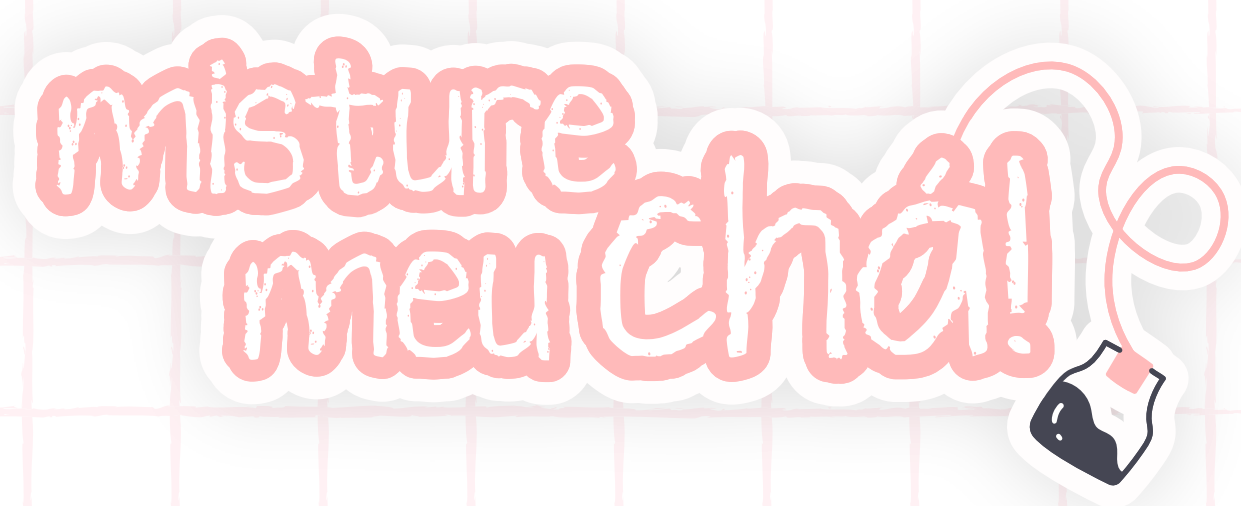


Trabalho final da disciplina de P00 I  
UFSC - Florianópolis  
2023.01







Pedro Henrique Gimenez - 23102766


# sobre o jogo

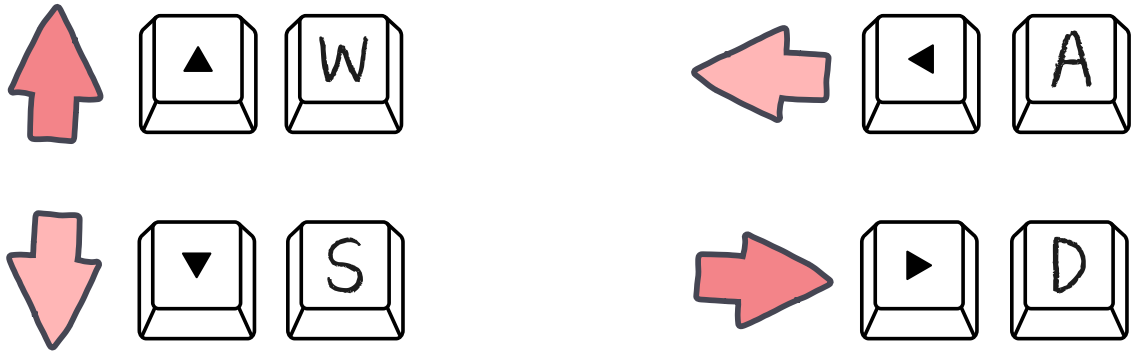
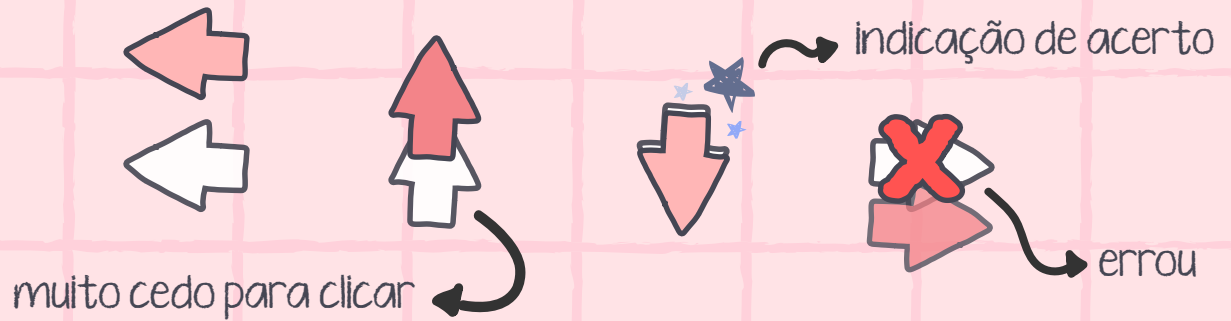
O jogo **misture meu chá!** foi inspirado em dois jogos clássicos da infância do desenvolvedor: "Guitar Hero" e "Club Penguin". O objetivo do jogo é seguir o ritmo das setas que descem pela tela e pressioná-las no momento correto para misturar o chá da xícara e se manter vivo!

## Regras

- O gameplay consiste em setas que descem em ordem aleatória. Quanto maior o tempo de jogo, maior será a velocidade e menor será o intervalo entre novas setas.
- No canto inferior direito, há 3 corações . Eles indicam as vidas do jogador. A cada seta que não foi apertada no momento certo, uma vida é perdida  e a xícara fica mais triste. Ao errar 3 vezes, o jogo acaba.
- Se o jogador acerta o timing da seta, aparece a indicação  , mostrando que a seta foi pontuada. Caso contrário, a indicação de erro  aparece, mostrando que uma vida foi perdida.
- A pontuação é calculada pelo tempo que o jogador permanece vivo, o objetivo do jogo é ficar vivo pelo máximo de tempo possível

## Como jogar

- As setas brancas  no meio da tela indicam o gabarito de onde a seta deve ser pressionada para pontuar. Após 1 segundo de jogo, as setas começam a aparecer.
- O jogador deverá esperar que a seta correspondente desça até a seta gabarito, quando as duas estiverem sobrepostas, o jogador deverá apertar no teclado as teclas correspondentes:



# executando o jogo

Para executar o jogo, é necessário que o **usuário tenha Python instalado** na máquina.

- Com o python instalado, vá ao terminal do computador e escreva:

```
> cd <arraste aqui a pasta do jogo>
```

- Depois, instale os requirements (apenas se for a primeira vez iniciando o jogo):

```
> pip install -r requirements.txt
```

Obs: se estiver num Mac, digite pip3 ao invés de pip

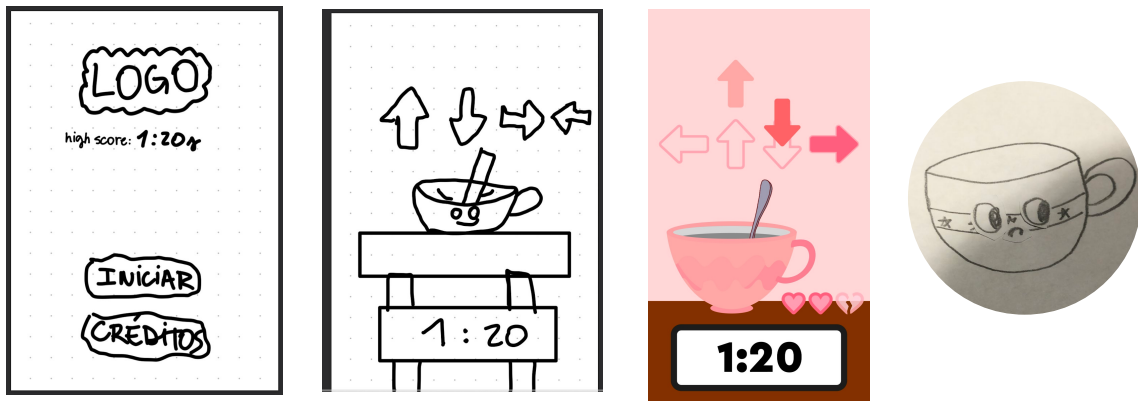
- Por último, inicie o jogo:

```
> python main.py
```

Obs: se estiver num Mac, digite python3 ao invés de python

# desenvolvimento

Antes de começar o desenvolvimento do código, montei frameworks de baixa fidelidade com a ideia na cabeça que tinha sobre o jogo.



E em seguida, comecei a trabalhar no Adobe Illustrator até chegar no resultado final. Depois exportei todos os assets e comecei a estudar o **Pygame** e **Programação Orientada a Objetos** para montar o jogo.

O jogo foi separado em um arquivo principal **main.py** em que roda o **loop principal do pygame**, utilizado para controlar as mudanças de estado e verificação se o jogo fechou ou se algum botão foi apertado. E depois separei em **2 pastas: source**, para os arquivos principais de funções importantes de funcionamento, e a **pasta entities**, para armazenar todos os objetos interativos (como xícara, setas e botões).

**Alguns conceitos de POO foram utilizados no código, como classes.** Sendo algumas importantes a **Classe de Variáveis** para poder chamar em todos os arquivos e fazer modificações de maneira global, e também a **Classe de entities** como a da **Xícara**.

```
misture-minha-xicara - configs.py

16 # Inicializacao de variaveis globais
17 class Var:
18     # codigos dos estados
19     state_menu = 0
20     state_credits = 1
21     state_gameplay = 2
22     state_gameover = 3
23     game_state = state_menu
24
25     scroll = -83 # velocidade do scroll do bg
26     timer = None # controla o timer do gameplay
27     highs = False # verifica se o highscore foi atingido
28     cor_mortos = 0 # numero de coracoes mortos
29     contador = 0 # contador de frames para gameplay
30     score = '' # score atual
31     score_max = '' # highscore
```

```
misture-minha-xicara - xicara.py

6 class Xicara:
7     def __init__(self, x, y, cara, Bool=True):
8         self.x = x
9         self.y = y
10        # inicializacao de imagens
11        self.xic_tras = xic_tras
12        self.colher = colher
13        self.xic_frente = xic_frente
14        self.cara = cara
15        self.Bool = Bool #booleana para dizer se a xícara chamada tem colher
16        self.angulo = 0
17        # controle da colher
18        self.cx = x+98
19        self.cy = y-38
20        self.velocidade = 2
21
22    def draw(self): #desenha separadamente a colher para poder movimentar e a xícara
23        SCREEN.blit(self.xic_tras, (self.x, self.y))
24        if self.Bool:
25            colher_rotacionada = transform.rotate(self.colher, self.angulo)
26            SCREEN.blit(colher_rotacionada, (self.cx, self.cy))
27        SCREEN.blit(self.xic_frente, (self.x, self.y))
28        SCREEN.blit(self.cara, (self.x+67, self.y+54))
```

Por exemplo, na Classe Botão:

```
misture-minha-xicara - botao.py

9 class Botao:
10     def __init__(self, nome, nome_f, x, y, estado, fx=10, fy=7):
11         self.nome = nome #imagem da frente
12         self.nome_f = nome_f #imagem da sombra
13         self.x = x
14         self.y = y
15         self.fx = fx #posicao da sombra
16         self.fy = fy
17         self.estado = estado #para qual estado que deve mudar
18         self.pressed = False
19
20     def draw(self):
21         rect = Rect(self.x, self.y, self.nome.get_width(), self.nome.get_height())
22         if rect.collidepoint(mouse.get_pos()): #animacao se passar o mouse por cima
23             pos_texto = (self.x + self.fx/3, self.y + self.fy/7)
24             if self.pressed: #animacao quando apertado
25                 pos_texto = (self.x + self.fx, self.y + self.fy)
26             else:
27                 self.pressed = False
28                 pos_texto = (self.x, self.y)
29
30         SCREEN.blit(self.nome_f, (self.x + self.fx, self.y + self.fy))
31         SCREEN.blit(self.nome, pos_texto)
```

```
33 def handle_event(self, event, Bool=False): #muda de estado se for apertado
34     rect = Rect(self.x, self.y, self.nome.get_width(), self.nome.get_height())
35     if event.type == MOUSEBUTTONDOWN:
36         if event.button == 1:
37             if rect.collidepoint(mouse.get_pos()):
38                 self.pressed = True
39     elif event.type == MOUSEBUTTONUP:
40         if event.button == 1:
41             if rect.collidepoint(mouse.get_pos()):
42                 if self.pressed:
43                     if Bool: #booleana para saber se o botão reseta tudo se for apertado
44                         Var.cor_mortos = 0
45                         xicara_gameplay.update_face(0)
46                         xicara_gameplay.angulo = 0
47                         Var.timer = None
48                         Var.highs = False
49                         xicara_gameplay.cx = 119
50                         xicara_gameplay.cy = 380
51                         Seto.lista_setas = [None for i in range(16)]
52                         Seto.lista_tipos = []
53                         Var.idx = 0
54                         Var.tempo = 1
55                         Var.contador = 0
56                         som_botao.set_volume(0.5)
57                         som_botao.play()
58                         Var.game_state = self.estado
59                         self.pressed = False
60
61 # definindo botões
62 b_iniciar = Botao(iniciar, iniciar_f, 97, 457, Var.state_gameplay)
63 b_creditos = Botao(bcreditos, bcreditos_f, 77, 540, Var.state_creditos)
64 b_voltar = Botao(voltar, voltar_f, meio(voltar), 400, Var.state_menu)
65 b_sair = Botao(sair, sair_f, 314, 11, Var.state_menu, 6, 4)
66 b_menu = Botao(b_menu, b_menu_f, 196, 79, 564, Var.state_menu)
67 b_denovo = Botao(denovo, denovo_f, 185, 501, Var.state_gameplay)
```

Crio animações (dentro do método draw()) de quando se passa o mouse em cima e clicado:



estado normal



mouse em cima

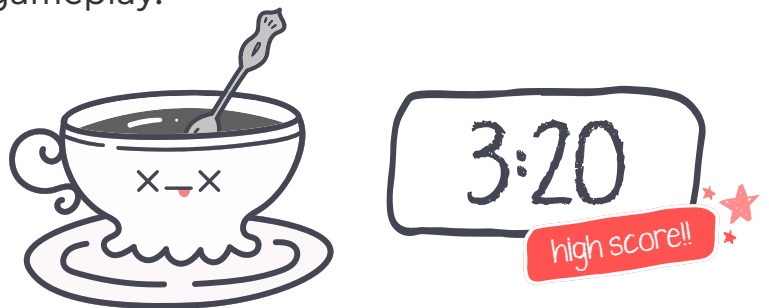


botão clicado

Já o método handle\_event(), confere se o botão foi clicado e muda pro estado chamado ao criar o objeto de cada botão. Também, se caso chamado com a Bool True, ele faz o botão resetar as configurações do jogo – Utilizado para voltar de game over para menu ou gameplay.

```
misture-minha-xicara - gameover.py

9 def gameover():
10     # para a música atual
11     if Var.playing_gameplay_music:
12         mixer.music.fadeout(200)
13         Var.playing_gameplay_music = False
14
15     Var.tempo = 1 #reinicia o contador de tempo para velocidade
16     Var.contador = 0 #reinicia o contador de frames
17     background()
18     # estetica
19     SCREEN.blit(mesa, (0, 418))
20     SCREEN.blit(game_over, (meio(game_over), 53))
21
22     # computa o score e se foi highscore
23     SCREEN.blit(timerbox, (meio(timerbox), 131))
24     score_text = fonte_score.render(Var.score, True, "#454653")
25     score_rect = score_text.get_rect(centers=(WIDTH//2, 195))
26     SCREEN.blit(score_text, score_rect)
27     if Var.score > Var.score_max:
28         Var.highs = True
29         Var.score_max = Var.score
30         set_highscore(Var.score)
31     if Var.highs: #se for, mostra as estrelas e a mensagem
32         SCREEN.blit(highscore_star, (191.86, 201.52))
33
34     #controlador da musica
35     if not Var.playing_high and Var.highs:
36         mixer.music.load("assets/high.wav")
37         mixer.music.play(-1)
38         mixer.music.set_volume(0.1)
39         Var.playing_high = True
40
41     if not Var.playing_dead and not Var.highs:
42         mixer.music.load("assets/gameover.wav")
43         mixer.music.play(-1)
44         mixer.music.set_volume(0.1)
45         Var.playing_dead = True
46
47     # inicia a xicara e os botoes
48     xicara_gameplay.draw()
49     draw_coracoes(Var.cor_mortos)
50     b_menu.draw()
51     b_denovo.draw()
```



Também crio funções para cada tipo de estado do jogo diferente. O exemplo da imagem é do estado gameover.

Nele, faço inicialização da música tanto se fizer highscore ou não. Também são realizadas inicializações de elementos estéticos como a mesa, a xícara morta e o timerbox mostrando o score realizado no jogo.

Nesta função também é realizado o cálculo se o highscore foi atingido e realiza a gravação do tempo obtido (caso atingido) no arquivo score.json.

# screenshots

menu

best score 2:20

misture  
meu chá!



iniciar

créditos

créditos

misture  
meu chá!

criado e ilustrado por  
Pedro Gimenez

voltar



gameplay



best 2:20

1:20

game over

GAME OVER

3:20

high score!!



de novo!

menu