Лабораторная работа № 11

студента группы ИТз-221

Дмитриева Дмитрия Анатольевича

*Выполнение: \_\_\_\_\_\_\_\_\_ Защита: \_\_\_\_\_\_\_\_\_\_\_*

Документирование кода с помощью XML-комментариев

*Цель работы***:** Знакомство с тегами XML, используемыми для документирования. Получение практических навыков создания документации кода C# с помощью XML-комментариев.

**Ход работы:**

***Вариант - 12***

1. Для класса Time добавил XML- комментарии и сгенерировал XML- документацию
2. Класс Time с XML документацией

namespace lab2

{

/// <summary>

/// Класс, представляющий время в формате "часы:минуты:секунды"

/// </summary>

/// <see href="https://classroom.google.com/c/MjI3Nzg0NDI3NzA3/a/NDkzNjU0NjI0NDI1/details">Лабораторная</see>

public class Time

{

private uint \_h, \_m, \_s;

/// <summary>

/// Получает или задает количество часов

/// </summary>

/// <example>17</example>

/// <value>

/// Значение от 0 до 23

/// </value>

/// <exception cref="ArgumentException">Если значение больше 23</exception>

public uint Hours

{

get => \_h;

set

{

if (value > 23) throw new ArgumentException("Часы должны быть от 0 до 23");

\_h = value;

}

}

/// <summary>

/// Получает или задает количество минут

/// </summary>

/// <example>13</example>

/// <value>

/// Значение от 0 до 59

/// </value>

/// <exception cref="ArgumentException">Если значение больше 59</exception>

public uint Minutes

{

get => \_m;

set

{

if (value > 59) throw new ArgumentException("Минуты должны быть от 0 до 59");

\_m = value;

}

}

/// <summary>

/// Получает или задает количество секунд

/// </summary>

/// <value>

/// <example>50</example>

/// Значение от 0 до 59

/// </value>

/// <exception cref="ArgumentException">Если значение больше 59</exception>

public uint Seconds

{

get => \_s;

set

{

if (value > 59) throw new ArgumentException("Секунды должны быть от 0 до 59");

\_s = value;

}

}

/// <summary>

/// Новый экземпляр класса Time

/// </summary>

/// <param name="hours">Часы (0-23)</param>

/// <param name="minutes">Минуты (0-59)</param>

/// <param name="seconds">Секунды (0-59)</param>

public Time(uint hours = 0, uint minutes = 0, uint seconds = 0)

{

Hours = hours;

Minutes = minutes;

Seconds = seconds;

}

/// <summary>

/// Новый экземпляр класса Time

/// </summary>

/// <param name="hours">Часы (0-23)</param>

/// <param name="minutes">Минуты (0-59)</param>

public Time(uint hours = 0, uint minutes = 0)

{

Hours = hours;

Minutes = minutes;

Seconds = 0;

}

/// <summary>

/// Новый экземпляр класса Time

/// </summary>

/// <param name="hours">Часы (0-23)</param>

public Time(uint hours = 0)

{

Hours = hours;

Minutes = 0;

Seconds = 0;

}

public Time(Time other) : this(other.\_h, other.\_m, other.\_s)

{

}

/// <summary>

/// Выводит время в формате "часы:минуты:секунды" в терминал

/// </summary>

/// <remarks>

/// <para>Метод использует перегрузку ToString()</para>

/// </remarks>

public void Display()

{

Console.WriteLine(ToString());

}

/// <summary>

/// Преобразует время в количество секунд

/// </summary>

/// <returns>Количество секунд, равное времени в объекте</returns>

public uint ToSeconds() => \_h \* 3600 + \_m \* 60 + \_s;

/// <summary>

/// Преобразует время в количество минут

/// </summary>

/// <returns>Количество минут, равное времени в объекте</returns>

public double ToMinutes() => \_h \* 60 + \_m + \_s / 60.0;

/// <summary>

/// Находит разницу между двумя экзеплярами времени в секундах

/// </summary>

/// <param name="t1">Первое время</param>

/// <param name="t2">Второе время</param>

/// <returns>Разница в секундах</returns>

public static int DifferenceInSeconds(Time t1, Time t2)

{

return Math.Abs((int)t1.ToSeconds() - (int)t2.ToSeconds());

}

/// <summary>

/// Оператор сложения

/// </summary>

/// <param name="t">Время</param>

/// <param name="seconds">Количество секунд, на которое увеличивается время</param>

/// <returns>Новое время</returns>

public static Time operator +(Time t, int seconds)

{

var totalSeconds = (uint)(t.ToSeconds() + seconds);

return FromSeconds(totalSeconds);

}

/// <summary>

/// Оператор вычитания

/// </summary>

/// <param name="t">Время</param>

/// <param name="seconds">Количество секунд, на которое уменьшается время</param>

/// <returns>Новое время</returns>

public static Time operator -(Time t, int seconds)

{

var totalSeconds = (int)t.ToSeconds() - seconds;

if (totalSeconds < 0)

{

totalSeconds = 0;

}

return FromSeconds((uint)totalSeconds);

}

/// <summary>

/// Оператор инкремента

/// </summary>

/// <param name="t1">Время</param>

/// <returns>Новое время, увеличенное на 1 секунду</returns>

public static Time operator ++(Time t1)

{

var seconds = t1.ToSeconds() + 1;

return FromSeconds(seconds);

}

/// <summary>

/// Оператор декремента

/// </summary>

/// <param name="t1">Время</param>

/// <returns>Новое время, уменьшенное на 1 секунду</returns>

public static Time operator --(Time t1)

{

var seconds = t1.ToSeconds() - 1;

return FromSeconds(seconds);

}

/// <summary>

/// Оператор больше

/// </summary>

/// <param name="t1">Первое время</param>

/// <param name="t2">Второе время</param>

/// <returns>Логическое значение</returns>

public static bool operator >(Time t1, Time t2) => t1.ToSeconds() > t2.ToSeconds();

/// <summary>

/// Оператор меньше

/// </summary>

/// <param name="t1">Первое время</param>

/// <param name="t2">Второе время</param>

/// <returns>Логическое значение</returns>

public static bool operator <(Time t1, Time t2) => t1.ToSeconds() < t2.ToSeconds();

/// <summary>

/// Оператор больше или равно

/// </summary>

/// <param name="t1">Первое время</param>

/// <param name="t2">Второе время</param>

/// <returns>Логическое значение</returns>

public static bool operator >=(Time t1, Time t2) => t1.ToSeconds() >= t2.ToSeconds();

/// <summary>

/// Оператор меньше или равно

/// </summary>

/// <param name="t1">Первое время</param>

/// <param name="t2">Второе время</param>

/// <returns>Логическое значение</returns>

public static bool operator <=(Time t1, Time t2) => t1.ToSeconds() <= t2.ToSeconds();

/// <summary>

/// Оператор true

/// </summary>

/// <param name="t">Время для проверки</param>

/// <returns>Логическое значение</returns>

public static bool operator true(Time t) => t.ToSeconds() > 0;

/// <summary>

/// Оператор false

/// </summary>

/// <param name="t">Время для проверки</param>

/// <returns>Логическое значение</returns>

public static bool operator false(Time t) => t.ToSeconds() == 0;

/// <summary>

/// Преобразует Time в строку

/// </summary>

/// <returns>Строка в формате "часы:минуты:секунды"</returns>

public override string ToString()

{

return $"{\_h:D2}:{\_m:D2}:{\_s:D2}";

}

/// <summary>

/// Преобразует количество секунд в объект Time

/// </summary>

/// <param name="totalSeconds">Количество секунд</param>

/// <c>var Time time = Time.FromSeconds(1000)</c>

/// <returns>Объект Time</returns>

private static Time FromSeconds(uint totalSeconds)

{

var hours = (totalSeconds / 3600) % 24;

var minutes = (totalSeconds / 60) % 60;

var seconds = totalSeconds % 60;

return new Time(hours, minutes, seconds);

}

}

}

1. Текст Xml документации

<?xml version="1.0"?>

<doc>

<assembly>

<name>lab2</name>

</assembly>

<members>

<member name="T:lab2.Time">

<summary>

Класс, представляющий время в формате "часы:минуты:секунды"

</summary>

<see href="https://classroom.google.com/c/MjI3Nzg0NDI3NzA3/a/NDkzNjU0NjI0NDI1/details">Лабораторная</see>

</member>

<member name="P:lab2.Time.Hours">

<summary>

Получает или задает количество часов

</summary>

<example>17</example>

<value>

Значение от 0 до 23

</value>

<exception cref="T:System.ArgumentException">Если значение больше 23</exception>

</member>

<member name="P:lab2.Time.Minutes">

<summary>

Получает или задает количество минут

</summary>

<example>13</example>

<value>

Значение от 0 до 59

</value>

<exception cref="T:System.ArgumentException">Если значение больше 59</exception>

</member>

<member name="P:lab2.Time.Seconds">

<summary>

Получает или задает количество секунд

</summary>

<value>

<example>50</example>

Значение от 0 до 59

</value>

<exception cref="T:System.ArgumentException">Если значение больше 59</exception>

</member>

<member name="M:lab2.Time.#ctor(System.UInt32,System.UInt32,System.UInt32)">

<summary>

Новый экземпляр класса Time

</summary>

<param name="hours">Часы (0-23)</param>

<param name="minutes">Минуты (0-59)</param>

<param name="seconds">Секунды (0-59)</param>

</member>

<member name="M:lab2.Time.#ctor(System.UInt32,System.UInt32)">

<summary>

Новый экземпляр класса Time

</summary>

<param name="hours">Часы (0-23)</param>

<param name="minutes">Минуты (0-59)</param>

</member>

<member name="M:lab2.Time.#ctor(System.UInt32)">

<summary>

Новый экземпляр класса Time

</summary>

<param name="hours">Часы (0-23)</param>

</member>

<member name="M:lab2.Time.Display">

<summary>

Выводит время в формате "часы:минуты:секунды" в терминал

</summary>

<remarks>

<para>Метод использует перегрузку ToString()</para>

</remarks>

</member>

<member name="M:lab2.Time.ToSeconds">

<summary>

Преобразует время в количество секунд

</summary>

<returns>Количество секунд, равное времени в объекте</returns>

</member>

<member name="M:lab2.Time.ToMinutes">

<summary>

Преобразует время в количество минут

</summary>

<returns>Количество минут, равное времени в объекте</returns>

</member>

<member name="M:lab2.Time.DifferenceInSeconds(lab2.Time,lab2.Time)">

<summary>

Находит разницу между двумя экзеплярами времени в секундах

</summary>

<param name="t1">Первое время</param>

<param name="t2">Второе время</param>

<returns>Разница в секундах</returns>

</member>

<member name="M:lab2.Time.op\_Addition(lab2.Time,System.Int32)">

<summary>

Оператор сложения

</summary>

<param name="t">Время</param>

<param name="seconds">Количество секунд, на которое увеличивается время</param>

<returns>Новое время</returns>

</member>

<member name="M:lab2.Time.op\_Subtraction(lab2.Time,System.Int32)">

<summary>

Оператор вычитания

</summary>

<param name="t">Время</param>

<param name="seconds">Количество секунд, на которое уменьшается время</param>

<returns>Новое время</returns>

</member>

<member name="M:lab2.Time.op\_Increment(lab2.Time)">

<summary>

Оператор инкремента

</summary>

<param name="t1">Время</param>

<returns>Новое время, увеличенное на 1 секунду</returns>

</member>

<member name="M:lab2.Time.op\_Decrement(lab2.Time)">

<summary>

Оператор декремента

</summary>

<param name="t1">Время</param>

<returns>Новое время, уменьшенное на 1 секунду</returns>

</member>

<member name="M:lab2.Time.op\_GreaterThan(lab2.Time,lab2.Time)">

<summary>

Оператор больше

</summary>

<param name="t1">Первое время</param>

<param name="t2">Второе время</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.op\_LessThan(lab2.Time,lab2.Time)">

<summary>

Оператор меньше

</summary>

<param name="t1">Первое время</param>

<param name="t2">Второе время</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.op\_GreaterThanOrEqual(lab2.Time,lab2.Time)">

<summary>

Оператор больше или равно

</summary>

<param name="t1">Первое время</param>

<param name="t2">Второе время</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.op\_LessThanOrEqual(lab2.Time,lab2.Time)">

<summary>

Оператор меньше или равно

</summary>

<param name="t1">Первое время</param>

<param name="t2">Второе время</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.op\_True(lab2.Time)">

<summary>

Оператор true

</summary>

<param name="t">Время для проверки</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.op\_False(lab2.Time)">

<summary>

Оператор false

</summary>

<param name="t">Время для проверки</param>

<returns>Логическое значение</returns>

</member>

<member name="M:lab2.Time.ToString">

<summary>

Преобразует Time в строку

</summary>

<returns>Строка в формате "часы:минуты:секунды"</returns>

</member>

<member name="M:lab2.Time.FromSeconds(System.UInt32)">

<summary>

Преобразует количество секунд в объект Time

</summary>

<param name="totalSeconds">Количество секунд</param>

<c>var Time time = Time.FromSeconds(1000)</c>

<returns>Объект Time</returns>

</member>

</members>

</doc>

**Вывод**: ознакомился с тегами XML, используемыми для документирования. Получил практические навыки создания документации кода C# с помощью XML-комментариев.