目次

第 1 章
はじめに

この本を手に取っていただき、ありがとうございます。この本は Microsoft Windows の環境で動作する DirectX 9.0 （正確には DirectX 9.0Ex） を紹介するものです。

1.1 なぜ今、DirectX 9.0Ex なのか

今さら DirectX 9.0 かという声が聞こえてきそうです。現在においては、最新のグラフィックス API だと DirectX 12 (D3D12) や Vulkan といったものがあります。また歴史の長い OpenGL の API ですら進化を続けて、バージョン 4.6 まで到達しました。敢えてこれらの API を解説せず、 DirectX 9.0Ex を選んだかというと、DirectX 9.0 はバランスのとれた扱いやすい 3D の API だと私は考えています。特に、ゲームエンジンに頼らずに 3D のプログラミングをやってみたい、という人に適しています。

しかしながら、古い API であるが故に問題が多くあります。この問題というのは、実行できる環境が少ないとか不具合とかそういったものではありません。DirectX 9.0c に関してのプログラミングの情報は多いものの、現在においてはコンパイルできる状態で説明されているものは少ないかと思います。こういった状況のため、現在の環境に合わせた DirectX 9.0c プログラミングの書籍を作成することにしました。今 DirectX 9.0c を Windows 10 の環境で触るという点で、 DirectX 9.0Ex を対象として説明を行っていきます。また、よく問題に上がる作成したプログラムの配布に関しての話題も取り上げてみました。

特に以下の点で悩んだことがある人は、この本の内容に納得してもらえると思っています。

- 環境構築においての DirectX SDK の問題
  - 近年の Windows.h との相性
- D3DX が使えないことへの対処
  - 算術系
  - シェーダーのコンパイル
  - テクスチャの利用
- アプリの配布に関しての問題
  - d3dx_**.dll への依存

対象読者

この本の読者として、

- DirectX 9.0 以降を触ったことはある人
- DirectX 9 世代の開発環境構築に苦戦している人
- DirectX 11 を触ったけど、DirectX 9 世代が手軽だったと思っている人
- DirectX 9.0 でもまだまだやれると思っている人
- DirectX 9.0 Ex 初心者の人

を想定しています。特に、現在の最新環境で DirectX 9.0 プログラミング環境を構築する点で悩んでいる人に最適です。DirectX 9.0 の情報がインターネット上には数多くありますが、環境のセットアップに関しては現在は通用しないものがほとんどです。この本では、そういった部分を補うものとなっています。

動作環境も Windows 10 を前提として、 DirectX 9.0 Ex を使用しています。DirectX 9.0 Ex については情報も少ないので、使い方についてはこの本が役立つものと信じています。

1.2 3D 数学について

3D を扱うにあたり、どうしても数学が必要になります。といっても初めは、行列というものを使用するといったこと、行列の演算によりどのような動きになるのか、といった点から感触を確かめていくのがよいかと思います。ただし、行列の演算については新旧変わりないため、この本では説明を除外しています[*1]。この本では行列用の算術ライブラリを使うという点について説明を行っています。

1.3 動作環境について

本書が想定している開発環境については表 1.1 のようになっています。いくつかのオープンソースを使用するため Git が必要となっています。

また実行環境については、実機を想定しています。そして、シェーダーモデル 3.0 というものが使用できる環境に限定しています。ただ現在のハードウェアでは、 DirectX11 対応していないものが見当たらないレベルなので気にしなくても大丈夫だと思っています。一方で、リモートデスクトップ経由や、仮想環境の中での動作については本書の対象外となっております。

[*1] ページ数も限られていますし・・・。

第 2 章
DirectX 9.0Ex の世界へ

それでは早速、 DirectX 9.0Ex の世界へ入っていきましょう。ここでは DirectX を初期化して、画面を塗りつぶすアプリケーションの作成を目指していきます。

2.1 DirectX 9.0Ex のメリット

学習によいバランスであると冒頭で説明しましたが、他のメリットについて説明していませんでした。実際のプログラミングに入っていく前に、よい点を述べておきます。

- 基本的にはデバイスロストしない
- Windows 10 の環境では追加のインストールなしで使用できる

従来の DirectX 9.0c では、デバイスロストが容易に発生しました。そして、そのデバイスロストの対策としてのコードもそれなりに記述する必要がありました。Windows 10 の環境と DirectX 9.0Ex の組み合わせでは、基本的にデバイスロストは発生しません。よって今までのデバイスロストからの復帰コードが必要だった箇所は不要になります。

また Windows 10 では DirectX 9.0Ex は初期状態から使用可能です。追加のコンポーネントのインストールは不要です。昔は「最新の DirectX コンポーネントをインストールしてください」と、よく言われたものでした。これは、 D3DX というライブラリに依存したプログラミングをしたために引き起こされました。初期状態で DirectX 9.0Ex が使えるメリットを殺さないためにも、 D3DX を使用しないプログラミングをこの本では説明していきます。

2.2 プロジェクトの準備

本章で作成するアプリケーションについてのプロジェクト設定について説明します。プロジェクト名やアプリケーション名といったものは、各節での名前で作成してください。画像は後節の ウィンドウアプリケーション で使用するもので表示しています。

Visual Studio 2017 を開いて、「新しいプロジェクト」を選択して作成します。「Visual C++」のカテゴリーの中にある「Windows デスクトップ」を選択します。この中に、「Windows デスクトップウィザード」という項目があります[*1]。これを使用します（図 2.1）。

[*1] 「Windows デスクトップアプリケーション」というプロジェクト構成もあって紛らわしいです。

3.2 DirectX 9.0 (Ex) の論理的なレンダリングパイプライン

現在においてのレンダリングパイプラインを説明しましたが、 DirectX 9.0 では、頂点シェーダーとピクセルシェーダーしかプログラマブルな部分はありません。言い換えれば、DirectX 9.0 においては、この２つのシェーダーのこと以外を知る必要がありません。

DirectX 9.0 が出た当時と現在の論理的なレンダリングパイプラインが合っていないという点を考慮しつつ、現在の用語と摺り合わせながら、先ほどのパイプラインを再構成してみたものが 図 3.2 となります。厳密には一致しなかったり、適用順序が変わる箇所もありますが大体こんな感じになります。

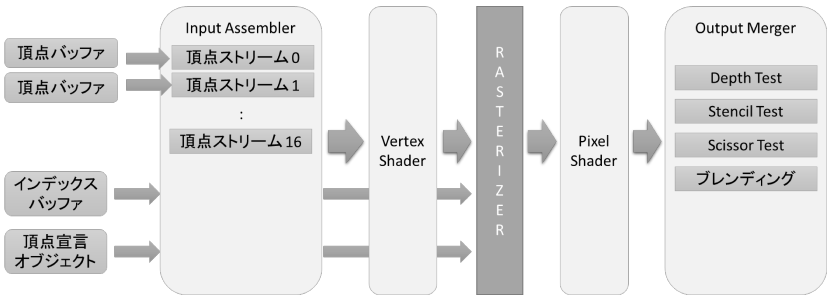

図 3.2: DirectX9 レンダリングパイプライン

3.3 Vertex Shader と Pixel Shader

シェーダーへの入出力の関係を図示すると 図 3.3 のようになります。

頂点シェーダーの場合の入力は頂点データ（位置、法線など）ですが、ピクセルシェーダーの場合には、各ピクセルにおける各属性情報となります。また出力先も、頂点シェーダーはラスタライザへ、ピクセルシェーダーはフレームバッファへと変わりますが、基本的に計算して結果を書き込むという動きは両者で同じです。またそれ以外のアクセスできるユニットも変わりない感じです。

第 6 章
配布について

アプリケーションが出来たら配布して使ってもらうことになるでしょう。このときに問題になるのが、自分の手元では動くのに、相手先ではエラーが出てうまく動かないといった点です。本章では、これについて注意すべきポイント、対応すべき内容についてまとめていきます。

6.1　リリースビルドを使う

通常開発しているときには、デバッグビルドを使用しています。これはブレークポイントを設定して、変数の値を見たり、ステップ実行を行ったりというときに都合のよいものとなっています。しかし、配布時ではそういったものは不要です。リリースビルドを使用すると、デバッグに役立つ情報が削ぎ落とされ、最適化がかかり実行時のパフォーマンスがよくなります。ビルドを切り替えるには、 図 6.1 のようにプルダウンから構成を選び直すだけです。

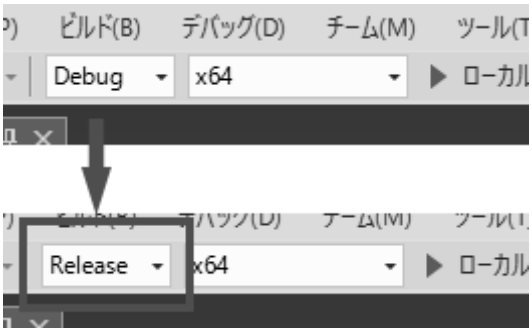

図 6.1: リリースビルドの選択

○ デバッグ情報

Visual Studio のプロジェクト設定では、なぜか標準状態でリリースビルドでもデバッグ情報の生成が有効になっています。配布するときには、デバッグ情報が完全に不要な場合もあるでしょう。このときには、リンカーの設定で、”デバッグ情報の生成” を "いいえ" に変更します。このデバッグ情報が生成されたままだと、実行体にビルド時のファイルパス関連情報が埋まっていて、（もしかすると）恥ずかしい思いをすることがあるかもしれません。