INSTALACIÓN DE DOCKER, SUGERENCIAS INICIALES.

1.- Instalar Docker según tus sistemas e instrucciones de la web:

ENLACES: www.docker.com

2.- Para familiarizarte se aconseja ejecutar el ejemplo de *docker/whalesay*, tal como veréis en las instrucciones de instalación de Docker.

Como ejemplo, te mostramos un típico error de la instalación de Docker en un Ubuntu. Cuando ejecutamos el comando Docker recibiremos este error además de otros mensajes:

"*WARNING: Error loading config file:/home/user/.docker/config.json - stat /home/user/.docker/config.json: permission denied*"

Esto quiere decir que te faltan permisos de usuario para poder ejecutar las acciones.

Para ello cambia los permisos ejecutando en consola:

```
sudo chown "$USER":"$USER" /home/"$USER"/.docker -R
sudo chmod g+rwx "/home/$USER/.docker" -R
```

3.- Una vez familiarizados te aconsejamos que borres los imágenes de ejemplo (docker/whalesay).

4.- Como hack te aconsejamos que sigas también los pasos de post-inslaación de Docker porque aprenderás a darle permisos root a docker para no estar continuamente introduciendo la contraseña... muy a mi pesar (Raul dixit).

DESCARGAR EL GIT DE CONSENSUS

1.- Descargamos el git de consensus:

```
git clone https://github.com/devscola/consensus
```

DOCKER

Docker separa el concepto de la imagen física (archivos que la conforman y configuraciones), del contenedor que está ejecutandose en tu sistema. Podriamos hacer una comparación muy simplista entendiendo que la imagen física sería como una ISO y el contenedor sería casi como una máquina virtual sin interfaz.

Vamos a tener imágenes físicas y contenedores ejecutándose. Los comandos que utilizarás habitualmente en Docker son:

Listar las imágenes:

```
docker images
```

Poner en marcha un contendor: ~ docker run [opciones] ~ (más adelante explicamos el comando que usaremos para poner en marcha nuestro docker)

Borrar imágenes que ya no nos sirven:

```
docker rmi docker/whalesay
```

Otro posible inconveniente para borrar una imagen es que no tenga asignado un tag, por lo que deberás mirar su IMAGE ID y ponerla como nombre de la imagen, ejemplo:

```
docker rmi 54e3454d8dd6
```

Si no nos deja borrar la imagen es porque esta utilizándose, deberás averiguar porqué y si puedes borrarla, si pese a todo quieres borrarla sí o sí, entonces ejecuta:

```
docker rmi docker/whalesay --force
```

Revisar los contenedores que hay corriendo:

```
docker ps -a
```

Borrar el contenedor:

```
docker rm whalesay
```

Cada vez que hagamos exit en nuestro docker, tendremos que borrar el contenedor y volverlo a poner en marcha.

```
docker rm nombre_del_contenedor
docker run -it --name consensus -v  $(pwd):/opt/consensus elferrer/ruby bundle exec rspec
```

Como último apunte indicar que los contenedores necesitan siempre que exista su imagen origen, por ello es habitual que tengas varios contenedores en marcha desde la misma imagen.

A. PREPARACIÓN DEL DOCKER, SOLO CADA VEZ QUE ACTUALICEMOS LA IMAGEN

1.- Nos situamos en la carpeta del git de consensus.

2.- Crear el docker file:

```
nano Dockerfile
```

En él deberemos escribir las reglas de ejecución del Docker. En nuestro caso, hemos escrito el siguiente contenido:

```
FROM ruby:2.4.0-alpine

RUN mkdir -p /opt/consensus
```

```
WORKDIR /opt/consensus
COPY Gemfile Gemfile
RUN apk update
RUN apk add g++
RUN apk add make
RUN bundle install
```

Explicación del contenido:

"FROM": imagen de docker que utilizamos como base para crear nuestro docker, esta imagen es la base de nuestro docker. Al poner una versión concreta y no utilizar "latest" nos evitamos que nuestra imagen se esté actualizando constantemente con las novedades de la imagen origen, salvo que esta misma versión tenga una actualización (parches de seguridad, etc.). "RUN": Comando que se ejecuta desde la consola. "WORKDIR": Directorio de trabajo. "COPY": Indicamos al Docker que coja un archivo de nuestra máquina local y lo copie al contenedor que inicializaremos.

3.- Revisar que existe el archivo Gemfile.

4.- Generamos la imagen de Docker:

```
docker build -t elferrer/ruby .
```

5.- Hay que crear el tag de la imagen, pero ¿qué es el tag?: El tag es el identificativo de la imagen que hemos creado para no tener que estar continuamente escribiendo el *image id*:

```
docker tag image_id elferrer/ruby
```

6.- Nos identificamos en la plataforma docker (hub.docker.com):

```
docker login
```

7.- Subimos la imagen:

```
docker push usuario/imagen
```

B. DESCARGAR EL DOCKER DE CONSENSUS

B1. DESCARGA Y EJECUCIÓN DEL DOCKER EN LINUX

1.- Hay que ir a la web: hub.docker.com y buscar la imagen: elferrer/ruby , o puedes directamente descargarla:

```
docker pull elferrer/ruby
```

2.- Ponemos en marcha la imagen ejecutando el test:

```
docker run -it --name consensus -v  $(pwd):/opt/consensus elferrer/ruby bundle exec rspec
```

B2. DESCARGA Y EJECUCIÓN DEL DOCKER PARA WINDOWS

1.- Hay que ir a la web: hub.docker.com y buscar la imagen: elferrer/ruby , o puedes directamente descargarla:

```
docker pull elferrer/ruby
```

2.- Debemos compartir el disco C: (el disco en el que tienes la imagen) configurando *settings* en la aplicacion Docker.

3.- Ponemos en marcha la imagen:

```
docker run -it --name consensus -v  c:/carpeta/compartida:/opt/consensus elferrer/ruby /bin/
```