

# Docker para usar SMBD

Eider Julián Viveros - 2059308

Santiago Gallardo-2058810

Joaquín Carvajal-2059128

Mauricio Lara- 2158513



# Que es Docker

- Docker es una plataforma que permite a los desarrolladores empaquetar aplicaciones en contenedores e imágenes.
  - Estos contenedores son autónomos y contienen todo lo necesario para ejecutar la aplicación, incluido el sistema operativo, las bibliotecas y las herramientas necesarias.
- Esto significa que la aplicación funcionará igual independientemente del entorno en el que se ejecute.

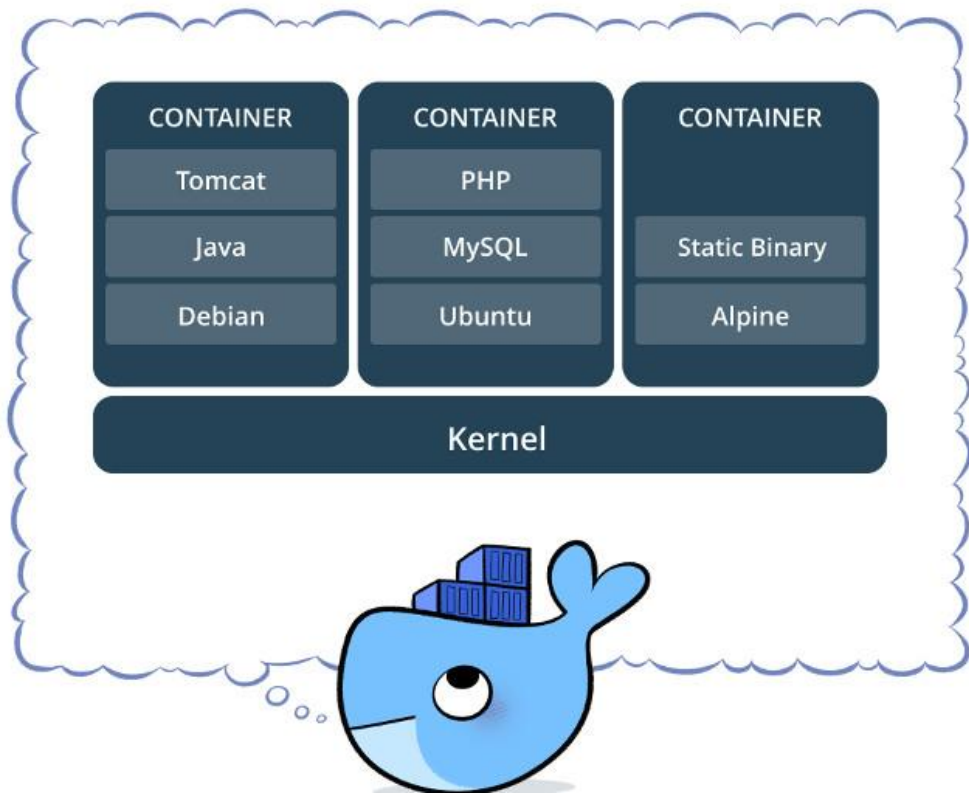




# Contenedores




- Los contenedores son instancias ejecutables de una imagen.
- Cada contenedor se ejecuta de manera aislada y tiene su propio conjunto de procesos, su propia red y su propio sistema de archivos montados a partir de la imagen.
- Aunque los contenedores se basan en una imagen, pueden tener un estado y datos que persisten entre las ejecuciones.
- Los contenedores son efímeros por naturaleza, lo que significa que puedes crearlos, eliminarlos y reemplazarlos fácilmente sin afectar al host ni a otros contenedores.



# Imágenes de Docker

- Las imágenes son la base de los contenedores.
- Son archivos inmutables que contienen un conjunto de instrucciones para crear un contenedor Docker.
- Las imágenes incluyen todo lo necesario para ejecutar una aplicación, como el código, las bibliotecas, las variables de entorno y los archivos de configuración.
- Las imágenes son portátiles y pueden ser compartidas entre diferente host de Docker. Y ser pueden almacenadas en repositorios locales o en la nube, como Docker Hub.

A pile of dark, 3D question marks on a dark background, creating a sense of mystery or inquiry.

# diferencia de imágenes y contenedores

En resumen, las imágenes de Docker son las recetas que contienen las instrucciones para crear contenedores mientras que los contenedores de Docker son las instancias en las que se va a hacer la ejecución de estas imágenes de Docker.

# Ventajas en el desarrollo

Docker ofrece varias ventajas a la hora de hacer el desarrollo de software, las más notables son:

- Aislamiento eficiente: es posible el ejecutar múltiples aplicaciones en contenedores sin preocuparse por conflictos en el sistema operativo.
- Portabilidad: una vez creado un contenedor, este se puede ejecutar en cualquier sistema que admita Docker simplificando la migración de aplicaciones facilitando la escalabilidad y implementación en la nube.
- Eficiencia de recursos: Los contenedores son eficientes en términos de recursos ya que comparten recursos con el sistema operativo. Esto significa que se pueden ejecutar más contenedores en una sola máquina física.

Por lo cual Docker facilita el desarrollo de software al proporcionar un entorno consistente para el desarrollo por lo que los desarrolladores pueden centrarse en escribir código sin preocuparse por las diferencias de entorno.



# Ventajas en despliegue

- Docker ofrece varias ventajas significativas en el despliegue de aplicaciones tales como:
  - Consistencia entre entornos: las aplicaciones se ejecutan de manera consistente en cualquier entorno por lo que, si una aplicación funciona en un contenedor Docker en tu máquina local, también funcionará en una máquina con Docker.
  - Despliegue rápido: puedes crear y lanzar contenedores en segundos siendo especialmente útil en entornos de implementación continua donde necesitas implementar nuevas versiones de una aplicación con frecuencia.
  - Docker y Kubernetes: estas tecnologías trabajan juntas en el despliegue ya que Docker se encarga de empaquetar las aplicaciones en contenedores y Kubernetes se encarga de gestionar esos contenedores en un clúster.
  - Mayor modularidad: El desarrollo con contenedores es ideal para un enfoque basado en microservicios para el diseño de aplicaciones complejas que se dividen en unidades más discretas y pequeñas facilitando el despliegue y la gestión de cada componente de manera individual mejorando la eficiencia y la escalabilidad.



# Ejemplo de uso

- Spotify: Spotify buscaba un método de despliegue que fuera rápido y sencillo con la mejor tolerancia a fallos.
- En su búsqueda llegó a la docker con el 2014 donde expusieron su anterior método de trabajo basado en las maquinas debian, donde los administradores realizaban conexiones ssh a cada nodo o maquina virtual para tener las dependías al día.
- Para después mostrar como con Docker los desarrolladores construían y probaban el despliegue de sus aplicaciones localmente y en un servidor de producción ahorrándose todos esos problemas



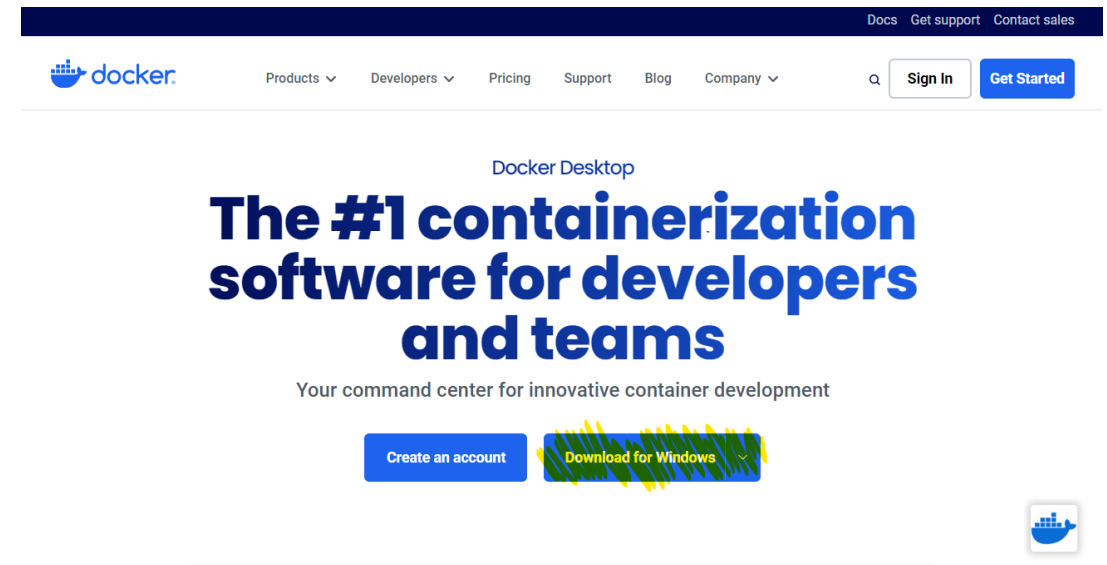


# Instalar Docker Desktop

Lo primero es entrar a para descargar el ejecutable:

<https://www.docker.com/products/docker-desktop/>

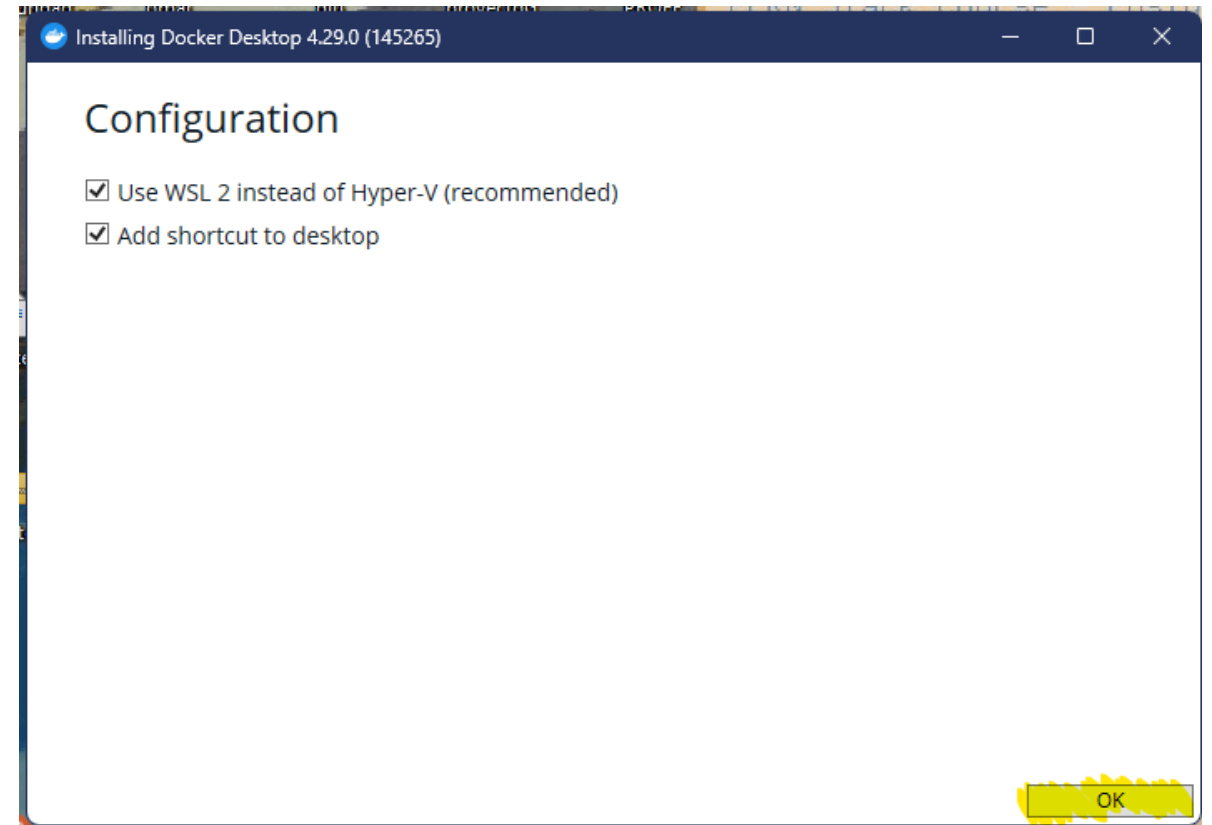
Donde nos descargara un archivo ejecutable llamado Docker desktop installer.exe



# Configurar instalación de docker

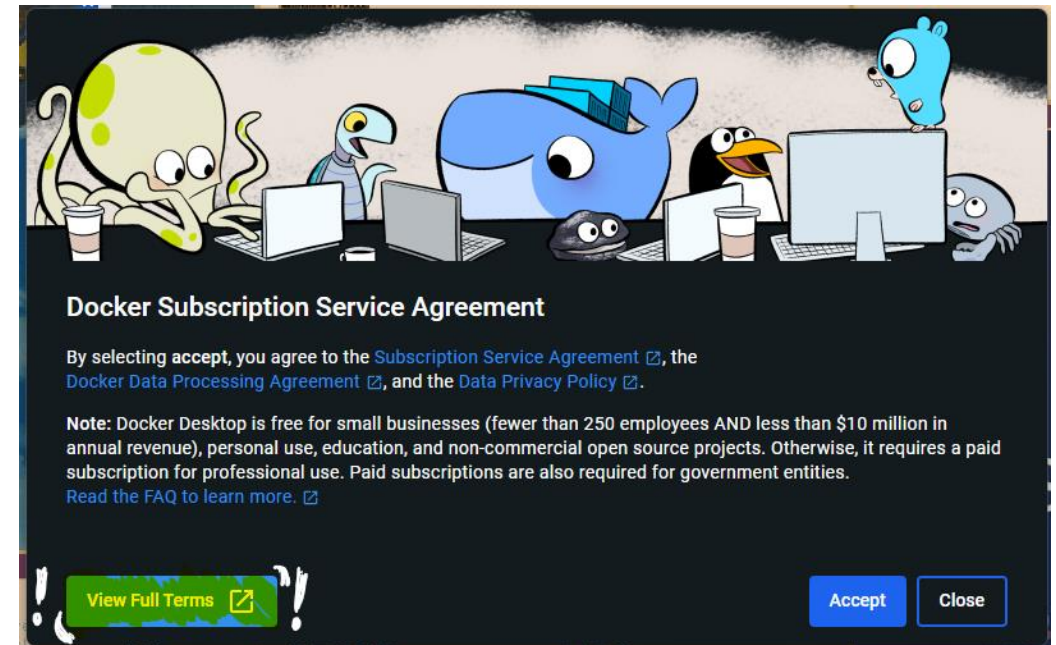
---

- Se recomienda Wsl 2(Windows subsystem for Linux 2) porque :
- ofrece un mejor rendimiento en las cargas de trabajo en el desarrollo porque su arquitectura es más liviana
- Wsl 2 está diseñado para tener una integración perfecta para permitirnos ejecutar herramientas y aplicaciones de Linux desde directamente la línea de comandos



# Ejecutar Docker desktop

- Al terminar las descargas nos quedara el siguiente icono de Docker desktop.
- Que al abrirlo por primera vez nos saldrá el Docker subscription service agreement que es obligatorio aceptar por los cual recomendamos leerlo, pero que en pocas palabras lo más importante lo de dice en la nota que es el cómo es un servicio gratis para empresas pequeñas o estudiantes.
- Después de aceptar tendrás una encuesta donde preguntaran por tu rol y para que usaras Docker la cual es opcional



# Instalar una imagen SQL

- Para instalar la imagen SQL entramos a [https://hub.docker.com/\\_/mysql](https://hub.docker.com/_/mysql) que es el repositorio de MySQL y copiamos el código y lo pegamos en la terminal de comandos Windows:
- `docker pull mysql`, nota: se puede agregar al final de la línea `:latest` para descargue la última versión

```
latest: Pulling from library/mysql
fcbdc4090331: Pull complete
95eb5073c36f: Pull complete
5e5ba0e6412a: Pull complete
ee4654eb29b7: Pull complete
1d2218160b86: Pull complete
bbe712936412: Pull complete
4a6efa152609: Pull complete
0f7c3b67bb2b: Pull complete
7ccd17e83c8a: Pull complete
04bf2c116556: Pull complete
Digest: sha256:4a4e5e2a19aab7a67870588952e8f401e17a330466ecfc55c9acf51196da5bd0
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```

## What's Next?

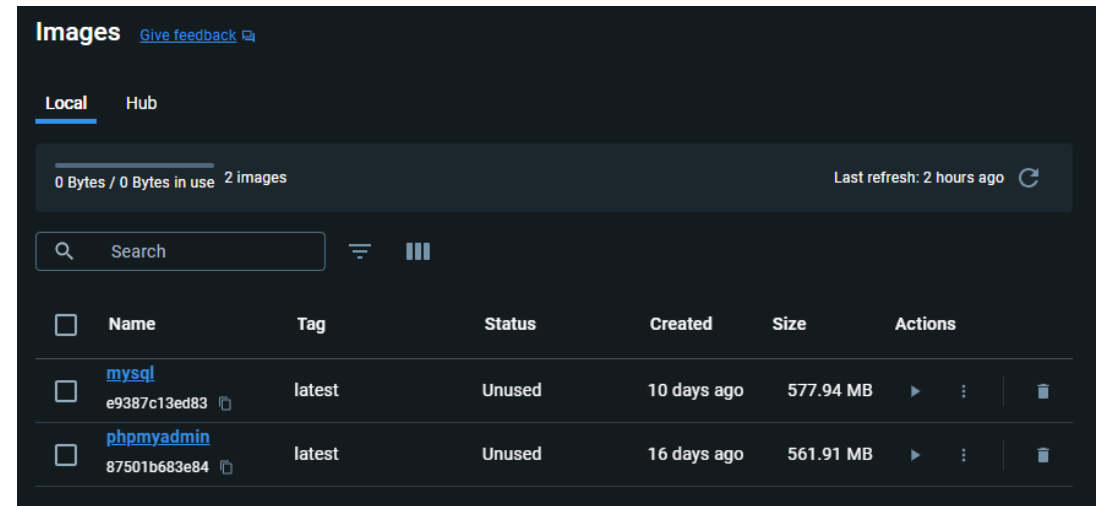
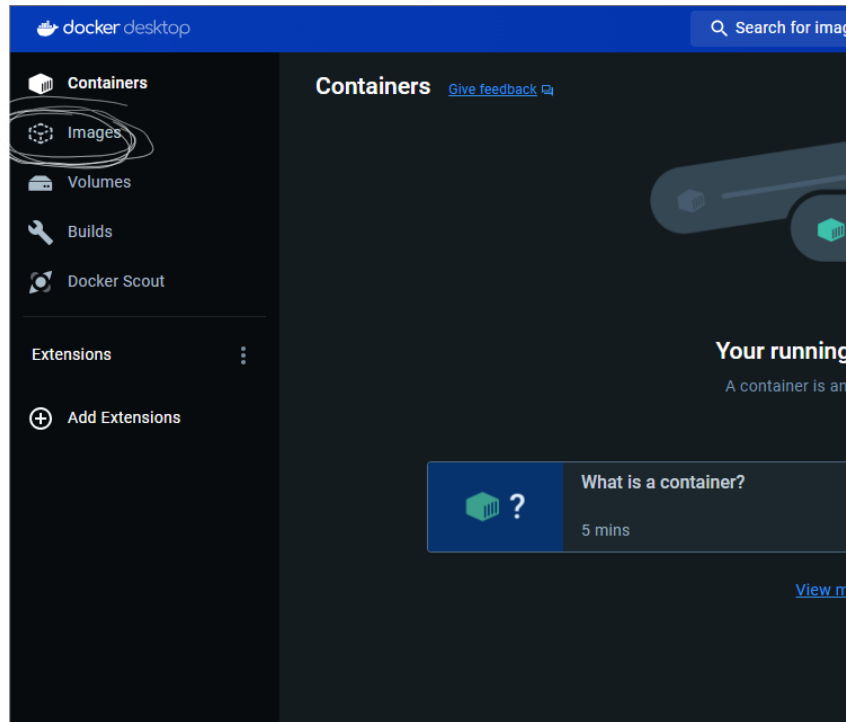
View a summary of image vulnerabilities and recommendations → `docker scout quickview mysql:latest`



# Instalar una imagen PHPMYADMIN

- Para instalar la imagen SQL entramos a [https://hub.docker.com/\\_/phpmyadmin](https://hub.docker.com/_/phpmyadmin) que es el repositorio de MySQL y copiamos el código y lo pegamos en la terminal de comandos Windows:
- `docker pull phpmyadmin`, nota: se puede agregar al final de la línea `:latest` para descargue la última versión

```
latest: Pulling from library/phpmyadmin
b0a0cf830b12: Pull complete
c93478d47932: Pull complete
e74cc574d0d2: Pull complete
e4782e138a90: Pull complete
cfeec87621ae: Pull complete
c1badcd002c0: Pull complete
e0d463a60cb6: Pull complete
d1ad50b335f5: Pull complete
98c64971444a: Pull complete
ba7d8962b7e1: Pull complete
e620d8fafd56: Pull complete
2038239aa3e1: Pull complete
df99eaff6870: Pull complete
1d61e18908d9: Pull complete
a503059f17a4: Pull complete
fb51dbc98e8f: Pull complete
a1b6a14258b3: Pull complete
cf8ad639c873: Pull complete
Digest: sha256:57de6b3d028cb77bca69523ab2f65a61b23d40db5f4c1c1163b7b16dc29e6a3d
Status: Downloaded newer image for phpmyadmin:latest
docker.io/library/phpmyadmin:latest
```



## Verificar instalación de imágenes

- Para verificar si la imagen se descargó correctamente entramos a docker desktop, al apartado de imágenes y tienen que salirnos las imágenes que descargamos

# Montar un contenedor mysql y phpmyadmin

- De mysql:

(docker run -d --name mysql-container -e MYSQL\_ROOT\_PASSWORD=12345 -p 5432:3306 mysql:latest)

```
PS C:\Users\jmlm1> docker run -d --name mysql-container -e MYSQL_ROOT_PASSWORD=12345 -p 5432:3306 mysql:latest  
f15bd4792cda7bee1f2bfc4abac34d2ff6a798b06425fc40eb779254152f4f30
```

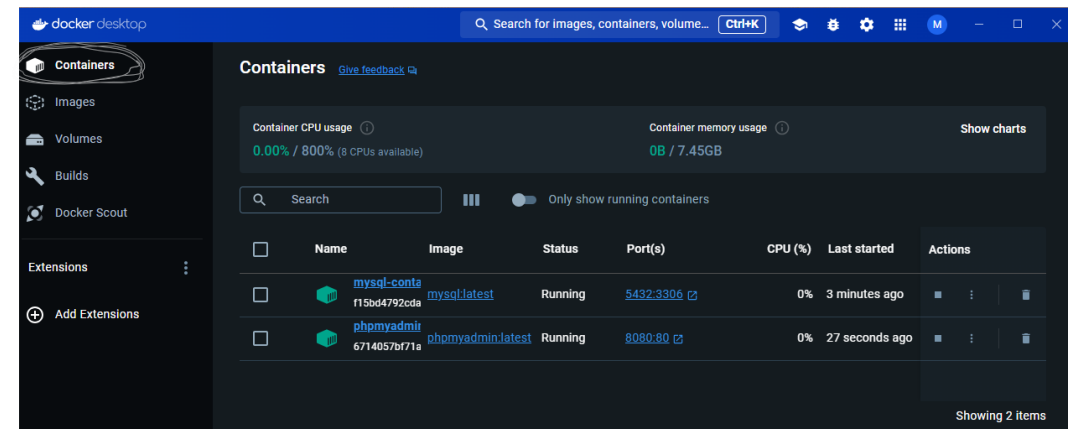
- De phpmyadmin:

(docker run -d --name phpmyadmin-container --link mysql-container:db -p 8080:80 phpmyadmin:latest)

```
PS C:\Users\jmlm1> docker run -d --name phpmyadmin-container --link mysql-container:db -p 8080:80 phpmyadmin:latest  
6714057bf71a96c18c35cbac49b161d5a39890ef52a96b047f54819695e39327
```

# Iniciar contenedores y usarlos

- Para verificar y usar los contenedores que creamos entramos nuevamente a docker desktop, al apartado de contenedores.
- Para iniciarlos empezamos con el contenedor mysql y después el phpmyadmin.
- Para ya usarlos en los contenedores le damos al puerto de phpmyadmin donde el usuario base será root y la clave será lo que pusimos en `MYSQL_ROOT_PASSWORD = 12345`,



<input type="checkbox"/>	Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
<input type="checkbox"/>	mysql-conta f15bd4792cda	mysql:latest	Running	5432:3306	0.76%	4 minutes ago	[Stop] [Refresh] [Delete]
<input type="checkbox"/>	phpmyadmin 6714057bf71a	phpmyadmin:latest	Running	8080:80	0%	2 minutes ago	[Stop] [Refresh] [Delete]