

テストに用いるStorybook



Storybookとは

play functionを使う

Storyをテストファイルに読み込む

導入した目的とそのギャップ

意外な恩恵

今後の課題

今までのStorybookの使い方と比較して

まとめ

ご清聴ありがとうございました！

自己紹介



小張 泰志(こばり たいし) 

- ・21卒
- ・フロントエンド

最近熱くなったこと

- ・WBC優勝

Storybookとは

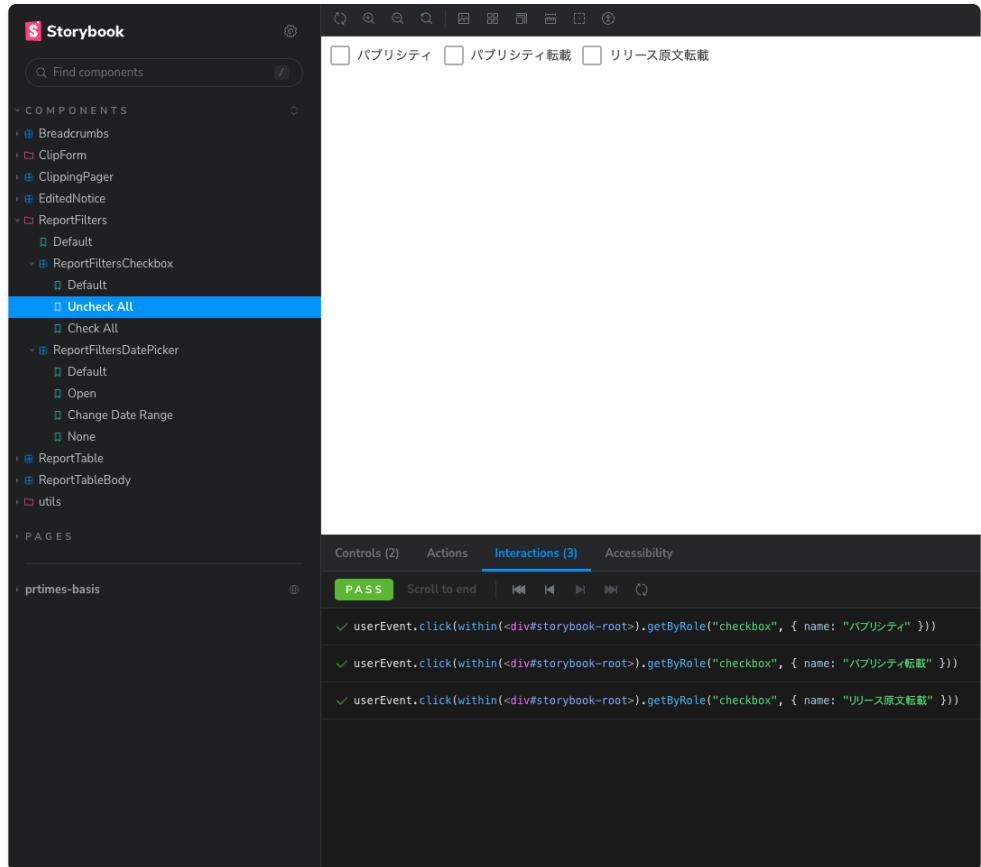
UIコンポーネントを可視化するツール

- ・デザイナー、PdMとの共有
- ・テストができる←今日話すこと

The screenshot shows the Storybook interface running at localhost:6006. On the left, there's a sidebar with navigation links like 'Introduction', 'Install and configure', and 'Changelog'. Below that is a 'LIBRARY' section with categories: 'Charts' (selected), 'Interstitial', 'Image', 'Tooltips', 'Badges', and 'Buttons'. Under 'Charts', sub-components like 'LineGraph', 'PieChart', 'SparkLine', 'Timeframe', and 'Histogram' are listed, with 'Histogram' being the active component. The main area displays a histogram titled 'Latency distribution' with blue bars representing data points from 20ms to 160ms. At the bottom, there's a table with columns 'Controls', 'Interactions', 'Design', 'Accessibility', and 'Actions'. The 'Controls' column shows configuration for ' dataType' (set to 'latency'), ' showHistogramLabels' (set to 'True'), ' histogramAccentColor' (set to '#1EA7FD'), and ' label' (with a hand cursor icon over it). The 'Actions' column contains a button labeled 'Latency distribution'.

play functionを使う

```
// ReportFiltersCheckbox.stories.tsx
export const UncheckAll = {
  name: '全てのチェックを外せること',
  play: async ({ canvasElement }) => {
    const canvas = within(canvasElement);
    // 全てのチェックを外す
    await userEvent.click(
      canvas.getByRole('checkbox', { name: 'パブリシティ' }),
    );
    await userEvent.click(
      canvas.getByRole('checkbox', { name: 'パブリシティ転載' }),
    );
    await userEvent.click(
      canvas.getByRole('checkbox', { name: 'リリース原文転載' }),
    );
  },
} satisfies Story;
```



Storyをテストファイルに読み込む

assertは  vitestで行う

- Storybookをbuildせずに実行できるので速い
- mockが使える

```
// ReportFiltersCheckbox.test.tsx
import { composeStories } from '@storybook/react';
import { render } from '@testing-library/react';
import * as stories from './ReportFiltersCheckbox.stories';

const { UncheckAll } = composeStories(stories);

it('全てのチェックを外せること', async () => {
  const { container, getByRole } = render(<UncheckAll />);
  await UncheckAll.play({ canvasElement: container });
  expect(getByRole('checkbox', { name: 'パブリシティ' })).not.toBeChecked();
  expect(
    getByRole('checkbox', { name: 'パブリシティ転載' }),
  ).not.toBeChecked();
  expect(
    getByRole('checkbox', { name: 'リリース原文転載' }),
  ).not.toBeChecked();
});
```

導入した目的とそのギャップ

1. play functionによるテストの可視化

達成できた

2. Figmaプラグインの導入

実装とFigmaのズレを減らしたかった

Chromaticを導入しないといけなかつたので断念

3. VRT

今後進めていきたい

意外な恩恵

- ・ユニットテストのデバッグが楽になった
- ・AAA(**Arrange, Act, Assert**)を意識するようになった

Assertを独立して書くので、テストの意図が明確になった。

Assert後にActするようなパターンも書きづらくなった。

今後の課題

データ取得するコンポーネントのStorybook化

mswなどを使ってmockできる部分とそうでない部分がある

どこまでを Storybook に書くのか

- ・境界があいまい
- ・すべてを Storybook に寄せることは不可能
- ・mockがStorybookに導入されたら？

We have a long list of quality of life improvements here that we'll be rolling out in 7.x, especially around better mocking, full page testing, and compatibility. [\[1\]](#)

今までのStorybookの使い方と比較して

今まで：実装者以外とのコミュニケーションとしてのツール

ただ、、、、

デザイナーは figma を見るし、QA は staging を見る

エンジニアもそんなに Storybook を見ない

テストツールとして活用後

エンジニアがStorybookを見る機会が増えた

- ・バックエンドができるまでの開発
- ・テストの説明

まとめ

Storybookをテストに活用して、テストが可視化された

まだ道半ば

- ・ VRTの導入
- ・ Storybookのアップデートにも注視していきたい

ご清聴ありがとうございました✨