遊ぶ人から創る人へ

Myゲーム作り

第1回　ビリヤード<前編>文字や図形の描画

佐々木 弘隆

今回から3回にわたってPC上で動くビリヤード・ゲームをPythonで作ります．Pythonは汎用的な言語ですが，ゲームに必要な処理を作るのは大変です．しかし，Pythonはいろいろな機能を取り込めるメリットがあることから，ゲームを作ることも可能です．今回は開発環境の準備やプログラムの実行，文字や図形を描画するプログラムなどを作ってみます．

環境準備

● Python3のインストール

最初にPC上で動くゲームを開発するための環境を整えます．まずは，PCにPython3をインストールします．Python公式サイト（https://www.python.org/）から使用するPCのOSに合ったインストーラをダウンロードします．

● pygameのインストール

Pythonをインストールしたら，コマンドプロンプトを立ち上げ，ゲーム開発用のライブラリpygameを以下のコマンドでインストールします．

```
python -m pip install pygame
```

インストールが正常に行われたかどうか確認します．

```
python
>>>import pygame
```

エラー・メッセージが出なければインストール完了です．Pythonが対話モードで起動したままなので終了させます．exit()と入力するか「Ctrl+Z」キーを押して，最後に「Enter」キーを押して，Pythonに対話の終了を知らせます．

なお，ここではWindowsを想定していますが，MacOSやUNIX系のOSでも環境は構築できます．最初からPythonが動く環境の場合も多いです．Windowsと違う部分は公式サイトの記述を参考に読み替えてください．

プログラムは本誌サポート・ページから入手できます．
https://interface.cqpub.co.jp/2209py/

リスト1　ゲーム・プログラムの基本構造

```
 1 import pygame  #pygameをインポート
 2 from pygame.locals import*  #pygameの使用を簡単にする
 3 import sys     #sysをインポート
 4
 5 pygame.init()  #pygameの初期化
 6
 7 #スクリーンサイズを設定
 7 screen=pygame.display.set_mode((640, 480))
 8 #タイトルの設定
 8 pygame.display.set_caption("ウィンドウだけ")
 9
10 while(1):
11     pygame.display.update()          #スクリーンの更新
12
13     for event in pygame.event.get(): #イベントの設定
14         if event.type == QUIT:
                              #もし終了ボタンをおされたら
15             pygame.quit()    #終了する
16             sys.exit(0)      #正常終了
```

ゲーム・プログラムの実行とその構造

● 実行手順

▶ステップ1…プログラムを記述して保存

テキスト・エディタで**リスト1**のプログラムを打ち込み，pgtest.pyとして保存します．名前は自由に決めて構いませんが，ここではpgtest.pyとした前提で説明します．テキスト・エディタですが，筆者はVisual Studio Codeをお勧めします．使ってみてください．

▶ステップ2…ディレクトリの移動

保存したらコマンドプロンプトを起動して，Pythonプログラム（以下スクリプト）を保存したディレクトリに移動します．移動するためのコマンドとしてCDコマンドやCHDIRコマンドがあります．

▶ステップ3…コマンドの実行

スクリプトが保存されているのを確認したら以下のコマンドを打ち込んで実行します．

```
python pgtest.py
```

▶ステップ4…ウィンドウの表示確認

実行するとコマンド・プロンプトとは別のウィンドウが作られます（**図1**）．ウィンドウ上部には「ウィン

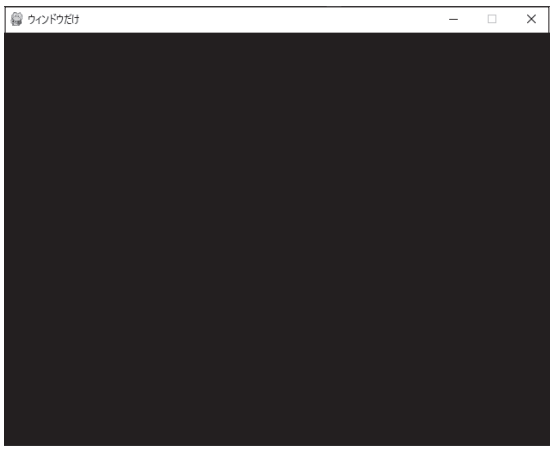

図1　プログラムを実行するとウィンドウが表示される

図2　文字列は複数の関数を順番に呼び出せば表示できる

リスト2　CUIならprint関数だけで簡単に文字列を表示できる

```
print("文字列") # print関数に指示した"文字列"がコマンドライン上に表示される
```

リスト3　フォントや大きさを指定して文字列を表示する必要がある

```
fontType = pygame.font.SysFont("hg正楷書体pro", 20)      # 使うフォントと文字サイズを指定(1回だけでよい)
rText = fontType.render("文字列", True, (255,255,255))   # "文字列"を白色(R=255:G=255:B=255)でレンダリング
screen.blit(rText, (x, y))                               # レンダリング結果をx, y座標に張り付け
```

ドウだけ」と書かれていて，右上の「X」を押すと終了するだけの真っ黒で地味ですがWindows用のアプリケーションができました．これを元に開発することにしましょう．

● プログラムの基本構造

リスト1を記述しましたが，内容は今のところ意味不明だと思います．**リスト1**の内容は，pygameを使う場合の基本的な構造の1つで，暗記する必要はありませんがここで説明します．

▶1～3行目

importはPythonが持っていない機能を取り込む処理です．pygameはゲーム関連の機能で，sysはPythonを実行しているOSの機能です．アプリケーションの終了にはsysが必要になります．

▶5行目

pygame.initを最初に実行することでpygameを使えるようにします．その後は定期的にpygame.display.updateを実行する必要があります．

▶7～8行目

ループを開始する前にウィンドウの設定をする必要があります．pygame.display.set_modeはウィンドウの横幅と縦幅を指定して，pygame.display.set_captionはウィンドウの上に表示される文字列(「キャプション」と呼ぶ)を指定します．

▶10行目

while(1)は永遠に繰り返すループという意味です．

▶13行目

forループでWindowsからのイベントを取り出して個別処理します．イベントと言われても分かりづらいかもしれません．Windowsはイベント・ドリブンと呼ばれる処理方式になっていて，例えばマウスを動かしたとかキーボードを押したなど，さまざまな情報がイベントとしてアプリケーションなどに通知される仕組みになっています．ここでは，イベントの内容がQUIT，つまりアプリケーションを終了してほしいというイベントが通知されたことを判断しています．

▶15～16行目

アプリケーションを終了する必要があるのでpygameを終了して，Windowsにこのアプリケーションを終了するリクエストを出します．コマンドライン・アプリケーションと違い，この辺りが独特で分かりづらいとは思いますが，お約束の処理と考えれば大丈夫です．

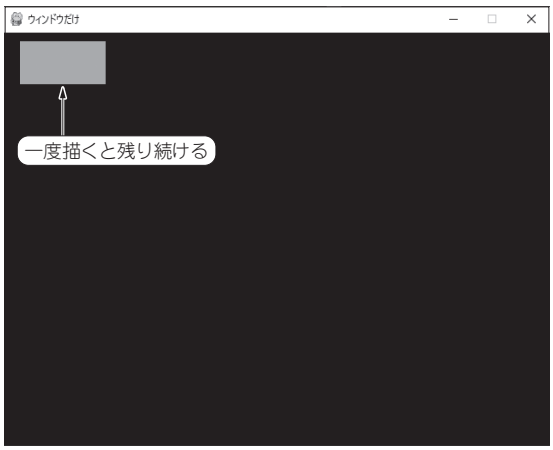

図3　screec.fill関数は四角形の範囲内を指定色で塗りつぶす

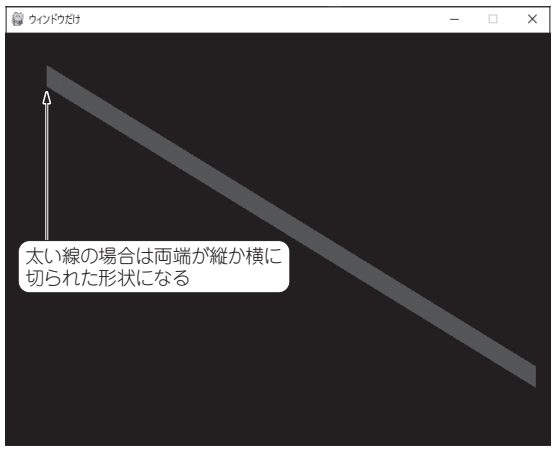

図4　pygame.draw.line関数は線を描画する

リスト4　fill関数の使用例

```
screen.fill((0, 0, 0))  # 画面全体を黒（赤=0,緑=0,青=0）で塗りつぶす
screen.fill((20, 210, 70), (20, 10, 100, 50))  #左上基準から（20,10）座標から幅100高さ50の四角を緑で塗りつぶす
```

リスト5　line関数の使用例

```
pygame.draw.line(screen, (0,95,0), (startX, startY), (endX, endY), 25)  # 色(R,G,B) 開始座標(X,Y)終了座標(X,Y)線幅を指定
```

文字や図形を描画するプログラム

● 文字の表示

PythonでCUIプログラミングする場合，**リスト2**のようなprint関数を使えば簡単に文字を表示できますが，WindowsなどのGUIアプリケーションの場合はグラフィックスとして描画する必要があります．

これは，**リスト3**のように描画する文字のフォントと大きさを決めて，表示したい文字列をレンダリングと呼ぶ工程でグラフィックスとして作成し，最後にウィンドウの特定の場所に張り付ける流れです．**リスト3**の実行結果を**図2**に示します．

これを聞くと何だか難しそうですが，それぞれの難しい作業をする関数が用意されているので順番に呼び出せば大丈夫です．

● 基本図形の描画

せっかくのGUI（Graphical User Interface）向けアプリケーションですので，きれいなグラフィックスを表示したいですね．ウィンドウ内に小さな色の点をチマチマ書いていくこともできますが非常に面倒です．そこで，幾つかの基本的な図形を描画する関数が用意されていますので以下で説明します．

▶ screec.fill関数

ウィンドウ画面全体，またはウィンドウ内の一定の四角形の範囲内を指定色で塗りつぶすことができます（**リスト4**）．実行結果を**図3**に示します．

一度画面に書いた画像は残り続けますので，ゲームの進行に応じてキャラクタを動かす前に全体を塗りつぶす必要があります．

▶ pygame.draw.line関数

線を描画する機能を持っています（**リスト5**）．実行結果を**図4**に示します．この関数は太い線を斜めに描画すると線の両端が縦か横に切られた形状になってしまうのが難点です．使用するには工夫が必要ですね．

他にも多数の機能がありますので，複雑な処理を使いたい場合は公式のドキュメントを確認してください．

https://www.pygame.org/docs/

画像ファイルの描画

基本図形の組み合わせでも画面を彩ることはできますが，美しい背景やすてきなキャラクタを表現するのは大変です．そこで，グラフィックス・ソフトウェアなどで作成した外部の画像ファイルを使うことができます．

表1　pygameで利用可能な画像ファイル・フォーマット

拡張子	特　徴
JPG (JPEG)	ディジタル・カメラの標準的なフォーマットで自然画像が得意．非可逆圧縮なので完全に元の画像に戻すことはできない
PNG	ウェブ系で主に使われるフォーマットで可逆圧縮なので画像を劣化させることなく使える．手書きイラストやボタンなどの素材に向いている．透過可能
GIF	ウェブ系で主に使われていたフォーマットで最大256色までの可逆圧縮．(pygameで使う場合は) PNGの下位フォーマットと言える．透過可能
BMP	Windowsで標準的なフォーマットで基本的に無圧縮なので画像の劣化はないがファイル・サイズが大きい
PCX	Windows以前のフォーマットで可逆圧縮で複数のカラー・タイプに対応しているが古いPCX素材が利用できる以外の利点は特にない
TGA	一部のグラフィックス業界で使われているフォーマットでpygameでは無圧縮形式のみ対応している他，対応環境の面から多くの人にとってはお勧めできない
TIF	BMPのような無圧縮のビット・マップ・フォーマットで特定のOS向けではない．多くのOS，多くのアプリケーションで使われている
LBM	Amigaというコンピュータで使われていたフォーマットで古いLBM素材を再利用する以外では特に利点はない
PBM	テキスト形式のビット・マップ・フォーマットでテキスト・エディタで開くことができて画像の形が分かるのが特徴．しかし，ファイル・サイズはとても大きい．モノクロ2色
PGM	グレー・スケール版のPBMフォーマット
PPM	フル・カラー版のPBMフォーマット
XPM	X Window Systemで使われたテキスト形式のビット・マップ・フォーマット

図5　ボールのデザインにはIllustratorを使う

図6　さらにPhotoshopでサイズ調整とPNG出力をする

● 画像ファイルの使い分け

表1にpygameで利用可能な画像ファイル・フォーマットを示します．pygameは多くの画像ファイル・フォーマットに対応していますが，特に理由がなければJPGとPNGだけでよいと思います．

JPG (JPEG) は非可逆圧縮ですのでディジタル・カメラで撮影した風景写真をゲームの背景画像として使う場合などが向いています．一方，PNGは可逆圧縮で画像が劣化しないのと，画像に透明な箇所 (透過) を含めることができるのでUIボタンやキャラクタなどに使います．

以上のように，画像フォーマットの特徴を理解した上で自分が作りたいアプリケーションの設計を考えていきます．上記以外の組み合わせがよい場合もあると思いますが，分からないうちは上記の使い分けだと思ってください．

● ステップ1…画像ファイルの準備と権利上の注意点

画像ファイルの入手には，大きく分けて次の3つの方法が考えられます．

▶1，ディジタル・カメラで撮影

自分でディジタル・カメラやスマートフォンのカメラを使って撮影していく方法です．分かりやすいですね．

自分で撮影した写真データの権利は自分にありますので基本的には自由に使えますが，他者の権利を侵害しないように気を付ける必要があります．写り込んでしまった他人の顔には肖像権が，看板のロゴや特定商品には商標権や意匠権などがありますので，許可なく使うと法的な問題に発展する可能性があるので注意しましょう．

▶2，画像をもらう

個人や企業などが撮影/作成した写真や絵を使わせてもらう方法です．もちろんこの場合にも気を付けることはあります．

有料販売している画像の場合，料金を支払う必要があります．他にも，

- 画像の加工ができるか
- 画像をアプリケーションに組み込めるか
- 利用テーマに制限があるか
- 権利者の表記が必要か

など，さまざまな条件が有料/無料共に明示されてい

リスト6　任意の画像を表示するプログラム

```
sBall9 = pygame.image.load("grp/bb09n.png").convert_alpha()  #9ボール画像ファイル(bb09n,png)を透過ありで読み込む
screen.blit(sBall9, (100, 50))  # 9ボールをX=100,Y=50に描画
```

る場合が多いです．分からない場合は画像の権利者に確認するようにしましょう．

▶3，画像を作る

一番手間は掛かりますが一番安全な方法です．他者の著作物を描かない限りは全ての権利は自分のモノです．

基本的に画像はラスタ・グラフィックス・アプリケーションを使って作るのですが，PNGかJPGの出力に対応しているアプリケーションでしたら何を使っても構いません．また，Windowsに標準搭載されているペイントやAdobeのPhotoshopなど数多くの有料/無料のアプリケーションが存在しますのでお好みの方法を探ってみましょう．

今回はIllustratorでボールのデザイン(**図5**)をしてPhotoshopでサイズ調整とPNG出力をしてみました(**図6**)．この場合の著作権者は筆者になりますが，本稿を読んでいる方に向けて加工，アプリケーション組み込みを含みあらゆる利用を許諾しますのでご自由に使ってください．

● ステップ2…画像描画

画像ファイルが用意できたので画面に描画してみます．プログラムを**リスト6**に，実行結果を**図7**に示します．

プログラムは，pygame.image.load関数を使って画像ファイルから画像データとしてアプリケーションに読み込み，screen.blit関数で指定した座標に画像データを描画します．たったこれだけで複雑な画像も描画できるようになります．

グラフィックスの消去と描画を繰り返す「フレーム処理」を作る

● 工程は3つある

ゲームに限らずディスプレイを使用するアプリケーションやハードウェアは基本的に，

- 1，グラフィックスを消去する
- 2，新しいグラフィックスを描画する
- 3，ディスプレイに転送する

の流れで動いています．

図7　数行のプログラムで画像表示することができる

1つのフレームは静止画ですので，動いているように見せるためには一定時間の間に複数のフレームを切り替える必要があります．これらの処理を**リスト7**に示します．なお，1秒間にフレームが何回切り替わるかを「fps(Frame per second)」と呼びます．

● 映像媒体のフレーム数

▶映画とテレビ

表2より映画が最も低い24fpsですから，これ以上のfps値であれば動画と認識できるでしょう．また，テレビに出力する場合は60fpsとした場合，都合が良さそうです．

▶PC

PCの場合は特殊で，決まったfpsはありません．多

リスト7　フレームの処理内容

```
clock = pygame.time.Clock()
clock.tick()      # ループの最初は「前フレーム」が存在しないのでclockを更新
timeCounter = 0   # 経過時間をフレーム時間の累積で格納する
# ゲームのループ
while True:
    # フレームの更新時間を測定
    deltaTime = clock.get_time() / 1000.0          # 前フレームの更新からの経過時間を抽出する(実時間に合わせた物理演算のため)
    clock.tick()     # clockの更新
    timeCounter += deltaTime                       # フレームの経過時間を加算する
    print("前フレームの処理時間" + str(deltaTime))     # 前フレームの処理時間を表示
    print("経過時間" + str(timeCounter))              # アプリ起動からの経過時間を表示
    clock.tick(60)            # 1秒間のフレーム更新最大数を指定(非リアルタイムOS上では基本的に正確なfpsにはならない)
```

コラム　プログラム言語の選定要素

佐々木 弘隆

● 機能/速度/汎用性をもとに選ぶ

コンピュータが産まれてから，たくさんのプログラム言語が開発されてきましたが，最終的にはどの言語で開発してもコンピュータが実行することには変わりません．

プログラム言語の違いを挙げるなら**表A**のような感じです．これは，負担を掛けるのが人間側か，コンピュータ側かで反対になっており，それがそのまま機能と速度の対比にもつながる傾向にあります．それ以外では汎用性が挙げられます．

▶用途が多岐にわたる場合は汎用性が高い言語を選ぶ

汎用性が低いプログラム言語は使える場所が限定されてしまうのが難点ですが，専用の便利な機能が多い傾向にあるので，目的に合っている場合は強力なプログラム言語となります．いろいろな目的で使う場合は汎用性が高い言語を使うのが良さそうです．

● 各言語の特徴

▶COBOL

事務処理用に作られたプログラム言語なので，それ以外に使うのはほとんどできません．

▶PHP

元はウェブ・ページ作成用のツールから始まったプログラム言語なのでウェブ・ページ処理以外を不得意としています．

▶C言語（と派生のC++やC#）

昔からゲーム開発に使われている実績のあるプログラム言語です．処理速度が速いなどの利点も多いですが，難易度が高くて手間が掛かります．しかし，プログラム開発者になれたら挑戦するのもよいでしょう．

▶Java

身近な物ではAndroidスマートフォンで使われています．C言語より難易度は低いですが，若干処理が遅いという特徴があります．

▶Python

分かりやすさ重視で設計された言語です．処理速度の遅さは不利な点ではありますが，プログラム開発に慣れていない人が作っては修正をするという一連の流れを容易に行えるのが利点です．また，近年人気の人工知能やビッグ・データ解析などの分野でも人気の言語です．覚えやすさと，今後いろいろなジャンルにチャレンジしたい人にとってPythonはよい選択肢になると思います．

表A　各項目に対するプログラム言語の違い

	低 ←			→ 高
人間からの分かりやすさ	アセンブリ言語	C言語	Java	Python
コンピュータからの分かりやすさ	アセンブリ言語	Java	C言語	Python
高度な機能の多さ	アセンブリ言語	C言語	Java	Python
処理の速さ	アセンブリ言語	Java	C言語	Python
複数の目的に対応する汎用性	PHP/COBOL	C言語/アセンブリ言語	Python	Java

表2　映像媒体のフレーム・レート

映像媒体	fps	特　徴
映画	24	アナログ・フィルムの取り扱いに関連して決められた．違和感のない動画として最も低いフレーム数
テレビ（欧州）	50/25	欧州のテレビの方式は日本と異なるPALと呼ばれる方法．映画に近いフレーム数になってる
テレビ（日本）	60/30	アナログ・テレビのデータ転送量の関係で決められた．映画より動きが鮮明．
ビデオ・ゲーム（テレビ・ゲーム機）	60～240	テレビ電波のデータ転送速度に依存しない外部機器は60fps超によるリッチな動画体験ができるので，ゲームにおけるスタンダードなフレーム・レートとなった
人の目	70付近	個人差が大きいが，視細胞から神経系，脳まで伝わる反応速度から70fps付近になる．ただし，脳が視覚情報として認識できるのは半分以下と言われている

くは60fps強のようですが，グラフィックス・システムの能力や処理内容によって上にも下にも変化します．従って，PCでは1フレームごとに掛かった時間を計測して処理に反映させる必要があります．

ささき・ひろたか