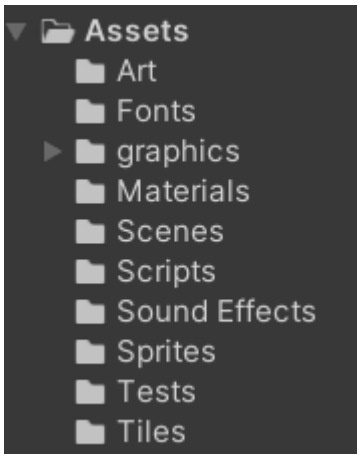
Documentação do ClickClassicos

Por meio deste documento, venho explicar a parte técnica do jogo, e explicar um pouco sobre a plataforma escolhida para fazer o jogo. A plataforma que escolhemos foi a Unity, que é uma plataforma bem conhecida e que já foi usada para fazer vários outros jogos, como podemos ver abaixo:



O Unity possui está pagina chamada Assests, mas o que isso significa? O Assests ele é uma pasta fundamental do Unity, ela sempre é criada quando iniciamos um projeto no Unity e ela serve para que possamos armazenar, todos os ativos do projeto criado, que são eles os modelos 3d, texturas, scripts, áudio e etc.

PONG

A maior parte do Pong está relacionado aos scripts que são eles

. Ball

. MenuMain

. VideoController

. MusicController

. Holder

.Goal

. GameManager

. ControllerPlayers

Mas o que significa scripts?

Os scripts são componentes escrito em uma linguagem de programação que permite adicionar lógica e comportamento do jogo, no caso do Unity a linguagem usada é C#.

Agora que foi explicado os scripts vamos começar a falar do **GameManager,** pode-se dizer que ele é o script mais importante, pois é com ele que gerenciamos o fluxo e a lógica principal do jogo. Como por exemplo

- **Iniciar e encerrar o jogo.**
- **Gerenciar a pontuação do jogador.**
- **Controlar o estado do jogo, como pausa e reinício.**
- **Controlar a progressão do jogo, como níveis e fases.**
- **Gerenciar a lógica do jogo, como colisões e eventos específicos.**
- **Lidar com o gerenciamento de recursos, como carregamento de níveis, texturas, sons, etc.**
- **Gerenciar o salvamento e o carregamento de progresso do jogador.**

Como mostrado o **GameManager** ele é um script importantíssimo para o funcionamento do jogo, mas com eles ainda tem os outros como o **MenuMain** ele é o responsável pelos eventos selecionados

Quando o usuário escolher uma dessas opções ele será direcionado para uma nova cena, A escolher **JOGAR** vai começar o jogo direto, caso escolha as **OPÇÕES** você vai poder habilitar e desabilitar a música do jogo, **SOBRE** será mostrado para o jogador como jogar, nesta tela vai mostrar as teclas para jogar, e caso seja selecionado o **SAIR** o jogo será fechado.

O **Ball** é um objeto e dentro desse script por meio dele é que além de criar a bolinha do **PONG** vamos conseguir controlar sua velocidade, sua colisão para quando acertar a barrinhas que os jogadores estão controlando e com tudo isso também é nela que adicionamos a aleatoriedade dela.

O **VideoController** ele é apenas uma imagem que fica de fundo no menu, para que não fique apenas um fundo escuro, elas serve pra deixar mais vistoso o menu do jogo.

Com o **MusicController** é nele que podemos ativar e desativar a música que está presente no jogo **PONG** sua única função é fazer com que quando selecionado para ativar a música funcione e quanto desativado a música pare.

Holder serve para mostrar o ícone de desativar e ativar a música, de forma mais técnica ele segura a opção de desativar e ativar a música para que ela não seja apagada.

ControllerPlayers aqui entramos na parte que vai diretamente para os jogadores, neste script é onde colocamos o movimentos das barras que são controladas pelos jogadores, aqui também é colocado a velocidade das barras, como por exemplo caso o jogador aperte a tecla **W** a barra do lado esquerdo ela vai subir, da mesma forma que se o jogador apertar a seta pra cima do teclado a barra do lado direito vai subir.

Goal o código ele vai no **GameManager** e pega os métodos `UpSocorePlayerOne()` `UpSocorePlayerTwor()` `UpDifficulty()` para gerenciar a pontuação e a dificuldade do jogo, com isso encerramos a parte do Pong,

Tetris

Agora com a finalização do Pong vamos começar a falar do Tetris, ele contém os seguintes scripts:

.Peca

.Tetromino

.Fantasma

.Borda

.Dados

.PauseTetris

- Vamos começar pelas **PECAS** O método **Initialize** é usado para configurar os dados da peça, o tabuleiro e sua posição inicial.

O método **Update** é chamado a cada quadro e é responsável por controlar a lógica da peça.
A peça pode ser movida para a esquerda ou direita usando as teclas A e D, respectivamente.
A peça pode ser rotacionada no sentido horário ou anti-horário usando as teclas Q e E, respectivamente.A peça pode ser movida para baixo mais rapidamente usando a tecla de espaço A peça avança para a próxima linha a cada intervalo de tempo especificado.
Quando a peça atinge o tempo de bloqueio, ela é travada no tabuleiro e as linhas completas são eliminadas.
O método **Move** verifica se uma determinada posição é válida para mover a peça e, se for o caso, atualiza a posição da peça.
O método **Rotate** realiza a rotação da peça usando uma matriz de rotação.
O método **HardDrop** move a peça para baixo até atingir a posição final.
O método **Lock** coloca a peça no tabuleiro, limpa as linhas completas e gera uma nova peça.

Com o **TETROMINO** se armazena informações sobre um tipo específico de tetrominó, para que facilite o gerenciamento e a manipulação dos tetrominós dentro do jogo Tetris.

O **FANTASMA** ele representa um tetrominó fantasma no jogo Tetris, ele fica responsável por rastrear um tetrominó atualmente em movimento para calcular sua posição mais baixa no tabuleiro e exibir um tetrominó fantasma correspondente no local.

A **BORDA** ela serve para gerenciar a lógica do tabuleiro no jogo Tetris. Ela controla a criação e movimento das peças, verifica se as posições são válidas, limpa as linhas preenchidas e atualiza o estado do tabuleiro.

A classe **DADOS** ela serve para armazenar os dados e constantes usados no jogo Tetris, como as coordenadas das células ocupadas pelos tetrominós, as matrizes de rotação e as coordenadas de deslocamento de parede para cada tipo de peça. Vale lembrar que esses dados serão usados em outras partes do código.

O **PAUSETETRIS** ele serve apenas para poder pausar o jogo caso o jogador precise parar e não queira sair do jogo.