

Por qué Playwright + Python

- Soporte para varios navegadores con primitivas modernas
- Modelo de localizador sólido y assertions basadas en la web
- El ecosistema Python se integra bien con los flujos de trabajo de control de calidad

Referencias: <https://playwright.dev/docs/intro> | <https://docs.pytest.org/en/stable/>

Flujo de trabajo principal del dramaturgo

- Inicie el navegador/contexto/página de forma determinista
- Usa localizadores resilientes y assertions explícitas
- Recopilar rastros/capturas de pantalla de fallas

Referencias: <https://playwright.dev/docs/intro> |
<https://playwright.dev/docs/test-assertions>

Patrones de integración de Pytest

- Usa fixtures para la reutilización del montaje/desmontaje
- Parametrizar pruebas para escalar la cobertura
- Usa marcadores para segmentar conjuntos por riesgo

Referencias: <https://docs.pytest.org/en/stable/>

Pruebas basadas en datos y conscientes del medio ambiente

- Leer configuración de variables de entorno
- Separar secretos del código fuente
- Usa un ciclo de vida de datos de prueba limpio por ejecución

Referencias: <https://12factor.net/config>

Depuración de pruebas de IU fallidas

- Reproducir con modo encabezado y rastros.
- Inspeccionar la red/consola antes de cambiar las assertions
- Corregir errores del producto antes de las pruebas de parcheo

Referencias: <https://playwright.dev/docs/trace-viewer-intro>

