Практическая работа 5
Проведение отладки программного модуля

Цель занятия: Получить практический опыт использования точек останова для отладки модулей

Перечень оборудования и программного обеспечения

Персональный компьютер
Microsoft Office (Word, Visio)
Microsoft Visual Studio 2015

Краткие теоретические сведения

Ошибки компиляции помогает отловить компилятор. В Visual Studio средства отладки достаточно мощные, чтобы найти логические ошибки.

Пример отладки программы в среде разработки Microsoft Visual 2010.

На форме размещена кнопка и событию Click для кнопки запишем код:

```
private void MyButton_Click(object sender, EventArgs e)
{
  int x = 10;
  int y = 15;
  x = Double(x);
  x = x / (y - 15);
   MessageBox.Show(x.ToString());
}
```

В этом коде в третьей строке происходит вызов метода Double. Это не какой-то стандартный метод, его нужно написать в коде формы:

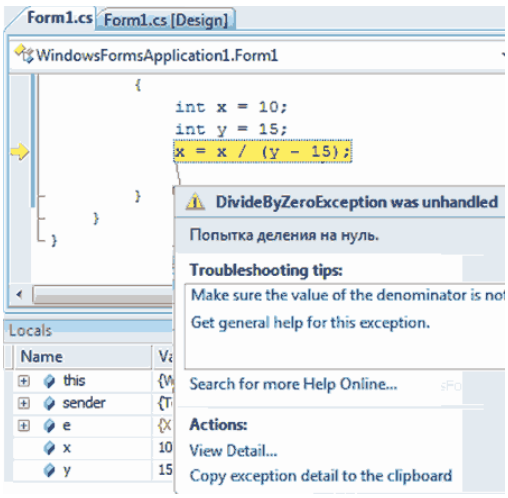
```
int Double(int x)
 {
   return x * 2;
 }
```

Полный исходный код формы будет выглядеть следующим образом:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        // конструктор
        public Form1()
        {
            InitializeComponent();
        }
        // обработчик события Click для кнопки
        private void MyButton_Click(object sender, EventArgs e)
        {
            int x = 10;
            int y = 15;
            x = Double(x);
            x = x / (y - 15);
            MessageBox.Show(x.ToString());
        }
        // наш метод Double
        int Double(int x)
        {
            return x * 2;
        }
    }
}
```

Если запустить программу из среды разработки и нажать на кнопку, то выполнение программы переключится на среду разработки и вы увидите сообщение об ошибке:



Произошла ошибка, и среда разработки взяла управление на себя, чтобы вы смогли просмотреть информацию об ошибке и попробовали

определить ее источник. В редакторе кода желтым цветом выделена строка кода, в которой и произошла ошибка. От этой строки идет стрелка к всплывающему окну, в котором исключительная ситуация описана более подробно.

Название ошибки можно увидеть в заголовке всплывающего окна. Чуть ниже под заголовком находится описание ошибки: деление на ноль.

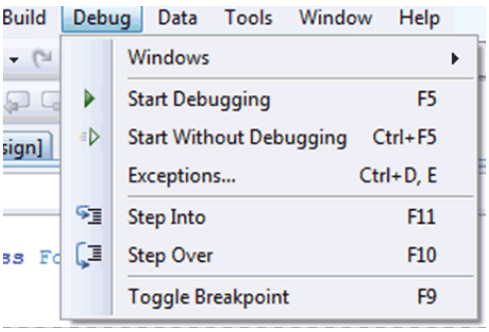
В списке Troubleshooting tips этого окна можно увидеть возможные подсказки, которые могут помочь вам найти возможное решение. В данном случае проблема проста и с ней можно разобраться без дополнительных советов, потому что в строке произошло деление на ноль и наша задача добавить перед выполнением кода проверку, которая предотвратит деление на 0. В данном случае, нужно проверить, чтобы переменная y не была равна 15:

```
int x = 10;
int y = 15;
if (y == 15)
  MessageBox.Show("Переменная y не должна быть равна 15 ");
 x = x / (y - 15);
 MessageBox.Show(x.ToString());
```

Но определение ошибки далеко не всегда так легко определимо. Бывают случаи, когда нужно пройти программу пошагово, строчка за строчкой, в поисках уязвимого места, которое привело к проблеме. Заранее определим блок кода, который нужно проанализировать. В нашем случае этим блоком кода будет обработчик события Click для кнопки.

Конфигурация компиляции

Для того, чтобы отладка программы была доступна, программа должна быть еще запущена в режиме отладки и из среды разработки. Простое нажатие клавиши F5 как раз запускает программу в отладочном режиме. Чтобы запустить программу без возможности отладки, нужно нажать Ctrl+F5. Соответствующие пункты меню можно найти в меню Debug:



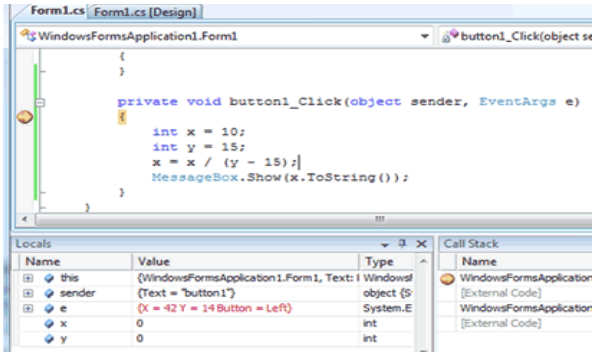
Debug | Start Debugging запустить в режиме отладки

Debug | Start Without Debugging запустить в режиме выполнения и без возможности отладки

Для отладки программа должна быть скомпилирована в конфигурации Debug, которая включает в исполняемый файл дополнительную информацию, необходимую при отладке программы. Конфигурацию можно выбрать на панели инструментов чуть правее кнопки запуска программы.

Точки останова

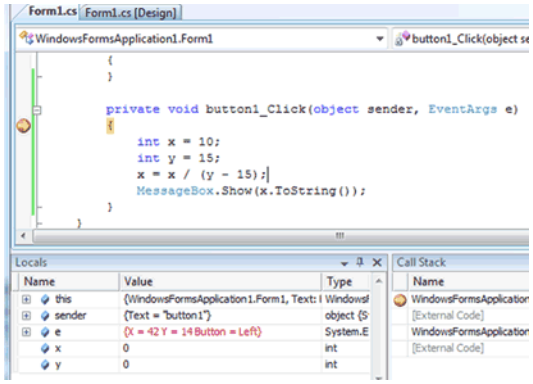
Нужно отладить метод и мы сделаем это с самого начала. Для этого нужно поставить точку прерывания в нужном нам месте. Точка прерывания - это точка в коде программы, при достижении которой выполнение программы будет прервано и управление будет передано среде разработки. Вы можете создавать точки останова в любом месте программы, но только там, где есть код и программа может прервать выполнение. Для создания точки перейдите на нужную строку и нажмите F9; выберите меню Отладка | Точки останова; дважды щелкните на полоске серого цвета, слева от строки текста в окне редактора кода



Напротив строки на полоске серого цвета слева от текста появиться красный кружок, символизирующий, что здесь стоит точка останова. Если еще раз попытаться поставить точку останова на этой же строке, то точка останова будет снята.

Так как мы отлаживаем метод с самого начала, то поставьте курсор на строку с именем метода и нажмите F9. Теперь можно запускать программу. Кстати, запустить программу можно было и раньше, потому что точку останова можно ставить в любой момент, даже во время выполнения программы.

Запустите программу и нажмите на кнопку. После нажатия кнопки должен сработать метод button1_Click. Так как мы поставили точку останова, среда разработки перехватит на себя выполнение и выделит оператор, который можно выполнить следующим шагом желтым цветом (назовем эту точку курсором пошагового выполнения программы):



Обратите внимание, что выделена строка с символом открывающейся фигурной скобки. Это потому, что выполнение метода начинается именно с фигурной скобки, а не с имени, где поставлена точка останова. Да, точка останова иногда может смещаться, но если вы поставите ее на конкретный оператор, который может быть выполнен, то точка останется там и никуда не поедет.

Внизу окна среды разработки появилось две панели:

- **Локальные переменные,** где в виде списка представлены все переменные, доступные в текущем методе. В этом списке переменные представлены в виде трех колонок:
  - Name - название переменной
  - Value - значение переменной
  - Type - тип
- Стек вызовов. В этом окне перечислены методы, которые уже были вызваны ранее.

Пошаговое выполнение

Итак, наш курсор пошагового выполнения остановился на какой-то точке и теперь хотим начать тестирование кода. Посмотрим на панель, где появилась новая панелька отладки, там же мы и найдем необходимые команды (в скобках указаны горячие клавиши):

- **Continue (F5)** - продолжить выполнение программы.
- **Stop debugging (Shift+F5)** - остановить отладку. При этом остановится и выполнение программы. Даже больше - выполнение программы прервется на той точке, на которой сейчас и находиться, т.е. оно не будет завершено корректно и ничего не сохранится, если вы в программе что-то делали;
- **Restart (Ctrl+Shift+F5)** - перезапустить программу. Выполнение программы будет прервано и запустится заново;
- **Show Next Statement (Alt + Num *)** - показать следующий оператор, т.е переместить курсор редактора кода в курсор пошагового выполнения. Курсор переместиться в начала оператора, который должен быть выполнен следующим. В редакторе кода он выделен желтым цветом;

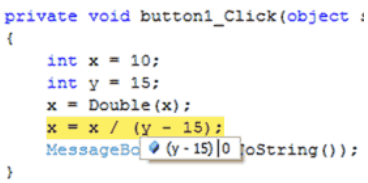
- **Step Into (F11)** - выполнить очередной оператор. Если это метод, то перейти в начало этого метода, чтобы начать отладку. Например, если вы находитесь на строке: x = Double(x) то, курсор пошагового выполнения перейдет на начало метода Double и вы сможете отладить этот метод;
- **Step Over (F10)** - выполнить очередной оператор. Если это метод, то он будет полностью выполнен, т.е. курсор выполнения не будет входить внутрь метода;
- **Step out (Shift + F11)** - выйти из метода. Если вы отлаживаете метод и нажмете эту кнопку, то метод выполнится до конца и курсор пошагового выполнения выйдет из метода и остановится на следующей строке после вызова данного метода. Например, если вы отлаживаете метод Double, нашего примера и нажмете эту кнопку, то метод выполнится до конца, а выполнение остановится на строке "x = x / (y - 15);" метода button1_Click.

Попробуйте пошагово выполнить код метода, нажимая клавишу **F10**. Потому запустите снова приложение и попробуйте пошагово выполнить его, нажимая клавишу F11. Когда произойдет ошибка, попробуйте прервать работу отладки и приложения, нажав **Shift+F5**.

Просмотр значений

Выполните пошагово код так, чтобы курсор выполнения остановился на следующей строке: x = x / (y - 15);

Нам нужно узнать, почему происходит ошибка. Так как ошибка в делении, то нужно просмотреть, на что происходит деление. В данном случае это скобка. Наведите мышкой на открывающуюся или закрывающуюся скобку и вы увидите всплывающую подсказку, в которой находится результат вычисления значения в скобках:



Результат находится после вертикальной черты и он равен нулю. Вы можете наводить мышкой на любую переменную, и отладчик покажет вам ее значение в виде всплывающей подсказки.

Посмотрите на панель Локальные переменные. По идее внизу у вас должны быть еще одна закладка Watch. Переключитесь на нее. Здесь находится список, в который вы можете вносить свои переменные и выражения, за значениями которых вы хотите наблюдать. Выделите первую пустую строку в списке (всегда одна пустая строка присутствует в списке) и в поле Name введите y*2 и нажмите Enter. В поле Value появится значение выражения. После каждого шага выполнения значение переменной будет пересчитано и отображено. Таким образом, вам не нужно после каждого шага наводить на переменную, чтобы узнать ее значение.

На закладке Локальные переменные видны переменные, которые актуальны для данного метода. Сюда включаются переменные, объявленные внутри метода и параметры метода, а так же переменная this. Параметры представлены в виде дерева для объектных типов данных. Раскрывая дерево объекта this вы можете увидеть значения всех свойств и даже объектов на текущей форме, ведь this всегда указывает на текущий объект, которым является форма:

Заключение

Отладка - это мощное средство поиска любого вида ошибок. Я надеюсь, что этот документ поможет вам разобраться с этим процессом. Да, документ нельзя назвать основательным, но я постарался включить в него все основные нюансы, которые вам могут пригодиться в реальной работе.

Задания

1 Изучить теоретические сведения и задание к работе

2 В соответствии с вариантом задания разработать отлаженный модуль.

Порядок выполнения работы

Содержание отчета

1. Название работы
2. Цель работы
3. Перечень оборудования и программного обеспечения
4. Порядок выполнения
5. Вывод.

Варианты заданий

Варианты заданий представлены в практической работе 1.

Используемая литература

1 Рудаков А.В. Технология разработки программных продуктов: учеб. пособие для студ. учреждений сред. проф. образования – 11-е изд., стер. – М.: Издательский центр «Академия», 2017. – 208 с.

2 Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: учеб. пособие для студ. учреждений сред. проф. образования / Г.Н. Федорова.–: М.: КУРС : ИНФРА – М, 2017. – 334 с

3 Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: Учебное пособие для СПО.- М.: КУРС, 2018. – 333 с.– ЭБС Знаниум

4 http://msdn.microsoft.com/ru-ru/library/67ef8sbd.aspx.