

Testes & TDD









Por que testar?

- Garantir que tudo vai continuar funcionando;
- Projeto pequeno vs Projeto grande;

Exemplo:

- E-mail de boas-vindas exibe o endereço do usuário;
- Após um tempo, o usuário agora pode ter mais de um endereço;







Tipos de testes

Testes unitários

• Testam uma função mínima e pura, que não realiza efeitos colaterais como acessos ao banco ou integrações externas;

Testes de integração

Os principais testes do back-end, testam funcionalidades completas como acesso à rotas até o retorno do controller;

Testes E2E

Testes de interface que simulam o acesso do usuário;









TDD

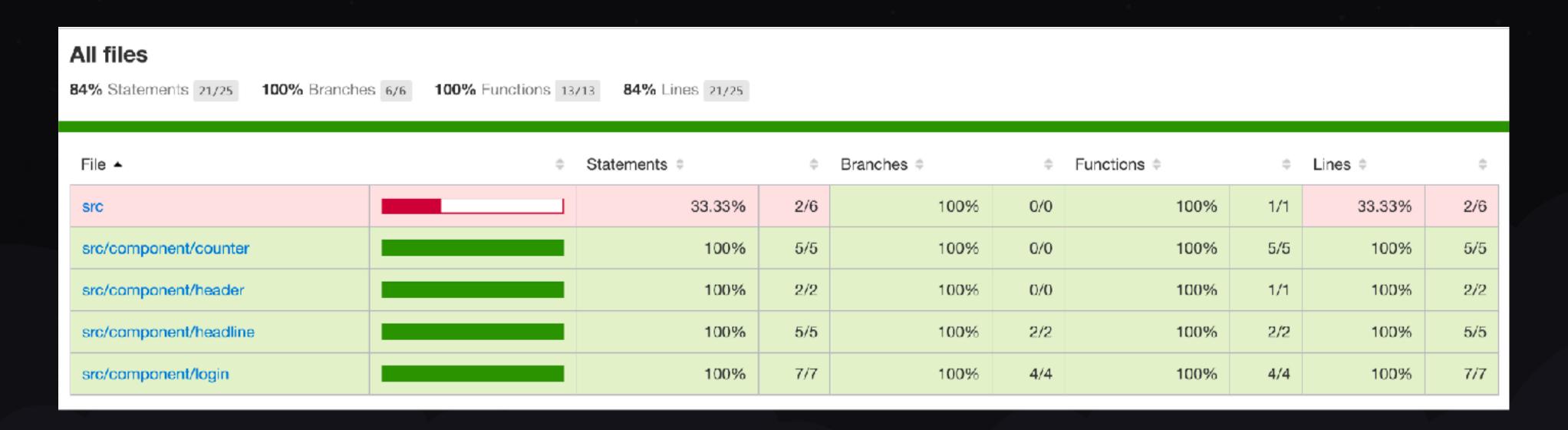
- Criar o teste antes da funcionalidade;
- Facilita visualização das regras de negócio;





Code Coverage

- O que testar? Testei o suficiente? Falta algo?
- Code Coverage coleta informações das linhas de código que seus testes atuais não alcançaram;











Jest

- Framework de testes do Facebook;
- Back-end / front-end / mobile;
- Tudo em um só pacote;
- Code coverage integrado;
- Multi-thread integrado;

```
# flavio @ Flavios-MacBook-Pro in ~/dev/jest [17:47:58]
  yarn test
yarn test v0.27.5
$ jest
 PASS ./math.test.js
  ✓ Adding 1 + 1 equals 2 (6ms)
  ✓ Multiplying 1 * 1 equals 1
  ✓ Subtracting 1 - 1 equals 0 (1ms)
✓ Dividing 1 / 1 equals 1 (1ms)
Test Suites: 1 passed, 1 total
              4 passed, 4 total
Tests:
Snapshots:
              0 total
Time:
              2.178s
Ran all test suites.
Done in 3.54s.
# flavio @ Flavios-MacBook-Pro in ~/dev/jest [17:51:25]
```





