３．ユニティちゃんをジャンプさせる

■スプライトの設定

学校サーバーからユニティちゃん\_ジャンプ2.pngをダウンロードしておく。

Sprites>Playerのフォルダにインポートする。

ユニティちゃん\_ジャンプ2.pngを選択して、待機の時と同じ設定をしていく。

Inspectorウィンドウから、  
SpriteModeをSingleからMultipleに変更。

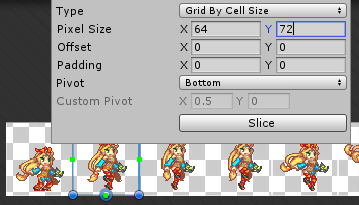
Pixels Per Unitを１にする。

FilterModeのBilinearからPoint(no filter)へ変更。

設定したらApply。

Sprite EditorでSliceする。  
TypeはGrid By Cell Size,Pixel Sizeはx64,y72、PivotはBottom

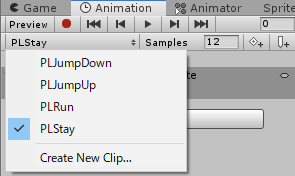
設定したらApplyで適応



■アニメーションの設定

Hierarchyのユニティちゃんを選択した状態（重要）で、  
AnimationタブのCreate New Clipを選択して、

PLJumpDownとPLJumpUpの二つを作成する。

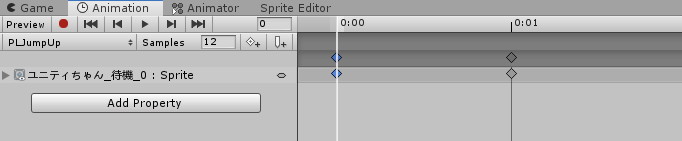


PLJumpUpを選択した状態でタイムライン上に

先ほど作成したユニティちゃんジャンプ2\_2と,ユニティちゃんジャンプ2\_3の二つだけを

選択して、タイムライン上にドラッグ＆ドロップする。

Samplesが60になっているとすごく早いので、12に変更しておく。

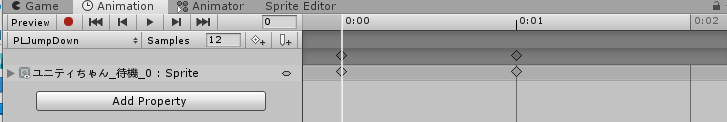


同様にPLJumpDownを選択した状態でタイムライン上に

ユニティちゃんジャンプ2\_6とユニティちゃんジャンプ2\_7の二つだけを選択して、

タイムライン上にドラッグ＆ドロップする。

Samplesを12に変更。

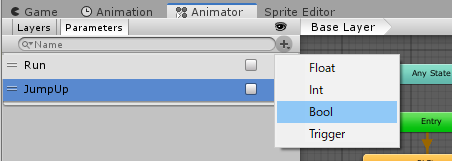


プレビューしてジャンプ上昇中とジャンプ下降中のアニメーションを確認する。

■アニメーターの設定

アニメーターでジャンプ上昇、ジャンプ下降を設定するようの変数を用意しておく。

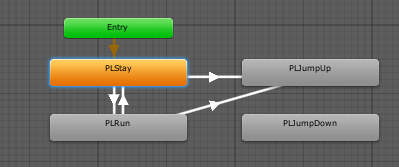
AnimatorのParametersを選択して、＋のボタンを押す。  
Bool型を選択して、JumpUpとJumpDownを作成する。



次はアニメーター上でジャンプ上昇と下降につながるように、矢印を追加していきます。

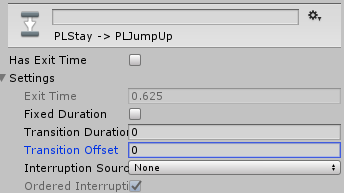
まずはジャンプ上昇へとつながるように繋ぎます。  
PLStayを選択して右クリック、Make TransitionでPLJumpUpにつなぎます。

同様にPLRunからも同じ手順をしてください。

  
  
矢印が追加できたら、追加した矢印を選択してInspectorを編集します。

Has Exit Timeのチェックを外す。（アニメーションの終了を待たず即切り替え）

Settings>Transition Dulationを0に。



Conditionsのプラスボタンを押してJumpUpを選択してtrueに設定

  
これでJumpUpがtrueになったときにジャンプ上昇アニメが再生されます。

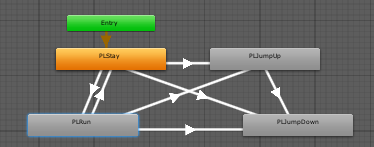
追加した２つの矢印に同じ設定をしてください。

次はジャンプ下降に矢印を繋ぎます。

歩き、走り、ジャンプ上昇から下降に繋がる可能性があるので、

全部繋いでいきます。

Stay>JumpDown、Run>JumpDown、JumpUp>JumpDownの３つの矢印が追加されました。

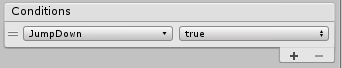


３つの矢印が追加できたら、追加した矢印を選択してInspectorを編集します。

Has Exit Timeのチェックを外す。（アニメーションの終了を待たず即切り替え）

Settings>Transition Dulationを0に。

Conditionsのプラスボタンを押してJumpDownを選択してtrueに設定

  
これでJumpDownがtrueになったときにジャンプ下降アニメが再生されます。

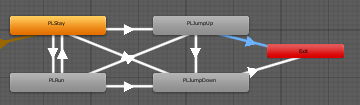
追加した3つの矢印に同じ設定をしてください。

次にジャンプ下降からは着地、歩きに移行する可能性があるので、

その設定をしていきます。

Exitに矢印を繋ぐとEntryに戻るので、それを利用します。

JumpUpとJumpDownからExitに繋ぎます。

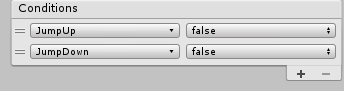


矢印の設定は以下の通りです。

Has Exit Timeのチェックを外す。（アニメーションの終了を待たず即切り替え）

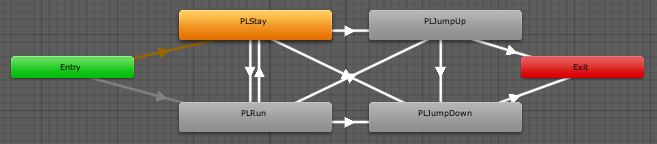
Settings>Transition Dulationを0に。

Conditionsのプラスボタンを押してJumpUp、JumpDownを選択して両方falseに設定

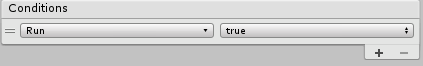
  
これでジャンプ中じゃなければ終了へ向かいます。

追加した2つの矢印に同じ設定をしてください。

ExitからはEntryに戻るのですが、Entryからは待機にしか行かないようになっているので、  
それを左右移動していたら歩きに移行させたいので矢印を追加します。



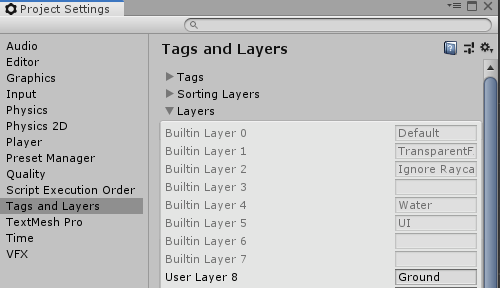
EntryからPLRunにつないだ矢印の設定を変更します。



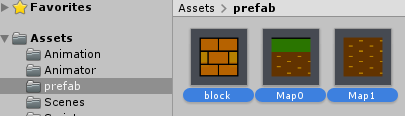
これでアニメーションの準備は終了です。

・地上チェック用のレイヤー設定をする

Edit>Project Settingsを開く。  
Tags and Layersを選択し、UserLayer 8 にGroundを追加する。



Prefabsに入っている地形すべてにレイヤー設定をする



全て選択した状態で、インスペクターのLayerをGroundに設定する。



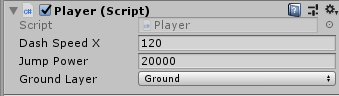
・ジャンプ用にPlayerスクリプトを修正する

追加するもの

・地形チェック処理（地面に立っているかの判定）

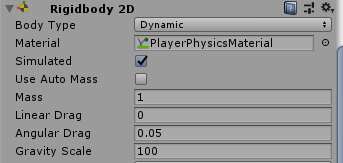
・ジャンプ処理

スクリプトの追加が終わったら、GroundLayerにGroundを設定する



ジャンプ力と重力の調整をする  
現状の設定だとちょっとふんわりしている気がするので、重力を変更。

Gravity Scaleを50から100に



ジャンプ中壁に向かって入力すると引っかかってしまうのがわかる。  
（横入力をしていると、落下しない）

これは壁との摩擦が設定されているせいなので、それを修正する。

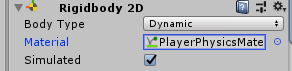
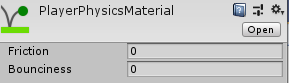
AssetesフォルダにMaterialフォルダを作成する。

作成したフォルダに右クリックからCretate>Physics Material2Dを選択して

PlayerPhysicsMaterialを作成。



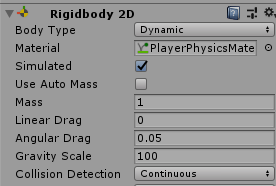
摩擦を０に設定して、ユニティちゃんにアタッチする。



これで壁に引っかからなくなっていることを確認する。

当たり判定の衝突判定をより正確にするため、

Collsion DetectionをDiscreteからContinuousに変更する。



これで一通りはＯＫだが…

落下する間際のところに立つと、落下している判定になって  
アニメーションが実行される。  
（なお、実際は地面に立っている）

その他いろいろと改良したスクリプトを作成したので、

それを更新してください（サーバーの授業３回目のplayer2.cs）

□追加修正点

・FixedUpdateメソッドの追加。

この内部で物理演算関連の処理を行う。  
※物理演算はFixedUpdate内部で行われているので、ここで物理演算関連の処理を行うのが本来は正しい。

・移動関連の入力をUpdate()内で取得、rigidbody2d関連の処理をすべてFixedUpdate()内に移動

・ジャンプ関連の移動値を修正

・地形チェックの処理を足元３点（左下、中央、右下）をチェックするように修正