Redes de Computadoras

Facultad de Matemática y Computación

Curso 2025–2026 — Material Complementario

Guía de Introducción a Docker CLI

Docker es una plataforma para desarrollar, enviar y ejecutar aplicaciones en contenedores. Permite empaquetar aplicaciones con todas sus dependencias, asegurando que se ejecuten de manera consistente en diferentes entornos.

Instalación de Docker

Para instalar Docker, sigue las instrucciones oficiales según tu sistema operativo:

- Linux: https://docs.docker.com/engine/install/
- Windows: https://docs.docker.com/desktop/install/windows-install/
- Mac: https://docs.docker.com/desktop/install/mac-install/

Para verificar la instalación:

```
docker --version
```

Conceptos Básicos

Imágenes

Las imágenes son plantillas inmutables utilizadas para crear contenedores.

- Listar imágenes locales:

```
docker images
```

Explicación: Muestra todas las imágenes almacenadas localmente.

- Descargar una imagen desde Docker Hub:

```
docker pull <nombre-imagen>
```

Explicación: Descarga la imagen especificada desde el repositorio Docker Hub.

Contenedores

Los contenedores son instancias ejecutables de imágenes.

- Crear y ejecutar un contenedor interactivo:

```
docker run -it <nombre-imagen> /bin/bash
```

Explicación: `-i` mantiene la entrada estándar abierta y `-t` asigna una terminal al contenedor.

- Listar contenedores en ejecución:

```
docker ps
```

Explicación: Muestra los contenedores actualmente en ejecución.

- Listar todos los contenedores:

```
docker ps -a
```

Explicación: Muestra todos los contenedores, incluidos los detenidos.

- Detener un contenedor:

```
docker stop <id-contenedor>
```

Explicación: Envía la señal de parada al contenedor.

- Eliminar un contenedor:
  docker rm
  Explicación: Borra un contenedor detenido de la lista de contenedores.

Dockerfile: Creación de Imágenes Personalizadas

Un Dockerfile es un script que contiene instrucciones para construir una imagen personalizada.

Ejemplo de Dockerfile para una aplicación Node.js:

```
# Usar una imagen base de Node.js
FROM node:18

# Establecer directorio de trabajo
WORKDIR /app

# Copiar archivos de la aplicación
COPY . .

# Instalar dependencias
RUN npm install

# Exponer el puerto de la aplicación
```

```
EXPOSE 3000

# Comando por defecto
CMD ["node", "app.js"]
```

Construcción y ejecución:

```
docker build -t mi-aplicacion .
docker run -p 3000:3000 mi-aplicacion
```

Imágenes Populares

Algunas imágenes útiles en Docker Hub:

- ubuntu
- alpine
- node
- python
- postgres
- redis

Flujo de Trabajo Completo

Ejemplo de desarrollo de una aplicación en contenedor:

1. Escribir un `Dockerfile`.
2. Construir la imagen (`docker build`).
3. Ejecutar un contenedor (`docker run`).
4. Probar la aplicación en el contenedor.

Ejercicios Prácticos con Respuestas

1. Crear un Dockerfile para ejecutar un script en Shell.

Código fuente (`script.sh`):

```
#!/bin/sh
echo "Hola desde un script en contenedor!"
```

Dockerfile:

```
FROM alpine:latest
WORKDIR /app
COPY script.sh .
RUN chmod +x script.sh
CMD ["/bin/sh", "script.sh"]
```

Construcción y ejecución:

```
docker build -t shell-app .
docker run shell-app
```

2. Crear un Dockerfile para una aplicación en C y ejecutarla.

Código fuente (`hello.c`):

```
#include <stdio.h>
int main() {
    printf("Hola desde un contenedor Docker!\n");
    return 0;
}
```

Dockerfile:

```
FROM gcc:latest
WORKDIR /app
COPY hello.c .
RUN gcc hello.c -o hello
CMD ["./hello"]
```

Construcción y ejecución:

```
docker build -t mi-c-app .
docker run mi-c-app
```

3. Manipulación de imágenes y contenedores.

- Listar solo las imágenes:

```
docker images
```

- Listar solo los contenedores (incluidos los detenidos):

```
docker ps -a
```

- Eliminar todos los contenedores detenidos:

```
docker container prune
```

- Eliminar todas las imágenes sin nombre ():

```
docker image prune -a
```

Referencias

- Documentación Oficial de Docker: https://docs.docker.com/