

Unit 5: TI モジュール

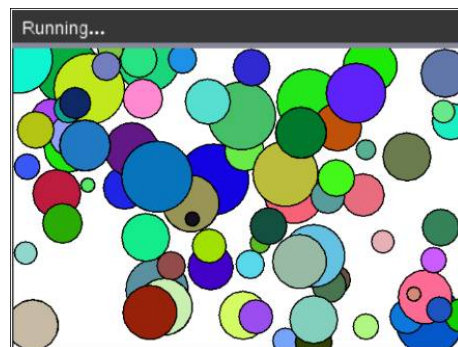
Skill Builder 3: どこでも円！

このレッスンでは、含まれているいくつかのモジュールを使って、スクリーンセーバースタイルのアニメーションを作成します。

目標

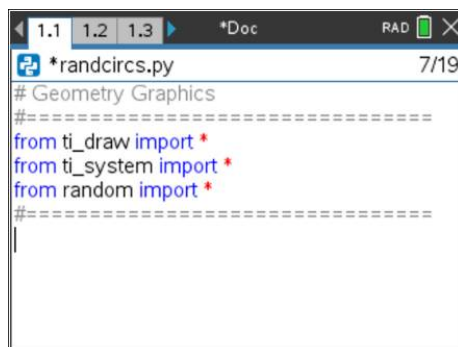
- 複数のインポートを使用
- 乱数の利用
- キーを押してプログラム停止
- 塗りつぶされた円を描く

このプロジェクトでは、キーが押されるまで画面上にランダムな円を描きます。これは一種の「スクリーンセーバー」(固定画像によるCRT画面の「焼き付き」を防ぐため、昔使用されていたプログラム的一种)です。



1. Type : Geometry Graphicsを指定して、新規のPythonファイルを開始します。

このテンプレートは、**ti_draw**モジュールを提供します。このプロジェクトには、さらに2つのモジュールが必要になります。1つは乱数用で、もう1つは「escを押して終了」機能用です。乱数関数は**random**モジュールにあり、**get_key**は**ti_system**モジュールにあります。両方のモジュールをインポートします。それらはさまざまなメニューにあります。



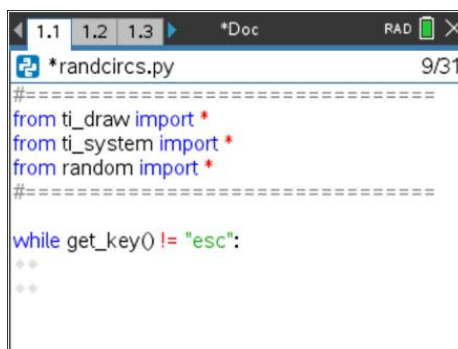
2. このプロジェクトでは、**set_window**を使わないでください。(0,0)が左上隅にあるデフォルトグラフィックウィンドウを使います。画面は318×212ピクセルです。右の画像には、四隅の座標とキャンバスの中心が示されています。

この画面と**set_window**を使って設計された画面の重要な違いは、次のとおりです。

ステートメント**draw_rect(x, y, width, height)**では、(x,y)が左上隅になり、y値が画面の下に向かって増加するため、高さの値は上から下に測定されます。



3. メインプログラムは、**esc**が押されると停止するループで構成されます。この**while**ステートメントは**menu > More Modules > TI System** (メニュー>その他のモジュール>TIシステム)にあります。

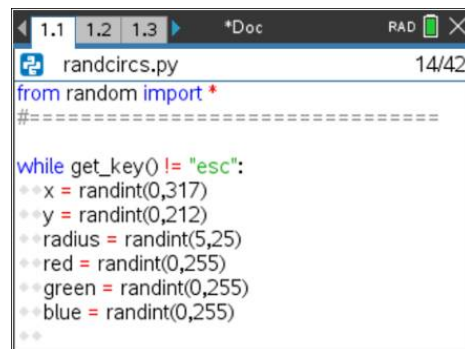


4. ランダムな色でランダムな円を描くには、ランダム値でいくつかの変数を設定する必要があります。円の場合、x, y, 半径が必要であり、色の場合は赤、緑、青の値が必要です。

これらの6つの変数はそれぞれ、次のように割り当てられます。

x = randint(0,317)

適切な範囲のランダムな整数を各変数に割り当てます。画面の寸法と色の値の制限に注意してください。



```

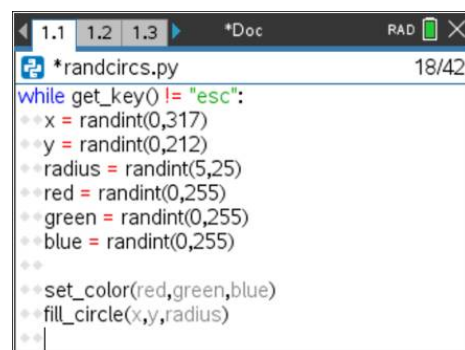
1.1 1.2 1.3 *Doc RAD 14/42
randcircs.py
from random import *
#=====
while get_key() != "esc":
    x = randint(0,317)
    y = randint(0,212)
    radius = randint(5,25)
    red = randint(0,255)
    green = randint(0,255)
    blue = randint(0,255)

```

5. 以下からset_color()関数とfill_circle()関数を追加します。

menu > More Modules > TI Draw
(メニュー>その他のモジュール>TI Draw)

引数の灰色のインラインプロンプト(行内入力要請)が変数名と同じ単語であることがわかりますか。残念ながら、そのままにしておくことはできません。インラインプロンプトの赤は変数名の赤ではありません。インラインプロンプトの代わりに変数名を入力する必要があります。



```

1.1 1.2 1.3 *Doc RAD 18/42
*randcircs.py
while get_key() != "esc":
    x = randint(0,317)
    y = randint(0,212)
    radius = randint(5,25)
    red = randint(0,255)
    green = randint(0,255)
    blue = randint(0,255)
    set_color(red,green,blue)
    fill_circle(x,y,radius)

```

これらの2つのステートメントのプロンプト(入力要請)を変数名に置き換えます。

準備ができたならプログラムを実行し、そしてescを押してプログラムを停止します。

乱数範囲を自由に変更してください。円をもっと赤みを帯びて見せたいとしましょう。どうしますか。

6. また、描画速度を制御する場合に備えて、timeモジュールでsleep()関数を再度、使います。timeモジュールからsleepをインポートします。

from time import sleep

ループの最後にsleep(1)を追加します(fill_circle関数の後、まだインデント(字下げ)されています)。

このステートメントは、処理を1秒間停止します。遅すぎますか。sleep()で小さい数値または大きい数値を使うことで、処理を速くしたり遅くしたりできます。



```

1.1 1.2 1.3 *Doc RAD 17/41
*randcircs.py
while get_key() != "esc":
    x = randint(0,317)
    y = randint(0,212)
    radius = randint(5,25)
    red = randint(0,255)
    green = randint(0,255)
    blue = randint(0,255)
    set_color(red,green,blue)
    fill_circle(x,y,radius)
    sleep(1)

```

7. このレッスンの冒頭の画像を見ると、各円の縁が黒いことが分かります。それを実現できますか。

