

Retoolとzxでコード健全性の可視化のダッシュボードを作成する



Press Space for next page →





自己紹介

- 📝 飯野陽平 ([wheatandcat](https://www.wheatandcat.me/))
- 🏢 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](#)

インスパイア元の記事紹介

- [zx + Datadog + GitHub Actions でフロントエンドのコードベースの健全性を可視化する](#)
- こちらの記事で紹介していた内容がインスパイア元の記事
- Datadogは、有料サービスなので個人開発では、ちょっと・・・という人向けに、別の方法を紹介

zxとは？

- リポジトリ: [zx](#)
- Google製のJavaScript文法でShellScriptが実行できる
- mjsの拡張子で、そのまま実行もできるし、JavaScript内にimportしてコード内で使用することも可能

.mjsで使用する場合は、以下みたいな感じで使用可能。

```
#!/usr/bin/env zx

await `$cat package.json | grep name`

let branch = await `$git branch --show-current`
await `$dep deploy --branch=${branch}`

await Promise.all([
  `$sleep 1; echo 1`,
  `$sleep 2; echo 2`,
  `$sleep 3; echo 3`,
])

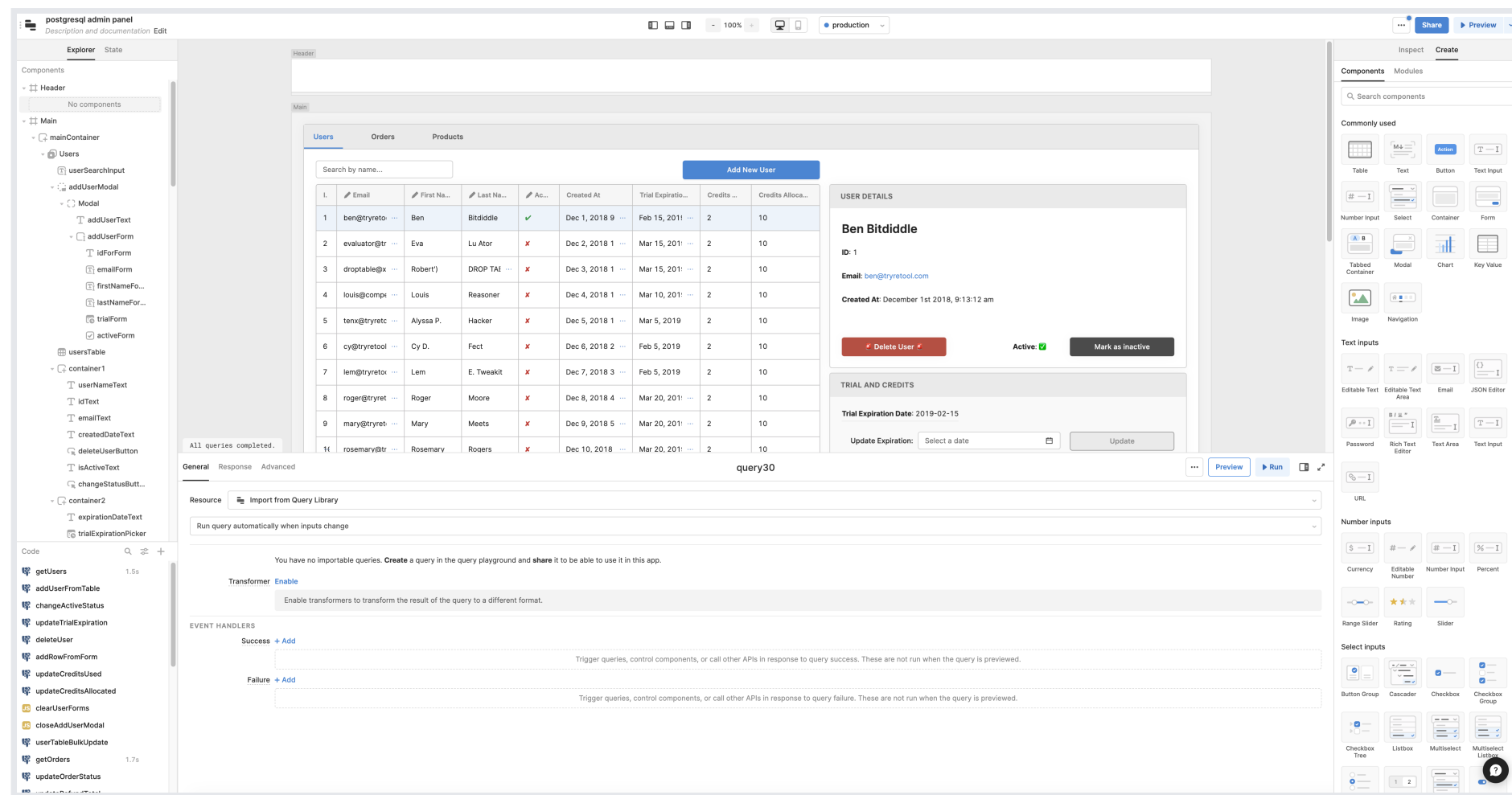
let name = 'foo bar'
await `$mkdir /tmp/${name}`
```

Retoolとは？①

- サービス: [Retool](#)
- ローコード開発ツールで、あらゆるデータベースを元にグラフ、チャート、検索フォーム、データ入力などのUIを作成できるサービス
- 個人の場合では無料で使用できる
 - [Pricing](#)
- 一部jsもサポートしているので、柔軟性も高い

Retoolとは？②

- こんな感じで使用できる（Demo）



今回の実装の概要

- 前提
 - 無料で利用できる
- 対象
 - フロントエンドのコードで実装
- 使用ツール & サービス
 - [TypeScript](#)
 - [zx](#)
 - [Firestore](#)
 - [GitHub Actions](#)
 - [Retool](#)

実装してみた①

- PR
 - ①. <https://github.com/wheatandcat/memoir/pull/239>
 - ②. <https://github.com/wheatandcat/memoir/pull/241>
- 実装の紹介 (Demo)
 - 基本は以下の記事の通りに実装
 - https://zenn.dev/ryo_kawamata/articles/create-frontend-dashboard
 - **zx**の使い方を説明
 - [Jest](#)からカバレッジを取得する方法
 - [対象コード](#)
 - Jestのカバレッジレポート確認
 - [Jest](#)からテストの実行時間とテスト数を取得する方法
 - [対象コード](#)
 - テスト結果をJSON形式で取得するオプションを使用

実装してみた②

- Firestoreへのデータ保存の方法
 - [対象コード](#)
- Github Actionsで定期実行
 - [対象コード](#)
- あとは、保存しているデータを元にRetoolでダッシュボードを作成
 - [ダッシュボード](#)
- Retoolでのダッシュボード作成のDemo

まとめ

- 複雑なShellScriptを書くより、素直にzxを使ったほうがスッキリしたコードが書ける
- Retoolは、汎用的に使えるローコード開発ツールなので、積極的に使っていきたい
- Retoolはサポートが、かなり充実している散らばっている情報をまとめるのにも、役立ちそう
 - <https://retool.com/integrations>

 **ご清聴ありがとうございました** 