Programação Básica em C# para Unity

Bruno dos Santos de Araújo, MSc

27 de agosto de 2019

Sumário

Introdução

Introdução

- Este treinamento tem os seguintes objetivos:
  - Entender o conceito de `game loop` e como se aplica ao Unity;
  - Mostrar a estrutura básica do tipo de script mais comum no Unity: `MonoBehaviour`;
  - Implementar uma versão simples porém funcional do jogo Breakout (também conhecido como Arkanoid).
- Existem diversas abordagens possíveis para os problemas que iremos resolver, neste treinamento utilizaremos as abordagens mais simples e didáticas e não necessariamente as mais eficientes.

Game loop

Game loop

- Quase todos os jogos utilizam um algoritmo chamado `game loop`, que basicamente possui 3 funções:
  - Ler e processar a entrada do usuário
  - Atualizar o estado interno do jogo
  - Desenhar a tela
- O número de vezes que esse algoritmo é executado por segundo determina a taxa de **quadros por segundo** do jogo, ou **FPS** (frames per second)
  - 30 FPS: a cada 33ms
  - 60 FPS: a cada 16ms

Game loop

- Visualização do algoritmo:

```
while(gameIsRunnning) {
  readInput();
  update();
  render();
}
```

- `readInput()`: lê a entrada do usuário
- `update()`: atualiza o estado interno do jogo
- `render()`: desenha a tela

Game loop

- Ao implementar código de gameplay no Unity, precisamos sempre implementar os equivalentes a `readInput()` e `update()` dentro de uma subclasse de `MonoBehaviour`.
  - `render()` é implementada via shaders; mesmo que você não escreva nenhum, o Unity utiliza shaders padrão
  - Boa referência sobre o assunto: `https://gameprogrammingpatterns.com/game-loop.html`

Estrutura de um MonoBehaviour

Estrutura de um MonoBehaviour

- `MonoBehaviour` é a classe da qual se deriva a maioria dos componentes dentro do Unity.
- Ao criarmos um novo `MonoBehaviour`, seja diretamente como componente num GameObject ou na hierarquia, nos deparamos com o seguinte código:

```
public class NewBehaviourScript : MonoBehaviour
{
  void Start()
  {
  }

  void Update()
  {
  }
}
```

- Podemos observar duas funções que foram criadas por padrão: `Start()` e `Update()`.

Estrutura de um MonoBehaviour

- `Start()`: esta função é chamada assim que o GameObject é ativado em uma cena;
  - Normalmente utilizada para inicializar quaisquer variáveis e chamar funções de inicialização.
- `Update()`: esta função é chamada a cada quadro, e é utilizada para atualizar o estado interno do componente.
  - No caso de ser um objeto controlável pelo jogador, também é onde se leem as entradas (teclas, botões, etc);
  - O tempo entre chamadas é aproximadamente o mesmo e pode variar de acordo com a quantidade de objetos na cena.

Estrutura de um MonoBehaviour

- Outras funções comumente utilizadas:
  - Awake(): é chamada no ato da criação do GameObject, sempre antes de Start(), independentemente do GameObject estar ativado ou não;
  - FixedUpdate(): semelhante a Update() mas é chamada em tempos fixos a cada passo da Física, que por padrão é 20ms;
    - Utilizada principalmente quando se utilizam funções que agem na física do GameObject.
  - OnEnable() e OnDisable(): chamadas na ativação e desativação do GameObject, respectivamente;