Vitestの紹介

Press Space for next page →

自己紹介

- 📝 飯野陽平（wheatandcat）
- 🏢 フリーランスエンジニア（シェアフル株式会社CTO）
- 💻 Blog: https://www.wheatandcat.me/
- 🛠️ 今までに作ったもの
  - memoir
  - ペペロミア
  - Atomic Design Check List

Vitestとは

- VitestはVite環境で動作する高速なテストフレームワーク
- 領域的にはJestと同じ
- 特徴は実行速度の速さ。以下、参考記事
  - Vitest はどれくらい早いのか ~ Jest と比較 ~
  - 特にwatchモードが高速
- その他にも細かいチューニングがされている

Vitestを導入してみる①

導入は以下を参照。

- Getting Started | Guide | Vitest

Vitestを導入してみる②

コードは以下みたいな感じで書ける。

```
import { describe, it, expect } from 'vitest'
import { add } from "./calc"

describe('suite', () => {
  it('concurrent test 1', async () => {
      const r = add(1, 2);
      expect(r).toEqual(3)
   })
})
```

Vitestを導入してみる③

コンポーネントのテストなら以下みたいな感じ。

```
import { describe, expect, it } from "vitest";
import { render, screen } from "@testing-library/react";
import Demo from "./Demo";

describe("Demo", () => {
  it("タイトルに、「デモ」が表示さてている", () => {
    const props = {
      title: "デモ"
    };
    render(<Demo {...props} />);

    expect(screen.getByText(/デモ/i)).toBeTruthy();
  });
});
```

Vitestを導入してみる④

テストの実行以下のコマンドでOK 👌。

```
$ vitest
```

デフォルトで**watchモード**で起動。

```
$ vitest

 DEV  v0.27.1 ./demo

 ✓ src/lib/calc.ts (2)
 ✓ src/components/organisms/Demo.test.tsx (1)

 Test Files  2 passed (2)
      Tests  2 passed (2)
   Start at  23:12:20
   Duration  3.25s (transform 1.06s, setup 712ms, collect 725ms, tests 70ms)
```

カバレッジ/Mocking/Testing Type

以下のあたりはJestとほぼ同様に使用できる。

- Coverage
- Mocking
- Testing Types

Vitest UI

VitestにUIからテスト実行/確認行える機能がある。

- Vitest UI

以下のコマンドで実行。

```
$ vitest --ui
```

※デモで説明。

In-source testing

好みは別れるが、Rustみたいにコード上にテストコードを書くこともできる。

```ts
// the implementation
export function add( ...args: number[]) {
  return args.reduce((a, b) => a + b, 0)
}

// in-source test suites
if (import.meta.vitest) {
  const { it, expect } = import.meta.vitest
  it('add', () => {
    expect(add()).toBe(0)
    expect(add(1)).toBe(1)
    expect(add(1, 2, 3)).toBe(6)
  })
}
```

- In-source testing | Guide | Vitest

Debugging

テストコードでデバッガも使用できる。

- Debugging | Guide | Vitest

※デモで説明。

まとめ

- Jest互換なので、移行も比較的に容易に行えるのでおすすめ。
- 機能もかなり充実しているのでプロダクトで使っても、問題なさ気。
- 現状だとWebpack→Vite移行が進んでいるので、合わせてテストフレームワークも移行を検討すると良いかなと思います。

ご清聴ありがとうございました