インタラクティブゲーム制作
ゲームプログラミング講座
第１回資料

竹内 亮太

(2009/5/15)

1　ゲームプログラミング事始め

1.1　開発環境の選択

コンピュータゲームを作ろうと思い立った時、一般人にとって一番現実的なプラットフォームは間違いなく PC(Windows) しかありません。ゲーム機の開発環境は、ハードメーカーに多額のお金を支払わないと触ることすらできないので、プロ以外には縁がないのが普通です。最近は Microsoft 社が XNA という開発環境を公開しており、条件次第では XBox360 上で自分の作ったゲームを動かすことができるようになっていますが、これはこれまでの情勢からすれば非常に画期的なことです。プロジェクト演習のチームによっては、これを採用しているところもありますね。

この XNA の環境も魅力的ですが、実は PC や XBox360 の性能を完全に引き出せるわけではありません。XNA は C#という「Microsoft 版 Java」とでも言うべき言語を使います。これは開発効率は非常に素晴らしいのですが、処理速度効率などにおいてはプロが一線で使用している言語に比べると数段劣ります。ゲームプログラムは処理速度が命ですから、元来がビジネスアプリユースの C#では荷が重すぎるということです。

ではプロが今でも現場で使用している言語とは何か、ということになります。それが今から皆さんに伝授する「C++」という言語です。C++はオブジェクト指向の思想を盛り込みつつも、処理速度効率を維持している上級者向けの言語です。Java や C#もオブジェクト指向の言語ですが、それらよりも先輩にあたります。車で例えるなら、Java などがオートマチック車で、C++はバリバリのマニュアル車と考えるとよいでしょう。エンストの危険性はありますが、扱いに慣れればマシン性能を限界まで引き出すことのできる言語です。このような特性を持つ言語がなかなか出てきていないため、当分 C++の必要性はなくならないと思われます。

1.2　グラフィクス API の選択

では C++とやらを使えばすぐにでもゲーム作れるの !? とせっつかれそうですが、残念ながらそういうわけにはいきません。C++はあくまで言語でしかないので、それ自体には 3DCG を表示したり、音を鳴らしたりする機能はありません。そういう機能は「誰かがまとめてくれた命令の集まり」を用意し、C++を通じて呼び出すことで利用します。そのような命令の集まりのことを「API」と呼びます。特にゲームプログラミングにおいて、一番いじくることになるのが見た目の表示部分です。これを司る API をグラフィクス API といいます。現在の情勢では、このグラフィクス API には大きく分けて 2 種類が存在します。Microsoft が提供する「DirectX」と、複数社によってまとめられている「OpenGL」です。これらの違いを語り出すとそれだけで 1 コマ使ってしまいそうなので、ざっくり特徴を述べます。

1.2.1　DirectX

- Windows 環境でゲーム作る際のスタンダードな API
- XNA や、当然 XBox も DirectX ベース
- サンプルや解説書が豊富
- でも複雑で難解、「サンプルの改造」レベルから抜け出せない可能性が高い
- バージョンアップが頻繁にあり、それまでの苦労が水泡に帰す恐れがある

1.2.2　OpenGL

- Windows 以外でも基本的には同じように動くので、作ったものを活用しやすい
- PS3 の開発環境に採用されている他、Wii や GC などでも OpenGL がベースになっている
- サンプルや解説書もそれなり、研究分野での利用実績が強い
- 本当に基本的な命令しかないので、ゲームで使うような実践的な処理は自力で用意する必要がある
- バージョンアップがあっても過去の資産が無駄にならない

Web などで調べると DirectX の方が目に付くと思いますが、私個人が両方使った経験も踏まえると OpenGL の方がおすすめできます。開発に手慣れてくればどちらを使っても同じようなものは作れるようになりますので、DirectX を学んでおかないと将来役に立たないんじゃ…という心配は無用です。それより何より「サンプルの改造から抜け出せない」危険性を今は回避します。

1.3 ライブラリの重要性

さて、OpenGL ベースで進めていこうと思うわけですが、「実践的な処理は自力で用意」というのが気に掛かりますね。ここがネックになって DirectX に走る人も多いのですが、この授業では「FK」というライブラリを利用することで、この問題をフォローします。ライブラリとは、プログラミングをする際に便利な部品の集まりのことです。API に似ていますが、この授業では「API を直接使うよりも機能的な構造を実現し、利用しやすくしたもの」をライブラリと呼んでいます。FK を利用することで、DirectX や OpenGL を直接操作する手間を大幅に軽減して、本来の目的であるゲームの画面構成や動作のプログラミングに注力することができます。

このライブラリという考え方は、利用するだけに留まるものではありません。皆さん自身が書いていくプログラムで有用な機能が生まれたら、それをどんどんライブラリにして再利用していくべきです。ゲームプログラミングはゼロから全て自力でやらないと意味がない、と考える人は多いです (私もそうでした) が、今のご時世ではそれは徒労に終わる場合が多いです。既に用意されているものを如何に理解し、利用できるかが今のプログラマーには求められます。効果的にライブラリを組み合わせ、あるいは自作することで目的を達成していくのです。

1.4 セットアップ

では前置きが長くなってしまいましたが、皆さんの PC に開発環境を整えることにします。ゲーム開発を行っていく上では、複数のプログラムをまとめて管理していく必要がありますので、Java で言うところの Eclipse のようなソフトが必要です。Windows で C++ プログラミングをする際には「Visual Studio」というソフトを使うのが一般的です。無料版が Web から入手できますので、そちらを利用することにしましょう。FK システムも Web から入手できます。授業資料ページから辿って入手してください。セットアップの具体的な手順は配布元のドキュメントを参照してください。

1.5 ゲームプログラムの基本構造

セットアップが済んだら早速「プロジェクト」を作ってみましょう。今のバージョンですとクリックするだけ簡単に作れますし、最初からサンプルコードが付いてきます。お得ですね。

さて、そのお得なサンプルコードを見て欲しいのですが、今まで習ってきたプログラム (特に Java) とは違う構造に見えるんじゃないかと思います。これは言語が C++であるということと、このプログラムが「リアルタイムなアプリケーション」特有の構造になっているためです。ゲームは通常、操作をしなくても画面は動き続けているのが当たり前になっています。プレイヤーの操作を受け取ることによって、その画面の様子が変化していく、という構造になります。今まで皆さんが見てきて書いてきたプログラムは「上から下まで駆け抜けて終わり (途中で分岐や繰り返しはあったでしょうが)」という構造がほとんどだったでしょう。これではゲームプログラムとしては厳しいですよね。

ではサンプルコードをもう一度よく見てください。C++では main() という関数 (Java でのメソッドとほぼ同じ意味合いです) からプログラムはスタートするのですが、この関数の中身はざっくり分けると以下のようになっています。

- 色々変数を準備している
- その変数に対して色々設定している
- →繰り返し開始 (無限ループ)
- 　ウィンドウが閉じられたらプログラム終了 (繰り返し脱出)
- 　画面を書き換える
- 　物を動かしたり変化させる処理を「1 フレーム分」行う
- ←繰り返し終了、開始ポイントへ戻る

このように、上から下へ処理が流れる原則は同じですが、ぐるぐるぐるぐる繰り返しが回りっぱなしの状態になるのが基本です。その繰り返しの中で「1 フレー

ム (コマ) 分の処理を書く」のがゲームプログラミングの難しいところです。物をある地点からある地点まで動く様子を表すには、それを少しずつ実行してやらないといけないわけです。

そして、このプログラムの中で使われている変数は、int や double のようなチャチなものではありません。「超変数」とでもいうべき、高性能な変数を駆使して書かれています。これが「クラスオブジェクト」であり、それを中心に進めていくプログラミングのことを「オブジェクト指向プログラミング」と呼びます。同名の演習が授業にありますが、本質的には同じ物です。ただ、Java と C++という言語の違いがあるので、細かい作法などは異なります。混同しないように気を付けてください。

1.6 宿題

今回の段階は、まずゲームプログラムの構造に慣れてもらうのが第一です。FK のユーザーズマニュアルの第 1 章を熟読してください。プログラムの構造は第 1 章を読めばだいたい理解できるはずです。

そして次の段階では「キー操作で表示物を動かす」という動作を目標にしてみましょう。キー操作に関連するのは fk_Window の変数、表示物を表しているのは fk_Model の変数のどれかです。プログラムを読み解けば、どの変数が何を指しているかは分かるはずです。

fk_Window については第 10 章、fk_Model については第 8 章に解説がありますから、そこで色んな命令を見つけ出し、試してみてください。