プログラミング教室のテクノロ

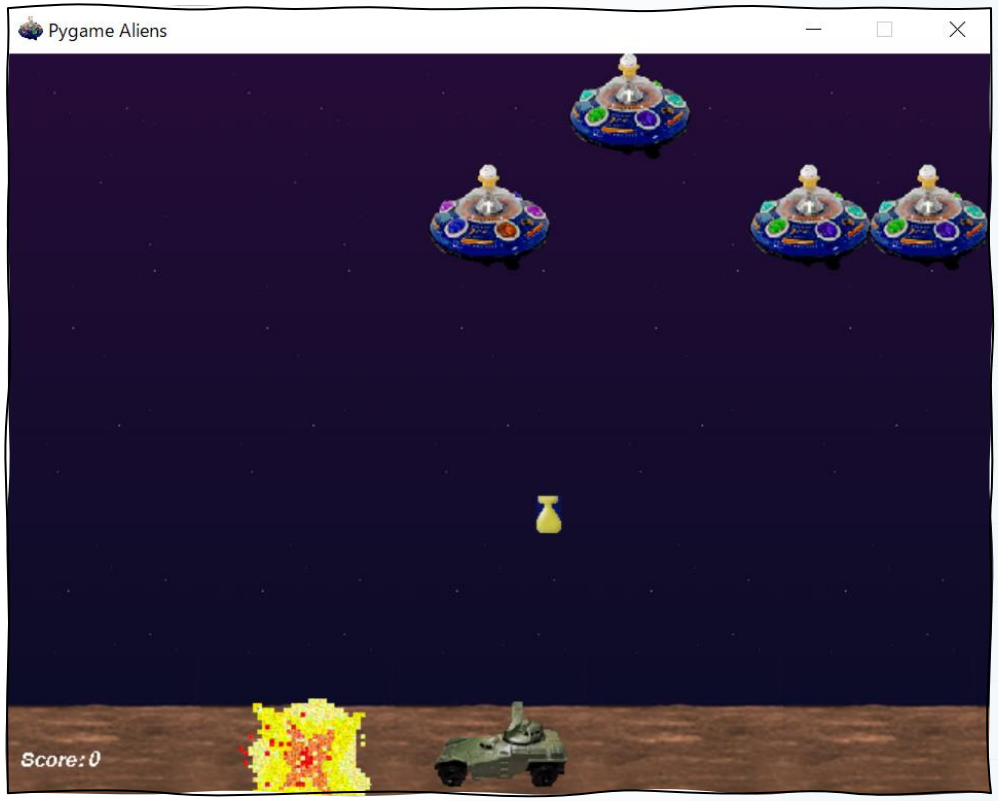
Pythonの道

pygameの使い方①

もくじ

・pygameの使い方

pygameを使って本格的なゲーム開発を学ぶ



pygame(パイゲーム)とは？

pygameはゲームを作るのに役立つ機能がたくさん入ったライブラリである。いろいろな機能があるが、まとめると主に以下のようなことができる。

①ゲーム用にウィンドウを表示できる

②グラフィックスを表示したり動かせる

③キーボードやマウスの入力を調べることができる

④音を鳴らすことができる

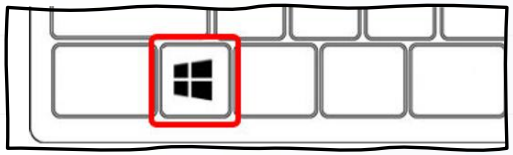
pygameをインストールする

pygameを使えるようにするにはPythonにpygameをインストールする必要がある。Windowsマシンにpygameをインストールする際、コマンドプロンプトを使う。

pygameのインストール

①コマンドプロンプトを起動する

⇒ウィンドウズキーを押して、cmdと入力後にEnterキーを押す

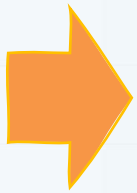


②コマンドが使えることを確認する

⇒コマンドプロンプトに「pip list」コマンドを実行する

※Pythonではパッケージのインストールを簡単にするためにpip(package installer for Python)というパッケージ管理ツールが用意されている。

```
py -m pip list
```



```
C:¥Users¥      >py -m pip list
Package    Version
---------- -------
pip        19.2.3
setuptools 41.2.0
```

pygameをインストールする

以下のような警告文がでた場合

```
WARNING: You are using pip version 19.2.3, however version 22.0.3 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
```

パッケージ管理ツール(pip)のバージョンが古いという警告が出た場合、次のコマンドを使ってpipをバージョンアップしよう

```
py -m pip install -upgrade pip
```

```
C:¥Users¥[redacted]>py -m pip install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/6a/df/a6ef77a6574781a668791419ffe366c8acd1c3cf4709d210cb53cd5ce1c2/pip-22.0.3-py3-none-any.whl (2.1MB)
     |████████████████████████████████| 2.1MB 6.4MB/s
Installing collected packages: pip
  Found existing installation: pip 19.2.3
    Uninstalling pip-19.2.3:
      Successfully uninstalled pip-19.2.3
Successfully installed pip-22.0.3
```

pygameをインストールする

③pygameをインストールする

⇒以下のコマンドを入力後にEnterキーを押す

```
py -m pip install pygame
```

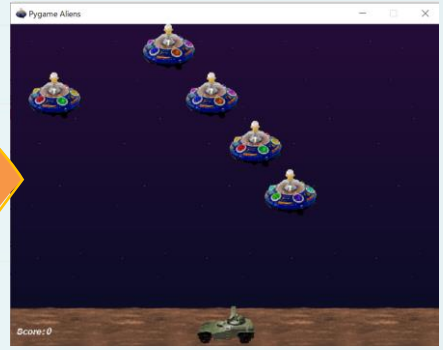
```
C:¥Users¥      >py -m pip install pygame
Collecting pygame
  Downloading pygame-2.1.2-cp38-cp38-win_amd64.whl (8.4 MB)
     ---------------------------------------- 8.4/8.4 MB 10.2 MB/s eta 0:00:00
Installing collected packages: pygame
Successfully installed pygame-2.1.2
```

④pygameが正しくインストールされていることをチェックする

⇒Pythonで以下のプログラムを作成し、実行する

File Edit Format Run Options Window Help

```
1 import pygame.examples.aliens as game
2 game.main()
```



pygameの基本

pygameの基本はループでできていて、ループの中で画像を表示したり、キー入力やマウスの動きを調べたりしている。
pygameのプログラムはゲーム作りの基本的な書き方があり、以下のステップで作っていく。

ゲームを作る7ステップ

1.ゲームの準備をする

2.この下をずっとループする

- 3.画面を初期化する
- 4.ユーザからの入力を調べる
- 5.絵を描いたり、判定したりする
- 6.画面を表示する
- 7.閉じるボタンが押されたら、終了する

四角形を描くプログラムを作る

最初にpygameを使って「四角形を描くだけのプログラム」を作ってみよう。今回はユーザからの入力はないのでステップ4は省略する。

```
import pygame as pg,sys ・・・ 解説1
pg.init() ・・・ 解説2
スクリーン = pg.display.set_mode((800,600)) ・・・ 解説3
while True: ・・・ 繰り返し処理
    スクリーン.fill("white") ・・・ 解説4
    pg.draw.rect(スクリーン,"red",(100,100,100,150)) ・・・ 解説5
    pg.display.update() ・・・ 解説6
    for event in pg.event.get(): ・・・ 解説7
        if event.type == pg.QUIT: ・・・ 解説7
            pg.quit() ・・・ 解説7
            sys.exit() ・・・ 解説7
```

解説1: import ライブラリ名 as 省略名

importしたライブラリ名やモジュール名に省略名をつける。

```
import pygame as pg
```

pygameをpgと省略する命令。「import pygame」とすると
pygame.init()のように毎回「pygame」と書く必要がある。

importしたライブラリやモジュールに省略名をつける
import ライブラリ名 as 省略名

この命令はさらに「import pygame as pg,sys」とカンマで区切ると複数のライブラリをインポートすることができる。

複数のモジュールやライブラリをimportする
import ライブラリ名1 , ライブラリ名2

解説2: ライブラリ名.init()

ライブラリ名.init()を使ってゲームを初期化させる。

pg.init()

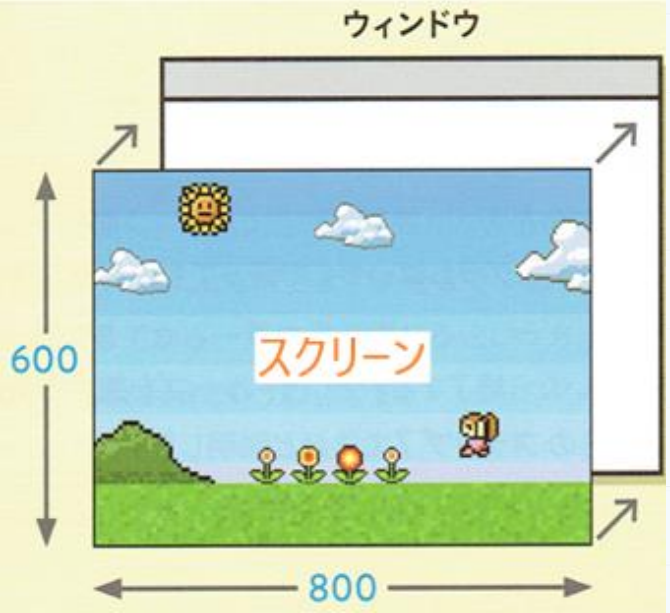
pygameでは、最初にpg.init()と書くことで「これからpygameを使いますよ」という初期化を行う。

initとは「initialize(初期化)」を略したもの。アプリやゲームを動かすときは、画面やキーボードの準備など「舞台を整えるための前準備」が必要になる。

解説3: スクリーンを作る

ライブラリ名.display.set_mode((〇,〇))を使って、ゲームを描くためのスクリーンを作る。tkinterモジュールのキャンバス命令と同じもの。

```
スクリーン ＝ pg.display.set_mode((800,600))
```



スクリーンを定義した後、
スクリーン.fill()命令で画面を初期化したり、
py.draw.rect(スクリーン,〇〇,△△)命令で
画面を表示したりすることができる。

変数「スクリーン」に横800ピクセル、縦600ピクセルのスクリーンを作る命令
tkinterモジュールのキャンバス命令と同じ

```
キャンバス ＝ tkinter.Canvas(ウィンドウ , width ＝ 800 , height ＝ 600)
```

解説4：fill()命令

fill()命令を使って事前に作成したスクリーンに色を付けることができる

スクリーン.fill(“ white”)

色

ライブラリ名.display.set_mode((〇,〇))で作成したスクリーンに色を付けることができる。

【使い方】

スクリーン= pg.display.set_mode((800,600))

スクリーン.fill(“ white”)

解説5：draw.rect()命令

draw.rect()命令を使って事前に作成したスクリーンに図形を描くことができる。

```
pg.draw.rect(スクリーン,”red”，(100,100,100,150))
```

色　x座標　y座標　幅　高さ

スクリーンに指定した色で四角形を描く命令

四角形をどこにどんな風に描くかは、X座標、Y座標、幅、高さの順で指定する。上記の命令ではX100,Y100の座標に幅100、高さ150の長方形を描くことができる。

解説6: display.update()命令

display.update()命令を使ってスクリーンに描いた絵をウィンドウに表示させることができる。

pg.display.update()

【詳細解説】

pygameでは、四角形を描けと命令してすぐ画面に表示されるわけではない。画面に描くという処理は、コンピュータにとって時間がかかる処理になる。ゲーム画面には、背景や主人公や敵など、たくさんのものを描画する必要がある。しかし、毎回画面に描き込んでいては時間がかかってしまってゲームにならない。

「画面に直接描く」という処理は時間がかかるが、「裏側（メモリ）で絵を描いておく」という処理は早く行うことができる。そこで、裏側でゲーム画面に必要なものをあらかじめ描いておいて、全部できてから一気に表示させるという方法で高速表示を行う。その描いておく裏側というのがスクリーンになる。

解説7: ゲームを終了させる処理

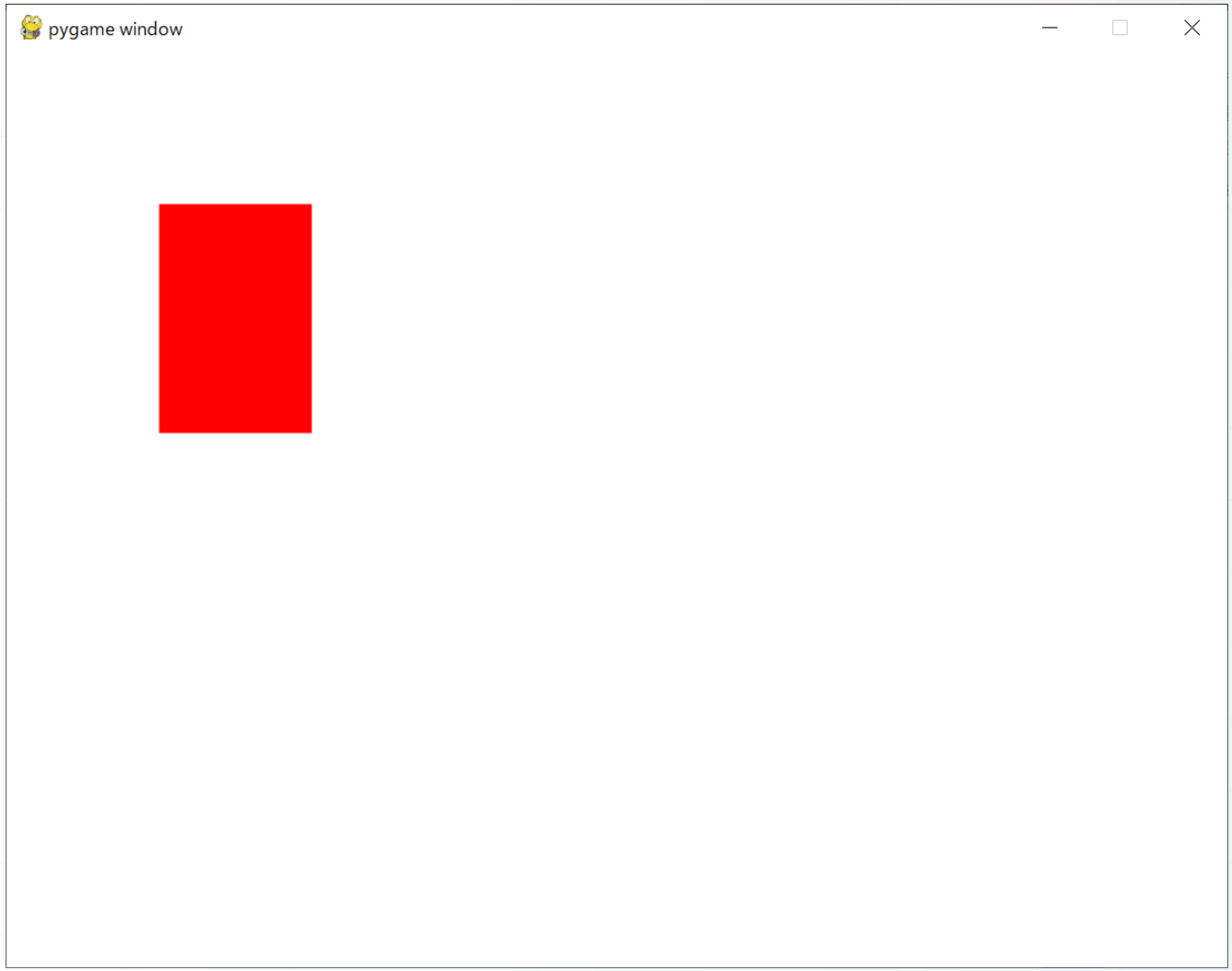
for event in pg.event.get():はイベントを調べる繰り返し文である。マウスやキーボードを使って発生したイベントをpg.event.get()で取得することができる。

```
for event in pg.event.get():
    if event.type == pg.QUIT:
        pg.quit()
        sys.exit()
```

event.typeはイベントの種類のこと。
pg.quit()はpygameを終了させる命令のこと。
sys.exit()はウィンドウを終了させる命令のこと。

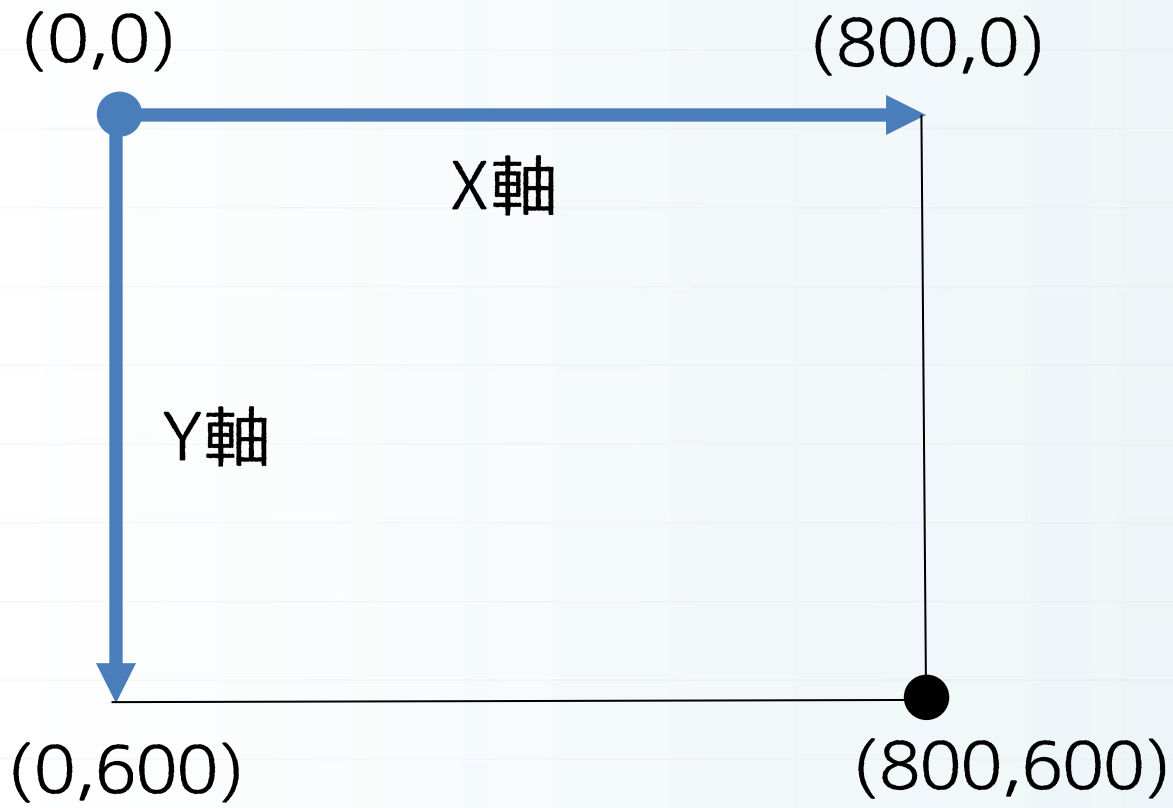
四角形を描くプログラムを作る

「Run Module」またはF5キーを押してプログラムを実行する。



図形や文字、画像を描く

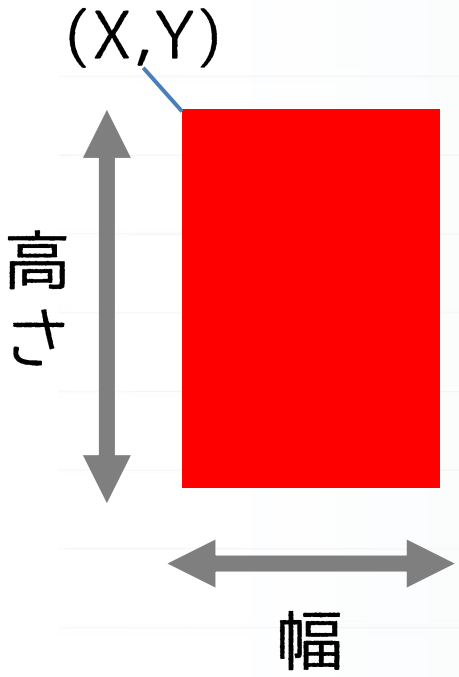
次に画面に絵を描く方法を学習する。pygameの画面の座標は、左上が原点(0,0)になっている。右に進むほどX軸の値が増えて、下へ進むほどY軸の値が増える。



図形や文字、画像を描く

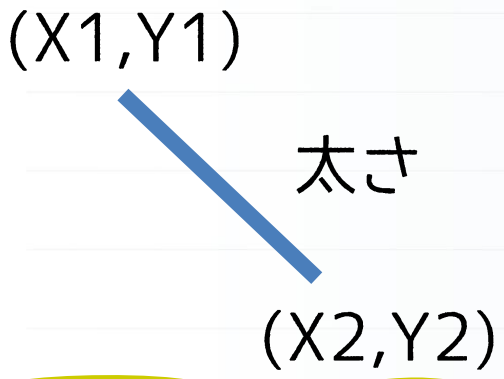
draw関数を使うと、四角形や線、円などの図形を描くことができる。

四角形を描く
pg.draw.rect(スクリーン，色，(x，y，幅，高さ))

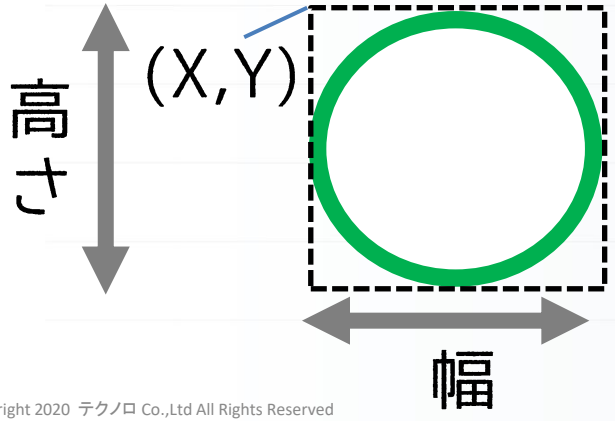


図形や文字、画像を描く

線を引く
pg.draw.line(スクリーン，色，(x1，y1)，(x2，y2)，太さ)



円を描く
pg.draw.ellipse(スクリーン，色，(x，y，幅 ，高さ)，太さ)



図形や文字、画像を描く

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("WHITE")
    # 5.絵を描いたり、判定したりする
    pg.draw.rect(スクリーン, "RED", (100, 100, 100, 150))
    pg.draw.line(スクリーン, "BLUE", (250, 100), (350, 250), 5)
    pg.draw.ellipse(スクリーン, "GREEN", (400, 100, 150, 150), 5)
```

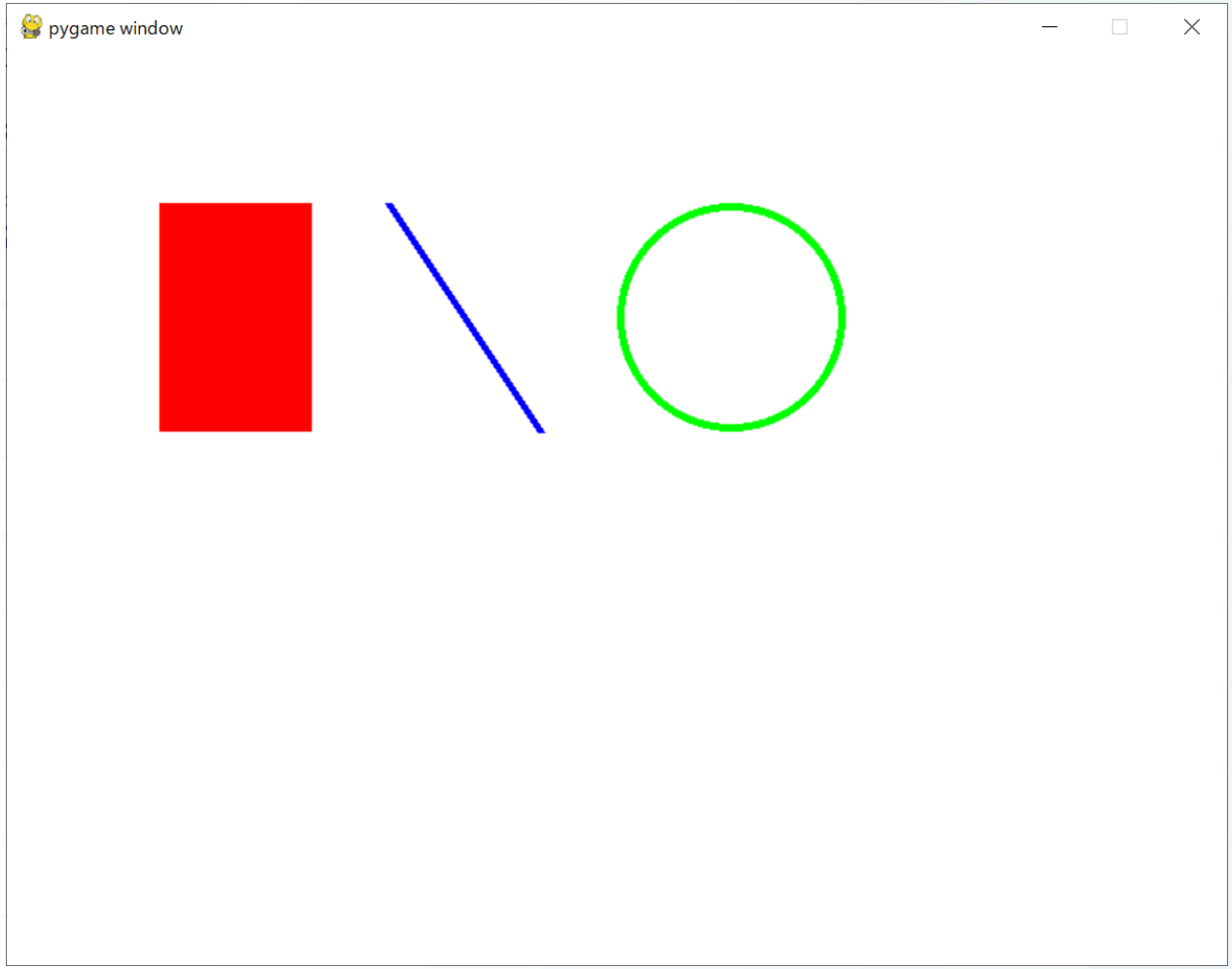
次のページに続く

図形や文字、画像を描く

```
# 6.画面を表示する
pg.display.update()
# 7.閉じるボタンが押されたら、終了する
for event in pg.event.get():
    if event.type == pg.QUIT:
        pg.quit()
        sys.exit()
```

図形や文字、画像を描く

「Run Module」またはF5キーを押してプログラムを実行する。



図形や文字、画像を描く

スクリーンに画像を描く場合、image.load関数を使って画像を読み込む。次にblit関数を使って読み込んだ画像を描画する。

画像を読み込む
画像変数=pg.image.load(“ 画像ファイルパス”)

読み込んだ画像を描画する
スクリーン.blit(画像変数,(X,Y))
※blit 転送する

図形や文字、画像を描く

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
img1 = pg.image.load("car.png")
# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("WHITE")
    # 5.絵を描いたり、判定したりする
    スクリーン.blit(img1, (100,100))
    # 6.画面を表示する
    pg.display.update()
```

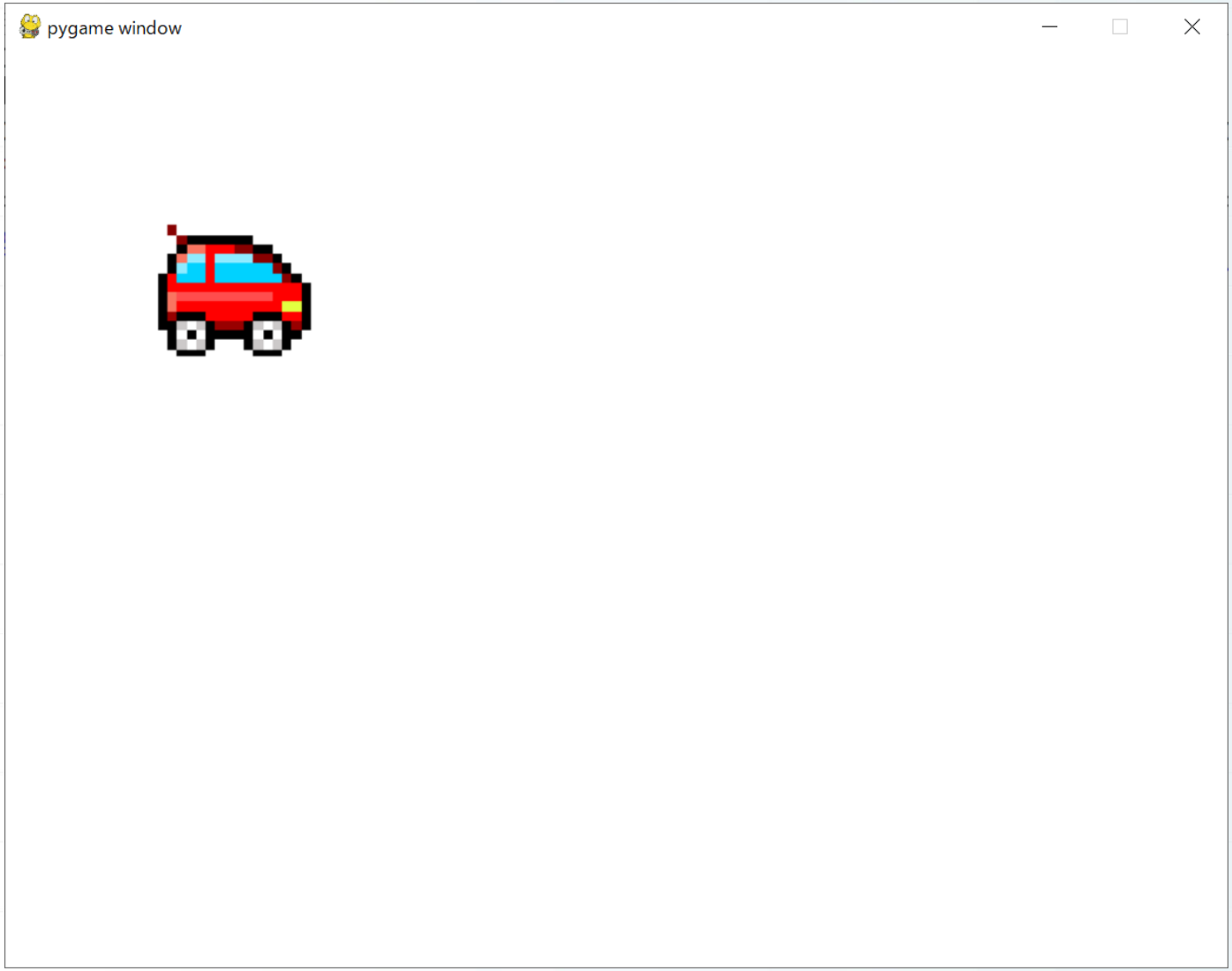
次のページに続く

図形や文字、画像を描く

```
# 7.閉じるボタンが押されたら、終了する
    for event in pg.event.get():
        if event.type == pg.QUIT:
            pg.quit()
            sys.exit()
```

図形や文字、画像を描く

「Run Module」またはF5キーを押してプログラムを実行する。



図形や文字、画像を描く

transform.scale関数を使って画像のサイズを変更する。

画像のサイズを変更する

画像変数 ＝ pg.transform.scale(画像変数, (幅 , 高さ))

画像サイズの変更手順
①画像を読み込む
②その画像のサイズを変更する
③その画像を描画する

図形や文字、画像を描く

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
img1 = pg.image.load("car.png")
img1 = pg.transform.scale(img1, (50, 50))
# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("WHITE")
    # 5.絵を描いたり、判定したりする
    スクリーン.blit(img1, (100,100))
```

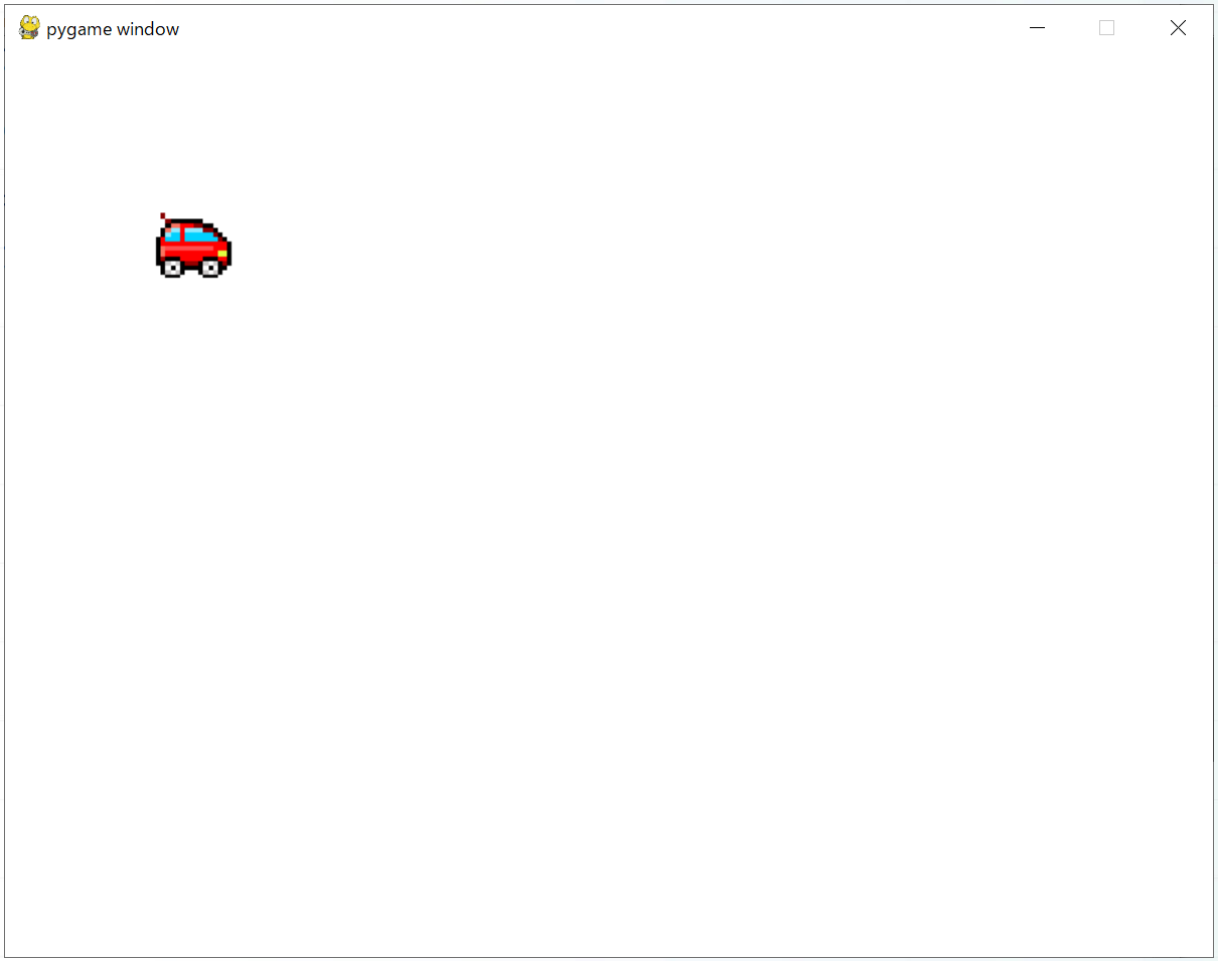
次のページに続く

図形や文字、画像を描く

```
# 6.画面を表示する
    pg.display.update()
    # 7.閉じるボタンが押されたら、終了する
    for event in pg.event.get():
        if event.type == pg.QUIT:
            pg.quit()
            sys.exit()
```

図形や文字、画像を描く

「Run Module」またはF5キーを押してプログラムを実行する。



図形や文字、画像を描く

pygameで文字列を表示させるには、文字を画像にしてからその画像を描画させる必要がある。

文字列を表示手順

①フォントを用意する

②そのフォントを使って、文字列の画像を作る

③その画像を描画する

フォントを読み込んで文字列の画像を作る

```
font = pg.font.Font(None, 文字サイズ)
```

※デフォルトのフォントを指定する場合、Noneを使う

```
画像変数 = font.render(文字列, True, 色)
```

※render 与える

※文字の境界を滑らかにするか

⇒True（滑らかにする）、False（かくかく）

図形や文字、画像を描く

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
font = pg.font.Font(None, 50)
text_img = font.render("hello!", True,"BLUE")

# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("WHITE")
    # 5.絵を描いたり、判定したりする
    スクリーン.blit(text_img, (200, 100))
```

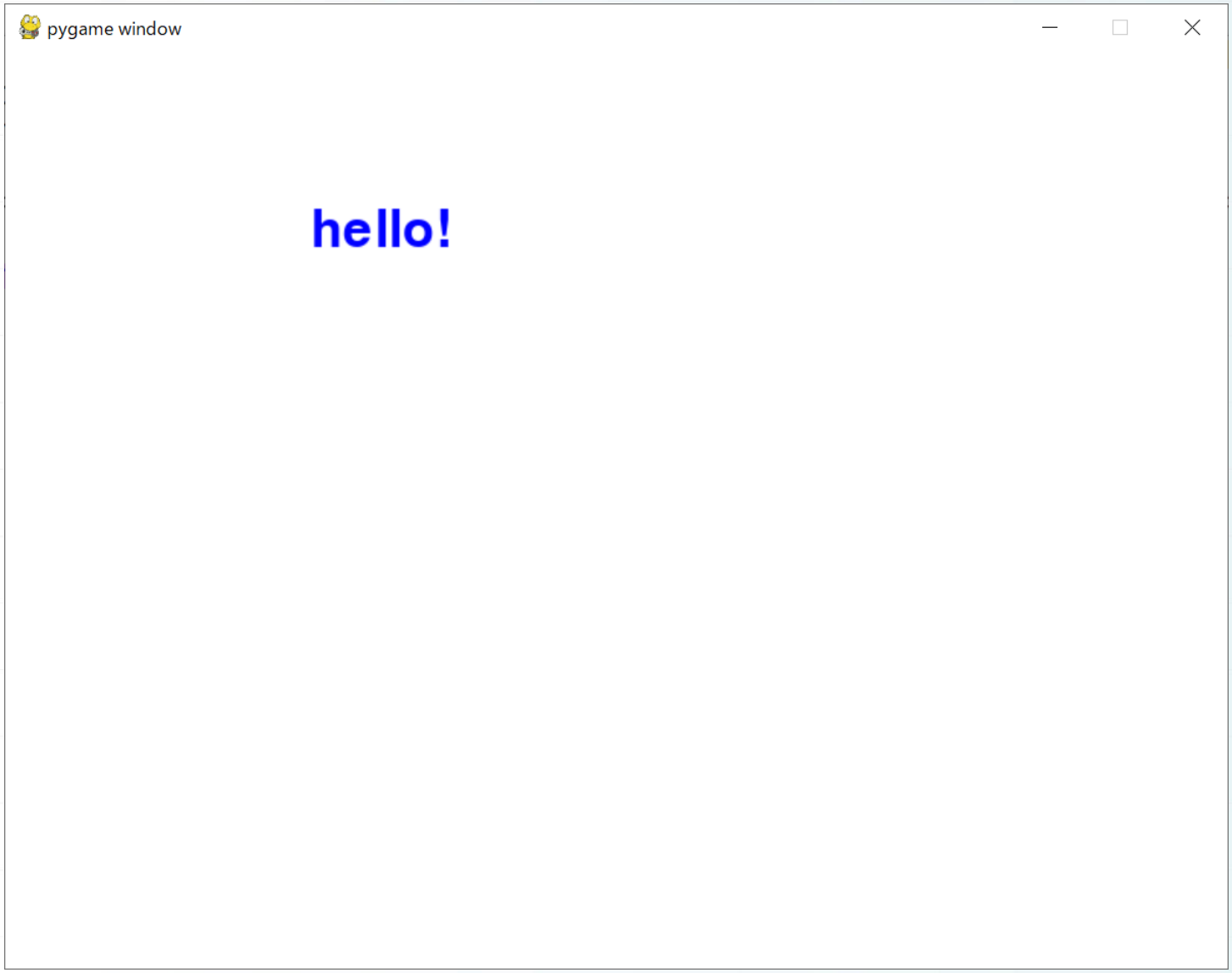
次のページに続く

図形や文字、画像を描く

```
# 6.画面を表示する
    pg.display.update()
    # 7.閉じるボタンが押されたら、終了する
    for event in pg.event.get():
        if event.type == pg.QUIT:
            pg.quit()
            sys.exit()
```

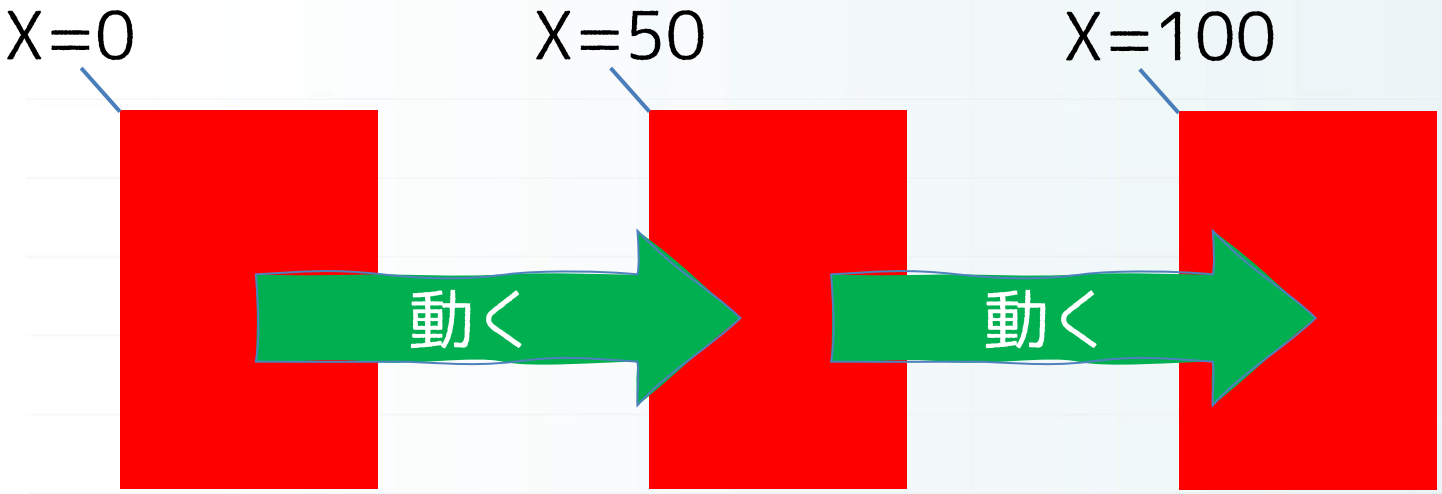
図形や文字、画像を描く

「Run Module」またはF5キーを押してプログラムを実行する。



グラフィックスを動かす

止まっている絵を動かす。絵を動かすためにプログラムをループさせて、X座標、Y座標を移動させていく。



ゲームの中では、ただキャラクタを移動させるだけでなく、「壁や敵と衝突したか」を調べる必要がある。この「X座標、Y座標、幅、高さ」をひとまとめにしたものが「Rect」命令になる。

位置と大きさをまとめて扱う
変数 ＝ pg.Rect(X, Y, 幅 , 高さ)

Rectの中の引数には、変数に「.」と「X,Y,width,height」をつける。

グラフィックスを動かす

四角形が横に移動するプログラムを作る。

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
図形位置 = pg.Rect(100,100,100,150) ・・・ 図形位置を制御する変数
# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("WHITE")
    # 5.絵を描いたり、判定したりする
    図形位置.x = 図形位置.x + 1 ・・・ 図形のX座標を１づつ変える
    pg.draw.rect(スクリーン, " red", 図形位置)・・・ 解説1
```

次のページに続く

グラフィックスを動かす

```
# 6.画面を表示する
pg.display.update()
pg.time.Clock().tick(60)  ・・・ 解説2
# 7.閉じるボタンが押されたら、終了する
for event in pg.event.get():
    if event.type == pg.QUIT:
        pg.quit()
        sys.exit()
```

解説1：draw.rect()命令

①ゲームの準備をする

```
図形位置 = pg.Rect(100,100,100,150)
```

図形をどこに表示させるかを変数 「図形位置」に定義する。

②絵を描いたり、判定したりする

```
図形位置.x = 図形位置.x + 1
pg.draw.rect(スクリーン,” red” ,図形位置)
```

図形位置でX,Y,幅,高さを指定

ループの中で、「図形位置」のｘに１を足して、表示させる位置を少し右に移動させる。

解説2: time.Clock().tick()命令

処理速度を調整するためにtime.Clock.tick()命令を使う

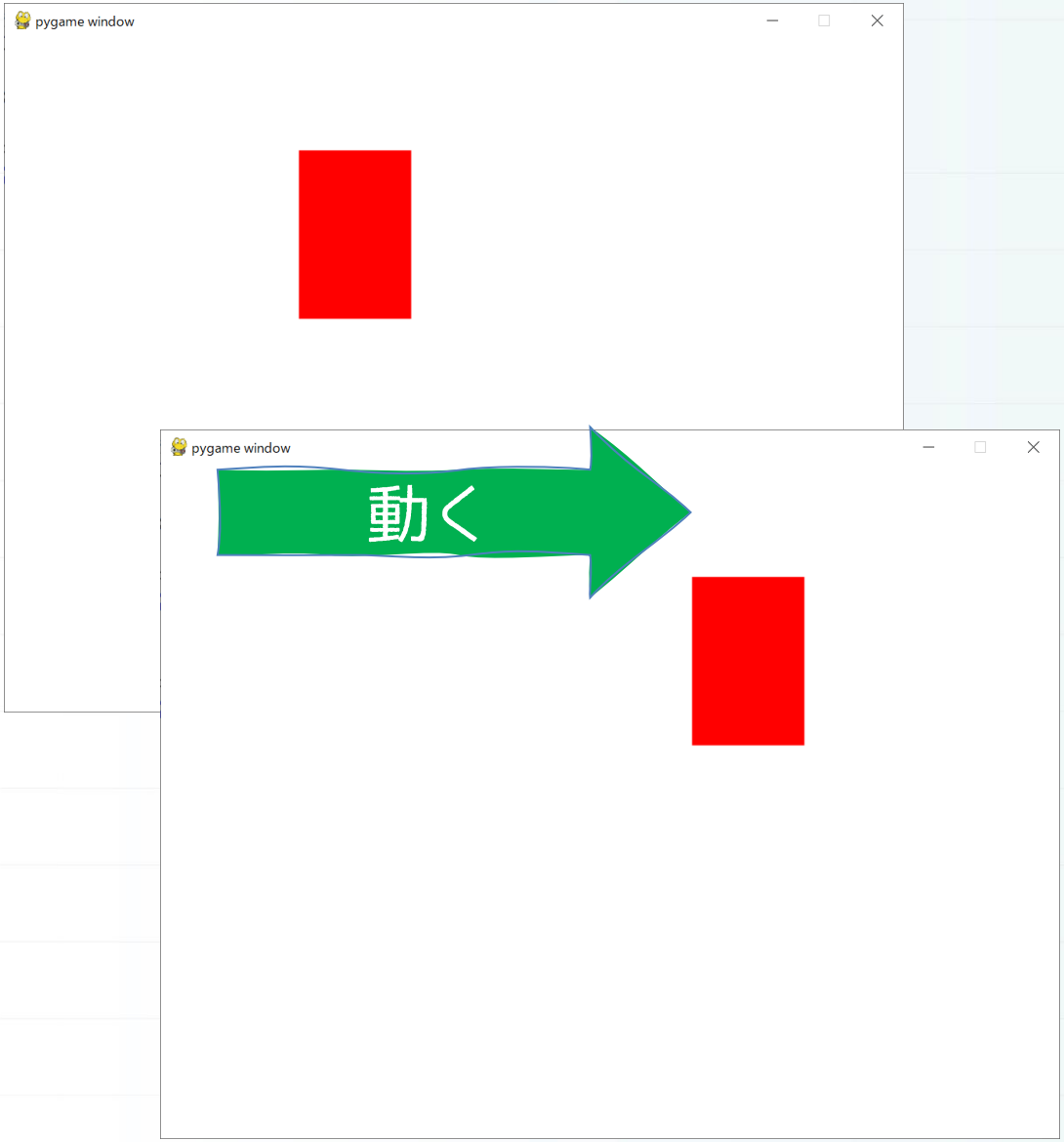
```
pg.display.update()
pg.time.Clock().tick(60)
```

pg.time.Clock().tick(60)と記載することで1秒間に60回以下のスピードにすることができる。

スピードを調整する
pg.time.Clock().tick()
1秒間にこの回数以下のスピードにする

グラフィックスを動かす

「Run Module」またはF5キーを押してプログラムを実行する。



車の画像を動かす

画像ファイルを読み込んで車の画像を動かしてみよう。

```
# 1.ゲームの準備をする
import pygame as pg, sys
pg.init()
スクリーン = pg.display.set_mode((800, 600))
img1 = pg.image.load("car.png")
img1 = pg.transform.scale(img1, (50, 50))
図形位置 = pg.Rect(100,100,50,50)
# 2.この下をずっとループする
while True:
    # 3.画面を初期化する
    スクリーン.fill("white")
```

次のページに続く

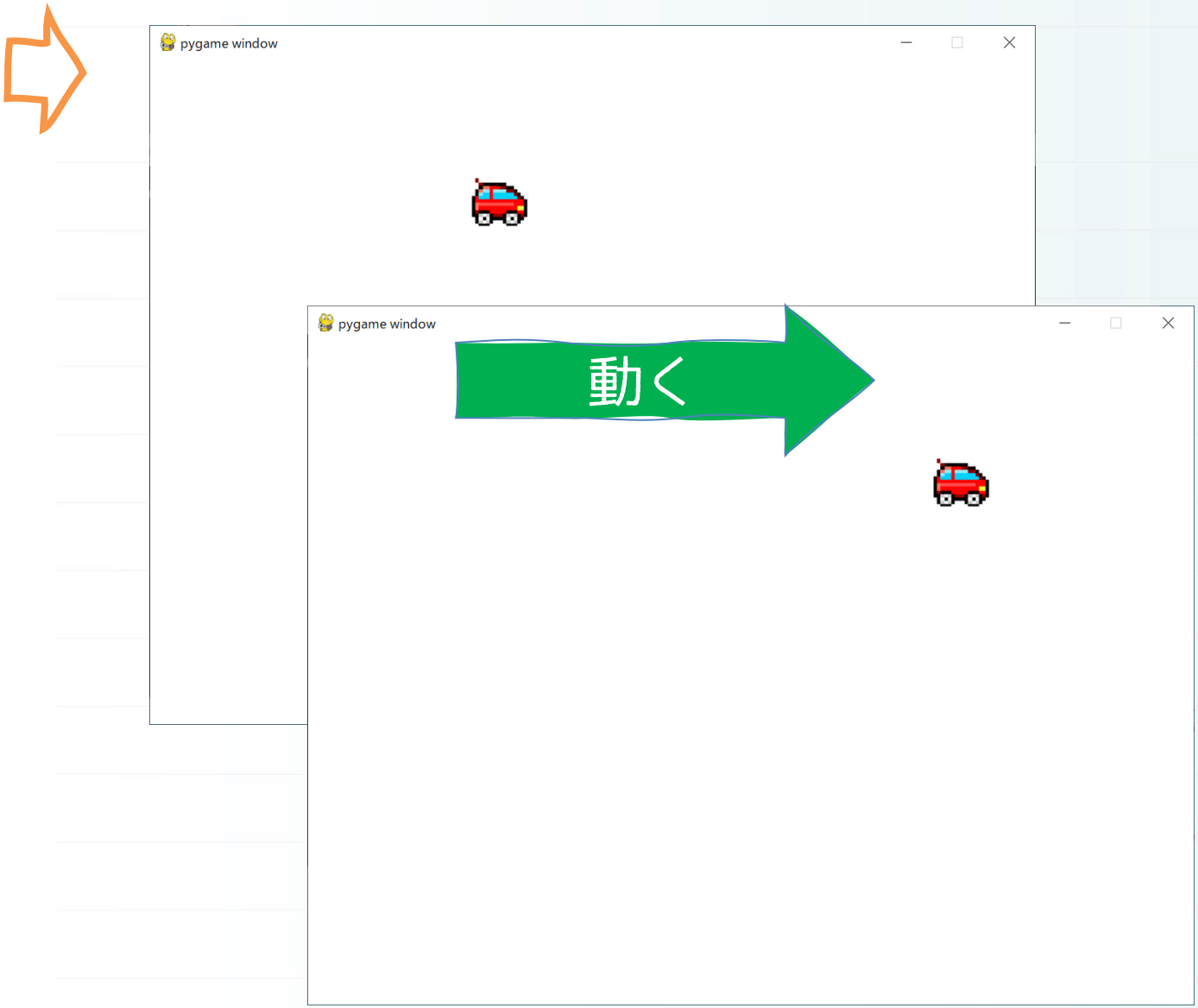
車の画像を動かす

画像ファイルを読み込んで車の画像を動かしてみよう。

```
# 5.絵を描いたり、判定したりする
    図形位置.x = 図形位置.x + 1
    スクリーン.blit(img1,図形位置)
    # 6.画面を表示する
    pg.display.update()
    pg.time.Clock().tick(60)
    # 7.閉じるボタンが押されたら、終了する
    for event in pg.event.get():
        if event.type == pg.QUIT:
            pg.quit()
            sys.exit()
```

車の画像を動かす

「Run Module」またはF5キーを押してプログラムを実行する。



復習＆チャレンジ

ここまで習ったことをScratchでもできるかチャレンジしてみよう。
その過程でScratchでできること、Pythonでないとできないことを整理してみよう。

メモ

プログラミング教室のテクノロ

なまえ：