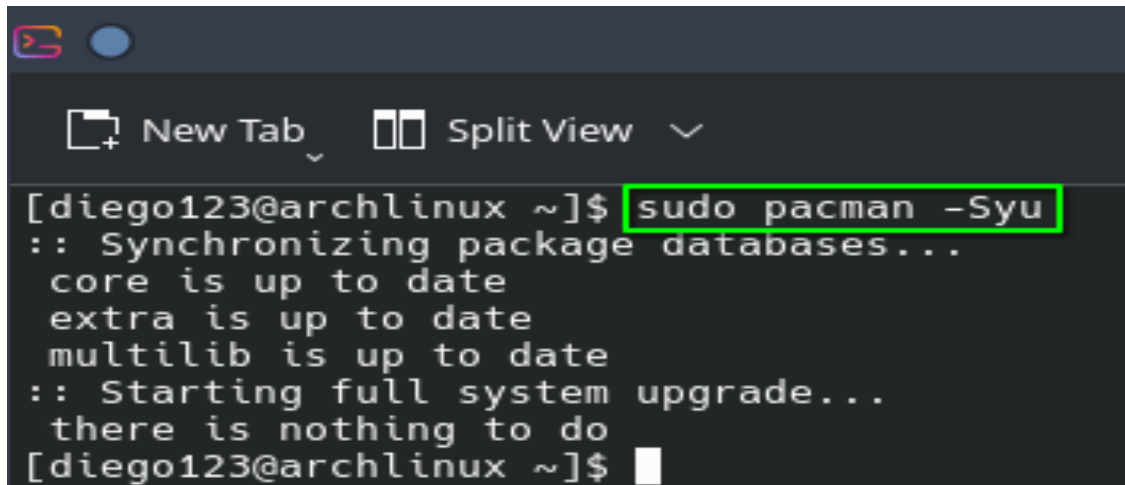


ELABORADO POR: JAIMES ESCOBAR DIEGO

INSTALAR DOCKER EN LINUX (ARCH LINUX-MANJARO LINUX, ENDEAVOUROS, ARCOLINUX, GARUDA LINUX, ARTIX LINUX, BLACKARCH LINUX, ETC)

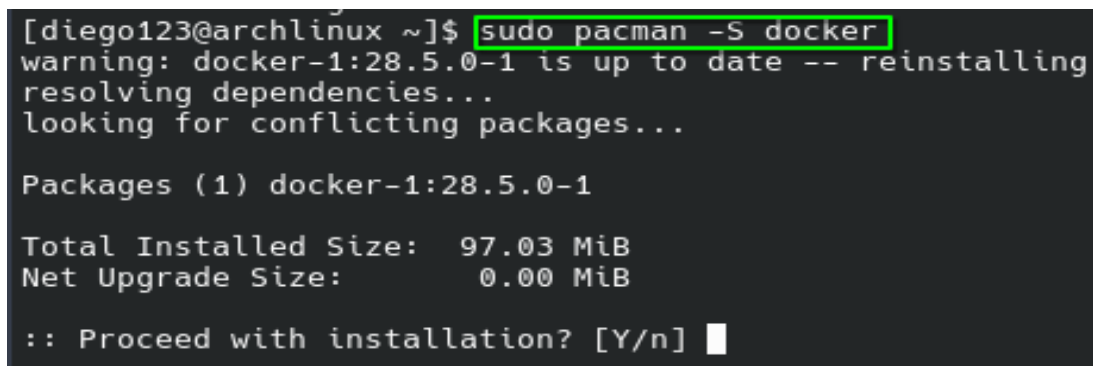
- (1) **sudo pacman -Syu** → Actualizar todas las dependencias que son recientes



```
[diego123@archlinux ~]$ sudo pacman -Syu
:: Synchronizing package databases...
core is up to date
extra is up to date
multilib is up to date
:: Starting full system upgrade...
there is nothing to do
[diego123@archlinux ~]$
```

- (2) Dependiendo de que uses (**yay** o **pacman**)

- **sudo pacman -S docker** ò **yay -S docker** → instala docker



```
[diego123@archlinux ~]$ sudo pacman -S docker
warning: docker-1:28.5.0-1 is up to date -- reinstalling
resolving dependencies...
looking for conflicting packages...

Packages (1) docker-1:28.5.0-1

Total Installed Size: 97.03 MiB
Net Upgrade Size: 0.00 MiB

:: Proceed with installation? [Y/n]
```

#NOTA: En la imagen podemos apreciar cuantos paquetes se instalan y el peso o tamaño de la descarga, se procede a dar Y o en algunos casos la S; y si en algun punto te arrepientes y ya no lo quieres instalar simplemente das en N

- (3) **sudo systemctl enable docker** → para habilitar docker (No esperes una salida o algo que te muestre, tal parece que no hace nada pero si lo activa)
- (4) **sudo systemctl start docker** → para iniciar docker (Lo mismo que el anterior, no dara una salida pero lo iniciara)

- (5) **sudo systemctl status docker** → para ver el estado del docker, nos dira si esta activo o si no lo esta.

```
[diego123@archlinux ~]$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
   Active: active (running) since Sat 2025-10-04 17:08:31 CST; 1h 41min ago
     Invocation: 25f779527d9c4444854e0f04607cfab8
   TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Main PID: 753 (dockerd)
       Tasks: 18
      Memory: 117M (peak: 130.2M)
         CPU: 2.389s
        CGroup: /system.slice/docker.service
                └─753 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

- (6) **sudo pacman -S docker-compose (Opcional)** → instala docker compose

```
[diego123@archlinux ~]$ sudo pacman -S docker-compose
[sudo] password for diego123:
resolving dependencies...
looking for conflicting packages...

Packages (1) docker-compose-2.40.0-1

Total Installed Size: 63.39 MiB

:: Proceed with installation? [Y/n]
```

#Nota: Este por lo general es opcional, si apenas empiezas y solo quieres probar con programas pequeños o que solo conlleven un contenedor pues no lo instales, pero si ya requieres de mas contenedores entonces instalalo.

- (7) **docker run hello-world** → solo es para verificar que funcione: dira “Hello from docker”

```
[diego123@archlinux ~]$ docker run hello-world
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.
```

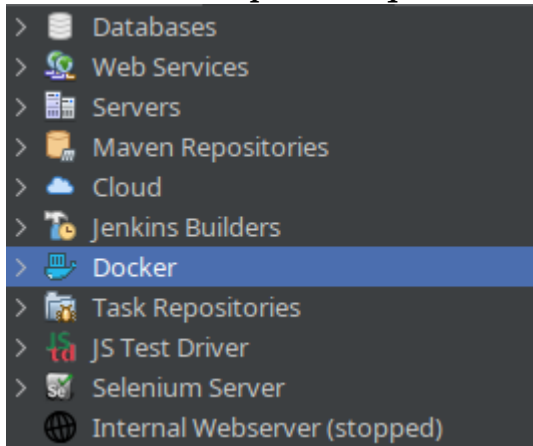
#NOTA: Hasta aquí seria todo para tener docker instalado.

ELABORADO POR: JAIMES ESCOBAR DIEGO

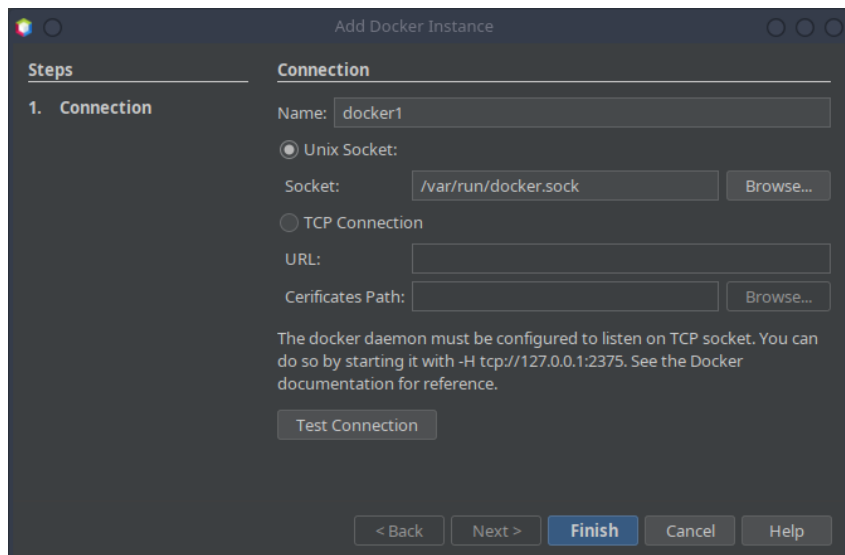
PROBAR DOCKER EN IDE (ENTORNO DE DESARRO INTEGRADO)

En este caso solo lo probare en el IDE de NetBeans

1. Ingresar a NetBeans y en barra de herramientas buscar **Services** y buscaremos el apartado que dice docker:



Click derecho y damos donde dice “Add docker” debe aparecerte algo asi:



por defecto como somos de linux, debe de tener seleccionada la opcion de **Unix Socket** y donde dice **Socket** debe de aparecerte esa ruta como la mostrada en pantalla, si no aparece, escribela. Despues de haber puesto la ruta en el socket, vamos a darle en donde dice **Test Connection** y debe de aparecer el mensaje de conexión exitosa, si aparece eso, damos en **Finish**

2. ingresa a tu proyecto o crea uno.

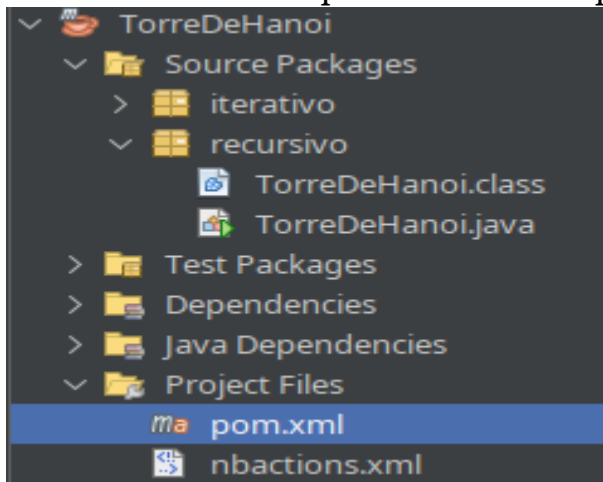
ELABORADO POR: JAIMES ESCOBAR DIEGO

3. Crea una main class o clase principal asi como esta:

```
public static void main(String[] args) {  
    int cuadritos = 3;  
    String p1 = "A", p2 = "B", p3 = "C";  
    Torre(cuadritos, p1, p2, p3);  
}
```

#Nota: Si tienes mas clases fuera del main, solo importamos esas clases al main y listo, lo importante es tener un main

4. Ubicamos el archivo pom.xml en la carpeta **Project Files**



5. Damos doble click y encontramos una estructura como la siguiente

```
<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:  
    <modelVersion>4.0.0</modelVersion>  
    <groupId>com.mycompany</groupId>  
    <artifactId>TorreDeHanoi</artifactId>  
    <version>1.0-SNAPSHOT</version>  
    <packaging>jar</packaging>  
    <properties>  
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
        <maven.compiler.release>21</maven.compiler.release>  
        <exec.mainClass>com.mycompany.torredehanoi.TorreDeHanoi</exec.mainClass>  
    </properties>  
</project>
```

6. Ahora la etiqueta que nos interesa por el momento es esta:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>TorreDeHanoi</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.release>21</maven.compiler.release>
    <exec.mainClass>com.mycompany.torredehanoi.TorreDeHanoi</exec.mainClass>
  </properties>
</project>
```

En esa parte reemplazamos lo que esta en rojito (en mi caso com.mycompany.torredehanoi.TorreDeHanoi) por el nombre de tu paquete, en mi caso es recursivo y el nombre de tu main que en mi caso es TorreDeHanoi, quedaria algo asi:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>TorreDeHanoi</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.release>21</maven.compiler.release>
    <exec.mainClass>recursivo.TorreDeHanoi</exec.mainClass>
  </properties>
</project>
```

7. Este paso es muy importante, agregaremos una estructura de etiquetas de la siguiente manera:

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-jar-plugin</artifactId>
      <version>3.2.2</version>
      <configuration>
        <archive>
          <manifest>
            <mainClass>recursivo.TorreDeHanoi</mainClass>
          </manifest>
        </archive>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Esto es muy importante cuando queremos un JAR ejecutable. Si maven genera un JAR pues este no sabe que cosa va a ejecutar y si por ejemplo llegamos a ejecutar **java -jar app.jar** va a colapsar y ahora la pregunta es... ¿Por qué pasa eso?, es sencillo, eso sucede porque manifest de nuestro queridísimo JAR no tiene definido el mainClass, para ser mas exactos, esta etiqueta **<mainClass>recursivo.TorreDeHanoi</mainClass>**, esta etiqueta lleva lo mismo que la etiqueta anterior que cambiamos y buenop, algo así quedaria nuestro cuerpo del pom.xml:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <modelVersion>4.0.0</modelVersion>
4    <groupId>com.mycompany</groupId>
5    <artifactId>TorreDeHanoi</artifactId>
6    <version>1.0-SNAPSHOT</version>
7    <packaging>jar</packaging>
8    <properties>
9      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
10     <maven.compiler.release>21</maven.compiler.release>
11     <exec.mainClass>recursivo.TorreDeHanoi</exec.mainClass>
12   </properties>
13   <build>
14     <plugins>
15       <plugin>
16         <groupId>org.apache.maven.plugins</groupId>
17         <artifactId>maven-jar-plugin</artifactId>
18         <version>3.2.2</version>
19         <configuration>
20           <archive>
21             <manifest>
22               <mainClass>recursivo.TorreDeHanoi</mainClass>
23             </manifest>
24           </archive>
25         </configuration>
26       </plugin>
27     </plugins>
28   </build>
29 </project>
```

#Nota: Guardaremos el archivo y saldremos, no modificaremos nada más

8. Abrimos una terminal y nos dirigimos a la ruta actual de nuestro proyecto y ejecutamos el comando **mvn clean package**:

```
[diego123@archlinux ~]$ cd NetBeansProjects/TorreDeHanoi
[diego123@archlinux TorreDeHanoi]$ mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.mycompany:TorreDeHanoi >-----
[INFO] Building TorreDeHanoi 1.0-SNAPSHOT
[INFO] from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- clean:3.2.0:clean (default-clean) @ TorreDeHanoi ---
[INFO] Deleting /home/diego123/NetBeansProjects/TorreDeHanoi/target
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ TorreDeHanoi ---
[INFO] skip non existing resourceDirectory /home/diego123/NetBeansProjects/TorreDeHanoi/src/main/resources
[INFO]
[INFO] --- compiler:3.13.0:compile (default-compile) @ TorreDeHanoi ---
[INFO] Recompiling the module because of changed source code.
[INFO] Compiling 2 source files with javac [debug release 21] to target/classes
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ TorreDeHanoi ---
[INFO] skip non existing resourceDirectory /home/diego123/NetBeansProjects/TorreDeHanoi/src/test/resources
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ TorreDeHanoi ---
[INFO] Recompiling the module because of changed dependency.
[INFO]
[INFO] --- surefire:3.2.5:test (default-test) @ TorreDeHanoi ---
[INFO]
[INFO] --- jar:3.2.2:jar (default-jar) @ TorreDeHanoi ---
[INFO] Building jar: /home/diego123/NetBeansProjects/TorreDeHanoi/target/TorreDeHanoi-1.0-SNAPSHOT.jar
[INFO]
```

#Nota: Asegurate de tener instalado maven ya que sino no te dejara ejecutar.

9. Regresamos a NetBeans y en nuestro proyecto, creamos un dockerfile (Se crea casi de la misma manera que un proyecto):

```
1
2 # Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3 # Click nbfs://nbhost/SystemFileSystem/Templates/Other/Dockerfile to edit this template
4
5 FROM alpine:latest
6
7 CMD ["/bin/sh"]
8
```

Dentro de este archivo vamos a hacer unos pequeños cambios los cuales son modificar el **FROM** por **FROM openjdk:21-jdk-slim** el cual es la version de java que tienes (mi caso la 21) y el slim que basicamente es para que la imagen que se crea sea de un peso menor. Despues agregamos a **WORKDIR /app** el cual lo agregaremos para que tengamos el directorio de trabajo en nuestro contenedor, basicamente es donde se ejecutaran las 2 instrucciones que vienen a continuacion. Tambie agregamos a **COPY target/TorreDeHanoi-1.0-SNAPSHOT.jar app.jar** que en pocas palabras es el que copia el JAR creado por maven dentro del contenedor con el nombre app.jar .Y por último modificamos el **CMD** **CMD ["/bin/sh"]** por **CMD ["java", "-jar", "app.jar"]** el cual indica que comandos va a ejecutar cuando nuestro contenedor este corriendo. Quedaria algo asi:

```
1
2 # Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change
3 # Click nbfs://nbhost/SystemFileSystem/Templates/Other/Dockerfile to edit this template
4
5 FROM openjdk:21-jdk-slim
6 WORKDIR /app
7 COPY target/TorreDeHanoi-1.0-SNAPSHOT.jar app.jar
8 CMD ["java", "-jar", "app.jar"]
9
```

#Nota: Guardamos los cambios que realizamos

ELABORADO POR: JAIMES ESCOBAR DIEGO

10. Volvemos a la terminal y en la misma ruta actual del proyecto, ejecutamos el siguiente comando: **docker build -t torredehanoi .**

```
[diego123@archlinux TorreDeHanoi]$ docker build -t torredehanoi .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 40.45kB
Step 1/4 : FROM openjdk:21-jdk-slim
--> a48f4cb73730
Step 2/4 : WORKDIR /app
--> Using cache
--> 5ce2c410a856
Step 3/4 : COPY target/TorreDeHanoi-1.0-SNAPSHOT.jar app.jar
--> 351fe9470c8a
Step 4/4 : CMD ["java", "-jar", "app.jar"]
--> Running in 792119eeeb8b
--> Removed intermediate container 792119eeeb8b
--> 3026388ab49f
Successfully built 3026388ab49f
Successfully tagged torredehanoi:latest
[diego123@archlinux TorreDeHanoi]$
```

#Nota: No se te vaya a olvidar el `.` que se encuentra en el comando, es importante.

11. Por ultimo ejecutamos el siguiente comando para ejecutar nuestro contenedor: **docker run --rm torredehanoi**

```
[diego123@archlinux TorreDeHanoi]$ docker run --rm torredehanoi
Mueve el cuadrito 1 de: A hacia C
Mueve el cuadrito 2 de: A hacia B
Mueve el cuadrito 1 de: C hacia B
Mueve el cuadrito 3 de: A hacia C
Mueve el cuadrito 1 de: B hacia A
Mueve el cuadrito 2 de: B hacia C
Mueve el cuadrito 1 de: A hacia C
[diego123@archlinux TorreDeHanoi]$
```

#Nota: En esta parte se muestra la ejecucion de nuestro programa pero la unica diferencia es que se encuentra en un contenedor :).

#Nota2: Cabe aclarar que si estamos o vamos a escribir por teclado en nuestro programa debemos agregar otra cosa al comando, el cual es `-it` quedando asi **docker run -it --rm nombredetuproyecto**

Manejar contenedores es esencial para tener el control de tus proyectos y sobre todo para manejar todo en un solo lugar sin que hagan falta dependencias en otras máquinas.

Gracias por tomarte el tiempo de leer esta pequeña documentación y ayuda para así instalar docker e implementarlo en algún IDE.

ME DESPIDO Y CUIDENSE, LES DESEO LO MEJOR DE LO MEJOR A TODOS :).