ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ  
"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"

Кафедра программной инженерии

им. Л.П. Фельдмана

# ОТЧЁТ

к лабораторной работе по дисциплине  
«Профессиональная практика программной инженерии»  
на тему: «Создание самодокументирующегося кода»

Вариант 12: 3D-редактор

Выполнил:

ст. гр. ПИ-19б  
Носаченко А. А.

Проверил:

асс. каф. ПИ  
Дмитрюк Т. Г.

Донецк 2023

Цель: научиться добавлять в программный код специальным образом оформление докблок-комментарии, для последующей автоматической генерации API reference, а также познакомиться с форматом оформления документации DocBook.

Задание:

Внедрить в ранее разработанный программный код докблоки.

Разработать DocBook и проверить корректность созданного DocBook файла с помощью средств редактора или одного из онлайн валидаторов.

Ход работы

Программный код до внедрения документационных комментариев:

public partial class MainWindow : Window

{

private Primitives.Primitives primitives = null;

private Modelling.Modelling modelling = null;

private Painting.Painting painting = null;

private Animation.Animation animation = null;

private Scene\_structure\_controller scene\_structure = null;

private Scene\_controller scene = null;

private User\_interfaces interfaces = null;

private Project\_resources\_controller project\_controller = null;

public MainWindow()

{

InitializeComponent();

Init\_fields();

Message();

}

private void Init\_fields()

{

primitives = new Primitives.Primitives();

modelling = new Modelling.Modelling();

painting = new Painting.Painting();

animation = new Animation.Animation();

scene\_structure = new Scene\_structure\_controller();

scene = new Scene\_controller();

interfaces = new User\_interfaces();

project\_controller = new Project\_resources\_controller();

}

private void Message()

{

StringBuilder message = new StringBuilder();

message.AppendLine(primitives .\_class\_name);

message.AppendLine(modelling .\_class\_name);

message.AppendLine(painting .\_class\_name);

message.AppendLine(animation .\_class\_name);

message.AppendLine(scene\_structure .\_class\_name);

message.AppendLine(scene .\_class\_name);

message.AppendLine(interfaces .\_class\_name);

message.AppendLine(project\_controller.\_class\_name);

MessageBox.Show(message.ToString());

}

}

Программный код после внедрения документационных комментариев:

/// <summary>

/// Функциональная составляющая основного класса программы

/// </summary>

public partial class MainWindow : Window

{

private Primitives.Primitives primitives = null;

private Modelling.Modelling modelling = null;

private Painting.Painting painting = null;

private Animation.Animation animation = null;

private Scene\_structure\_controller scene\_structure = null;

private Scene\_controller scene = null;

private User\_interfaces interfaces = null;

private Project\_resources\_controller project\_controller = null;

/// <summary>

/// Конструктор класса главного окна

/// </summary>

public MainWindow()

{

InitializeComponent();

Init\_fields();

Message();

}

/// <summary>

/// Обеспечивает инициализацию полей класса

/// </summary>

private void Init\_fields()

{

primitives = new Primitives.Primitives();

modelling = new Modelling.Modelling();

painting = new Painting.Painting();

animation = new Animation.Animation();

scene\_structure = new Scene\_structure\_controller();

scene = new Scene\_controller();

interfaces = new User\_interfaces();

project\_controller = new Project\_resources\_controller();

}

/// <summary>

/// Обеспечивает вывод сообщенияпользователю, содержащего названия классов полей.

/// </summary>

private void Message()

{

StringBuilder message = new StringBuilder();

message.AppendLine(primitives .\_class\_name);

message.AppendLine(modelling .\_class\_name);

message.AppendLine(painting .\_class\_name);

message.AppendLine(animation .\_class\_name);

message.AppendLine(scene\_structure .\_class\_name);

message.AppendLine(scene .\_class\_name);

message.AppendLine(interfaces .\_class\_name);

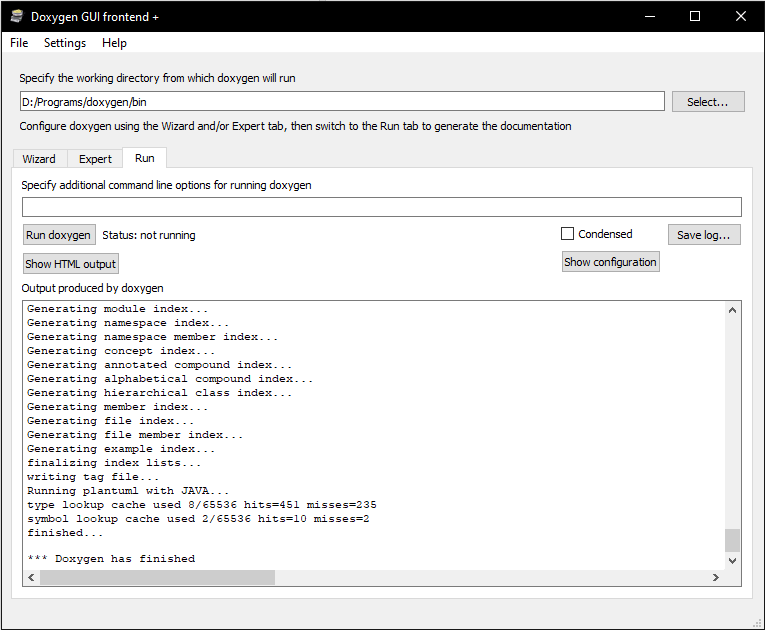
message.AppendLine(project\_controller.\_class\_name);

MessageBox.Show(message.ToString());

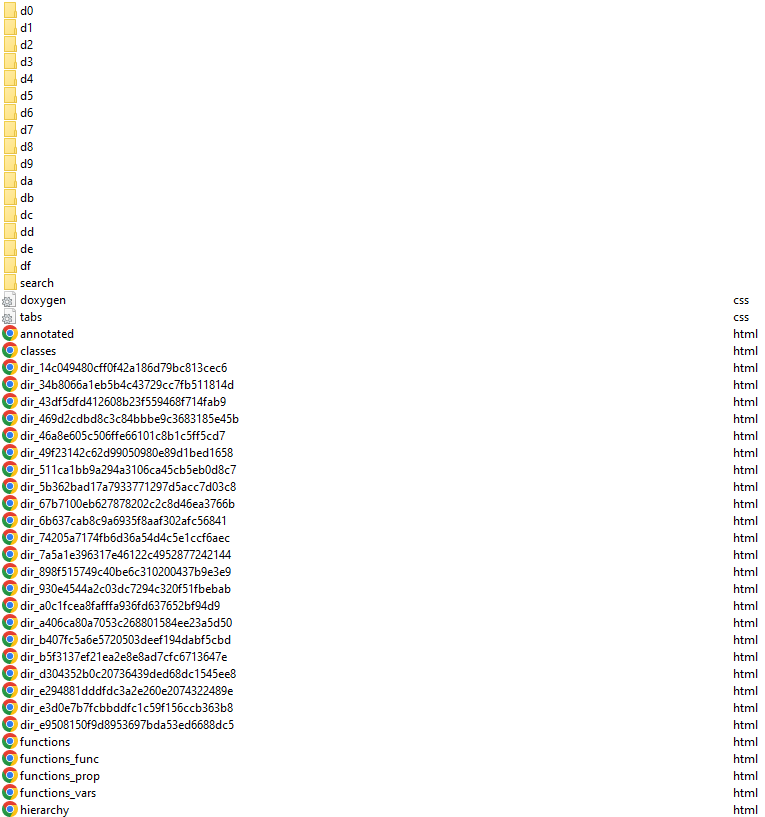
}

}

Приложение, создающее документацию на основе проекта - doxygen:



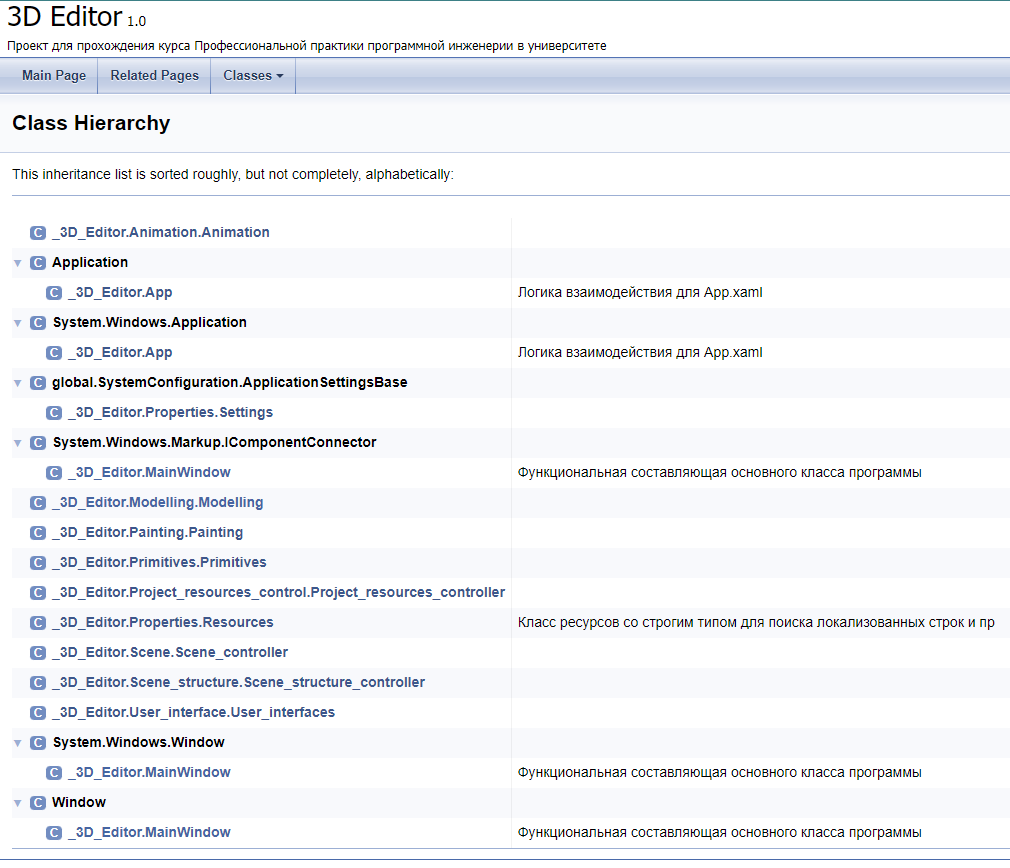
Созданный документационный сайт при помощи doxygen включает:



…



Созданный документационный сайт при помощи doxygen представлен в виде:



Вывод: в ходе лабораторной работы мы научились добавлять в программный код специальным образом оформленные докблок-комментарии, для последующей автоматической генерации API-документации, а также познакомились с форматом оформления документации DocBook.