C/C++, Java, JavaScript, PHP.	De uso público.
Microsoft Visual Studio.	Basic, C/C++, C#.	Propietario.
C++ Builder.	C/C++.	Propietario.
JBuilder.	Java.	Propietario.

No hay unos entornos de desarrollo más importantes que otros. La elección del IDE más adecuado dependerá del lenguaje de programación que vayamos a utilizar para la codificación de las aplicaciones y el tipo de licencia con la que queramos trabajar.

2.- Funciones de un entorno de desarrollo.



Caso práctico



Juan, que asume por fin su desconocimiento, habla con Ana para que le pase sus apuntes de entornos de desarrollo. Ésta se muestra encantada, y le anima a matricularse al ciclo de Desarrollo de Aplicaciones Multiplataforma (DAM) a distancia. Juan se muestra reacio (ya he estudiado el ciclo y durante cuatro años he cumplido con éxito en la empresa). Pero piensa que quizás debería reciclarse si no quiere quedarse atrás en los proyectos. Juan aprendió a programar usando un editor simple de textos, ¿qué ventajas tendrá programando con un IDE?

Como sabemos, los entornos de desarrollo están compuestos por una serie de herramientas software de programación, necesarias para la consecución de sus objetivos. Estas herramientas son:

- Un editor de código fuente.
- Un compilador y/o un intérprete.
- Automatización de generación de herramientas.
- Un depurador.



Las funciones de los IDE son:

- Editor de código: coloración de la [sintaxis](#).
- Auto-completado de código, atributos y métodos de clases.
- Identificación automática de código.
- Herramientas de concepción visual para crear y manipular componentes visuales.
- Asistentes y utilidades de gestión y generación de código.
- Archivos fuente en unas carpetas y compilados a otras.
- Compilación de proyectos complejos en un solo paso.
- Control de versiones: tener un único almacén de archivos compartido por todos los colaboradores de un proyecto. Ante un error, mecanismo de auto-recuperación a un estado anterior estable.
- Soporta cambios de varios usuarios de manera simultánea.
- Generador de documentación integrado.

- Detección de errores de sintaxis en tiempo real.

Otras funciones importantes son:

- Ofrece refactorización de código: cambios menores en el código que facilitan su legibilidad sin alterar su funcionalidad (por ejemplo cambiar el nombre a una variable).
- Permite introducir automáticamente tabulaciones y espaciados para aumentar la legibilidad.
- Depuración: seguimiento de variables, puntos de ruptura y mensajes de error del intérprete.
- Aumento de funcionalidades a través de la gestión de sus [módulos](#) y [plugins](#).
- Administración de las interfaces de usuario (menús y barras de herramientas).
- Administración de las configuraciones del usuario.



Autoevaluación

Un entorno integrado de desarrollo está compuesto por:

- Editor de código y traductor.
- Editor de código, compilador e interfaz de comandos.
- Editor de código, compilador, intérprete, depurador e interfaz gráfica.
- Interfaz gráfica, editor de código y depurador.

Incorrecta, se compone de más herramientas.

No es correcta porque la interfaz es gráfica.

Muy bien. Esa es la idea.

No es del todo correcta: faltaría el traductor de código (compilador e intérprete).

Solución

- 1.** Incorrecto (#answer-7_63)
- 2.** Incorrecto (#answer-7_81)
- 3.** Opción correcta (#answer-7_84)
- 4.** Incorrecto (#answer-7_87)

3.- Entornos integrados libres y propietarios.



Caso práctico



Juan ha buscado por Internet distintos entornos de desarrollo para aplicarlos en la fase de codificación.

—Cuidado —le dice Ada—. Ya sabes que es de vital importancia el tema de la Licencia de Software. Hay Entornos de desarrollo de licencia libre y otros no, y este aspecto es fundamental ni no queremos tener problemas.

Entornos Integrados Libres

Son aquellos con licencia de uso público.

No hay que pagar por ellos, y aunque los más conocidos y utilizados son Eclipse y NetBeans, hay bastantes más.

Tipos de entornos de desarrollo libres más relevantes en la actualidad.

IDE	Lenguajes que soporta	Sistema Operativo
NetBeans.	C/C++, Java, JavaScript, PHP, Python.	Windows, Linux, Mac OS X.
Eclipse.	Ada, C/C++, Java, JavaScript, PHP.	Windows, Linux, Mac OS X.
Gambas.	Basic.	Linux.
Anjuta.	C/C++, Python, Javascript.	Linux.
Geany.	C/C++, Java.	Windows, Linux, Mac OS X.
GNAT Studio.	Fortran.	Windows, Linux, Mac OS X.

El aspecto de la licencia del IDE que se elija para el desarrollo de un proyecto es una cuestión de vital importancia. En su elección prevalecerá la decisión de los supervisores del proyecto y de la dirección de la empresa.



Para saber más

En el siguiente enlace encontrarás un documento muy interesante, en inglés, donde por cada lenguaje de programación se indican los entornos de desarrollo más utilizados. Además, también podrás ver qué entornos fueron los más utilizados durante 2018.

[Entornos de desarrollo \(https://www.g2.com/articles/ide#best-ide\)](https://www.g2.com/articles/ide#best-ide)

Entornos Integrados Propietarios

Son aquellos entornos integrados de desarrollo que necesitan licencia. No son free software, hay que pagar por ellos.

El más conocido y utilizado es Microsoft Visual Studio, que usa el framework .NET y es desarrollado por Microsoft.

Tipos de entornos de desarrollo propietarios más relevantes en la actualidad.

IDE	Lenguajes que soporta	Sistema Operativo
Microsoft Visual Studio. FlashBuilder. C++ Builder. Turbo C++ profesional. JBuilder. JCreator. Xcode.	Basic, C/C++, C#. ActionScript. C/C++. C/C++. Java. Java. C/C++, Java.	Windows. Windows, Mac OS X. Windows. Windows. Windows, Linux, Mac OS X. Windows. Mac OS X.



Caso práctico

Microsoft Visual Studio hemos visto que es un entorno de desarrollo propietario que usa el framework .NET de Microsoft. Sin embargo, existe otro entorno de desarrollo llamado Visual Studio Code, ¿sabrías identificar las características de este último IDE? ¿es libre o propietario? ¿qué lenguajes soporta? ¿para qué sistemas operativos está disponible? ¿se utiliza mucho?



Autoevaluación

Relaciona los siguientes entornos de desarrollo con sus características, escribiendo el número asociado a la característica en el hueco correspondiente.

Ejercicio de relacionar

Entorno de desarrollo.	Relación	Características.
Microsoft Visual Studio.	<input type="text"/>	1. Libre. Soporta C/C++, Java, PHP, Javascript, Python.
NetBeans.	<input type="text"/>	2. Propietario. Soporta Basic, C/C++, C#.
C++ Builder.	<input type="text"/>	3. Propietario. Soporta C/C++.

Enviar

En la elección del entorno de desarrollo más adecuado para desarrollar un proyecto de software influye el tipo de licencia del entorno y los lenguajes de programación que soporta.

4.- Estructura de entornos de desarrollo.



Caso práctico



Juan aprendió a programar utilizando un editor de textos, un compilador y un depurador. Todas estas herramientas se instalaban de forma independiente. A Ana le cuesta creer que los programadores tuvieran que buscar estas herramientas e instalarlas por separado. –En un entorno se integran todas estas cosas y muchas más, y sin salir del mismo puedes programar en varios lenguajes y puedes documentar y.... –Ya lo veo, –le replica Juan–.¿Cuántos componentes tiene el entorno en total?

Los entornos de desarrollo, ya sean libres o propietarios, están formados por una serie de componentes software que determinan sus funciones. Estos componentes son:

- **Editor de textos:** Resalta y colorea la sintaxis, tiene la función de autocompletar código, ayuda y listado de parámetros de funciones y métodos de clase. Inserción automática de paréntesis, corchetes, tabulaciones y espaciados.
- **Compilador/intérprete:** Detección de errores de sintaxis en tiempo real. Características de refactorización [\[+\]](#).
- **Depurador:** Botón de ejecución y traza, [puntos de ruptura](#) [\[+\]](#) y seguimiento de variables. Opción de depurar en [servidores remotos](#) [\[+\]](#).
- **Generador automático de herramientas:** Para la visualización, creación y manipulación de componentes visuales y todo un arsenal de asistentes y utilidades de gestión y generación código.
- **Interfaz gráfica:** Nos brinda la oportunidad de programar en varios lenguajes con un mismo IDE. Es una interfaz agradable que puede acceder a innumerables [bibliotecas](#) [\[+\]](#) y plugins, aumentando las opciones de nuestros programas.

```
TIJ.java:11
1 package TIJ;
2
3 import java.util.Scanner;
4
5 public class TIJ {
6
7     public static void main(String[] args) {
8
9         Scanner teclado = new Scanner(System.in);
10        System.out.print("Introduce un número: ");
11        int numero = teclado.nextInt();
12        System.out.println("El " + numero + "º número de la serie de F");
13        int fibo1=0;
14        int fibo2=1;
15
16        System.out.print(fibo1 + " ");
17        for(int i=2;i<numero;i++){
18            System.out.print(fibo2 + " ");
19            fibo2 = fibo1 + fibo2;
20            fibo1 = fibo2 - fibo1;
21        }
22        System.out.println();
23    }
24 }
```

Problems View
TIJ [Java Application] /src/TIJ/TIJ.java [Modificado] (11.0 kB) [Cambiado] [Renglon] [Dif.] [C] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Y] [Z]
Exception in thread "main" java.lang.ArithmaticException: / by zero
at TIJ.TIJ.main(TIJ.java:24)

5.- Instalación de entornos integrados de desarrollo.



Caso práctico



Juan está decidido a aprender a usar un entorno de desarrollo. Después de documentarse, piensa que lo idóneo es trabajar con un IDE libre. Además, el tema del sistema operativo que soporta es importante. Juan quiere trabajar bajo Windows, y se decide por el entorno Eclipse. Ahora bien, ¿Qué hay que hacer para instalarlo?

Vamos a realizar la instalación de Eclipse, existen diferentes formas de instalarlo.

5.1.- Instalación de JDK.

La instalación del IDE NetBeans y Eclipse, ya sea en Linux, Windows o Mac OS X, requiere la instalación previa del JDK compatible con la versión de NetBeans o Eclipse que se quiera instalar.

Lo primero es instalar el JDK en el sistema operativo. Esta será la plataforma del entorno, imprescindible para que éste pueda ser instalado en el sistema operativo y funcionar.

Su instalación en un Sistema Operativo como Windows o MacOS es trivial. Por ello, a continuación se darán unas instrucciones concretas en caso de querer instalarlo en un Sistema Operativo Linux.

Tanto la versión del jdk como de Ubuntu no son las más recientes; no obstante, el proceso de instalación es semejante.

El proceso de instalación sólo podrá ser realizado por el root, que es el super-usuario. Por ello, la instalación se realizará desde la consola de comandos:

Versión de JDK elegida:

JDK-6u24-linux-i586.

Órdenes en la consola de comandos:

- Obtener el archivo, que se adjunta como recurso en la presente unidad.
- Mover el archivo a `/usr/local`.
- Darle permisos de ejecución, como root del sistema.
- Ejecutarlo, como root.

El proceso de instalación en Linux consta de una serie de pasos, y se explican con detalle en el siguiente documento:

[Instalación de JDK en Ubuntu 10.10](#)

JDK son las siglas de Java Development Kit: Kit de desarrollo de Java. Consiste en la plataforma del entorno, imprescindible para que éste pueda ser instalado y ejecutado.

Para saber más

Durante una tutoría de la primera UD, un compañero ha comentado que Java a partir de la versión 11 había cambiado su licencia (podéis ver el foro) y que para llevar tus programas a producción había que pagar a Oracle (su propietario). Os animo que leais el siguiente artículo para saber más sobre esto.

<https://www.campusmvp.es/recursos/post/java-11-ya-esta-aqui-te-toca-pagar-a-oracle-o-cambiarte-a-otras-opciones.aspx>

5.2.- Instalación de Eclipse

La instalación de Eclipse es muy sencilla, se realiza desde: <https://www.eclipse.org/downloads/>. Debemos seleccionar la última versión para nuestro sistema operativo.

The screenshot shows the Eclipse Foundation website's download section. At the top, there are navigation links for 'Log in', 'Manage Cookies', 'Projects', 'Working Groups', 'Members', 'More', and a search bar. A prominent orange banner on the left says 'Download Eclipse Technology that is right for you'. To the right, there is a 'Sponsored Ad' for 'Red Hat Developer' with the text 'Build here. Go anywhere.' and a button to 'Join the DevNation'. Below the ad, there is a 'Tool Platforms' section featuring 'Eclipse Che' and 'ORION'.

The Eclipse Installer 2021-09 R now includes a JRE for macOS, Windows and Linux.



Get **Eclipse IDE 2021-09**

Install your favorite desktop IDE packages.

[Download x86_64](#)

[Download Packages](#) | [Need Help?](#)

Tool Platforms



Eclipse Che is a developer workspace server and cloud IDE.



A modern, open source software development environment that runs in the cloud.

Una vez se descargue el instalador, él se ocupa de la creación automática de iconos, entradas en el menú inicio, etc.. Un detalle importante es fijarse a la elección entre 32 y 64 bits, ya que eclipse necesita que tengamos instalada la máquina virtual java. Seleccionaremos 32 ó 64 bits para que ambas versiones (la de Eclipse y la de la máquina virtual de java o JRE) coincidan.



Orientaciones para el alumnado

En el siguiente enlace puedes acceder a un tutorial de instalación de Eclipse.

<https://www.eclipse.org/downloads/packages/installer>

6.- Configuración y personalización de entornos de desarrollo.



Caso práctico

Juan está consternado. Eclipse parece albergar tanta información que no sabe por donde empezar. Le gustaría personalizar la configuración de su primer proyecto en el IDE (que va a ser un aplicación de Java). ¿Cómo lo hace? ¿Qué parámetros puede configurar?

Una vez tenemos instalado nuestro entorno de desarrollo podemos acceder a personalizar su configuración.

Al abrir un proyecto existente, o bien crear un nuevo proyecto, seleccionaremos un desplegable con el nombre de "configuración" desde el que podremos personalizar distintas opciones del proyecto.

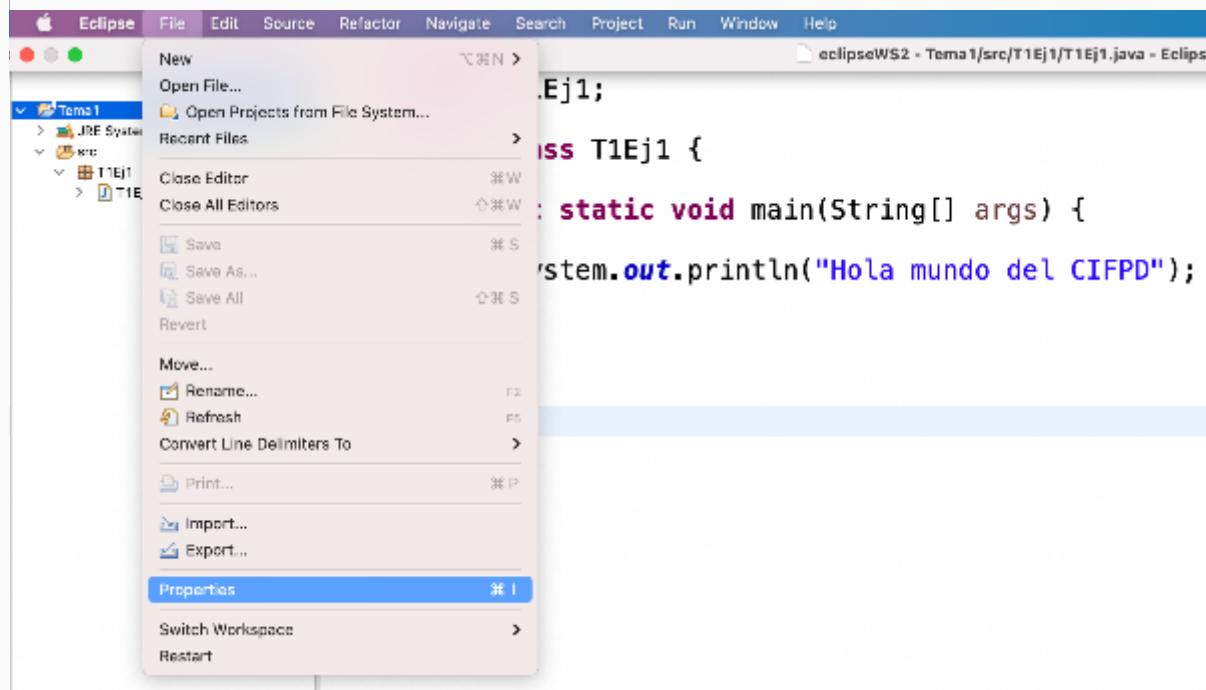
Podemos personalizar la configuración del entorno sólo para el proyecto actual, o bien para todos los proyectos, presentes y futuros.

Parámetros configurables del entorno:

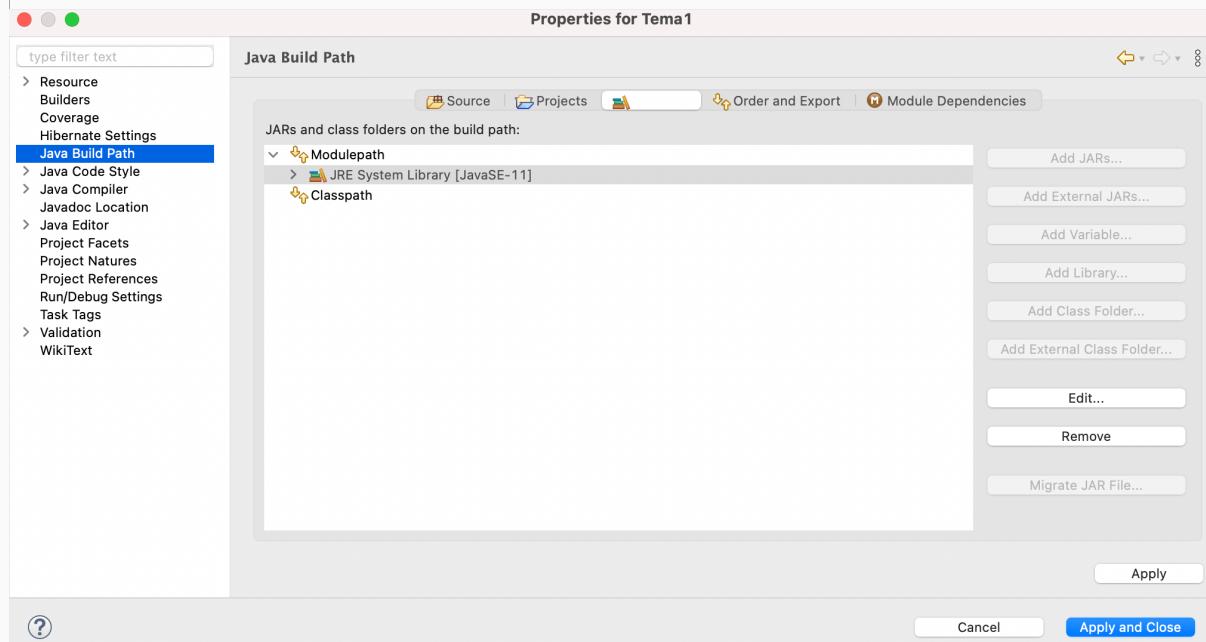
- Carpeta o carpetas donde se alojarán todos los archivos de los proyectos (es importante la determinación de este parámetro, para tener una estructura de archivos ordenada).
- Carpetas de almacenamiento de paquetes fuente y paquetes prueba.
- Administración de la plataforma del entorno de desarrollo.
- Opciones de la compilación de los programas: compilar al grabar, generar información de depuración.
- Opciones de [empaquetado](#) de la aplicación: nombre del archivo empaquetado (con extensión .jar, que es la extensión característica de este tipo de archivos empaquetados) y momento del empaquetado.
- Opciones de generación de documentación asociada al proyecto.
- Descripción de los proyectos, para una mejor localización de los mismos.
- Opciones globales de formato del editor: número de espaciados en las sangrías, color de errores de sintaxis, color de etiquetas, opción de autocompletado de código, propuestas de insertar automáticamente código.
- Opciones de combinación de teclas en teclado.
- Etc.

Configurar versión Java (Eclipse)

En primer lugar, deberemos acceder a File > Properties



Una vez haces clic en las propiedades, podrás acceder a una ventana nueva donde existen numerosos aspectos a configurar. En nuestro caso entraremos dentro del apartado *Java Build Path*. En la imagen podemos ver que la versión Java que se está utilizando es la 11.



Si deseamos cambiar la versión de Java con la que se ejecutarán nuestros proyectos, deberemos hacer clic sobre la configuración actual (en nuestro caso JRE System Library [JavaSE-11]). Al hacer clic sobre él se habilitará el botón *Edit...* y podremos ver la siguiente ventana.

Edit Library

JRE System Library

Select JRE for the project build path.



System library

- Execution environment: JavaSE-11 (Java SE 11.0.8 [11.0.8])
- Alternate JRE:
- Workspace default JRE (Java SE 11.0.8 [11.0.8])



Cancel

Finish

Dentro de esta ventana, haremos clic en *Alternate JRE* y podremos ver una lista desplegable de las versiones Java que tenemos instaladas en el ordenador.

Edit Library

JRE System Library

Select JRE for the project build path.



System library

- Execution environment: JavaSE-11 (Java SE 11.0.8 [11.0.8])
- Alternate JRE:
- Workspace default JRE
- ✓ Java [1.8.271.09]
 - Java SE 11.0.8 [11.0.8]
 - Java SE 8 [1.8.0_241]



Cancel

Finish

En el caso de querer instalar un nuevo JRE puedes seleccionar *Installed JREs* y añadirlos a la lista que os aparece en la nueva ventana. Para ello deberás indicar dónde está localizado el JRE y darle un nombre descriptivo.

Preferences (Filtered)

The selected JRE does not support the current compiler compliance level of 11

Add, remove or edit JRE definitions. By default, the checked JRE is added to the build path of newly created Java projects.

Installed JREs:

Name	Location
<input type="checkbox"/> Java [1.8.271.09]	/Library/Internet Plug-Ins/JavaAppletPlugin.plu
<input type="checkbox"/> Java SE 11.0.8 [11.0.8]	/Library/Java/JavaVirtualMachines/jdk-11.0.8.j
<input checked="" type="checkbox"/> Java SE 8 [1.8.0_24...]	/Library/Java/JavaVirtualMachines/jdk1.8.0

Add... **Edit...** **Duplicate...** **Remove** **Search...**

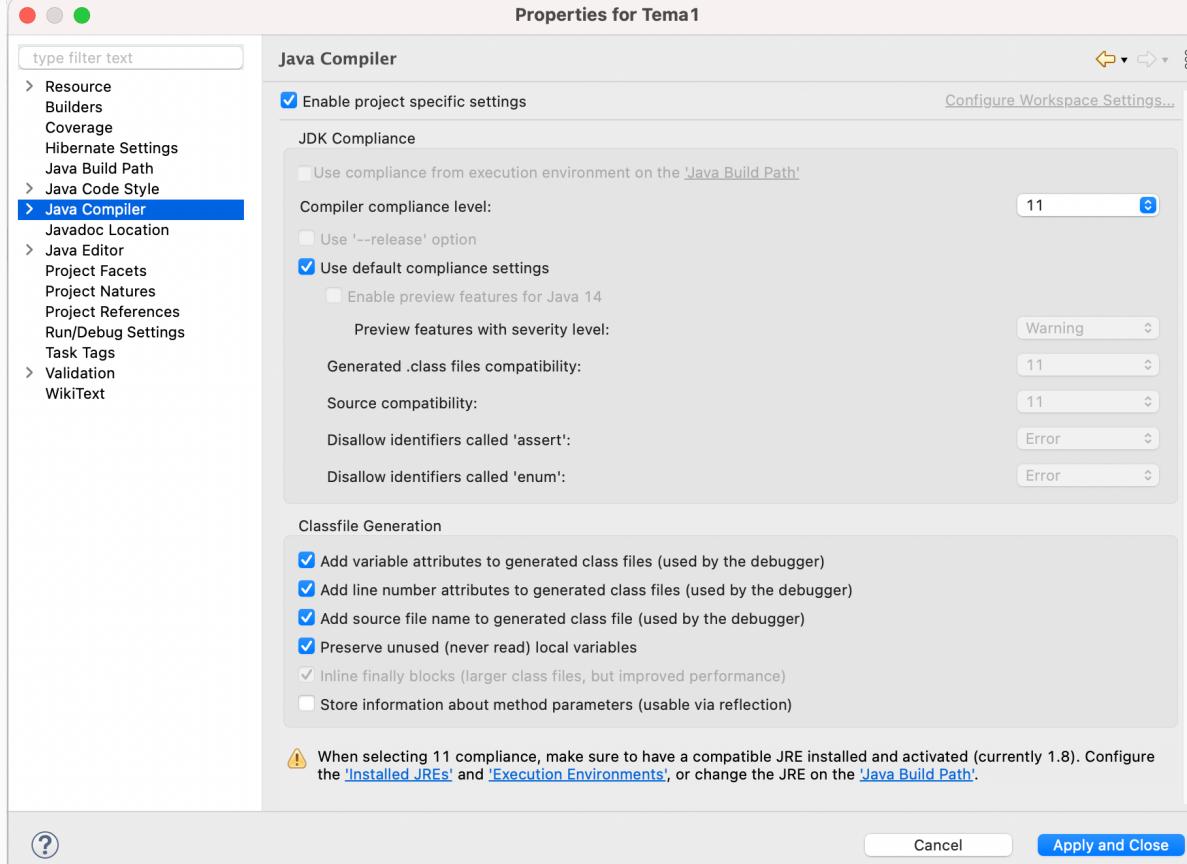
Conflicting compliance settings can be changed on the [Compiler](#) page.

Apply

Cancel Apply and Close

Si al cambiar la librería de Java modificas la versión (como es el caso anterior) tendremos que configurar la compilación de los proyectos también.

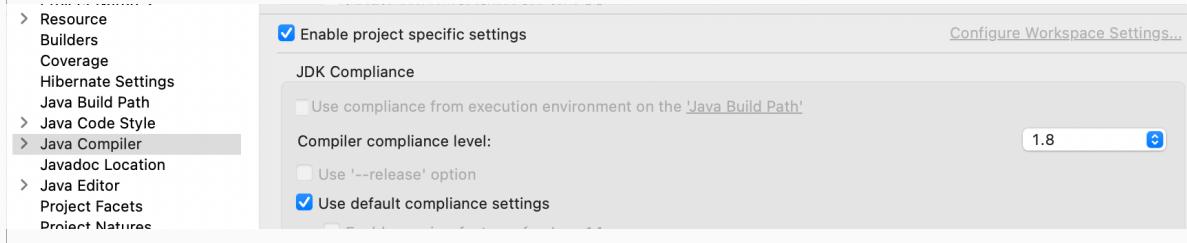
Para cambiar la configuración de la compilación, volveremos a la ventana de propiedades y seleccionaremos el apartado *Java Compiler*. Como podemos ver en la siguiente imagen, la configuración de la compilación está definida para la versión 11; sin embargo, la versión que hemos configurado anteriormente es la 1.8.



Cancel

Apply and Close

Para que el proyecto pueda ser compilado y ejecutado, tendremos que configurarlo también para la versión 1.8, como aparece a continuación.



7.- Gestión de módulos en Eclipse



Caso práctico



Después de haber probado a configurar algunos aspectos del entorno, ahora Juan desea empezar a programar. Tiene un trabajo pendiente en Python, pero observa que, tristemente, este lenguaje no es soportado por Eclipse.

—¿Cómo que no? —Le dice Ana. —Basta con encontrar el módulo de Python (estructuras del lenguaje más bibliotecas asociadas) y añadirlo como complemento al entorno. Entonces sí que podrás programar (también) en ese lenguaje.

A Juan le parece fascinante.

Con la plataforma dada por un entorno de desarrollo como Eclipse podemos hacer uso de módulos y plugins para desarrollar aplicaciones.

En la página oficial de Eclipse encontramos una relación de módulos y plugins. Un módulo es un componente software que contiene clases de Java que pueden interactuar con las API del entorno de desarrollo y el manifest file, que es un archivo especial que lo identifica como módulo.

Los módulos se pueden construir y desarrollar de forma independiente. Esto posibilita su reutilización y que las aplicaciones puedan ser construidas a través de la inserción de módulos con finalidades concretas. Por esta misma razón, una aplicación puede ser extendida mediante la adición de módulos nuevos que aumenten su funcionalidad.

Existen en la actualidad multitud de módulos y plugins disponibles para todas las versiones de los entornos de desarrollo más utilizados. En las secciones siguientes veremos dónde encontrar plugins y módulos para Eclipse que sean de algún interés para nosotros y las distintas formas de instalarlos en nuestro entorno.

También aprenderemos a desinstalar o desactivar módulos y plugins cuando preveamos que no los vamos a utilizar más y cómo podemos estar totalmente actualizados sin salir del espacio de nuestro entorno.

Veremos las categorías de plugins disponibles, su funcionalidad, sus actualizaciones...



Autoevaluación

¿Cómo crees que influye el hecho de tener módulos y plugins disponibles en el éxito que tenga un IDE?

- Contribuyen al éxito del entorno.
- No influyen en el éxito del entorno.

Efectivamente. Poder añadir funcionalidades concretas según lo que necesitemos hacen que el IDE sea muy aceptado por los usuarios.

No. La aceptación de un entorno será mayor si permite que el usuario decida qué funcionalidades desea incluir.

Solución

1. Opción correcta (#answer-32_63)

2. Incorrecto (#answer-32_243)

7.1.- Añadir.



Caso práctico

Ya sabemos que podemos añadir funcionalidades a nuestro entorno. Pero ni Juan ni Ana saben cómo hacerlo. Piden ayuda a María, que decide ayudarles.

—Añadir módulos y plugins es muy sencillo, prestad atención.



Añadir un módulo (o plug-in) va a provocar dotar de mayor funcionalidad a nuestro IDE; funcionalidad que por algún motivo determinado no viene por defecto. . .

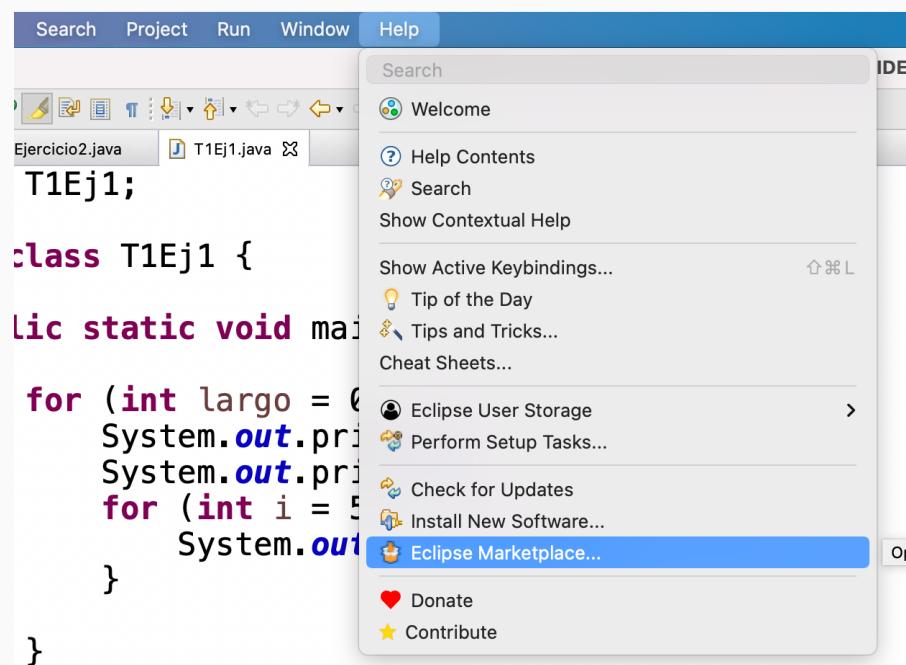
Para añadir un nuevo plug-in tenemos dos opciones principalmente:

1. Buscar un nuevo módulo e instalarlo a través del Eclipse Marketplace.
2. Instalar un nuevo plug-in a través de su URL.

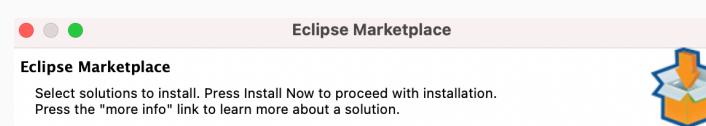
Lo más usual es que si vamos a instalar un plug-in muy concreto que no es habitual o queremos instalar una versión concreta, utilicemos la segunda forma. Por el contrario, si el plug-in es muy habitual, utilizaremos la primera.

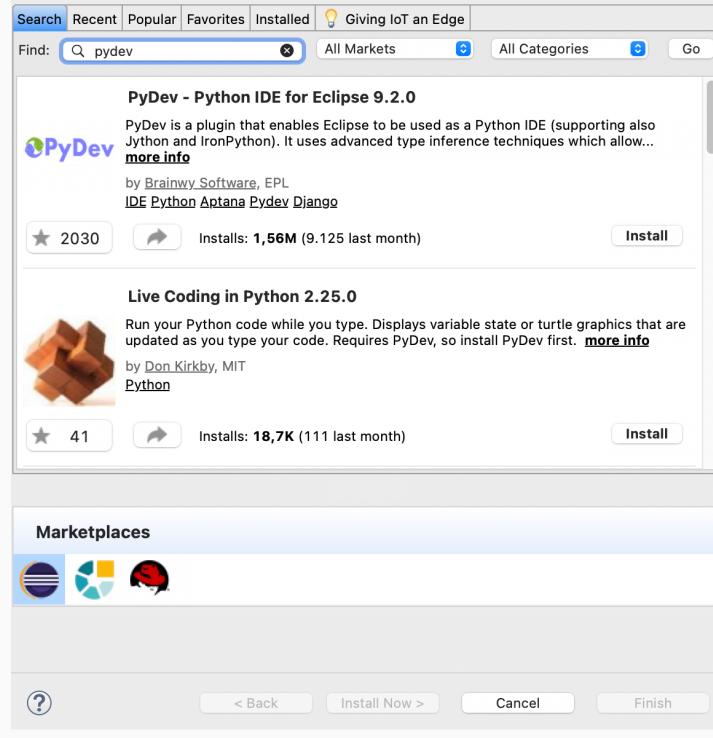
A través del Marketplace

Respecto a la primera forma de instalación. Únicamente tendrás que acceder a Help > Eclipse Marketplace...



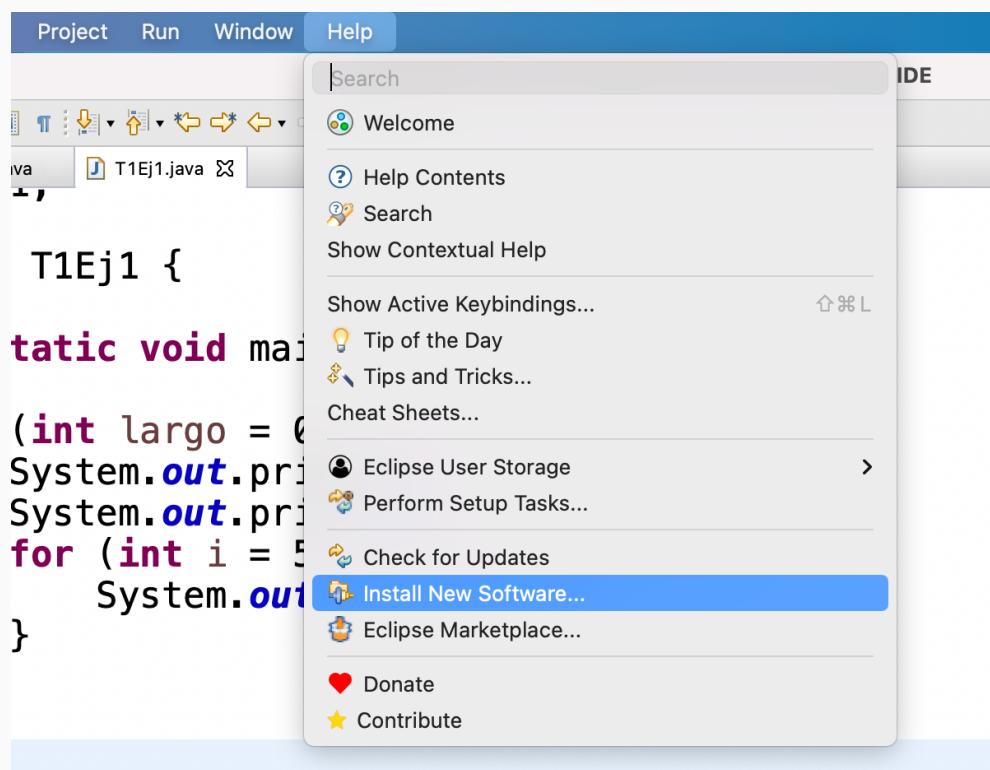
Una vez que accedas, verás una ventana semejante a la siguiente imagen. En la imagen puedes observar como he buscado un plug-in llamado PyDev. Si estoy contengo con la búsqueda, y quiero instalar uno de los que aparecen, únicamente tendré que hacer clic en el botón *Install*.





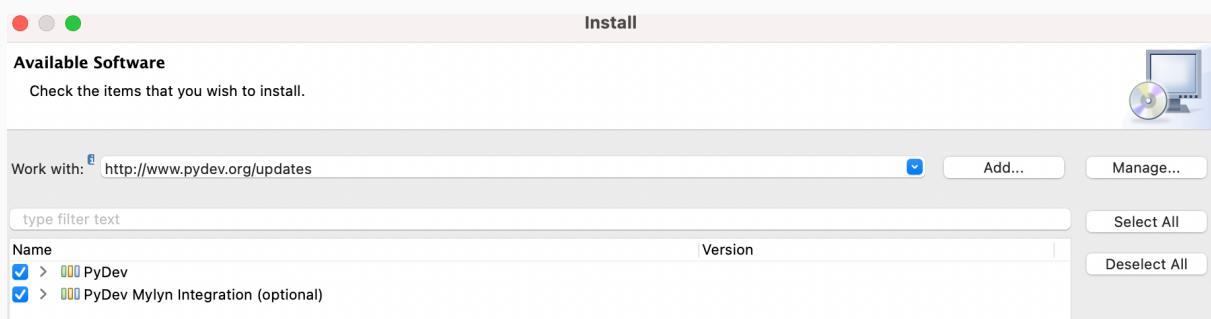
A través de la URL

Para ello tendremos que hacer clic en Help > Install New Software



Una vez haces clic, verás una ventana donde puedes introducir la URL de un sitio donde se localice un plugin. A modo de ejemplo, introduciremos la URL de PyDev (<http://www.pydev.org/updates>) para ver cómo sería su instalación. Al introducir dicha URL en el cuadro de texto, veremos los plug-in disponibles en dicha URL.

Lo único que tendríamos que hacer es seleccionarlo y seguir los pasos que nos ofrece la ventana de diálogo.



3 items selected

Details

- Show only the latest versions of available software
 Group items by category
 Show only software applicable to target environment
 Contact all update sites during install to find required software
- Hide items that are already installed
What is [already installed?](#)



< Back

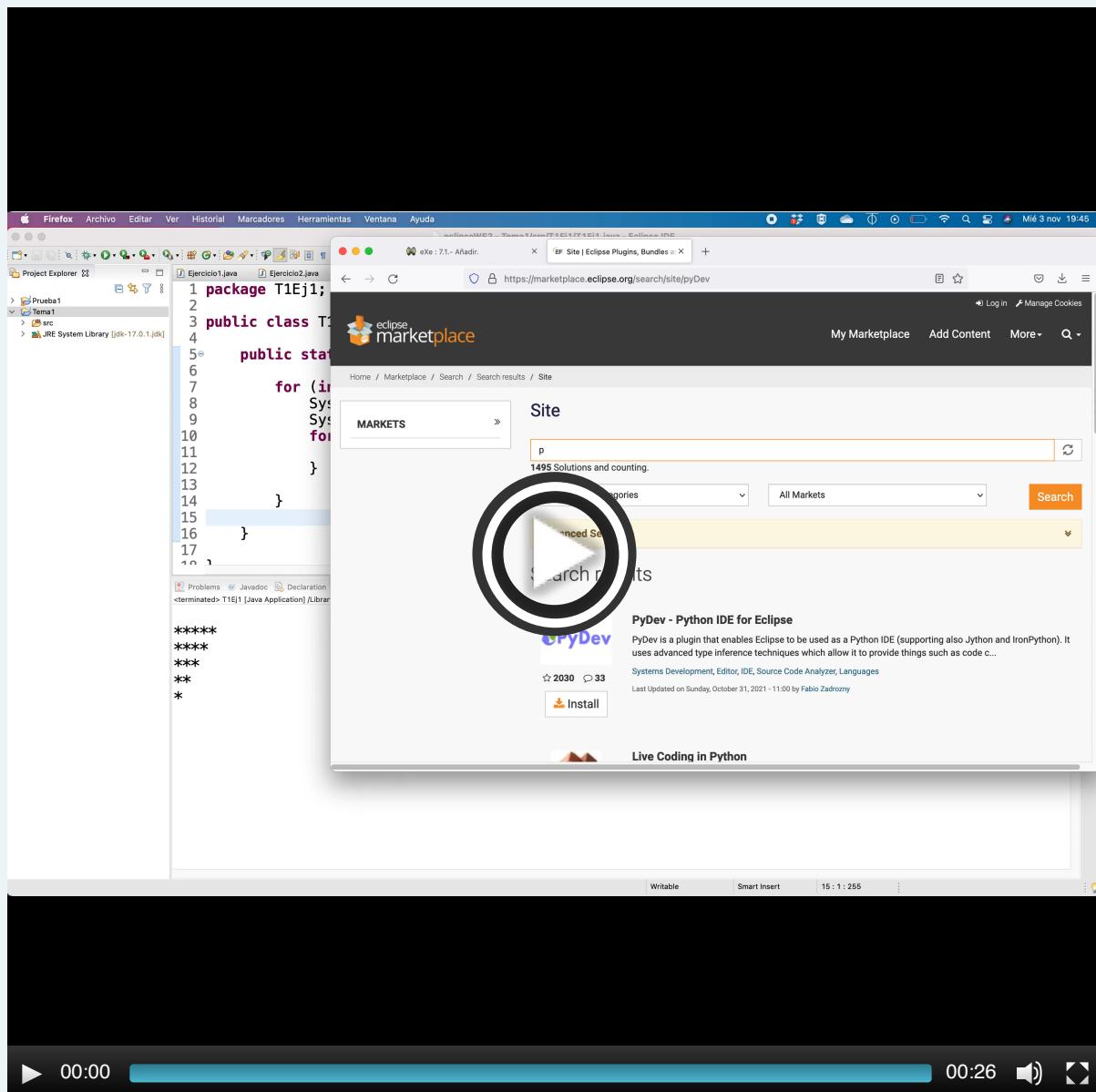
Next >

Cancel

Finish

Para saber más

Aunque no he hablado de ello, otra forma de instalar un plug-in en Eclipse es buscándolo en la web <https://marketplace.eclipse.org/> y arrastrar el ícono Install a tu ventana de Eclipse (OJO. No dentro del editor de texto de Eclipse). Puedes ver el siguiente vídeo.

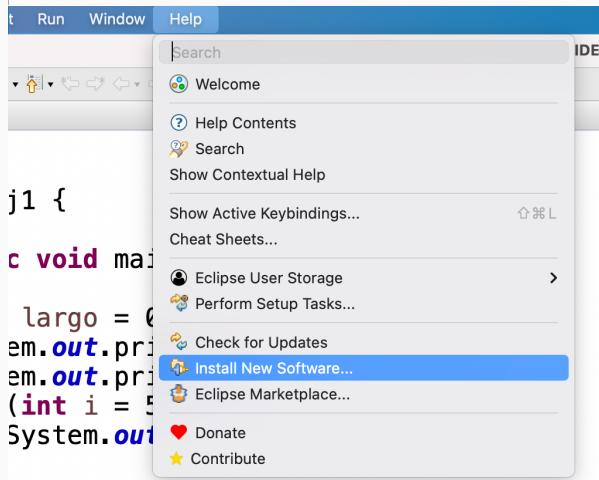


7.2.- Eliminar.

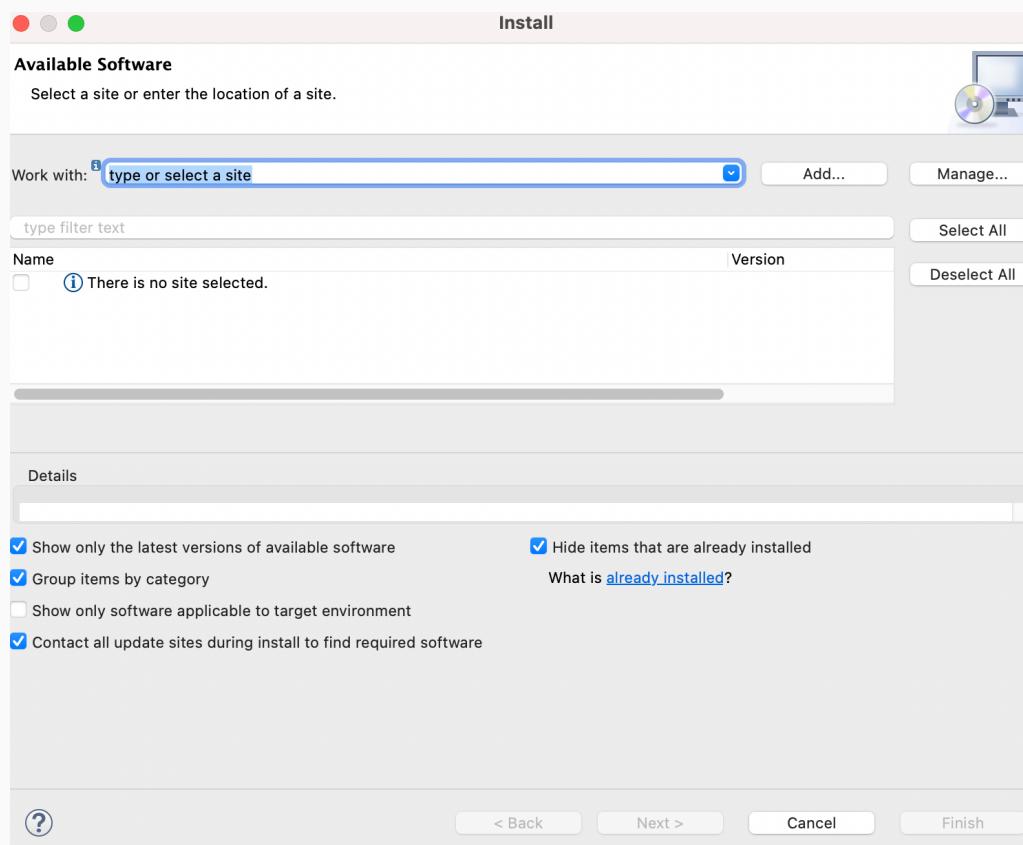
Cuando consideramos que algún módulo o plugin de los instalados no nos aporta ninguna utilidad, o bien que el objetivo para el cual se añadió ya ha finalizado, el módulo deja de tener sentido en nuestro entorno. Es entonces cuando nos planteamos eliminarlo.

Eliminar un módulo es una tarea trivial que requiere seguir los siguientes pasos.

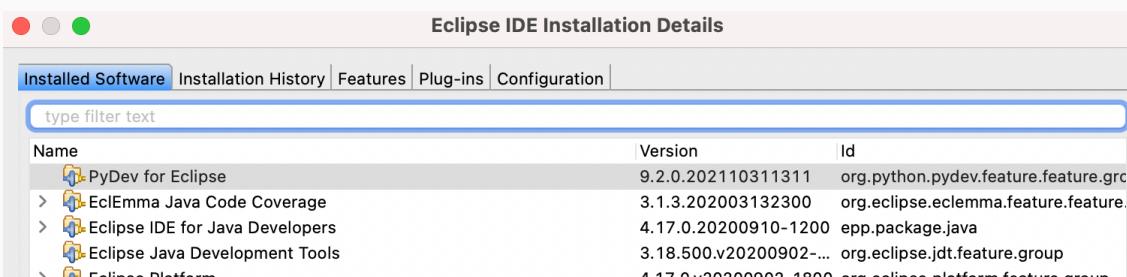
1.Help > Install New Software...



2. Clic en "already installed"



3. Desde la pestaña *Installed Software* se puede acceder al plugin que deseamos desinstalar. En este caso hemos seleccionado PyDev, y haremos clic en Uninstall... Posteriormente lo único que tendremos que hacer es seguir los pasos.



>  Eclipse Platform	4.17.0.v20200902-1800	org.eclipse.platform.feature.group
>  Eclipse RCP	4.17.0.v20200902-1800	org.eclipse.rcp.feature.group
>  Git integration for Eclipse	5.9.0.202009080501-r	org.eclipse.egit.feature.group
 Hibernate Tools	5.4.10.v20200903-1131	org.hibernate.eclipse.feature.feature
 Java implementation of Git	5.9.0.202009080501-r	org.eclipse.jgit.feature.group
>  JBoss Maven Hibernate Configurator	1.9.100.v20200609-1...	org.jboss.tools.maven.hibernate.feat
 JBoss Tools Usage Reporting	2.2.300.v20200323-1...	org.jboss.tools.usage.feature.feature
>  m2e - Maven Integration for Eclipse (includes Incubating components)	1.16.1.20200710-1008	org.eclipse.m2e.feature.feature.grou
 m2e - slf4j over logback logging (Optional)	1.16.0.20200318-1040	org.eclipse.m2e.logback.feature.feat
>  Marketplace Client	1.8.4.v20200709-0857	org.eclipse.epp.mpc.feature.group
 Mylyn WikiText Editors	3.0.38.202008172112	org.eclipse.mylyn.wikitext.editors_fe
>  Oomph Setup	1.18.0.v20200901-1012	org.eclipse.oomph.setup.feature.gro
 POM Editor using LemMinX language server (includes Incubating c...	1.16.2.20200902-0711	org.eclipse.m2e.lemminx.feature.fea
>  PyDev for Eclipse	9.2.0.202110311311	org.python.pydev.feature.feature.grp
 Pydev Mylyn Integration	0.6.0	org.python.pydev.mylyn.feature.feat
 Tip of the Day UI Feature	0.2.1100.v20200724-...	org.eclipse.tips.feature.feature.grou
 Wild Web Developer XML tools	0.10.2.202007231627	org.eclipse.wildwebdeveloper.xml.fe
>  Xtend IDE	2.24.0.v20201130-1016	org.eclipse.xtend.sdk.feature.group
>  Xtext Complete SDK	2.24.0.v20201130-1016	org.eclipse.xtext.sdk.feature.group

Python Development Environment



Update...

Uninstall...

Properties

Close

7.3.- Funcionalidades.



Caso práctico

—Para que sepas qué puedes encontrar en los complementos de Eclipse, te recomiendo que tengas claras las funcionalidades que ofrece, teniendo en cuenta que se van ampliando día a día, —le comenta Ana a Juan.



Los módulos y plugins disponibles para los entornos de desarrollo, en sus distintas versiones, tienen muchas y muy variadas funciones.

Podemos clasificar las distintas categorías de funcionalidades de módulos y plugins en los siguientes grupos:

- 1. Construcción de código:** facilitan la labor de programación.
- 2. Bases de datos:** ofrecen nuevas funcionalidades para el mantenimiento de las aplicaciones.
- 3. Depuradores:** hacen más eficiente la depuración de programas.
- 4. Aplicaciones:** añaden nuevas aplicaciones que nos pueden ser útiles.
- 5. Edición:** hacen que los editores sean más precisos y más cómodos para el programador.
- 6. Documentación de aplicaciones:** para generar documentación de los proyectos en la manera deseada.
- 7. Interfaz gráfica de usuario:** para mejorar la forma de presentación de diversos aspectos del entorno al usuario.
- 8. Lenguajes de programación y bibliotecas:** para poder programar bajo un Lenguaje de Programación que, en principio, no soporte la plataforma.
- 9. Refactorización:** hacer pequeños cambios en el código para aumentar su legibilidad, sin alterar su función.
- 10. Aplicaciones web:** para introducir aplicaciones web integradas en el entorno.
- 11. Prueba:** para incorporar utilidades de pruebas al software.

7.4.- Herramientas concretas.

Existe una gran cantidad de plug-ins para Eclipse, y los desarrolladores cada dia ponen a disposición de la comunidad más. Un ejemplo de los plug-ins más utilizaos lo podemos ver en el

Marketplace de Eclipse.

Spring tools 4

Nos permite utilizar de forma más sencilla las utilidades que ofrece el framework Spring.

Darkest Dark Theme with DevStyle

Plug-in para cambiar el estilo de Eclipse a oscuro.

Eclipse Color Theme

Plug-in para cambiar el estilo de Eclipse.

PyDev - Python IDE for Eclipse

Nos permitirá desarrollar en python dentro de Eclipse.

WindowBuilder

Nos permitirá desarrollar interfaces gráficas de forma más sencilla.

The screenshot shows the Eclipse Marketplace interface with a sidebar on the left and a main content area on the right. The sidebar includes links for 'Marketplace', 'Install New Software...', 'Available Updates...', 'Help', and 'Logout'. The main content area has tabs at the top: 'Alphabetical', 'Foundation Members', 'Last Updated', 'Most Popular' (which is selected), and 'Top Favorites'. Below these tabs are five plugin cards:

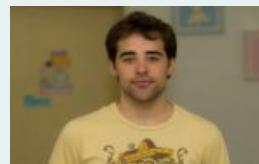
- Spring Tools 4 (aka Spring Tool Suite 4)**: A green gear icon. Rating: ★ 3279. Downloads: 124. Last updated: Wednesday, October 27, 2021 - 09:56 by Martin Lippert. Includes: IDE. [Install](#)
- Darkest Dark Theme with DevStyle**: A dark theme icon. Rating: ★ 4430. Downloads: 313. Last updated: Thursday, September 16, 2021 - 15:35 by Tim Webb. Includes: IDE, UI. [Install](#)
- Eclipse Color Theme**: A colorful icon. Rating: ★ 1498. Downloads: 53. Last updated: Monday, February 4, 2019 - 13:52 by Felix H. Dahlke. Includes: Editor, IDE, UI. [Install](#)
- PyDev - Python IDE for Eclipse**: A blue icon with a Python logo. Rating: ★ 2030. Downloads: 33. Last updated: Sunday, October 31, 2021 - 11:00 by Fabio Zadrozny. Includes: Editor, Systems Development, Languages, Source Code Analyzer, IDE. [Install](#)
- WindowBuilder**: A window builder icon. Rating: ★ 758. Downloads: 12. Last updated: Thursday, October 21, 2021 - 16:52 by Holger Voermann. Includes: Tools, UI, Editor, IDE. [Install](#)

8.- Uso básico de entornos de desarrollo.



Caso práctico

—En qué partes se divide el espacio principal del entorno? Vamos a echar un vistazo, —le comenta Juan a Antonio. (A Juan le gusta explicárselo a su compañero, ahora que va descubriendo las ventajas de los IDE).



En el sitio principal del entorno de desarrollo de Eclipse nos encontramos con la siguiente ventana, que aparece cuando seleccionamos File > New > Java Project

The screenshot shows the Eclipse IDE interface. The top menu bar includes Eclipse, File, Edit, Refactor, Source, Navigate, Search, Project, Run, Window, and Help. The title bar indicates "eclipseWS2 - Tema1/src/T1Ej1/T1Ej1.java - Eclipse IDE". The left side features the "Project Explorer" view, which displays a project named "Tema1" containing a "src" folder with a "T1Ej1" package and a file "T1Ej1.java". Below this is a "JRE System Library [jdk-17.0.1.jdk]". The main workspace shows the Java code for "T1Ej1.java":

```
1 package T1Ej1;
2
3 public class T1Ej1 {
4
5     public static void main(String[] args) {
6
7         for (int largo = 0; largo < 5; largo++) {
8             System.out.println();
9             System.out.print("*");
10            for (int i = 5; i > largo+1; i--) {
11                System.out.print(" *");
12            }
13        }
14    }
15 }
16
17 }
18 }
```

The code prints a diamond shape of asterisks. The bottom part of the interface shows the "Console" tab with the output:

```
* * * * *
* * *
* *
* *
*
```

Vemos que el espacio se divide en dos ventanas principales.

Ventana Izquierda: ventana de proyectos.

Aquí irá apareciendo la relación de proyectos, archivos, módulos o clases que vayamos abriendo durante la sesión.

Cada proyecto comprende una serie de archivos y bibliotecas que lo componen.

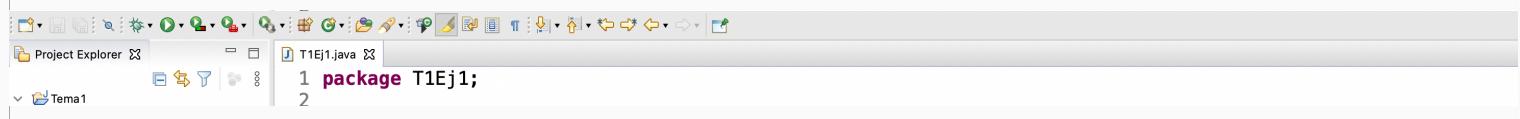
El principal archivo del proyecto Java es el que tenga la función main.

Ventana derecha: espacio de escritura de los códigos de los proyectos.

Aquí aparece el esqueleto propio de un programa escrito en lenguaje Java.

Dicho código mostrará un triángulo equilátero de lado 5 en base a asteriscos.

BARRA DE HERRAMIENTAS: Desde aquí podremos acceder a todas las opciones del IDE.



8.1.- Edición de programas.



Caso práctico

—Vamos a hacer el primer ejemplo —comenta Ana, entusiasmada—. Después de todo, no debemos perder de vista la finalidad de la herramienta, ESCRIBIR PROGRAMAS

En este sencillo ejemplo se ve una modificación de las líneas de código en la ventana de codificación del archivo **T1EJ1.java** del proyecto **Tema1** que acabamos de crear.

The screenshot shows the Eclipse IDE interface with the title bar "eclipseWS2 - Tema1/src/T1Ej1/T1Ej1.java - Eclipse IDE". The Project Explorer view on the left shows two projects: "Prueba1" and "Tema1". The code editor window displays the following Java code:

```
1 package T1Ej1;
2
3 public class T1Ej1 {
4
5     public static void main(String[] args) {
6
7         for (int largo = 0; largo < 5; largo++) {
8             System.out.println();
9             System.out.print("*");
10            for (int i = 5; i > largo+1; i--) {
11                System.out.print("*");
12            }
13        }
14    }
15
16
17 }
```

No hay que decir que la programación en Java no es objeto del presente módulo, pero puedes probar con algunos ejemplos en Java que tengas de otros módulos.

Mientras escribimos en el editor de textos nos percatamos de varias características de Eclipse que ya hemos señalado en páginas anteriores:

- Autocompletado de código.
- Coloración de comandos.
- Subrayado en rojo cuando hay algún error y posibilidad de depuración y corrección de forma visual, mediante un pequeño ícono que aparece a la izquierda de la línea defectuosa.

9.- Actualización y mantenimiento de entornos de desarrollo.



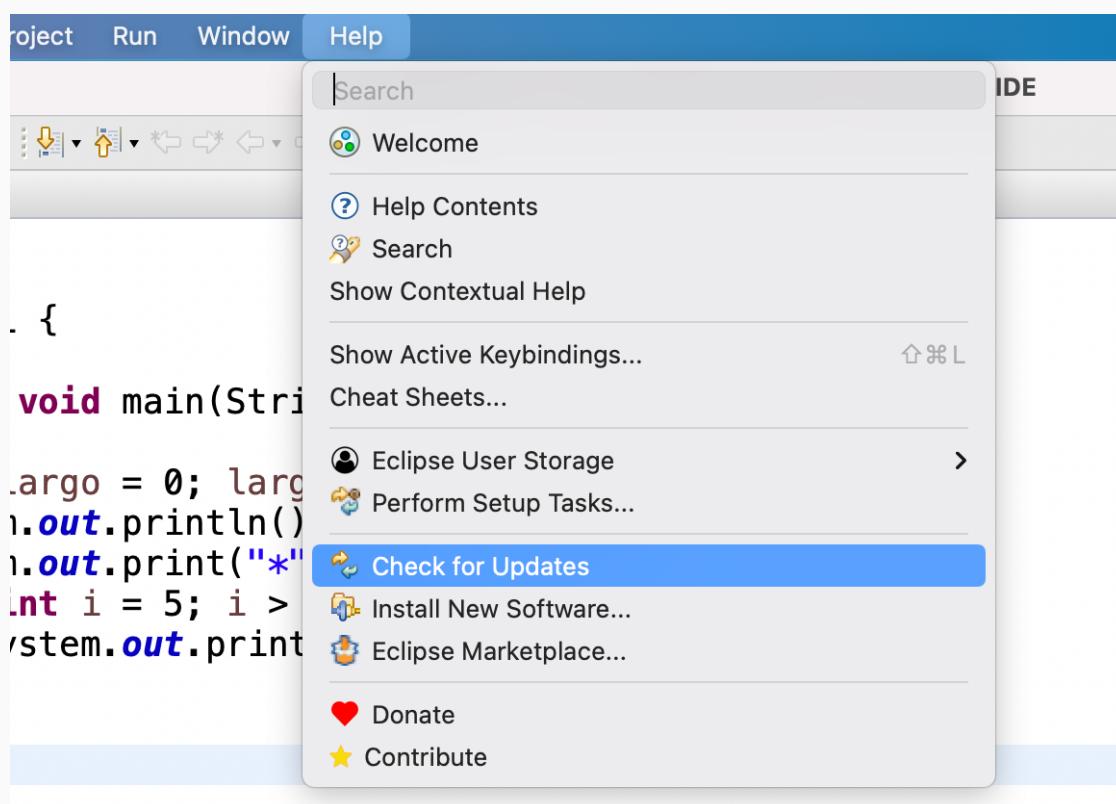
Caso práctico

—Por último, es de vital importancia el mantener y actualizar el entorno de desarrollo —comenta Ana—. Deberíamos tener permanentemente actualizados todos los complementos y realizar un correcto mantenimiento a las bases de datos asociadas a nuestros proyectos.

El mantenimiento del entorno de desarrollo es una tarea fundamental que requiere tener todos sus componentes periódicamente actualizados.

También es de vital importancia realizar copias de seguridad sobre las bases de datos de nuestros proyectos por si ocurriera algún error o proceso defectuoso poder restaurarlos.

El mantenimiento y las actualizaciones se hacen de forma on-line. En Eclipse podemos comprobar si existen actualizaciones de algún plug-in a través de Help > Check for Updates



Para añadir módulos y plugins on-line, hay que tener este complemento instalado en el entorno.

La gestión de las bases de datos asociadas a nuestros proyectos es muy importante. Habrá que realizarles copias de seguridad periódicamente, para asegurar su restauración en caso de fallos en el sistema, y mantenerlas actualizadas para su posible portabilidad futura a nuevas versiones del entorno que utilicemos.



Autoevaluación

¿Cuál es la razón, en tu opinión, de que salgan nuevas versiones de los entornos de desarrollo tan rápidamente?

- Para adaptarse a la evolución del hardware.
- Para incluir y modificar funcionalidades del entorno.

Incorrecto. Esto no es significativo en la evolución de los entornos de desarrollo.

Así es. Cada nueva versión tiene mejoras que permite aumentar la funcionalidad del entorno.

Solución

- 1.** Incorrecto (#answer-53_63)
- 2.** Opción correcta (#answer-53_297)

Anexo I.- Instalación JDK en Ubuntu 10.10.

PASOS:

1.

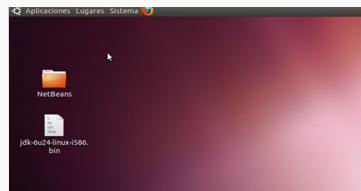
Descargar el JDK de la siguiente URL:

Descarga del JDK.

El archivo de JDK utilizado es: [jdk-6u24-linux-i586.bin](#)

2.

Guardar el archivo en el escritorio de Linux.



3.

Mover el archivo al directorio /usr/local

El movimiento del archivo a esta ruta sólo puede ser realizado por el root del sistema.

Para poder ejecutarlo como un usuario normal, basta poner el comando **sudo** antes de la orden. Esto implica que todas las operaciones a partir de este momento deberemos realizarlas desde la terminal del sistema operativo.

Para acceder a la **terminal**, pulsamos sobre la pestaña de:

Aplicaciones - Accesorios - Terminal

Las acciones a realizar serán las siguientes:

Entramos en el escritorio:

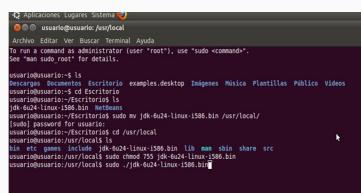
```
$ cd Escritorio  
$ sudo mv jdk-6u24-linux-i586.bin /usr/local
```

4.

Darle permiso de ejecución al archivo jdk y ejecutarlo

Entramos en la ruta:

```
$ cd /usr/local  
$ sudo chmod 755 jdk-6u24-linux-i586.bin  
$ sudo ./jdk-6u24-linux-i586.bin
```



Comienza la instalación...

```

Aplicaciones Lugares Sistema
usuario@usuario:~$ local
Archivo Editar Ver Buscar Terminal Ayuda
inflating: jdk1.6.0_24/db/lib/derbyLocale_zh_CN.jar
inflating: jdk1.6.0_24/db/lib/derbyLocale_zh_TW.jar
inflating: jdk1.6.0_24/db/lib/derbynet.jar
inflating: jdk1.6.0_24/db/lib/derbyrun.jar
inflating: jdk1.6.0_24/db/register.jar
inflating: jdk1.6.0_24/jre/register.html
creating jdk1.6.0_24/jre/lib/rt.jar
Creating jdk1.6.0_24/jre/lib/jse.jar
Creating jdk1.6.0_24/jre/lib/charsets.jar
Creating jdk1.6.0_24/jre/lib/ext/locatedata.jar
Creating jdk1.6.0_24/jre/lib/plugin.jar
Creating jdk1.6.0_24/jre/lib/javaws.jar
Creating jdk1.6.0_24/jre/lib/deploy.jar
Java(TM) SE Development Kit 6 successfully installed.

Product Registration is FREE and includes many benefits:
• Notification of new versions, patches, and updates
• Special offers on Oracle products, services and training
• Access to early releases and documentation

Product and system data will be collected. If your configuration
supports a browser, the JDK Product Registration form will

```

```

Java(TM) SE Development Kit 6 successfully installed.

Product Registration is FREE and includes many benefits:
• Notification of new versions, patches, and updates
• Special offers on Oracle products, services and training
• Access to early releases and documentation

Product and system data will be collected. If your configuration
supports a browser, the JDK Product Registration form will
be presented. If you do not register, none of this information
will be saved. You may also register your JDK later by
opening the register.html file (located in the JDK installation
directory) in a browser.

For more information on what data Registration collects and
how it is managed and used, see:
http://java.sun.com/javase/registration/JDKRegistrationPrivacy.html

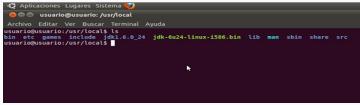
Press Enter to continue.....

Done.
usuario@usuario:~$ local

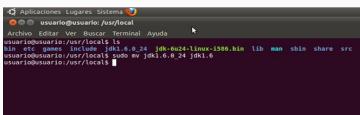
```

5.

Renombramos la carpeta que se ha creado durante la instalación del archivo.



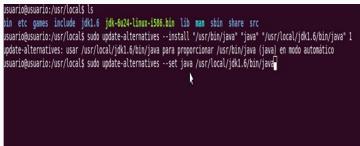
```
$ sudo mv jdk1.6.0_24 jdk1.6
```



6.

Ejecutamos los siguientes comandos:

```
$ sudo update-alternatives --install "/usr/bin/java" "java" "/usr/local/jdk1.6/bin/java" 1
$ sudo update-alternatives --set java /usr/local/jdk1.6/bin/java
```



7.

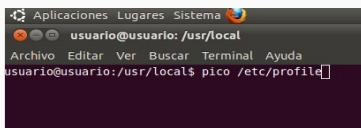
Editamos el archivo /etc/profile y agregamos las siguiente líneas al final del mismo:

```
export JAVA_HOME=/usr/local/jdk1.6
export PATH=$JAVA_HOME/bin:$PATH
```

Para editar el archivo podemos usar el comando:

```
$ pico /etc/profile
```

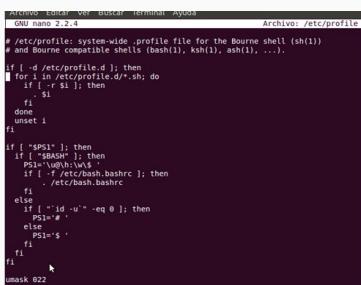
como se ve en la imagen:



o utilizar el comando:

```
$ nano /etc/profile
```

Cualquiera de los dos editores de texto (pico o nano) pueden ser usados en Linux. Despues de teclear cualquiera de los dos comandos anteriores nos aparece la siguiente ventana:

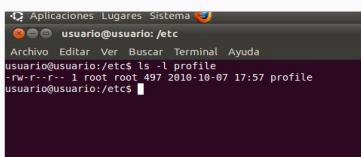


Nos colocamos al final del archivo y escribimos esas dos líneas:



Guardamos el archivo y nos dice que no tenemos permisos para modificarlo.

Por tanto, tenemos que darle a /etc/profile permiso de modificación:



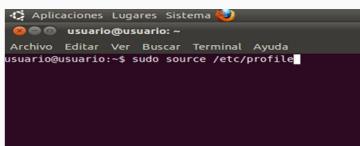
Ya sí podemos modificar el archivo agregándole las dos líneas al final del mismo (Repetir el paso de antes y guardar el archivo)

8.

Salimos de la terminal, tecleando el comando exit, y volvemos a entrar en ella.

Teclear lo siguiente:

```
$ sudo source /etc/profile
```



9.

Probar el funcionamiento de Java.

Teclear el siguiente comando:

```
$ java -version
```

