

unity であそぼう！

目次

1. Unityとはなにか？

2. ゲーム開発の基礎知識

3. 解説 KitagawaShooting

4. まとめ

1. Unityとはなにか？

Unityとはなにか？

Unityは、IDEを内蔵するゲームエンジンである。

► IDE

統合開発環境（Integrated Development Environment）の略。

システム開発で使うさまざまなツールをひとまとめに提供してくれるソフトウェアのこと。

► ゲームエンジン

ゲーム開発に必要なさまざまな機能をパッケージ化したソフトウェアのこと。

物理演算や3DCGの機能をなども標準で備えている。

Unityを使えば、誰でも簡単にゲームを開発できる。

Unityで作られているゲーム



Pokémon
GO

原神
Genshin

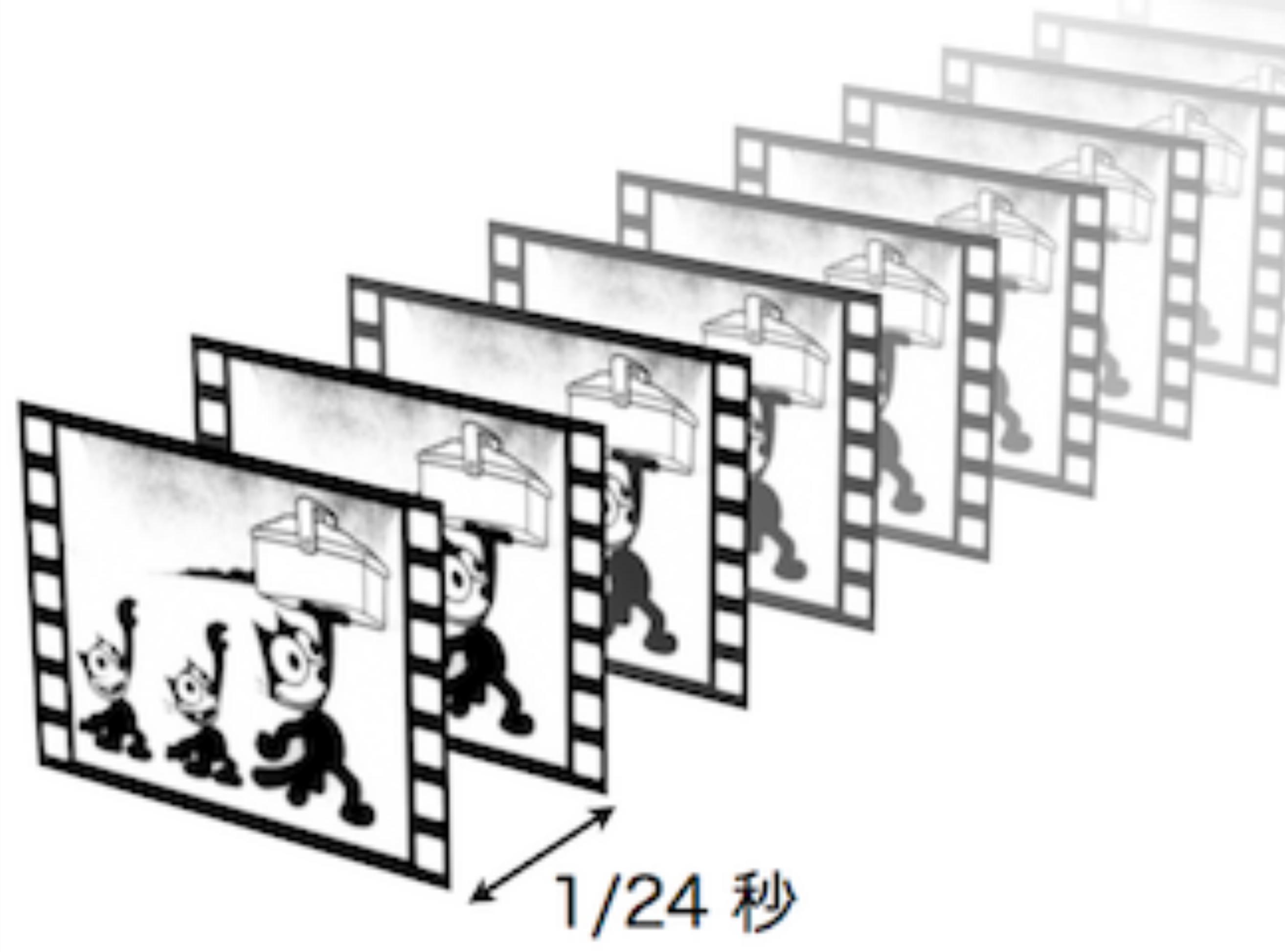
ちなみに...



NEW GAME!! にも協賛してるので

2. ゲーム開発の基礎知識

フレームとfps



▶ フレーム

1つの静止画を表示させてから、次の静止画を表示させるまでの間隔のこと。

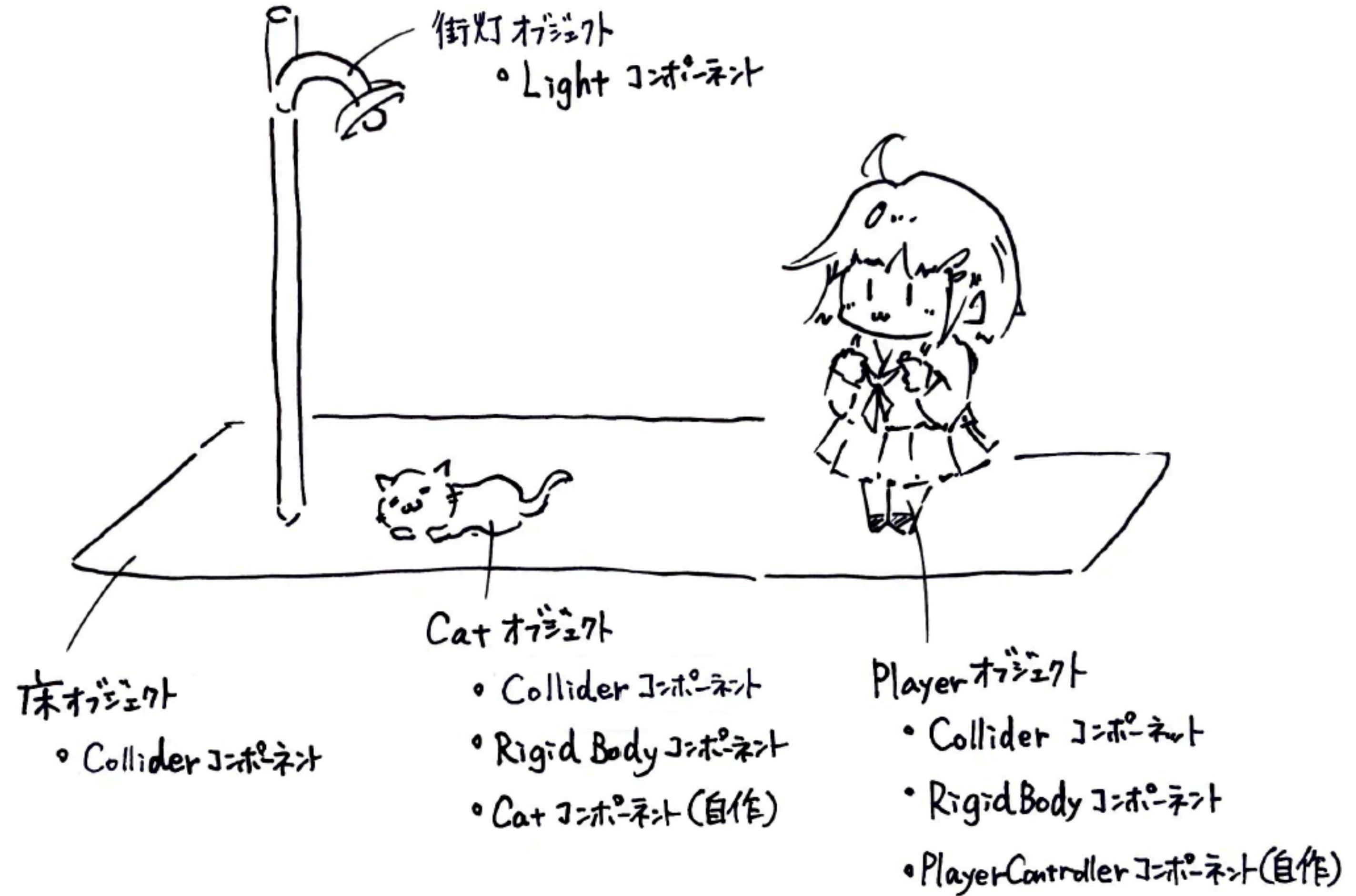
▶ fps

1秒間に何回フレームが切り替わっているかを表す単位。

※計算が間に合わず画面表示が一瞬の間途切れることを「フレーム落ち」と言う。

https://unity-yb.github.io/articles/frame_and_update.html

オブジェクトとコンポーネント



▶ オブジェクト

ゲーム画面上に存在する物体。

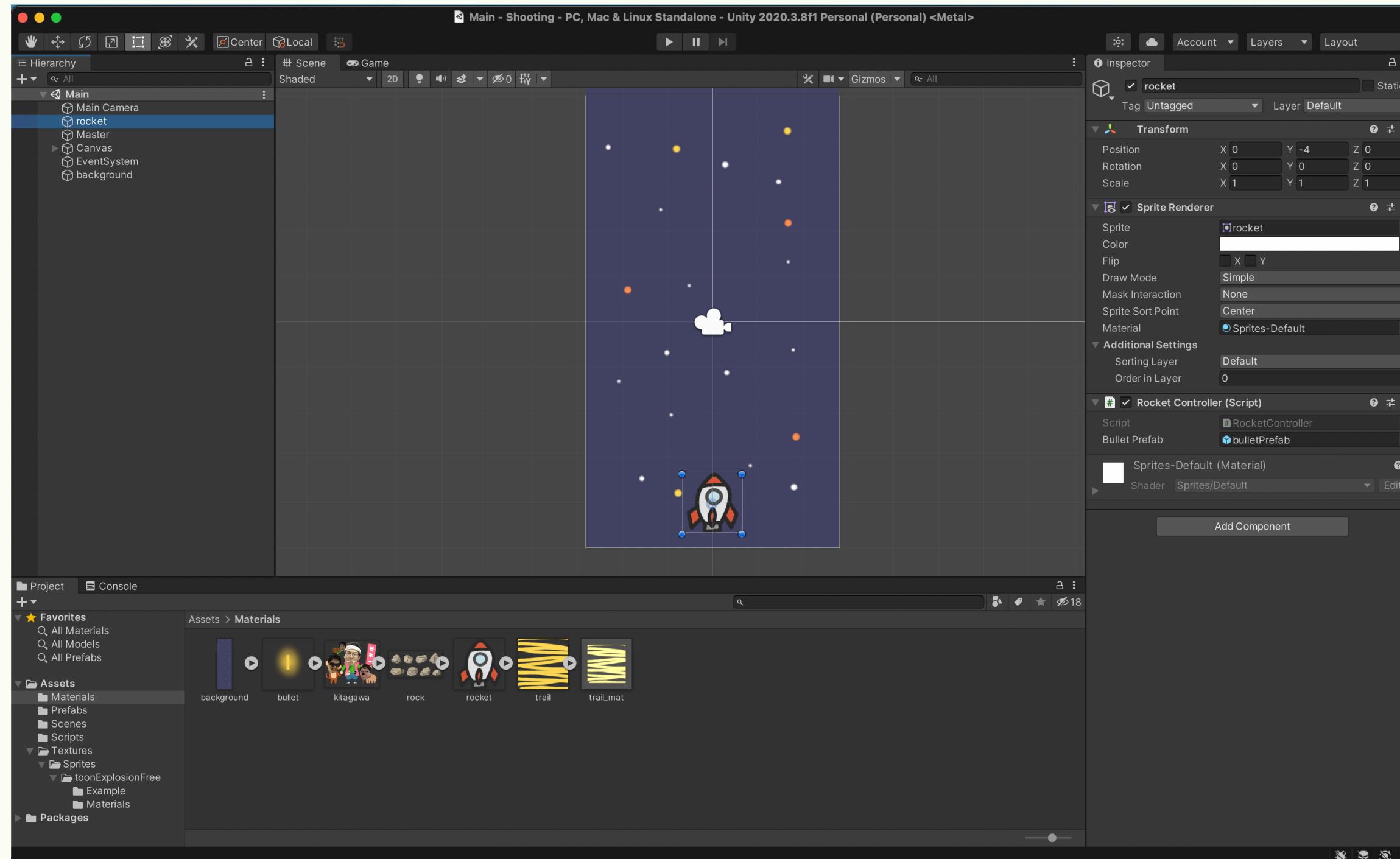
▶ コンポーネント

オブジェクトが持っている属性。
位置・色・材質など。
各オブジェクトの動作を制御する
スクリプト（ソースコード）も、
コンポーネントの一種として
オブジェクトに紐づく。

<https://qiita.com/nmxi/items/7950fb12ef925efa276d>

Unityでの開発方法

GUIでゲーム画面、コーディングでゲームの動作を実装する。



UnityのGUIでゲーム画面を実装

The screenshot shows a code editor window for the "UIController.cs" script. The code defines a class "UIController" that inherits from "MonoBehaviour". It includes methods for "GameOver", "AddScore", "Start", and "Update". The "GameOver" method sets the text of a game over UI element to "GameOver". The "AddScore" method increments a score variable by 10. The "Start" method finds score and game over text UI elements and assigns them to "scoreText" and "gameOverText". The "Update" method updates the score text every frame. The code uses Unity's UnityEngine and UnityEngine.UI namespaces.

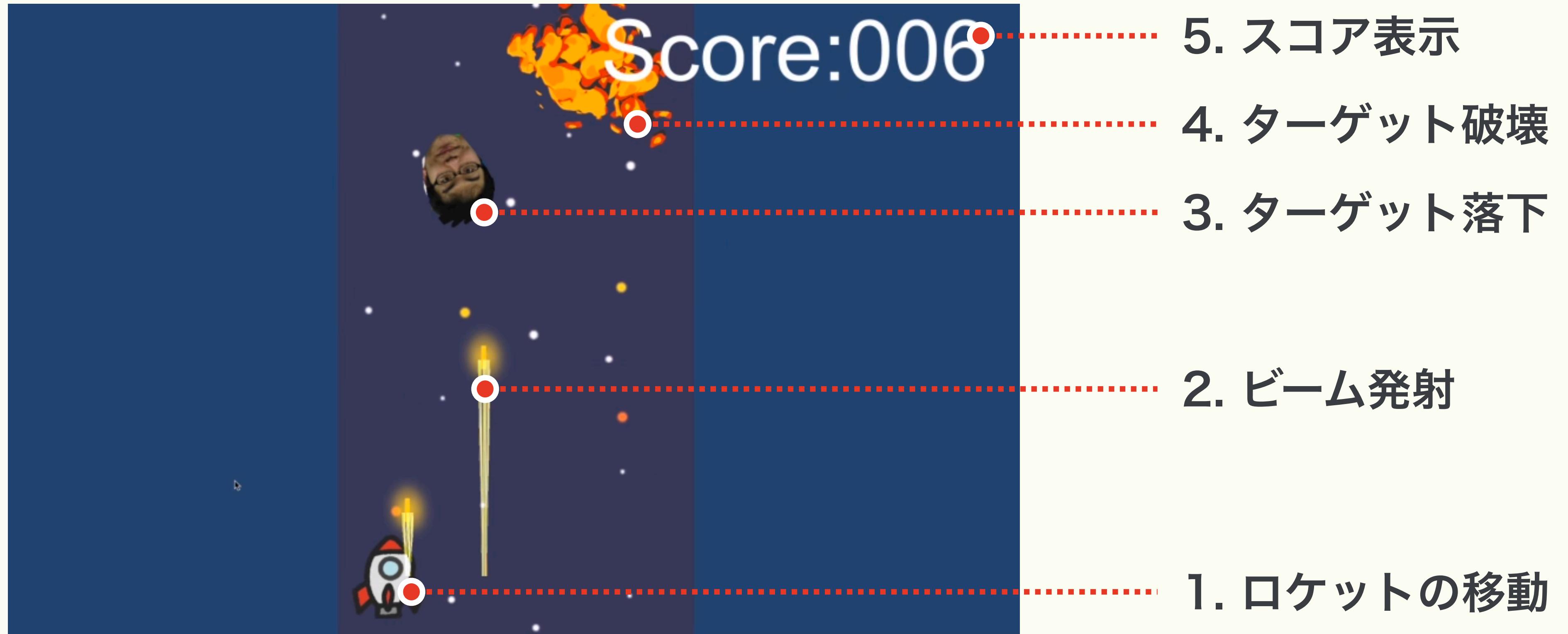
```
1  using System.Collections;
2  using UnityEngine;
3  using UnityEngine.UI;
4
5  public class UIController : MonoBehaviour
6  {
7      int score = 0;
8      GameObject scoreText;
9      GameObject gameOverText;
10
11     public void GameOver()
12     {
13         this.gameOverText.GetComponent<Text>().text = "GameOver";
14     }
15
16     public void AddScore()
17     {
18         this.score += 10;
19     }
20
21     // Start is called before the first frame update
22     void Start()
23     {
24         this.scoreText = GameObject.Find("Score");
25         this.gameOverText = GameObject.Find("GameOver");
26     }
27
28     // Update is called once per frame
29     void Update()
30     {
31         scoreText.GetComponent<Text>().text = "Score:" + score.ToString("D4");
32     }
33 }
34 }
```

VS Codeでゲームの動作を実装

3. 解説 KitagawaShooting

KitagawaShooting

ノリと勢いで作った自作ゲーム。（製作時間：約2時間）



このゲームの裏で何が行われているのか？を解説してみる。

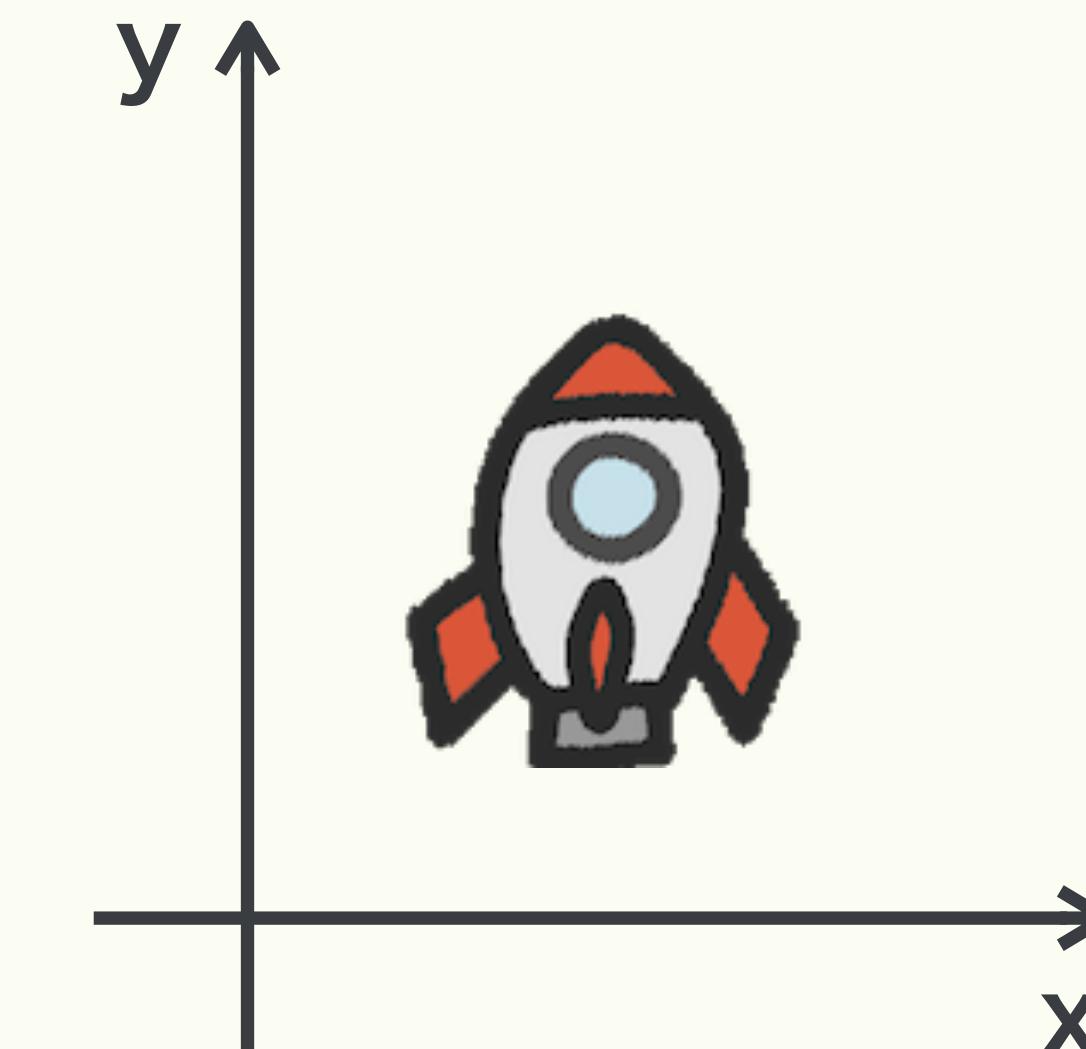
1. ロケットの移動

キー入力判定+オブジェクト位置の変更

```
18 if (Input.GetKey(KeyCode.LeftArrow))
19 {
20     if (rocketPosition.x > -2.0f)
21     {
22         transform.Translate(-0.1f, 0, 0);
23     }
24 }
25 if (Input.GetKey(KeyCode.RightArrow))
26 {
27     if (rocketPosition.x < 2.0f)
28     {
29         transform.Translate(0.1f, 0, 0);
30     }
31 }
```

左矢印が入力されたら

ロケットを左に0.1移動



2. ビーム発射

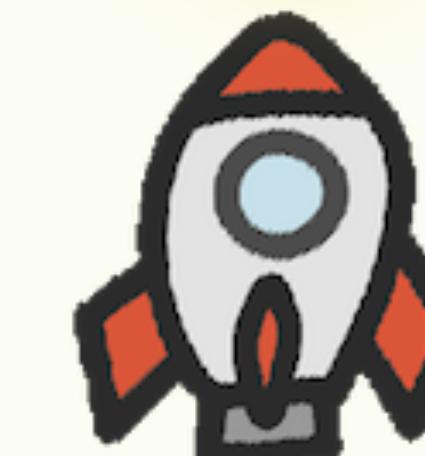
キー入力判定+オブジェクト生成+オブジェクト位置の変更

```
32     if (Input.GetKeyDown(KeyCode.Space)) {  
33         Instantiate(bulletPrefab, transform.position, Quaternion.identity);  
34     }  
35 }
```

スペースが入力されたら
ビームオブジェクトを生成

```
14 // Update is called once per frame  
15 void Update()  
16 {  
17     transform.Translate(0, 0.2f, 0);  
18  
19     if (transform.position.y > 5)  
20     {  
21         Destroy(gameObject);  
22     }  
23 }
```

毎フレーム上に0.2移動



3. ターゲット落下

オブジェクト位置の変更 + 回転運動

```
9 // Start is called before the first frame update
10 void Start()
11 {
12     this.fallSpeed = 0.01f + 0.1f * Random.value;
13     this.rotSpeed = 5f + 3f * Random.value;
14 }
15
16 // Update is called once per frame
17 void Update()
18 {
19     transform.Translate(0, -fallSpeed, 0, Space.World);
20     transform.Rotate(0, 0, rotSpeed);
21
22     if (transform.position.y < -5.5f)
23     {
24         Destroy(gameObject);
25     }
26 }
```

ターゲット生成時に初速と回転を与える

回転させながら落下させる

Translate



Rotate

4. ターゲット破壊

あたり判定 + 爆発エフェクトを生成 + オブジェクトを削除

```
25 void OnTriggerEnter2D(Collider2D coll) {  
26  
27     // Add score when Kitagawa and bullet collide  
28     GameObject.Find("Canvas").GetComponent<UIController>().AddScore();  
29  
30     // Create explosion effect  
31     Instantiate(explosionPrefab, transform.position, Quaternion.identity);  
32  
33     // Delete Objects  
34     Destroy(coll.gameObject);  
35     Destroy(gameObject);  
36 }
```

あたり判定で呼び出される

爆発エフェクトを生成

ビームとターゲットを削除



爆発エフェクトの実体

5. スコア表示

あたり判定+スコアの加算+画面表示

```
25 void OnTriggerEnter2D(Collider2D coll) {  
26     // Add score when Kitagawa and bullet collide  
27     GameObject.Find("Canvas").GetComponent<UIController>().AddScore();  
28 }  
29 
```

あたり判定で呼び出される

```
16     public void AddScore()  
17     {  
18         this.score += 10;  
19     }  
20  
21     // Start is called before the first frame update  
22     void Start()  
23     {  
24         this.scoreText = GameObject.Find("Score");  
25         this.gameOverText = GameObject.Find("GameOver");  
26     }  
27  
28     // Update is called once per frame  
29     void Update()  
30     {  
31         scoreText.GetComponent<Text>().text = "Score:" + score.ToString("D4");  
32     }  
33 
```

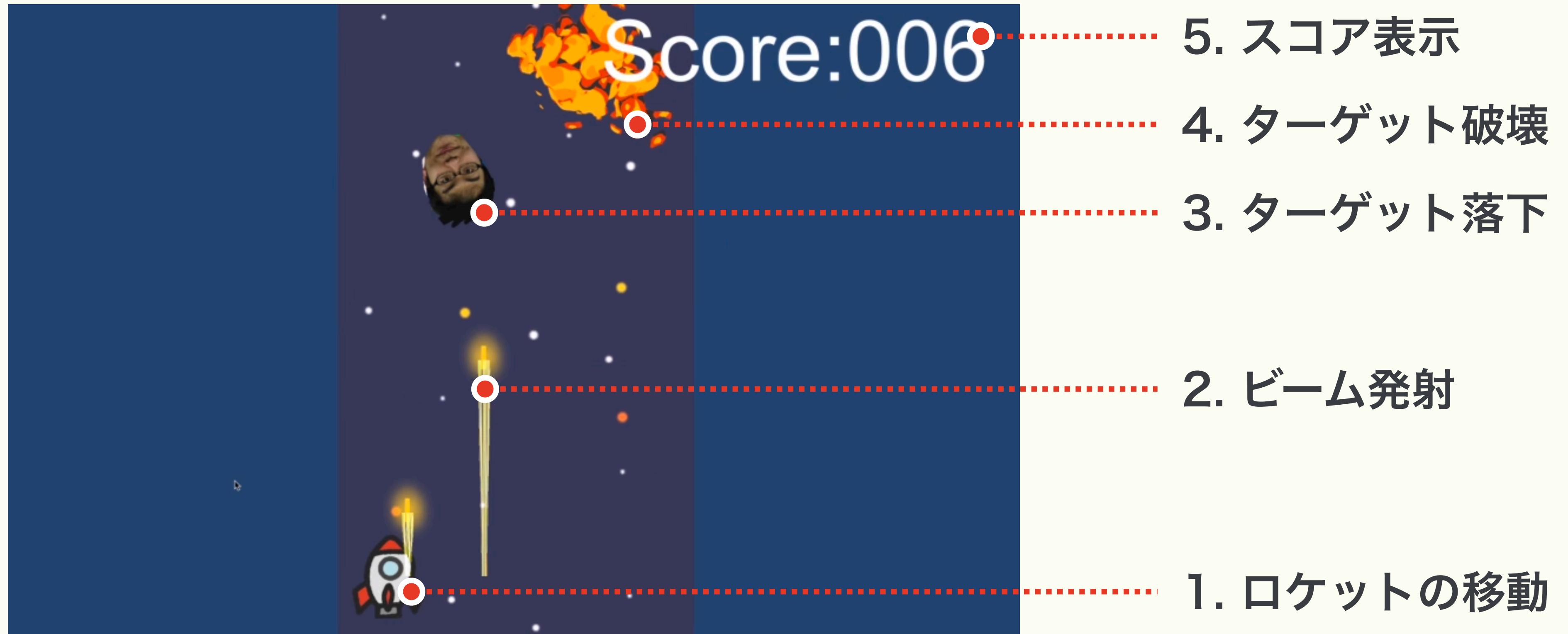
スコア加算処理を呼び出す

スコアの加算

毎フレームスコアを表示

要するに

オブジェクトのさまざまな属性を連続的に変化させて画面上に描画することで、ゲームは成り立っている。



4. まとめ

まとめ

▶ Unityとは？

Unityは、IDEを内蔵するゲームエンジンである。

▶ ゲームの開発方法

GUIでゲーム画面、コーディングでオブジェクトの動作を実装することで、ゲームを作り上げていく。

▶ ゲームの裏側

画面上のオブジェクトやそのコンポーネントを連続的に更新し描画することで、プレイヤーの操作に応じた挙動を表現している。

意外と簡単でしょ？

まとめ

Let's create NEW GAME!!
+ * + ニュー ゲーム + * +



参考

参考

▶ Unity入門

https://dotinstall.com/lessons/basic_unity_v2

▶ おもちゃラボ

<https://nn-hokuson.hatenablog.com/entry/2016/07/04/213231>

▶ 今日からはじめるUnity

<https://qiita.com/nmxi/items/7950fb12ef925efa276d>

