僕たちはそろそろ カスタムブロック開発の後方互換が 辛い問題に立ち向かわなければ いけない。



自己紹介



- 大橋直記 / Naoki Ohashi
- Engineer / VK Blocks VK Blocks Pro
- 本州最南端在住

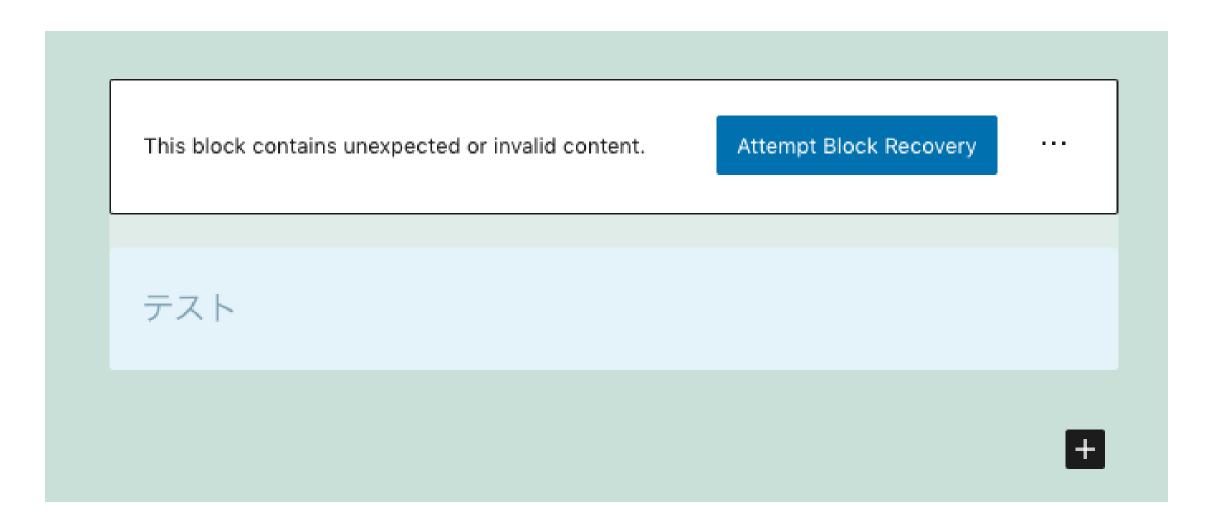
Twitter:@naogify



今日話すこと

VK Blocks・VK Blocks Proに後方互換テストを導入(中)したその経緯と実装方法について。

ブロックが壊れた経験ないですか?



ブロックは壊れやすい

ブロックのDOM構造を変更すると壊れる

旧

```
save({ attributes }) {
  return (<div>Alert Block</div>);
}
```

新

```
save({ attributes }) {
  return (Alert Block);
}
```

後方互換を保つには

```
// 互換性を保つコードを追加する必要がある。
registerBlockType('example/alert', {
 deprecated:[
         attributes: {
         save({ attributes }) {
           return (<div>Alert Block</div>);
```

課題

- ブロックが壊れた報告がきた場合、コミット履歴から、 過去バージョンごとにテストしていく必要がある。
- チーム開発時に 「後方互換処理が動作」の確認コスト が バカにならない。

辛すぎる…。

確認コストに時間が取られて、新規開発ができない。

Gutenbergは、 どうしているのか…。?

コアの後方互換テスト調べて 実装してみた。

テストのサンプルコードをGitHubで公開。

https://github.com/vektor-inc/block-test-kit



テストファイルの構成

```
root/
 src/
   L lindex.js、edit.js等 (ブロック本体)
 test/
   L e2e-tests/fixtures/blocks/
      block-test-kit alert.html (ブロックデータ)
      block-test-kit__alert.parsed.json (ブロックデータパース後)
      block-test-kit__alert.json (ブロックデータJSON)
      block-test-kit__alert serialized htm (ブロックデータシリアライズ化)
 integration/full-content/full-content.test.js(後方互換テストファイル)
 utils js(テストに必要な関数)
 page json 等環境設定ファイル
```

テストの概要

各バージョンのブロックのデータを、 投稿が保存されるフローをコードで再現してブロックが 壊れないか確認。

Version 1.0

```
<!-- wp:block-test-kit/alert -->Alert Block<!-- /wp:block-test-kit/alert -->
```

Version 0.9

```
<!-- wp:block-test-kit/alert --><div>Alert Block</div><!-- /wp:block-test-kit/alert -->
```

デモ



テスト内容

```
import { parse as grammarParse } from '@wordpress/block-serialization-default-parser';

// block-test-kit__alert.html を パースした結果と、
const parserOutputActual = grammarParse( htmlFixtureContent );

// block-test-kit__alert.parsed.jsonとを、
const parserOutputExpected = JSON.parse( parserOutputExpectedString );

//比較してテスト
expect( parserOutputActual ).toEqual( parserOutputExpected );
```

詳細なテストの手順はこちらのブログを参考に

https://www.vektor-inc.co.jp/post/deprecated-test-for-custom-block/



メリット

- レビューの効率化(確認コストが段違いに減る!)
- フォーラムへのバグ報告数減(予定)
- ・新規開発の時間の確保

ありがとうございました

