**Chapter 1 当たり判定**

Unityで当たり判定を行うためにはColliderコンポーネントとRigidBodyコンポーネントをゲームオブジェクトに貼り付ける必要があります。単なる物理シミュレーションを行うだけの場合はColliderとRigidBodyを張り付けるだけでいいのですが、それだけでゲームは作れません。今回はいくつかのシナリオを想定して、それを実現する方法を見ていきましょう。

**1.1 シューティングゲームで弾丸が壁にヒットした事を検出する場合。**

　弾の挙動は物理エンジンに任せることを想定します。この場合、弾のゲームオブジェクトにはColliderとRigidBodyを貼り付けることになります。そして、壁にはColliderを設定しましょう。

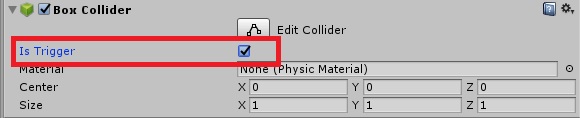
壁には弾がヒットしたら自身を削除するというスクリプトを記述します。弾がヒットした場合はOnCollisionEnterというコールバック関数が呼ばれます。

下記のようなスクリプトを記述して、壁のゲームオブジェクトに貼り付けてください。

|  |
| --- |
| public class Box : MonoBehaviour {  // Use this for initialization  void Start () {    }  // Update is called once per frame  void Update () {    }  //剛体と衝突した場合に呼ばれるコールバック関数。  void OnCollisionEnter(Collision collision)  {  Destroy(gameObject);  }  } |

**1.2 ある地点をキャラクターが通過したら、ゲームクリアなどといったイベントを検出する場合。**

　ゲームではゴール地点や、敵の罠、チェックポイントなどといった見えない当たり判定が必要です。しかし、今までの知識ではColliderを設置するとキャラクターはそこを通過することができなくなってしまい、このようなシナリオは実現できません。UnityではColliderのインスペクタでIs Triggerのチェックボックスをオンにすることで、このシナリオを実現できます。



IsTriggerをオンにした場合、先ほどのOnCollisionEnterはコールされなくなります。そして代わりにOnCollisionTriggerがコールされるようになります。

では下記のようなスクリプトを壁のゲームオブジェクトに貼り付けてください。

|  |
| --- |
| public class Box : MonoBehaviour {  // Use this for initialization  void Start () {    }  // Update is called once per frame  void Update () {    }  //剛体がトリガーに衝突した場合に呼ばれるコールバック関数。  void OnTriggerEnter(Collision collision)  {  Destroy(gameObject);  }  } |

**1.3 マウスやスマホのタッチを検出したい場合。**

　Colliderをゲームオブジェクトに貼り付けることに変わりありません。マウスが押された(タッチされた)場合はOnMouseDownコールバック関数が呼ばれます。

下記のようなスクリプトを壁のゲームオブジェクトに貼り付けてください。

|  |
| --- |
| public class Box : MonoBehaviour  {  // Use this for initialization  void Start()  {  }  // Update is called once per frame  void Update()  {  void OnMouseDown()  {  Destroy(gameObject);  }  } |

クリックすると壁を削除できたはずです。