Unityはじめるよ
～武器を持たせてみる～

統合開発環境を内蔵したゲームエンジン
http://japan.unity3d.com/

※いろんな職業の方が見る資料なので説明を簡単にしてある部分があります。正確には本来の意味と違いますが上記理由のためです。ご了承ください。
この資料内の一部の画像、一部の文章はUnity公式サイトから引用しています。

武器を持たせてみる

今回はスクリプトから

武器の持ち替えができるように

制御してみます。

■まずは、武器を持たせる原理

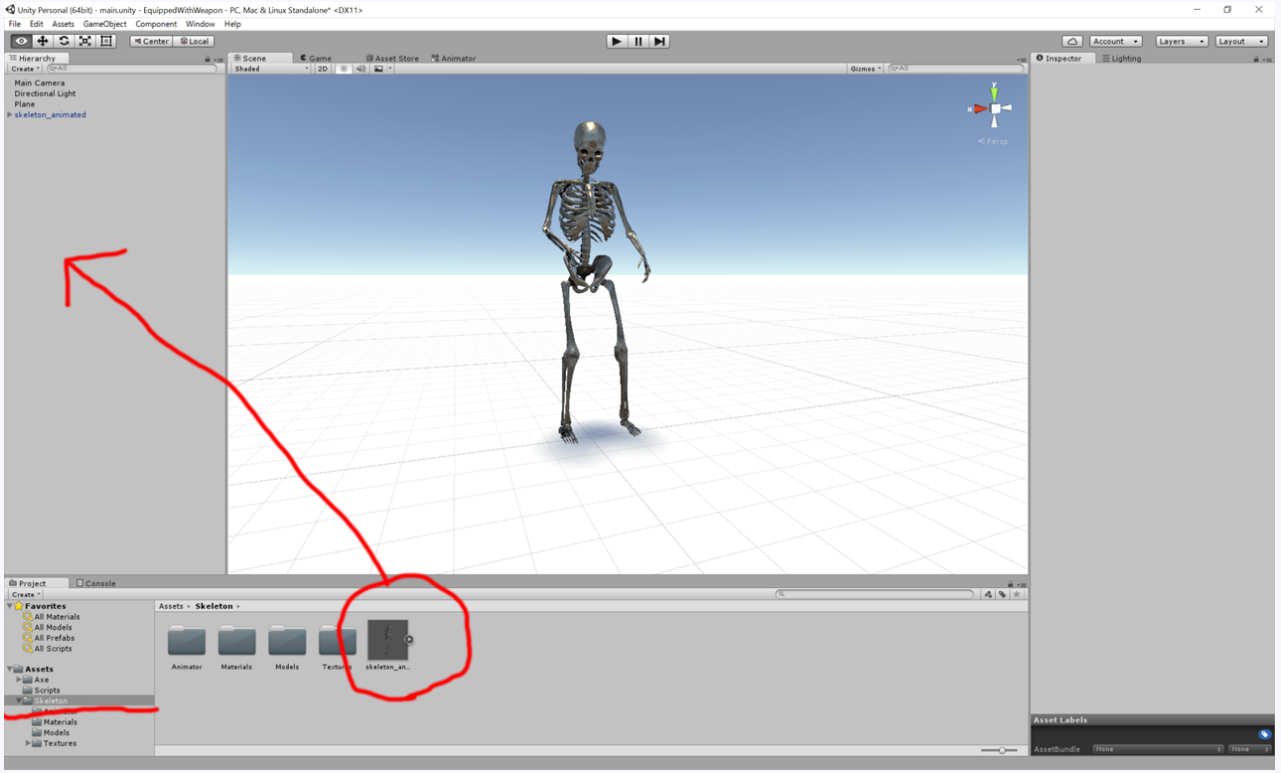
３Dキャラクターをアニメーションさせる為に、ボーンが仕込んであるよね。

装備させたい武器を「手のボーンの子」に設定してあげれば、手の動きに合わせて武器を動かすことができるんだ。

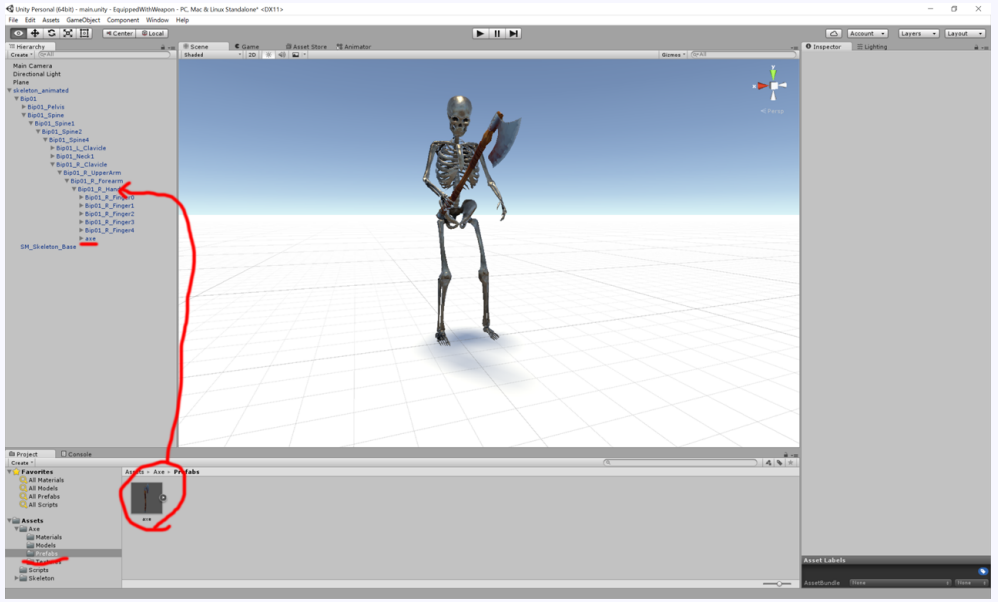
実にシンプル。

ちょいと試してみよう

Projectビューの「Sleleton/Skeleton animated.FBX」をHierarchyビューにドラッグ＆ドロップ



Projectビューの「Axe/Prefabs/axe.prefab」を
Hierarchyビューの「Skeleton/..../Bip01_R_Hand」
にドラッグ＆ドロップ



位置と向きと大きさがおかしいので直す。
Position(-0.0418, 0.0231, -0.0987)
Rotation(291.87, 6.4, 2.9)
Scale(0.2, 0.2, 0.2)

ちゃんと手に武器を持てました。

だけどエディタでこの作業を行うと、
アプリ配信中に武器を追加なんてことが
難しいよね。

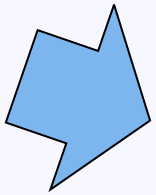
そんじゃこれをスクリプトからやって
しまおうと。

■下準備

スクリプトから簡単に制御するために、
武器の大きさ、向き、位置を調整しておく。

武器を自作する場合は問題ないけど、
アセットストアからDLする場合は、
大きさや向き、原点がバラバラなので、
Unity上で事前に修正しておく。
その際、**原点は手で持つ部分**にしておく。

例えば「axe」というオブジェクトがあり、
原点がモデルの中心にあった場合



空のゲームオブジェクトを作成し、
axeを空のゲームオブジェクトの子に設定し、
親の原点の位置に、手を持つ部分を移動させる。

試してみよう

■下準備２

見えない武器オブジェクト

をキャラクターの手の子として
追加しておく。
武器の位置や向きの調整用として用意し、
すべての武器で共通の
Transformの値として利用する。

※一旦見える武器を持たせて位置や向きの調整を行い、 調整が終わったら、
その武器からコンポーネントを削除して空のゲームオブジェクトにする。

試してみよう

下準備はOKだ。
スクリプトを作成しよう。

■どんなスクリプトを書けばいいんだ？

●やりたいことは・・・

・武器Prefabの読み込み
（今回はResourcesからにしとく）

・キャラクターの手に持たせる

・汎用的な作りとする

ふむふむ。つまり、

・武器装備関数を呼び出した時に、
武器Prefabをインスタンス化する。

・キャラクターの手に持たせるので、
キャラクターにアタッチするコンポーネントとする。

・武器の管理は外部で行ったほうが良いので、
武器装備関数はpublicにし、
別スクリプトから呼ぶようにする。

スクリプトは用意した。
ソースを確認するぜ。

武器装備変更スクリプト

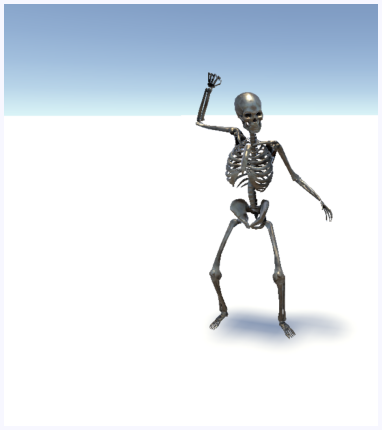
```
1 using UnityEngine;
2 using System.Collections;
3
4 public class EquipmentManager : MonoBehaviour {
5
6     public string   WeaponTemplateName;         /// 見えない武器のゲームオブジェクト名
7
8
9     private GameObject  m_WeaponTemplateName;   /// 見えない武器ゲームオブジェクト
10    private GameObject  m_Weapon = null;        /// 差し替え用武器用ゲームオブジェクト
11
12
13    // Use this for initialization
14    void Start () {
15
16        // 見えない武器を探す
17        var children = GetComponentsInChildren<Transform>( true );
18        foreach ( var transform in children )
19        {
20            if ( transform.name == WeaponTemplateName )
21            {
22                m_WeaponTemplateName = transform.gameObject;
23            }
24        }
25    }
26
27
28    /// <summary>
29    /// 武器を装備する.
30    /// </summary>
31    /// <param name="name">Name.</param>
32    public void EquipWeapon(string name)
33    {
34        // すでに作られていたらまず削除
35        if (m_Weapon != null) {
36            Destroy (m_Weapon);
37            m_Weapon = null;
38            Resources.UnloadUnusedAssets ();
39        }
40
41        // Prefabをインスタンス化
42        m_Weapon = Instantiate(Resources.Load(name), m_WeaponTemplateName.transform.position, m_WeaponTemplateName.transform.rotation) as GameObject;
43        // 見えない武器の子として登録
44        m_Weapon.transform.parent = m_WeaponTemplateName.transform;
45    }
46 }
47
```

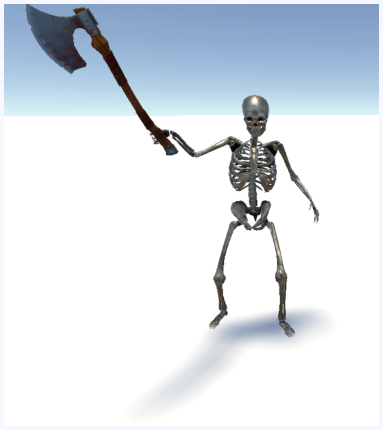
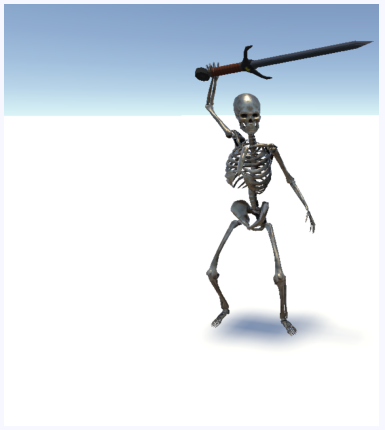
呼び出し側

```
using UnityEngine;
using System.Collections;

public class GameManager : MonoBehaviour {

    public GameObject   Charcter;   /// キャラクターゲームオブジェクト

    /// <summary>
    /// 武器の種類.
    /// </summary>
    public enum WeaponTypes:int
    {
        AXE,            /// 斧.
        SWORD,          /// 剣.
        SICKLE,         /// 鎌.
        NUM,            /// 数.
    }

    private int         m_weaponNo = 0;         /// 武器番号

    /// 武器Prefabのファイルパス
    private string[]    m_weaponsPath = new string[] {
                                                "axe",
                                                "sword",
                                                "sickle"
                                                };


    private float       m_keyInterval = 0f;     /// 連続でキーが押されないように


    private EquipmentManager    m_charcterEquipmentManager;

    // Use this for initialization
    void Start () {
        m_charcterEquipmentManager = Charcter.GetComponent<EquipmentManager> ();
    }


    // Update is called once per frame
    void Update () {

        // キーが連続で押せないように制御
        if (Time.time - m_keyInterval > 0.5f) {

            // 右矢印キーが押されたら次の武器へ
            if (Input.GetKey (KeyCode.RightArrow) == true) {
                m_keyInterval = Time.time;
                NextWeapon ();
            }
            // 左矢印キーが押されたら前の武器へ
            else if (Input.GetKey (KeyCode.LeftArrow) == true) {
                m_keyInterval = Time.time;
                PrevWeapon ();
            }

        }
    }

    /// <summary>
    /// 次の武器へ
    /// </summary>
    private void NextWeapon ()
    {
        m_weaponNo++;
        if (m_weaponNo >= (int)WeaponTypes.NUM) {
            m_weaponNo = 0;
        }
        // 武器を変更
        m_charcterEquipmentManager.EquipWeapon (m_weaponsPath[m_weaponNo]);
    }


    /// <summary>
    /// 一つ前の武器へ
    /// </summary>
    private void PrevWeapon ()
    {
        m_weaponNo--;
        if (m_weaponNo < 0) {
            m_weaponNo = (int)WeaponTypes.NUM - 1;
        }
        // 武器を変更
        m_charcterEquipmentManager.EquipWeapon (m_weaponsPath[m_weaponNo]);
    }

}
```

```
60
61      /// <summary>
62      /// 次の武器へ
63      /// </summary>
64      private void NextWeapon ()
65      {
66          m_weaponNo++;
67          if (m_weaponNo >= (int)WeaponTypes.NUM) {
68              m_weaponNo = 0;
69          }
70          // 武器を変更
71          m_charcterEquipmentManager.EquipWeapon (m_weaponsPath[m_weaponNo]);
72      }
73
74
75      /// <summary>
76      /// 一つ前の武器へ
77      /// </summary>
78      private void PrevWeapon ()
79      {
80          m_weaponNo--;
81          if (m_weaponNo < 0) {
82              m_weaponNo = (int)WeaponTypes.NUM - 1;
83          }
84          // 武器を変更
85          m_charcterEquipmentManager.EquipWeapon (m_weaponsPath[m_weaponNo]);
86      }
87
88  }
```

装備変更できたぜ！



ご清聴ありがとうございました