Documentação

1. Introdução

O GamePy é um projeto de RPG de Ação em 2D desenvolvido em Python utilizando a biblioteca **Arcade**. O projeto segue princípios de engenharia de software, separando a lógica de jogo, objetos (entidades, itens, inventário) e a interface do usuário (UI) em módulos distintos para garantir manutenibilidade e escalabilidade.

1.1. Estrutura do Projeto

A organização dos arquivos no diretório src/ (código-fonte) é modular e se divide nas seguintes categorias principais:

- **src/main.py**: O ponto de entrada da aplicação, onde a janela principal do Arcade (MyGame) é inicializada e as principais "Views" do jogo são configuradas.
- **src/constants.py**: Contém todas as constantes globais usadas pelo jogo, como dimensões da tela, velocidades de movimento e caminhos de sprites.
- **src/game_objects/**: Contém as classes que definem os atores e elementos do mundo do jogo (Entidades, Jogador, Inimigos, Itens e Inventário).
- **src/views/**: Contém as classes que herdam de arcade.View e gerenciam as diferentes telas do jogo (Menu, Gameplay, Inventário, Pausa).
- **src/ui/**: Contém os componentes reutilizáveis da interface do usuário (Botões, Slots e Caixa de Log).

1.2. Como Executar o Projeto

Para compilar e executar o GamePy, você deve ter o Python instalado (recomendado 3.10+), além da biblioteca Arcade.

1. **Instalação de Dependências:** Instale a biblioteca Arcade (e quaisquer outras dependências futuras) usando o pip.
   pip install arcade

2. **Execução:** Execute o arquivo principal do jogo a partir da raiz do projeto:
   python src/main.py

2. Visão Geral

A documentação está organizada em três módulos principais de código: **Game Objects**, **Views** e **UI**.

3. Módulo game_objects

Este módulo contém as classes fundamentais que representam entidades e elementos interativos do mundo do jogo.

3.1. Classe Entity

Diretório: game_objects.entity

Representa a classe base para todos os seres vivos no jogo (Jogador, Inimigos).

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	max_hp	int (Valor inicial: Fornecido no init)	Vida máxima da entidade.
Atributo	current_hp	int (Valor inicial: max_hp)	Vida atual da entidade.
Atributo	attack_damage	int (Valor inicial: 0)	Dano base de ataque.
Método	_init_	image_path: str, scale: float, center_x: float, center_y: float, max_hp: int → None	Inicializa a entidade.
Método	take_damage	damage: int → None	Aplica dano à entidade.

3.2. Classe Player

Diretório: game_objects.player

A classe que representa o personagem jogável, herdando de Entity e adicionando mecânicas como movimento e inventário.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	velocity_x	int (Valor inicial: 0)	Velocidade horizontal de movimento.
Atributo	velocity_y	int (Valor inicial: 0)	Velocidade vertical de movimento.
Atributo	is_moving	bool (Valor inicial: False)	Estado de movimento do jogador.
Atributo	equipped_weapon	Item (Valor inicial: None)	Referência à arma equipada.
Atributo	inventory	Inventory (Valor inicial: Inventory())	Objeto que gerencia o inventário.
Método	_init_	→ None	Inicializa o jogador.
Método	update	*args, **kwargs → None	Lógica de atualização a cada quadro.
Método	equip_weapon	weapon, index: int → None	Equipa uma arma.
Método	unequip_weapon	→ None	Desequipa a arma atual.
Método	attack	target → None	Realiza um ataque contra um alvo.
Método	get_items	→ list	Retorna a lista de itens no inventário.

3.3. Classe Enemy

Diretório: game_objects.enemy

Representa inimigos no jogo, com lógica de drop de itens.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	name	str (Valor inicial: Fornecido no init)	Nome do inimigo.
Atributo	drops	list (Valor inicial: Carregado do JSON)	Lista de nomes de itens que o inimigo pode dropar.
Método	_init_	name: str, x: float, y: float → None	Inicializa o inimigo.
Método	on_die	→ Item	Executa a lógica de morte e gera o drop.

3.4. Classe Item

Diretório: game_objects.item

Representa um item colecionável, carregando dados de um arquivo JSON.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	name	str (Valor inicial: Fornecido no init)	Nome do item.
Atributo	image_file	str (Valor inicial: Carregado do JSON)	Caminho do sprite do item.
Atributo	description	str (Valor inicial: Carregado do JSON)	Descrição do item.
Atributo	type	str (Valor inicial: Carregado do JSON)	Tipo de item (ex:, "Material").
Atributo	stack_limit	int (Valor inicial: Carregado do JSON)	Limite de itens na pilha.
Atributo	stack	int (Valor inicial: 1)	Quantidade atual na pilha.
Método	_init_	name: str → None	Inicializa o item.
Método	get_damage	→ int	Retorna o valor de dano (se for uma arma), senão 0.
Método	get_drop_chance	→ int	Retorna a chance de drop (se aplicável), senão 0.

3.5. Classe Inventory

Diretório: game_objects.inventory

Gerencia a coleção de itens e o estado de equipamento do jogador.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	slot_items	list (Valor inicial: [])	Lista de itens nos slots não equipados.
Atributo	equipped_items	list (Valor inicial: [])	Lista de itens atualmente equipados.
Método	_init_	→ None	Inicializa o inventário.
Método	add_item	item: Item → None	Adiciona um item, empilhando se possível.
Método	remove_item	index: int → None	Remove um item pelo índice do slot.
Método	is_on_inventory	item: Item → bool	Verifica se um item já está no inventário.
Método	find_item_index	item: Item → int	Retorna o índice do item na lista de slots.
Método	equip_item	item: Item, index: int → None	Move um item de slot para equipado.
Método	unequip_item	item: Item → None	Move um item equipado para o inventário.

4. Módulo views

Este módulo define as classes de tela do jogo, gerenciando a transição entre estados (Menu, Jogo, Pausa, Inventário).

4.1. Classe View

Diretório: views.view

Classe base que estende arcade.View, fornecendo lógica comum para todas as telas, como gerenciamento de listas de sprites, botões e modo de desenvolvimento.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	background_sprite	arcade.Sprite (Valor inicial: None)	Sprite de fundo da View.
Atributo	general_sprite_list	arcade.SpriteList (Valor inicial: SpriteList())	Lista principal de sprites.
Atributo	button_list	arcade.SpriteList (Valor inicial: SpriteList())	Lista de objetos SpriteButton.
Atributo	window	arcade.Window (Valor inicial: arcade.get_window())	Referência à janela principal.
Atributo	developer_mode	bool (Valor inicial: False)	Flag para o modo de desenvolvedor.
Atributo	is_dragging	bool (Valor inicial: False)	Indica se um sprite está sendo arrastado.
Método	_init_	→ None	Inicializa a View.
Método	on_mouse_motion	x: int, y: int, dx: int, dy: int → None	Atualiza o estado de hover dos botões.
Método	on_key_press	key, modifiers → None	Trata o pressionamento de teclas (incluindo TAB para dev mode).
Método	on_mouse_drag	x, y, dx, dy, buttons, modifiers → None	Trata o arrasto de sprites (em modo dev).
Método	on_mouse_release	x, y, button, modifiers → None	Trata a liberação do mouse.
Método	on_draw	→ None	Método de desenho da View.

4.2. Classe Menu View

Diretório: views.meu_view

Tela inicial do jogo.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Método	_init_	→ None	Inicializa a View do Menu.
Método	on_show_view	→ None	Chamado ao exibir a View.
Método	on_resize	width, height → None	Ajusta os elementos ao redimensionar a janela.
Método	on_mouse_press	x: int, y: int, button: int, modifiers: int → None	Trata cliques do mouse (iniciar jogo).

4.3. Classe Pause View

Diretório: views.pause_view

Tela de pausa do jogo.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Método	_init_	→ None	Inicializa a View de Pausa.
Método	setup	→ None	Configura os botões da tela de pausa.
Método	on_show_view	→ None	Chamado ao exibir a View.
Método	on_mouse_press	x: int, y: int, button: int, modifiers: int → None	Trata cliques nos botões (continuar, inventário, sair).

4.4. Classe Game View

Diretório: views.game_view

View principal do jogo, onde o *gameplay* acontece.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	player	Player (Valor inicial: None)	Instância do Jogador.
Atributo	enemy	Enemy (Valor inicial: None)	Instância do Inimigo.
Método	_init_	→ None	Inicializa a View de Jogo.
Método	setup	→ None	Configura o jogador e inimigos iniciais.
Método	on_show_view	→ None	Chamado ao exibir a View.
Método	on_key_press	key, modifiers → None	Trata o pressionamento de teclas (movimento, ações, menus).
Método	on_key_release	key, modifiers → None	Trata a liberação de teclas (parar movimento).
Método	on_mouse_release	x, y, button, modifiers → None	Trata o ataque com o mouse.
Método	on_update	delta_time → None	Lógica de atualização (movimento, checagem de morte de inimigos).
Método	equip_item_on_game	item: Item → None	Adiciona o sprite do item equipado ao jogo.
Método	unequip_item_on_game	item: Item → None	Remove o sprite do item desequipado do jogo.

4.5. Classe Inventory View

Diretório: views.inventory_view

Tela de Inventário do jogo

| Tipo de Membro | Nome | Parâmetros/Tipo de Retorno | Descrição |
| Atributo | inventory_elements | arcade.SpriteList (Valor inicial: SpriteList()) | Lista de sprites

dos slots. |
Atributo	item_sprites	arcade.SpriteList (Valor inicial: SpriteList())	Sprites dos itens em slots não equipados.
Atributo	equipped_item_sprites	arcade.SpriteList (Valor inicial: SpriteList())	Sprites dos itens equipados.
Atributo	item_detail_view	ItemDetailView (Valor inicial: None)	Tela de detalhes do item selecionado.
Método	_init_	→ None	Inicializa a View de Inventário.
Método	setup	→ None	Configura os slots de equipamento e inventário.
Método	on_show_view	→ None	Chamado ao exibir a View.
Método	on_draw	→ None	Método de desenho, inclui a tela de detalhes se ativa.
Método	on_key_press	key, modifiers → None	Trata o pressionamento de teclas (menus).
Método	on_mouse_drag	x, y, dx, dy, buttons, modifiers → None	Trata o arrasto de elementos (em modo dev).
Método	on_mouse_release	x, y, button, modifiers → None	Trata a liberação do mouse.
Método	on_mouse_press	x, y, button, modifiers → None	Lógica principal de interação: abre detalhes, equipa/desequipa/descarta.
Método	create_slots	→ None	Cria os slots do inventário dinamicamente.
Método	get_free_slot_index	→ int	Retorna o índice do primeiro slot livre.
Método	restructure_slots	→ None	Reorganiza os sprites dos itens após uma mudança no inventário.
Método	add_item_on_display	item: Item → None	Adiciona um sprite de item a um slot disponível na tela.
Método	equip_item_on_display	item: Item → None	Move o item para o slot de equipamento e atualiza o display.

4.6. Classe Item Detail View

Diretório: views.item_detail_view

Classe auxiliar para tela de detalhes do item

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	background_sprite	arcade.Sprite (Valor inicial: Inicializado no init)	Sprite do fundo da caixa de detalhes.
Atributo	item	Item (Valor inicial: Fornecido no init)	O item a ser detalhado.
Atributo	index	int (Valor inicial: Fornecido no init)	Índice do item no inventário.
Atributo	detail_elements	arcade.SpriteList (Valor inicial: SpriteList())	Elementos visuais como slot e botões.
Atributo	equip_button	arcade.Sprite (Valor inicial: Inicializado no setup)	Botão de equipar/desequipar.
Atributo	discard_button	arcade.Sprite (Valor inicial: Inicializado no setup)	Botão de descarte.
Método	_init_	item: Item, center_x: float, center_y: float, index: int → None	Inicializa a View

de Detalhes. |
Método	setup	→ None	Configura o layout e os textos do item.
Método	on_draw	→ None	Desenha a View, incluindo textos e sprites.
Método	on_mouse_drag	x, y, dx, dy → None	Trata o arrasto (em modo dev).
Método	on_mouse_release	x, y → None	Trata a liberação do mouse.
Método	on_mouse_press	x, y → str	Trata cliques nos botões (retorna "equip", "unequip" ou "discard").

5. Módulo UI

Este módulo define os componentes de interface do usuário (UI) reutilizáveis.

5.1. Classe SpriteButton

Diretório: ui.button

Classe auxiliar para botões.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	normal_texture	arcade.Texture (Valor inicial: Definido no init)	Textura quando o mouse está fora.
Atributo	hover_texture	arcade.Texture (Valor inicial: Carregado no init)	Textura quando o mouse está sobre.
Atributo	name	str (Valor inicial: Fornecido no init)	Nome identificador do botão.
Atributo	is_hovered	bool (Valor inicial: False)	Estado de hover.
Método	_init_	name, normal_texture_path, hover_texture_path → None	Inicializa o botão.
Método	on_hover	→ None	Muda a textura para o estado de hover.
Método	on_unhover	→ None	Volta a textura para o estado normal.

5.2. Classe LogBox

Diretório: ui.logbox

Classe auxiliar para slots do inventário e de outras telas.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	max_lines	int (Valor inicial: 7)	Número máximo de linhas a serem exibidas.
Atributo	messages	collections.deque (Valor inicial: Inicializado no init)	Fila de mensagens com tamanho máximo fixo.
Método	_init_	x: float, y: float, width: int, height: int, max_lines: int = 7 → None	Inicializa a caixa de log.
Método	add_message	text: str → None	Adiciona uma nova mensagem.
Método	on_draw	→ None	Desenha a caixa de log e as mensagens.

5.3. Classe Slot

Diretório: ui.slot

Representa um slot de inventário ou equipamento na UI.

Tipo de Membro	Nome	Parâmetros/Tipo de Retorno	Descrição
Atributo	slot_type	str (Valor inicial: Fornecido no init)	Tipo do slot ("normal", "weapon", etc.).
Atributo	item	Item (Valor inicial: None)	Item atualmente contido no slot.
Atributo	normal_slot	arcade.Texture (Valor inicial: Carregado no init)	Textura do slot vazio.
Atributo	uslot	arcade.Texture (Valor inicial: Carregado no init)	Textura do slot com item.
Atributo	index	int (Valor inicial: Fornecido no init)	Índice do slot.
Método	_init_	image_file: str, position_x: float, position_y: float, scale: float, slot_type: str, index: int → None	Inicializa o Slot.
Método	add_item_on_slot	item → arcade.Sprite	Adiciona um item ao slot, muda a textura e retorna o sprite do item.
Método	remove_item_from_slot	→ None	Remove o item e retorna à textura de slot vazio.