FRUIT CATCHER(フルーツキャッチャー)

GameDoc(ゲームの説明書)

最初にGameDocを読んでみましょう。

```
いくつかのフルーツが画面上に飛んできます。
それらのフルーツをクリックして真っ二つにします。
ハイスコアを目指しましょう！
```

始めましょう

1. 先生がテスト用の画面を準備してくれます。あなたの画面には何も書かれていないの "game.py" ウィンドウが表示されていると思います。
2. 必要なものはすべて準備されています！　複雑な設定をプログラムしたり、画面を表示するためのコードなど必要はありません。
3. 先生と一緒に作業していきましょう！

モジュール

必要なモジュール(=部品)をすべてインポートして(=取り込んで)みましょう。

```
import screen_setting
import fruit
import scoreboard
import random
```

Pythonでは他の人が作った部品を取り込んで使うことがよくあります。今回は以上の4つのモジュールを使うために取り込むコードを書く必要があります。

モジュールの紹介

- `screen_setting` ..... 画面を表示するためのモジュール。
- `fruit` フルーツ自体のモジュールです。
- `scoreboard` スコアの表示などを扱うモジュールです。
- `random` ランダムな値を使うためのモジュールです。

オブジェクトの初期化

ゲームウィンドウとスコアボードを作成しましょう。

```
game_win = screen_setting.Game_Screen()
scoreboard = scoreboard.Scoreboard(game_win.surface, 0, 0)
```

次に、フルーツのリストを作成しましょう。 フルーツのリストと、空のリストを作成します。

```
fruits = ["apple", "avocado", "banana", "cherry", "grapes", "orange",
"strawberry"]
fruit_list = []
```

後で、Pythonにどのフルーツを欲しいか伝える必要があります。今の段階では、混んでいるレストランに行って注文する前に席を確保するように、場所を印を付けることのような感じになります。

```
this_fruit = None
```

これまでのところ順調です

初期化を2つ作成する必要があります。最初のものを作りましょう！

```
def init():
    global fruit_list
    for i in range(0, len(fruits)):
        fruit_list.append(fruit.Fruit(game_win.surface,
fruit_type=fruits[i]))
    scoreboard.score = 0
```

`init`は初期化を表し、ゲームやプログラムの最初に設定する項目を決める部分になります。

ここでは、`global` キーワードを使用して `fruit_list` にアクセスします。これは出てくるフルーツのリストになります。 `for` ループの中では、0から `fruits` リストの長さまでの値をセットして、`fruit.Fruit` からオブジェクトを作成して `fruit_list` に `append` します。私たちは `fruits` リストのすべてのフルーツを使用して、リストを埋めます。

また、スコアを0に設定します。

次に、フルーツを取得しましょう。

```
def get_fruit():
    global this_fruit
    this_fruit = fruit_list[random.randrange(0, len(fruit_list))]
    this_fruit.posY = 500
    this_fruit.posX = random.randrange(0,
this_fruit.fruit_sprite.get_width())
    this_fruit.isVisible = True
    this_fruit.set_speed()
```

ここでは、`global` キーワードを使用して `this_fruit` にアクセスします。

1. `this_fruit` を `fruit_list` かごの中のランダムなフルーツに設定します。
2. y 座標を500に設定します。画面の少し下に配置します。
3. x 座標を画面の左端から右端までのランダムな位置に設定しますが、フルーツのスプライト画像の幅を引いた位置にします。
    1. これは、フルーツの一部が画面に表示されたり、画面が一時的に空白になるのを避けるためです。それでは楽しいゲームになりません。
4. フルーツの表示を `True` に設定します。
5. フルーツの速度を設定します。

次に、スコアボードを更新しましょう。

```
def update_score():
    if not this_fruit.isVisible:
        scoreboard.score += 1
        scoreboard.say(scoreboard.score)
```

これは、フルーツがクリックされ、フルーツを取るときに起動されます。

すばらしい仕事！もうすぐ完成です！ では、`run()` というメインのゲーム関数を作成しましょう。

```
def run():
    init()
    get_fruit()
    while True:
        game_win.update()
        this_fruit.update()
        this_fruit.draw()
        scoreboard.draw()
        update_score()
        if this_fruit.posY >= 500 or not this_fruit.isVisible:
            get_fruit()
            continue
```

`run()` を実行すると、以下が行われます：

1. `init()` を実行します。
2. 次に、`get_fruit()` を実行します。
3. その後、`while True` の中でゲームループが行われます。
    1. 画面の表示をフレームごとに更新します。
    2. フルーツをフレームごとに更新します。
    3. フルーツを画面に描画します。これはフレームごとに更新されます。
    4. スコアを画面に描画します。
    5. スコアをフレームごとに更新します。新しいスコアを表示します。
    6. フルーツが画面の下に落ちた場合、またはフルーツが見えなくなった場合、新しいフルーツを取得します。
    7. `continue` を呼び出してループを繰り返します。

すばらしい仕事！さあ、スクリプトを完成させましょう！

次のスクリプトの部分は重要です。多くのモジュールがあるため、Pythonにこのスクリプトがモジュールではなくメインのスクリプトであることを伝える必要があります。次のようにします：

```
if __name__ == '__main__':
    run()
```

CTRL+S を押して保存してください。

その後、**F5** を押して実行してください！

遊び方

ゲームの動作方法は次の通りです。

- 画面中央のフルーツをクリックします。捕まえることができますか？
- フルーツを捕まえると、1ポイント獲得できます！
- 終了する場合は、ゲームを終了するにはESCキーを押します。

チャレンジ

ゲームを自分のスタイルにカスタマイズする方法は見つけられますか？ 各モジュールの `.py` ファイルを見て、そのコードを自分で確認することを恐れずに行ってみてください。よりクールなものにしたいと思うかもしれませんね？

アイデア:

- 勝ち負けの条件を追加する
- 効果音を追加する

もしゲーム作成が楽しかったら、IT KiDSに参加しましょう

このようなゲームの作り方やもっと多くを学べます！　Pythonプログラミングが興味を持ったけど、自分が入力した内容で、何が起こったのかよくわからない、それでも全然問題ありません！　IT KiDSには、Pygameのカリキュラムの一部としてPython入門コースがあります！　お気軽にお問い合わせください！

また、お会いできることを楽しみにしています！！