Създаване на програмен код. Операции във Visual Basic. Аритметични и логически операции

I. Етапи при разработката на приложения

1 **Уточняване на потребителския интерфейс –** Обикновено този етап преминава в съвместна работа с човека, поставил задачата. Уточняват се вид и брой менюта, бутони – брой и разположение, размер и брой форми, цветово оформление и т.н.(*Имайте предвид, че никой клиент не знае точно каква програма иска, но всеки съвсем точно знае каква не иска*.)

2 **Създаване на програмен код за активизация на визуалния интерфейс, подготвен в първия етап -** Обектите в начертаната вече форма сами по себе си не биха правили нищо, докато не се напише програмен код, за начина, по който те би трябвало да реагират. (*Една програма върши това, което сте записали в нея, а не това, което Ви се иска*.)

3 **Търсене на грешки в програмния код –**

- ***a*** *Тъй като първият вариант на програмата никога не е верен, започвайте направо с втория.*
- ***b*** *Ако в програмата има грешки, значи сте работили правилно*

4 Отстраняване на грешките

II. Създаване на програмен код – писане на програма. Програмния код написан на Visual Basic коренно се различава от този на Turbo Pascal, който познавате. Разликите се състоят в начина на писане и изпълнение. В Turbo Pascal една програма се изпълнява от начало до край, като в нея се предвиждат всички възможни потребителски варианти на реакция. При VB се пише код за реакция на дадено конкретно събитие – кликване с мишка на бутон, натискане на клавиш от клавиатурата и т. н. Далеч не за всички събития се съставя код (те могат да бъдат много и хаотични – блъскане по клавиатура, щракване с друг бутон на мишка, кликване извън дадено поле), а само за определено съзнателно очаквано действие. В момента, когато настъпи такова събитие, VisualBasic търси код, който да указва какво да направи при настъпването му. Кодът, който реагира на конкретно събитие, се нарича „манипулатор за обработка на събитие". Точно това е една от основните особености на събитийно-управляемото програмиране: част от програмния код се изпълнява само при настъпване на конкретна ситуация. Дали компютърът ще реагира на определено действие от страна на потребителя зависи от нас. Затова трябва много добре да преценим какви манипулатори ще пишем и на какви събития ще реагират те.

Всеки обект може да има един или няколко манипулатора за обработка на събития. Всеки манипулатор отговаря за обработката на едно конкретно събитие. Този вид програмиране се нарича СЪБИТИЙНО ПРОГРАМИРАНЕ

1. Видове събития
    a. събития свързани с клавиатурата
    b. събития свързани с мишката
    c. програмни събития
2. Създаване на манипулатор за обработка на събития

a. Да се определи коя част на потребителския интерфейс трябва да реагира на събитието
b. Да се отвори прозореца Code
c. Да се определи събитието, което трябва да реагира ВБ
d. Да се напише код за обработка на събитието

Преди да се пишат манипулатори трябва да се убедим, дали всички обекти имат имена. Не може да се променят имената след написване на манипулатора, защото в противен случай манипулаторите трябва да се напишат отново.

Частите на интерфейса, за които могат да се напишат манипулатори са следните: ФОРМИ, ОБЕКТИ (командни бутони, контролни полета...) и ПАДАЩИ МЕНЮТА

Манипулаторите на събития предствляват програмен код и той се пише в съответния прозорец, който се извиква по следния начин: F7(Code) или двукратно щракване с левия бутон на мишката. Когато се пише манипулатор за падащо меню се щраква върху командата, за която ще се пише манипулатора.

3. Части на манипулатор за обработка на събитие – ВБ извежда на екрана празен манипулатор за обработка на събитие. Той се състои от два реда:

```
Private Sub mnuEditUndu_Click()

End Sub
```

Private Sub – идентифицира процедурата като подпрограма

Име на обекта – подменю Undo на менюто Edit

Име на събитие – натисване на бутона на мишката (Click)

Двойка скоби – там евентуално се съдържат данни, необходими на програмата да работи, в този случай няма

ВСЕКИ ПЪТ, КОГАТО СЕ ПРОМЕНЯ ИМЕТО НА ОБЕКТ, ТО ТРЯБВА ДА СЕ ПРОМЕНЯ И ВЪВ ВСИЧКИ МАНИПУЛАТОРИ, КЪДЕТО Е ВПИСАНО

4. Разделяне на прозореца Code на две части Window\Split или с натиснат ляв бутон върху лентата Split
5. Редактиране в прозореца Code – работи като проста текстообработваща програма. Изтриването става по познатия начин.
6. Преглеждане на различни манипулатори за обработка на събития –

   - избира се име на обект от списъчното поле Object – Procedure
   - използване прозореца Object Browser

7. Избиране на манипулатор за обработка на събитие чрез прозореца Object Browser – най-полезен когато се разглеждат събития записани в различни файлове. Извежда се на екрана View\Object Browser. Избира се проект от библиотеката. Избира се файл на форма от показаните в левия панел (Classes)

   Избира се манипулатор за обработка и се щраква двукратно върху него. Записаните събития са в получер шрифт.

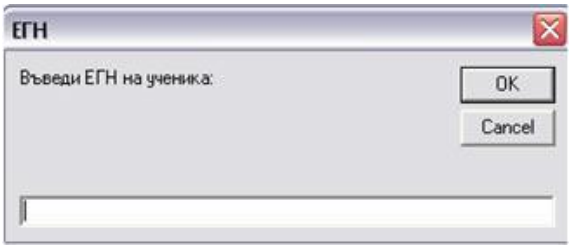
В една и съща форма не може да има два манипулатора за събитие с едно и също име. Това, обаче е възможно в различни форми.

8. Манипулатор за обработка на събитие, който е необходим на всяка програма – манипулатор за спиране на програмата. Използват се два варианта End и Unload Me. За препоръчване е да се използва Unload Me. За тази цел, обече е задължително всички форми да се затварят по този начин. Дори само една да не се затваря така, се налага използването на командата End.

III. Операции във Visual Basic

1. **Операции за вход и изход –** с тях ще с езапознаем по-подробно, когато изучаваме контролите. На този етап ще въведем две функции – InputBox и MassageBox

InputBox - За въвеждане на кратка информация, за която не е оправдано създаването на текстово поле



Прозорецът InputBox се състои от четири елемента:

- заглавен ред;
- подканящо съобщение за ввеждане на информация (Prompt);
- поле за въвеждане, евентуално съдържащо стойност по подразбиране;
- два бутона (ОК и Cancel).

Синтаксис:

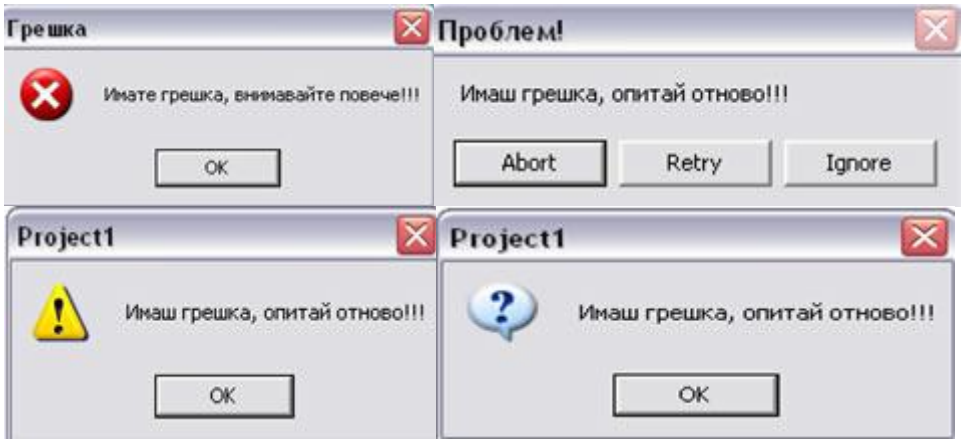
Идентификатор = InputBox(prompt, [title] [,default] [,Xpos] [,Ypos] [,helpfile] [, context])

Например:

```
Dim S as String
    S = InputBox (“Въведи ЕГН на ученика:”, „ЕГН”)
```

Параметърът Prompt задава текста, извеждан в диалоговия прозорец като подканящо съобщение (в примера «Въведи ЕГН на ученика:»).
Параметърът Title задава надписа в заглавната ивица на прозореца (в примера «ЕГН»); ако този параметър не е зададен, се извежда заглавието на приложението.
Параметърът Default задава стойност по подразбиране, която се извежда в реда за вход.
Параметрите Xpos и Ypos определят координатите на горния ляв ъгъл на прозореца. По премълчаване прозорецът се изобразява в центъра на екрана. Тези два параметъра трябва да се използват съвместно:
Функцията InputBox връща като резултат стринга, въведен от потребителя. При натискане на бутон Cancel се връща празен стринг.
Функцията InputBox притежават още два незадължителни параметъра — HelpFile и Context, които позволяват да се отворят определени файлове от справочната система.

MessageBox - извеждане на различни съобщения Почти всички приложения на Windows използват MessageBox, тъй като този компонент влиза в състава на Windows, а Visual Basic само предоставя възможност за неговото извикване.



Видът на прозореца MessageBox може да бъде различен, но в неговия състав винаги влизат:

- текстът на съобщението;
- заглавие;
- иконка;
- набор бутони.

Синтаксис:

Идентификатор = MsgBox (Prompt [, Buttons] [, Title] [, Helpfile] [,Context])

Например:

s = MsgBox("Имаш грешка, опитай отново!!!", vbQuestion)

Параметрите Prompt и Title нямат нужда от обяснение. С тях вече се запознахме в предния урок.
Параметърът Buttons определя външния вид на MessageBox. Стойността на този параметър се формира от няколко части, които може да се сумират:
Buttons = Button + Icon + Default + Modal + Extras + Extras

За категориите параметри Button, Icon, Default и Modal може да се използва само една от допустимите константи. А за категория Extras се допуска използването на комбинация от стойности.
Ето и най-често използваните стойности на тези константи по категории:

Категория Button:

Константа	Стойност	Описание
vbOKOnly	0	Извежда само бутон ОК
vbOKCancel	1	Извежда бутони ОК и Cancel
vbAbortRetryIgnore	2	Извежда бутони Abort, Retry, Cancel
vbYesNoCancel	3	Извежда бутони Yes, No, Cancel
vbYesNo	4	Извежда бутони Yes, No
vbRetryCancel	5	Извежда бутони Retry и Cancel

Категория Icon:

Константа	Стойност	Описание
vbCritical	16	Показва икона Critical Message
vbQuestion	32	Показва икона Warning Query
vbExclamation	48	Показва икона Warning Message
vbInformation	64	Показва икона Information Message

Категория Default:

Константа	Стойност	Описание
vbDefaultButton1	0	По подразбиране е активен първия бутон
vbDefaultButton2	256	По подразбиране е активен втория бутон
vbDefaultButton3	512	По подразбиране е активен третия бутон
vbDefaultButton4	768	По подразбиране е активен четвъртия бутон

Категория Extras:

Константа	Стойност	Описание
vbMsgBoxHelpButton	16384	Допълнителен бутон за справка
vbMsgBoxSetForeground	65536	Извежда диалоговия прозорец във фонов режим
vbMsgBoxRight	524288	Текстът се подравнява по десния край
vbMsgBoxRtlReading	1048576	Текстът се извежда отдясно наляво (арабски)

2. Аритметични оператори

VisualBasic предоставя следните типове оператори: аритметични, логически и оператори за сравнение. По начин на действие те са почти същите, както и в останалите езици за програмиране.

Оператор	Действие	Пример
+	Събира две числа	Sum=X+23
-	Изважда две числа	X=Sum-100
*	Умножава две числа	Y=25*3.55
/	Обикновено реално делене на две числа	104/162=0.6419753
\	Целочислено деление	
Mod	Остатък от целочислено делене	15 mod 4 = 3
^	Повдига число на степен със зададен степенен показател	2^5=32
&	Събира (конкатенира) два низа	“ab”&”12”=”ab12”

При използването на оператора \ има една особеност – числата, които ще се делят се закръглят предварително. Резултатът е ЦЯЛАТА част от полученото число.

Пример: 2.5\1.5 ⇨ 1 защото 2.5 ⇨3, 1.5 ⇨2, 3/2 ⇨2.5 ⇨2

3. **Логически оператори**

Това са оператори, реализиращи основните логически операции. Най-често използваните логически оператори са:

Оператор	Действие
Not	Логическо отрицание
And	Логическо „И”
Or	Логическо „ИЛИ”

3. Оператори за сравнение

Във VisualBasic се използват познатите шест оператора за сравнение: <, >, <=, >=, =, <>

При използването на тези оператори няма никакви разлики в сравнение с другите езици.

4. Приоритет на аритметичните и логически операции

Приоритетът на операциите задава редът, в който се изчисляват тези операции, в случаите, когато не е зададен явно (например чрез скоби). В следващата таблица е даден приоритета на операциите, подредени от най-голям към най-малък.

Оператор	Тип на оператора
Повдигане на степен (^)	Аритметичен
Смяна на знака (-)	Аритметичен
Умножение и деление (* и /)	Аритметичен
Целочислено делене (\)	Аритметичен
Делене по модул (mod)	Аритметичен
Събиране и изваждане (+, -)	Аритметичен
Конкатенация (&)	Аритметичен

Равенство (=)	Сравнение
Неравенство (<>)	Сравнение
По-малко от (<)	Сравнение
По-голямо от (>)	Сравнение
По-малко или равно (<=)	Сравнение
По-голямо или равно (>=)	Сравнение
Like	Сравнение
Is	Сравнение
Логическо отрицание (Not)	Логически
Логическо И (And)	Логически
Логическо ИЛИ (Or)	Логически

И тук, когато в един израз имаме няколко операции с еднакъв приоритет, а липсват скоби, операциите се изпълняват последователно отляво надясно.