

UNIDAD DIDÁCTICA 2

ENTORNOS DE DESARROLLO

**MÓDULO PROFESIONAL:
ENTORNOS DE DESARROLLO**



CESUR
Tu Centro Oficial de FP

Índice

RESUMEN INTRODUCTORIO	2
INTRODUCCIÓN	2
CASO INTRODUCTORIO	3
1. ENTORNOS DE DESARROLLO	4
1.1 Herramientas CASE	4
1.1.1 Objetivo	5
1.1.2 Estructura	6
1.1.3 Repositorio	7
1.1.4 Clasificación	8
1.2 Entornos de Programación	11
1.2.1 Entornos profesionales más usados	14
1.3 Ejemplos de Herramientas CASE	16
1.3.1 Microsoft Project	16
1.3.2 Oracle JDeveloper	17
2. INSTALACIÓN DE ECLIPSE	19
RESUMEN FINAL	40

RESUMEN INTRODUCTORIO

En esta unidad vamos a estudiar conceptos relacionados con entornos de desarrollo y herramientas CASE. Es necesario para poder desarrollar con agilidad y fluidez controlar las herramientas CASE, ya que harán que el trabajo esté más organizado y permite la colaboración entre distintos miembros del equipo. También vamos a conocer un entorno de desarrollo completo, como puede ser Eclipse. A la hora de desarrollar se pueden usar distintos entornos de desarrollo, ya que al conocer un entorno y todas las posibilidades que tiene, el resto de entornos son bastante parecidos. Cómo se ha comentado trabajarás con Eclipse, aunque recomendamos que también por vuestra cuenta consultéis sobre Visual Studio Code e IntelliJ.

INTRODUCCIÓN

Como has estudiado en la unidad anterior, el proceso para desarrollar software está dividido en fases. Las fases están relacionadas entre sí, incluso como vimos, la fase de documentación es una fase transversal. Estas distintas fases conllevan una sincronización casi perfecta y debemos automatizar todo lo posible para que tampoco sea un trabajo arduo. Por eso, como profesionales en entornos de desarrollo necesitamos conocer herramientas que nos ayuden a automatizar el proyecto de desarrollo del software para nuestro día a día.

Además, las metodologías ágiles son muy utilizadas en la actualidad, por lo que es importante tener unas nociones básicas. Este tipo de metodologías son aplicables tanto al proceso de desarrollo software como en empresas u organizaciones de otro sector. Entre ellas se destaca la metodología SCRUM, cuyo fundamento se basa en llevar a cabo reuniones diarias entre los miembros del equipo y realizar entregables al cliente de forma periódica. Esta metodología de trabajo permite un mayor control sobre el producto a entregar y una relación más cercana entre cliente y equipo de trabajo. Las reuniones son muy útiles para solucionar errores que estén ocurriendo y, de esta forma, no alargar estos problemas en el tiempo.

CASO INTRODUCTORIO

Desde la empresa INGEN TECHNOLOGIES CONSULTING, tú y tu equipo de analistas habéis establecido una reunión con un cliente que tiene varias tiendas de ropa. El objetivo es obtener los datos necesarios para poder realizar una aplicación. Una vez recopilados todos los requisitos y analizadas todas las características de la aplicación software, os disponéis a ejecutar la fase de diseño. En ella, empezáis a hacer uso de las herramientas CASE instaladas en su equipo: Umbrello para el diseño de diagramas UML, Git para el control de versiones de los ficheros de código y los correspondientes a la documentación, y MySQL Workbench, para diseñar la base de datos. Además, el equipo de programadores pone a punto sus entornos de desarrollo, mediante los cuales implementarán todo el código de la aplicación. Al ser un software creado en lenguaje Java, se acuerda configurar Eclipse con la última versión de JDK disponible. Con todas estas herramientas, y con tu gestión para el reparto de tareas entre los miembros de tu equipo, se procede al desarrollo de las fases del software.

Al finalizar la unidad conocerás las principales herramientas CASE del mercado para automatizar las fases del ciclo de vida del software, también algunos de los entornos de desarrollo que más se usan, la instalación, actualización, configuración de Eclipse, y la creación de ejecutables.

1. ENTORNOS DE DESARROLLO

Para poder llevar a cabo la aplicación para la gestión de las distintas tiendas de ropa para el cliente, tienes que elegir las herramientas CASE que se van a utilizar para automatizar lo máximo posible las fases de desarrollo del software. Y, en este momento, te dispones a ello junto con tu equipo de trabajo.

Los **entornos de desarrollo** están formados por un conjunto de software que facilitan o automatizan las actividades de desarrollo. Lo ideal sería poder automatizar todo el proceso de desarrollo de una aplicación desde el principio (fase de análisis) hasta el final (explotación y mantenimiento) pero normalmente sólo se automatizan las fases de implementación y las pruebas del software. Las llamadas actividades verticales del ciclo de vida son las específicas de una fase de la vida del software, por ejemplo, la etapa de análisis o diseño. Por su parte, las actividades horizontales son actividades que tienen lugar en cualquiera de las fases del ciclo de vida del software, por ejemplo, la documentación. Los entornos de desarrollo deben dar soporte tanto a las actividades verticales como horizontales.

1.1 Herramientas CASE

Las siglas CASE provienen del inglés “Computer Aided Software Engineering”, que traducido al inglés podría entenderse como Ingeniería del Software Asistida por Ordenador. El uso de estas herramientas empezó a popularizarse a partir de los buenos resultados obtenidos a través de herramientas informáticas de ayuda al diseño (CAD) y a la fabricación (CAM).

Como se estudiará más adelante, las herramientas CASE pueden cubrir una o varias de las actividades de ingeniería del software. Por ejemplo, se puede acudir a una herramienta CASE para representar el modelo Entidad Relación de una base de datos, y extraer automáticamente de dicho modelo la documentación correspondiente al **diccionario de datos**.

Igual ocurre con otras áreas, como son la estimación de recursos y costes, la planificación, el diseño de datos y control, etc. De hecho, las últimas tendencias en herramientas CASE apuntan hacia la integración de diferentes sub-herramientas (correspondientes a distintas actividades) que compartan y aprovechen los resultados que cada una va obteniendo. Estas herramientas se las conocen como ‘herramientas CASE integradas’.



PARA SABER MÁS

Para ampliar información sobre las herramientas CASE, su historia, clasificación y ejemplos, visita esta la web:



1.1.1 Objetivo

Para mejorar la calidad y la productividad de los sistemas de información a la hora de construir software se plantean los siguientes objetivos:

- Permitir la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta se consigue agilizar el trabajo.

- Facilitar la realización de prototipos y el desarrollo conjunto de aplicaciones.

- Simplificar el mantenimiento de los programas.

- Mejorar y estandarizar la documentación.

- Aumentar la portabilidad de las aplicaciones.

- Facilitar la reutilización de componentes software.

- Permitir un desarrollo y refinamiento visual de las aplicaciones, mediante la utilización de gráficos.

Objetivos a la hora de construir un software.

1.1.2 Estructura

De una forma esquemática se puede afirmar que una herramienta CASE se compone de los siguientes elementos:

- **Repositorio** (diccionario) donde se almacenan los elementos definidos o creados por la herramienta, y cuya gestión se realiza mediante el apoyo de un Sistema de Gestión de Base de Datos (SGBD) o de un sistema de gestión de ficheros.

- **Metamodelo** (no siempre visible), que constituye el marco para la definición de las técnicas y metodologías soportadas por la herramienta.

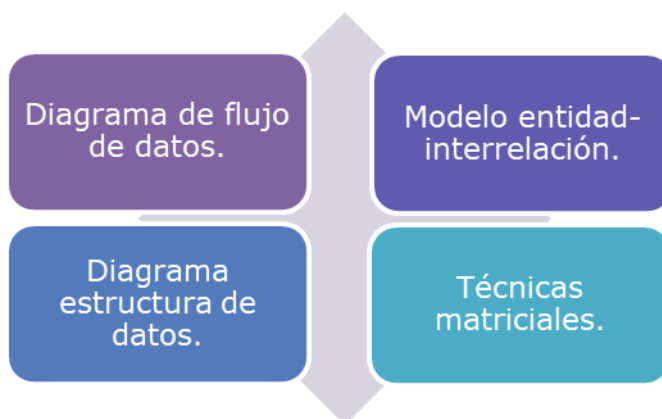
- **Carga o descarga de datos**, son facilidades que permiten cargar el repertorio de la herramienta CASE con datos provenientes de otros sistemas, o bien generar a partir de la propia herramienta esquemas de base de datos, programas, etc. que pueden, a su vez, alimentar otros sistemas. Este elemento proporciona así un medio de comunicación con otras herramientas.

- **Comprobación de errores**, facilidades que permiten llevar a cabo un análisis de la exactitud, integridad y consistencia de los esquemas generados por la herramienta.

- **Interfaz de usuario**, que constará de editores de texto y herramientas de diseño gráfico que permitan, mediante la utilización de un sistema de ventanas, iconos y menús y con la ayuda del ratón, definir los diagramas, matrices, etc. que incluyen las distintas metodologías. Los módulos de diagramación y modelización se encargan de proporcionar las utilidades para diseñar las interfaces de usuario.

Elementos de una herramienta CASE.

Algunos de los diagramas y modelos utilizados con mayor frecuencia son:



Diagramas y modelos realizados con herramientas CASE.



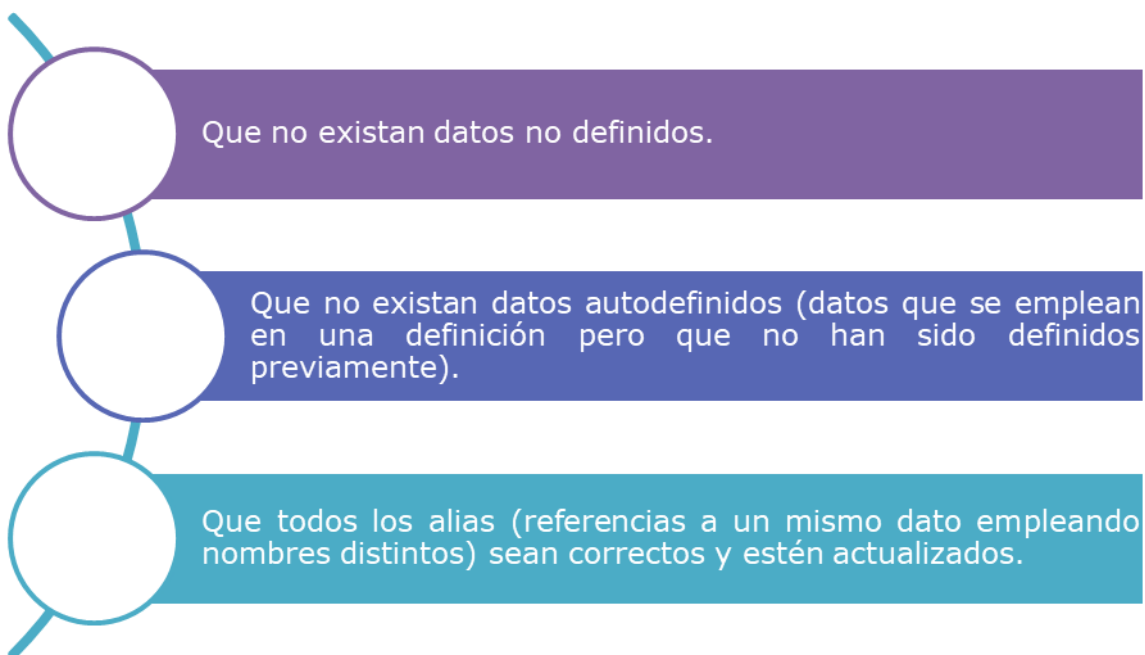
PARA SABER MÁS

Conoce más información sobre el diagrama de flujo de datos y ver más ejemplos:



1.1.3 Repositorio

Se conoce como **repositorio** a la base de datos central de una herramienta CASE. El repositorio amplía el concepto de diccionario de datos para incluir toda la información que se va generando a lo largo del ciclo de vida del sistema, como, por ejemplo: componentes de análisis y diseño (diagramas de flujo de datos, diagramas entidad-relación, esquemas de bases de datos, diseños de pantallas), estructuras de programas, algoritmos, etc. En algunas referencias se le denomina Diccionario de Recursos de Información. Apoyándose en la existencia del repositorio, se efectúan comprobaciones de integridad y consistencia:



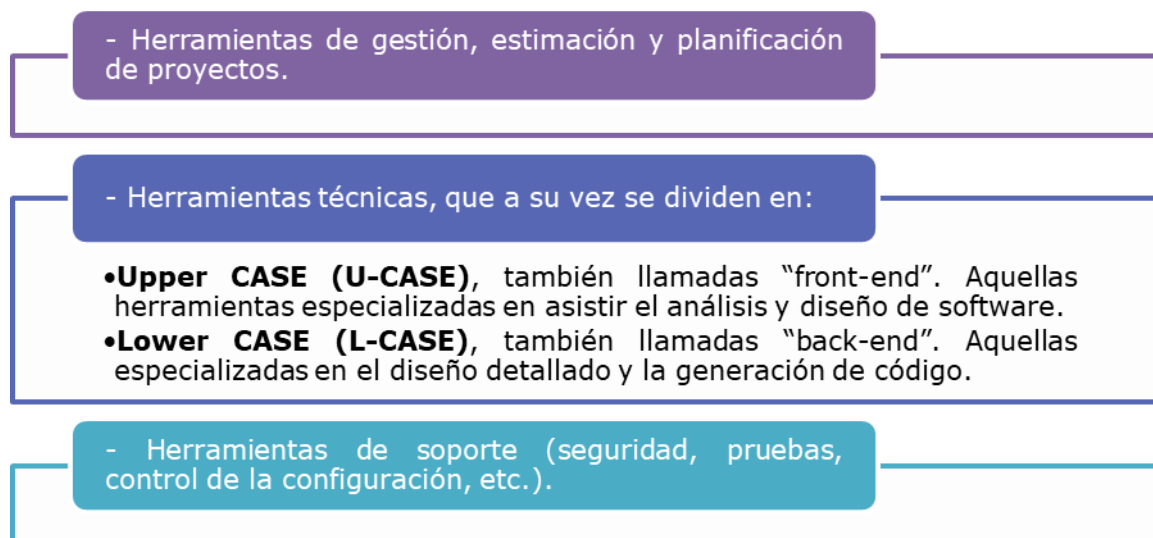
Comprobaciones de integridad y consistencia.

Las características más importantes de un repositorio son:

- **Tipo de información que contiene:** metodología concreta, datos, gráficos, procesos, informes, modelos o reglas, etc.
- **Tipo de controles:** Si incorpora algún módulo de gestión de cambios, de mantenimiento de versiones, de acceso por clave, de redundancia de la información. La gestión de cambios y el mantenimiento de versiones ayudarán en el caso de que convivan diferentes versiones de la misma aplicación o se tengan que realizar simultáneamente cambios en la versión de producción y de desarrollo.
- **Tipo de actualización:** Si los cambios en los elementos de análisis o diseño se ven reflejados en el repositorio en tiempo real o mediante un proceso por lotes (batch). Esto será importante en función a la necesidad de que los cambios sean visibles en el momento por todos los usuarios.
- **Reutilización de módulos para otros diseños:** El repositorio es la clave para identificar, localizar y extraer distintos elementos para su reutilización.
- **Posibilidad de exportación e importación** para extraer información del repositorio y tratarla con otra herramienta (formateo de documentos, mejora de presentación) o incorporar al repositorio, información generada por otros medios.
- **Interfaces automáticas** con otros repositorios o bases de datos externas.

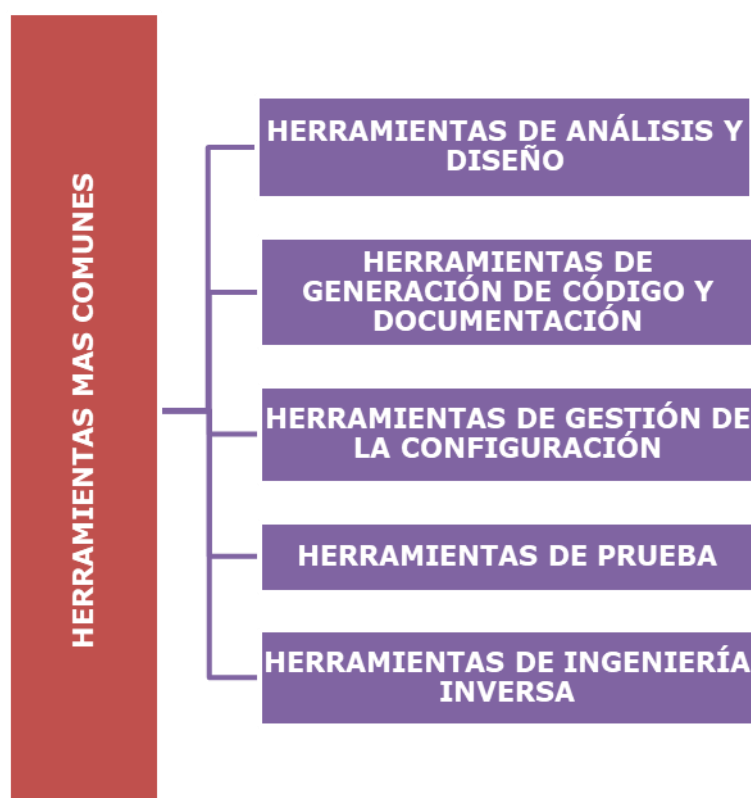
1.1.4 Clasificación

Generalmente se han clasificado las herramientas CASE atendiendo a la actividad o fase de la ingeniería del software a la que asisten o automatizan. En ocasiones es difícil asignar una herramienta comercial concreta a una categoría, ya que puede cubrir varias actividades distintas.



Clasificación de las herramientas CASE.

A continuación, se analizan con detalle los tipos más comunes de herramientas CASE.



Tipos más comunes de herramientas CASE.

Herramientas de Análisis y Diseño: Las herramientas de análisis y diseño son, de entre todas las categorías, las de mayor difusión en la actualidad. Su principal objetivo es ayudar a la definición de requisitos del sistema y sus propiedades. Dentro de esta categoría destacan las herramientas que asisten las fases de análisis y diseño facilitando la aplicación de una determinada metodología. Así, estas herramientas permiten editar

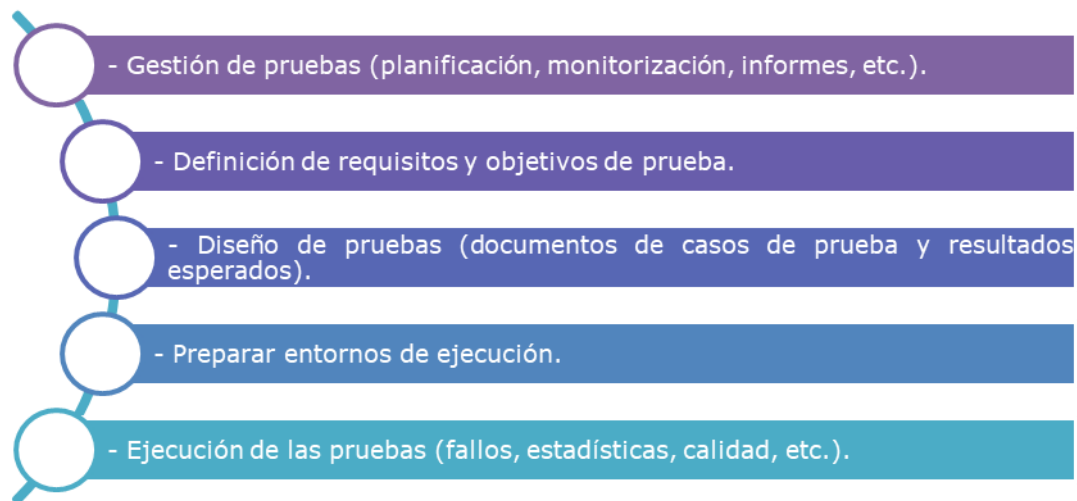
diagramas E/R, diagramas de flujo de datos, diagramas de clases, etc. También son muy importantes las herramientas de prototipado, como los diseñadores de pantallas, generadores de menús, generadores de informes, etc.

Herramientas de Generación de código y documentación: A partir de las especificaciones de diseño se puede generar código, tanto programas (por ejemplo, en lenguaje C o Java) como esquemas de base de datos (sentencias SQL de definición). Actualmente, las herramientas CASE ofrecen interfaces con diversos lenguajes de cuarta generación para la construcción de sistemas de forma rápida. Las herramientas CASE también soportan la creación automatizada de un conjunto muy variado de documentación, que va desde la descripción textual de un pseudocódigo hasta diagramas más o menos complejos.

Herramientas de Gestión de la Configuración: En entornos de desarrollo complejos, especialmente si se integran diversas herramientas de ingeniería de software, se hace imprescindible la incorporación de una herramienta capaz de gestionar la configuración de los sistemas. Este tipo de herramientas ofrecen distintas capacidades:

- **Control de versiones:** Es decir, capacidad de proporcionar almacenamiento y acceso controlado a los datos por parte de uno más usuarios, así como registrar los cambios sobre los mismos, y poder recuperar versiones anteriores.
- **Trazabilidad de requisitos:** Permiten que un requisito pueda ser rastreado hasta su implementación.
- **Análisis de impacto:** Permite conocer los elementos del sistema que se ven afectados ante un cambio.

Herramientas de Prueba: Las herramientas de prueba, o CAST (Computer Aided Software Testing), son desarrollos especialmente recientes dentro de la tecnología CASE. Algunas funcionalidades que suelen ofrecer este tipo de herramientas son las siguientes:



Funcionalidades de las herramientas CAST.

Herramientas de Ingeniería Inversa: El objetivo de la ingeniería inversa es obtener información a partir de un producto, con el fin de determinar cómo fue fabricado. El método se denomina así porque avanza en dirección opuesta a las tareas habituales de ingeniería. Dentro de este apartado destacan diversas herramientas, que llevan a cabo las siguientes tareas:

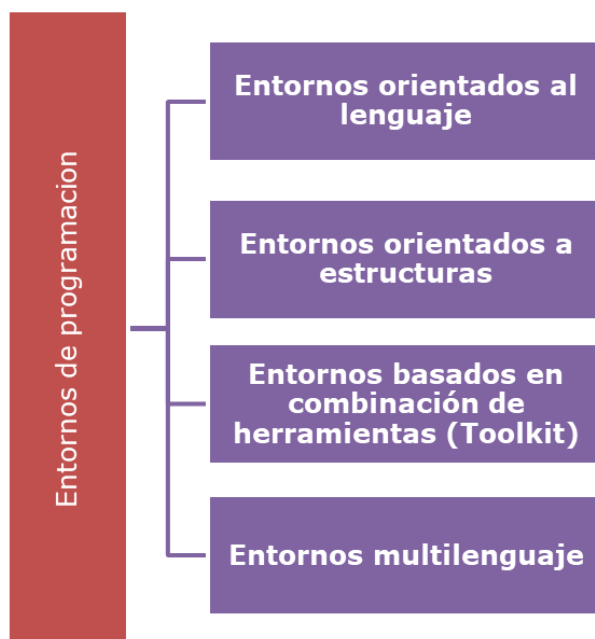
- **Ingeniería inversa de datos**, capaces de extraer información del código fuente que describe la estructura de los datos, por ejemplo, construyendo diagramas E/R partiendo de un modelo de datos ya implementado.
- **Ingeniería inversa de procesos**, a partir del código permiten aislar la lógica de las entidades y las reglas de negocio.
- **Reestructuración de código fuente**, modifican su formato o implantan un formato estándar.
- **Análisis de código**, cuyas funcionalidades van desde la tabulación automática del código fuente, hasta la posibilidad de ir visualizando dinámicamente las llamadas del mismo.

1.2 Entornos de Programación

Son las herramientas CASE que se encargan de dar soporte al desarrollador para la creación del código de los programas que se van a implementar. Juegan un papel importante en las fases de diseño y codificación del ciclo de vida del software, siendo útil en las fases de pruebas y explotación y mantenimiento. En los entornos de programación se tendrá soporte para la codificación, edición del código fuente, ejecución del software, compilación, creación de ejecutables, etc.

También se incluyen herramientas muy útiles como la búsqueda, el reemplazo, control de versiones, generación automática de pruebas, ejecución en modo depuración, etc.

Un entorno de desarrollo estará como mínimo formado por un editor de texto orientado al lenguaje de programación en el que se desee crear el programa, además de contener un depurador. Un entorno de programación u otro puede estar organizado de maneras muy diferentes. Los entornos de programación se pueden clasificar según el siguiente esquema:



Clasificación de los entornos de programación.

- **Entornos orientados al lenguaje**

Son entornos específicos de un lenguaje en particular, fuertemente integrados como una única herramienta. Su editor de código fuente está muy orientado al lenguaje de modo que cualquier fallo en alguna de las palabras reservadas del lenguaje se mostrará como un error.

Debido a esto son muy poco flexibles en la integración o interoperación con otros productos, la principal ventaja es que son muy fáciles de usar. Ejemplos: DEV-C++, VISUAL C++, DELPHI.

- **Entornos orientados a estructuras**

También son orientados al lenguaje, por lo que tienen todas las ventajas e inconvenientes de estos. Además del editor de código fuente posee un editor de estructura o editor sintáctico, basado en representar internamente el código fuente de la estructura que se está implementando o editando, para crear, editar código en lenguaje XML, HTML, por ejemplo.

- **Entornos basados en combinación de herramientas (Toolkit)**

Presentan una integración débil, fáciles de ampliar y adaptar a nuevas herramientas. Presentan al desarrollador un conjunto de herramientas de modo que son capaces de interoperar entre ellas. Ejemplos de este tipo de entornos: EMAX, GUIM o VIM, entre otros.

- **Entornos multilenguaje**

Están a disposición de desarrolladores que necesitan crear aplicaciones que combinan código fuente en distintos lenguajes de programación. Algunas posibilidades de combinación son:

Entornos genéricos:

- No combina lenguajes en un mismo programa, sino que hay varios programas cada uno con su lenguaje de programación, siendo tarea del desarrollador combinar las correspondientes herramientas.

Entornos específicos:

- Son como los entornos orientados al lenguaje pero admiten más de un lenguaje de programación, la integración está fuertemente controlada. Ejemplos: Con GPS, se puede combinar ADA y C++.

Lenguajes ejecutables sobre máquina virtual:

- En este lenguaje, es la máquina la que establece el formato del código binario; pueden combinarse módulos escritos en diferentes lenguajes para los que exista el compilador apropiado. Pueden combinarse los diferentes entornos de programación o tener uno específico para todos ellos. Ejemplo: Eclipse.

Entornos multilenguaje.

1.2.1 Entornos profesionales más usados

Los entornos profesionales más usados son:

Eclipse

Eclipse es un entorno de desarrollo que agrupa un conjunto de herramientas que se usan para la programación en multiplataforma. Eclipse puede utilizarse principalmente con el lenguaje de programación JAVA, aunque la plataforma ha ido evolucionando y es un entorno de desarrollo multilenguaje.

Eclipse tiene una comunidad de usuarios bastante extendida, lo que hace que se pueda encontrar ayuda y manuales con facilidad.

Fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Ahora mismo este IDE (entorno de desarrollo integrado) es desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Este IDE permite una actualizaciones e instalación de distintos plug-in que hace que sea un entorno de desarrollo muy completo.



Eclipse.

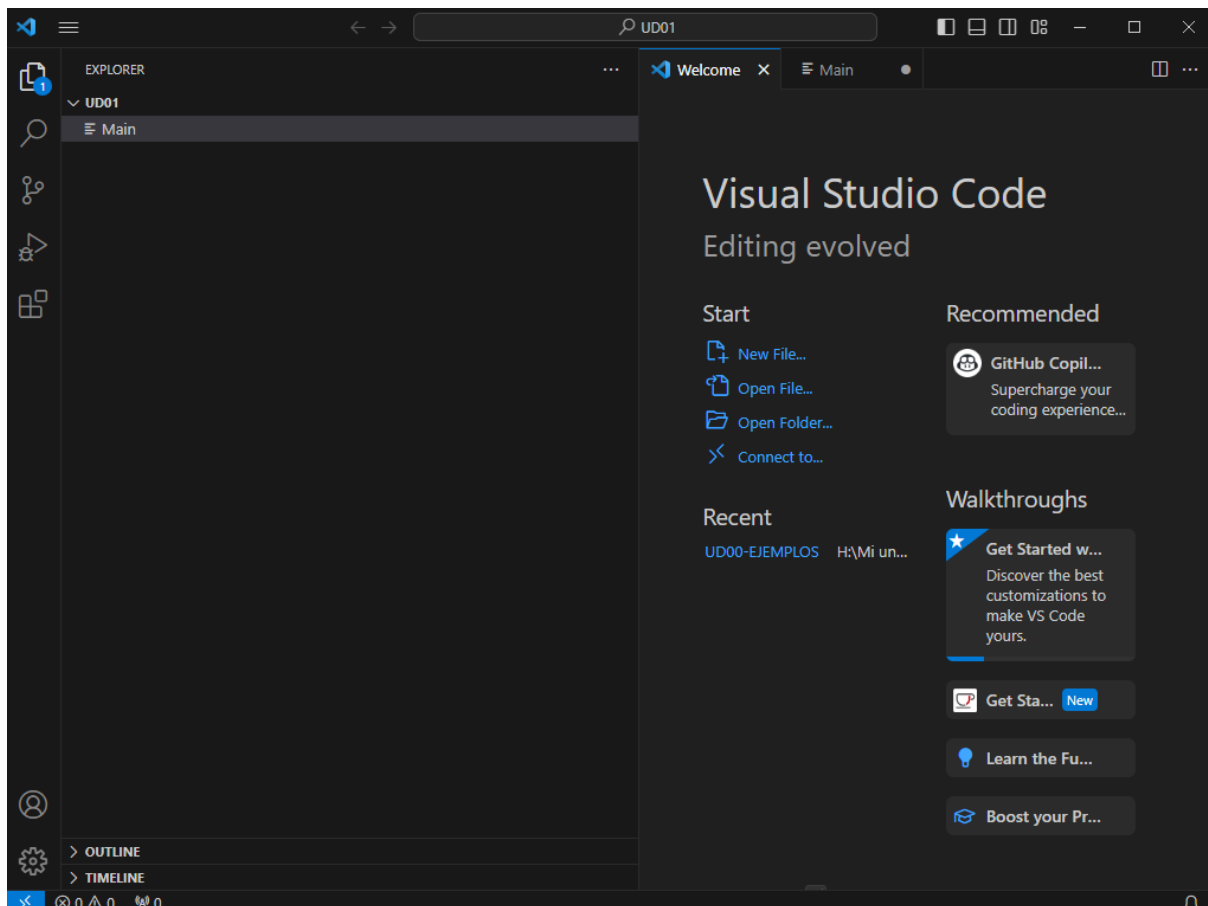
Fuente: <https://www.eclipse.org/>

Visual Studio Code

Se puede definir Visual Studio Code como un editor de texto enriquecido parecido a Sublime Text, Atom, Notepad++.

VS Code trae de soporte integrado para JavaScript, TypeScript y Node JS, pero puedes usarlo para codificar otro tipo de lenguajes de programación, ya que se permite descargar una serie de extensiones que equipan al entorno de las herramientas necesarias para los distintos lenguajes. Estas extensiones pueden estar hechas por Microsoft, que el propietario de VS Code o por externos.

Con VS Code además se obtiene un depurador y detección inteligente de errores.



Visual Studio Code.

Fuente: Propia.

IntelliJ

IntelliJ IDEA es uno de los editores de código más potentes del mercado, permite detectar errores sobre la marcha, sugerir opciones de finalización de código, realizar refactorización. Los lenguajes de programación predefinidos para este IDE son JAVA y KOTLIN, de hecho, Android Studio es una especialización de IntelliJ IDEA que el propio Google, le pidió a la empresa creadora de este IDE (JetBrains).

IntelliJ, permite además trabajar con código en otros lenguajes incrustado en nuestro código.



IntelliJ.

Fuente: <https://www.jetbrains.com/es-es/>

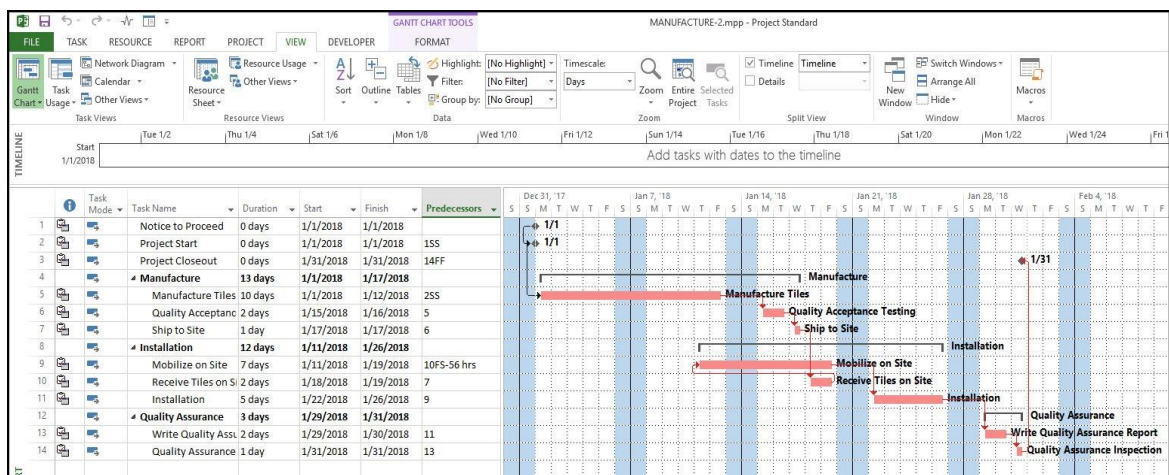
1.3 Ejemplos de Herramientas CASE

Como se ha estudiado, existe una gran variedad de herramientas CASE con características muy específicas. A continuación, se describen algunas de ellas:

1.3.1 Microsoft Project

Microsoft Project es un software de administración de proyectos para asistir a administradores de proyectos en el desarrollo de planificaciones, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

La aplicación puede crear un calendario de rutas críticas, y reconocer diferentes clases de usuarios, los cuales pueden contar con distintos niveles de acceso a proyectos, vistas y otros datos. Los objetos personalizables como calendarios, vistas, tablas, filtros y campos son almacenados en un servidor que comparte la información a todos los usuarios.



Interfaz de Microsoft Project 2019.

1.3.2 Oracle JDeveloper

Es un entorno integrado que cubre todo el ciclo de vida de desarrollo de software e integra distintas tecnologías utilizadas para la construcción de aplicaciones empresariales estándar. Sus principales características son:

- **Entorno de desarrollo integrado:** Integra en una única herramienta el desarrollo en Java, SOA, Web 2.0, bases de datos, XML y servicios Web, incluyendo la posibilidad de compartir elementos de la aplicación.
- **Soporte al ciclo de vida completo:** JDeveloper cubre el ciclo de vida completo de una aplicación. Los desarrolladores pueden diseñar, generar y visualizar su código con diagramas UML, Java y de base de datos. El entorno de codificación y los editores declarativos y visuales facilitan el desarrollo. Las pruebas integradas y las auditorías de código aseguran la calidad de la aplicación.
- **Framework de desarrollo:** Oracle JDeveloper es el entorno de desarrollo para Oracle ADF (Oracle Application Development Framework), un entorno de trabajo completo que proporciona la infraestructura necesaria para las distintas capas de una aplicación (acceso a datos, desarrollo de servicios de negocio, control de la aplicación, interfaz gráfica, seguridad, etc.).



ENLACE DE INTERÉS

Visita la web oficial de Oracle Developer para ampliar información:



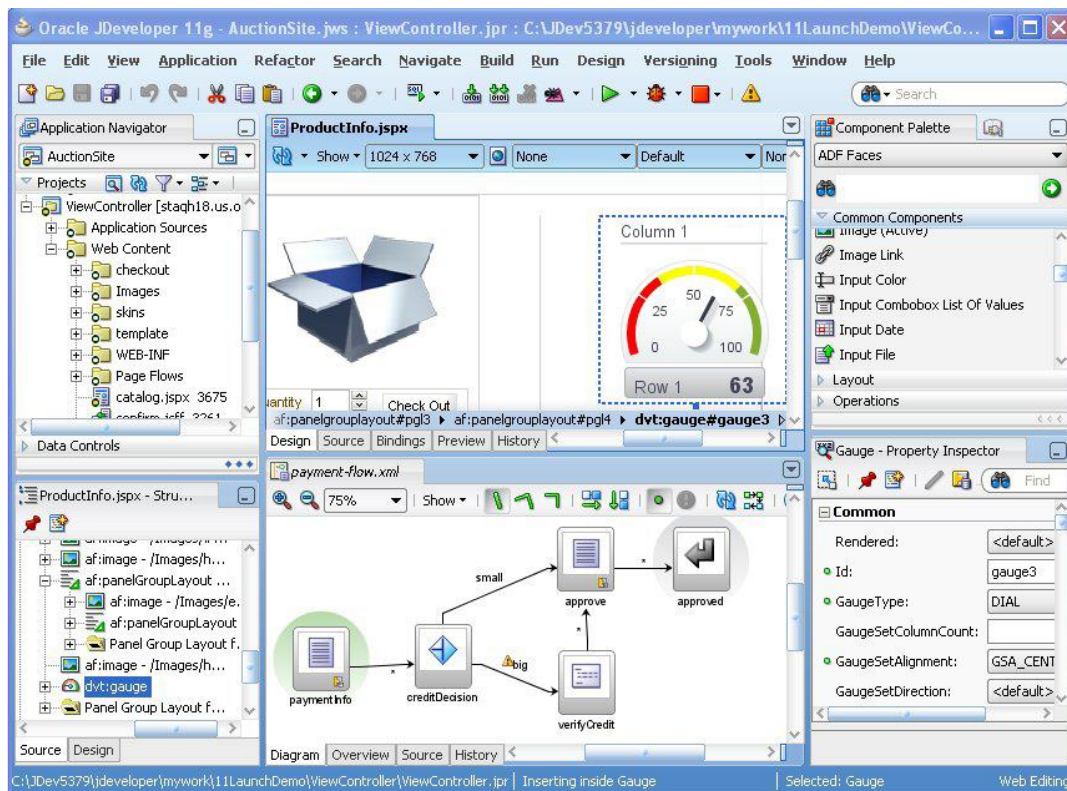


Imagen: Interfaz de Oracle JDeveloper.



EJEMPLO PRÁCTICO

Un jefe de proyecto debe llevar un control total sobre el trabajo de su personal. Como tal, es su labor realizar una gestión del proyecto atendiendo a varios criterios: número de miembros del equipo, tiempos, retrasos, recursos materiales, etc. ¿Qué tipo de herramienta CASE será necesario utilizar por un jefe de proyecto para controlar todos estos elementos?

Una herramienta CASE como Microsoft Project o GanttProject, que permite crear una situación actual en base a todos los elementos que intervienen en el proyecto y estimar tiempo y esfuerzo.



VÍDEO DE INTERÉS

En este vídeo se muestra un ejemplo de utilización de Microsoft Project, atendiendo a: Inicio del proyecto, Vinculación de tareas de forma simple, diagrama de Gantt y cálculo de ruta crítica.



2. INSTALACIÓN DE ECLIPSE

En la fase de diseño de la aplicación para la gestión de tiendas de ropa, has decidido instalar Eclipse como entorno de desarrollo, también utilizar el último JDK, que viene en la instalación predeterminada de Eclipse. Se planteo utilizar IntelliJ, ya que la mayoría de programadores están de acuerdo en que es el mejor entorno que se puede usar para programar en JAVA, pero la inexperiencia de nuestro equipo hace que nos decantemos con un entorno más sencillo y fácil de entender como puede ser Eclipse.

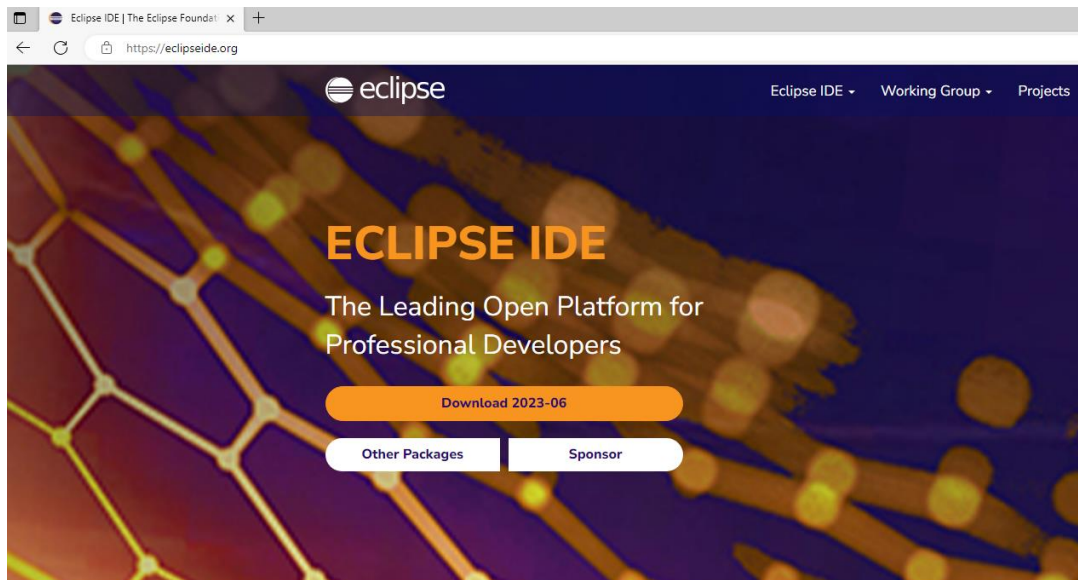
A la hora de elegir un entorno de desarrollo es importante que ayude a programar y haga más fácil la tarea del programador, en este caso se va a instalar la última versión de Eclipse, un IDE principalmente orientado a JAVA, aunque puede ser utilizado por otros lenguajes.

Otras alternativas que se pueden plantear son Visual Studio Code, o IntelliJ.

Este IDE se ha elegido en consonancia con el módulo de programación, además se instalará automáticamente el Kit de desarrollo de Java (JDK – Java Development Kit) necesario para tener las librerías básicas para poder programar en Java, y la Máquina Virtual de Java (JVM – Java Virtual Machine) necesaria para poder interpretar el código que se ha escrito en Java.

A continuación, se pasa a detallar el proceso de instalación de dicho programa:

1. Accedemos a la página de Eclipse, dónde siempre estará la última versión del programa.



BETTER THAN EVER

THE ECLIPSE IDE DELIVERS WHAT YOU NEED TO RAPIDLY INNOVATE

Web de descarga de instalador de Eclipse.

Fuente: www.eclipse.org



ENLACE DE INTERÉS

Descarga la última versión de Eclipse desde la web oficial:



2. Nos aparece el sitio de descarga, podemos elegir entre los diferentes sistemas operativos que hay ahora mismo en el mercado, macOS, Windows, Linux.

The screenshot shows the Eclipse Foundation website. The header includes the Eclipse Foundation logo and navigation links for 'Projects' and 'Working Groups'. The breadcrumb trail is 'Home / Downloads / Packages / Eclipse Installer 2023-06 R'. Below this, there are tabs for 'Eclipse Installer', 'Eclipse Packages', and 'Eclipse Developer Builds'. The main heading is 'Eclipse Installer 2023-06 R'. A dark blue banner contains the text: 'The Eclipse Installer 2023-06 R now includes a JRE for macOS, Windows and Linux.' Below this, there are two columns. The left column is titled 'Try the Eclipse Installer 2023-06 R' and describes it as 'The easiest way to install and update your Eclipse Development Environment.' It also shows download statistics: '1,318,169 Installer Downloads' and '1,290,362 Package Downloads and Updates'. The right column is titled 'Download' and lists the available operating systems and architectures: 'macOS x86_64 | AArch64', 'Windows x86_64', and 'Linux x86_64 | AArch64'. Below the banner, there is a section titled '5 Steps to Install Eclipse' which explains that the Eclipse Installer is a new and more efficient way to install Eclipse, providing a self-extracting download that leads through the installation process. It also mentions that for those who prefer not to use the Installer, packages and zip files are still available on the 'package download' page.

Web de descarga de instalador de Eclipse.

Fuente: www.eclipse.org

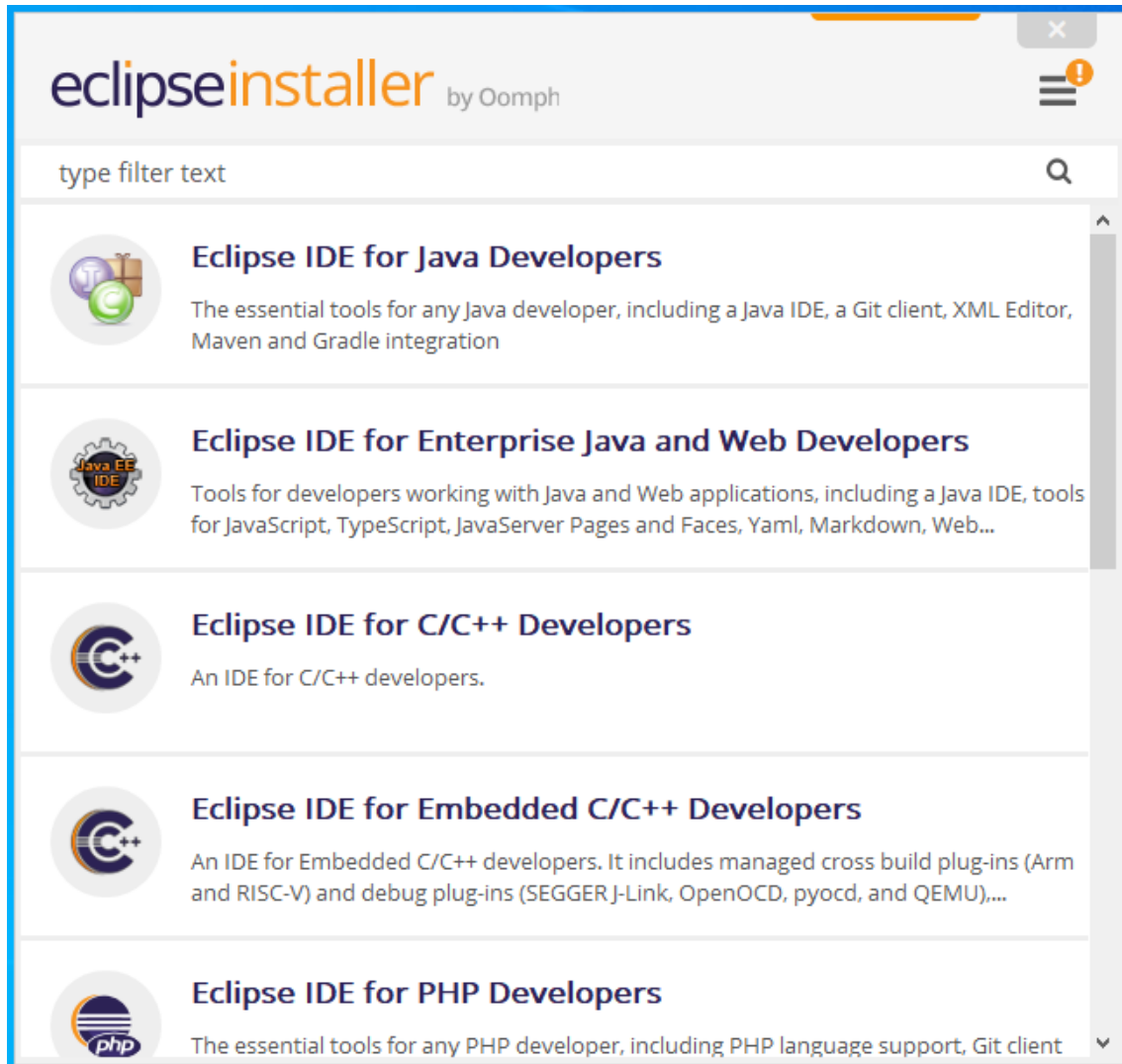
3. Le damos al botón de descarga

The screenshot shows the Eclipse Foundation website. The header includes the Eclipse Foundation logo and navigation links for 'Projects' and 'Working Groups'. The breadcrumb trail is 'Home / Downloads / Eclipse downloads - Select a mirror'. Below this, there is a disclaimer: 'All downloads are provided under the terms and conditions of the Eclipse Foundation Software User Agreement unless otherwise specified.' In the center, there is a large orange button with a download icon and the text 'Download'. Below the button, it says 'Download from: Germany - University of Erlangen-Nuremberg (https)'. Then, it shows the file name 'File: eclipse-inst-jre-win64.exe' and the SHA-512 hash. At the bottom, there is a link '>> Select Another Mirror'.

Web de descarga de instalador de Eclipse.

Fuente: www.eclipse.org

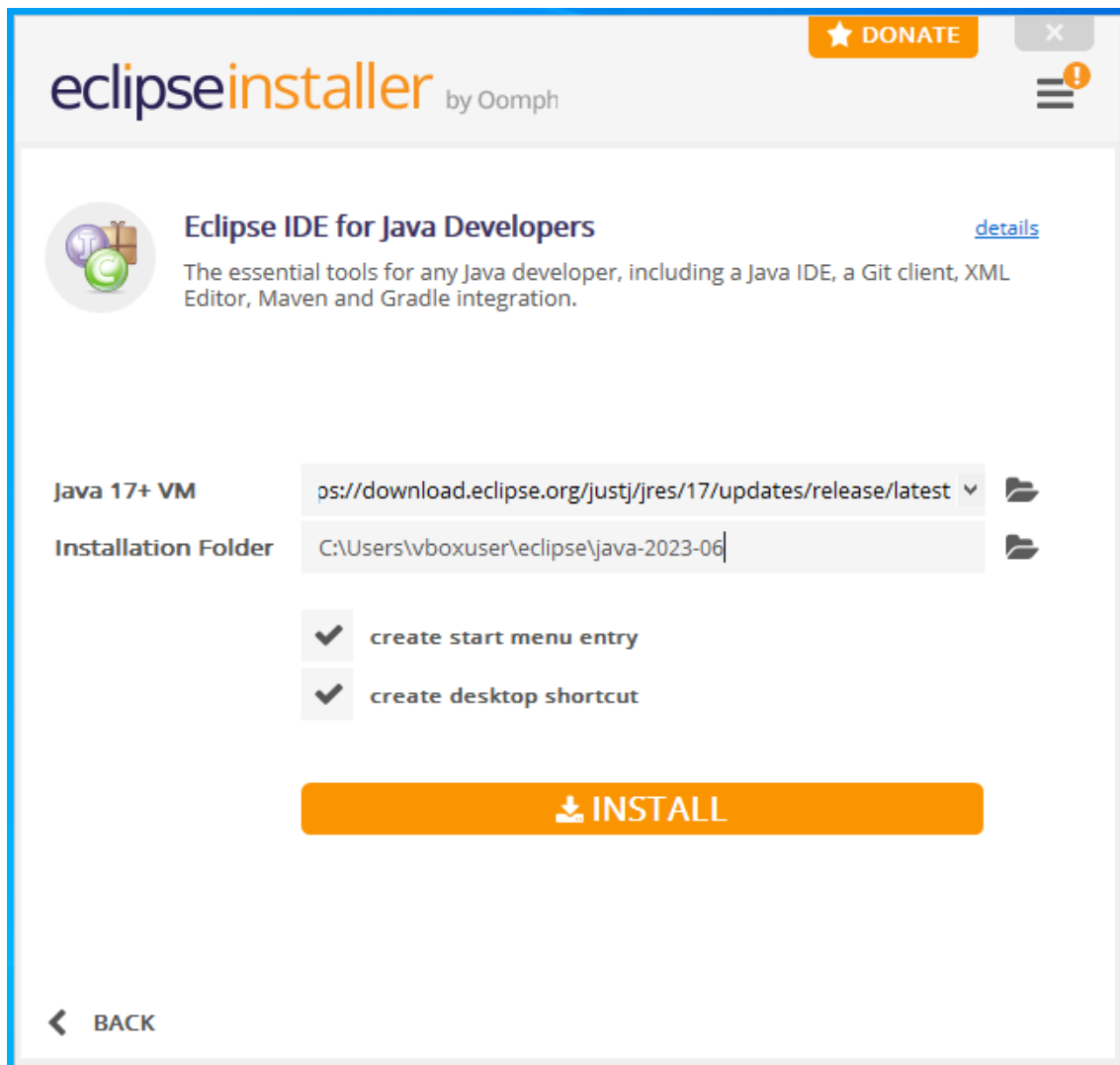
4. Ejecutamos el archivo descargado, y se carga el instalador para el que necesitamos acceso a internet, elegimos la primera opción, "Eclipse IDE for Java Developers". En esta versión se descargará tanto el IDE como el JDK (Java Development Kit - Kit de desarrollo de Java) y la JVM (Java Virtual Machina - Máquina Virtual de Java) de forma automática.



Instalador de Eclipse.

Fuente: Ejecutable instalador Eclipse.

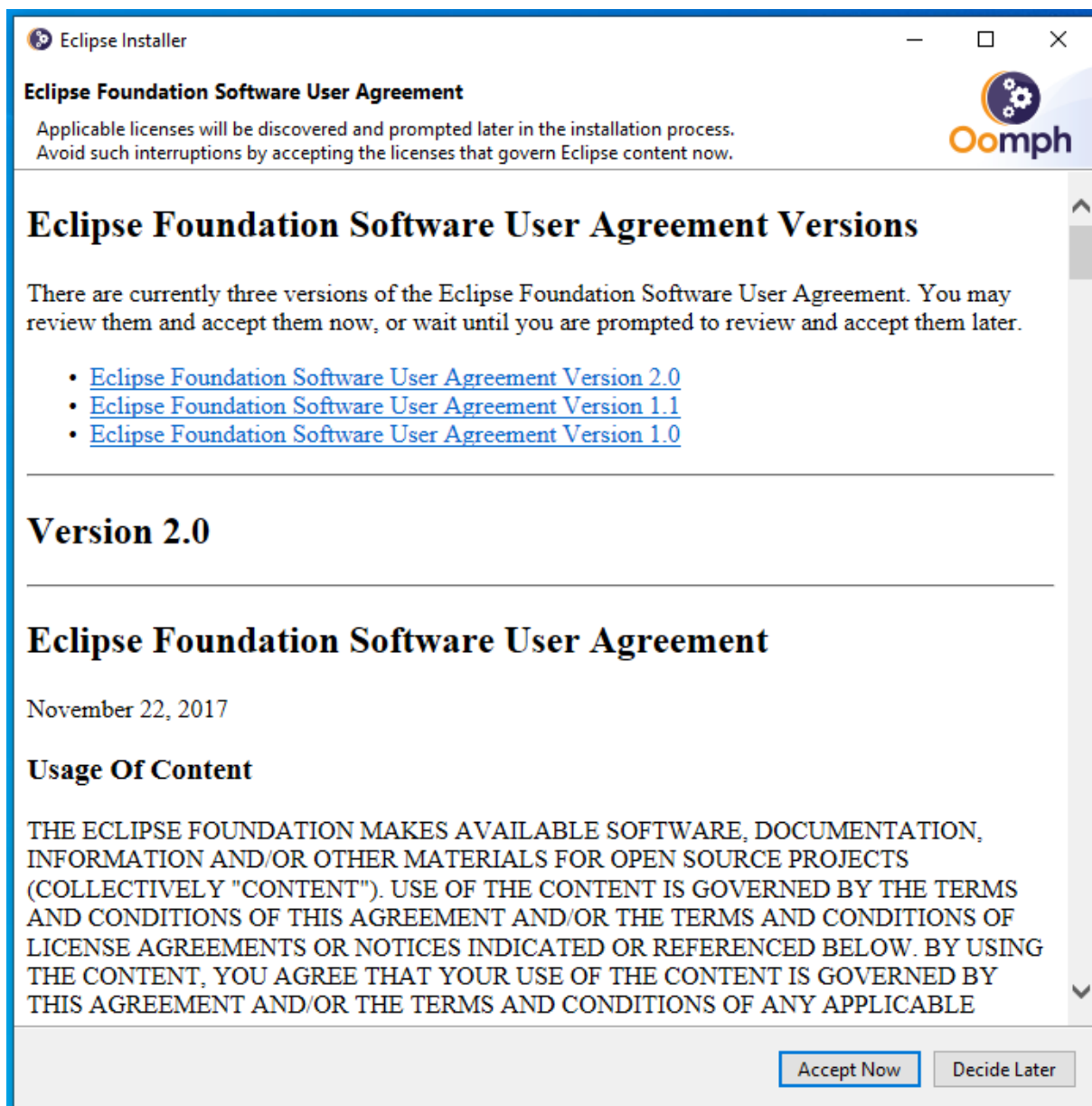
5. En esta imagen se puede apreciar las rutas de instalación de Eclipse y de los paquetes de instalación necesarios para JAVA, tal y como se han mencionado en el apartado anterior.



Rutas de instalación, accesos directos.

Fuente: Ejecutable instalador Eclipse.

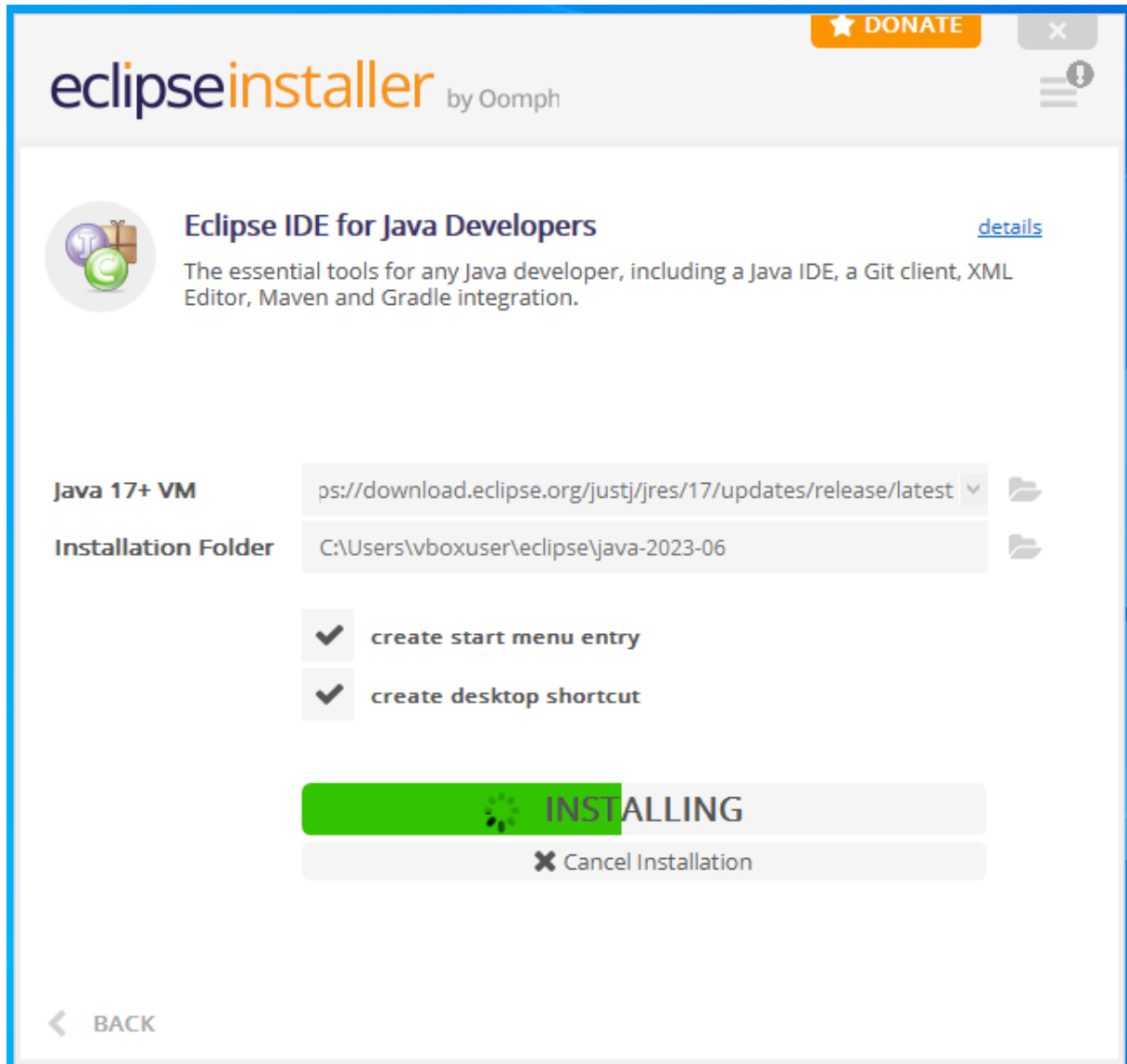
6. Aceptamos los términos y condiciones de Eclipse



Aceptación de las condiciones de uso de Eclipse.

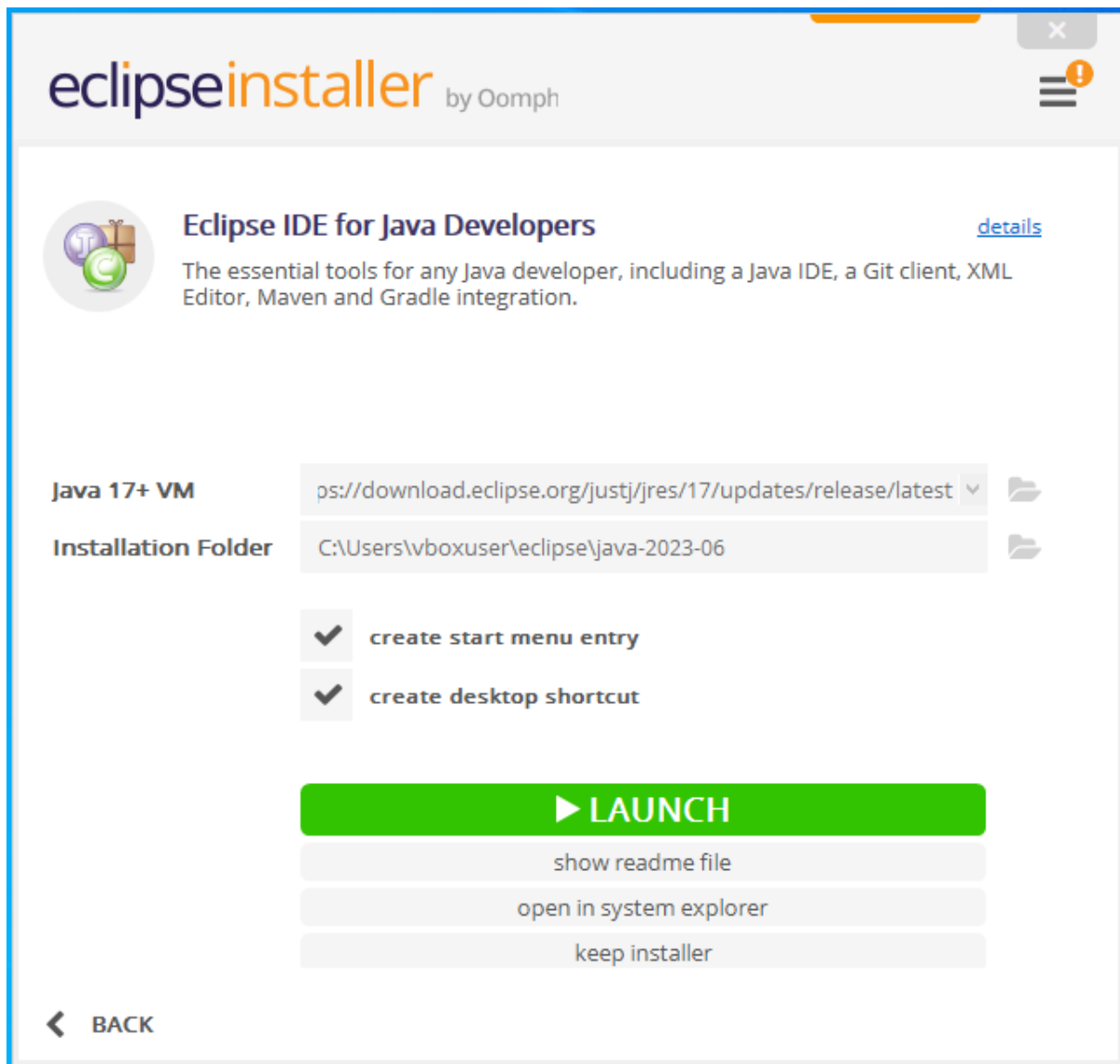
Fuente: Ejecutable instalador Eclipse.

7. Aparece la barra de instalación y el botón para cancelarla si algo sale mal, recordar que es necesario internet para todo el proceso.



Barra de progreso de instalación de Eclipse.
Fuente: Ejecutable instalador Eclipse

8. Una vez instalado, se nos habrán creado accesos directos en el escritorio y en el menú de comienzo, y se podrá lanzar la aplicación presionando en LAUNCH



Pantalla de finalización de instalación de Eclipse.
Fuente: Ejecutable instalador Eclipse.



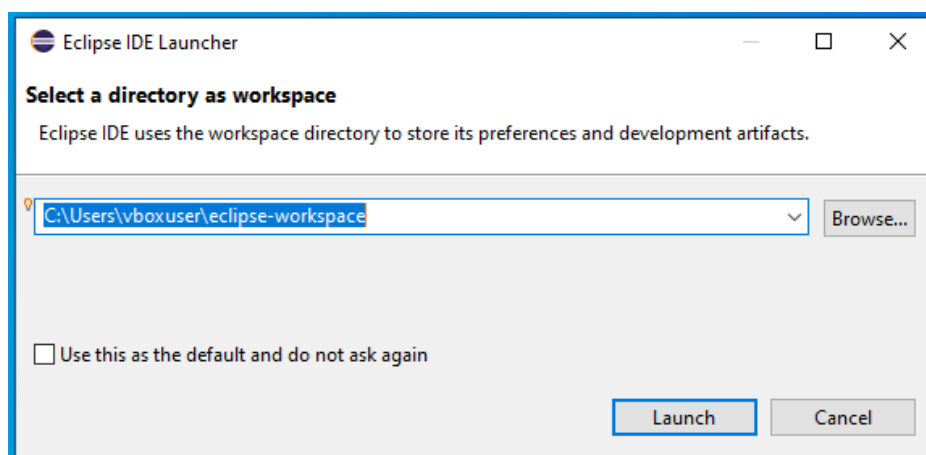
VÍDEO DE INTERÉS

Conoce cómo instalar Eclipse en Windows 10 en este vídeo:



Creación del primer programa en Eclipse con JAVA:

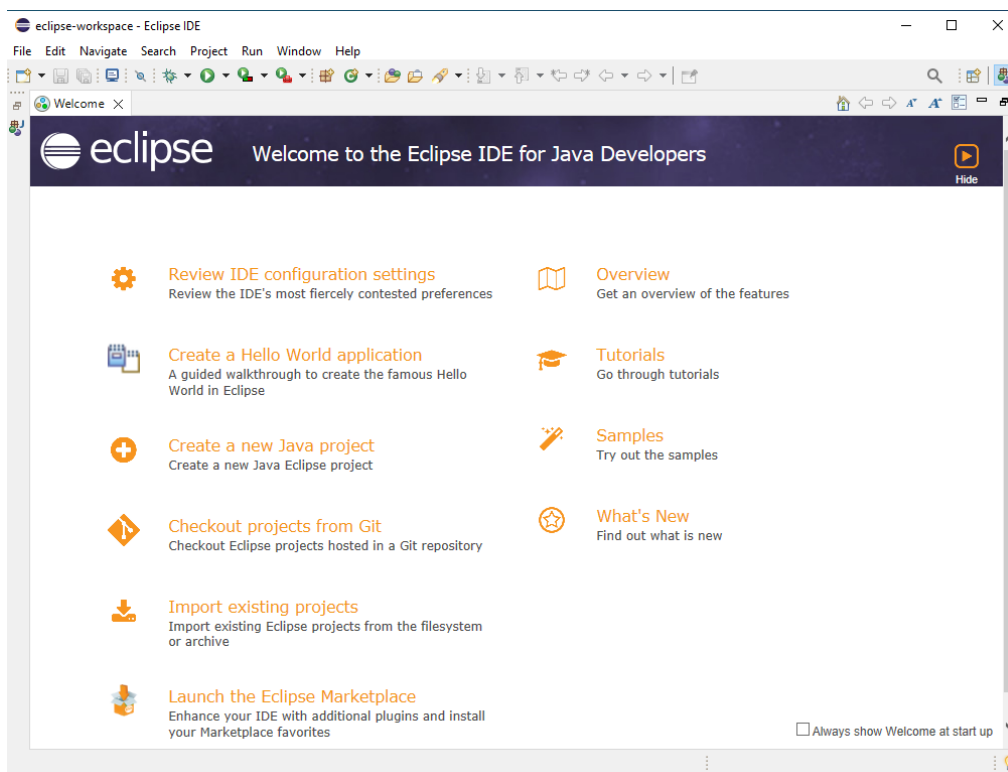
1. Abrimos Eclipse, nos aparece el mensaje de la localización del workspace (espacio de trabajo), es donde se almacenarán todos los proyectos que realicemos con Eclipse. Podemos cambiar la ruta, o dejarla por defecto, una ruta recomendable es C:\\ProyectosEclipse, ya que es una ruta que no lleva acentos, ni espacios, en la ruta por defecto en el nombre del usuario de Windows hay veces que aparecen estos símbolos y a la hora de ejecutar pueden producir algún tipo de error.



Elección espacio de trabajo.

Fuente: Propia.

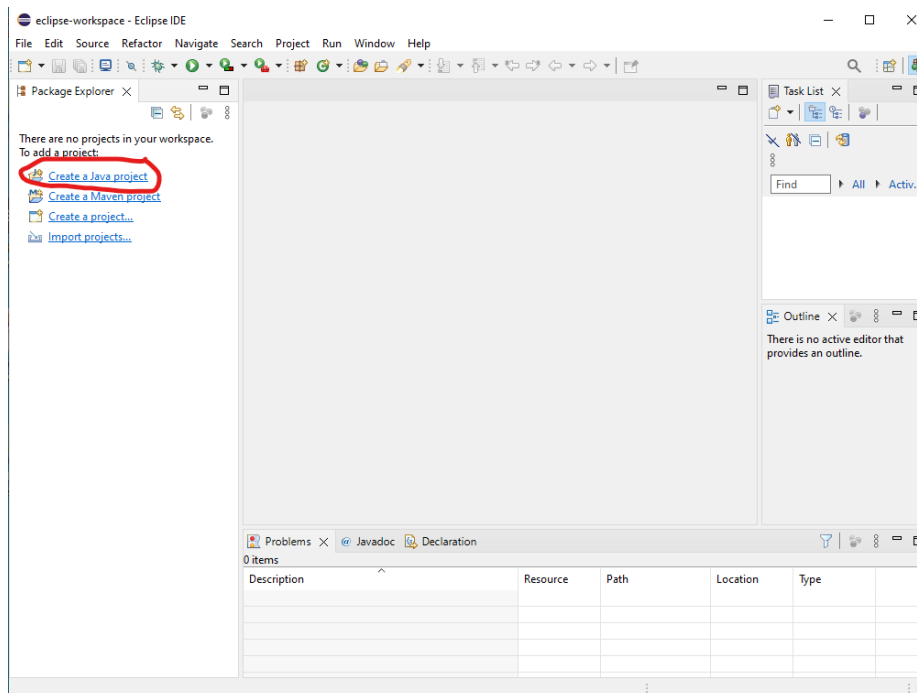
2. Nos aparece esta pantalla de bienvenida que podemos cerrar.



Bienvenida de Eclipse.

Fuente: Propia.

3. Creamos un proyecto en Java presionando "Create a Java Project".



Creación proyecto Java en Eclipse.

Fuente: Propia.

4. En la pantalla que aparece escribimos en el nombre del proyecto en nuestro caso “HolaMundo”. Desmarcamos la casilla “Create module-info.java file”, y presionamos en “Finish”.

New Java Project

Create a Java Project
Create a Java project in the workspace or in an external location.

Project name:

☒ Use default location
Location: [Browse...](#)

JRE

☒ Use an execution environment JRE: [Configure JREs...](#)

☐ Use a project specific JRE:

☐ Use default JRE 'jre' and workspace compiler preferences

Project layout

☐ Use project folder as root for sources and class files

☒ Create separate folders for sources and class files [Configure default...](#)

Working sets

☐ Add project to working sets [New...](#)

Working sets: [Select...](#)

Module

☐ Create module-info.java file

Module name:

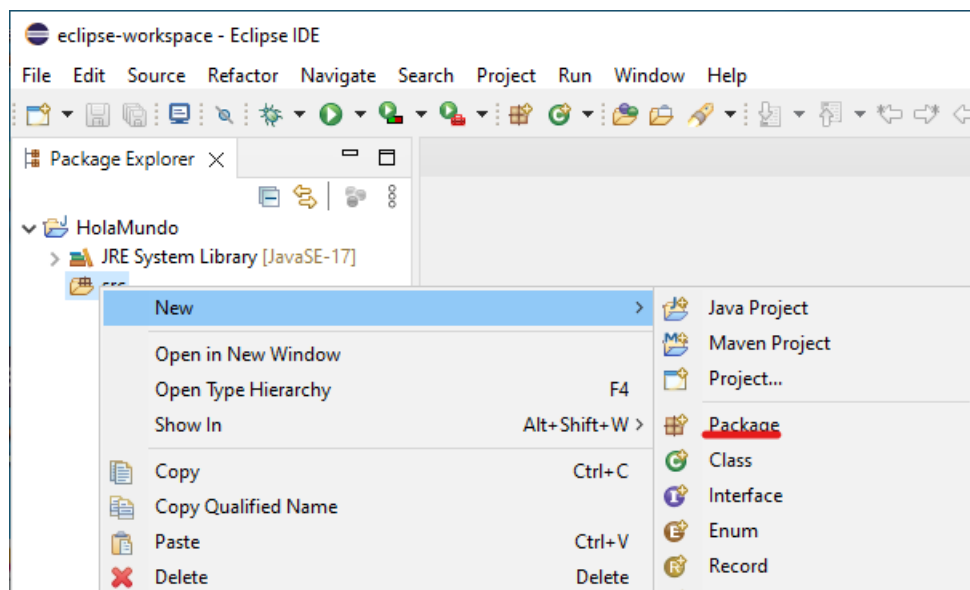
☐ Generate comments

[?](#) [< Back](#) [Next >](#) **Finish** [Cancel](#)

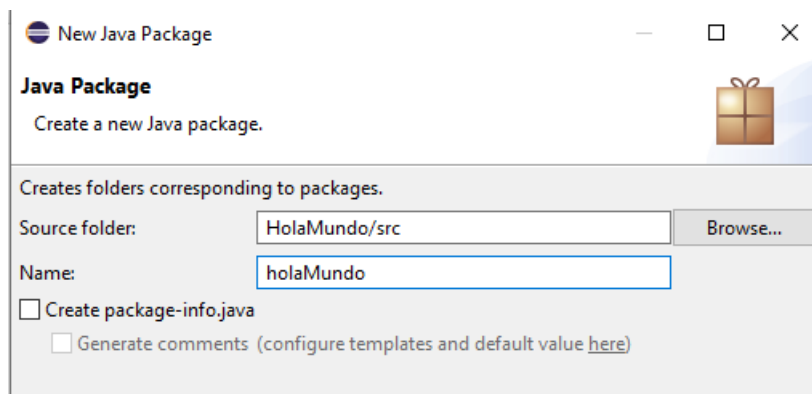
Creación de proyecto JAVA.

Fuente: Propia.

5. Nos aparece la estructura de un proyecto en Java, empezamos creando un paquete donde contendrá los archivos java.



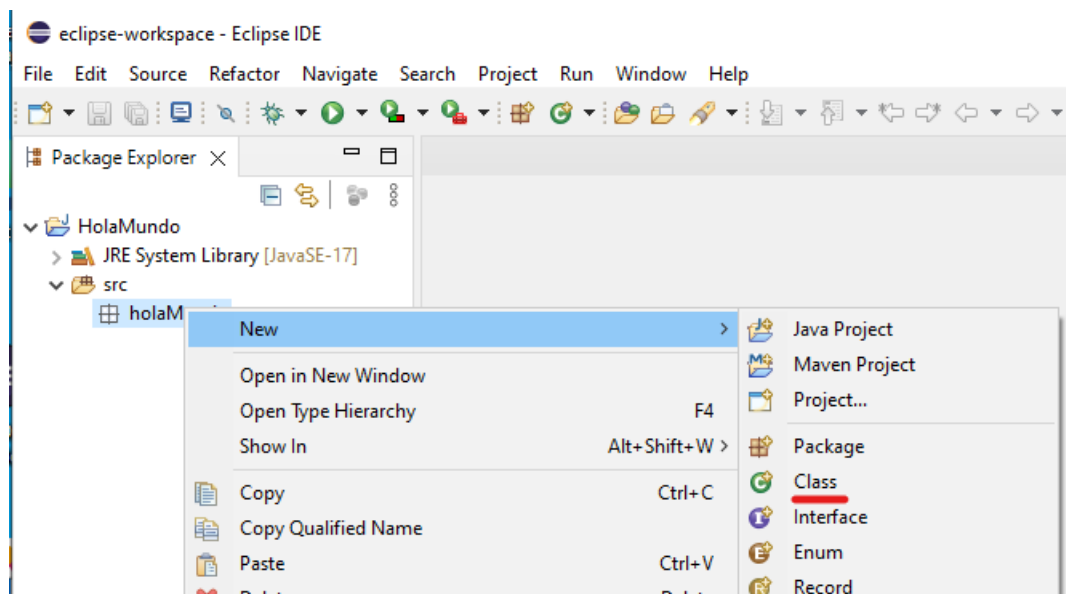
Creación de paquete en Eclipse.



Pantalla de creación de paquete.

Fuente: Propia.

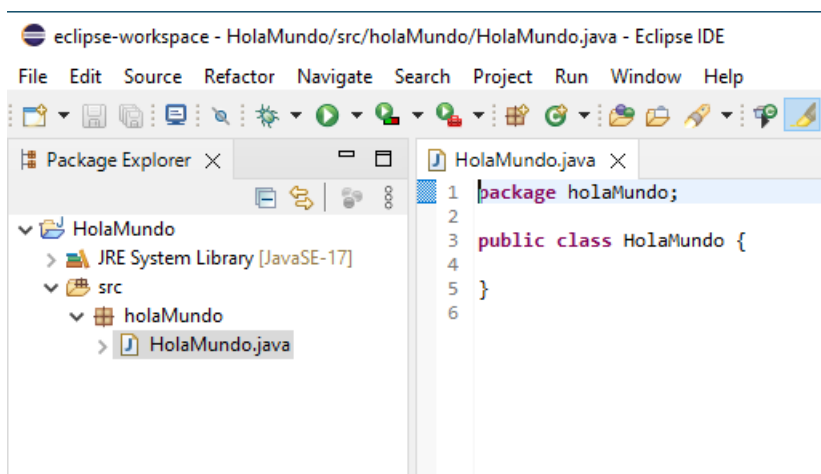
6. Después creamos una clase llamada “HolaMundo”



Creación clase JAVA.

Fuente: Propia.

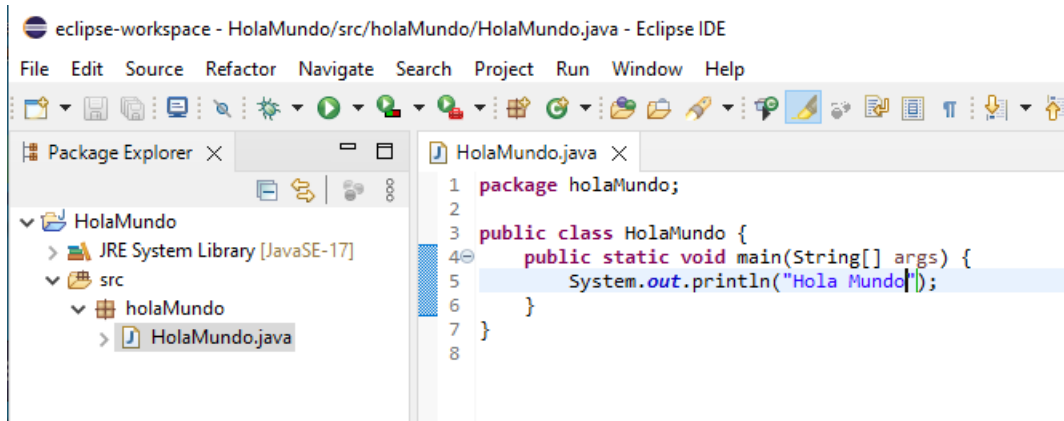
7. Se nos crea un archivo en la estructura llamado “HolaMundo.java”, y se nos abre dicho archivo en el programa.



Pantalla con clase HolaMundo.java creada.

Fuente: Propia.

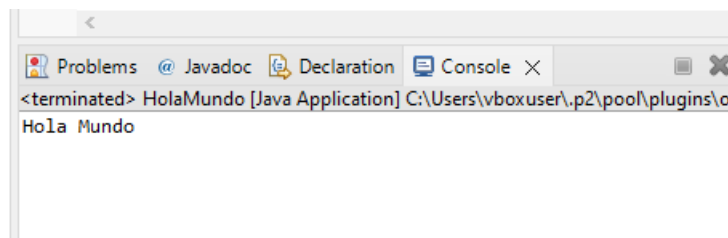
8. Escribimos en la línea 4 main y presionamos CTRL+ESPACIO, se nos crea el método main o principal, que será el que se ejecute en el programa. Dentro del método escribimos syso y presionamos CTRL+ESPACIO, se nos autorrellena el System.out.println. Entre las comillas escribimos lo que deseamos que muestre por pantalla.



“Hola Mundo” en java.

Fuente: Propia

9. Ejecutamos presionando el botón  y en la consola nos aparece el mensaje.



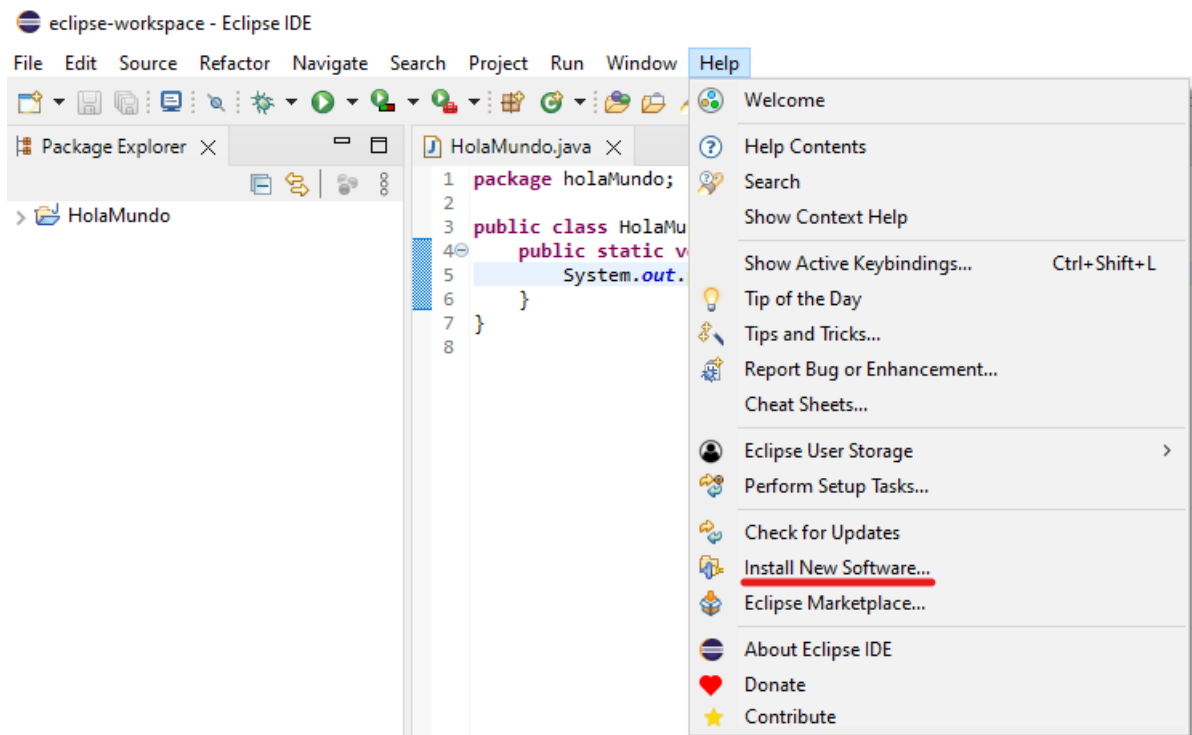
Consola de ejecución Eclipse.

Fuente: Propia.

Actualización del entorno de desarrollo Eclipse: Se pueden añadir plugins, es decir, actualizar el entorno de desarrollo Eclipse de dos formas, manualmente y automática.

- **Agregar plugins manualmente:** Para agregar un plugin de manera manual, se deben seguir los siguientes pasos:

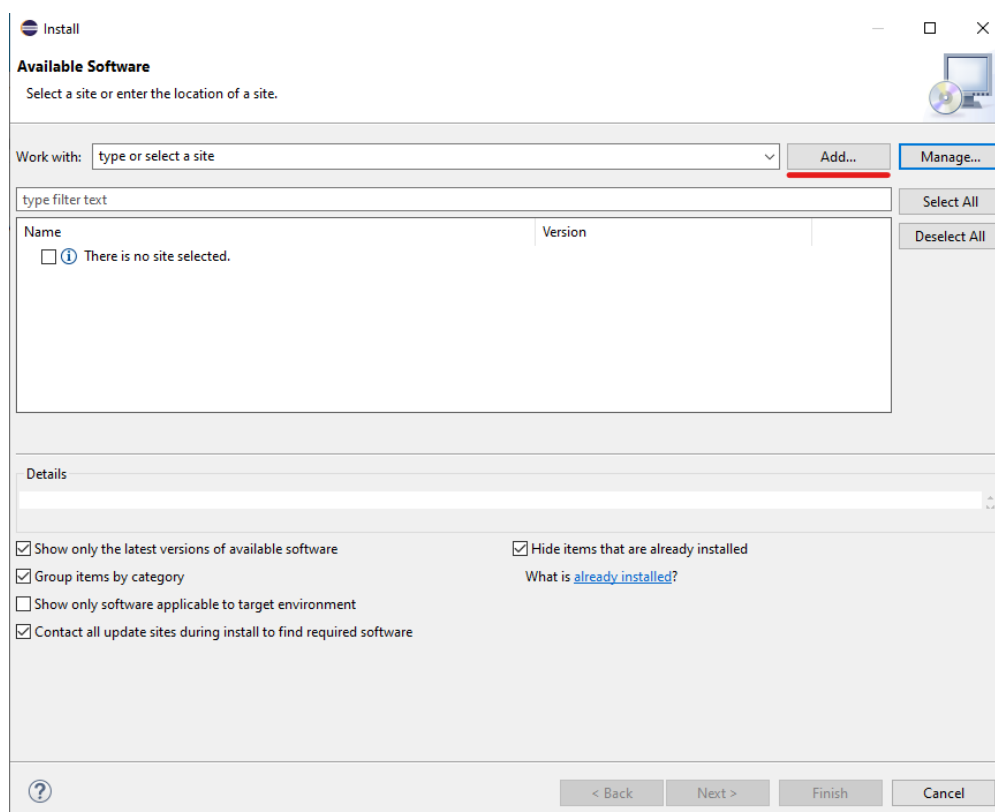
1. Con Eclipse abierto, nos vamos a “Help → Instal New Software...”



Install New Software.

Fuente: Propia.

- Nos aparece la siguiente pantalla y le damos a añadir el software presionando en "ADD".

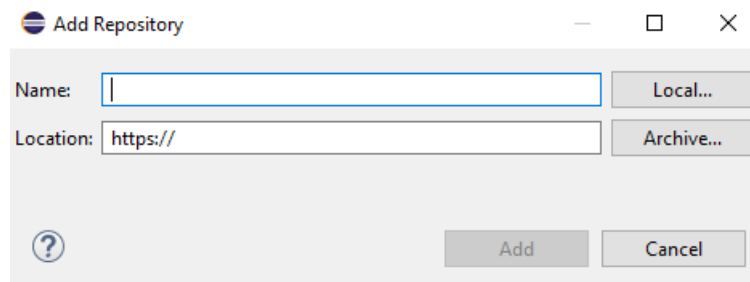


Pantalla para añadir software.

Fuente: Propia.

3. En la ventana que nos aparece tenemos dos opciones:

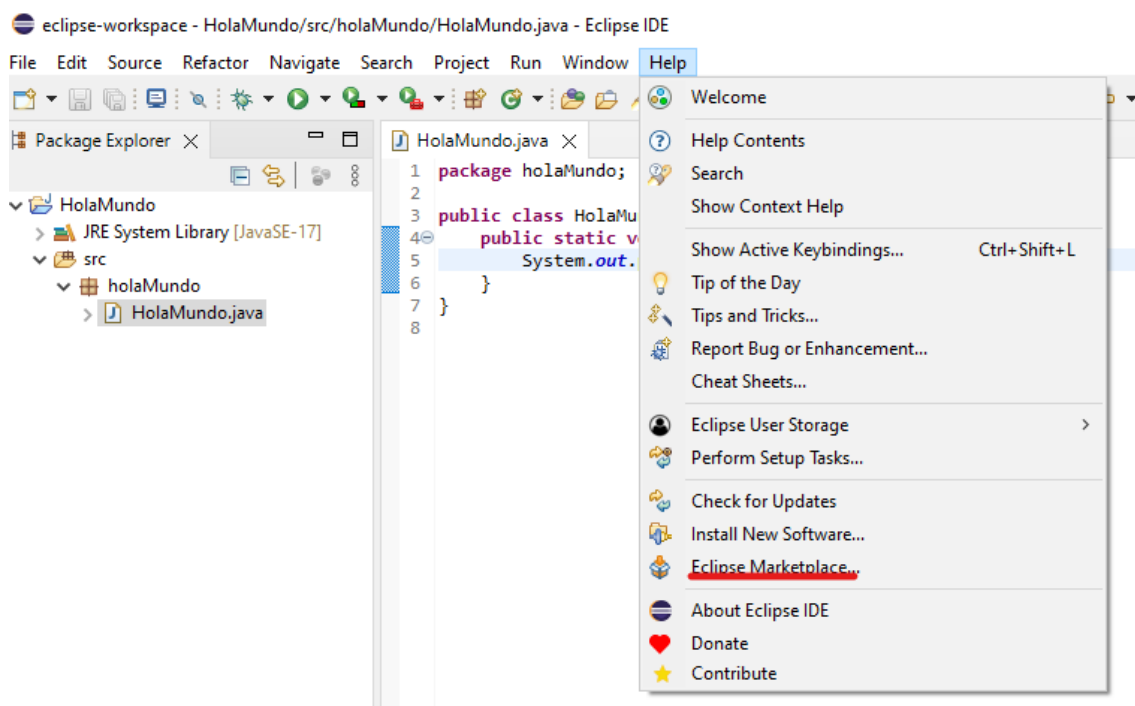
- Si presionamos en “Local”, tenemos que seleccionar el archivo que previamente nos hemos tenido que descargar con el plugin que queremos instalar, y que se encuentra dentro de nuestro equipo local.
- En location, podemos poner la url donde se encuentre dicho archivo para que el propio programa se lo descargue. Le damos al botón “Add”, y después a “Finish”.



Ventana para añadir plugin.

- **Agregar plugins de manera automática:** Para agregar un plugin de manera automática, se deben seguir los siguientes pasos:

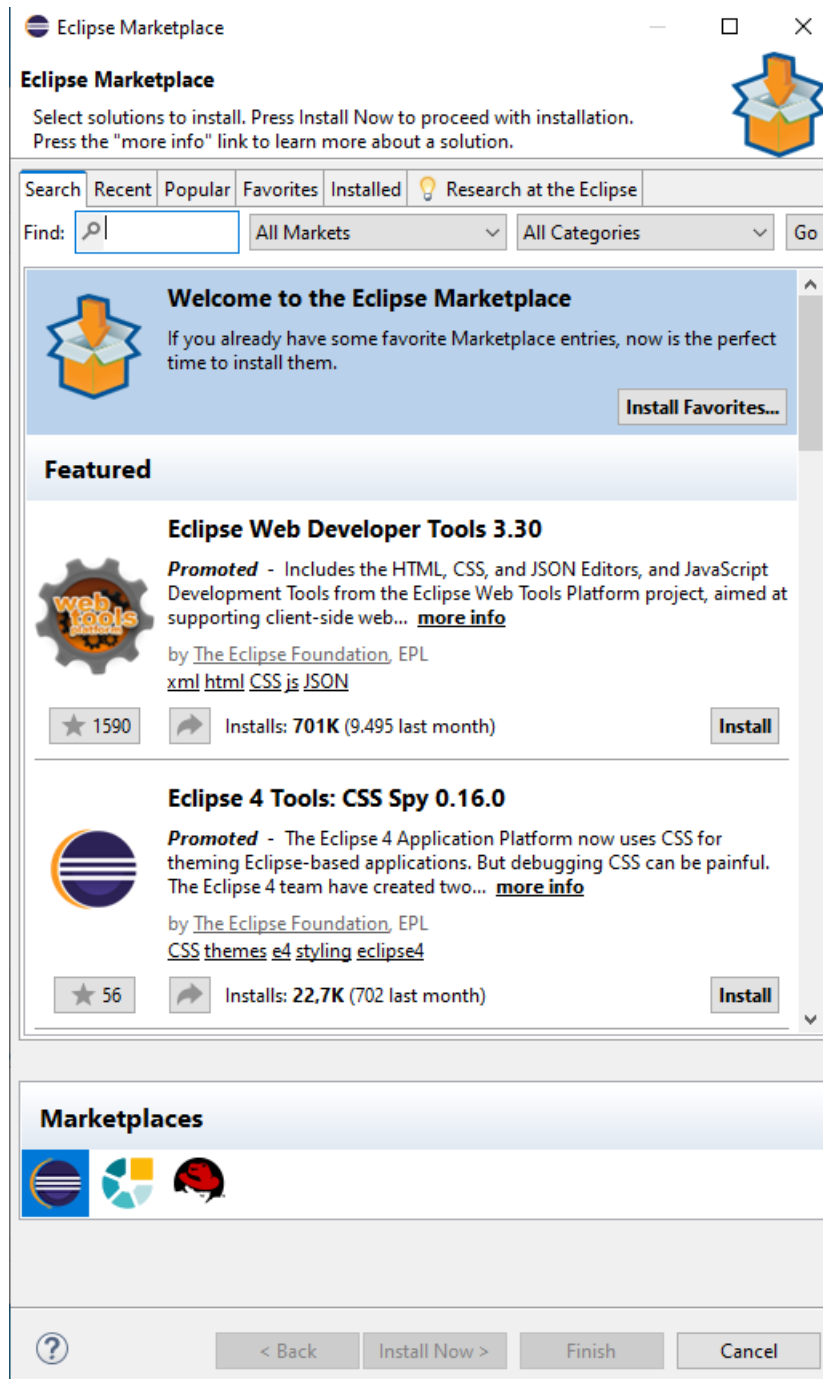
1. Con Eclipse abierto, nos vamos a “Help → Eclipse Marketplace...” para ir a la tienda de plugins de Eclipse.



Ventana para añadir plugin.

Fuente: Propia.

2. Aparece esta ventana, en la que podemos buscar el plugin por nombre, ver los añadidos recientemente, los más populares, e incluso los que ya tenemos instalados. Simplemente le damos a instalar y se instalan.



Ventana Marketplace Eclipse.

Fuente: Propia.

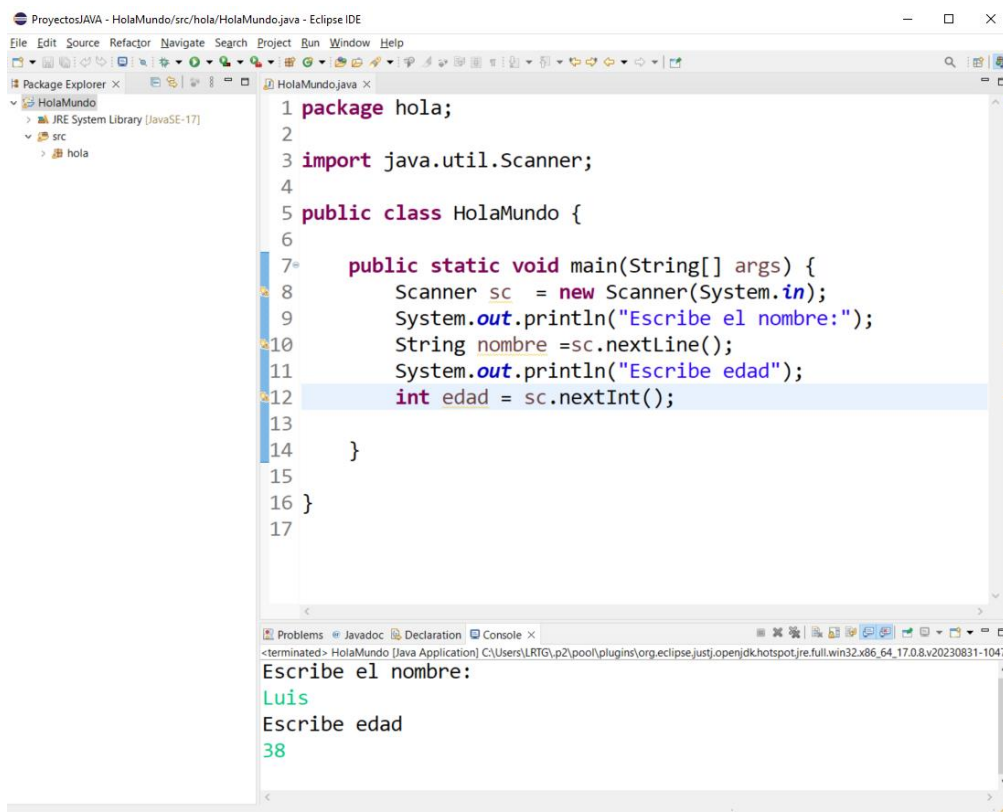
3. Al acabar nos pedirá que reiniciemos Eclipse para aplicar los cambios al programa.

Por último, vemos como actualizar Eclipse, para ello seguiremos los siguientes pasos.

1. Con Eclipse abierto, nos vamos a “Help → Check for Updates”.
2. En la parte inferior derecha de Eclipse aparece una barra pequeña en la que si presionamos podemos seguir la instalación de las actualizaciones.
3. Al acabar el programa preguntará si desea reiniciar el programa. En la mayoría de casos es recomendable reiniciar Eclipse para que se apliquen los cambios.

Creación ejecutable (jar) en Eclipse:

Para crear un ejecutable de un proyecto que hemos creado en java con Eclipse utilizaremos el siguiente código de ejemplo. Cómo se puede apreciar, el programa pide el nombre y la edad por teclado y lo almacena en variables.



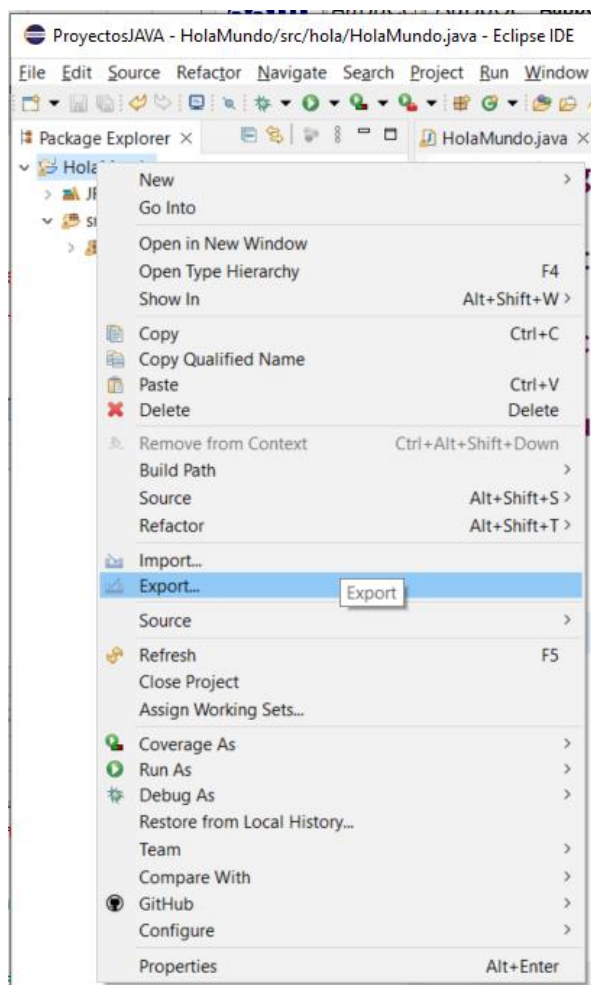
```
1 package hola;
2
3 import java.util.Scanner;
4
5 public class HolaMundo {
6
7     public static void main(String[] args) {
8         Scanner sc = new Scanner(System.in);
9         System.out.println("Escribe el nombre:");
10        String nombre = sc.nextLine();
11        System.out.println("Escribe edad");
12        int edad = sc.nextInt();
13    }
14 }
15 }
16 }
17 }
```

Escribe el nombre:
Luis
Escribe edad
38

Ventana Eclipse.

Fuente: Propia.

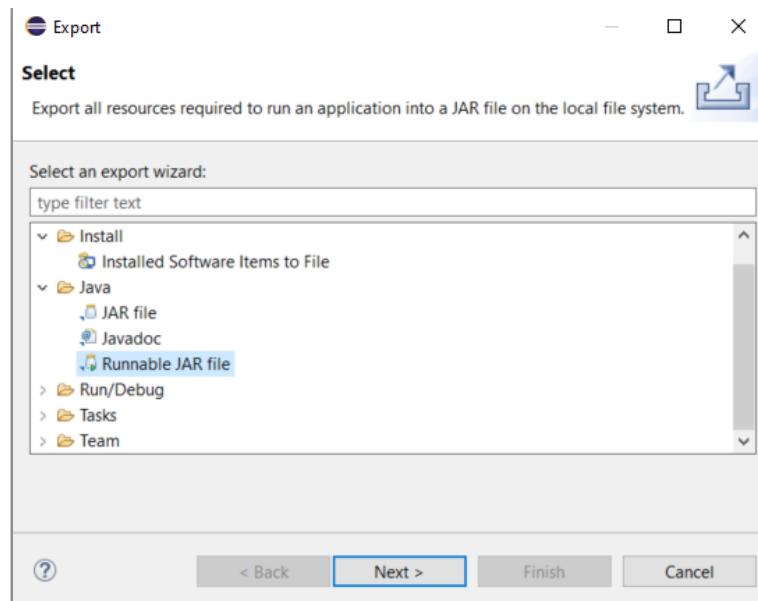
Una vez que se tiene el proyecto que se quiere trasladar a un archivo ejecutable, se presiona sobre el proyecto con el botón derecho del ratón para que aparezcan las opciones de la imagen, y se pueda elegir la opción Export.



Opciones de proyecto.

Fuente: Propia.

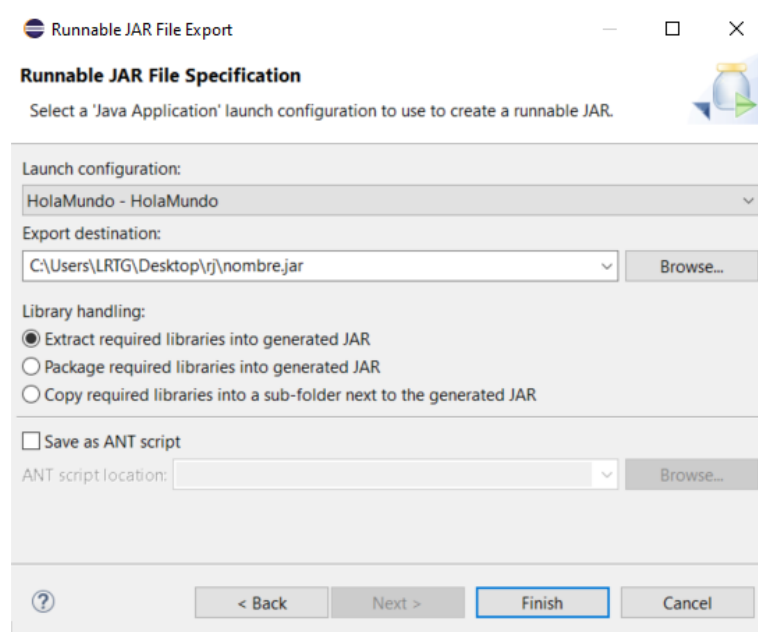
Aparecerán las siguientes opciones, se elige dentro de la carpeta Java, la opción Runnable JAR file, y se pulsa en siguiente.



Opciones de ejecutable.

Fuente: Propia.

En la ventana siguiente se debe seleccionar la clase que tenga el método principal o MAIN, y la carpeta donde se desea crear el archivo ejecutable. Tras esto se crea un archivo ejecutable con extensión .jar que al hacer doble click, se ejecuta.



Opciones de creación.

Fuente: Propia.



ENLACE DE INTERÉS

En esta web puedes encontrar más información sobre la instalación y configuración de Eclipse:



EJEMPLO PRÁCTICO

Cómo jefe de proyecto has elegido JAVA para desarrollar el mismo, ¿qué entorno de desarrollo elegirías?

Dependerá de muchos factores, tenemos tres IDEs o entornos de desarrollo posibles para la creación de programas en JAVA de forma profesional, las tres opciones serían:

- Eclipse: Es un entorno sencillo, fácil de configurar, la única pega que se le puede poner es que cuantos más plugin tenga instalados, para github, sonar, etc. más lento es.
- Visual Studio Code: Programa muy sencillo, la configuración no es tan fácil como Eclipse y puede desconfigurarse fácilmente.
- IntelliJ: Entorno muy profesional, preparado para trabajar con otras herramientas como GITHUB.

Aunque, todos tienen sus pros y contras, todo dependerá de la formación que tengan los miembros del equipo en cada uno de estos entornos, y del tiempo del que se disponga para hacerlo, si se dispone de tiempo para que puedan aprender a manejar un entorno que se supone es mejor o no.

RESUMEN FINAL

Tal y como hemos estudiado, los entornos de desarrollo están formados por un conjunto de software que nos facilitan o automatizan las actividades de desarrollo. Estos entornos de desarrollo deben dar soporte a las actividades verticales y horizontales.

Las herramientas CASE, por ejemplo, pueden cubrir una o varias de las actividades de ingeniería del software y se compone de distintos elementos, como: repositorio, metamodelo, carga o descarga de datos, comprobación de errores e interfaz de usuario.

Los entornos de programación, a su vez, son herramientas CASE, que se encargan de dar soporte al desarrollador para la creación del código de programas que se van a implementar. Los entornos de programación se pueden clasificar en: entornos orientados al lenguaje, entornos orientados a estructuras, entornos basados en combinación de herramientas y entornos. Algunas de las herramientas CASE que podemos encontrar son Microsoft Project y Oracle JDeveloper.

Uno de los principales IDEs existentes en el mercado para JAVA es Eclipse, y de esta manera hemos terminado la unidad acercándonos a la planificación y elección de un IDE para poder llevar a cabo un proyecto real.