Understanding Playwright Core Interfaces and First Tests

Andrejs Doronins

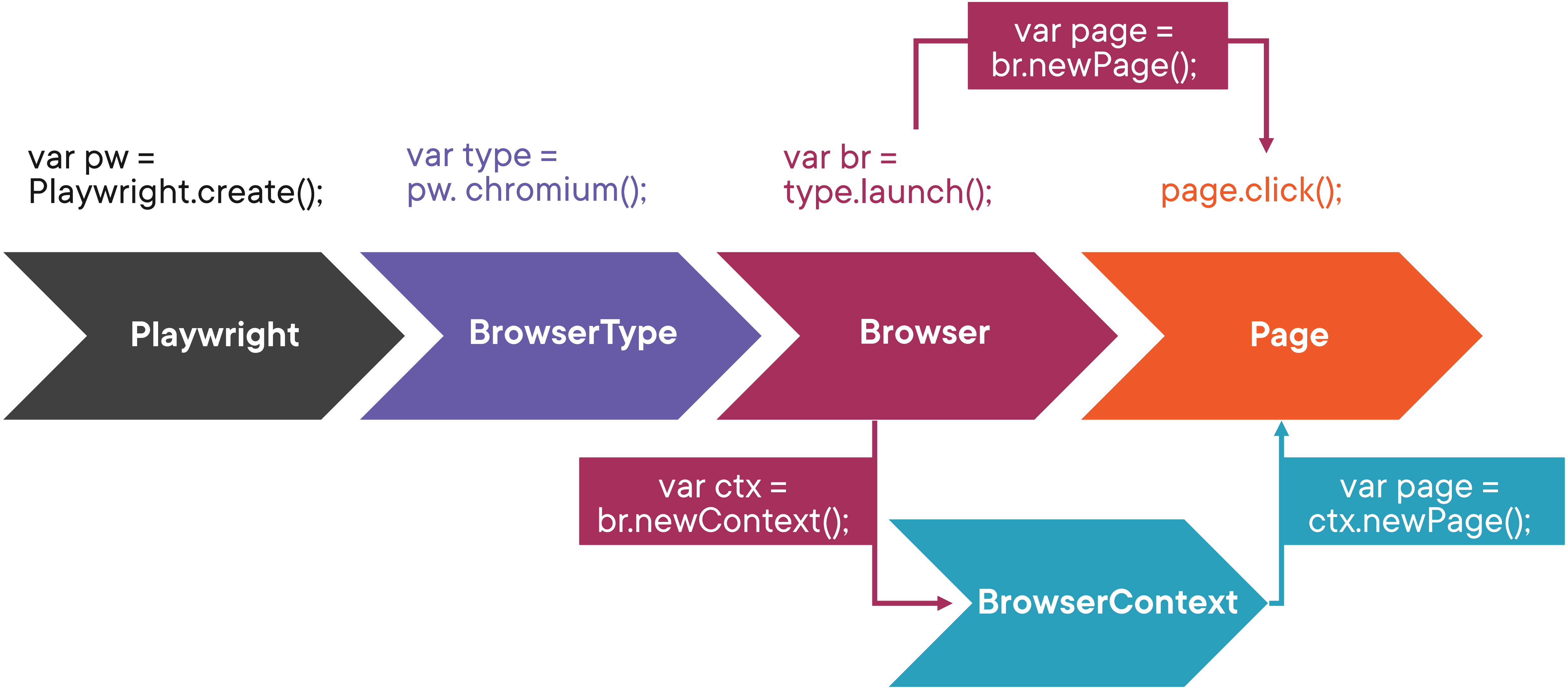
Overview

Review core Playwright interfaces

- including BrowserContext

Write tests

Refactor boilerplate code



```
var pw =
Playwright.create();
```

```
var type =
pw. chromium();
```

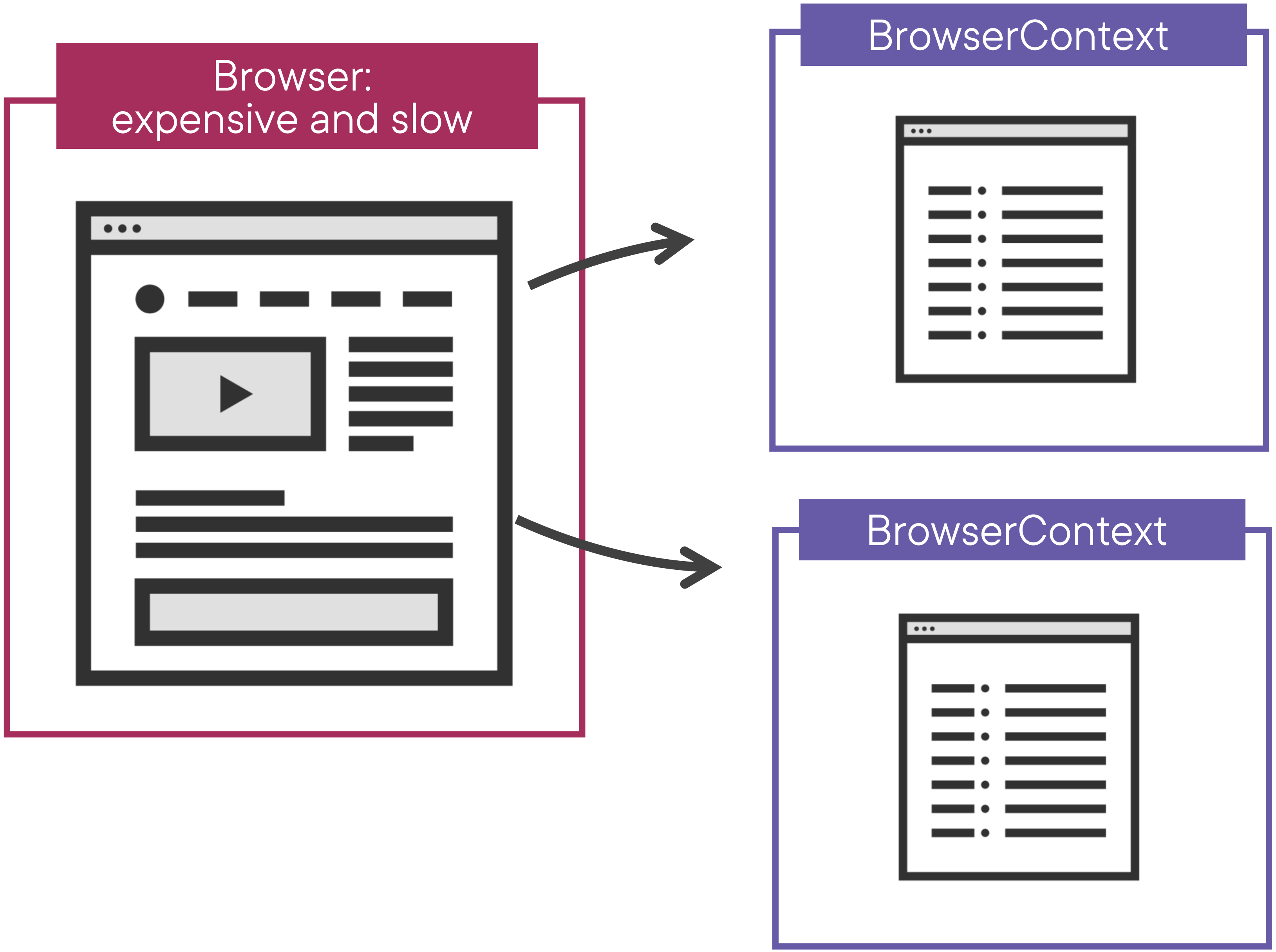
```
var br =
type.launch();
```

```
var page =
br.newPage();
```

```
page.click();
```

```
var ctx =
br.newContext();
```

```
var page =
ctx.newPage();
```



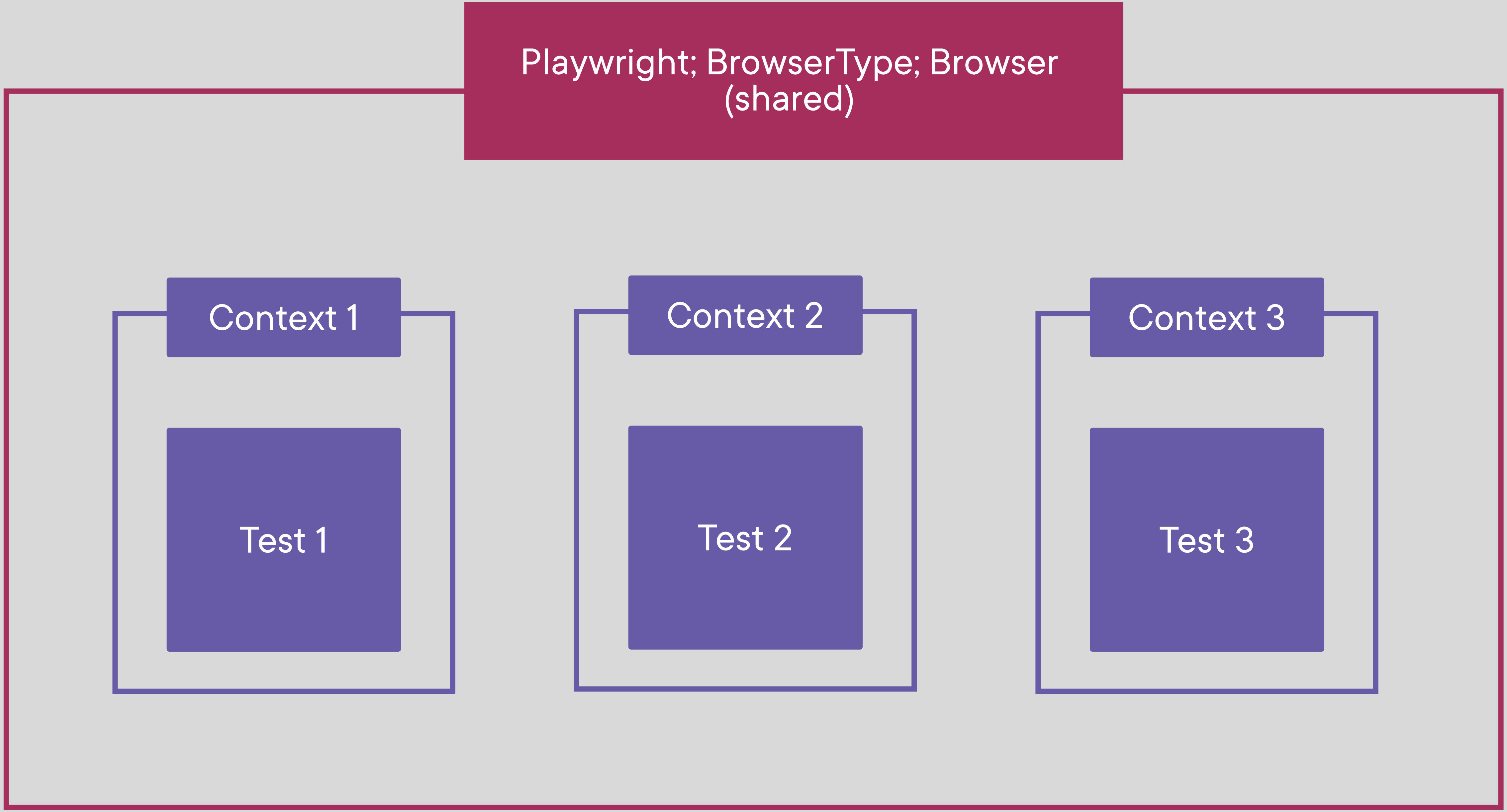
Browser:
expensive and slow

BrowserContext

BrowserContext

Benefits:

- fast and cheap
- isolated
- no shared cookies, local storage, session info, etc.



Playwright; BrowserType; Browser (shared)

Context 1

Test 1

Context 2

Test 2

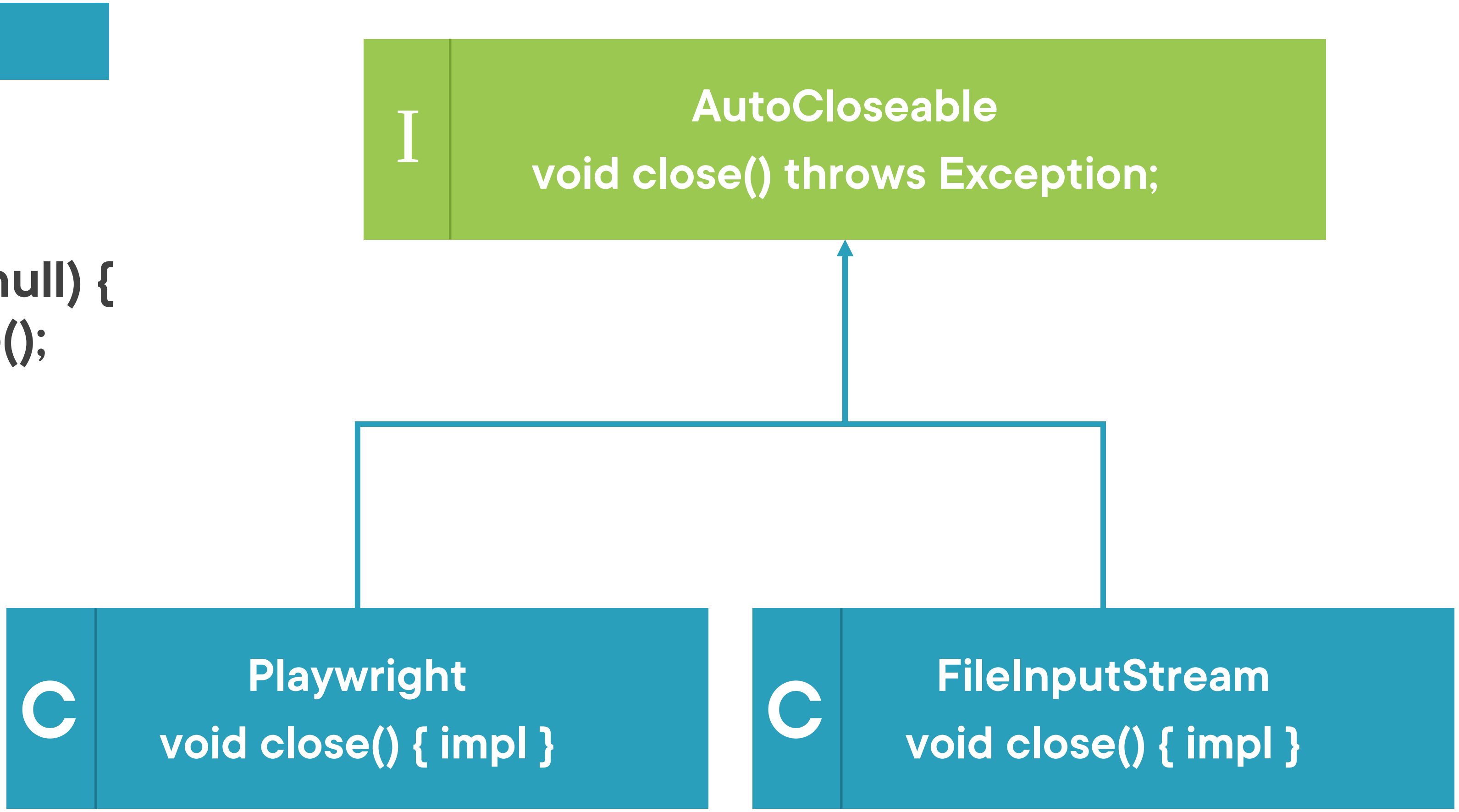
Context 3

Test 3

Java

I’ll do it for you

```
finally {
          if (res !=null) {
          res.close();
          }
}
```



I

AutoCloseable

void close() throws Exception;

C

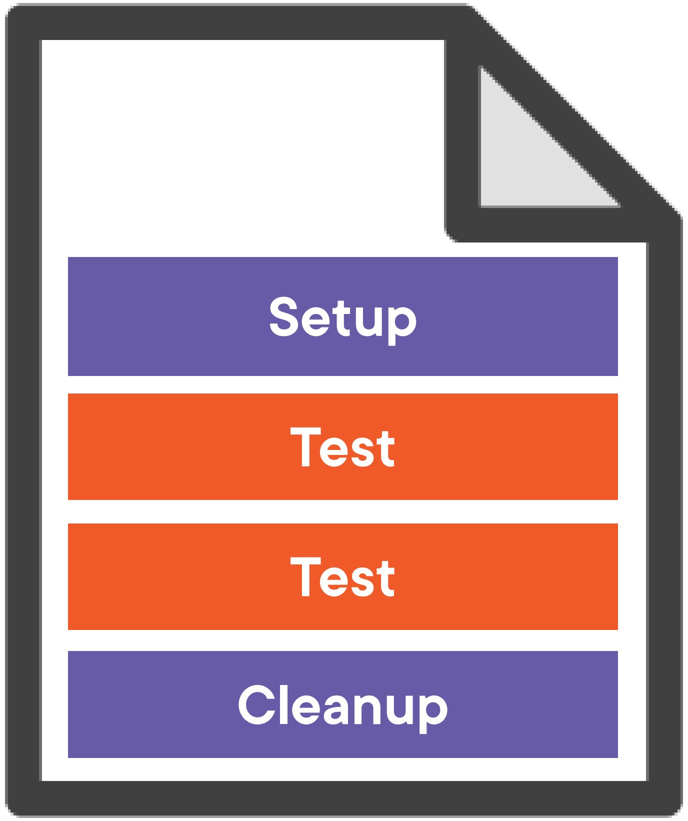
Playwright

void close() { impl }

C

FileInputStream

void close() { impl }



Setup

Test

Test

Cleanup

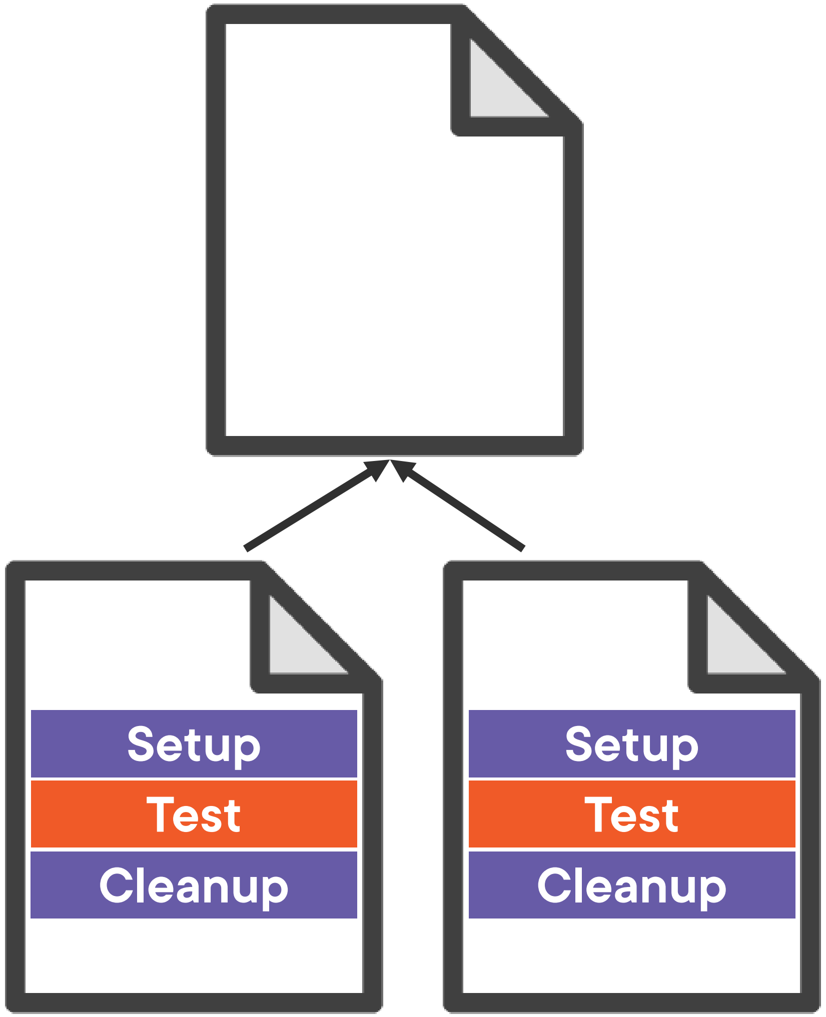
JUnit 5

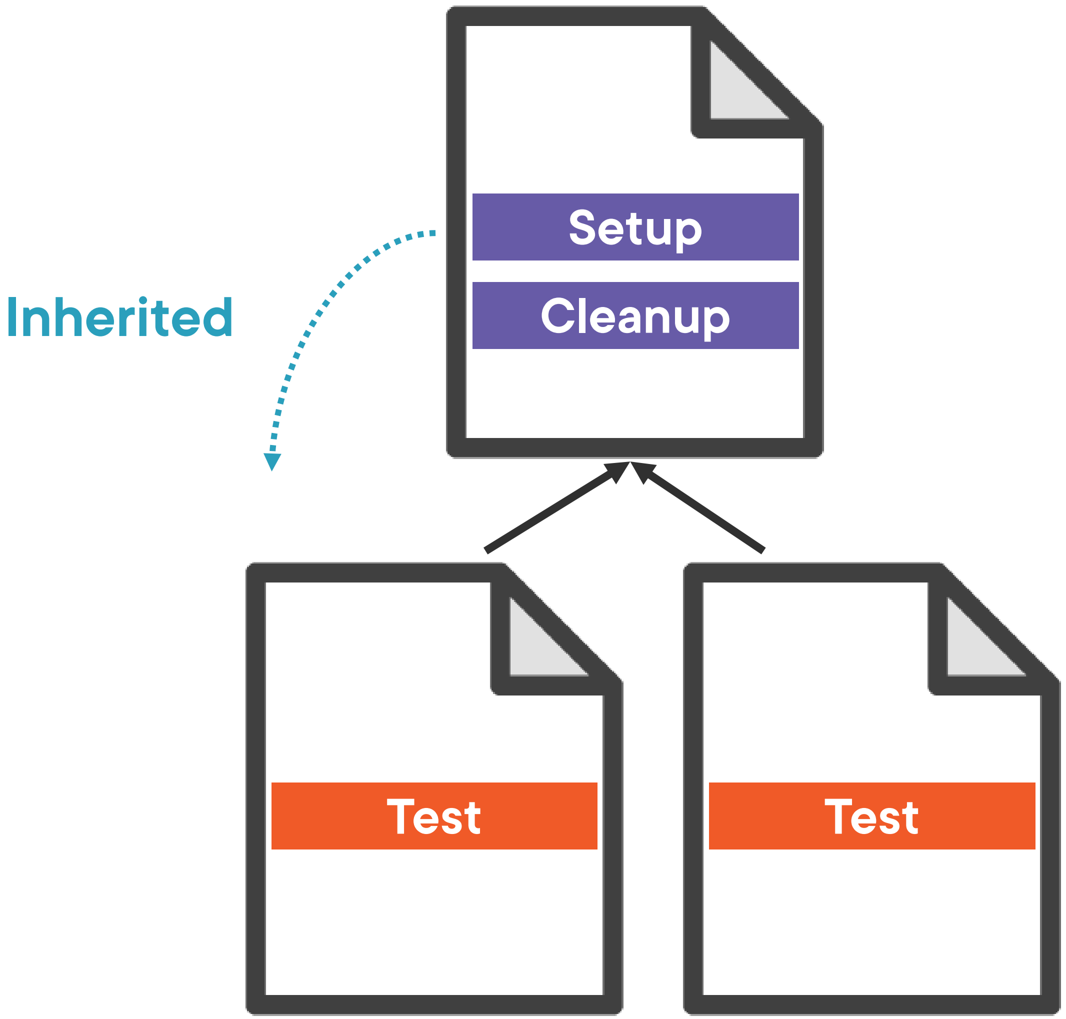
@BeforeAll
@BeforeEach
@AfterEach
@AfterAll

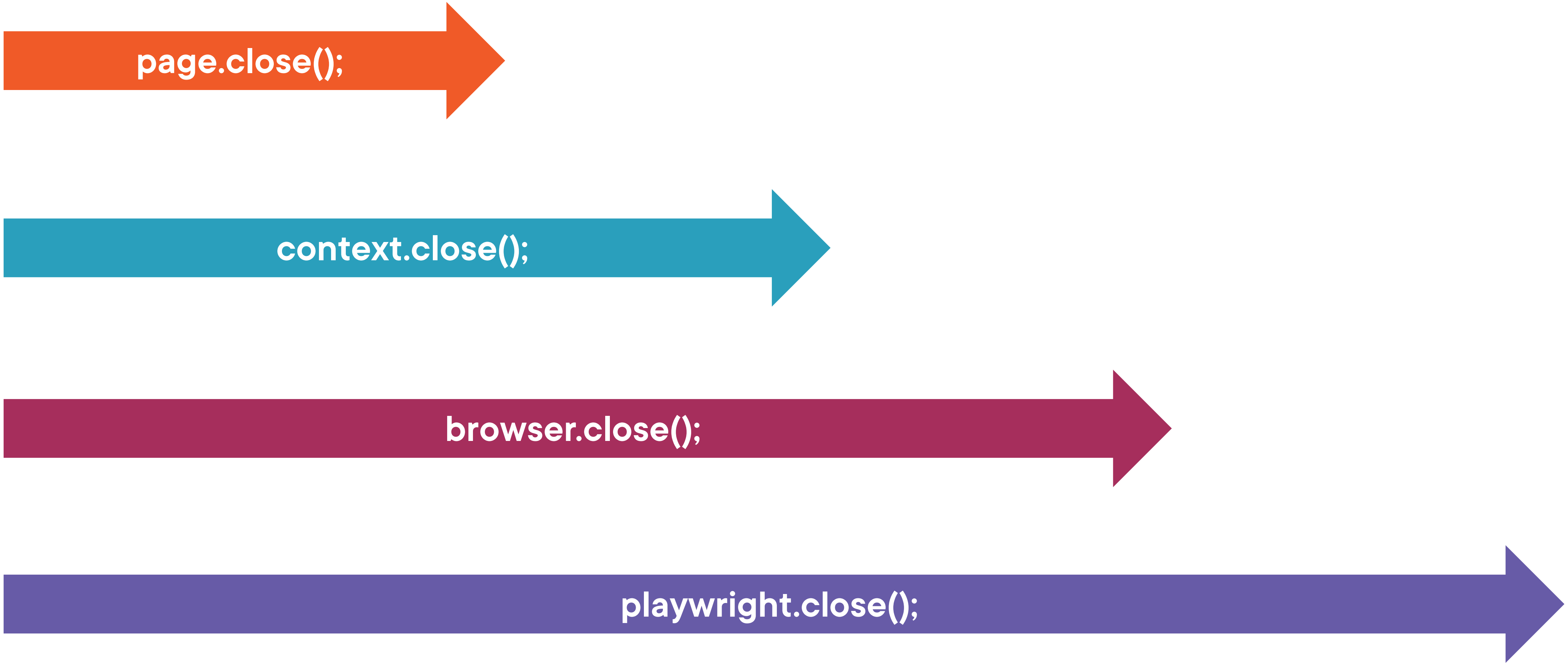
TestNG

@BeforeSuite
@BeforeMethod
@AfterMethod
@AfterSuite

(alwaysRun = true)


Setup
Test
Cleanup
Setup
Test
Cleanup


Setup
Cleanup
Inherited
Test
Test



```
page.close();
context.close();
browser.close();
playwright.close();
```

Summary

Playwright core interfaces:

- Playwright, BrowserType, Browser, BrowserContext, Page

Some implement the auto-closeable interface

- .close() should be invoked

Refactoring setup and cleanup

Up Next:
Using Playwright Basic Actions