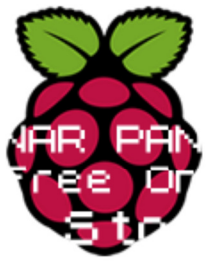


LUNAR PANDA PROGRAMMING TUTORIAL

WRITTEN BY DEAN EDIS ON 16 APRIL 2013.

Download Source Code (Python)

INTRODUCTION

Making changes to a Python game can be very satisfying, and does not need to be difficult. This is especially the case with Lunar Panda. With a little experience changes can be made which completely change the look and feel of the game.

GAME RESOURCES

Some of the simplest changes you can make to the game do not even require looking at the code.

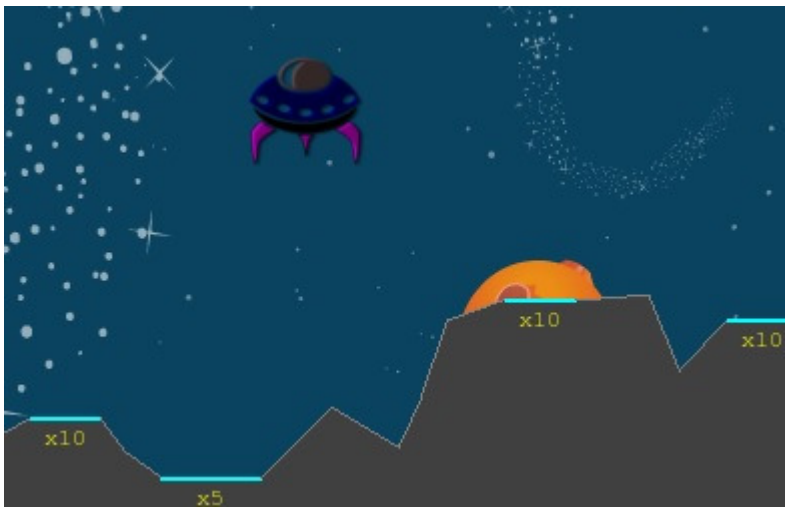
If you find the folder containing your game you will notice a file called LunarPanda.py. This is the Python code which defines the game and how it behaves. Lets ignore this for now and instead look at the folder called 'Resources'. The 'Resources' folder contains a series of files used by the code. The .ogg files contain the in-game music (An 'ogg' file is quite similar to an MP3. See http://en.wikipedia.org/wiki/Ogg), and the graphics files which have a .png extension.

RESOURCES - MR PANDA

To keep things simple we have three different frames of animation for Mr Panda when he is flying:
- panda0.png - Floating through space
- panda1.png - Thrust frame A
- panda2.png - Thrust frame B

To keep the code simple to follow we have made all of these images have the same dimension.

You are welcome to change or replace these files. To do so double-click to open them in your favourite image editing utility. How about drawing an alien, a spaceship, or just superimposing your face on Mr Panda?



Here's some other images you might want to change:
- background.png - The space background used during the game
- pandaLand*.png - Images used when Mr Panda gets close to landing.
- titleimage.jpg - The main title screen

CHANGING GAME DIFFICULTY

There are several factors which govern how difficult the game is to play. We've chosen default values which we feel make landing a challenge, but not so difficult it can't be achieved (With a little practice!).

LANDING AREAS - QUANTITY

To change the number of landing areas have a look in the game code (LunarPanda.py) and search for:
def _createSurfacePoints(self)

Look a few lines down and you'll see an 'IDEA' comment. (We've added comments like these throughout the code to help show where you might like to make changes.)

Try changing the 3 and 2 in this line and running the game again:
while numberofFlats != 3 or numberofEasyFlats != 2

To make the game harder to play try increasing numberofFlats and/or decreasing numberofEasyFlats

LANDING AREAS - SIZE

Now search for the following lines of code:
_flatWidth = 35
_easyFlatWidth = 50

These values define how wide (in pixels) the different types of landing 'flat' are. Feel free to change them, but remember to make them big enough for Mr Panda to actually land on!

You could also try making the values based on a random number using Python's random.randint(a, b) function. See http://docs.python.org/2/library/random.html#random.randint.
For example:
_easyFlatWidth = random.randint(50, 70)

...which makes the 'easy' flat between 50 and 70 pixels wide.

LANDING SPEED

When Mr Panda touches the ground we have to perform several checks to see if he has landed safely. Did he land feet-first? Were both feet on the landing area? Was he coming down too fast?

Try changing this line of code:
maxLandingVelocity = -0.20

Increasing this number will make it easier; it will increase the speed at which Mr Panda can land safely. Note: The minus sign is important. A positive velocity means Mr Panda is moving upwards, away from the ground.

ADDING A HIGH SCORE

At the end of each level Mr Panda will be given score, partly based on how much fuel he has left. If you'd like to change the code so it keeps a high score, just follow the steps below.

STEP 1

```
background=pygame.image.load("Resources/background.png").convert()
titleImage = pygame.image.load("Resources/titleimage.jpg").convert()
uiFont = pygame.font.SysFont("monospace", 12)

background_image_height = Surface.get_height(background)
screenScroll = 0

highScore = 0
```

Add the highlighted line near the top of the LunarPanda.py code. To start off with this value will be zero, but after each successful landing we'll update it so we can remember the highest score,

STEP 2

```
def show_score(gameStats):
    global highScore

    """ Draw the score in the top-left corner of the screen. """
    label = uiFont.render("Score: " + str(gameStats.displayScore), True, (255, 255, 0))
    screen.blit(label, (10, 55))

    label = uiFont.render("High score: " + str(highScore), True, (255, 255, 0))
    screen.blit(label, (10, 70))
```

We need to change the show_score function so the player can see what their current high score is as they are playing. This change draws the 'High score:' text at the top of the screen, just under where the 'Score:' label is positioned. Remember to add the 'global highScore' line at the top of the function!

STEP 3

```
def pandaLanded(mrPanda, flatScore, lunarSurface, gameStats):
    """
    This is called when Mr Panda has made a safe landing.
    The score is added up as the fuel is counted down.

    IDEA: You may want to take into account how long it took Mr Panda
    to land.  A quicker time could mean more score!
    """
    global highScore

    levelScore = int(mrPanda.fuel * flatScore)
    gameStats.totalScore += levelScore
    font = pygame.font.Font("resources/retro.ttf", 45)

    pygame.mixer.music.load('resources/beep.ogg')
    pygame.mixer.music.set_volume(0.5)

    fuelRemaining = int(mrPanda.fuel) + 1
    while fuelRemaining > 0:
        fuelStep = min(33, fuelRemaining)
        if fuelStep == 33:
            pygame.mixer.music.play()
        else:
            pygame.mixer.music.stop()

        fuelRemaining -= fuelStep
        gameStats.displayScore += fuelStep

        redrawScreen(mrPanda, lunarSurface)
        show_stats(mrPanda)

        show_score(gameStats)
        blitTextHorizontalCentered(font, 'Level Cleared', 225, 365, 5
        label = font.render("Remaining Fuel: " + str(fuelRemaining),
        screen.blit(label, (230, 280))

        pygame.display.update()
        pygame.time.wait(75)

    pygame.time.wait(3000)

    if highScore < gameStats.displayScore:
        highScore = gameStats.displayScore
```

And finally, this change will make sure our new highScore variable is kept up to date. This function is called when Mr Panda has made a successful landing.

We hope you enjoy changing our code to add your own tweaks to Lunar Panda. Please feel free to share updates and screenshots with us to let us know how you get on!

< Prev