Fuzz Testing

古殿直也 2025-08-25

背景

timeパッケージ、素晴らしいのですが時刻ではなくて日付を扱いたいときに面倒

- タイムゾーンの扱いに気を遣う
- 月末が欲しい、とか期間の日数が欲しい、とかは実装されていない
- シュッと書くことはできるし難しくない。でも気を遣う

そういう処理をいい感じにパッケージにまとめた。**テストをどうするかが今回の話題**

品質保証を楽に行いたい

- 品質保証のためのテストをしたい
  - TDDとかリグレッション回避のためのテストではなく、今書いているプログラムが本当に意図通り全てのケースで動作するかを担保したい
- とはいえテストケースをたくさん書くのは嫌
  - メンテナンスがめんどいし、自分の思いつく範囲には限界がある。体力勝負になって美徳に反する
- **プロパティベーステストが良さそう**

プロパティベーステスト

- 「プログラムの性質を関数として記述して、プログラムへの入力をランダムに生成してその性質をテストする」
  - Table drive test のテーブルの値を自動で生成するイメージ
- 引数から出力を得ることを目的とするプログラムと相性がいい（つまり関数型なプログラムと相性がいい）
  - 副作用のあるプログラムに比べて、プロパティを記述するのが簡単なので
- 例: 「日付データaのn日後の日付bを生成したとき、それらの間の日数は常にn+1である」
- 関数型言語の界隈でよく聞く
  - Elixir の書籍 https://www.lambdanote.com/products/proper
  - Haskell の論文 https://users.cs.northwestern.edu/~robby/courses/395-495-2009-fall/quick.pdf

Fuzz testing

- プロパティベーステストのための機能がGo1.18で追加された（Fuzzing）
- testingパッケージの一つの機能で、FuzzXXX という名前のテスト関数を定義して、その引数を *testing.F 型にする
- go test -fuzz=FuzzXXX とすると、定義したFuzzXXX関数が実行される
- Go Fuzzing
- pkg.go.dev/testing

Fuzz testingnの例

デモタイム

- テストの書き方
- それぞれのコンポーネントの説明
- 実行例
  - 失敗の例
  - いつまで経っても終わらない例

まとめ

- プロパティベーステストは関数型なプログラムのテスト網羅性を簡単に挙げられる
- Fuzz testing がGoの標準にあってプロパティベーステストできる

参考資料

- 実践プロパティベーステスト
- QuickCheck
- Go Fuzzing