**Guia de Desenvolvimento**

**BPM Game Engine**

**Versão 1.0**

Fabio Takeshi Ishikawa

Janeiro de 2020

Contents

[1. Introdução 2](#_Toc34600654)

[2. Configuração do Ambiente de Desenvolvimento 3](#_Toc34600655)

[3. Visão Geral da Arquitetura 3](#_Toc34600656)

[4. Dependências 5](#_Toc34600657)

[5. Debugando seu código 5](#_Toc34600658)

[6. Classes e Objetos Globais 5](#_Toc34600659)

[6.1 Classe Game Engine 5](#_Toc34600660)

[6.1.1 Atributos 5](#_Toc34600661)

[6.1.2 Métodos 5](#_Toc34600662)

[6.2 Classe Window 6](#_Toc34600663)

[6.2.1 Atributos 6](#_Toc34600664)

[6.2.2 Métodos 6](#_Toc34600665)

[6.2 Classe ApiWrapper (classe abstrata) 7](#_Toc34600666)

[6.2.1 Atributos 7](#_Toc34600667)

[6.2.2 Métodos 7](#_Toc34600668)

[6.3 Classe WinApi Wrapper (herdade de ApiWrapper) 7](#_Toc34600669)

[6.3.1 Atributos 7](#_Toc34600670)

[6.3.2 Métodos 7](#_Toc34600671)

[6.4 Classe GameEngineEventHandler (classe abstrata) 7](#_Toc34600672)

[6.5 Classe Render 7](#_Toc34600673)

[6.6 Classe IO 8](#_Toc34600674)

[6.7 Classe Mouse 8](#_Toc34600675)

[6.8 Classe Teclado 8](#_Toc34600676)

[6.9 Classe AssetManager 8](#_Toc34600677)

[7. Personalização 8](#_Toc34600678)

[8. Convenção de Código 8](#_Toc34600679)

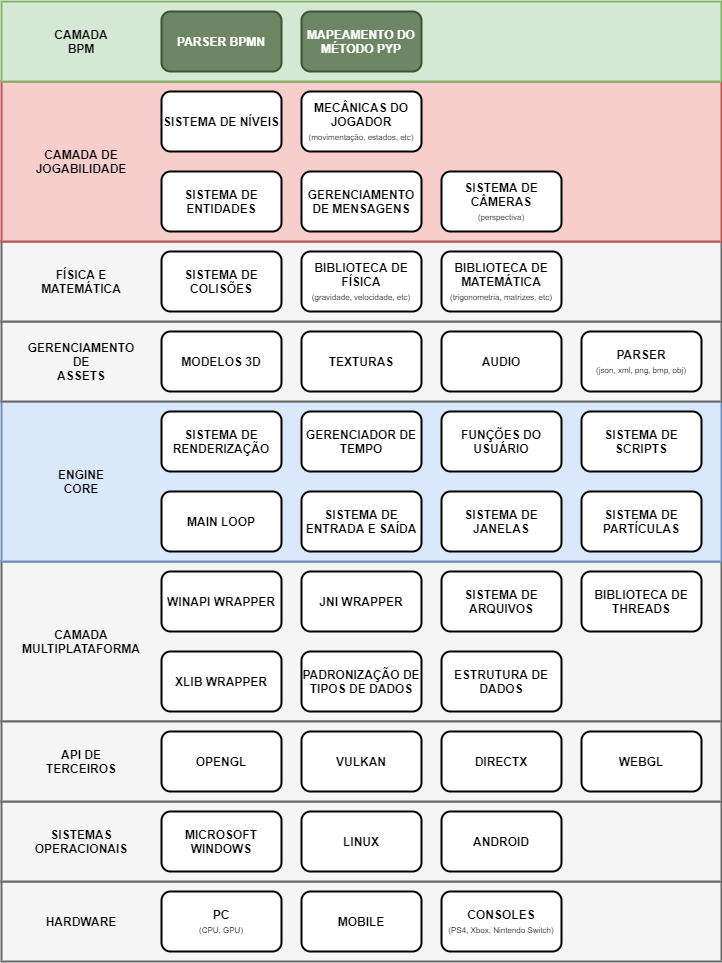
# 1. Introdução

Este guia tem como objetivo descrever em detalhes a arquitetura do BPM Game Engine, as classes que fazem parte do sistema, gerenciamento de configuração, detalhes de desenvolvimento como convenção de código, depuração, testes e outras práticas relacionada ao desenvolvimento e como realizar a customização do sistema.

# 2. Configuração do Ambiente de Desenvolvimento

# 3. Visão Geral da Arquitetura

Visão geral dos componentes que compõe a BPM Game Engine. A imagem está dividida em várias camadas.



*Figura 01. Visão geral dos componentes de BPM Game Engine dividido em camadas.*

# 4. Dependências

# 5. Debugando seu código

# 6. Classes e Objetos Globais

## 6.1 Classe Game Engine

### 6.1.1 Atributos

Objeto APIWrapper (classe abstrata)

Este objeto é responsável por executar métodos específicos do sistema operacional. Para cada sistema operacional, é necessário criar uma classe derivada de APIWrapper e definir os métodos virtuais. O tipo do objeto é definido no momento da compilação através de uma constante de compilação.

Objeto GameEngineEventHandler (classe abstrata)

Este objeto especifica as funções de callback da game engine como, por exemplo, as funções de inicialização, renderização e de finalização. Como padrão, a game engine cria as suas próprias funções de callback vazias. Elas devem ser configuradas com funções de callback definidas pelo usuário.

String Name

Especifica o nome do aplicação ou jogo e será utilizada como texto na barra de títulos de uma janela.

### 6.1.2 Métodos

SetGameEngineEvents

Método utilizado para configurar as funções de callback.

mainLoop

Função de execução da game engine. Assim que esse método é chamado, a game engine obtem o controle total da aplicação. Obviamente, ela respeitará as funções de callback do usuário para responder aos eventos conforme o desejo do usuário.

InitGameEngine

Método para inicializar a game engine.

## 6.2 Classe Window

### 6.2.1 Atributos

String titulo

Especifica o texto que será exibido na barra de títulos da janela de aplicação.

Int largura

Especifica a largura da janela em pixels

Int altura

Especifica a altura da janela em pixels

Int estilo

Especifica o estilo da janela. Os estilos possívels para uma janela são: com barra de título, sem barras de titilo, diferentes tipos de botões do sistema, todas as opções, janela redimensionavel, e não redimensionavel, janela maximizada, minimzada)

Int posicaoInicialX

Especifica a posição inicial da janela no eixo X.

Int posicaçãoInicialY

Especifica a posição inicial da janela no eixo Y.

### 6.2.2 Métodos

CreateWindow

Método para criar uma janela de aplicação.

DestroyWindow

Método para destruir uma janela de aplicação.

ResizeWindow

Método para redimensionar o tamanho de uma janela de aplicação.

setWindowPosition

Método para posicionar a janela

## 6.2 Classe ApiWrapper (classe abstrata)

Esta classe é responsável por tornar portável os atributos e métodos específicos de um sistema operacional. Ela é uma classe abstrata e os seus métodos devem ser definidos pelas classes derivadas.

### 6.2.1 Atributos

### 6.2.2 Métodos

## 6.3 Classe WinApi Wrapper (herdade de ApiWrapper)

Esta classe define os atributos e os métodos específicos do sistema operacional Windows.

### 6.3.1 Atributos

WGLOpenGL

WindowClass

### 6.3.2 Métodos

Window Procedure

Método responsável por tratar os eventos do sistema operacional.

## 6.4 Classe GameEngineEventHandler (classe abstrata)

Esta classe é responsável por definir funções de callback que serão chamadas pela game engine implicitamente. O usuário deverá criar uma classe derivada e definir as suas próprias funções de callback e configurar a game engine.

## 6.5 Classe Render

Esta classe é responsável por armazernar diversas configurações relacionadas renderização.

## 6.6 Classe IO

Esta classe é responsável por armazenar diversas informações a respeito de entrada e saída de dados.

## 6.7 Classe Mouse

Esta classe é responsável por receber os dados de entrada do mouse e disponibilizar para a game engine.

## 6.8 Classe Teclado

Esta classe é responsável por receber os dados de entrada do teclado e disponibilizar para a game engine.

## 6.9 Classe AssetManager

Esta classe é responsável por gerenciar diversos recursos como assets.

# 7. Personalização

# 8. Convenção de Código