1. プロジェクト設定について

プロジェクト設定にあるマップ&モード内のローカルマルチプレイヤーの設定を下の画像を見て確認して ください。



次にエンジン→レンダリングの中にある自動露光をオフにします。



次に、エンジン→インプットの中にあるコントローラの入力設定を以下の画面通りに設定します。 マッピングの名前を間違えると正しく動作しませんので注意願います。

エンジン - インプット				
	デフォルトとして設定	エクスポート	インポート	デフォルトにリセット
∩ これらの設定は現在書き込み可能な DefaultInput.ini に保存されます。				6
▲ Bindings				
動作と軸マッピングは、インブットビヘイビアとこれを起動するキーの間にレイヤー概念を挿入する 軸マッピングは連続的な範囲のインプットを受け取る一方で、動作マッピングはキーを押したり離	ることにより、マップキーと軸に・ したりするインプットを受け取り	インプットビヘイビア)ます。	をタイミング良く渡	オメカニズムです。 🕜
🔺 Action Mappings 🕂				
⊿ Jump + ×				
🐷ゲームパッド方向キー上 🚽 Shift 🛄 Ctrl 🔛 Alt 🛄 Cmd 📰 🗶				
✓ Crouch + ×				
🔄 ゲームパッド方向キー下 🚽 Shift 🛄 Ctrl 🔜 Alt 🔛 Cmd 🛄 🗙				
A Punch + X				
🥁ゲームパッドXボタン 🗸 Shift 🔤 Ctrl 📄 Alt 📄 Cmd 📄 🗙				
⊿ Kick + ×				
🥁ゲームパッドAボタン 🔻 Shift 🔤 Ctrl 🔤 Alt 🔤 Cmd 📰 🗙				
✓ Right_input				
🥁 ゲームパッド方向キー右 🚽 Shift 🔤 Ctrl 🔜 Alt 🔤 Cmd 📰 🗶				
▲ Left_input + ×				
🥌 ゲームパッド方向キー左 🚽 Shift 🔤 Ctrl 🔤 Alt 🔤 Cmd 🔜 🗙				
▲ Throw + ×				
🥁 ゲームパッド左ショルダー 🚽 Shift 🔤 Ctrl 🔤 Alt 🔤 Cmd 🔤 🗶				
🔺 Axis Mappings 🛨 💼				
⊿ Left ×				
🛫ゲームパッド方向キー左 🚽 Scale <mark>-1.0 💽</mark> 🗙				
⊿ Right + ×				
🛫ゲームパッド方向キー右 🛛 🔻 Scale 1.0 💽 🗙				

最後にレベルエディタ→プレイにて実行時のウインドウサイズを指定します。

縦横の比が16:9になるサイズを入力します。

(例:1280*720 / 1600*900 / 1920*1080)



これでプロジェクト設定は完了です。

プロジェクト内のコンテンツにある Stage フォルダの Start_Here ファイルを開いていることを 確認したら、実行を試してください。

(新規エディタウインドウでサイズが反映されたか確認してください。)



ゲーム中の操作方法については、<u>https://youtu.be/BXCtcZlAi-s</u>を見てください。

2. フォルダ構成とファイルについて

以下フォルダとファイルの概要を説明します。

■Blueprints フォルダ

Search Paths	▼フィルター 検索 Blueprints
▲ コンテンツ ▲ Fighting_template Blueprints ▷ Characters ▷ Effect ▷ Sounds ▷ Stages ▷ Widget Splash	BP_Common_ BP_Fighting BP_Global Camera
	3 アイテム

BP_CommonController: このプロジェクトのメインプログラムファイルです。

		プレイヤーの入力を受け取り、キャラクタや各種ゲージの制御を行います。
		Player2 の制御は CommonController-1 という形でファイルが生成され、
		Controller_ID という変数の値により、Player1 と2の識別が行われます。
BP_	_FightingGameMode	:このプロジェクトの初期設定を行うファイルです。
		基本的に設定を変更する必要はありません。
ΒP	GlobalCamera	: カメラをコントロールしているプログラムファイルです。

- ■Character フォルダ
 - ・Char01フォルダ

キャラクタ毎の専用ファイルが保存されています。

キャラクターを追加する際は Char02、 Char03 とフォルダを増やしてください。

Animations



キャラクタのアニメーションが保存されています。

• Blueprints

Search Paths D	▼フィルタ ▼	検索 Blueprints									рe
▲ コンテンツ ▲ Fighting_template ■ Blueprints ▲ Characters	A.	A	d de		*	AL.	-	they -			NAK CE
Char01 Animations Blueprints Effect	AM_Attack_in _Air	AM_CR_ AM_D Attack	amage AM_Da	h AM_Guard	AM_Other	AM_Result	AM_ST_ Attack		BP_Char01_ bullet	BP_Char01_ SP	Char01_Anim BP
▷ ■ Materials ■ Mesh ■ Sounds ■ Taxturee	A.										
P I Effect P Stages	Char01_Walk _Blend									• #	テオプションマ

• BP_Char01

キャラクタの攻撃・ダメージの判定 BOX の設定や相手の Capsule component とオーバラップした 場合の処理を記載しています。

Char01_AnimBP

アニメーションの再生を制御しています。

BP_Char01_Bullet

必殺技使用時に発生するアクター(FireBall)を制御しています。

BP_Char01_SP

SP ゲージがフルの時に使用できる必殺技使用時に発生するアクター(IceBall)を制御しています。

- ・その他アニメーションモンタージュ
 立ちやしゃがみなどキャラクタの状態毎にアニメーションモンタージュを設定しています。
 このファイル内で、通知を利用して攻撃判定の発生・消失のタイミングを指定しています。
- Char01_Walk_Blend(ブレンドスペース 1D)
 待機状態と前進、後退のアニメーションを制御しています。

・Effect/Sounds フォルダ



キャラクタ専用必殺技用のパーティクルやサウンドが保存されています。

・ Materials/Mesh/Textures フォルダ



UE4 標準キャラクターであるグレーマンの素材が保存されています。

■Effect フォルダ



ヒットやガードしたときのパーティクルとその素材が保存されています。

■ Sounds フォルダ

ヒット時やウィジェットで使用するサウンドが保存されています。

■ Stages フォルダ

ステージの素材やレベルのファイルが保存されています。

■ Widget

メニュー画面等で必要なテクスチャを制御する Blueprint が保存されています。

3. ステージ追加の仕方

Stages フォルダに素材を格納するフォルダとレベルファイルを追加します。

ファイルとフォルダの名称は、Stage02、Stage03、Stage04 ・・・と追加してください。



Stage02のレベルファイルを開き、レベルを作成します。

ここでは、事例として、BP_Sky_Sphere と DirectionalLight を追加しておきます。

次に地面となる Plane を追加しますが、壁があるレベルにする場合は、Plane の面積を 20m*20m に、

壁がないレベルにする場合は、面積を 60m*60m にすることをお勧めします。

(ここでは 60m*60m の Plane を追加します。)

次に、座標軸の確認をします。

このプロジェクトで、カメラに向かって正面となるのは、以下の向きの座標軸です。



ここまで確認できたら、好みのレベルを作成してください。

ここでは、Paragonの無料アセットを事例として作成します。

Stage02 のライティングビルドをして、保存したら、StartHere レベルを開き、Stage02 をサブレベルと して追加します。

Stage02 の地面の座標を X:0 Y:2000 Z:-70 としてください。



上記の状態で、Stage02のライティングシナリオを ON にして、ライティングビルドを行います。



これ以降、StartHere レベルでサブレベルを編集する際は、可視化するのはどれか一つにしてください。

例えば、Stage01 と Stage02 を同時に可視化すると警告が表示されます。

次にゲーム実行画面内のステージセレクト画面で表示するステージ用の画像を準備します。

Stage02_Small.png(250*250pixel)

Stage02_img.png (640*480Pixel)

作成した画像は Widget→images フォルダヘインポートしておきます。

次に Widget ブループリントを開き、デザイナー画面で以下のように、Stage02_Small.png を指定します。



[Grid_SelectStage]にある Stage02 を選択し、Uncheckedimage に Stage02_Small.png を選択します。 次に Widget ブループリントのグラフ画面で Select Stage の関数を開き、Set Brush from Texture に Stage02_img.png の指定と選択後に処理を続けるノードの接続を行います。



これでビルドを行い、セレクトステージ画面とバトル画面で以下のように反映されていれば OK です。



4. キャラクター追加の仕方

キャラクターを追加する場合は、Characters フォルダに Char02 フォルダを追加します。

ここでは、Paragon アセットの Phase を事例として説明します。



キャラクター BP のファイル名称を BP_Char02、アニメーション BP を Char02_AnimBP とします。

まずは、キャラクター BP の BP_Char02 を開き、Mesh や Capsule component、攻撃、ダメージを判定 する Box Collision の設定を行います。

			3 m -				- 31	É	1	
Chard2(self) CapuleComponent (開茶) ArrawComponent (開茶) Cross_attack Damage_for_Initow Throw Cross_attack Damage_for_Throw Throw Cross_attack Damage for Throw Cross_attack Dudge of Damage Sudge of Damage for Throw	+コンホーネントを追加 - 検索	Q	コンパイル	保存	Browse	Find	クラス設定	クラスのデフォルト	シミュレーション	プレイ
1 CapabuleComponent ((伊希) ArrowComponent (伊希) Damage Amm,R Damage Amm,R Foot,R Foot,R Foot,R Thow Damage_for_thulet Cross_attack Damage_for_throw Throw	1 Char02(self)	ľ	🛄 ビューボート	- ×		ントグラフ	×			
ArrowComponent (推新) Damage Arm.R Arm.R Arm.L Fool.R Fool.R Damage_for_bullet Codes_attack Damage_for_bullet Codes_attack Damage_for_Throw Throw three (推新) Codes_attack Damage_for_Throw Throw three (推新) Codes_attack Damage_for_Throw Throw three (TATACHARM C C C C C C C C C C C C C C C C C C C	🔺 🏮 CapsuleComponent (継承)	- Id								
▲ (Mesh (能求) Damage A mm_R A mm_L FooLR FooLL Damage for_bullet Cross_attack Damage for_throw Throw CharacterMovement (能承) Cross_attack Damage for_throw Throw CharacterMovement (能承) Cross_attack Damage for_throw Throw Cross_attack Damage for_throw Cross_attack Damage for throw Cross_attack Damage for throw Cross_Attack D	🔨 ArrowComponent (継承)			ペクティフ	57	ティンクあ	2			
Damage Amr_B Amr_L Fool,R Fool,R Fool,R Damage_for_bullet Cross_attack Damage_for_Introw Throw CharacterMovement (他示) CharacterMovement (▲ 🏭 Mesh (継承)									
Am,A Am,A Foul Foul Damage_for_bullet Cross_stack Damage_for_Throw Throw CharacterMovement (継承) CharacterMovement (継承) Cross_stack Damage for_bullet Cross_stack Damage for_bullet Cross_stack Damage for bullet Damage for bullet Dudge of Damage for bullet Dudge of Dudge	Tamage									
ArmL Fool R Fool R Cross.attack Damage.for_bullet Cross.attack Damage.for_bruw Throw CharacterMovement (港家) CharacterMovement (BaracterMovement (Arm_R									
Frouch ForoL Damage_for_bullet Cross_attack Damage_for_Throw Throw CharacterMovement (催発) マイブルーブリント サ研究 イベフルーブリント サ研究 イベライトーブリント サインノトライト可能 ・	Arm_L							Jacob Contraction		
Portuge for bullet Cross_attack Pamage_for_throw Throw ↑ CharacterMovement (継承) Ard 70 Ard 7	Foot I							MESSAW.		
Cross_attack Damage_for_Throw Throw CharacterMovement (推革) マイブループリント / + 新規追加 K密 200 - イベラフ - ロカフフ - 町数 (30 オーバーライド可能) - ケコンストラクションスクリプト - マクロ - ひ - Judge of Damage - Judge of Damage for bullet - Judge of Damage for bullet - Judge of Damage for Throw - Z820 - Damage for Throw - Z820 - Damage for Throw - Z820 - Damage for bullet - Damage fo	Damage for bullet							1000		
Damage_for_Throw Throw CharacterMovement (継承)	Cross_attack						A			
 Throw CharacterMovement (推承) Cr/TJUEF × ● TATATUE 70 UP K ● TATATUE 70 UP K	Damage_for_Throw								to the	
 [↑] CharacterMovement (様芽) [↑] CATACTERMOVEMENT (様芽) [↑] STACHONE SET SET SET SET SET SET SET SET SET SE	🗊 Throw							ZH C		
 ▲ マイブループリント、 ★ 新規追加 ● 該案 ● ● ● ● ○ ○ ● ▲ が分フ ● ▲ パクフ ● ▲ パクフ ● ● ● ○ → パーライド可能) ● ● ↑ ○ → ストラクションスクリプト ▲ マグロ ● ● ○ → パーズ → パーブ ● ○ → ○ → ○ → ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	[◇] †CharacterMovement (継承)						H	9		
 + 新規道加 ● 酸素 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	🚨 マイブルーブリント 🛛									
4/977 ・ ・ ・ ・ イベントグラフ 4開数 (30 オーバーライド可能) ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	+新規追加 - 検索 🔎 👁	••								- Lines Steam
▶ ■ イベントグラフ ▲開数 (30 オーバーライド可能) ↑ コンストラクションスクリプト ▲マクロ ↓ Judge of Damage for bullet ↓ Judg	▲グラフ	+							A	
▲開放(30 ホーバーライド可能) ・ ・ ・ ・ ・ ・ ・ コンストラクションスクリプト ▲マクロ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	▷ 📑 イベントグラフ									
* 「コンストラクションスクリプト オマクロ ・ * Judge of Damage * Judge of Damage for bullet * Judge of Damage for Throw #250 ■ Player01 Collision ■ Battlestart	▲関数 (30 オーバーライド可能)	+								
↓ Judge of Damage ↓ Judge of Damage for bullet ↓ Judge of Damage for Throw ★ Judge of Damage for Throw ↓ Judge of Damage	◆ コンストラクションスクリプト								51	
は Judge of Damage Judge of Damage for bullet Judge of Damage for Throw 4変数 ⇒ コンポーネント Player01 Collision Battlestart	▲マクロ	+						$\langle -4 $		
A Judge of Damage for bullet Judge of Damage for Throw Agg Agg Damage for Throw Agg Damage for Throw Data and the set of the set	🔅 Judge of Damage									
◆ Judge of Damage for Throw ▲変数 ● コンポーネント ● Player01 Collision ■ Battlestart	🗘 Judge of Damage for bullet									- 74- 1
▲変数 ト D コンパイル結果 ● 結果検索 ×	💭 Judge of Damage for Throw									
▷コンパイル結果 ○結果検索 ×	▲変数	•					1.A		7	
Player01 Collision Colli							< P			
Collision 2 コンパイル結果 ① 結果検索 × Battlestart		۲L								
Battiestart			∑ コンパイル線	果	🔎 結果	検索	*			
Begin judge for OL Uファレンスの検索に関数またはイベント名を入力	Begin judge for OL		リファレンスの検	索に関数ま	たはイベン	小名を入力	1			
■ Begin judge for OL	Battlestart		リファレンスの検	索に関数ま	たはイベン	小名を入力]			

Capsule component

Shape の値の以下のように設定します。

⊿ Shape			
Capsule Half Height	70.0	S 3	
Capsule Radius	32.0	N 1	
	5	₹	

Collision と Tag を以下のように設定します。

✓ Collision		
Simulation Generates Hi	a 🔳	
Phys Material Override	None なし →	
Generate Overlap Events	15 🔽	
Can Character Step Up C	C No 🔻	
┛ コリジョンプリセット	Custom 🔻 ⊐	
Collision Enabled	Collision Enabled (Query and Physics) -	
Object Type	WorldDynamic 👻	
	無視する オーバーラブロック	
コリジョンレスポンス	0 0 0	
トレース応答		
Visibility		
Camera		
オブジェクト応答		
WorldStatic		
WorldDynamic		
Pawn		
PhysicsBody		
Vehicle		
Destructible		
	₹	
l⊿ Tags		
▲ Component Tags	1 配列エレメント 🕂 💼 🤉	
0	Player 🗢 🗢	

• Mesh

Animation の Anim Class は"なし"にしてください。

キャラクターセレクト時に Animation Blueprint は必要ないため、バトル前に手動で呼び出します。



Mesh の位置と回転を以下のように設定します。

▲ トランスフォーム				62			2		
位置 🚽	X	0.0	Y	0.0		Z	-70.0	l	t
回転 🛨	X	-0.0 ° 🖍	Y	0.0 °	2	Z	270.0 °		Ð
拡大•縮小 ▼	X	1.0	Y	1.0		Z	1.0	1	

Damge Box Collision (Mesh のサブコンポーネントとして追加します。)
 位置と回転、Shape、コリジョンを以下のように設定します。
 位置と Shape についてはキャラクターの体格に合わせて設定しますので、変更しても構いませんが

高さをあまり低くしてしまうと、他のキャラの上段攻撃が当たらなくなるので注意してください。



・他の Box Collision について

Arm_R/Arm_L/Foot_R/Foot_L/Damage_for_bullet/Cross_attack/Damage_for_Throw/Throw に ついては Char01 の設定値を参考に位置と回転、Shape、親ソケット、コリジョンを設定します。 各 Box Collision の名称について注意してください。

これからイベントグラフに Char01の Blueprint をコピーしますので、間違えていると正しく動きません。

CharacterMovement について
 以下のように値を設定します。

▲ Character Movement: Jumping / Fa	lling				
Jump Z Velocity	420.0				
Braking Deceleration Falling	0.0				
Air Control	0.0	5			
Air Control Boost Multiplier	0.0	5			
Air Control Boost Velocity Threshold	0.0	5			
Falling Lateral Friction	0.0				
Impart Base Velocity X	N				
Impart Base Velocity Y	N				
Impart Base Velocity Z	S		Character Movement: Walking		
Impart Base Angular Velocity	S		Max Step Height	45.0	
Notify Apex			Walkable Floor Angle	45.0	•
			Walkable Floor Z	0.707107 🔊	
Character Movement (General Sett	ings)		Ground Friction	0.0	•
Gravity Scale	4.5	5	Max Walk Speed	600.0	
Max Acceleration	30000.0	5	Max Walk Speed Crouched	300.0	
Braking Friction Factor	2.0		Min Analog Walk Speed	0.0	
	0.0		Braking Deceleration Walking	30000.0	•
Use Separate Braking Friction			Sweep While Nav Walking	v	201340101
Crouched Half Height	40.0		Can Walk Off Ledges	Z	
Mass	60.0	t	Can Walk Off Ledges when Crouching		
Default Land Movement Mode	Walking 🔻		Maintain Horizontal Ground Velocity		
Default Water Movement Mode	None 🔻	t	Ignore Base Rotation		
—			_		

・イベントグラフについて

まずは変数を以下のように作成します。



次に Char01 のイベントグラフの Blueprint 全体を選択してコピー&ペーストします。

ただし④赤枠内はマクロがあり、マクロはコピーされませんので、Char01のイベントグラフでマクロを

展開したノードをコピー&ペーストして折りたたむかマクロ化してください。



次にキャラクタ個別で設定必要な箇所を説明します。

上図の赤枠①:キャラクターセレクト画面のキャラ選択前のアニメーションシーケンスを選択します。

上下2つあるのは Player1 と2の場合です。

上図の赤枠②: Player2 のキャラ選択時に Player1 と同キャラクターが選択された場合の処理です。 同キャラクターメッシュ内のマテリアルの色を変更するか、メッシュ自体を変更します。 メッシュ自体を変更した場合は、アニメーションシーケンスとアニメーションモンター ジュをそのメッシュ専用で別途用意する必要がありますので注意してください。

上図の赤枠③:キャラクター選択時のアニメーションシーケンスと使用するアニメーションブルー プリントを設定します。

これで Char02 ブループリントの設定は完了です。

次に Char02_Anim のアニメーションブループリントを設定します。

アニメーションブループリントを設定するためにはアニメーションシーケンスとアニメーション モンタージュが必要になるので、以下に一覧を記載します。

準備が必要なアニメーションシーケンス一覧

半開が必要なゲーバ	
立ち待機状態	立っている状態で何も入力されていないときのアニメーション。Char01の Idle が該当。
前歩き	前進するときのアニメーション。Char01のFwd_Walkが該当。
後ろ歩き	後退するときのアニメーション。Char01の Bwd_Walk が該当。
しゃがみ待機状態	しゃがんでいる状態のアニメーション。Char01のCrouch_Idleが該当。
しゃがみ→立ち移行	しゃがんでいる上から立つときのアニメーション。Char01の End_Crouch が該当。
ジャンプ開始	立ち状態からジャンプを始めるときのアニメーション。Char01の Start_Jump が該当。
空中待機状態	ジャンプが完了したときのアニメーション。Char01 の Loop_Jump が該当。
着地	ジャンプから着地するときのアニメーション。Char01の End_Jump が該当。
前ステップ	前ステップするときのアニメーション。Char01のFwd_Dashが該当。
バックステップ	バックステップするときのアニメーション。Char01の Bwd_Step が該当。
立ちパンチ	立ちパンチするときのアニメーション。Char01のST_Punchが該当。
立ちキック	立ちキックするときのアニメーション。Char01のST_Kickが該当。
ジャンプパンチ	ジャンプパンチするときのアニメーション。Char01の Air_Punch が該当。
ジャンプキック	ジャンプキックするときのアニメーション。Char01の Air_Kick が該当。
しゃがみパンチ	しゃがみパンチするときのアニメーション。Char01のCR_Punchが該当。
しゃがみキック	しゃがみキックするときのアニメーション。Char01のCR_Kickが該当。
立ちガード	立ちガードするときのアニメーション。Char01のGuardが該当。
しゃがみガード	しゃがみガードするときのアニメーション。Char01のC_Guardが該当。
立ち上段やられ小	立ち状態の上段やられ小のアニメーション。Char01の Damaged_Top が該当。
立ち上段やられ大	立ち状態の上段やられ大のアニメーション。Char01の Damaged_Top_L が該当。
立ち上段やられ中	立ち状態の上段やられ中のアニメーション。Char01の Damaged_Mid が該当。
しゃがみやられ	しゃがみやられするときのアニメーション。Char01のC_Damagedが該当。
空中やられ小	空中のやられ小のアニメーション。Char01の Damaged_inAir が該当。
空中やられ大	空中のやられ大のアニメーション。Char01の Damaged_inAir_L が該当。
カウンターやられ	カウンターでやられたときのアニメーション。Char01の Counter が該当。
KOやられ	地上で KO されたときのアニメーション。Char01 の KO が該当。
ダウン	ダウン属性の攻撃を受けたときのアニメーション。Char01の Down が該当。

15

Unreal Engine 格闘ゲーム テンプレートプロジェクト マニュアル

起き上がり通常	起き上がるときのアニメーション。Char01のGet_up01が該当。
起き上がりスタン	スタン状態から起き上がるときのアニメーション。Char01のGet_up02が該当。
スタン	スタン状態のアニメーション。Char01 の Stun が該当。
投げ	相手を掴むアニメーション。Char01 の Throw001 が該当。
投げ成功	相手を掴んで投げるアニメーション。Char01のThrow002が該当。
投げやられ	投げられるアニメーション。Char01の Damaged_Throw が該当。
投げ抜け攻め側	投げ抜けされた場合の攻撃側のアニメーション。Char01の Throw_tech_A が該当。
投げ抜け守備側	投げ抜けした場合のアニメーション。Char01の Throw_tech_D が該当。
必殺技(FireBall)	FireBal のアニメーション。Char01 の FireBall が該当。
必殺技(IceBall)	IceBallのアニメーション。Char01のSPが該当。
ラウンド勝利ポーズ	ラウンド勝利時のアニメーション。Char01の Win が該当。
ラウンド敗者ポーズ	タイムアウト時のラウンド敗者のアニメーション。Char01の Draw が該当。
ゲーム勝利ポーズ	ゲーム勝利時ののアニメーション。Char01の GameSet が該当。
チェンジサイド	キャラクターが向きを変えるときのアニメーション。Char01の Side_Change が該当。

準備が必要なアニメー	ションモンタージュ一覧
ST_Attack	立ち状態の攻撃アニメーションを登録します。AnimNotify で、カウンター攻撃を受ける時間帯、
	攻撃判定が発生する時間帯、必殺技へのキャンセルを受け付ける時間帯、攻撃判定消失、攻撃中
	フラグのオフを設定します。※
CR_Attack	しゃがみ状態の攻撃アニメーションを登録します。AnimNotify は ST_Attack と同様です。
Attack_in_Air	ジャンプ状態の攻撃アニメーションを登録します。AnimNotify は攻撃判定が発生する時間帯、攻
	撃判定消失、攻撃中フラグのオフを設定します。
Dash	前ステップやバックステップのアニメーションを登録します。AnimNotify はステップのオフとバ
	ックステップの無敵時間の解除タイミングを設定します。
Guard	立ちガードとしゃがみガードのアニメーションを登録します。AnimNotifyの設定はありません。
Damage	ダメージを受けたときの各種アニメーションを登録します。AnimNotify はダメージフラグをオフ
	するタイミングの設定やカウンターフラグのオフを設定します。
Throw	投げや投げ抜けのアニメーションを登録します。AnimNotify で、カウンター攻撃を受ける時間
	帯、攻撃判定が発生する時間帯、投げ抜けを許容する時間帯、攻撃判定消失、攻撃中フラグのオ
	フを設定します。
Other	ダウンや起き上がり、スタン、KO、向きの変更のアニメーションを登録します。
	AnimNotify で、ダウンや起き上がりフラグのオフを設定します。
Result	勝利、敗北のアニメーションを登録します。AnimNotify の設定はありません。

※Char01のアニメーションモンタージュを Char02のメッシュでリターゲットし、アニメーションモンタージュ内の アニメーションシーケンスを入れ替えれば、AnimNotify を再利用でき、通知のタイミングを調整するだけで済みます。

次に Walk_Blend(ブレンドスペース1D)を開き、前進と後退と待機状態を設定します。



次に以下のように値を設定します。

水平軸の Name を Speed にすることが重要で他の値は参考値となります。

▲ Axis Settings			
⊿ 水平轴			
Name	Speed 🗢		
Minimum Axis Value	-300.0		
Maximum Axis Value	300.0		
Number of Grid Divisions	16 🔊 🤋		
Interpolation Time	0.0		
Interpolation Type	Averaged Interpolation -		

これでアニメーションシーケンスとモンタージュの準備が完了しましたので、Char02_AnimBPを開き、 ステートマシーンの編集を行います。

まずは以下のように変数を設定します。



次にアニムグラフを以下のように設定します。



次にステートマシーンを以下のように設定します。



各項目の詳細は、Char01のステートマシーンを参考に設定してください。

次にイベントグラフの設定ですが、Char01_AnimBP からコピー&ペーストします。



全てコピー&ペーストしたらビルドしてください。 恐らく、いくつかのカスタムイベントでエラーが出ると思います。 コピー&ペーストしただけでは有効にならない場合があるので、その場合は同じノードを作り直して、 ビルドすると解消されます。

ビルドが通ったら、次に Play Montage や Montage Stop が参照しているアニメーションモンタージュを Char02 のモンタージュに設定し直します。

コピーしただけでは、Char01のモンタージュを参照しており、ビルドが通っても動作しません。

また、ダメージを受けたときのアニメーションを処理しているノードでは投げられたときのアニメー

ションが各キャラ固有になる場合がありますので、以下画像赤枠部分に、個別の設定をしてください。



Play Montage や Montage Stop の参照先をすべて Char02 用に変更したら Char02_AnimBP の設定は 完了です。

次に BP_Common_Controller の設定を行います。

変数の P01_Char_Num と P02_Char_Num をそれぞれ右クリックしてリファレンス検索を行い、

それぞれの検索先で Char01 の設定を参考にしながら Char02 の設定を行ってください。



例えば以下の画像は Player01 のキャラクターセレクト時の処理ノードになりますが、赤枠内の Char02 用の処理を追加しています。



全て処理を追加したら Common_Controller の設定は完了です。

最後にキャラクターセレクトとバトルで使用する画像を追加します。

まずは以下画像のようにデザイナー画面のキャラクターセレクトのグリッド背景に画像を追加します。

(125*300pixel)



次にイベントグラフ内の Select Character 関数内で以下画像のように参照 Texture を設定します。 Char_name02(300*50pixel): キャラクターセレクト時にキャラの下に表示される名前 Char_hp_name02(250*30pixel): バトル時に HP バーの下に表示される名前 Char02_P01/Char02_P02(304*335pixel): : バトル時に HP バーの横に表示されるキャラ画像

f Set Brush from Texture ターゲットは Image	f Set Brush from Texture ターゲットは Image		f Set Brush from Texture ターゲットは Image
• •	· · ·		• •
● ターゲット	 ターゲット 	P01 Char Num 2	🎐 ターゲット
Orange Char_name02 - Char_name	Char_hp_name02 - Char_hp_name02	• 9-4vh	Char02_P01 -
👝 Match Size 🛃	🕞 Match Size 🛃		🕒 Match Size 🗖
		Image P01 🗨	
ƒ Set Brush from Texture ターゲットは Image	ƒ Set Brush from Texture ターゲットは Image		ƒ Set Brush from Texture ターゲットは Image
• •			• •
ターゲット	● ターゲット	● P02 Char Num 2 ●	🔷 ターゲット
OP Texture Char_name02 - <	Char_hp_name02~ (~)	• 9-4yh	Char02_P02 -
💿 Match Size 🛃	🕞 Match Size 🗹	Image PO:	Match Size

ビルドして実行したら、以下画面のように反映されていれば OK です。



5. デバッグ時の注意点

動作確認をする際は、必ずフレームレートを確認して行ってください。

(アウトプットログで Stat fps コマンドを入力するか Tick に繋いだ PrintString に

1 を WorldDeltaSeconds で割った値をセットして表示させてください。)

フレームレートが下がっている状態で動作すると、フレームレートが下がっている状態でしか発生しない 不具合が発生することがあります。

以下画像のように Blueprint の実行を確認しながらプレイを行うとフレームレートが下がることを 確認しています。

これを解消するためにはマルチディスプレイを利用して、Blueprintの実行ウインドウとプレイウインドウ を別々のモニターで実行すると 60fps をキープしながら Blueprintの実行を確認できます。

ただし、Blueprintの実行確認範囲をズームアウトして広範囲にするとマルチディスプレイにしても、 フレームレートが下がる場合があります。



以上 RareEncounter Ver.1.0