関数の姿になったオブジェクトたちが繰り広げる大冒険

ラムダフレンズ

LAMBDA FRIENDS

GIO(ゴトー)さんから「C#におけるデリゲートとかラムダ式の話をして欲しい」という要望があったので、今日はその手の話をしよう。

でりげーとちほー

おどりゃ、きさまはそもそもdelegateというものを分かっちょるんか？

delegateっちゅうヤツはな、一言で言うと**「関数オブジェクトを格納できる変数」の型**なんじゃ！！

C++で言うところの「関数ポインタ」もしくは「関数オブジェクト」にそっくりなんじゃ！！

この手の仕組みは「コールバック」「遅延実行」「Stateパターン」としてよく使われるんじゃ！！

別にこの機能を使わなくても実装自体はできるし、簡単な話でもない。だから理解できないと思ったら理解しなくても良い。無理はするな。

型の定義の文法を説明しよう。

例えば、

```
void Saval(){
        Debug.Log("サーバル");
}
```

という具合だ。

であれば、これに対応するデリゲートは

```
delegate void Functor();
```

となる。

一般化して書くと

delegate 戻り値 関数オブジェクト型名(引数);

となる。ここまではいいかな？

では、例えば、引数なし戻り値なしの関数をつくるとする。
あくまでもこの Functor は「型名」であることに注意して欲しい。

例えば、

```
delegate void Functor();
Functor _functor;
void Saval(){
    Debug.Log ("すごーい");
}
```

のように書いておくとすると、_functor はまだ空っぽであるので、Start あたりで入れておく。

```
void Start () {
    _functor = Saval;
}
```

関数 A をデリゲートに突っ込んでいる。ただしまだ実行はされていない。
これが実際に実行されるのは_functor を実際に関数のようにコールしてからです。つまりクリックしたときに実行されるとするならば

```
void Update () {
    if (Input.GetMouseButtonDown (0)) {
        _functor ();
    }
}
```

こんな感じです。ね？簡単でしょう？

さて、このままでは面白くない。例えば

```
void Saval(){
    Debug.Log ("やったー");
}
void Lion(){
```

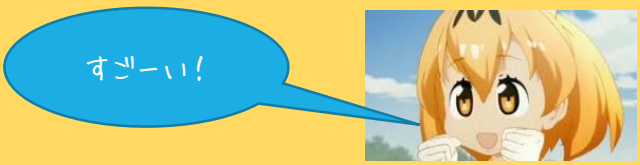
```
        Debug.Log ("すごーい");
    }
    void Ocelot(){
        Debug.Log ("たっのしー");
    }
```

というように関数を3つくらい作っておいて、これがクリックされるたびに
やったー→すごーい→たっのしー→やったー→すごーい→たっのしー
と実行されるようにしたい。どうしたら良いんだろうか？よく思いつくのが、現在の状況を表す変数を作っておき、関数が呼ばれる時に切り替えて switch～case で制御するという方法だ。
だが、この delegate を使用すれば switch～case も変数も必要ない。

それぞれの関数の最後で、次の関数へ切り替えれば終わりだからだ。

```
     void Saval(){
        Debug.Log ("やったー");
        _functor=Lion;
    }
    void Lion(){
        Debug.Log ("すごーい");
        _functor=Ocelot;
    }
    void Ocelot(){
        Debug.Log ("たっのしー");
        _functor=Saval;
    }
```

ね？簡単でしょう？



ちなみに、

```
    void Konoha(){
        Debug.Log ("我々は賢いのだ。");
    }
```

こんな関数を作っておいて

```
    void Start () {
        _functor = Saval;
        _functor+=Konoha;
    }
```
などとやれば、最初には関数 Saval と関数 konoha が実行され、その後は「お察しください」となります。これを関数の連結と言います。

これが delegate の基本です。

らむだしきちほー

ラムダ式は、多分、今まで見たこともない文法かもしれません。関数を作らなくても、関数オブジェクトをその場で作る方法です。なんか=>という演算子を使って作ります。作り方自体はいたって簡単。

(引数)=>{実行したい処理}

これ自体がもう関数オブジェクトです。関数オブジェクトなので、先程のデリゲートに突っ込むことができます。

```
    void Start () {
        _functor = () => {
            Debug.Log ("リア充消滅しろ");
            _functor = A;

        };
    }
```
ちなみに最後のセミコロン忘れやすいから注意ねー。

こうすると1回目のクリックでは「リア充消滅しろ」と表示され、後は先ほどと同じクリかえしとなります。

そんなわけないだろ、もうちょっと頑張るんだ