

## Протокол №2

# Тема:

**Обекти и визуално програмиране. Основни компоненти/контроли на графичния интерфейс. Свойства.**

Дата:

Изготвил:

В тетрадката да се запишат всички контроли и техните свойства, използвани в упражнение 1, 2 и задача 3.

#### Автоматично подреждане на контролите

1. Ограничаване на размера на формата  
Свойствата `MinimumSize` и `MaximumSize` се използват за да се ограничи размера на формата. Просто установете техните подсвойства `Width` и `Height`.
2. Свойството `ANCHOR` – когато потребителя променя размера на дадена форма, се променят размера на контролите така ,че да съответстват.
3. Свойството `Dock` – ви дава възможност да укажете на контрола да се прикрепи към някой от ръбовете на своя контейнер
4. Контроли контейнери `Windows Forms` за подреждане на контроли:
  - `Form`
  - `FlowLayoutPanel` – подрежда контролите отляво надясно и т.н.
  - `Panel`
  - `TableLayoutPanel` – подрежда контролите по редове и колони

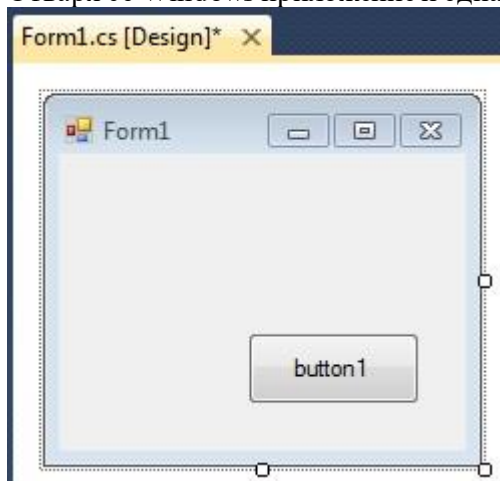
#### Въпроси и задачи:

##### 1. Упражнение BetterBookList - стъпка по стъпка

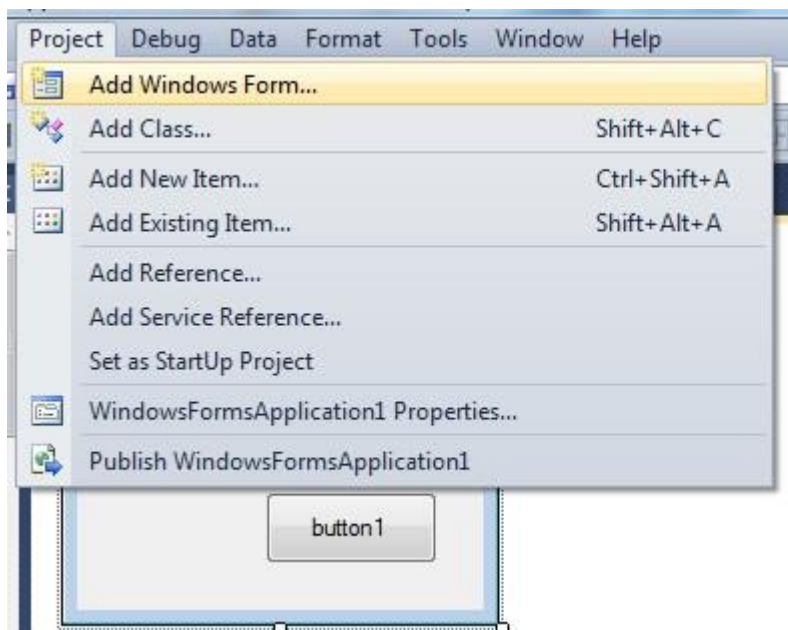
- Създайте нов проект, който се казва `BetterBookList`. Установете свойствата `Size` и `MinimumSize` на формата в 726,286
- Добавете една контрола `MenuStrip` към формата. За да направите менюто `File` , щракнете прозорчето `TypeHere` , който се показва и въведете `&File`. (Знакът амперсанд подчертава буквата F).
- Добавете една контрола `StatusStrip` към формата. Щракнете малката насочена надолу стрелка върху `StatusStrip` и изберете `StatusLabel`. Щракнете новия `StatusLabel` и използвайки прозореца `Properties`, установете неговия `Text` в `This is a StatusStrip`.
- Добавете един `Panel` към формата. Установете неговото свойство `Dock` в стойност `Fill`. Установете неговото свойство `BackColor` в светлозелено.

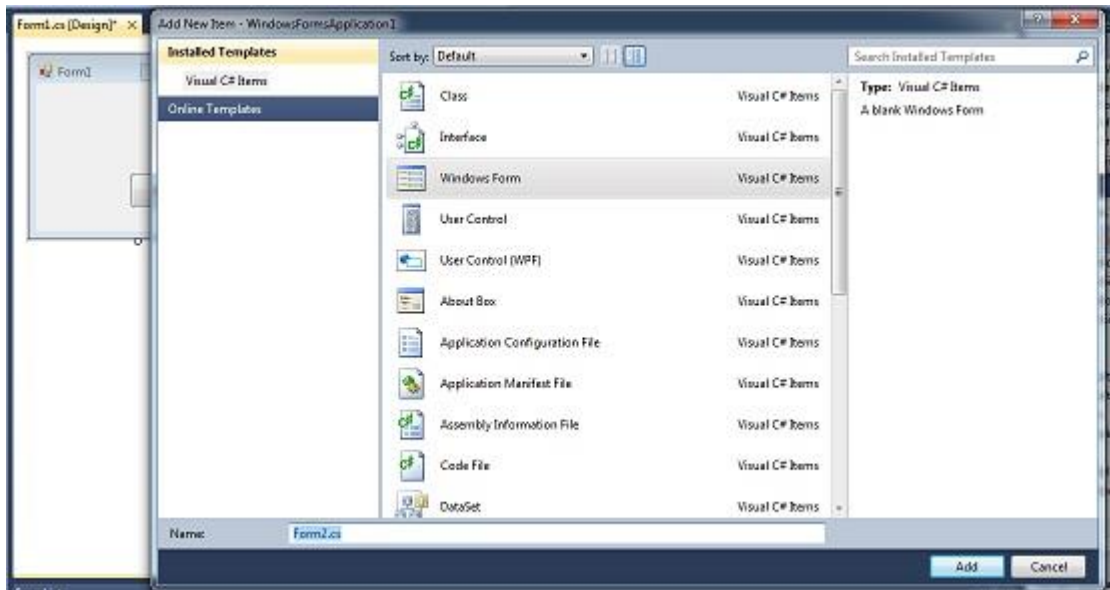
##### 2. Упражнение - Работа с няколко прозореца в Windows проекти на C#

- Отваря се Windows приложение и една форма с бутон в нея.



За следваща форма се използва менюто

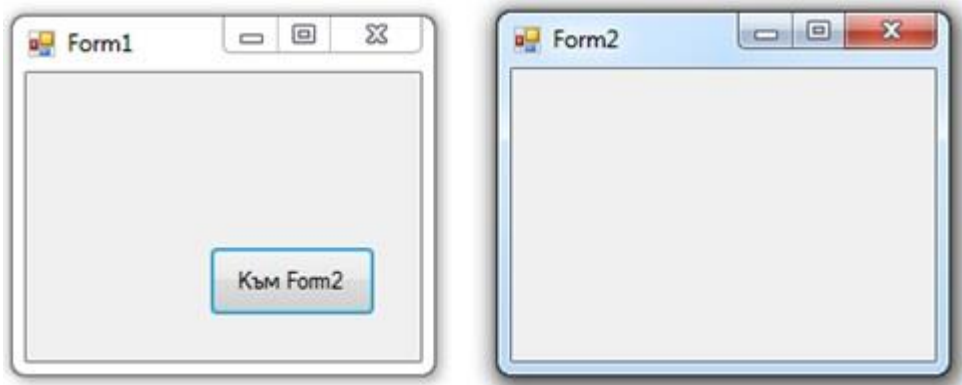
- A screenshot of the 'Add New Item...' menu in Visual Studio. The menu is open, showing options like 'Add Class...', 'Add New Item...', 'Add Existing Item...', 'Add Reference...', 'Add Service Reference...', 'Set as StartUp Project', 'WindowsFormsApplication1 Properties...', and 'Publish WindowsFormsApplication1'. The 'Add New Item...' option is highlighted, and its keyboard shortcut 'Ctrl+Shift+A' is visible. Below the menu, a portion of the form with the 'button1' is visible.



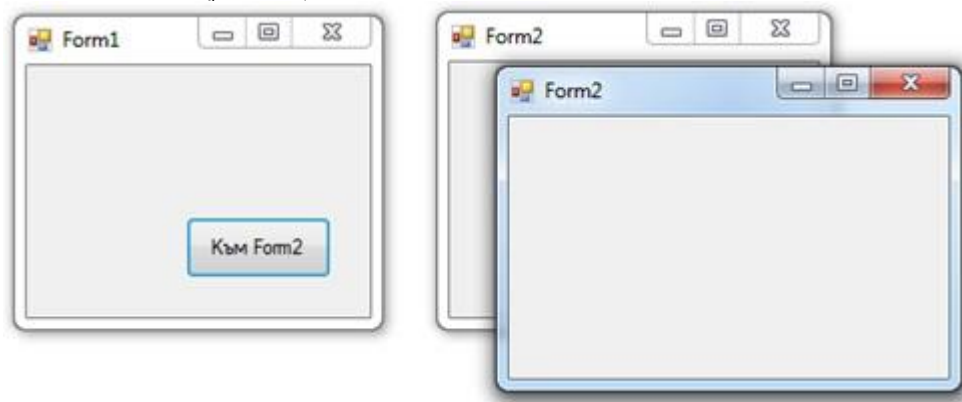
Добавя се код за бутона с надпис "Към Form2"

```
private void button1_Click(object sender, EventArgs e)
{
    Form2 secondForm = new Form2();
    secondForm.Show();
}
```

При стартиране на проекта, първото натискане на бутона води до следния резултат:



Ако бутонът не се скрие или поне да се забрани достъпа до него – резултатът при повторно използване на бутона ще е:



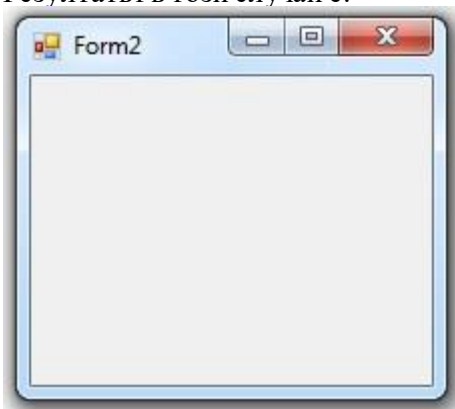
Ако Form1 (началната) се затвори, ще се затвори целият проект (всички прозорци, форми).



Друг вариант за кода на бутона е:

```
private void button1_Click(object sender, EventArgs e)
{
    Form2 secondForm = new Form2();
    secondForm.Show();
    this.Hide();
}
```

Резултатът в този случай е:



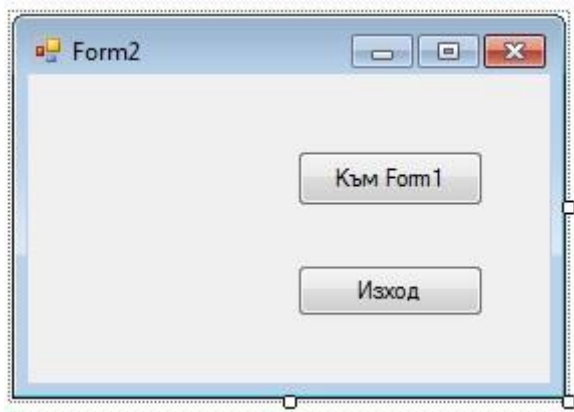
Ако само затворим Form2, проектът все още е активен - Form1 (началната форма) е само скрит. За да се избегне този проблем, може да се зададе на събитието FormClosing на Form2 следния код:

```
protected override void OnFormClosing(FormClosingEventArgs e)
{
    Application.Exit();
}
```

или

```
protected override void OnFormClosing(FormClosingEventArgs e)
{
    Environment.Exit(0);
}
```

Резултатът в този случай е:



За бутоните на Form2, може да се зададе следния код:

```
private void button1_Click(object sender, EventArgs e)
{
    Form1 secondForm = new Form1();
    secondForm.Show();
    this.Hide();
}
```

или

```
private void button1_Click(object sender, EventArgs e)
{
    Form1 secondForm = new Form1();
    secondForm.Show();
    this.Close();
}
```

```
private void button2_Click(object sender, EventArgs e)
{
    Application.Exit();
}
```

Файлът Program.cs задава от коя форма да започне проектът (началната форма).

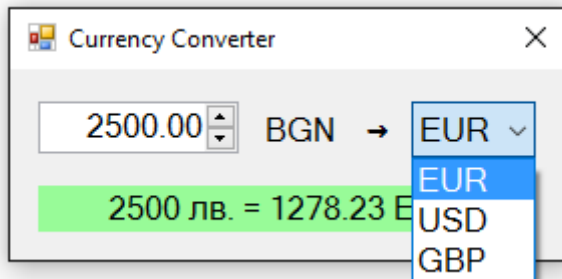
```
static void Main()
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new Form1());
}
```

Ако се започне от Form2, това е началната форма и this.Close(); ще затвори целия проект, а с this.Hide(); ще продължи.

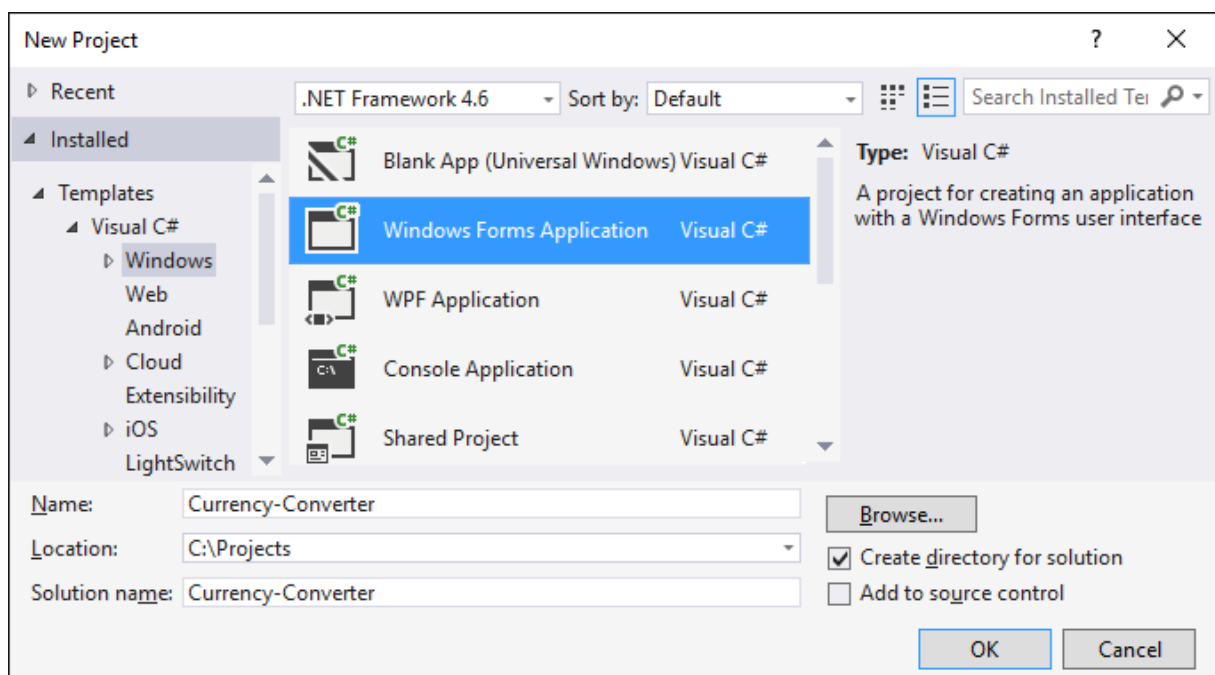
```
static void Main()
{
    Application.EnableVisualStyles();
    Application.SetCompatibleTextRenderingDefault(false);
    Application.Run(new Form2());
}
```

### 3. Задача - конвертор за валути

Графично (GUI) приложение за **конвертиране на валути**. Приложението ще изглежда приблизително като на картинката по-долу:



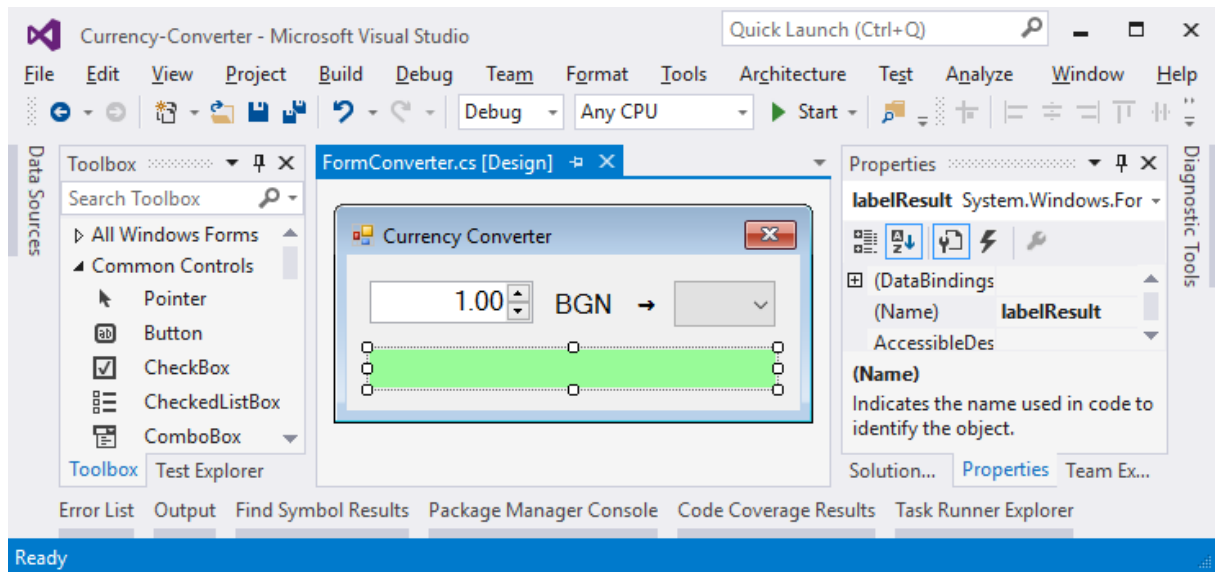
Създаваме нов **Windows Forms Application** с име “Currency-Converter”:



Нареждаме следните контроли във формата:

- Една кутийка за въвеждане на число (**NumericUpDown**)
- Един падащ списък с валути (**ComboBox**)
- Текстов блок за резултата (**Label**)
- Няколко надписа (**Label**)

Нагласяме **размерите** и свойствата им, за да изглеждат долу-горе като на картинката:



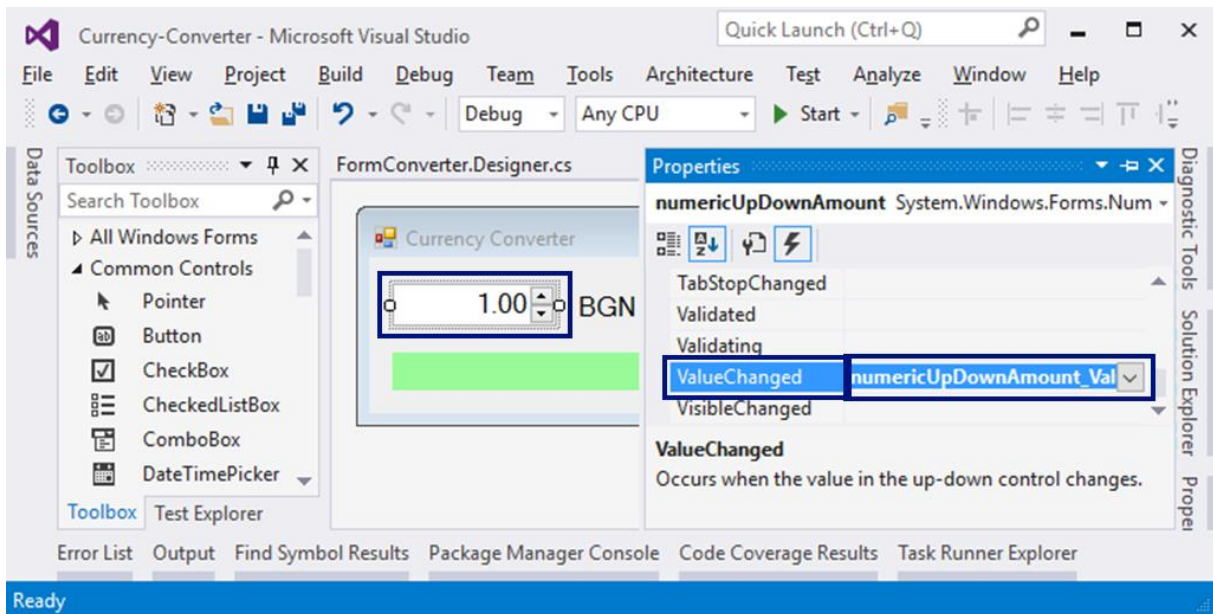
Задаваме следните **настройки на контролите**:

- За главната форма (Form), която съдържа всички контроли:
  - (name) = FormConverter
  - Text = "Currency Converter"
  - Font.Size = 12
  - MaximizeBox = False
  - MinimizeBox = False
  - FormBorderStyle = FixedSingle
- За полето за въвеждане на число (NumericUpDown):
  - (name) = numericUpDownAmount
  - Value = 1
  - Minimum = 0
  - Maximum = 1000000
  - TextAlign = Right
  - DecimalPlaces = 2
- За падащия списък с валутите (ComboBox):
  - (name) = comboBoxCurrency
  - DropDownStyle = DropDownList
  - Items =
    - EUR
    - USD
    - GBP
- За текстовия блок за резултата (Label):
  - (name) = labelResult
  - AutoSize = False
  - BackColor = PaleGreen
  - TextAlign = MiddleCenter
  - Font.Size = 14
  - Font.Bold = True

Трябва да хванем следните **събития**, за да напишем C# кода, който ще се изпълни при настъпването им:

- Събитието **ValueChanged** на контролата за въвеждане на число **numericUpDownAmount**:





- Събитието **Load** на формата **FormConverter**
- Събитието **SelectedIndexChanged** на падащия списък за избор на валута **comboBoxCurrency**

Ще използваме следния **С# код** за обработка на събитията:

```
private void FormConverter_Load(object sender, EventArgs e)
{
    this.comboBoxCurrency.SelectedItem = "EUR";
}

private void numericUpDownAmount_ValueChanged(object sender, EventArgs e)
{
    ConvertCurrency();
}

private void comboBoxCurrency_SelectedIndexChanged(object sender, EventArgs e)
{
    ConvertCurrency();
}
```

Задачата на горния код е да избере при стартиране на програмата валута “**EUR**” и при промяна на стойностите в полето за сума или при смяна на валутата, да изчисли резултата, извиквайки **ConvertCurrency()** метода.

Следва да напишем действието **ConvertCurrency()** за конвертиране на въведената сума от лева в избраната валута:

```
private void ConvertCurrency()
{
    var originalAmount = this.numericUpDownAmount.Value;
```

```
var convertedAmount = originalAmount;  
if (this.comboBoxCurrency.SelectedItem.ToString() == "EUR")  
{  
    convertedAmount = originalAmount / 1.95583m;  
}  
// ДОПИШЕТЕ КОДА  
  
this.labelResult.Text = originalAmount + " лв. = " +  
Math.Round(convertedAmount, 2) + " " + this.comboBoxCurrency.SelectedItem;  
}
```