

# iアプリ Java ゲームプログラミング

## 第5回 メインループ

### メインループ

ゲームプログラムは、「ものの状態の集まり」と考えることができます。キャラクターの座標やパラメータ、背景、画面エフェクトなどすべて、それぞれ「もの」の状態です。ゲームプログラムとは、このいろいろな「もの」の状態を時間に沿って更新し、それを画面に描画するプログラムと考えることができます。

つまり、ゲームプログラムの処理を非常に単純化すると、

- ・「もの」の状態を更新(内部処理)
- ・「もの」を画面に描画(描画処理)

という2つにまとめることができます。

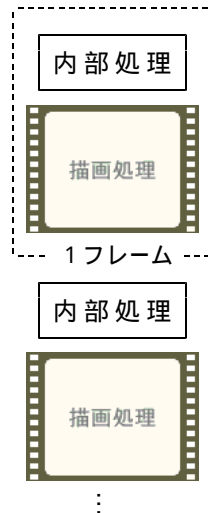
この2つの処理の繰り返しがメインループになります。このループ1回ぶんを「1フレーム」と呼びます。1フレームごとに、キャラクターの位置や状態、背景やそのほかの表示状態などを更新し、その状態を反映して描画を行う、という処理を繰り返していきます。

### フレームレート

ゲームプログラムでは、メインループ(正確には画面の更新)を一定の間隔で実行します。実行速度は環境によって異なるので、速い環境ほど高速に動作できます。

1秒間に何回画面を更新するのかをフレームレート(FPS:Frame Per Second)と呼びます。60FPSの場合、1秒間に60回画面を更新しているという意味になり、1フレームを約16.67ミリ秒の間隔で表示しています。

一般的な家庭用ゲームでは60FPSです。携帯電話では、液晶の反応速度の兼ね合いもあり、30FPS程度にしておきます。この数値は、表示する画像の大きさや数が増えるほど、または高性能な描画機能を使用すると目に見えて落ちていきます。



## 課題

メインループを作成しましょう。また、メインループが一定の間隔で実行されるようにしましょう。

(1)キャンバスクラスにメインループを作成します。

メインループは、キャンバスクラスのrunメソッド(ゲームを実行するためのメソッド)に記述します。

メインループに入る前に、initメソッドを呼び出して変数などゲームの状態を初期化しておきます。メインループでは、内部処理と描画処理を交互に行い、ゲームを進めていきます。

「？」を埋め、runメソッドを以下のように変更しましょう。

```
// --- 実行
void run() {
    init(); // 初期化

    // メインループ
    while(true) {
        ????: // 内部処理
        ????: // 描画
    }
}
```

(3) フレームレートを制御する機能を追加します。

フレームレートを保つには、メインループが一定の間隔で回るようにします。メインループの処理時間は、その都度変わるので、メインループ1回ごとに処理時間を計り、それが1フレームの間隔未満の場合は、時間に余裕があるので、その分だけプログラムを休止させます。

休止時間は、以下の式で計算するものとします。

休止時間 = 1フレームの間隔 - 1フレームを処理するのにかかった時間  
( 休止時間が負の場合は、処理落ちになります)

1フレームの間隔 = 1000 / FPS ( 1秒 = 1000ミリ秒)

1フレームを処理するのにかかった時間 = フレーム終了時間 - 前回のフレーム終了時間

キャンバスクラスに、フレームレートを制御するための定数を定義します。定数FPSおよびINTERVALを「public final class MyCanvas extends Canvas {」の直下に追加しましょう。

```
public final class MyCanvas extends Canvas {
    // --- 定数の定義(ゲームに必要な定数を以下に定義します)
    public final static int FPS = 30; // フレームレート
    public final static int INTERVAL = 1000 / FPS; // フレームの間隔(ミリ秒)

    // --- 変数の定義
}
```

定数FPSが1秒あたりのフレーム数、定数INTERVALが現フレームから次フレームまでの間隔です。

(4) フレームレートを制御するための変数を追加します。

(3)の方法でフレームレートを制御するには、「前回のフレーム終了時間」を変数に保存しておく必要があります。long型の変数「lastTime」を「// --- 変数の定義」の下に追加しましょう。

```
// --- 定数の定義
public final static int FPS = 30; // フレームレート
public final static int INTERVAL = 1000 / FPS; // フレームの間隔

// --- 変数の定義
long lastTime = 0; // 前回のフレーム終了時間

// --- 実行
void run() {
```

(5) 休止時間を求め、プログラムを一時休止させるsleepメソッドを作成します。以下のプログラムをキャンバスクラスに追加しましょう。

```
// --- 描画
void draw() {
}

// --- 休止
void sleep() {
    long wait = INTERVAL - (System.currentTimeMillis() - lastTime); // 1
    if(wait > 0) {
        try {
            Thread.sleep(wait); // 2
        } catch(Exception e) {
        }
    }
    lastTime = System.currentTimeMillis(); // 3
}
```

1は、(3)の方法で休止時間を求める部分です。休止時間がある場合、2のThread.sleepメソッドでプログラムを休止させます。3は、実質的にメインループの最後に行われる処理で、フレームの終了時間を保存する処理です。System.currentTimeMillisメソッドを呼び出し、現在の時間を取得しています。

(6) メインループの最後で休止するようにします。メインループを以下のように変更しましょう。

```
// --- 実行
void run() {
    init(); // 初期化

    // メインループ
    while(true) {
        ???(); // 内部処理
        ???(); // 描画
        sleep(); // 休止
    }
}
```