Стилове за създаване на интерфейс

Лекция 8

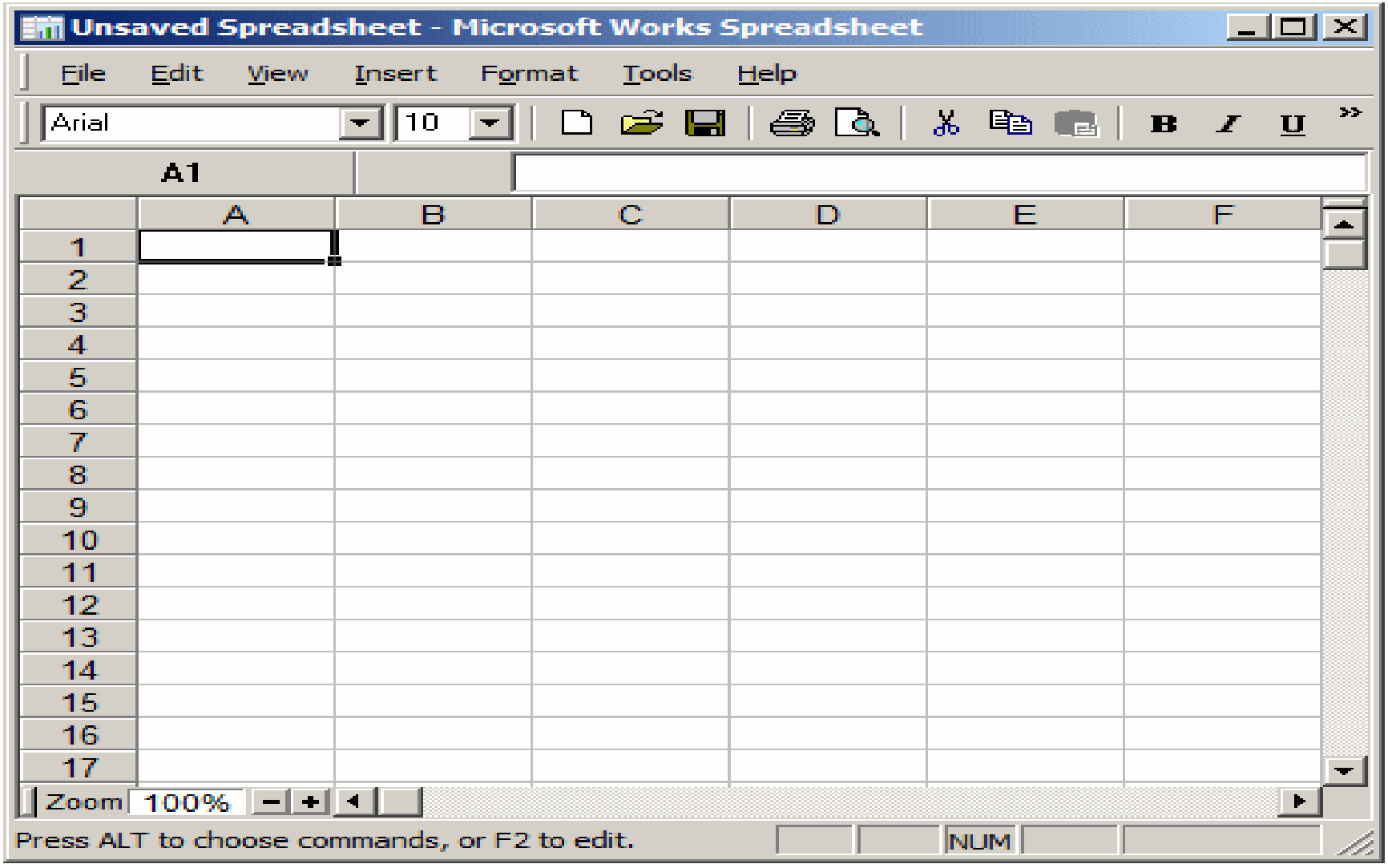
Съдържание

- Стилове за дизайн на графичен интерфейс
- Интерфейс с един документ – SDI (Single-Document Interface). Примери.
- Интерфейс с много документи - MDI (Multiple Interface Document). Примери.
- Модални и немодални форми.

Интерфейс с един документ – SDI Single-Document Interface

- **SDI** е приложение, което се състои основно **от** **една форма**, съдържаща меню
  - например текстов редактор Notepad
- **SDI** е **метод за организиране на приложения** с ГПИ в отделни прозорци, които се управляват от операционната система **отделно**
- Всеки прозорец съдържа **собствено меню или лента с инструменти**
- Всички прозорци са независими един от друг
- В някои случаи, SDI също може да има **лента с инструменти и/или лента на състоянието**

Single Document Interface



Single Document Interface

- Въпреки, че **Notepad е текстово-базиран**, един **SDI** може да бъде **всеки вид приложение**: текст, графики, таблици, **Label, TextBox, ComboBox, ListView, TreeView, Button, MenuStrip, и др.**
- Следователно, за да се **създаде SDI**, се започва от разработване на **една нормална форма**, добавя се **меню** към нея, което се конфигурира да изпълнява действията и командите, които искате
- За изграждане на SDI се използват **MenuStrip** и **Button GUI controls**
- За да създадете друг документ от същия вид, трябва да отворите **друг екземпляр на приложението**

Интерфейс с много документи – MDI Multiple Document Interface

- **MDI** приложенията поддържат **работа с няколко документа едновременно,** като всеки документ се показва в **свой собствен прозорец**, разположен във **вътрешността на главния прозорец**
- **MDI** е вид ГПИ, който представлява **един прозорец – родител (container), който е контейнер** за **други прозорци (child)** в специфично приложение
- Само **прозорецът - родител притежава меню или лента с инструменти**

Multiple Document Interface

- **MDI** позволява на **дъщерните прозорци (child windows) да вграждат** други прозорци **вътре в тях,** както и създаване **на сложни вложени йерархии**

MDI контейнери (MDI parents)

- **MDI** контейнерите са **форми**, които **съдържат други форми**
- За да укажем, че една форма е **MDI контейнер**, задаваме на нейното свойство **IsMdiContainer** стойност **true**
- Тези форми обикновено имат меню **Window** за смяна на активната форма (на свойството му **MdiWindowListItem** е зададена стойност **windowToolStripMenu)**

Multiple Document Interface

MDI форми (MDI children)

- **MDI** формите се **съдържат в контейнер-формата**
- За да укажем, че една форма е **MDI форма**, задаваме на свойството **MdiParent = <контейнер>**, където **контейнер** е MDI форма контейнер
- **Например:**

```
//декларираме нова форма като Child_Form1
  Child_Form1 childform = new Child_Form1();
//Задаваме главната форма като родител container
  childform.MdiParent = this;
```

Или

```
childform.MdiParent = MDI_Container.ActiveForm;
```

Основни предимства на MDI

- **Прозорците наследници (child windows)** се **управляват** лесно от **една родителска (container)** форма
- **Едно меню** и лента с инструменти може да бъде **споделено с други прозорци**
- **Възможността** да се работи с **множество документи** от **един прозорец** на същото приложение
- **Чрез затварянето на родителския прозорец (container)**, потребителят **затваря** и другите **дъщерни прозорци** (child windows)

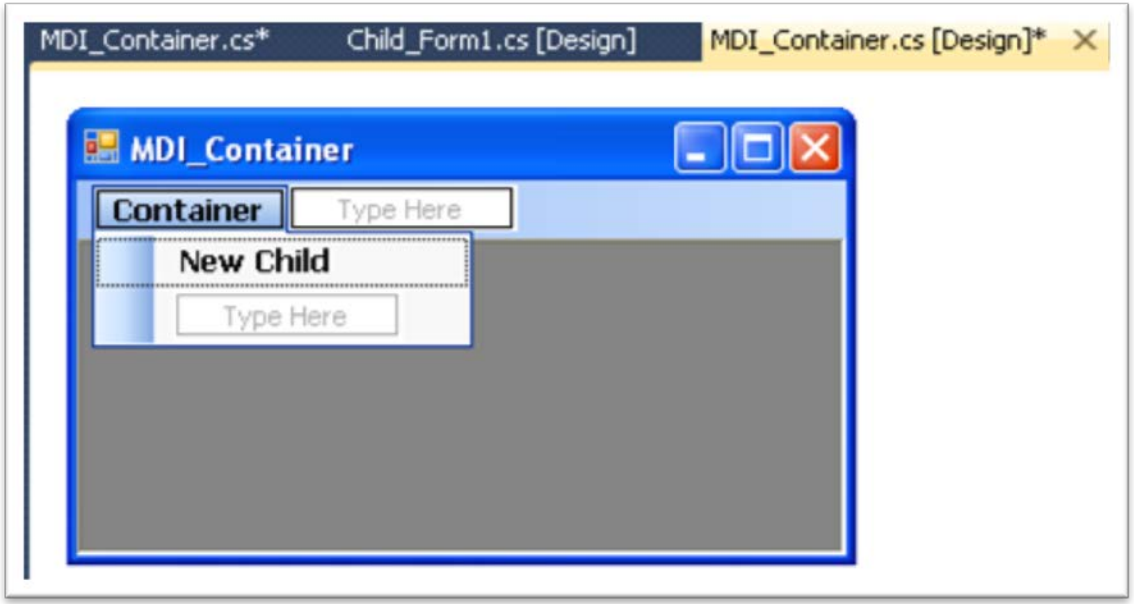
Създаване на MDI приложение MDI_Forms

1) Създаваме стандартно **C# Windows Forms Application.**
2) На **първо място** трябва да се **създаде форма контейнер.** Първата форма, която е създадена по подразбиране може да се **превърне в контейнер** само чрез **промяна на свойството IsMdiContainer = True**
3) Ще **добавим нова форма** към проекта. Тя ще бъде **дъщерна форма (child form).** Използваме ***Add New Item*** бутон от **IDE toolbar.** Избираме опция ***Add Windows Form...*** от падащото меню.
4) След като е създадена новата форма, трябва тя да **бъде извикана от родителската форма.** Ще създадем меню във формата-контейнер чрез контрола ***MenuStrip***

Създаване на MDI приложение

MDI_Forms

5) Създаваме към менюто основен елемент **"Container" и добавяме един поделемент (child element) "New Child...":**



Създаване на MDI приложение

- За да отворим елемента **“New child” от родителската форма, добавяме следния код към "New Child..." menu item:**

```
private void newChildToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            //декларираме нова форма като Child_Form1
            Child_Form1 childform = new Child_Form1();

            //Задаваме главната форма като родител container
            childform.MdiParent = this;

            //Показване на дъщерната форма
            childform.Show();
        }
```

Създаване на MDI приложение

- Ще добавим към основното меню няколко допълнителни опции, които са необходими, за да визуализираме **подреждането на дъщерните форми вътре в родителската форма**

- Има **четири типа подреждане на формите:**
  - **Каскадно**
  - **Хоризонтални редове**
  - **Вертикални колони**
  - **Подредени икони**

Подреждане на формите

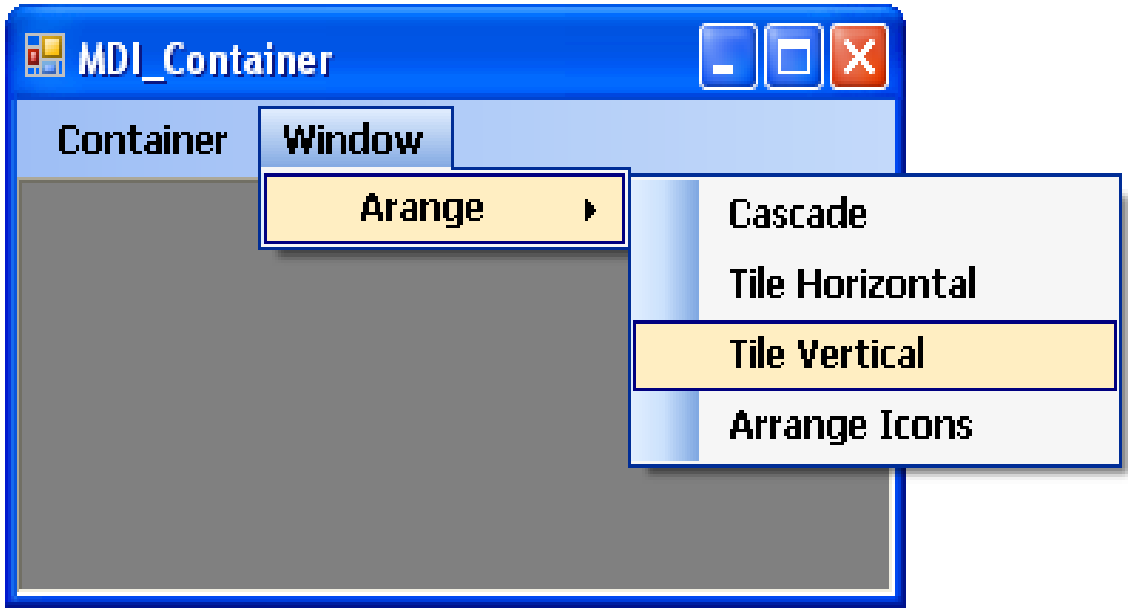
- Операционната система позволява на потребителя да избира между **4 различни** начина на подреждане
- За да реализира това, класът **Form** предоставя метод, наречен **LayoutMdi()**
- Синтаксисът му е следният:

```
public void LayoutMdi(MdiLayout value);
```

- Методът **LayoutMdi()** изисква **един аргумент**, който е член на **MdiLayout** enumeration
- Членове на това множество са:
  - **Cascade**
  - **TileHorizontal**
  - **TileVertical**
  - **ArrangeIcons**

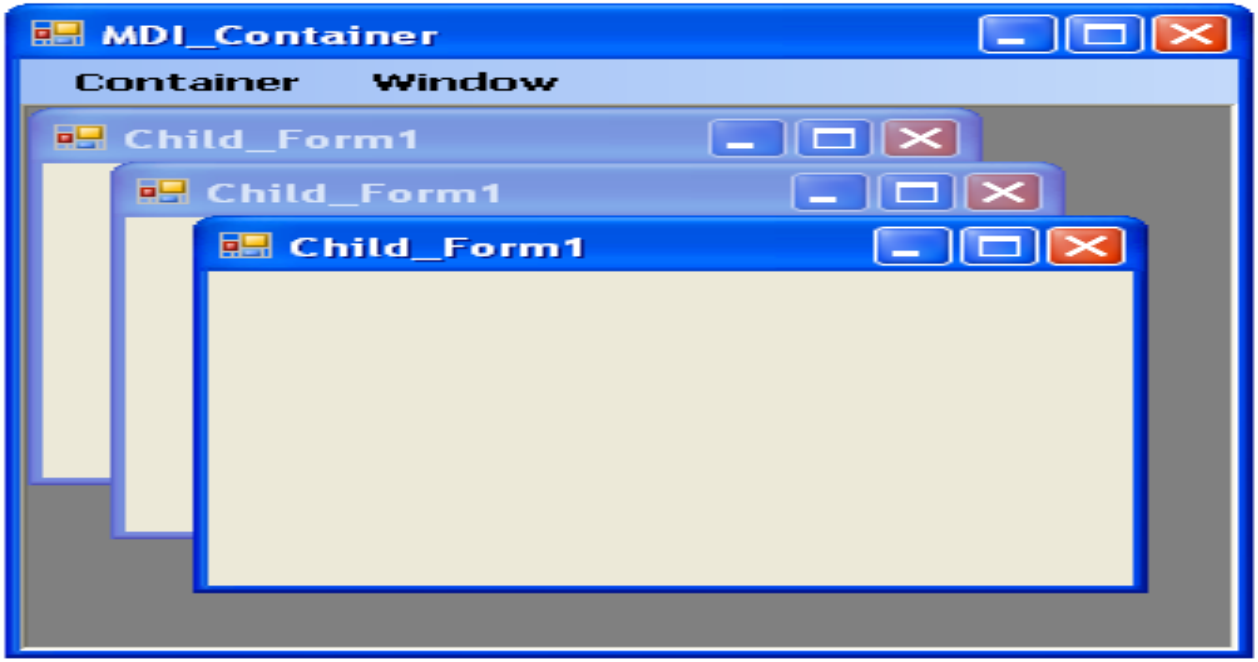
Създаване на MDI приложение

- Към менюто добавяме елемент **Window**, подменю **Arrange и четири елемента: Cascade, Tile Horizontal, Tile Vertical, Arrange Icons**



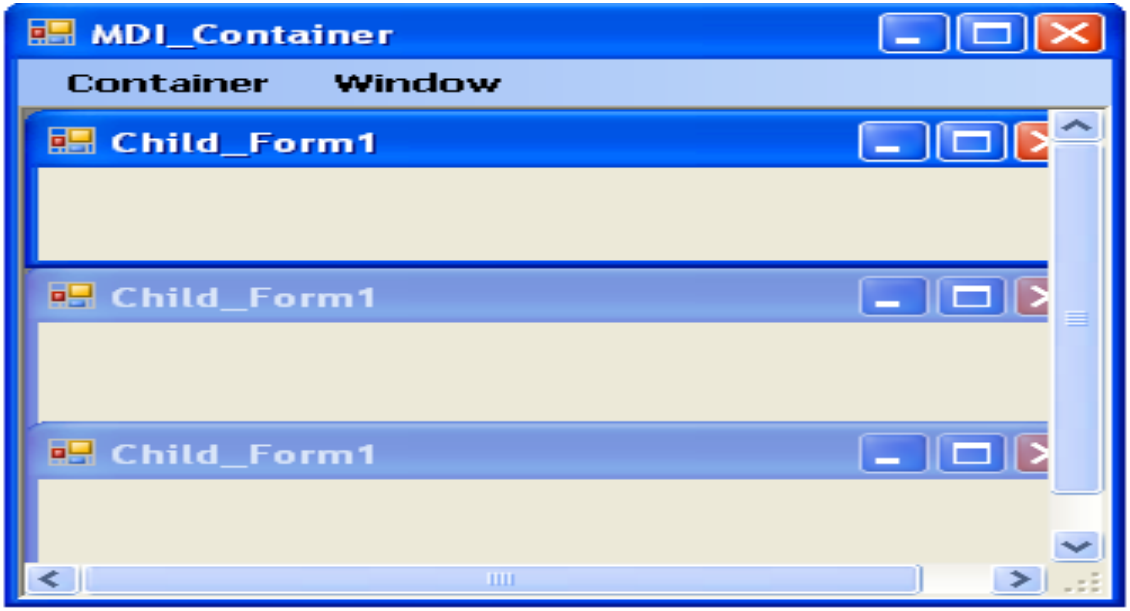
Каскадно подреждане (Cascade)

```
private void cascadeToolStripMenuItem_Click(object sender,
                                            EventArgs e)
        {
            this.LayoutMdi(MdiLayout.Cascade);
        }
```



Хоризонтално подреждане (Tile Horizontal)

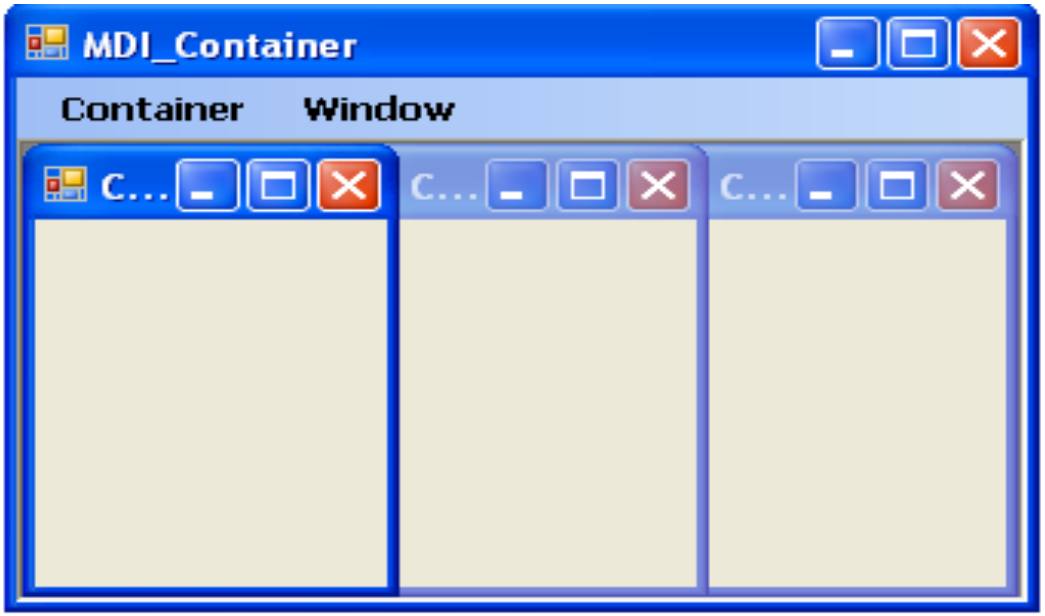
```
private void tileHorizontalToolStripMenuItem_Click(object sender,
                                                   EventArgs e)
   {
        this.LayoutMdi(MdiLayout.TileHorizontal);
   }
```



}

Вертикално подреждане (Tile Vertical)

```
private void tileVertikalToolStripMenuItem_Click(object sender,
                                                 EventArgs e)
    {
        this.LayoutMdi(MdiLayout.TileVertical);
    }
```



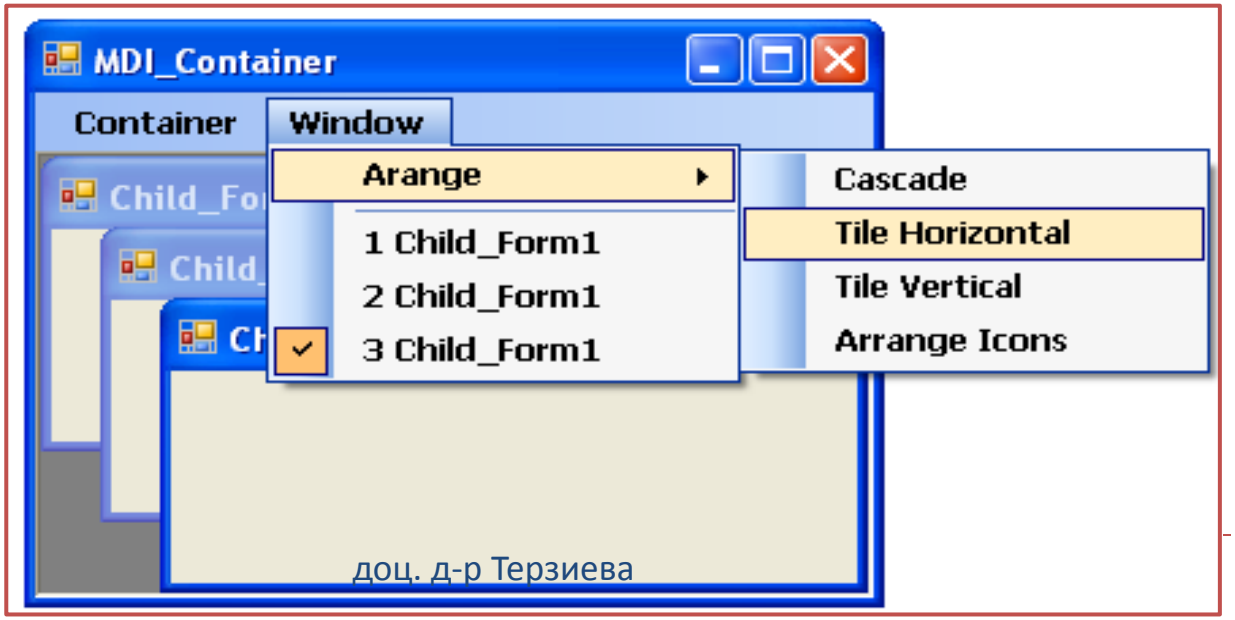
Подредени икони (Arrange Icons)

```
private void arrangeIconsToolStripMenuItem_Click(object sender,
                                                 EventArgs e)
    {
        this.LayoutMdi(MdiLayout.ArrangeIcons);
    }
```

- **Забележка:** Оформлението **на Arrange Icons** е **достъпно само за минимизирани деца форми**

Управление на формите

- Ако потребителят отвори **много форми** деца, то става по-трудно придвижването между тях
- За да се визуализират всички форми на **свойството** **MdiWindowListItem** на менюто **(menuStrip**), задаваме **windowToolStripMenuItem**, така че всички отворени прозорци деца да бъдат изброени в меню Window:



Създаване на MDI приложение в Microsoft Visual Studio 2013 MDI_Application

- Създайте нов проект **New Project → Name = MDI_Application...**
- За да добавите нова форма, от главното меню изберете **PROJECT → Add Windows Form...** или кликнете с десен бутон върху **MDI_Application** на Solution Explorer и изберете **Add → Windows Form..**
- Задайте име на новата форма **Name = SampleDocument** и изберете **Add**.
- От секцията с основни контроли Common Controls на средствата с инструменти (Toolbox), изберете контрола **TextBox** и я позиционирайте във формата.
- Задайте следните свойства на текстовата кутия:
  Text: **Information**
  Dock: **Fill**
  Multiline: **True**
  ScrollBars: **Vertical**

Създаване на MDI приложение в Microsoft Visual Studio 2013 MDI_Application

- За да добавите нова форма от главното меню изберете **PROJECT → Add New Item...** и изберете от списъка **MDI Parent form**
- Задайте **Name = MDIParentDocument и изберете Add**
- Свойството **Text = MDIParentDocument**
- Кликнете с десен бутон на произволно място във формата и изберете **View Code**

Направете следните промени в кода:

```
public partial class MDIParentDocument : Form
{
   private int childFormNumber = 1;

   public MDIParentDocument()
   {
      InitializeComponent();
      ShowNewForm(null, null);
}
```

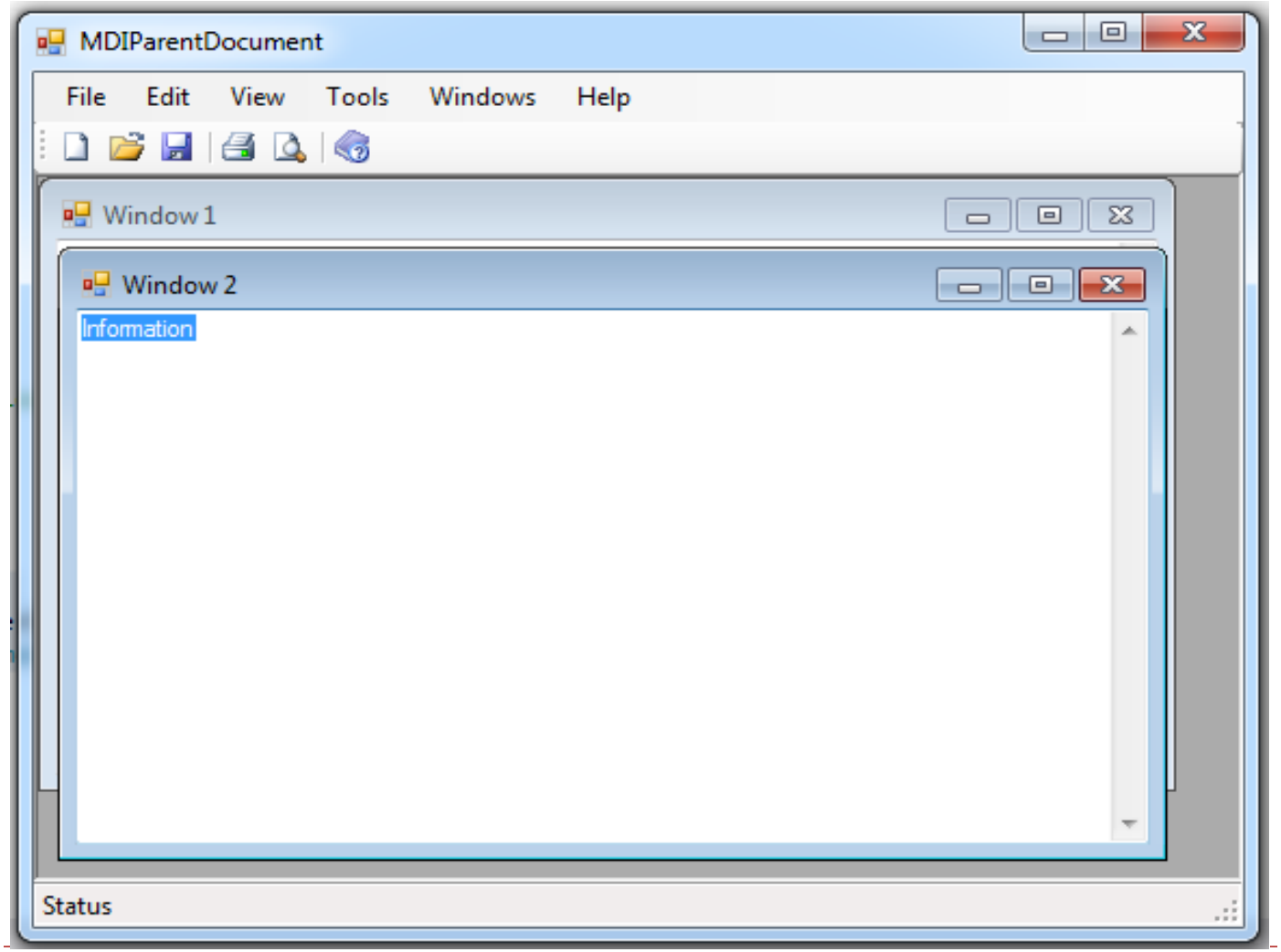
```
private void ShowNewForm(object sender, EventArgs e)
{
     SampleDocument childForm = new SampleDocument();
     childForm.MdiParent = this;
     childForm.Text = "Window " + childFormNumber++;
     childForm.Show();
}
```

Създаване на MDI приложение в Microsoft Visual Studio 2013

- От **Solution Explorer**, кликнете два пъти върху **Program.cs** или с десен бутон изберете **View Code** и направете следните промени:

```
    static void Main()
     {
          Application.EnableVisualStyles();
          Application.SetCompatibleTextRenderingDefault(false);
          Application.Run(new MDIParentDocument());
     }
```

Получава се следното приложение:



Модални и немодални форми

- **Модална форма** е тази, която при показването си прави **неактивни всички останали форми** на приложението и позволява достъпът до тях единствено след нейното затваряне

- **Немодална форма** е тази, която може да бъде използвана **едновременно с другите форми** на приложението, като фокусът по всяко време може да бъде превключван между тях

Употреба на модални форми

- **Модалните форми** се използват тогава, когато е нужно потребителят да **избере или укаже нещо** преди да продължи нататък - например в диалози за настройки
- Например: Потребителя избира бутон (**ОК, Cancel и други**), за да продължи работата си
- Всеки опит за преминаване в друга форма не успява, докато **потребителят не затвори модалната форма**
- Модалността може да **се задава първоначално**, но не може да се променя след като формата е вече показана

Създаване на модални форми - пример

1. Създаваме приложение с нова форма, задаваме свойство **Name = mainForm, Text = Main Form**
2. Добавяме втора форма към приложението

   Add New Item – Name = subForm
3. Добавяме по два бутона във всяка форма
4. От бутона **Name = buttonOK** на главната форма **mainForm** създаваме събитието **Click**

```
private void buttonOK_Click(object sender, EventArgs e)
    {
        subForm newForm = new subForm();
        newForm.ShowDialog();
    }
```

Създаване на модални форми - пример

5. Във втората форма създаваме бутони **OK и Cancel**
6. Създаваме събитие **Click на двата бутона**

```
// бутони на subForm
private void buttonOK_Click(object sender, EventArgs e)
    {
        this.DialogResult = DialogResult.OK;
    }

private void buttonCancel_Click(object sender, EventArgs e)
    {
        this.DialogResult = DialogResult.Cancel;
    }
```

Употреба на немодални форми

- **Немодалните форми** се използват, когато е нужно **няколко форми да са видими и достъпни едновременно на екрана**
- Например: при палитри с инструменти
- Трябва да се осигури обновяване на състоянието на всяка форма спрямо това на останалите

Създаване на немодална форма - пример

1. Създаваме приложение с нова форма, задаваме свойство **Name = mainForm, Text = Main Form**
2. Добавяме втора форма към приложението

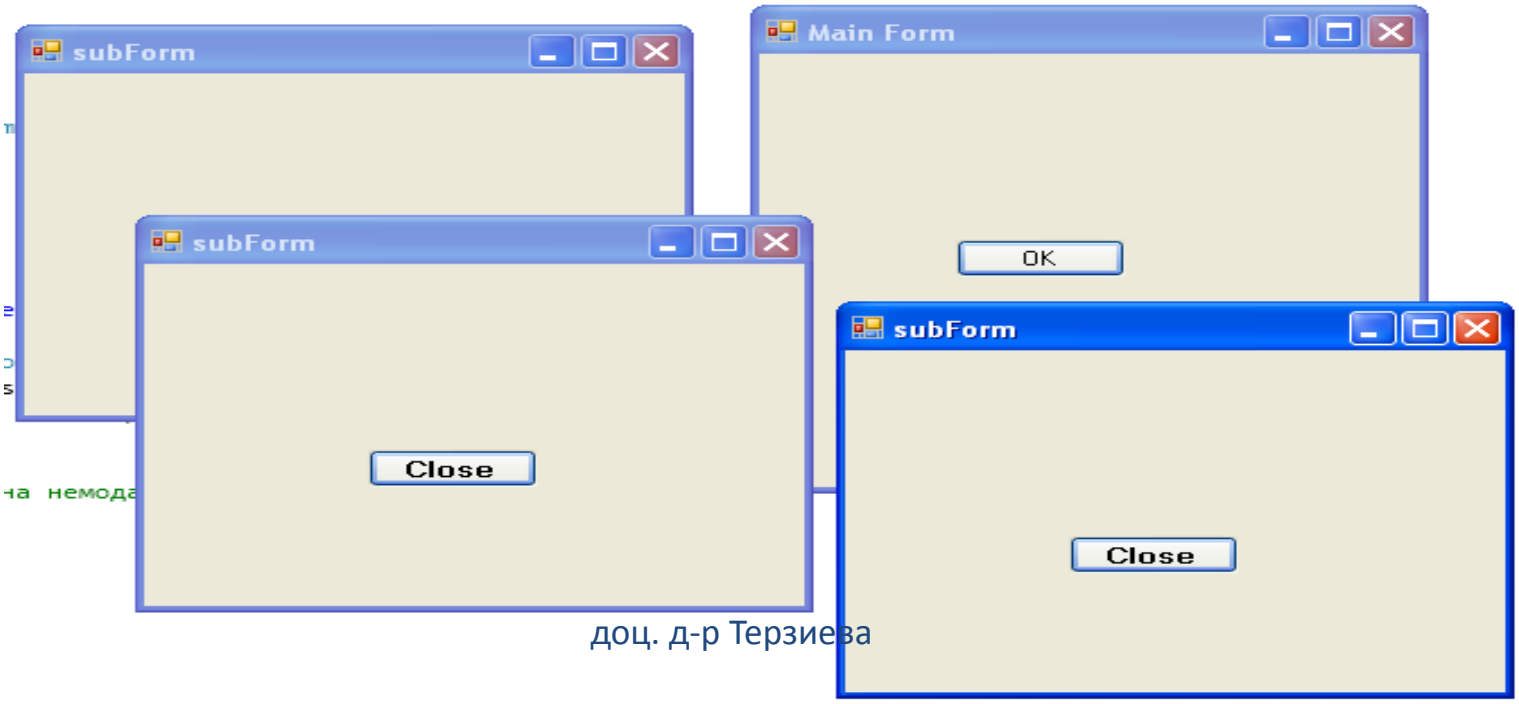
   Add New Item – **Name = subForm**
3. Добавяме по един бутон във всяка форма
4. От бутона **Name = buttonOK** на главната форма **mainForm създаваме събитието** Click

```
private void buttonOK_Click(object sender, EventArgs e)
        {
            subForm myNewForm =new subForm();
            // Първи начин за създаване на немодална форма
            myNewForm.Show();
            //Втори начин за създаване на немодална форма
            myNewForm.Visible = true;
        }
```

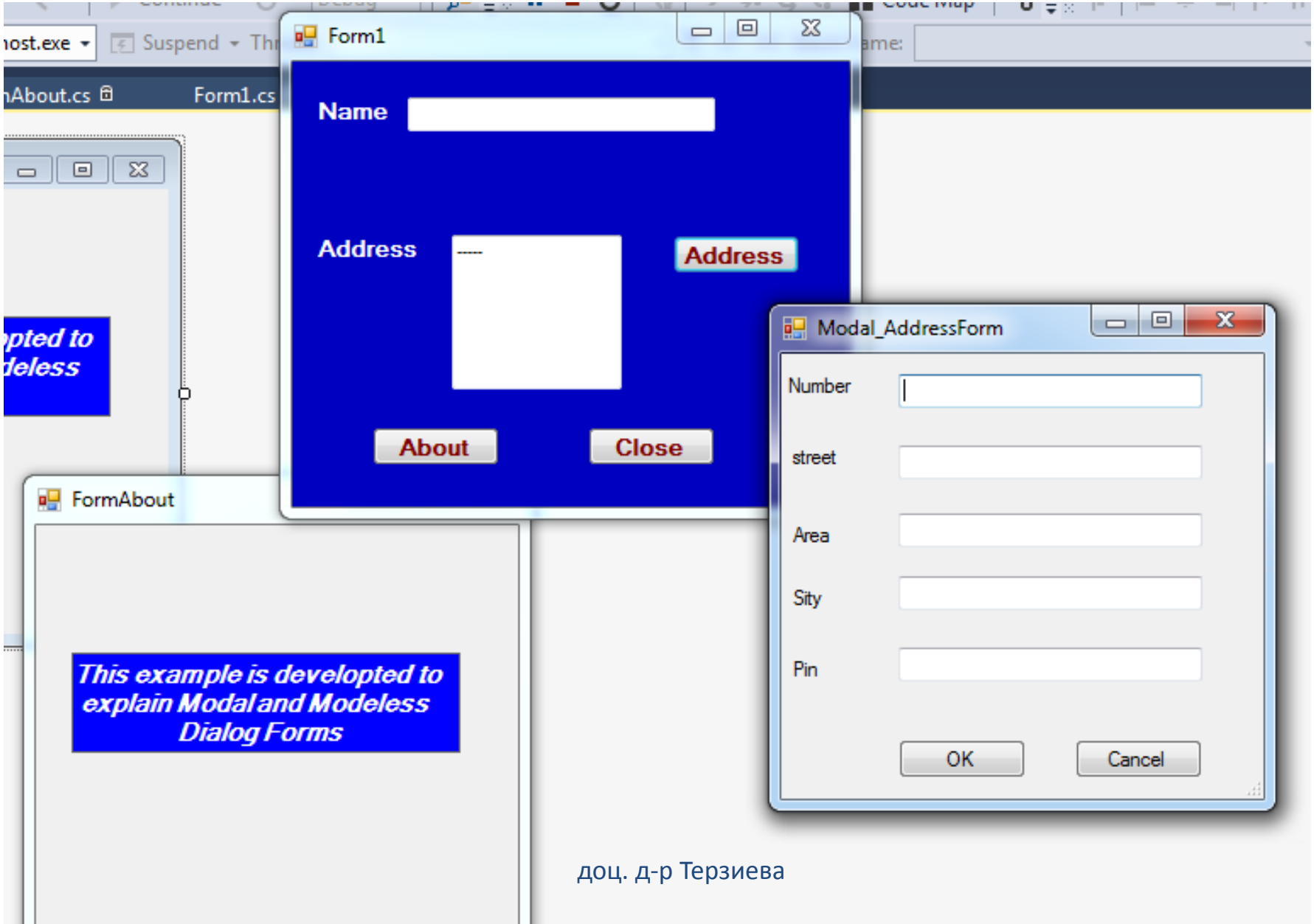
Създаване на немодална форма – пример

- От бутона **Name = buttonClose** на втората форма **subForm, създаваме събитието** Click за затваряне на формата

```
private void buttonClose_Click(object sender, EventArgs e)
        {
            this.Close();
            //или
            this.Hide();
        }
```



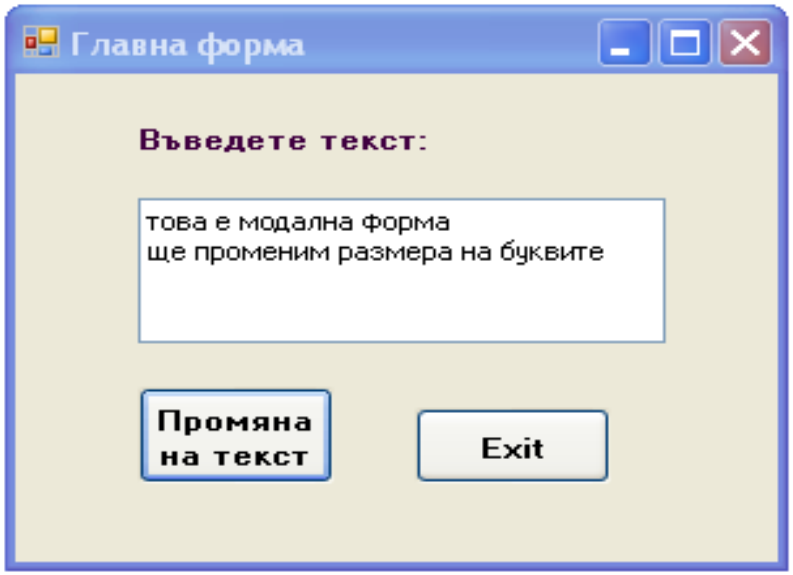
WinForm_Modal

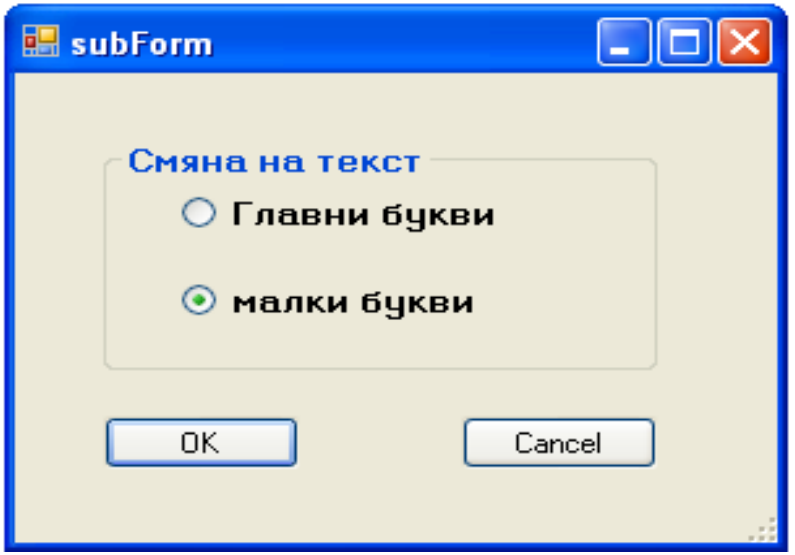


доц. д-р Терзиева

Модални форми – Пример 1/3

- Ще създадем приложение с две форми





Модални форми – Пример 2/3

- Към главната форма добавяме следният код:

```
subForm newForm = new subForm();
public static TextBox textb = new TextBox();
private void buttonOK_Click(object sender, EventArgs e)
   {
       // извикваме диалог за промяна на текст от втората форма
       newForm.ShowDialog();
   }
 private void buttonExit_Click(object sender, EventArgs e)
   {
        Application.Exit();
   }
 private void Main_Form_Load(object sender, EventArgs e)
   {
         textb = textBox1;
   }
```

Модални форми – Пример 3/3

- Към втората форма **subForm** добавяме следния код:

```
private void buttonOK_Click(object sender, EventArgs e)
        {
            string changeCase = Main_Form.textb.Text;
            if (radioButUpper.Checked == true)
            {
                changeCase = changeCase.ToUpper();
            }
            if (radioButtlower.Checked == true)
            {
                changeCase = changeCase.ToLower();
            }
            Main_Form.textb.Text = changeCase;
            this.DialogResult = DialogResult.OK;
        }
        private void buttonCancel_Click(object sender, EventArgs e)
        {
            this.DialogResult = DialogResult.Cancel;
        }
```

Благодаря за вниманието!