????

[illegible]

?????

[illegible] Unity[illegible]

```
// [illegible]
EnhancedTouchSupport.Enable();
```

[illegible]

```
// [illegible]
Touch.onFingerDown += [illegible];

// [illegible]
Touch.onFingerMove += [illegible];

// [illegible]
Touch.onFingerUp += [illegible];
```

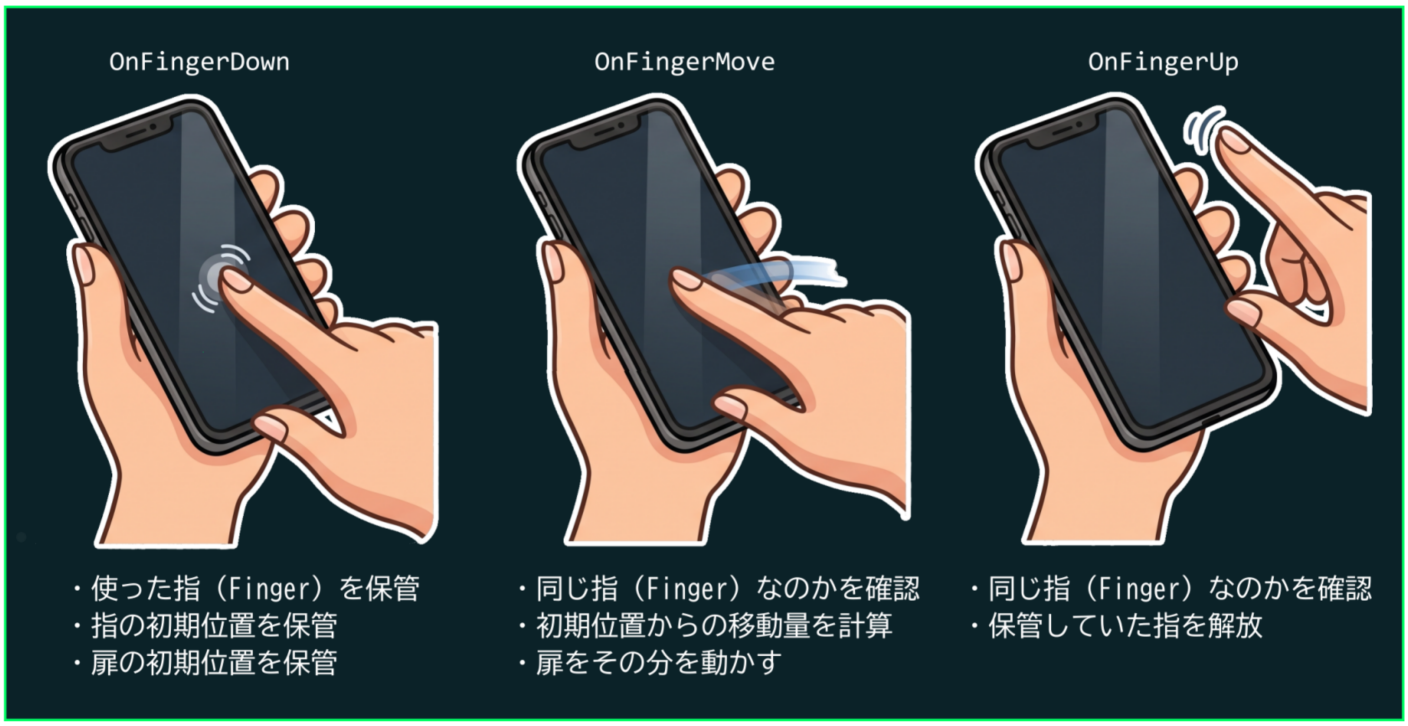
[illegible] Finger[illegible]

```
private void Start()
{
    //[illegible]
    EnhancedTouchSupport.Enable();

    //[illegible]
    Touch.onFingerDown += OnFingerDown;
}

// [illegible]
private void OnFingerDown(Finger finger)
{
```

```
//指の画面の位置を表示
Debug.Log(finger.screenPosition);
}
```

????????

扉を指でスライドするのを次の考えで実現できる:



初めて画面を触れるとき、指データを保管しておく。 Unity
がマルチタッチ（多数の指）に対応できるので、同じ指かどうかの確認が必要。そして、指が動くときに、
移動量を計算し、その分を使って扉をスライドする。最後に、指を離したら、保管したデータを解放し、処
理終了にする。

?????

DoorSlider のスクリプトを作成し、まず水平（ Horizontal）の扉を移動しましょう:

```
using UnityEngine.InputSystem.EnhancedTouch;
using Touch = UnityEngine.InputSystem.EnhancedTouch.Touch; // << ここ注意!

// 扉を移動（スライド）する
public class DoorSlider : MonoBehaviour
{
    // 水平の扉のグループ（親オブジェクト）
```

```
[SerializeField] private Transform horizontalDoors;

// 使った指
private Finger useFinger = null;

// 指の初期位置
private Vector2 fingerStart;

// 水平扉の初期位置（X軸のみ）
private float doorStartX;

private void Start()
{
    // タッチ機能有効化
    EnhancedTouchSupport.Enable();

    // 各種のタッチイベント登録
    Touch.onFingerDown += OnFingerDown;
    Touch.onFingerMove += OnFingerMove;
    Touch.onFingerUp += OnFingerUp;
}

private void OnDestroy()
{
    // 各種のタッチイベント登録解除
    Touch.onFingerDown -= OnFingerDown;
    Touch.onFingerMove -= OnFingerMove;
    Touch.onFingerUp -= OnFingerUp;

    // タッチ機能無効化
    EnhancedTouchSupport.Disable();
}

private void OnFingerDown(Finger finger)
{
    // マルチタッチできるため、すでに指一本を使っていたら、2本目から無視する
    if (useFinger != null) return;

    // 指を保管
    useFinger = finger;
```

```
        // 初期位置も取っておく
        fingerStart = finger.screenPosition;
        doorStartX = horizontalDoors.position.x;
    }

    private void OnFingerMove(Finger finger)
    {
        // マルチタッチできるので、同じ指かどうかを確認
        if (useFinger != finger) return;

        // 指の移動量を計算（画面のピクセル単位）
        Vector2 delta = useFinger.screenPosition - fingerStart;

        // 様々な画面サイズがあるので、移動量正規化しましょう（最大値：1）
        delta.x /= Screen.width;
        delta.y /= Screen.height;

        // 現在の位置からを取得し、X軸だけを更新
        Vector3 hPos = horizontalDoors.position;
        hPos.x = doorStartX + delta.x;
        horizontalDoors.position = hPos;
    }

    private void OnFingerUp(Finger finger)
    {
        // 指を解放する
        if (useFinger == finger)
            useFinger = null;
    }
}
```

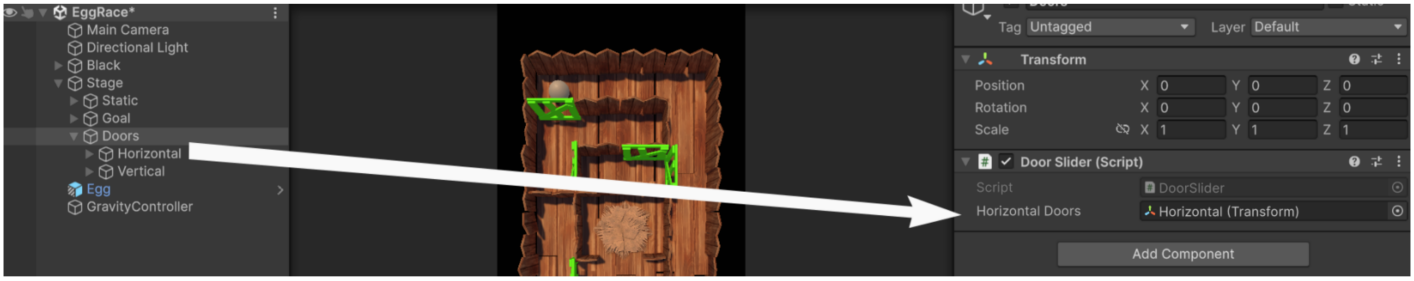
???????

いろんなスマホがあり、それぞれの画面サイズが異なるので、同じ指の動きでも大幅に異なる移動量が発生する可能性がある。例えば：

スワイプの長さを画面サイズ（ `Screen.width` と `Screen.height` ）で割れば、必ず 0 と 1 の間になり、どの画面サイズであっても、同じ数値になる。この計算が「正規化」という。

Unity???

スクリプトを「 Stage > Doors」にアタッチし、「 Horizontal Doors」に「 Stage > Doors > Horizontal 」を入れてください



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

タッチで扉を動くようにしたので、どれだけマウスを使っても、扉が全く動かない。そのため、 Unityは

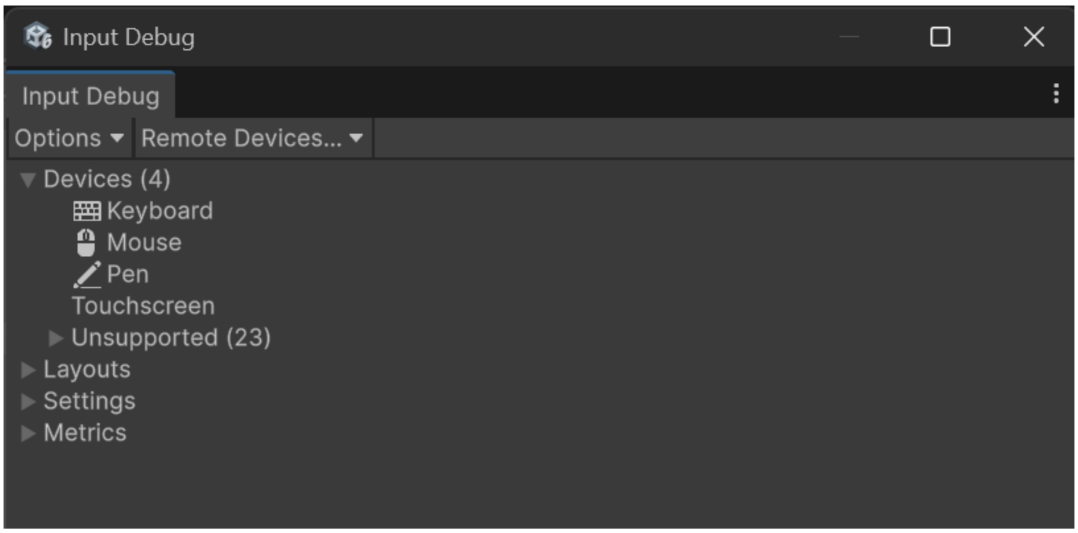
マウスを使ってタッチを疑似する

指1本のみで、マルチタッチが再現できない

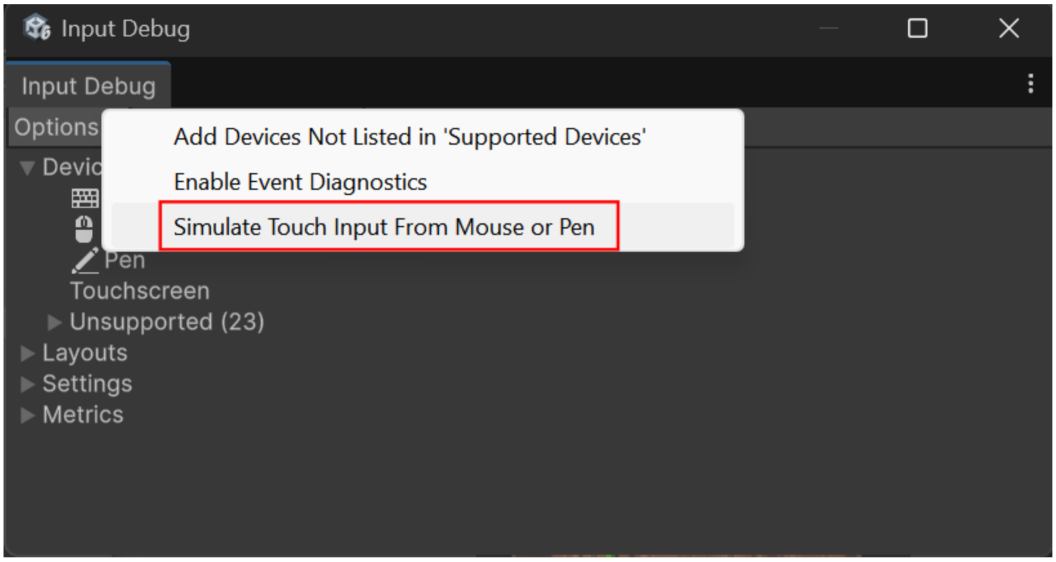
機能がある。もちろん、マウス1つしかないので、

が、今回のゲームなら十分。

疑似タッチを有効化するのは Window > Analysis > Input Debugger を選択し、次のウィンドウを開く。



このウィンドウの「Options」を開く、「Simulate Touch Input From Mouse or Pen」を選択すれば、マウスでタッチを疑似する。



??????????

これで、扉が移動できるようになったが、2つの問題が現れた。

1. 扉の動きが多少遅く感じる
2. 画面から消えるぐら、無限にスライドできる

この2つの問題を解決しましょう:

????

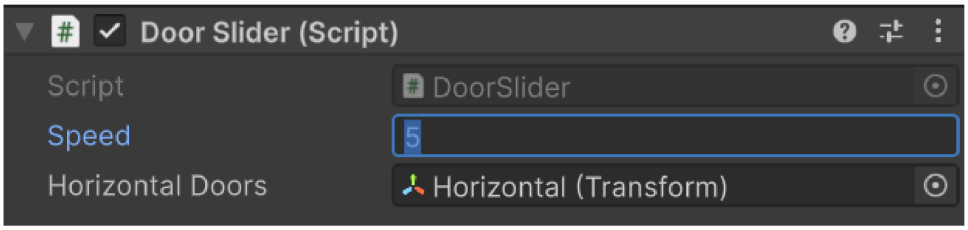
スクリプトに調整できる「speed」変数を追加し、`OnFingerMove`で使いましょう。先頭に:

```
// スライドの速度
[SerializeField] private float speed;
```

`OnFingerMove` では:

```
// 現在の位置からを取得し、X軸だけを更新
Vector3 hPos = horizontalDoors.position;
hPos.x = doorStartX + delta.x * speed; // 速度を適用
horizontalDoors.position = hPos;
```

Unity で適切に速度を設定し、確認してください



????

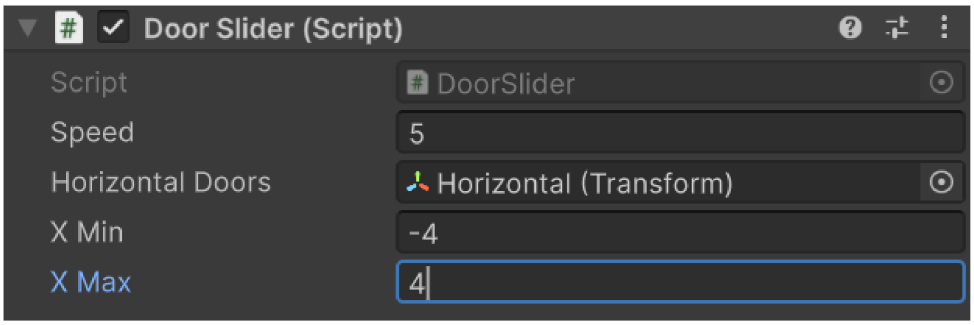
Unityで最大Xと最小Xの位置を設定できるようにし、 `OnFingerMove` で制限をかける。まず、スクリプトの先頭に:

```
// 最低 X 位置
[SerializeField]
private float xMin;

// 最大 X 位置
[SerializeField]
private float xMax;
```

そして、 `OnFingerMove` では:

```
// 現在の位置からを取得し、X軸だけを更新
Vector3 hPos = horizontalDoors.position;
hPos.x = doorStartX + delta.x * speed;      // 速度を適用
hPos.x = Mathf.Clamp(hPos.x, xMin, xMax);// 移動制限を適用
horizontalDoors.position = hPos;
```

Unity で適切に数値を設定し、確認してください



???????

同じ処理を使って、 DoorSlider スクリプトに垂直の扉が Z 軸で移動できるようにしましょう。スクリプトの先頭に:

```
// 垂直の扉のグループ（親オブジェクト）
[SerializeField]
private Transform verticalDoors;

// 最低 Z 位置
[SerializeField]
private float zMin;

// 最大 Z 位置
[SerializeField]
private float zMax;

//（他の[SerializeField]）

// 垂直扉の初期位置（Z軸のみ）
private float doorStartZ;
```

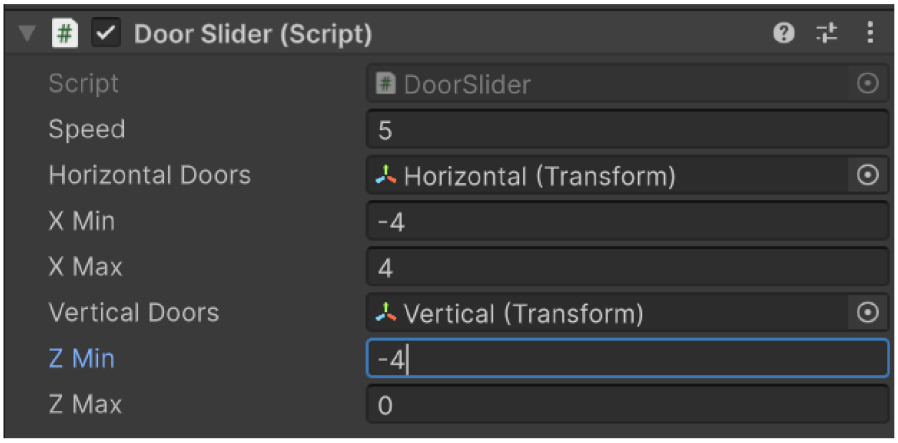
つぎ、 `OnFingerDown` では、垂直扉の初期位置を取る

```
// 初期位置も取っておく
fingerStart = finger.screenPosition;
doorStartX = horizontalDoors.position.x;
doorStartZ = verticalDoors.position.z;
```

そして、 `OnFingerMove` では:

```
// 現在の位置からを取得し、Z軸だけを更新
Vector3 vPos = verticalDoors.position;
vPos.z = doorStartZ + delta.y * speed;      // 速度を適用
vPos.z = Mathf.Clamp(vPos.z, zMin, zMax);// 移動制限を適用
verticalDoors.position = vPos;
```

最後に　Unity でパラメータを設定して、試してみてください:



?????

つぎ、端末を接続し、「　Build And Run」でタッチと傾きを使用し、卵を巣まで動かしてみてください。

Revision #2
Created 2026-07-15 03:06:20 UTC by Menendez Francisco
Updated 2026-07-21 07:13:50 UTC by Menendez Francisco