DECLARATIVE E2E WITH WINDOW DRIVERS

E2E TESTS

E2E TESTS

- Expensive to write and maintain
- Fragile, prone to breaking when UI changes
- Often coupled to implementation details

COMMON E2E PAINPOINTS

- Writing tests is repetitive
- Code reuse?
  - Copy / Paste -> Duplication
  - Functions -> Not discoverable
  - Commands -> Lack structure

As is common in computer science, we can solve almost any problem by adding a layer of indirection

-- Roberto Vitillo, Understanding Distributed Systems

THE WINDOW DRIVER PATTERN

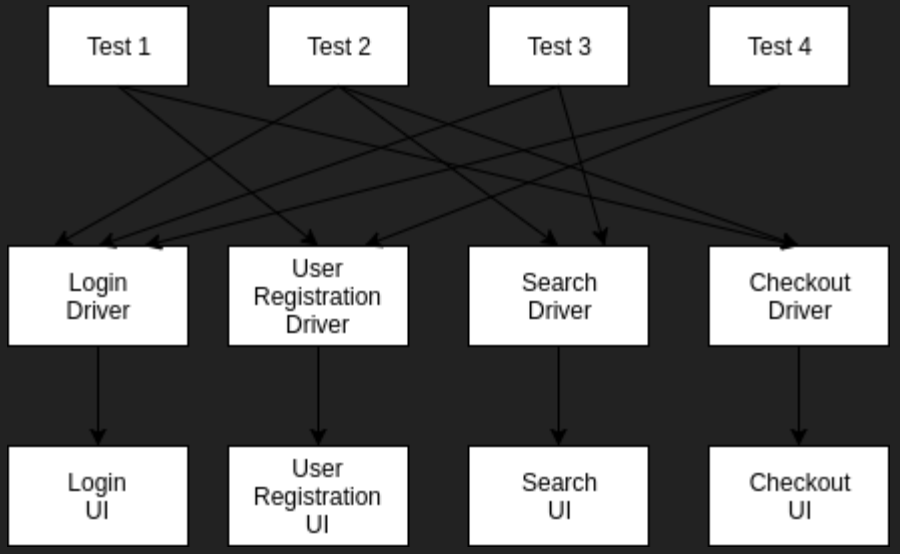
A Window Driver is an programmatic API for a UI window. A Window Driver ***should allow programs to control all dynamic aspects of a window****, invoking any action and retrieving any information that's available to a human user.*

-- Martin Fowler

DOESN'T CYPRESS/PLAYWRIGHT ALREADY DO THIS?

- Yes, but *imperatively*
- Window drivers let you build tests *declaratively*
  - Abstracting away impl. details

WHAT IS A WINDOW DRIVER?



A DRIVER?

```
export interface TodoDriver {
  addTodo(text: string): TodoDriver;
  toggleTodo(text: string): TodoDriver;
  deleteTodo(text: string): TodoDriver;

  assertTodoCount(count: number): TodoDriver;
  assertTodoCompleted(text: string): TodoDriver;
  assertTodoNotCompleted(text: string): TodoDriver;
  assertTodoExists(text: string): TodoDriver;
  assertTodoDoesNotExist(text: string): TodoDriver;
}
```

BENEFITS

- Makes you think about the API design of your components
- Excellent reusability
  - esp. if you use Cypress/Playwright component testing too
- Single touch-point for UI changes

<CodeDemo />