**Лабораторная работа № 1**

Тема: Знакомство с визуальной средой программирования Microsoft Visua Studio и языком программирования С #

Цель: Научиться создавать проекты в среде Visual Studio.

**теоретические сведения**

**Платформа .NET Framework** - это управляемое среду, предоставляет различные службы выполняемым в ней программам. Оно состоит из двух основных компонентов: среды CLR (ядро исполнения, которое обрабатывает выполняемые программы), и библиотеки класив.NET Framework (которая предоставляет библиотеку проверенного кода, который можно использовать несколько раз и который можно вызвать из программ). Службы, платформа .NET Framework предоставляет работающим программам, включают следующее:

* **Управление памятью.** В многих языках программирования программисты самостоятельно назначают и выделяют ресурсы памяти вашего компьютера и решают вопросы, "связанные с время жизнь о объектов. В программах платформы .NET Framework среда CLRпредоставляет эти службы от имени программы.
* **Система общего типа.** В традиционных языках программирования, базовые типы определяются компилятором, что затрудняетвзаимодействие между языках. В платформе .NET Framework базовые типы определяются системой типа .NET Framework. при этомиспользуются одни и те же базовые типы для всех как .NET Framework.
* **Расширенная библиотека классов.** вместо того чтобы писать много кода для выполнения стандартных низкоуровневых операцийпрограммирования, разработчики могут использовать легкодоступную библиотеку типов и члены с библиотеки классов .NET Framework.
* **Платформы и технологии разработки.** Платформа .NET Framework включает библиотеки для конкретных областей разработкиприложений, например ASP.NET для веб - приложений, ADO.NET для доступа к данных и Windows Communication Foundation дляпрограмм, ориентированных на службы.
* **Взаимодействие языков.** языковые компиляторы, нацеленные на платформу .NET Framework, дают промежуточный код, называемыйязыке CIL (Common Intermediate Language), который, в свою очередь, компилируется под время выполнения средой CLR. по помощьюэтой функции, подпрограммы пользователя одной языке, доступны в других языках, а разработчики могут сосредоточиться на созданииприложений на любимом языке или языках.
* **Совместимость версий.** по редкими исключениями, программы, разрабатываются за помощью платформы .NET Frameworkопределенной версии, могут выполняться без изменений на более поздней версии.
* **Параллельное выполнение.** Платформа .NET Framework помогает в решении конфликтов версий, позволяя инсталляцию несколькихверсий среды CLR на одном компьютере. это означает, что несколько версий программ также могут сосуществовать, и что программаможет выполняться на версии платформы .NET Framework, для которой она была создана.
* **Настройка для различных версий.** ориентируясь на переносимую библиотеку классов платформы .NET Framework, разработчикимогут создавать код, который работает на нескольких платформах .NET Framework, например .NET Framework, Silverlight, Windows Phone 7 или Xbox 360.

**Создание программного обеспечения в среде Microsoft Visual Studio (на примере Microsoft Visual Studio 2010)**

Рассмотрим для начала возможности данной среды на примере создания консольного приложения.

Поскольку консольные программы выполняют ввод и вывод данных через командную строку, они идеально подходят для быстрого ознакомления с возможностями языка и написания служебных программ командной строки.

**Пример 1.1.** **Консольное приложение Hello World!**

1. Запустите среду Microsoft Visual Studio. В меню Файл выберите команду Создать проект (рис.1.1).

Рисунок 1.1 - Создание нового проекта в среде Visual Studio

1. Откроется диалоговое окно Создание проекта. В этом диалоговом окне выводится список различных типов приложений по умолчанию, которые можно создавать с помощью Visual C #, экспресс-выпуск.

В качестве типа проекта выберите Консольную программу и измените имя приложения (рис. 1.2). Можно оставить расположение проекта по умолчанию или указать новый путь по своему усмотрению.

Рисунок 1.2 - Выбор типа программы

1. Нажмите кнопку ОК.

Visual C #, экспресс-выпуск создаст новую папку для проекта с таким же именем, как у проекта. Откроется главное окно Visual C #, экспресс-выпуск и панель кода для ввода и изменения исходного кода программы C #. (рис. 1.3)

Рисунок 1.3 - Окно редактора кода (обозначено на рисунке красным прямоугольником)

Обратите внимание на панель инструментов в верхней части окна. Панель инструментов содержит значки для создания, загрузки и сохранения проектов, редактирования исходного кода, построения программы, сокрытие и отображение других окон, является частью среды Visual C #, экспресс-выпуск. На правом краю панели инструментов находятся пять значков для открытия важных окон, таких как Обозреватель решений и Панель элементов. При наведении указателя мыши на любой из этих значков появляется всплывающие подсказки.

1. Убедитесь, что отображается Обозреватель решений, щелкнув вкладку Обозреватель решений в правой части экрана или значок Обозреватель решений в панели инструментов (рис. 1.4).

Рисунок 1.4 - Окно обозревателя решений (обозначено на рисунке красным прямоугольником)

Панель Обозреватель решений является очень полезной, на ней отображаются различные файлы, которые являются частью проекта. Важнейшим файлом проекта является "Program.cs", содержащий исходный код программы.

Важно знать, как открывать и скрывать окна, подобные обозревателю решено, это позволит сохранять Visual C # в надлежащем виде. Обозреватель решений по умолчанию просматривается. Чтобы скрыть Обозревателя решений щелкните значок Автоприховування (значок канцелярской кнопки в строке заголовка) или откройте меню Параметры в строке заголовка Обозревателя решений и выберите Автоприховування. Другие окна, такие как Представление классов и Свойства, также имеют эти значки.

1. Введите имя класса **Console** в редакторе кода.

Если обозреватель решений прежнему скрывает панель Код для панель Код для его сокрытия. Теперь щелкните правую открытую фигурную скобку ({) внутри метода Main и нажмите ВВОД для перехода на новую строку. Обратите внимание, что редактор делает отступления автоматически.

Редактор кода всегда пытается форматировать введенный код в стандартной, удобной для восприятия структуре. Если код выглядит некрасиво, можно переформатировать весь документ, выбрав пункты Дополнительно Форматировать документ в меню Правка или нажав клавиши CTRL + E, D.

При вводе имени класса или ключевого слова C # есть выбор: либо закончить ввод слова самостоятельно, или позволить сделать это встроенному в панель Код инструмента IntelliSense. Например, при вводе символа "c" появится всплывающие список слов, предлагаемых IntelliSense, из которого можно выбрать слово, набирается. В этом случае, слово "Console" пока еще не видно. Или прокрутите вниз список, или продолжайте вводить слово "console". Когда слово "console" будет выделено в списке, нажмите клавишу ENTER, или TAB, или дважды щелкните мышью, Console будет добавлено в код (рис. 1.5).

Рисунок 1.5 - Использование Intellisence

Преимуществом использования IntelliSense является уверенность в правильности выбора регистра и написания слова. Пользователь может либо сам вводить код, или использовать технологию IntelliSense.

1. Введите точку и имя метода **WriteLine.**

Рисунок 1.6 - Использование Intellisenceдля выбора метода класса

Как только пользователь введет точку после Console, IntelliSense отобразит другой список. Список содержит имена всех возможных методов и свойств, является частью класса Console. Необходим метод WriteLine можно увидеть в конце списка. Или закончите ввод слова WriteLine, или нажмите клавишу СТРЕЛКА ВНИЗ для выбора, затем ENTER, или TAB, или дважды щелкните мышью. Метод WriteLine будет добавлен в код (рис. 1.6).

Введите открывающую скобку. Появится сообщение в виде всплывающей подсказки с надписями методов, еще одной функцией IntelliSense. В этом случае отображается 19 различных надписей, которые можно просмотреть с помощью клавиш СТРЕЛКА ВВЕРХ и СТРЕЛКА ВНИЗ.

1. введите или вставьте следующий код:

static void Main (string [] args)

{

Console.WriteLine ( "Hello World!");

// Сохранить окно консоли открытым до нажатия любой клавиши

Console.WriteLine ( "Press any key to exit. ');

Console.ReadKey ();

}

Последняя строка в программе Console.ReadKey () предназначен для приостановления исполнения программы до нажатия любой клавиши. Если не добавить эту строку, окно с командной строкой сразу же истребляет и пользователь не сможет увидеть вывода результата выполнения программы. Если создаваемая служебная программа всегда будет использоваться в консоли командной строки, вызов метода Console.ReadKey () можно опустить.

1. выполните программу.

Теперь программа завершена, готова для компиляции и выполнения. Для этого либо нажмите клавишу F5, или воспользовавшись кнопкой Пуск на панели инструментов (рис. 1.7).

Рисунок 1.7 - Кнопка «Запуск» (выделена красным прямоугольником)

1. После компиляции и выполнения откроется окно к онсол и (р ис. 1.8).

Рисунок 1.8 - Результат компиляции и выполнения программы

**Пример 1.2.** **Windows-приложение простой калькулятор**

Рассмотрим пример создания простого калькулятора (с графическим интерфейсом пользователя) в среде VisualStudio.

Последовательность действий для создания калькулятора:

1) Нажмите Файл → Создать проект ... Откроется окно, а котором надо выбрать WindowsFormsApplication

2) На открывшейся форме после предыдущего действия, следует разместить 3 текстовых поля (объект TextBox) и 5 ​​кнопок (Button). Объекты, которые можно разместить на форме находятся в панели ToolBox (рис. 1.9).

Рисунок 1.9 - Панель ToolBox *(выделена красным прямоугольником)*

3) В кнопок надо изменить свойство Text на следующие значения:

* + В button1 свойство Text - "CE";
  + В button2 свойство Text - "\*";
  + В button3 свойство Text - "/";
  + В button4 свойство Text - «-»;
  + В button5 свойство Text - "+".

Свойства меняются на панели свойств, которая выглядит следующим образом:

Рисунок 1.9 - Панель свойств (выделена красным прямоугольником)

После указанных действий должно получиться следующее изображение:

Рисунок 1.9 - Внешний вид программы Калькулятор

4) Щелкните два раза в созданной кнопке "\*" (умножение). Откроется окно Редактора кода, в котором курсор мыши будет расположен внутри функции, обрабатывает нажатия на кнопку "\*". Добавьте туда следующий код:

double result = Convert.ToDouble (textBox1.Text) \* Convert.ToDouble (textBox2.Text)

В этом коде к переменной result с типом double присваивается значение текстовых полей, которые надо умножить между собой.

Чтобы получить значение текстового поля пишем имя текстового поля (textBox1), и свойство Text (textBox1.Text).

Но почему мы не написали так: textBox1.Text \* textBox1.Text?

А так как в поле храниться текст, а при умножении нужно использовать числа.

Поэтому с помощью функции Convert.ToDouble мы меняем тип переменной со срочной в числовую (с плавающей точкой, так как возможно придется умножать дроби).

Далее результат, храниться в переменной result записываем в текстовое поле 3, при этом меняем тип переменной назад в числовом в срочный:

textBox3.Text = result.ToString ();

5) Пример обработки ошибки ввода:

private void button2\_Click (objectsender, EventArgs e)

{

// Если ничего не введено

if (textBox1.Text == "")

MessageBox.Show ( "Данные не указаны")

else

{

... // Осуществляем умножения

}

}

6) Для очистки текстовых полей калькулятора применим следующий код:

// Очищаем текстовые поля (кнопка "СЭ")

private void button1\_Click (object sender, EventArgs e)

{

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox4.Text = "";

}

**Задания лабораторноироботы:**

Создать программу «Калькулятор» на языке C # с графическим интерфейсом, которая должна выполнять четыре базовые арифметические действия (+, -, \*, /), а также операции х 2, х! и вычисления остатка от деления (%).

***Контрольнизапитання:***

1. Опишите особенности платформы .NetFramework 4.5. Ее преимущества и недостатки по сравнению с другими платформами.
2. Другие платформы для работы с визуальными языками программирования вы знаете?
3. Как создать программу в среде Visual Studio?

https://www.gstatic.com/images/branding/googlelogo/1x/googlelogo_color_48x16dp.png