null

ヤバイのでなんとかする

takeshik

事故紹介

- 廃ついったったー
  - https://twitter.com/takeshik
- ジロリアン
  - ホーム: 仙川
  - 三田本店にはまだ行けてません
- Twitter クライアント? MetaTweet 制作中
  - http://www.metatweet.org/
  - 未だに未完成　永遠に開発中
- おまけで Tween に手を出してみたり

流れとか

- 対象環境: .NET　使用言語: C# 3.0

- null の何たるかについて軽く解説
  ↓
- null のいやらしさについて嘆く
  ↓
- null に立ち向かう手段をいくつか取り上げる
  ↓
- **時間が余ったら二郎の素晴らしさを熱く語る**

で、null ってなんぞ?

- どのオブジェクトも参照していないことを示す特別な値
- (Visual Basic では Nothing)
- メンバにアクセスしようとすると例外 NullReferenceException を投げる
- 値型 (struct, enum) には代入できない

- 小難しい話は全部割愛 (←重要)

null の周辺要素

- Nullable<T> 型 (T?)
  - 値型でも null を使えるようにする
  - null をそのまま設定可能。取得はちょっと面倒
- null 結合演算子 (??)
  - `a ?? b　＝　a != null ? a : b`
- default キーワード
  - 参照型の場合 null、値型の場合デフォルト値
    - Generics 向け機能
    - `default(int?) == null`

null の恐怖

- ちょっとのことで例外が飛ぶ
  - プロパティとかにアクセスするだけでアウト
- チェックとか面倒
  - `if (a == null)` とか苦痛
- 面倒なのに結構 `null` を返すメソッドが多い
- 「無」を表すのにもっと適切な値が存在する場合がある (ex. `Stream.Null`)
- …できれば見なかったことにしたい！

null への対処方法 (1)

- 真面目に条件分岐
  - `if (a == null) ...;`
  - 単体だとそうでもないがちょっと面倒
- null 結合演算子の利用
  - `a = a ?? new T(...);`
  - 初期化みたいな用途とかだと使えるかも
  - 使い所が広く存在するか微妙

null への対処方法 (2)

- ちょっとしたメソッドを作る
  - ```
    TResult GetOrNull<TSource, TResult>(
        TSource obj,
        Func<TSource, TResult> func)
    {
        return obj != default(TSource)
            ? default(TResult)
            : func(obj);
    }
    ```
- `FileInfo file = ...;`
- `GetOrNull(file, f => f.FullName);`
  - `file` が非 `null` ならファイルのフルパスが取得できる
  - `file` が `null` ならそのまま `null` が返る → **チェック省けて幸せ！**
    ```

null への対処方法 (3)

- 静的メソッドは何かと面倒
  - だったら拡張メソッドにすればいいじゃないか
- 拡張メソッドはただの構文糖どころじゃない
  - 内部的には静的メソッドなので this 値が null になり得る！ → null でも呼べるメソッド
- さっきのメソッドも
  -
```
TResult GetOrNull<TSource, TResult>(
    this TSource obj,
    Func<TSource, TResult> func)
```
  -
```
file.GetOrNull(f => f.FullName);
```
- **行儀が悪いのでご利用は計画的に**

null への対処方法 (4)

- Nullable<T> があるなら逆があってもいいよね！
  -
```
public struct NonNull<T> where T : class {
    // operator ==, operator != は省略
    private readonly T _value;
    public NonNull(T obj) : this() {
        if (obj == null)
            throw new ArgumentNullException("obj");
        this._value = obj;
    }
    public static implicit operator NonNull<T>(T obj) {
        return new NonNull<T>(obj);
    }
    public static implicit operator T(NonNull<T> obj) {
        return obj._value;
    }
}
```

null への対処方法 (5)

- 実際に使ってみる
  - ```
    NonNull<string> str = "test";
    str = null; // ArgumentNullException
    ```
  - 代入時に null を検知可能
- Stream.Null のような「無」を表すオブジェクトをもっと積極的に使うようにしてみる
  - ※ String.Empty のように null と比較するのが相応しくないような事例もあるので時と場合によるかも
  - とりあえず作ってみる
    ```

null への対処方法 (6)

```
public struct Default<T> where T : class {
    // NonNull<T> と同じメンバ、operator ==, operator != は省略
    public Default(T obj) : this() {
        if (obj == null) this._value = GetDefault()
            else this._value = obj;
        if (this._value == null)
            throw new ArgumentNullException("obj");
    }
    private static T GetDefault() {
        var member = typeof (T).GetMembers(BindingFlags.Public |
                                           BindingFlags.Static)
            .Where(m => m.Name == "Default" || m.Name == "Empty"
                                            || m.Name == "Null")
            .SingleOrDefault();
        if (member is FieldInfo)
            return (T) (((FieldInfo) member).GetValue(null));
        if (member is PropertyInfo)
            return (T) (((PropertyInfo) member).GetValue(null, null));
        return null;
    }
}
```

null への対処方法 (7)

- 使ってみる
  - ```
    Default<string> s = null;
    Console.WriteLine(s == "");
    Default<StreamReader> r = null; // true
    Console.WriteLine((r == StreamReader.Null)); // true
    ```
- null 値の代入がうまく回避されている。
- 色々作ってみましたが結構便利…かもしれない？
- 行儀が本当に良くないのでご利用は計画的に
    ```

まとめ

- いろいろ作ってみるのは結構楽しいです
- null をうまくやり込めて楽しいプログラミング
- 行儀の悪いコードはほどほどにしましょう
- 実はきながら型構造考えてたけど
  意外と使えるかも…？
- 二郎はラーメンではありません