ひよこProject コードレビュー

注意

- ひよこは、C#, XNA という開発環境を用いてゲームを作っています。

- 多くのチームはC++, FK などでつくっていると思われます。この違いから、コードの書き方などもかなり変わってきてしまうので、話すことの全てがみなさんの環境で当てはまるわけではありません。

講義の進め方

- いままでの制作の流れ
- 問題点
- ちょっと後悔している事
- よかった点
- チーム制作で起こる問題を解決するためのアドバイス

いままでの制作の流れ

制作の流れ　前半

- 2年次前期~2年次夏休み
  - フレームワークの作成

- ～3年次冬休み
  - ステージエディタ作成
  - アイディアをプロトタイプに反映させ、どんな感じか確認
  - 必要なエフェクトなどを実験

制作の流れ　後半

- ～現在
  - 各ステージを作成
    - 仮素材を置き、ゲームとして成り立つかどうかをまず確認
    - 仕様に問題があることが分かったら、それを改善した仕様を考え、ステージを更新

問題点について

問題点1
命名規則

- 制作初期段階で、C#流の書き方や命名規則を知らず、Java等の書き方をそのまま使っていた。

- 勉強していく上で、新たに知った命名規則もあった。

- 現在の状態から正しく書き直すのは困難。

C#独特の命名規則

- カプセル化に、get…()や、set…()というアクセサメソッドをつくってはいけない。代わりに、プロパティというものを使う。

- パブリックメンバの始まりは、基本的に大文字を使う。

プロパティの例

```
private int age = 0;

public int Age{
        get {
                return age;
        }
        set {
                age = value;
        }
}
```

しかし
命名規則に凝りすぎると進まなくなる

- チーム内で、「こっちの方が見やすい」という共通の認識があれば、命名規則は破っていい…と個人的には思います。

- 命名規則に限らず、一部に凝りすぎて全体が進まなくなるのは問題。

問題点2
テストが不十分なフレームワーク

- ゲームの基盤となるフレームワーク部分のテストが不十分

- 実際にフレームワークの上にゲームを作り始めてから気が付いたバグが多かった。
  - 解決しづらい、根が深いバグに

問題点3
コメントが少ない

- 全体的にコメントが少ない
- 昔書いたコードに何か書き加える際、思い出す時間が無駄に多く取られてしまう
- 今日自分で書いたコードは、3日後には他人が書いたコード同然

すこし後悔している点

既存のライブラリを使うことを避けていた

- 制作の初期段階では、既存のライブラリやフレームワークを使うことに抵抗があった。（全部自分でやってやる！というダメな意地）

- 既存のものをもっと上手く使えば、作業時間を短縮できた（かもしれない）。

よかった点

よかった部分1
ステージエディタ

- 早い段階で、ステージエディタをつくることができた。

- ステージの作成が容易になった。
  - 大きなステージ・複雑なステージ作成に力を発揮
  - プロトタイピングの助けに

書かなければいけないコードの量を減らそう

- コードを書く努力だけでなく、できる限り書かなければいけないコードの量を減らすという努力も必要

- ステージエディタのようなGUIツールだけではなく、描画や接触判定・音楽再生を簡単にするようなプログラムを早めにつくれるといい。（特にFK以外の環境で作る場合）

よかった部分2
FKの座標変換の命令の移植

- FK座標変換の命令で便利だったものを、XNAでも使えるようにした。

- 以前FK上で作った動きを、そのままの方法でXNAで再現することができるようになった。

- 全体として書かなければいけないコードの量もかなり減らせた。

よかった部分3
フレームワークを先に開発した

- 2年次の夏休みにゲームの基盤部分を使えるようにしたことで、その後の作業が円滑に進むようになった。

- 仕様が決まる前だからといって何もしないのではなく、何かできることを探して動くことが大切。

チーム制作で起こる問題を解決するためのアドバイス

確定している仕様が不十分

- つくるべきものが曖昧なまま、何かをつくらなければいけないということが起こる。
- プログラマはゲームを実際に組み立てている立場のため、企画者よりもこの仕様の曖昧な部分に気が付くことができる。
- 少人数のチームで、トップダウンな仕様の確定に拘りすぎるのは無謀。（必ずしも全ての仕様を企画者が決められるわけではない）
  - 曖昧な仕様に気が付いたら早めに報告
  - 曖昧な部分を自分で補完して、動くものを見せることも大事

必要な素材と、作ってもらっている素材の間に齟齬ができる

- 「どのように動かすので、どのような素材が、どのような形式で必要である」という情報を、プログラマ・プロデューサ・グラフィッカ間で把握しておくようにしましょう。

- これをしないと、リテイクが頻発して険悪なムードになります。

なかなか素材が上がってこない

- プログラマが、しっかりと「いつまでにこの素材が必要である。」と言うようにしましょう。

- プログラマの言葉は、場合によってはとても説得力があり、効果的です。

- おわりです
- ご清聴ありがとうございました。