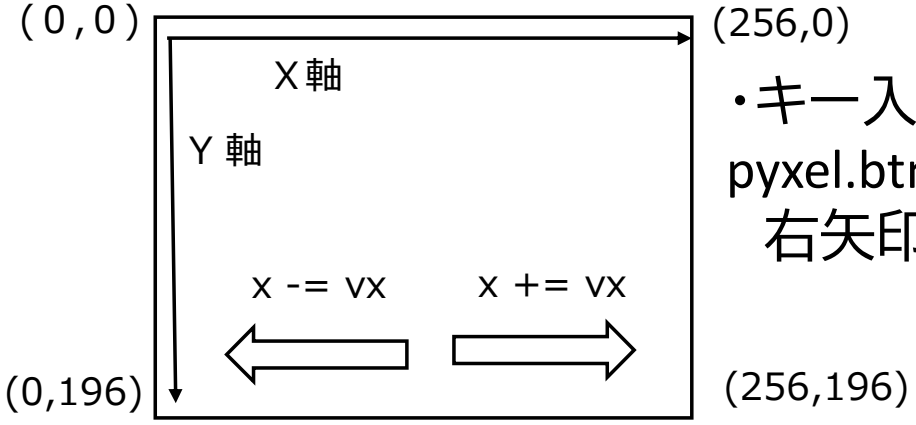
GAME-MASTER MISSION_2

アニメでゲーム作りを学ぼう（2/3）

ホワイトボード

ゲーム開発マスター　MISSION 2　　ピクセルマン the アニメを作る（2/3）

●ピクセルマンの左右の動き



・キー入力の関数

pyxel.btn(pyxel.KEY_RIGHT)

　右矢印キーがおされている間 True

●花火の作り方（花火のクラス）

```
class Hanabi :          ←クラス(設計図)「Hanabi」の定義がはじまる
    def __init__(self) :
        ～              ←ここに初期処理を書く
    def update(self) :
        ～              ←ここに花火の座標の更新(移動)を書く
    def draw(self) :
        ～              ←ここに花火の描画を書く
```

このクラス（設計図）は、Pixelmanの中で実体化する！

■時間割

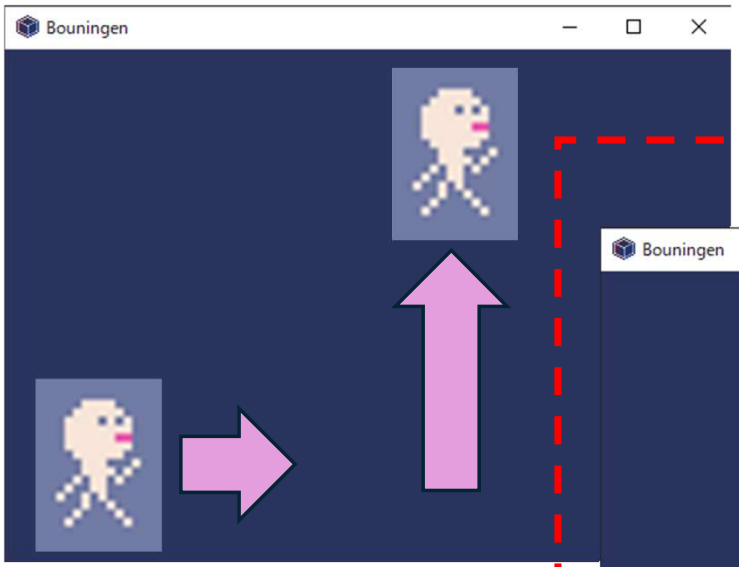
MISSION_2

寿司打	5分		
キー入力で左右に動く	50分		
キー入力で花火を発射	120分	花火クラスの説明/作成	・・・60分
寿司打	5分	組込みの説明/作成	・・・60分

合計　　180分

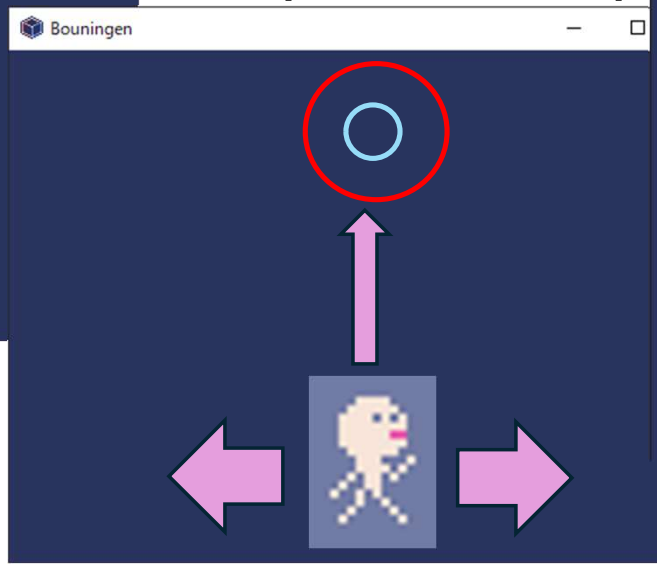
アニメでゲーム作りを学ぶ

前回(MISSION_1)



ﾋﾟｸｾﾙﾏﾝ、走る、ジャンプ

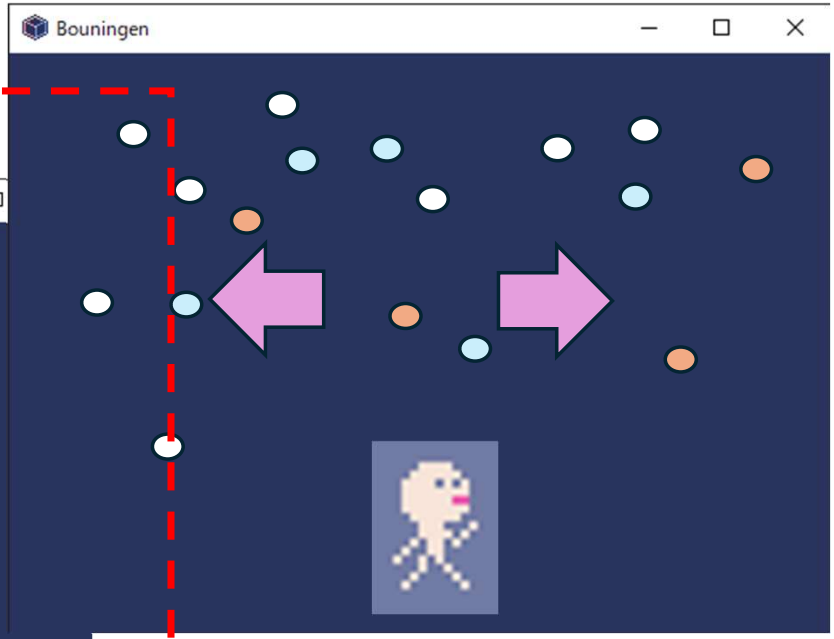
今回(MISSION_2)



ﾋﾟｸｾﾙﾏﾝ、左右に走る、花火を上げる

次回(MISSION_3) （完成）

ﾋﾟｸｾﾙﾏﾝ、走って、ジャンプ、花火を上げる、背景を動かして完成させよう！



ピクセルマンをキー操作で左右に動かす

- Pixelmanのクラスを改造して、キー操作で左右に動かす。

プログラムを打ち込んで、
動かしてみよう！

```
def update(self,anime)
    if self.bn_ptn == 0 :  ･･･ピクセルマンが飛んでるとき
      ～～～
    else :                 ･･･ピクセルマンが飛んでるとき以外(横移動)
        if pyxel.btn(pyxel.KEY_RIGHT) == True :  ･･･「→」が押されたら
           self.bn_x += self.bn_vx
        elif pyxel.btn(pyxel.KEY_LEFT) == True : ･･･「←」が押されたら
           self.bn_x -= self.bn_vx

        if self.bn_x > Anime.SCREEN_WIDTH :      ･･･画面の右端からでたら
           self.bn_x = 0
        elif self.bn_x < -16 :                   ･･･画面の左端からでたら
           self.bn_x = Anime.SCREEN_WIDTH - 16

    ～～～
```

参考:キー操作の関数

- キー操作の関数3種類

```
pyxel.btnp(キーの値)   ・・・キーが押されたら       True
pyxel.btn(キーの値)    ・・・キーが押されている間 True
pyxel.btnr(キーの値)   ・・・キーがはなされたら     True
```

- 主なキーの値

英字
- KEY_A
- KEY_B
- ～

数字
- KEY_1
- KEY_2
- ～

矢印
- KEY_LEFT
- KEY_RIGHT
- KEY_UP
- KEY_DOWN

その他
- KEY_SPACE
- KEY_RETURN
- KEY_TAB
- KEY_LSHIFT
- KEY_RSHIFT
- KEY_ESCAPE

花火のクラスを作成しよう(1/3)

・まずは、初期処理

```
class Hanabi ：

    def __init__(self,anime,x,y) ：          ･･･初期処理の定義

        self.anime = anime                   ･･･花火はAnimeから呼ばれるので宣言
        self.anime.hanabis.append(self)      ･･･花火は発射の都度、実体化するので
                                                 Animeで宣言したhanabisに追加
        self.hana_x = x + 8                  ･･･花火の初期のX座標は、ピクセルマン
                                                 の頭上となるように8(16/2)を足す
        self.hana_y = y                      ･･･花火の初期のY座標は、ピクセルマン
                                                 と同じ
        self.hana_cnt = 0                    ･･･花火の爆発までを数える変数(ｶｳﾝﾄ)

        self.hana_tim = 150                  ･･･花火が爆発する数値
```

花火のクラスを作成しよう(2/3)

・つぎは、更新処理(花火の座標の更新と、花火の形)

```
def update(self) :
        self.hana_cnt += 1                                  ･･･爆発までのカウント
        if self.hana_cnt < self.hana_tim :                  ･･･カウントが150になるまで
            self.hana_y -= 1                                  ･･･上に動かす(Y座標を-1)
            self.hana_ptn = 0                                 ･･･花火の元の形(火の塊)
        elif self.hana_cnt < self.hana_tim+30 :             ･･･カウントが150を超え180まで
            self.hana_ptn = 1                                 ･･･最初の輪
        elif self.hana_cnt < self.hana_tim+60 :             ･･･カウントが180を超え210まで
            self.hana_ptn = 2                                 ･･･次の輪
        elif self.hana_cnt < self.hana_tim+90 :             ･･･カウントが210を超え240まで
            self.hana_ptn = 3                                 ･･･最後の輪
        else :                                              ･･･カウントが240を超えたら
            if self.anime.hanabis :                           ･･･花火を消す
                self.anime.hanabis.remove(self)
```

花火のクラスを作成しよう(3/3)

・最後は、描画処理(画面に花火を描く)

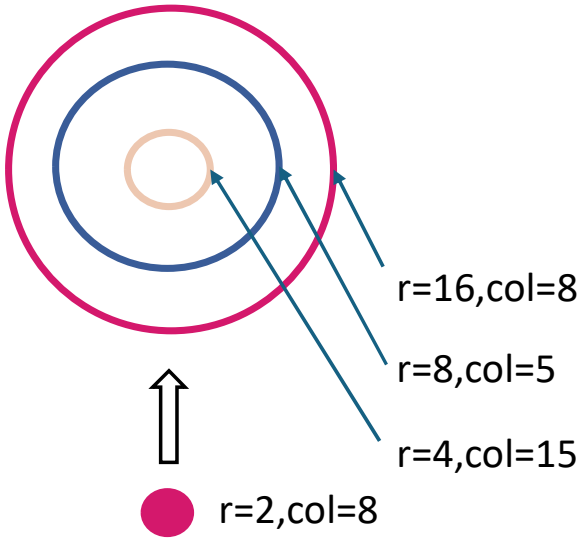
```
def draw(self) :
    if self.hana_ptn == 0 :                              ・・・カウントが150になるまで
        pyxel.circ(self.hana_x,self.hana_y,2,8)          ・・・花火の元の形(火の塊)

    elif self.hana_ptn == 1 :                            ・・・カウントが150を超え180まで
        pyxel.circb(self.hana_x,self.hana_y,4,15)        ・・・最初の輪

    elif self.hana_ptn == 2 :                            ・・・カウントが180を超え210まで
        pyxel.circb(self.hana_x,self.hana_y,4,15)        ・・・最初の輪　と
        pyxel.circb(self.hana_x,self.hana_y,8, 5)        ・・・次の輪

    elif self.hana_ptn == 3 :                            ・・・カウントが210を超え240まで
        pyxel.circb(self.hana_x,self.hana_y,8, 5)        ・・・次の輪　　と
        pyxel.circb(self.hana_x,self.hana_y,16,8)        ・・・最後の輪
```

参考:円を描く関数

- ピクセルで円を描く関数のまとめ

  pyxel.circ(x, y, r, col)
  半径 r、色 colの円を（x, y）に描画します。

  pyxel.circb(x, y, r, col)
  半径 r、色 colの円の輪郭線を（x, y）に描画します。



カラーコード表（colの値は、０～１

0 #000000 0,0,0	1 #2B335F 43,51,95	2 #7E2072 126,32,114	3 #19959C 25,149,156
4 #8B4852 139,72,82	5 #395C98 57,92,152	6 #A9C1FF 169,193,255	7 #EEEEEE 238,238,238
8 #D41B6C 212,24,108	9 #D38441 211,132,65	10 #E9C35B 233,195,91	11 #70C6A9 112,198,169
12 #7696DE 118,150,222	13 #A3A3A3 163,163,163	14 #FF9798 255,151,152	15 #EDC7B0 237,199,176

花火のクラスを組み込もう(1/3)

- まずは、クラス(設計図)「Pixelman」の中に組み込むよ。
- 組み込む内容は、「リターンキー」が押されたら、花火を実体化する。

```
class Pixelman :
      ～
    def __init__(self,anime) :
        ～
    def update(self,anime) :
        ～
        if pyxel.btnp(pyxel.KEY_RETURN) :
            Hanabi(self.anime, self.bn_x , self.bn_y)
        ～

def draw(self) :
        ～
```

・・・初期処理には影響なし

・・・更新処理に組み込む

・・・「リターンキー」が押されたら、花火を実体化！

ピクセルマンのX座標、Y座標を引数として渡す

花火のクラスを組み込もう(2/3)

- 次に、クラス(設計図)「Anime」の中に組み込むよ。
- 組み込む内容は、まずは、花火の配列の初期化

```
class Anime :
    ～
    def __init__(self) :
        ～
        self.hanabis=[]        ･･･花火は複数実体化するので
        ～                          配列「hanabis」を作っておく
```

花火のクラスを組み込もう(3/3)

- つぎに、花火の座標の更新の呼び出し、花火の描画の呼び出しを組み込む。

```
class Anime :
    ～
    def update(self) :
        self.pixelman.update(Anime)
        for hanabi in self.hanabis.copy() :
            hanabi.update()

    def draw(self) :
        ～
        for hanabi in self.hanabis.copy() :
            hanabi.draw()
```

Pixelmanから花火を実体化するため、Animeを渡す

実体化している花火の分だけ花火の座標の更新を呼ぶ

実体化している花火の分だけ花火の描画を呼び出す