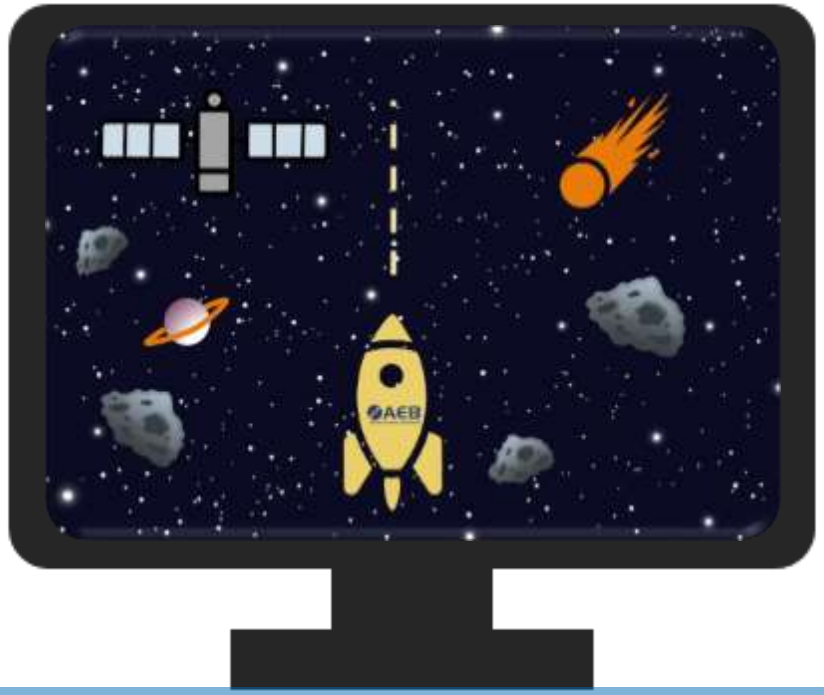


Minicurso de Programação de Jogos em Python

Aula 5
Pontuação e Trilha Sonora

Acesse

aebescolavirtual.aeb.gov.br/

Professora: Brenda Micaelle Santos Figueiredo- Aluna de Graduação do ITA

PROGRAMA ESPACIAL BRASILEIRO

AEB escola

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

Tópicos

- Exibição da pontuação
- Trilha Sonora
- Inclusão do som da batida nas margens

Fonte: Dribbble (2018).

Lembrando ...

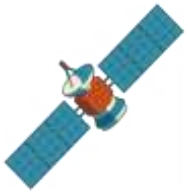
- Avançamos bem na criação do Jogo Fuga Espacial.
- Criamos elementos fundamentais do jogo, como:

- Plano de Fundo ou Background

- Player

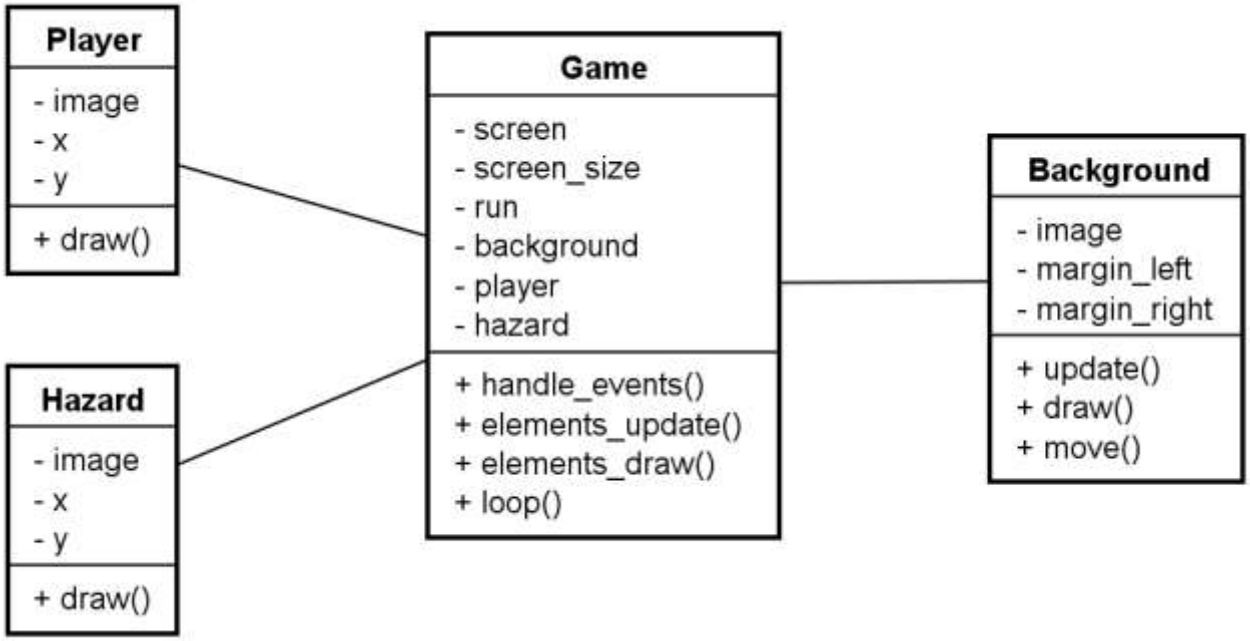
- Hazards




Lembrando ...

- O Diagrama de Classes já contempla as todas classes inicialmente planejadas:



- Vamos seguir complementando o jogo para que ele fique mais interessante ainda ...

Exibição da pontuação

PROGRAMA ESPACIAL BRASILEIRO

AEB escola

AEB

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

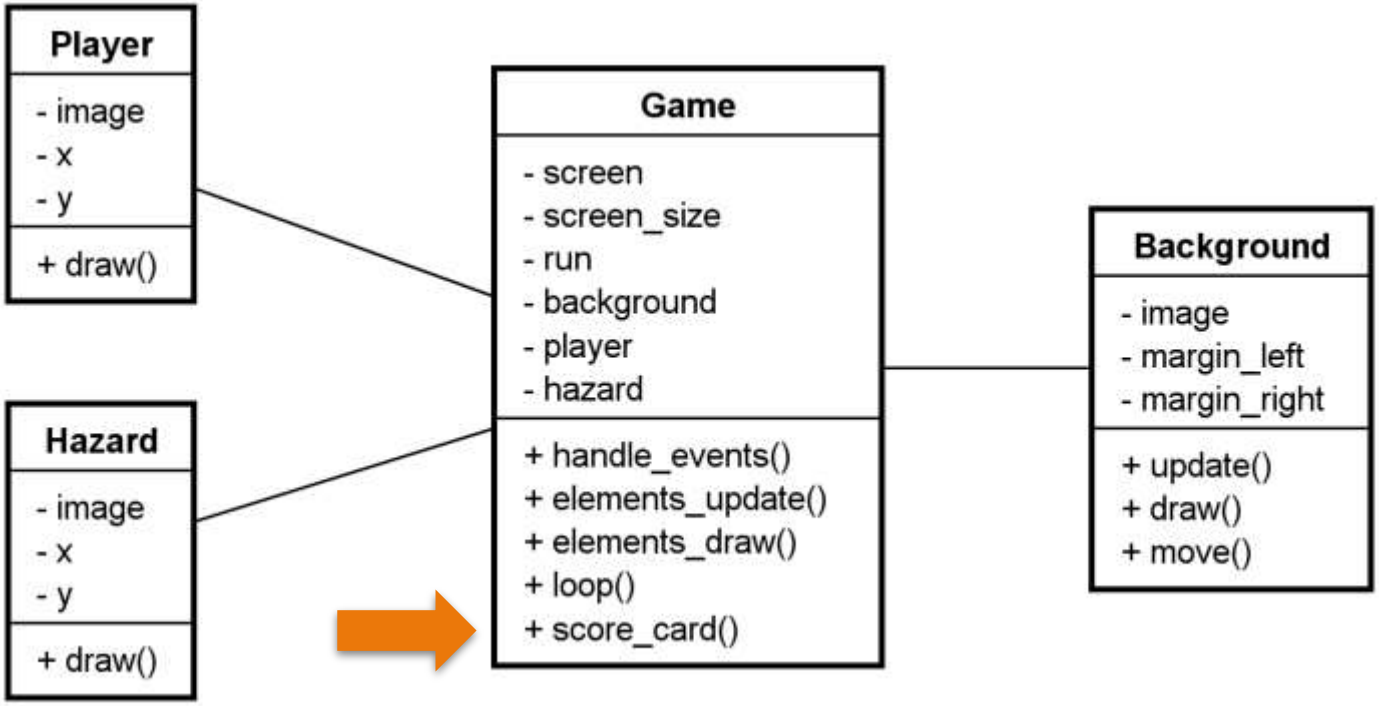
GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

Exibição da pontuação

- Essa parte é a mais tranquila e a mais legal :) Queremos ver na tela do nosso jogo a contagem de quantos hazards nosso foguete conseguiu desviar e a nossa pontuação conforme vamos avançando.
- Para isso, vamos primeiro construir uma função que renderiza (mostra na tela) a contagem e a pontuação com os valores atuais.

Atualizando o Diagrama de Classes

- Vamos inserir a função *score_card()* na classe **Game**



Função *score_card()*

 # Definição da função

```
178     # Informa a quantidade de hazard que passaram e a Pontuação
179     def score_card(self, screen, h_passou, score):
180         font = pygame.font.SysFont(None, 35)
181         passou = font.render("Passou: " + str(h_passou), True, (255, 255, 128))
182         score = font.render("Score: " + str(score), True, (253, 231, 32))
183         screen.blit(passou, (0, 50))
184         screen.blit(score, (0, 100))
185     #score_card()
```

Como usar fontes no pygame

- Há dois jeitos:

```
# usando pygame.font.Font()
my_font = pygame.font.Font("Fonts/Fonte4.ttf", 100)
```

```
# usando pygame.font.SysFont()
font = pygame.font.SysFont(None, 35)
```

```
# vamos usar esse!
```

Função *score_card()*

 # Definição da função

```
178     # Informa a quantidade de hazard que passaram e a Pontuação
179     def score_card(self, screen, h_passou, score):
180         font = pygame.font.SysFont(None, 35)
181         passou = font.render("Passou: " + str(h_passou), True, (255, 255, 128))
182         score = font.render("Score: " + str(score), True, (253, 231, 32))
183         screen.blit(passou, (0, 50))
184         screen.blit(score, (0, 100))
185     #score_card()
```

 # Define e renderiza os textos

 # Sobrepõe os textos na tela

Declarando variáveis

```
187     def loop(self):
188         """
189         Esta função contém o laço principal
190         """
191         score = 0
192         h_passou = 0
```

Inicializando variáveis

Acréscimo nos valores

```
283         # definindo onde hazard vai aparecer, recomeçando a posição do obstaculo e da faixa
284         if h_y > self.height:
285             h_y = 0 - h_height
286             h_x = random.randrange(125, 650 - h_height)
287             hzrd = random.randint(0, 4)
288             # determinando quantos hazard passaram e a pontuação
289             h_passou = h_passou + 1
290             score = h_passou * 10
```

 # Faz o acréscimo dos valores

Exibição da pontuação

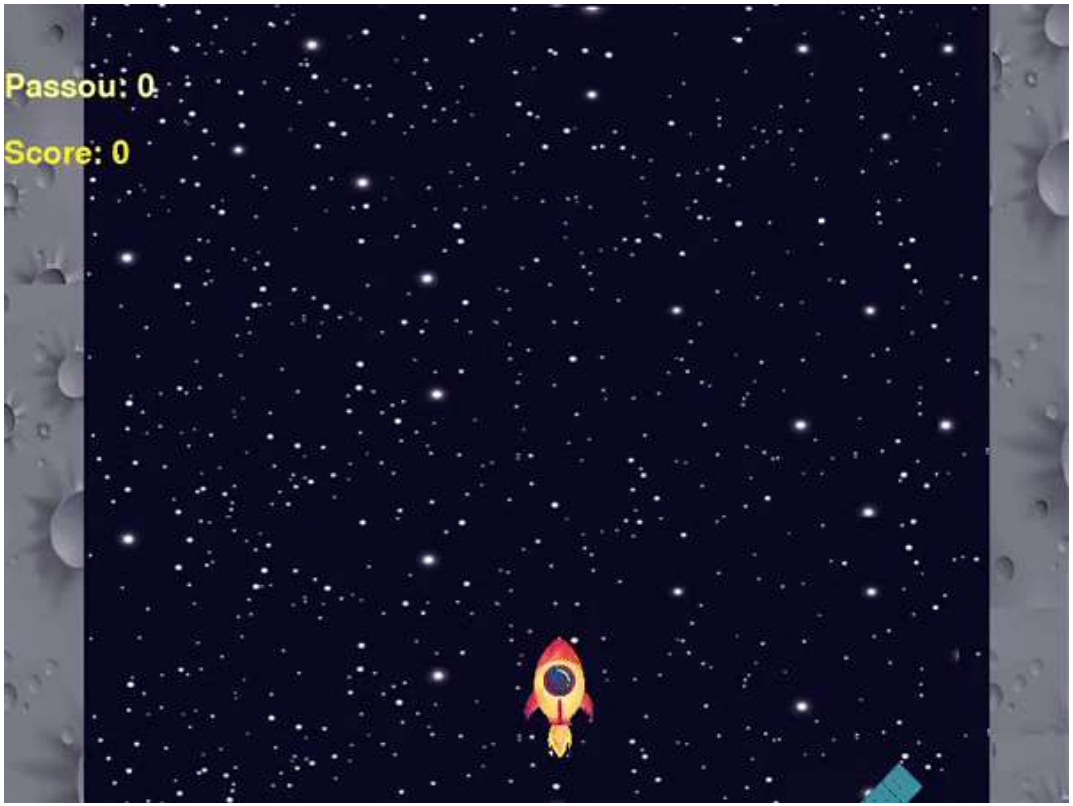
```
263             # Desenha o Player
264             self.player.draw(self.screen, x, y)
265
266             # Mostrar score
267             self.score_card(self.screen, h_passou, score)
268
```

 # Chama a função score_card

Exibição da pontuação

- Agora temos um score e um contador completamente funcionais! :D Vamos rodar o código e ver como ficou?

Executando o código



Tela do jogo mostrando

a Pontuação do Jogador

Trilha Sonora

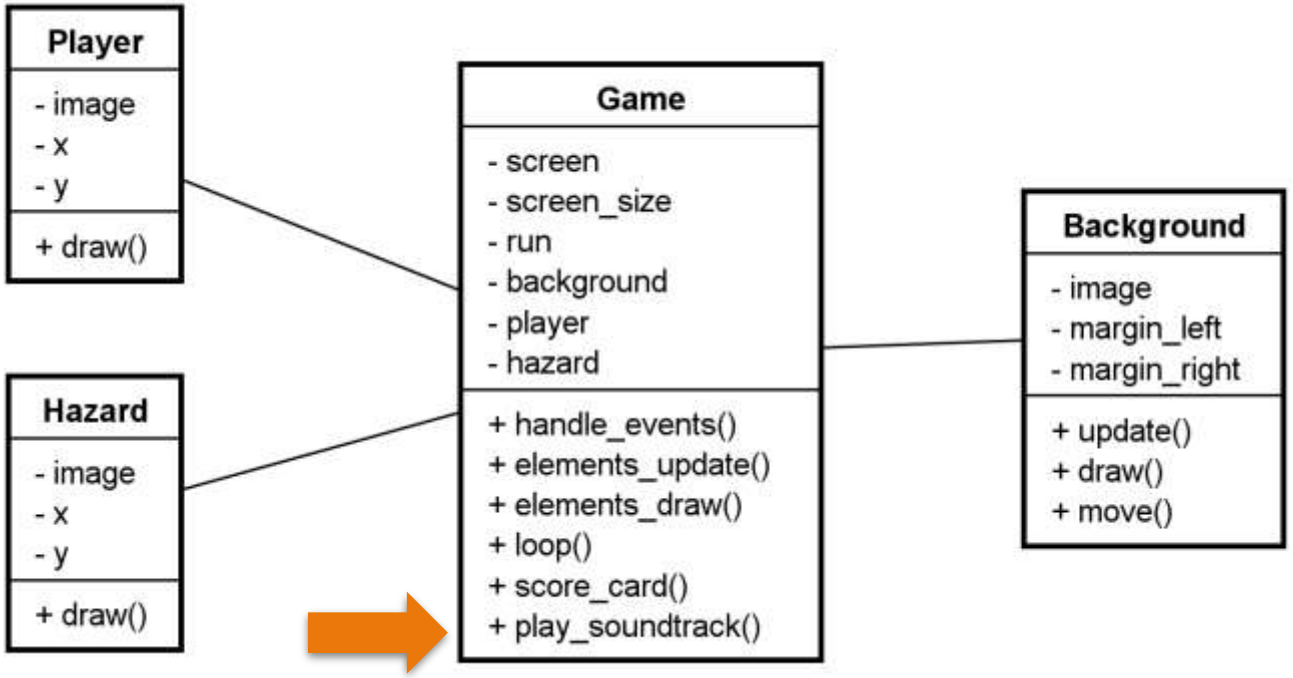
PROGRAMA ESPACIAL BRASILEIRO
AEB escola
AEB
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO
GOVERNO FEDERAL
BRASIL
UNIÃO E RECONSTRUÇÃO

Inclusão da trilha sonora

- Conseguimos fazer com que a nossa pontuação aparecesse, mas jogar sem música é meio chato né?
- A música, tanto nos filmes quanto nos jogos, possui um papel fundamental no envolvimento do espectador ou do jogador com o que está sendo passado, pois causa emoções e cria memórias afetivas em nós;
- Assim, para o nosso jogo ficar mais legal ainda, vamos colocar a trilha sonora?

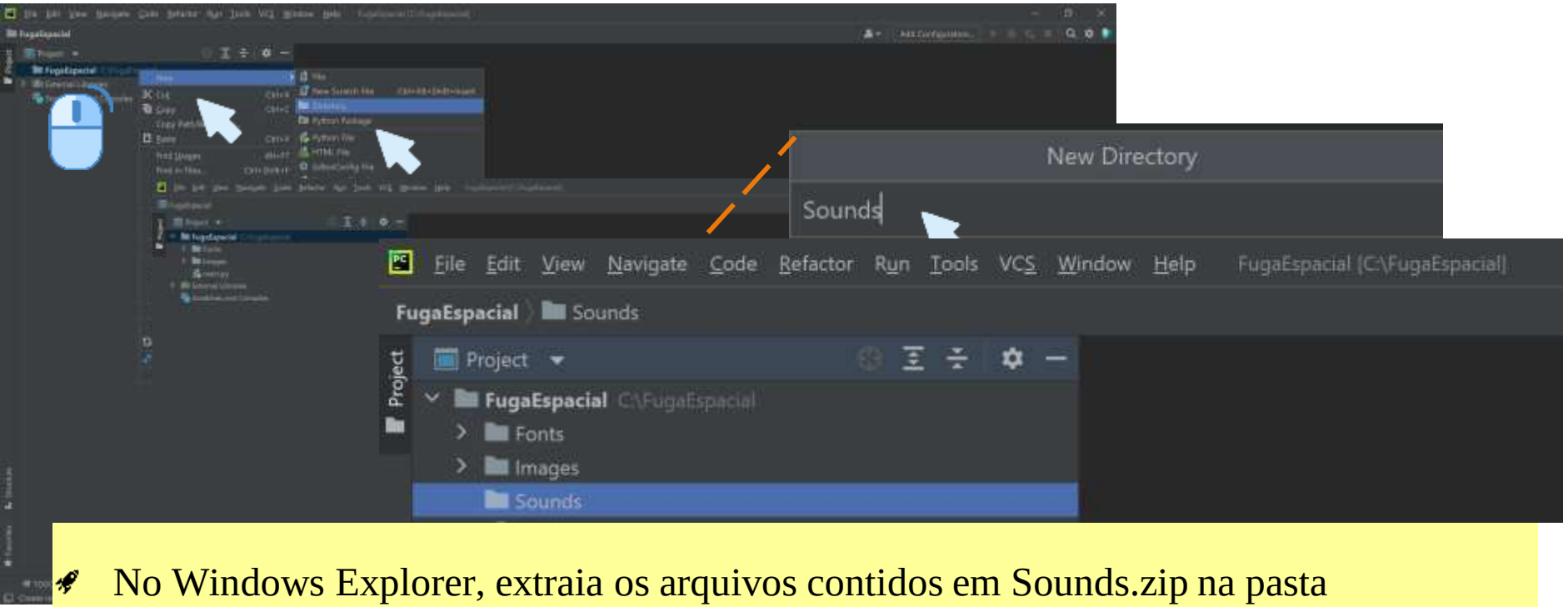
Atualizando o Diagrama de Classes

- Vamos inserir a função *play_soundtrack()* na classe **Game**



Criando pasta Images

- Antes de tudo, precisamos criar da pasta (diretório) para os sons do jogo: *Sounds*



- No Windows Explorer, extraia os arquivos contidos em Sounds.zip na pasta

  C:\FugaEspacial\Sounds

Inclusão da trilha sonora

```
6
7   import pygame
8   import time # uso da função-membro time.sleep(...) in loop
9   import random # uso da função-membro random.randrange (...) em loop
10  import os # usa função-membro os.path.isfile(...) em play_soundtrack
11
```

 # Importação da biblioteca *os*

Função *play_soundtrack()*

- Vamos inserir a função *play_soundtrack()* na classe **Game**

```
188     def play_soundtrack(self):
189         # Inclui trilha sonora
190         if os.path.isfile('Sounds/song.wav'):
191             pygame.mixer.music.load('Sounds/song.wav')
192             pygame.mixer.music.set_volume(0.5)
193             pygame.mixer.music.play(loops=-1)  # set loops to -1 to loop the music indefinitely
194         else:
195             print("Sounds/song.mp3 not found... ignoring", file=sys.stderr)
196     #play_soundtrack()
197
198     def loop(self):
```

 # Definição da função

 # Confere se o arquivo de som existe

 # Carrega o arquivo, ajusta o volume do som e coloca para rodar infinitamente

 # Mostra mensagem de erro caso o arquivo não exista

Chamada da função

```
225         # Criar o Plano de fundo
226         self.background = Background()
227
228         # Incluir trilha sonora
229         self.play_soundtrack()
```

 # Chama a função no início do jogo

Inclusão da trilha sonora

- Agora temos um jogo com música de fundo! Vamos rodar o código e ver como ficou?

Fonte: GIPHY.

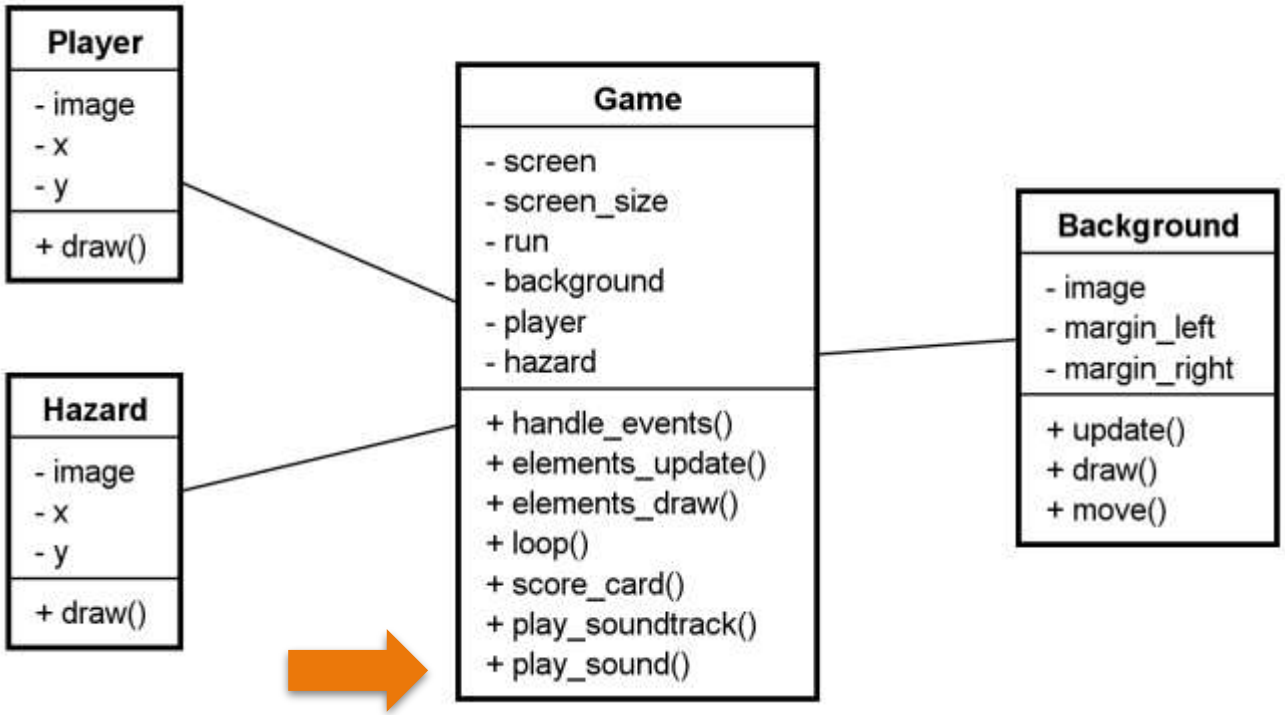
Inclusão do som das batidas nas margens

Inclusão do som das batidas

- O nosso jogo está ficando cada vez melhor, mas ainda falta um pouco de sonoplastia para dar um pouco mais de emoção! Vamos adicionar o som de batida quando o nosso foguete encosta nas margens da tela :D

Atualizando o Diagrama de Classes

- Vamos agora inserir a função *play_sound()* na classe **Game**



Função *play_sound(self, sound)*

 # Definição da função

```
197
198     def play_sound(self, sound):
199         # Som
200         if os.path.isfile(sound):
201             pygame.mixer.music.load(sound)
202             pygame.mixer.music.set_volume(0.5)
203             pygame.mixer.music.play()
204         else:
205             print("Sound file not found... ignoring", file=sys.stderr)
206     #play_sound()
207
208     def loop(self):
209         """
```

 # Confere se o arquivo de som existe

 # Carrega o arquivo, ajusta o volume do som
#e coloca para tocar uma vez

 # Mostra mensagem de erro

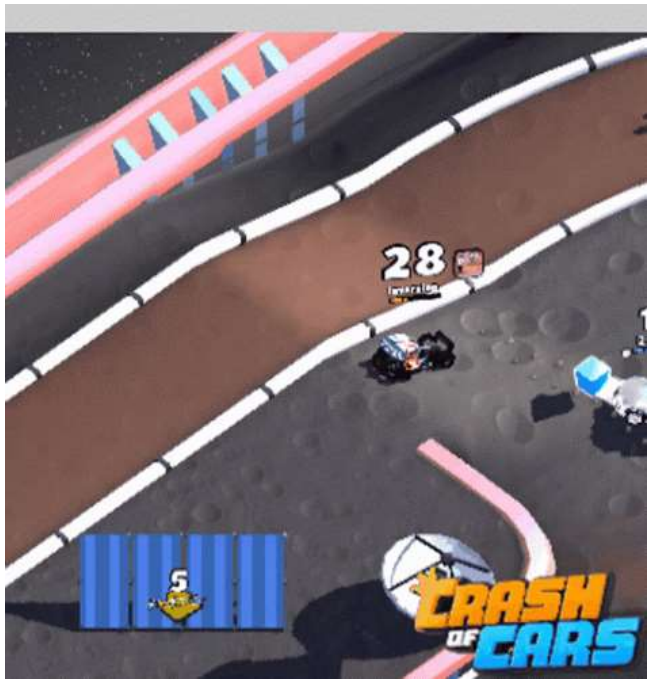
Chamada da função

```
293         # Restrições do movimento do Player
294         # Se o Player bate na lateral não é Game Over
295         if x > 760 - 92 or x < 40 + 5:
296
297             # Som da colisão nas margens
298             self.play_sound('Sounds/jump2.wav')
299
```

 # Chama a função em caso de colisão

Inclusão do som das batidas

- Agora podemos ouvir o crash quando a nossa nave bate na margem! Vamos rodar o código e ver como ficou?



Fonte: Tenor.

Otimizando o código

PROGRAMA ESPACIAL BRASILEIRO

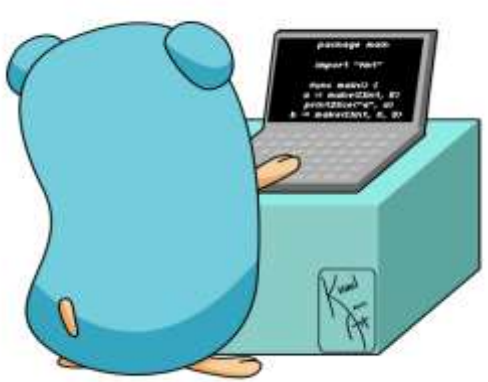
AEB escola

AEB

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

Otimizando o código



- Em geral, quando estamos codificando, acabamos por encontrar outras formas de fazer com que o programa funcione de maneira otimizada

- Assim, é bom ter em mente que fazer ajustes no código constantemente é uma prática comum na programação de computadores

Otimizando o código

- Alterações

1. Eliminação dos atributos DIREITA e ESQUERDA na classe Game
   - Altera a classe Game, função-membro handle_events()

2. Criação da Classe Soundtrack
   - Inclusão do módulo sys (utilizado na classe Soundtrack)
   - Altera a classe Game, função-membro loop()

Otimizando o código

1. Eliminação dos atributos DIREITA e ESQUERDA na classe Game

```
132  class Game:
133      screen = None
134      screen_size = None
135      width = 800
136      height = 600
137      run = True
138      background = None
139      player = None
140      hazard = []
141      soundtrack = None
142      render_text_bateulateral = None
143      render_text_perdeu = None
144
145      # movimento do Player
146      #DIREITA = pygame.K_RIGHT
147      #ESQUERDA = pygame.K_LEFT
148      mudar_x = 0.0
149
```

 # remover atributos

Otimizando o código

1. Eliminação dos atributos DIREITA e ESQUERDA na classe Game

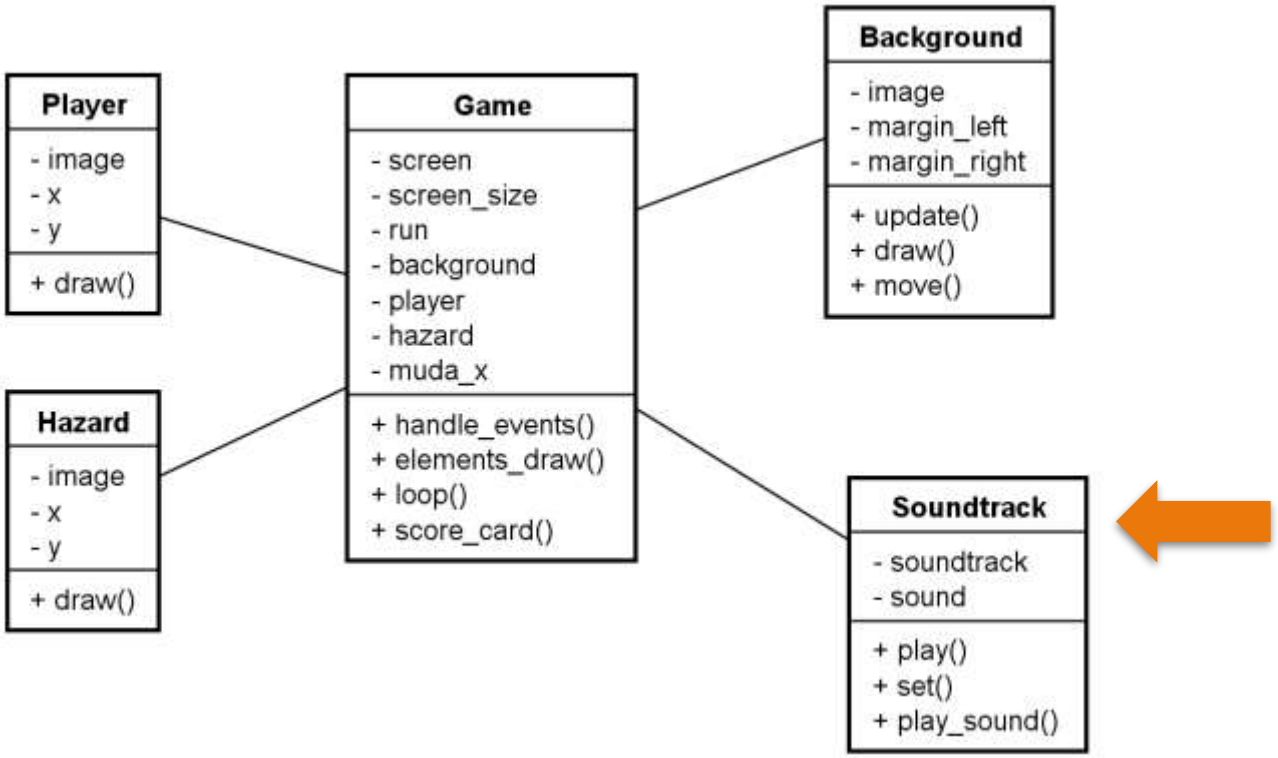
- Altera a classe Game, função-membro handle_events()

```
    def handle_events(self):
        """
        Trata o evento e toma a ação necessária.
        """
        for event in pygame.event.get():

            if event.type == pygame.QUIT:
                self.run = False

            # se clicar em qualquer tecla, entra no if
            if event.type == pygame.KEYDOWN:
                # se clicar na seta da esquerda, anda 3 para a esquerda no eixo x
                if event.key == pygame.K_LEFT:
                    self.mudar_x = -3
                # se clicar na seta da direita, anda 3 para a direi
                if event.key == pygame.K_RIGHT:
                    self.mudar_x = 3

            # se soltar qualquer tecla, não faz nada
            if event.type == pygame.KEYUP:
                if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
                    self.mudar_x = 0
    # handle_events()
```

1. **substituir ESQUERDA por pygame.K_LEFT**
2. **substituir DIREITA por pygame.K_RIGHT**

Atualizando o Diagrama de Classes

- Inicialmente inseriremos a classe **Soundtrack** no Diagrama de Classes



Otimizando o código

2. Criação da

Classe Soundtrack

```
class Soundtrack:
    soundtrack = None
    sound = None

    def __init__(self, soundtrack):
        if os.path.isfile(soundtrack):
            self.soundtrack = soundtrack
        else:
            print(soundtrack + " not found... ignoring", file=sys.stderr)
    # __init__()

    def play(self):
        pygame.mixer.music.load(self.soundtrack)
        pygame.mixer.music.set_volume(0.5)
        pygame.mixer.music.play(loops=-1)  # set loops to -1 to loop the music indefinitely
    # play()

    def set(self, soundtrack):
        if os.path.isfile(soundtrack):
            self.soundtrack = soundtrack
        else:
            print(soundtrack + " not found... ignoring", file=sys.stderr)
    # set()

    def play_sound(self, sound):
        if os.path.isfile(sound):
            self.sound = sound
            pygame.mixer.music.load(self.sound)
            pygame.mixer.music.set_volume(0.5)
            pygame.mixer.music.play()
        else:
            print(sound + " file not found... ignoring", file=sys.stderr)
    # play_sound()
# Soundtrack:
```

Otimizando o código

- Função __init__()

2. Criação da

Classe Soundtrack

```
107  class Soundtrack:
108      soundtrack = None
109      sound = None
110
111      def __init__(self, soundtrack):
112          if os.path.isfile(soundtrack):
113              self.soundtrack = soundtrack
114          else:
115              print(soundtrack + " not found... ignoring", file=sys.stderr)
116      # __init__()
```

Otimizando o código

2. Criação da

Classe Soundtrack

- Método ou função-membro play()

```
117
118     def play(self):
119         pygame.mixer.music.load(self.soundtrack)
120         pygame.mixer.music.set_volume(0.5)
121         pygame.mixer.music.play(loops=-1)  # set loops to -1 to loop the music indefinitely
122     # play()
```

Otimizando o código

- Método ou função-membro set()

2. Criação da

Classe Soundtrack

```
124     def set(self, soundtrack):
125         if os.path.isfile(soundtrack):
126             self.soundtrack = soundtrack
127         else:
128             print(soundtrack + " not found... ignoring", file=sys.stderr)
129     # set()
```

Otimizando o código

2. Criação da

Classe Soundtrack

- Método ou função-membro play_sound()

```
    def play_sound(self, sound):
        if os.path.isfile(sound):
            self.sound = sound
            pygame.mixer.music.load(self.sound)
            pygame.mixer.music.set_volume(0.5)
            pygame.mixer.music.play()
        else:
            print(sound + " file not found... ignoring", file=sys.stderr)
    # play_sound()
# Soundtrack:

class Game:
    screen = None
```

Otimizando o código

2. Criação da Classe Soundtrack

- Inclusão do módulo sys (utilizado na classe Soundtrack)

```
10      import os # usa função-membro os.path.isfile(...) em play_soundtrack
11      import sys # classe Soundtrack
```

Otimizando o código

2. Criação da Classe Soundtrack

- Altera a classe Game, função-membro loop()
  - Instancia o objeto da classe Soundtrack

```
        # Criar os Hazards
        self.hazard.append(Hazard("Images/satelite.png", h_x, h_y))
        self.hazard.append(Hazard("Images/nave.png", h_x, h_y))
        self.hazard.append(Hazard("Images/cometaVermelho.png", h_x, h_y))
        self.hazard.append(Hazard("Images/meteoros.png", h_x, h_y))
        self.hazard.append(Hazard("Images/buracoNegro.png", h_x, h_y))

        # Criar trilha sonora
        self.soundtrack = Soundtrack('Sounds/song.wav')
        self.soundtrack.play()
```

 # A trilha sonora é instanciada

Otimizando o código

2. Criação da Classe Soundtrack

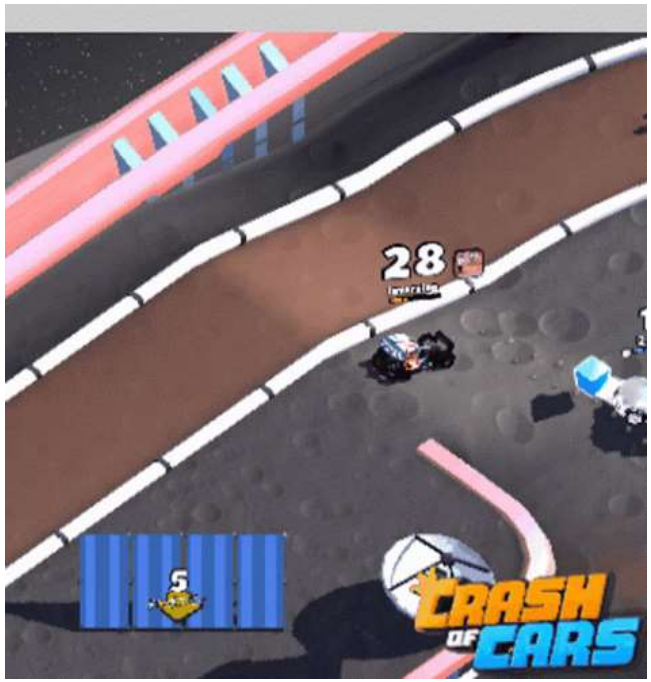
- Altera a classe Game, função-membro loop()
  - Chamada da função-membro self.soundtrack.play_sound()

```
299             # Restrições do movimento do Player
300             # Se o Player bate na lateral não é Game Over
301             if x > 760 - 92 or x < 40 + 5:
302
303                 # Som da colisão nas margens
304                 self.soundtrack.play_sound('Sounds/jump2.wav')
305
306                 # Exibe mensagem
307                 self.screen.blit(self.render_text_bateulateral, (80, 200))
308                 pygame.display.update()  # atualizar a tela
309                 time.sleep(3)
310                 self.loop()
311
```

 # Chamada da função em caso de colisão com a margem

Inclusão do som das batidas

- Vamos executar o código?



Fonte: Tenor.

Atividades práticas

Mude o código para melhor entendê-lo

PROGRAMA ESPACIAL BRASILEIRO

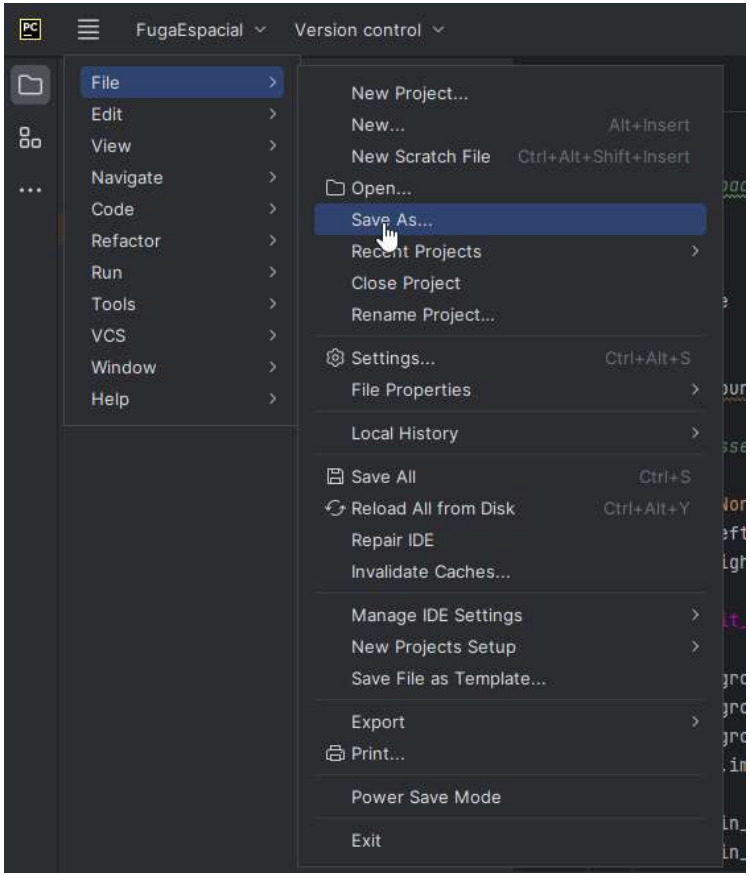
AEB escola

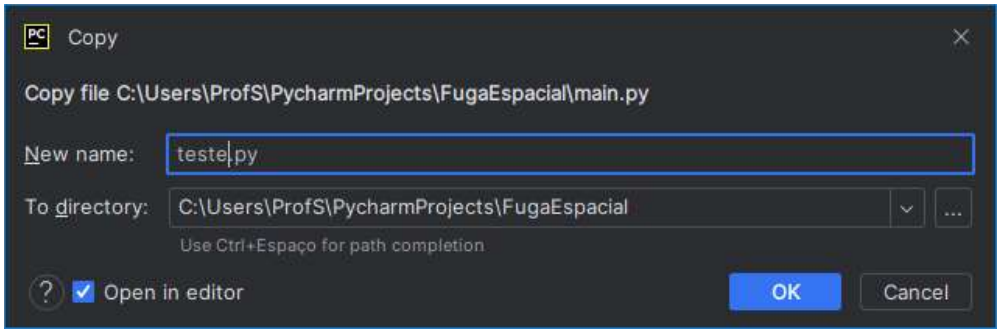
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO

ANTES DE COMEÇAR

Salve o código como teste.py para não perder o código do jogo já desenvolvido





Utilizando a classe Soundtrack

Modifique a trilha sonora do jogo após a emissão do som de colisão do jogador com a margem na função `loop` da classe `Game` e execute o jogo.

Dica: use a função-membro play() da classe Soundtrack

Observe o que acontece após a colisão. Você pode bater na margem várias vezes nesta atividade.

Pergunta: O que houve com a trilha sonora?

Utilizando a classe Soundtrack

Insira o som de colisão do jogador com a margem do cenário do jogo (Sounds/jump2.wav) após a atualização da pontuação na função `loop` da classe `Game` e execute o jogo.

Observe a sonorização durante algumas passagens do player por ameaças.

Pergunta: O que acontece com a trilha sonora a cada passagem por uma ameaça?

Fonte: Imagem de catalyststuff no Freepik

Obrigada

Até a próxima Aula!

PROGRAMA ESPACIAL BRASILEIRO

AEB escola

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÃO

GOVERNO FEDERAL BRASIL UNIÃO E RECONSTRUÇÃO