Documentação do Código

# Cenas

## GameObjects

Cena somente para criar GameObjects.

## Scene1

Cena principal que carrega botões de cada diagrama.

## SceneDiagram

Cena responsável por renderizar o diagrama selecionado em Scene 1.

# Scripts

## Program.cs

Arquivo principal que é chamado pelo Unity.

## XMI.cs

Responsável por ler o arquivo XMI e armazenar os valores em variáveis.

## Diagram.cs

E o mediador entre XMI.cs e Program.cs. Este arquivo le as informaçoes do XMI.cs e as estrutura de forma a facilitar o manuseio pelo Program.cs ou outros script. Desta maneira, existe uma classe que corresponde a um diagrama. Por exemplo, caso exista no arquivo.xmi diagramas do tipo sequencia, a classe Sequence possui instancias de cada diagrama de sequencia.

## SceneDiagram.cs

Script para controlar o diagrama selecionado em "Scene 1".

## Prefabs/ButtonDiagram.cs

Para acessar cada diagrama, este script cria um botão para cada diagrama identificado no arquivo.xmi.

## Prefabs/Life.cs

Script para controlar o GameObject que renderiza o "quadrado" do objeto ou ator e a "linha" representando a lifeline.

## Algoritms/CenterObject.cs

Centralizar os GameObjects em cena.

# Classes

## Program

### Variáveis

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| - | Diagram | diagram |  |
| + | string | arquivoXMI |  |
| + | GameObject | ButtonDiagramGameObject |  |
| - | GameObject | ins |  |
| + | ArrayList | instances |  |

### Constantes

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| - | int | SPACEX |  |

### Métodos

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Retorno** | **Nome** | **Descrição** |
| + | Void | Start |  |
| - | Void | ButtonForEachSequenceDiagram |  |

## XMI

### Variáveis

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| - | String | ArquivoXmi | URL do arquivo XMI. |
| - | XmlDocument | xmlDocument | Objeto do C# para manipular arquivos XML. |
| + | ArrayList | OwnedBehavior | Armazena as informações de <ownedBehavior>. |
| + | ArrayList | Lifeline | Armazena as informações de <lifeline> dentro de < ownedBehavior >. |
| + | ArrayList | Message | Armazena as informações de <message> dentro de < ownedBehavior >. |
| + | Dictionary<string, ArrayList> | Diagrams | Essa variável armazena todos os diagrama do arquivo XMI. Em string, será armazenado o valor de xmi:id e em ArrayList, todos os nós filhos e seus atributos como objetos do tipo Element. |
| - | ArrayList | Element |  |

### Constantes

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
|  |  |  | As constantes são os valores exatos das tags e atributos no XMI. |

### Construtor

|  |  |  |
| --- | --- | --- |
| **Vis.** | **Nome** | **Descrição** |
| + | XMI | São instanciadas a variáveis é chamado o método readTag() para ler as tags OwnedBehavior, Diagram e Elements do XMI. |

### Métodos

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Retorno** | **Nome** | **Descrição** |
| - | Void | readTag(XmlNodeList nodeList) | Principal método. Ele é chamado no construtor para ler o XMI e armazenar os valores nas variáveis.  Lê a tag especificada em nodeList e chama o método loopChild para ler seus nós filhos. |
| - | Void | loopChild(XmlNode node) |  |
| - | Dictionary<string,string> | readAttributes(XmlNode node) | Lê os atributos e seus valores de uma tag específica. |
| - | Void | createElement( XmlNode tag , Dictionary<string,string> attr) | Este método identifica as tags e seus atributos e armazena em suas respectivas variáveis. |

## Element

### Variáveis

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| + | String | Tag | O nome da tag. |
| + | Dictionary<string, string> | AttributesElement | Dicionário que armazena primeiro o nome do atributo e depois seus valor |

### Construtor

|  |  |  |
| --- | --- | --- |
| **Vis.** | **Nome** | **Descrição** |
| + | Element( string tag , Dictionary<string, string> attr ) | Esta classe trata cada tag como um elemento que possui nome, atributos e valores de seus atributos. |

## Diagram

### Variáveis

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| + | string | Id | Todo diagrama no XMI possui um Id, por isso, deve ser herdado por todas as classes de diagrama. |
| + | string | Name | Todo diagrama no XMI possui um Nome, por isso, deve ser herdado por todas as classes de diagrama. |
| - | XMI | FileXMI | Chama a classe XMI. |
| + | ArrayList | SequenceDiagrams | Armazena os diagramas de sequência identificados pelo método *identifyTypeDiagram* como objetos da classe Sequence. |

### Construtor

|  |  |  |
| --- | --- | --- |
| **Vis.** | **Nome** | **Descrição** |
| + | Diagram( string xmi ) | Carrega o arquivo XMI, variáveis e métodos importantes. |

### Métodos

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Retorno** | **Nome** | **Descrição** |
| - | Void | identifyTypeDiagram() | Lê a variável Diagrams de XMI, identifica o tipo de cada diagrama e armazena em cada variável da classe Diagram. |
| + | Void | setIdandName(ArrayList d) | Este método é para ser reutilizado pelas classes que os estenderem. Como todo diagrama tem um id e nome, todo diagrama instanciado será setado o id e nome. |

## Behavioral : Diagram

## Sequence : Behavioral

### Variáveis

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Tipo** | **Nome** | **Descrição** |
| + | ArrayList | ElementsSequence |  |
| + | Dictionary<Element,Element> | Objects | As informações de cada Objeto/Ator estão em duas tags: <lifeline> e <element>. Então, no primeiro Element será armazenado o lifeline e o segundo element. Ambos tratam do mesmo Objeto/Ator, porém com estruturas diferentes. |
| + | ArrayList | Messages |  |

### Construtor

|  |  |  |
| --- | --- | --- |
| **Vis.** | **Nome** | **Descrição** |
|  |  |  |

### Métodos

|  |  |  |  |
| --- | --- | --- | --- |
| **Vis.** | **Retorno** | **Nome** | **Descrição** |
| - | Void | setObjects() | Seu objetivo é identificar armazenar os Objetos ou Atores na variável Objects.  As informações relevantes estão em <elements> dentro de <diagram> e <lifeline> dentro de <ownedBehavior>.  A ideia é salvar essas duas tags referente a Objetos/Atores na variável Objects. |