

1. Introducción

Docker se ha convertido en una herramienta esencial para facilitar el despliegue de entornos de prueba, especialmente combinándolo con frameworks de automatización como **Selenium** y **Python**. Permite tener ambientes de prueba consistentes, escalables y fáciles de administrar.

2. ¿Docker requiere instalación?

Sí.

Es necesario instalar **Docker** en tu máquina. Puedes descargarlo desde Docker Desktop.

3. Diferencias entre Docker y Docker Compose

Docker

Corre contenedores individuales.

Comandos: docker run, docker stop.

Docker Compose

Orquesta múltiples contenedores en un solo archivo .yaml.

Comando: docker-compose up, docker-compose down.

4. Uso de Docker Compose para Selenium Grid con VNC (Selenium 3)

Usando imágenes *-debug que traen un servidor **VNC** integrado.

yaml

CopiarEditar

```
version: "3"
```

```
services:
```

```
  selenium-hub:
```

```
    image: selenium/hub:3.141.59
```

```
    container_name: selenium-hub
```

```
    ports:
```

```
      - "4444:4444"
```

chrome-debug:

image: selenium/node-chrome-debug:3.141.59

container_name: chrome-debug

depends_on:

- selenium-hub

environment:

- HUB_HOST=selenium-hub

ports:

- "5900:5900" # VNC Chrome

shm_size: 2gb

firefox-debug:

image: selenium/node-firefox-debug:3.141.59

container_name: firefox-debug

depends_on:

- selenium-hub

environment:

- HUB_HOST=selenium-hub

ports:

- "5901:5900" # VNC Firefox

shm_size: 2gb

- **Importante:** Firefox mapea el puerto 5901:5900.

5. Uso de Docker Compose para Selenium Grid con Selenium 4 (sin imágenes debug)

En Selenium 4, las imágenes **no incluyen VNC**.

yaml

CopiarEditar

version: "3"

services:

selenium-hub:

image: selenium/hub:4.20.0

container_name: selenium-hub

ports:

- "4442:4442"

- "4443:4443"

- "4444:4444"

chrome:

image: selenium/node-chrome:4.20.0

container_name: chrome

depends_on:

- selenium-hub

environment:

- SE_EVENT_BUS_HOST=selenium-hub

- SE_EVENT_BUS_PUBLISH_PORT=4442

- SE_EVENT_BUS_SUBSCRIBE_PORT=4443

shm_size: 2gb

firefox:

image: selenium/node-firefox:4.20.0

container_name: firefox

depends_on:

- selenium-hub

environment:

- SE_EVENT_BUS_HOST=selenium-hub

- SE_EVENT_BUS_PUBLISH_PORT=4442

- SE_EVENT_BUS_SUBSCRIBE_PORT=4443

shm_size: 2gb

6. Alternativa moderna: Selenoid

¿Qué es Selenoid?

- Un servidor de automatización de navegadores liviano basado en Docker.
 - Permite lanzar instancias de navegadores en contenedores bajo demanda.
 - Permite ver navegadores vía **VNC** y grabar videos.
-

♦ Cómo instalar Selenoid con el Configuration Manager

Paso 1: Descargar cm (Configuration Manager):

bash

CopiarEditar

```
curl -s https://aerokube.com/cm/latest/cm_linux_amd64 -o cm
```

```
chmod +x cm
```

(En Windows puedes descargar el ejecutable .exe desde aerokube/cm).

♦ Cómo conectar cm con Docker e iniciar Selenoid

1. Inicializar Selenoid:

bash

CopiarEditar

```
./cm selenoid start --vnc
```

Esto hará:

- Descargar el contenedor de Selenoid.
- Descargar imágenes de navegadores compatibles automáticamente.
- Levantar Selenoid en el puerto 4444.

2. Inicializar Selenoid UI:

bash

CopiarEditar

```
./cm selenoid-ui start
```

Esto levantará una **interfaz gráfica** en el puerto 8080.

◆ **Cómo descargar imágenes de navegadores en Selenium**

- Puedes listar navegadores disponibles:

```
bash
```

```
CopiarEditar
```

```
./cm selenium browsers
```

- Puedes instalar navegadores específicos:

```
bash
```

```
CopiarEditar
```

```
./cm selenium update --browsers chrome
```

(Esto descarga todas las versiones disponibles de Chrome compatibles).

◆ **Configuración de browsers.json manual**

Si deseas personalizarlo:

```
json
```

```
CopiarEditar
```

```
{  
  "chrome": {  
    "default": "118.0",  
    "versions": {  
      "118.0": {  
        "image": "selenium/vnc_chrome:118.0",  
        "port": "4444",  
        "path": "/",  
        "tmpfs": {"/tmp": "size=512m"}  
      }  
    }  
  },  
}
```

```
"firefox": {  
  "default": "118.0",  
  "versions": {  
    "118.0": {  
      "image": "selenoid/vnc_firefox:118.0",  
      "port": "4444",  
      "path": "/",  
      "tmpfs": {"/tmp": "size=512m"}  
    }  
  }  
}
```

◆ Docker Compose de Selenoid + Selenoid UI (alternativa manual)

yaml

CopiarEditar

version: "3"

services:

selenoid:

image: aerokube/selenoid:latest-release

container_name: selenoid

network_mode: bridge

ports:

- "4444:4444"

volumes:

- "/var/run/docker.sock:/var/run/docker.sock"

- "./browsers.json:/etc/selenoid/browsers.json"

command: [

```
"-limit", "10",  
"-timeout", "3m"  
]
```

selenoid-ui:

image: aerokube/selenoid-ui:latest-release

container_name: selenoid-ui

network_mode: bridge

depends_on:

- selenoid

ports:

- "8080:8080"

environment:

- SELENOID_URI=http://selenoid:4444

7. ¿Qué es VNC?

Virtual Network Computing (VNC) permite acceder al navegador en ejecución dentro de un contenedor de forma gráfica.

Cientes recomendados:

- **RealVNC:**
 - VNC Viewer (con 3rd party servers).
- **TightVNC:**
 - Alternativa open-source.
 - Instalador contiene cliente y servidor juntos.

8. Advertencia sobre imágenes *-debug

Las imágenes node-firefox-debug y node-chrome-debug están **deprecadas** después de Selenium 4.

9. ¿Qué es Docker Swarm?

Docker Swarm es la herramienta nativa de Docker para orquestar múltiples contenedores en clústeres.

Comandos básicos:

- Iniciar Swarm: `docker swarm init`
- Desplegar stack: `docker stack deploy -c docker-compose.yml nombre`
- Escalar servicios usando `deploy` en Compose.

Diferencias:

Docker Compose Docker Swarm

Solo local. Clúster distribuido.

No usa `deploy`. Usa `deploy` para réplicas.

Importante:

Sin `deploy`, el `docker-compose` clásico solo crea una instancia de cada navegador.

10. ¿Qué es Percy?

[Percy](#) permite **pruebas visuales automáticas** capturando capturas de pantalla durante las ejecuciones de prueba.

Planes:

- Gratuito: Hasta 5,000 capturas por mes.
- Planes pagos disponibles.

Requiere:

- Crear cuenta en Percy.
 - Obtener **TOKEN** personal.
 - Instalar SDK de Percy en el proyecto.
-

11. Estructura del Proyecto

bash

CopiarEditar

test-automation-project/

|

└─ docker-compose.yml


```
└─ browsers.json (Selenoid)
└─ tests/
|   └─ test_login.py
|   └─ test_checkout.py
└─ requirements.txt
└─ README.md
```

12. Esquema Visual: Flujo de Herramientas

plaintext

CopiarEditar

Test Automation Framework (Python + Selenium)

|

v

Selenium WebDriver

|

v

Docker Compose / Selenoid

|

+--> Selenium Hub o Selenoid

|

+--> Chrome Node (VNC)

+--> Firefox Node (VNC)

|

+--> Selenoid UI (opcional)

|

+--> Percy (Capturas Visuales)

13. 🚀 Flujo práctico de ejemplo: Selenoid + Selenium + Python

1. Levantar Selenoid:

bash

CopiarEditar

```
docker-compose up -d
```

2. Código Python para conectarse a Selenoid:

python

CopiarEditar

```
from selenium import webdriver
```

```
options = webdriver.ChromeOptions()
options.add_argument("--no-sandbox")
options.add_argument("--disable-dev-shm-usage")
```

```
driver = webdriver.Remote(
    command_executor="http://localhost:4444/wd/hub",
    options=options
)
```

```
driver.get("https://example.com")
print(driver.title)
```

```
driver.quit()
```

3. Ver ejecución en Selenoid UI:

- Acceder a <http://localhost:8080>.

14. Ejemplo práctico de integración de Percy

1. Instalar Percy Python SDK:

bash

CopiarEditar

```
pip install percy-selenium
```

2. Código de prueba integrando Percy:

```
python
```

```
CopiarEditar
```

```
from percy import percy_snapshot
```

```
from selenium import webdriver
```

```
options = webdriver.ChromeOptions()
```

```
options.add_argument("--no-sandbox")
```

```
options.add_argument("--disable-dev-shm-usage")
```

```
driver = webdriver.Remote(
```

```
    command_executor="http://localhost:4444/wd/hub",
```

```
    options=options
```

```
)
```

```
driver.get("https://example.com")
```

```
# Capturar snapshot visual usando Percy
```

```
percy_snapshot(driver, 'Homepage Example')
```

```
driver.quit()
```

3. Configurar variables de entorno:

- PERCY_TOKEN: tu token de proyecto Percy.

4. Correr las pruebas y ver resultados en Percy Dashboard.

✅ Conclusiones Finales

- Docker permite montar entornos reproducibles de automatización.
- Puedes usar Selenium Grid clásico o alternativas modernas como **Selenoid**.

- Percy facilita pruebas visuales sin esfuerzo.
- Con Swarm puedes escalar múltiples instancias de navegadores fácilmente.
- TightVNC y RealVNC permiten monitoreo visual en tiempo real.