

# ENTREGA 2 – ALGORITMOS E LÓGICA DE PROGRAMAÇÃO

--> Aplicações de “Funções” e seus procedimentos no sistema de Inventário:

- **Awake()**
  - Cria instância única do inventário.
  - Se já existir um inventário, o objeto atual é destruído.
- **Start()**
  - Inicializa os slots dos itens filhos.
- **SetItem(Item item)**
  - Adiciona um item ao inventário.
  - Encontra e atualiza o primeiro slot vazio e atualiza a UI com a sprite 2D que for atrelada ao item.
- **HasItem(Item item)**
  - Verifica se um determinado item existe no inventário.
- **SetSelectedSlot(ItemSlot slot)**
  - Define qual slot do inventário está selecionado.
- **DeselectAllSlots()**
  - Deseleciona todos os slots do inventário (se você tiver um item selecionado no slot, é só clicar nele novamente para que o DeselectAllSlots() seja ativado).
- **GetSelectedItem()**
  - Retorna o item do slot que está selecionado ou nulo, caso não tenha nenhum item selecionado.
- **RemoveItem(Item itemToRemove)**

- Remove o item do inventário.
- Limpa slot correspondente e a UI (sprite 2d) dele no slot.

## **--> Aplicações de “Funções” e seus procedimentos no sistema de ItemSlot:**

- **OnPointerClick(PointerEventData eventData)**
  - Detecta o clique esquerdo ou direito.
  - Chama OnLeftClick() ou OnRightClick().
- **OnLeftClick()**
  - Seleciona ou desseleciona o slot.
  - Atualiza o slot no Inventory/Inventário.
- **Select() / Deselect()**
  - Ativa ou desativa o shader de seleção.
  - Atualiza flag thisItemSelected.
- **ClearSlot()**
  - Remove item do slot e atualiza UI.

## **--> Aplicações de “Funções” e seus procedimentos no sistema da Door/Porta:**

- **Interact()**
  - Verifica se a porta está trancada.
  - Checa se o item selecionado é a chave correta.

- Se correto, chama `OpenDoor()` e remove o item do inventário.
- Se incorreto, exibe mensagem de porta trancada.
- **OpenDoor()**
  - Destranca a porta.
  - Desativa o collider de bloqueio da porta (permitindo que o jogador “saia” e encerre o jogo).
  - Destrói o objeto na cena com um `“Destroy.(gameObject)”`;

## **--> Aplicações de “Funções” e seus procedimentos no sistema de Item Coletável/Collectable Item:**

- **Interact()**
  - Checa se o item existe.
  - Adiciona o item ao inventário (por meio também do `Inventory.SetItem(item)`).
  - Remove o objeto/item e seus clones/cópias da cena.

## **--> Aplicações de “Funções” e seus procedimentos no sistema de Sanidade:**

- **SanityManager.Start()**
  - Inicializa o slider de sanidade.

- Configura os valores máximos e reseta efeitos.
- **StartLosingSanity() / StopLosingSanity()**
  - Inicia ou para a perda de sanidade.
- **LoseSanity() (Coroutine)**
  - Reduz o valor da sanidade gradualmente.
  - Aplica alguns efeitos visuais conforme o nível de sanidade do player.
- **ApplyEffects (float p)**
  - Atualiza Vignette, Chromatic Aberration, Film Grain e Lens Distortion.
- **LampManager.ControlarLampada()**
  - Simula as piscadas aleatórias da lâmpada.
  - Inicia a perda de sanidade se a lâmpada for apagada (queimar depois de uma certa quantidade de piscadas), levando o player a falhar no jogo (morrer) caso sua sanidade chegue a 0.
  - Permite reacender a lâmpada coletando uma nova na cena, e assim, interrompendo o consumo/gasto de sanidade.