ジェネラティブ VJ ソフトウェアの開発を容易にするシステム
― VJ から始まるエンジニアへの道 ―

1. 背景

本プロジェクトの背景として最も影響しているものは、椎名のプログラマ、VJ としてのキャリアである。椎名は VJ というリアルタイム映像表現に大きな魅力を感じ「VJ になりたい」という強いモチベーションを持った。さらに、自ら VJ ソフトウェアを開発し VJ パフォーマンスを行う人々が居ることを知り、椎名はプログラミングに強い興味を持った。

加えて、その時期に椎名は VRDG+H #1 Keijiro Takahashi × DUB-Russell というライブパフォーマンス動画に出会い強い感動と衝撃を受ける。本ライブパフォーマンスの映像表現は Unity で実装されており、所謂ジェネラティブ VJ である。

これらの出会いにより、椎名はプリレンダ VJ とジェネラティブ VJ、プログラミングに強い興味を持ち、プログラマ、VJ としてのキャリアを歩み始めた。キャリアを歩む中で、VJ を広めることで次の世代の人々が VJ をきっかけに新たなキャリアを歩めるのではないかと考えた。

まず VJ パフォーマンスを行うことの楽しさをより多くの人が体験できないかと考え、市販の VJ ソフトウェアよりも安価で、かつ本格的なプリレンダ VJ ソフトウェア SynapseRack を開発、販売した。

次にジェネラティブ VJ を始めるための敷居を下げることを目指した。これらが本プロジェクトの背景である。

2. 目的

本プロジェクトの目的は以下の 3 つである。

① ジェネラティブ VJ ソフトウェアの開発を容易にすること
② ジェネラティブ VJ の楽しさを多くの人が体験できること
③ ノンプログラマが本プロジェクトをきっかけにプログラマとして成長すること

①はジェネラティブ VJ ソフトウェアを開発するためにはプログラミングに関する知識が必要であるという部分を解決することである。本プロジェクトではノードベースプログラミング環境「Fluss」を実装・提供することで、アーティストやノンプログラマがノーコードでジェネラティブ VJ ソフトウェアの開発を行えることを目指す。

②は多くの人が VJ パフォーマーとして、または VJ パフォーマンスを見るお客さんとしてジェネラティブ VJ の楽しさを体験できることを意味する。現状、ジェネラティブ VJ の認知度は低い。こういった問題を、①を通して多くの人を開発者かつ VJ パフォーマーとして促すこと、さらに VJ パフォーマーにはなるつもりが無い人にも、2 月に開催される成果報告会にて椎名が行うジェネラティブ VJ のデモと発表内容を通してコンピュータ技術を用いた VJ の存在と楽しみ方を知ってもらうことを目指す。

③は椎名が VJ の感動をきっかけにプログラマとしてのキャリアを歩んだという経験から、本プロジェクトを通して誰か一人でも椎名と同じように VJ というエンターテイメントから感動

を受け、ジェネラティブ VJ として活躍し、プログラマとして成長、キャリアを歩むことを目指す。

3. 開発の内容

3.1. ノードベース UI の実装

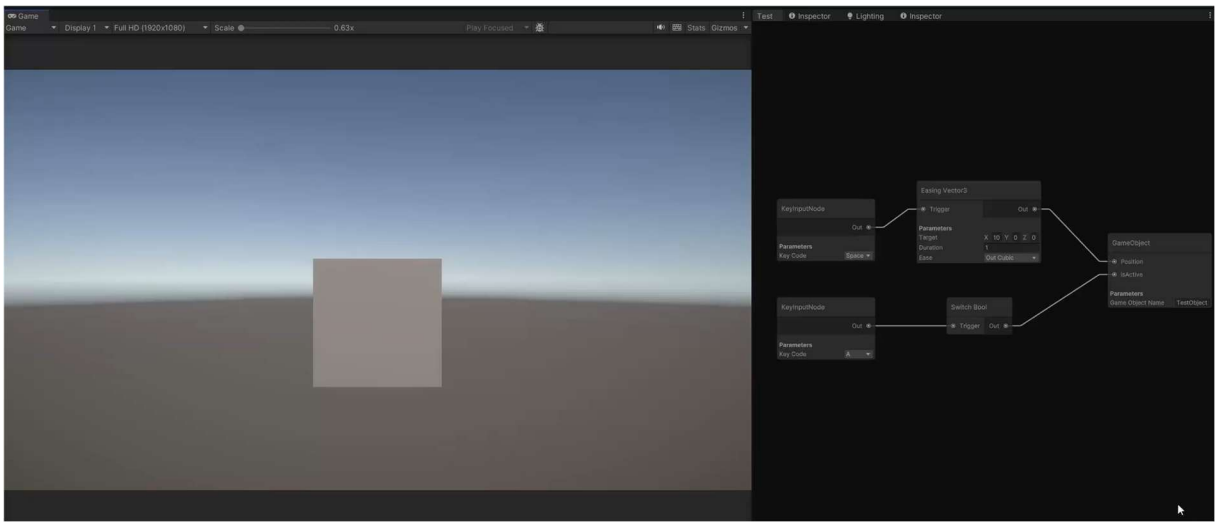

図 1：Unity エディタ拡張として実装したノードベースプログラミング環境

Fluss は Unity から提供されている Graph View というノードベース UI を実装するための API を使用し、Unity エディタ拡張としてノードベースプログラミング環境を実現している。図 1 はプロトタイプのスクリーンショットである。Graph View API について調査し、簡単なノードベースプログラミングを実装した。より効率的にノードベースプログラミング環境を実装するため Graph View API をラップしたライブラリ NodeGraphProcessor を使用し開発を続けた。

3.2. 変数を参照する機能の実装

Unity でジェネラティブ VJ ソフトウェアのプログラミングを行うためには、変数と参照が必須である。

Unity で扱われているデータはエディタ上で扱うのか、ランタイム（シーン）で扱うのかで大きく 2 つに分かれている。Unity の仕様として、これらはライフサイクルが違うため、エディタ上のものをランタイムで参照したり、逆にランタイムでエディタ上のものを参照したりすることは困難である。そのため、ランタイム上にある 3D オブジェクトを、ノードベースプログラミング環境のエディタ上で参照する機能を実装するためには工夫が必要だった。既に Unity Visual Scripting や、Unity Timeline といったエディタ上の機能がランタイム上のものを参照している機能があったため、それを調査し、変数を参照する機能の実装を行った。

3.3. ホットリロードの実装

Unity C#にはコンパイル時間が長いという欠点がある。逆にほかのノードベースツールではノードをつないだら即座に結果が反映され、プログラミングのイテレーションが早く回せる

という利点がある。そこで、本プロジェクトも Unity 上でジェネラティブ VJ ソフトウェアの開発のイテレーションを素早く回せるようにし、よりよい開発体験を提供するためにホットリロード機能を実装した。

3.4. 様々なノードの実装

ジェネラティブ VJ ソフトウェアの開発に必要な機能を持ったノードを開発していった。実装したノードの一部を挙げると「ランダムな数字を出力するノード」「キーボードの入力を受け付けるノード」「GUI の入力を受け付けるノード」「数値をアニメーションさせるノード」「複数のカメラを切り替えるノード」「3D オブジェクトの座標を書き換えるノード」などである。

3.5. ノードベースプログラミング環境「Fluss」の完成

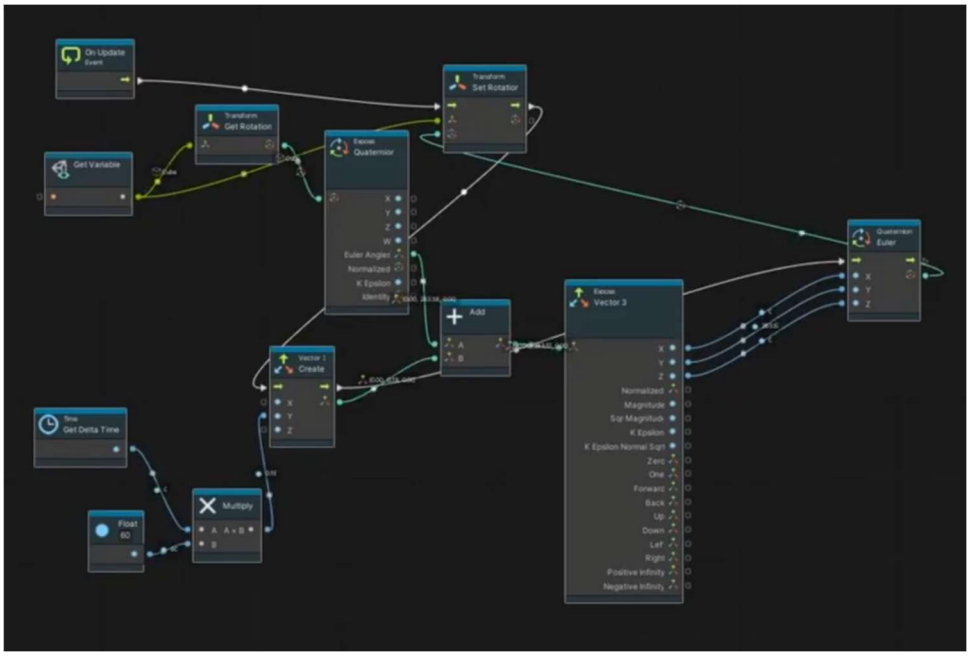

図 2：Unity Visual Scripting で実装した様子

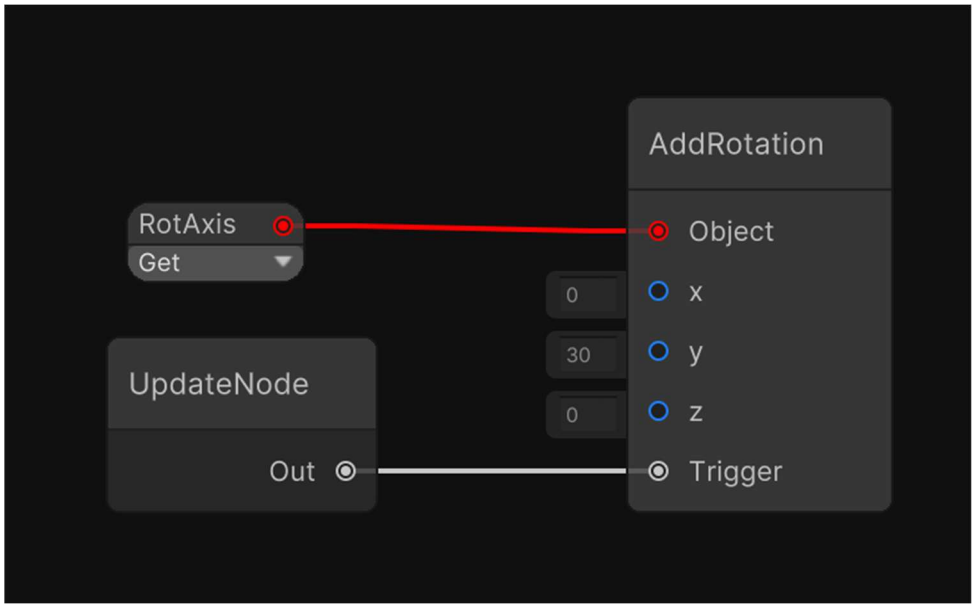

図 3：Fluss で実装した様子

図 2 は Unity に標準で搭載されているノードベースプログラミング環境 Unity Visual Scripting で「オブジェクトを回転させる」という実装をした時のスクリーンショットである。対して図 3 は Fluss で同じ動作を実装した時のスクリーンショットである。Unity Visual Scripting でもジェネラティブ VJ ソフトウェアの開発は行えるが、Fluss と比べてノードの数が多く、より複雑になっている。

それに対して本プロジェクト Fluss はたった 3 つのノードで「オブジェクトを回転させる」という実装を行えている。このように Fluss はジェネラティブ VJ ソフトウェアの開発に特化した、ノンプログラマやアーティストが扱える容易なノードベースプログラミング環境となった。ジェネラティブ VJ ソフトウェア開発においてはソフトウェア開発やゲームプログラミングのように「なんでもできる」必要はないが、容易さを優先した結果「誰が作っても同じような表現になる」ことは問題である。Fluss はそのようなバランスを調整し、容易でかつ、プログラマブルであることを実現した。Fluss は現在オープンソースソフトウェアとして MIT License のもと公開されている。

4. 従来の技術（または機能）との相違

本プロジェクト以外にもノードベースプログラミング環境は沢山ある。ジェネラティブ VJ 表現に使えるものに限っても TouchDesigner、vvvv、cables.gl、Notch などが挙げられる。これらと Fluss の相違は大きく 3 つある。

1 つ目はノードベースのプログラムがコンパクトであることが挙げられる。これはリアクティブプログラミングというプログラミング手法をベースにノードベースの処理を実装したことによって、ノードベースプログラミングの UI が宣言的であることを有効的にノードベースの処理に活用できたからである。

2 つ目はジェネラティブ VJ 表現に特化しているという点である。一般的なノードベースプログラミング環境では、なんでもできる事を目標にかなり細かい単位の機能を 1 つのノードとして扱っている。しかし、ジェネラティブ VJ ソフトウェアの開発において「なんでもできる」必要はない。Fluss ではジェネラティブ VJ 表現に必要な細かさで機能を分け、VJ 表現に特化したうえで、プログラマブルであるという特徴を持っている。

3 つ目はユーザーがソフトウェアエンジニアとしてのキャリアを歩む可能性を持っている点である。Fluss ではカスタムノードを実装することができる。これはユーザーがテンプレートに沿った C#コードを書くことで、容易に新しい機能を持ったノードを実装することができる

この機能をきっかけにノンプログラマやアーティストが C#に触れプログラマとして成長する可能性を秘めている。さらに Fluss は Unity 上で動作しており、Unity は商業的に使用されることが多い。つまり、ユーザーがこのカスタムノードを実装することで得た知識は商業的に価値があるといえる。

5. 期待される効果

本プロジェクトで開発した Fluss は Unity 上で動作するノードベースプログラミング環境となった。その結果、Unity エンジニアから 3D アーティストまで幅広いユーザーが使用することができる。そういった点で今までプログラマでかつアーティストである人しか触れることができなかったジェネラティブ VJ、ジェネラティブアート、デモシーンといったコンピュータアート

カルチャーにより多くの人が参入し盛り上がる効果が期待できる。
　また本プロジェクトを期にコンピュータアートに触れたアーティストはプログラマ、エンジニアとしてのキャリアを歩む可能性、逆にプログラマ、エンジニアはアーティストとしてのキャリアを歩む可能性があり、背景にあった椎名の想いである「次の世代の人々が VJ をきっかけに新たなキャリアを歩めるのではないか」というモチベーションを達成し、多くの人のキャリアを変える効果も期待できる。

6.　普及（または活用）の見通し

　本プロジェクトでは Unity 上で動作するジェネラティブ VJ ソフトウェアの開発に特化したノードベースプログラミング環境「Fluss」を実装することができた。Fluss は GitHub にて OSS として公開しており、Unity や Blender 等の DCC ツールは使えるがプログラミングはできないといったアーティストがジェネラティブ VJ を始めるきっかけとして今後機能することを期待している。そのためにドキュメントの整備や、自身が Fluss を使って様々なジェネラティブ VJ を行うこと、ノードの追加などのジェネラティブ VJ ソフトウェアの開発のための新機能の追加などを考えている。
　また、背景でも触れたように、本プロジェクトは自分の人生へ大きな影響を与えたジェネラティブ VJ という文化へ貢献していきたいという強いモチベーションによる活動の 1 つである。引き続きこのモチベーションを元に、ソフトウェア開発に限らず、自身が VJ として活躍し多くの人に感動を与えること、イベント運営者として文化が続いていくための土台を作っていくことなど、人生を通して様々なアプローチを行っていく。

7.　クリエータ名（所属）

　椎名　貫太（大阪電気通信大学　総合情報学部　デジタルゲーム学科　学部 4 年）

（参考）関連 URL

- 開発成果を公開している Web サイト
  https://mitou.sainakey.com/
- ジェネラティブ VJ ソフトウェアの開発を容易にするシステム「Fluss」のリポジトリ
  https://github.com/SainaKey/Fluss.BRP-public
- プリレンダ VJ ソフトウェア「SynapseRack」の公式 Web サイト
  https://synapserack.com/
- 開発者個人 Web サイト
  https://www.sainakey.com/