Игра CubeBuild

Презентация Дастана Мусрепова, о игре под названием CubeBuild, сделанной на Panda3D.

Мой код - game.py

```
from direct.showbase.ShowBase import ShowBase
from mapmanager import Mapmanager
from hero import*
class Game(ShowBase):
    def __init__(self):
        ShowBase.__init__(self)
        self.land=Mapmanager()
        x,y=self.land.loadLand('land.txt')
        self.hero=Hero((x//2,y//2,2),self.land)
        base.camLens.setFov(90)
game=Game()
game.run()
```

В этом файле я сначала импортирую библиотеку и другие части кода, создаю класс ‘Game’, в нем делаю ‘__init__’, загружаю карту, создаю игрока и поворачиваю камеру на 90 °, так чтобы она смотрела прямо. Создаю и запускаю игру.

Мой код - mapmanager.py

```
class Mapmanager():
    def __init__(self):
        self.model='block'
        self.texture='block.png'
        self.block=loader.loadModel(self.model)
        self.block.setTexture(
            loader.loadTexture(self.texture))
        self.color=(0.2,0.2,0.35,1)
        self.startNew()
        self.addBlock((0,10,0))
    def startNew(self):
        self.land=render.attachNewNode('Land')
    def loadLand(self,filename):
        with open(filename)as file:
            y=0
            for line in file:
                x=0
                line=line.split(' ')
                for z in line:
                    for z0 in range(int(z)+1):
                        block=self.addBlock(
                            (x,y,z0))
                    x+=1
                y+=1
        return x,y
    def addBlock(self,pos):
        self.block=loader.loadModel(self.model)
        self.block.setTexture(
            loader.loadTexture(self.texture))
        self.block.setPos(pos)
        self.block.setColor(self.color)
        self.block.setTag('at',str(pos))
        self.block.reparentTo(self.land)
```

В этой части я создаю класс ‘Mapmanager’ и делаю ‘__init__’, создаю карту, читаю файл с нужной картой, и расставляю блоки.

Мой код - mapmanager.py

```
    def findBlocks(self,pos):
        return self.land.findAllMatches(
            '=at='+str(pos))
    def isEmpty(self,pos):
        blocks=self.findBlocks(pos)
        if blocks:
            return False
        else:
            return True
    def findHighestEmpty(self,pos):
        x,y,z=pos
        z=1
        while not self.isEmpty((x,y,z)):
            z+=1
        return(x,y,z)
    def buildBlock(self,pos):
        x,y,z=pos
        new=self.findHighestEmpty(pos)
        if new[2]<=z+1:
            self.addBlock(new)
```

```
    def delBlock(self,pos):
        blocks=self.findBlocks(pos)
        for block in blocks:
            block.removeNode()
    def delBlockFrom(self,pos):
        x,y,z=self.findHighestEmpty(pos)
        pos=x,y,z-1
        for block in self.findBlocks(pos):
            block.removeNode()
```

В этой части кода я создаю функций для нахождения блоков, постройки карты, создания блока, удаления блока и удаления всех блоков на определенной координате.

Мой код - hero.py

```
key_switch_camera='c'
key_switch_mode='l'
key_forward='w'
key_back='s'
key_left='a'
key_right='d'
key_up='o'
key_down='k'
key_turn_left='z'
key_turn_right='x'
key_build='q'
key_destroy='e'
class Hero():
    def __init__(self,pos,land):
        self.land=land
        self.mode=True
        self.hero=loader.loadModel('smiley')
        self.hero.setColor(1,0.5,0)
        self.hero.setScale(0.3)
        self.hero.setPos(pos)
        self.hero.reparentTo(render)
        self.cameraBind()
        self.accept_events()
```

```
    def cameraBind(self):
        base.disableMouse()
        base.camera.setH(180)
        base.camera.reparentTo(self.hero)
        base.camera.setPos(0,0,1.5)
        self.cameraOn=True
    def cameraUp(self):
        pos=self.hero.getPos()
        base.mouseInterfaceNode.setPos(
            -pos[0],-pos[1],-pos[2]-3)
        base.camera.reparentTo(render)
        base.enableMouse()
        self.cameraOn=False
```

В этой части кода я задаю кнопки, создаю класс в котором делаю инит и добавляю функций для изменения положения камеры.

Мой код - hero.py

```
    def changeView(self):
        if self.cameraOn:
            self.cameraUp()
        else:
            self.cameraBind()
    def turn_left(self):
        self.hero.setH((self.hero.getH()+5)%360)
    def turn_right(self):
        self.hero.setH((self.hero.getH()-5)%360)
    def look_at(self,angle):
        x_from=round(self.hero.getX())
        y_from=round(self.hero.getY())
        z_from=round(self.hero.getZ())
        dx,dy=self.check_dir(angle)
        x_to=x_from+dx
        y_to=y_from+dy
        return x_to,y_to,z_from
    def just_move(self,angle):
        pos=self.look_at(angle)
        self.hero.setPos(pos)
    def move_to(self,angle):
        if self.mode:
            self.just_move(angle)
```

В этой части кода я создаю функций для изменения режима игры, т.е положения камеры, поворота налево и направо, установки угла камеры и движения.

Мой код - hero.py

```
    def check_dir(self,angle):
        if angle>=0 and angle<=20:
            return(0,-1)
        elif angle<=65:
            return(1,-1)
        elif angle<=110:
            return(1,0)
        elif angle<=155:
            return(1,1)
        elif angle<=200:
            return(0,1)
        elif angle<=245:
            return(-1,1)
        elif angle<=290:
            return(-1,0)
        elif angle<=355:
            return(-1,-1)
        else:
            return(0,-1)
    def forward(self):
        angle=(self.hero.getH())%360
        self.move_to(angle)
```

```
    def back(self):
        angle=(self.hero.getH()+180)%360
        self.move_to(angle)
    def left(self):
        angle=(self.hero.getH()+90)%360
        self.move_to(angle)
    def right(self):
        angle=(self.hero.getH()+270)%360
        self.move_to(angle)
```

В этой части кода я создаю функций для определения направления и движения в разные стороны.

Мой код - hero.py

```
    def accept_events(self):
        base.accept(key_turn_left,self.turn_left)
        base.accept(key_turn_left+'-repeat',
                    self.turn_left)
        base.accept(key_turn_right,
                    self.turn_right)
        base.accept(key_turn_right+'-repeat',
                    self.turn_right)
        base.accept(key_forward,self.forward)
        base.accept(key_forward+'-repeat',
                    self.forward)
        base.accept(key_back,self.back)
        base.accept(key_back+'-repeat',self.back)
        base.accept(key_left,self.left)
        base.accept(key_left+'-repeat',self.left)
        base.accept(key_right,self.right)
        base.accept(key_right+'-repeat',
                    self.right)
        base.accept(key_switch_camera,
                    self.changeView)
        base.accept(key_switch_mode,
                    self.changeMode)
```

```
        base.accept(key_up,self.up)
        base.accept(key_up+'-repeat',self.up)
        base.accept(key_down,self.down)
        base.accept(key_down+'-repeat',self.down)
        base.accept(key_build,self.build)
        base.accept(key_destroy,self.destroy)
    def changeMode(self):
        if self.mode:
            self.mode=False
        else:
            self.mode=True
```

В этой части кода я создаю функций для обработки событий, т. е нажатий на клавиши, и вызова остальных функций и изменения режима игры.

Мой код - hero.py

```
    def try_move(self,angle):
        pos=self.look_at(angle)
        if self.land.idEmpty(pos):
            pos=self.land.findHighestEmpty(pos)
            self.hero.setPos(pos)
        else:
            pos=pos[0],pos[1],pos[2]+1
            if self.land.isEmpty(pos):
                self.hero.setPos(pos)
    def up(self):
        if self.mode:
            self.hero.setZ(self.hero.getZ()+1)
    def down(self):
        if self.mode and self.hero.getZ()>1:
            self.hero.setZ(self.hero.getZ()-1)
    def build(self):
        angle=self.hero.getH()%180
        pos=self.look_at(angle)
        if self.mode:
            self.land.addBlock(pos)
        else:
            self.land.buildBlock(pos)
```

```
    def destroy(self):
        angle=self.hero.getH()%180
        pos=self.look_at(angle)
        if self.mode:
            self.land.delBlock(pos)
```

В этой части кода я создаю функций для движения, движения вверх или вниз и строительства или уничтожения блоков.