?????

このステップで、宝箱のアニメーションを作成し、開く。ただし、主人公が事前に鍵を持っていることを確認しなければならない。また、遷移のないアニメーションの切り替えも紹介する。

?????

シーンに宝箱のモデルをシーンに追加しましょう。

Coin と Key

と同様に、空のゲームオブジェクトを作成し、

3Dモデルを子オブジェクトとして追加した方がおすすめ。

サイズが多少小さいので

3Dモデル（子オブジェクト）のサイズは

2倍ぐらい大きくしましょう。

なお、ちゃんとぶつかるように、宝箱はコライダーが必要。

3Dモデルの子オブジェクトに

BoxCollider

を追加し、サイズを調整してください:

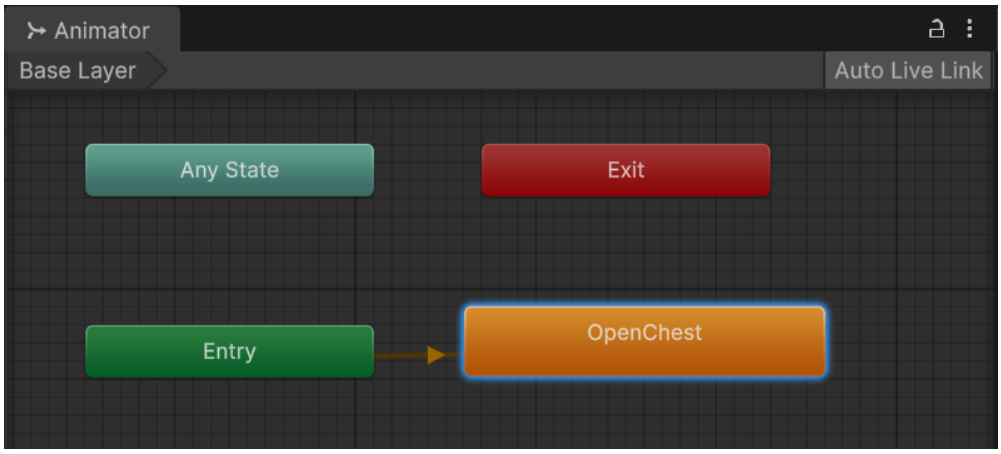
???????

もし、主人公が会議を持っていれば、宝箱が開くので、ふたが開くアニメーションが必要。アニメーション
パネルを使用し、「 OpenChest」作ってください:

フレーム	0	30
lid の回転	0, 0, 0	-120, 0, 0

なお、ループしないように、出来上がったアニメーションを Project フォルダーで探し、 Loop Time
のチェックを外してください

このまま実行すると、宝箱が勝手に開いてしまうので、アニメーションコントローラーの調整が必要。現状
を見ると:



ゲーム開始のときに「 OpenChest
」が実行するので、勝手に開いていしまうことが確認できる。そうしないように、空のアニメーション「
Empty」を作成しましょう。アニメーターパネルの中に右クリックし、 Create State → Empty
を選択してください。

この新しいアニメーションの名前を「 Empty
」にし、デフォルトアニメーションにしましょう。アニメーションの上に右クリックし、「 Set as Layer Default State」を選択しましょう、これで、ゲーム開始の時、何もしない。

今回のアニメーションは、ただの切り替えなので、遷移を使用しない。このため、パラメータを作成する必要なく、 Transition も使わない。

?????

今回のスクリプトは:

- 何か衝突のとき、
- 衝突したゲームオブジェクトが Inventory があるのかを確認 → なければなにもしない
- Inventory の中に鍵の数を問い合わせる。
- 鍵なければ → なにもしない
- 1つ以上あれば、鍵を消費し、宝箱が開く。
- 開いていれば、2度もチェックしないこと。

??????????

では「 OpenWithKey」のスクリプトを作成しましょう

```
// 鍵を持っていれば開く
public class OpenWithKey : MonoBehaviour
{
    // すでに開いている?
    private bool isOpen = false;

    // 衝突の時
    private void OnCollisionEnter(Collision other)
    {
        // 開いていれば、何も処理しない
        if (isOpen)
```

```
            return;

        // 衝突してきたゲームオブジェクトのInventoryを取得
        Inventory inventory = other.gameObject.GetComponent<Inventory>();

        // Inventoryのないオブジェクト（ゾンビなど）を衝突する可能性があるので
        // インベントリが存在することを確認
        if (inventory == null) // ない!
            return; // なにもしない

        // 鍵の数を求める
        int count = inventory.GetCount("Key");

        // 1つもない? なにもしない
        if (count < 1)
            return;

        // 鍵を消費
        inventory.UseItem("Key", 1);

        // 遷移せず、直接にアニメーションを再生
        Animator anime = GetComponent<Animator>();
        anime.Play("OpenChest");

        // 2度も実行しないように、開いていることにする
        isOpen = true;
    }
}
```

??????????????????

```
// 衝突してきたゲームオブジェクトのInventoryを取得
Inventory inventory = other.gameObject.GetComponent<Inventory>();

// Inventoryのないオブジェクト（ゾンビなど）を衝突する可能性があるので
// インベントリが存在することを確認
if (inventory == null) // ない!
    return; // なにもしない
```

宝箱と何が衝突するかわからない。もちろん、 Inventory

コンポーネントを持っているプレーヤーの衝突を期待しているが、もしかしたら他のゲームオブジェクト（

落下してきた岩など）も衝突する可能性がある。

そのために、「衝突してきたオブジェクトが Inventory のコンポーネントがあるか」のを確認している。そのコンポーネントがないと、`inventory` が `null` になるので、必ずプレーヤーであることを保証ができる。

??????????????????????

```
// 遷移せず、直接にアニメーションを再生
Animator anime = GetComponent<Animator>();
anime.Play("OpenChest");
```

今まで、パラメータを使用し、遷移でアニメーションを切り替えが行ってきたが、直接に「このアニメーションを再生しろ」のもできる。今回の単純な切り替えで、これで十分。

??????????????

```
// 鍵の数を求める
int count = inventory.GetCount("Key");

// 鍵を消費
inventory.UseItem("Key", 1);
```

Inventory???

上記の２つのメソッドまだ作成していないので、実装しましょう。 Inventory クラスを編集し、

- 在庫を返す `GetCount` のメソッドを作成。引数は確認したいアイテムのタグ名
- アイテムを消費する `UseItem` のメソッドを作成。引数は消費したいアイテムのタグ名とその数

```
// アイテムの数を返す
public int GetCount(string item)
{
    // アイテム存在しなければ、ゼロを返す
    if (!itemCount.ContainsKey(item))
        return 0;
    return itemCount[item];
}

// アイテムを消費する
public void UseItem(string item, int count)
{
```

```
    // 現在の数を確認
    int nowCount = GetCount(item);

    // 足りたら、消費する
    if (nowCount >= count)
        itemCount[item] -= count;
}
```

「OpenWithKey」を宝箱のコライダーにアタッチし、確認してみましょう。

Revision #3
Created 2026-05-04 08:13:30 UTC by Menendez Francisco
Updated 2026-06-02 06:28:32 UTC by Menendez Francisco