# 31. ПРОЕКТ «КАЛЬКУЛЯТОР»

Наш первый проект – «Калькулятор».[1] Мы создадим простейший калькулятор, который сможет выполнять четыре «самых знаменитых» арифметических действия: сложение, вычитание, умножение и деление. Мы не ставим перед собой задачи потягаться с Microsoft и создать нечто подобное калькулятору, встроенному в Windows[2]; наша цель скромнее: понять методологию (если хотите, «фишку» :)) программирования на Delphi и подобных ему визуальных объектно-ориентированных языках программирования. При создании нашей программы мы применим «визуальный подход», так сказать, «в чистом виде»: сначала нарисуем форму (по сути изображение калькулятора), а затем «вдохнем жизнь» в его кнопки и табло.

Наш калькулятор будет выглядеть так:

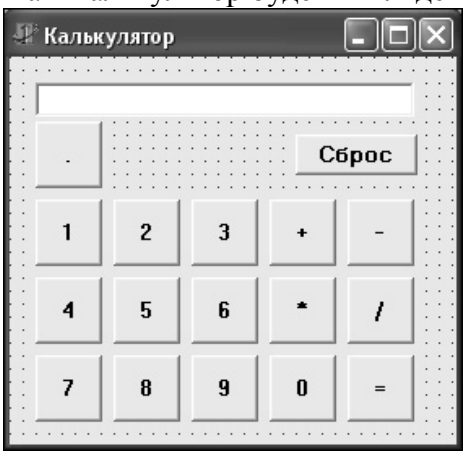


**Рис.25. Калькулятор.**

Он представляет собой форму FormCalc – объект класса TFormCalc, порожденного от TForm. На форме располагаются следующие объекты-компоненты:

- 10 кнопок с цифрами от 0 до 9 Button0…Button9.
- 4 кнопки со знаками действий: ButtonPlus (“+”), ButtonMinus (“-“), ButtonMult (“*”) и ButtonDiv (“/”).
- Кнопка ButtonRez (“=”), «подсчитывающая результат», кнопка ButtonPoint (“.”) для ввода десятичной точки и кнопка ButtonReset (“Сброс”) для сброса информации на табло. Все перечисленные кнопки – объекты класса TButton.
- Табло EditTablo – объект класса TEdit.

Здесь все названия объектов – наши собственные, а не назначенные Delphi по умолчанию: для использования собственных названий проще всего поменять значения свойства Name для соответствующих объектов в Инспекторе Объектов.

Все перечисленные объекты автоматически объявляются в модуле формы при их на ней размещении (а изначально берутся они, как мы помним, из палитры компонентов). Delphi автоматически делает форму и собственником (Owner), и родителем (Parent) всех размещенных на ней компонентов. Впоследствии мы увидим, как можно противодействовать такому «дельфийскому произволу». Для надписей на форме и кнопках нужно поменять значения свойства Caption соответствующих объектов, а для «большей выразительности» еще и внести коррективы в значения свойства Font у кнопок.

Теперь что касается «вдыхания жизни» в наш калькулятор. Вначале поговорим о «глобальных» «действующих лицах» нашего «калькуляторного» модуля.[3] Само собой, это

---

[1] Все Delphi-проекты на прилагаемом к книге диске расположены в папке «Delphi-projects». Разумно скопировать эту папку в то место на жестком диске, куда вы установите Delphi (например, путь к папке с проектами может быть таким: «C:\Program Files\Borland\Delphi5\Delphi-Projects»).

[2] Хотя в принципе могли бы, а? :)

[3] В нашу первую программу на Delphi будет входить всего один модуль (модуль формы, «скелет» которого сформирует сам Delphi).

прежде всего переменная, представляющая калькулятор как таковой – переменная FormCalc типа TFormCalc. Остальные «глобальные действующие лица» непосредственно связаны с калькуляторными вычислениями и их отображением на табло. Итак:

Operand1 и operand2 – это операнды, а operation – знак операции. Например, для выражения 34.5+20 операндами будут числа 34.5 и 20, а знаком операции «+».

В item «накапливается» вводимый в данный момент элемент. Для нашего примера: если мы занимаемся вводом первого операнда, то item вначале равен «3», затем «34», далее «34.» и «34.5». Таким образом в item в качестве текущего элемента «накоплен» первый операнд. Присутствие знака «+» за цифрой «5» «говорит» о том, что на смену предыдущему item – первому операнду – идет следующий item – знак операции. Далее в item будет «накапливаться» второй операнд, пока процесс не «наткнется» на новый знак операции или знак «=» и т.д. (подробнее узнаете из текста программы).

ItemNumber есть номер item. Первый операнд имеет номер 1, знак операции – номер 2, второй операнд – номер 3. Результат проделанной операции автоматически становится новым первым операндом.

Is_point – логическая переменная, фиксирующая, имеется ли десятичная точка в текущем операнде.

Symbol – очередной символ, «нажимаемый» пользователем калькулятора.

Stroka есть строка для отображения действий пользователя, показываемая на табло.

Теперь разберемся с «обычными» подпрограммами нашего модуля.[4] Процедура show служит для показа текущей информации на табло. Процедура sbros приводит «действующие лица» в «начальное состояние». Процедура input(symbol:char) анализирует ввод очередного символа, и, если символ «подходящий», «включает его в работу». Если при вводе очередного символа нужно посчитать и отобразить результат только что выполненной операции, процедура input делает это. Заслуживает внимания «стандартная» процедура Val(…), которая «занимается» преобразованием строки в число, например, ’25.5’ преобразуется в 25.5. Существует и обратная ей процедура Str(…), преобразующая число в строку. Подобные подпрограммы преобразования весьма часто используются на практике. Попробуйте разобраться с работой процедур Val и Str самостоятельно (удобнее всего воспользоваться для этого системой помощи)[5]. Результат выполненной операции всегда отображается на табло в виде «<число>+», например, «54.5+» (даже в начале работы мы увидим на табло «0+»). То есть калькулятор автоматически «всегда готов» к сложению. Это позволяет, во-первых, не нажимать лишний раз на «+», если мы хотим сложить операнды, а во-вторых, упрощает программирование «начальной ситуации», когда еще никаких вычислений не было: «начальная ситуация» сводится к «обычной», когда результатом предыдущей операции как будто бы был 0. Функция minus_space_zero(StrNum:string):string удаляет из результата операции лишние пробелы, нули и точки, например, из «  26.000» получается «26».

Теперь о методах-обработчиках событий. Создание «кликовых» методов (методов, связанных со щелчком левой кнопки мыши на соответствующем объекте) начинается либо с двойного щелчка мышью напротив события OnClick для соответствующего объекта (закладка Events Инспектора Объектов), либо (что еще проще) с двойного щелчка мышью на самом объекте. Таким образом получаются заготовки для процедур TFormCalc.ButtonResetClick, TFormCalc.ButtonPointClick, TFormCalc.Button1Click и т.п. Названия, как всегда, Delphi «взял на себя», но сейчас у нас нет резона их менять: они вполне понятны и «красноречивы». Название – это не более чем «обертка»; «сила» методов-обработчиков событий в том, что, во-первых, им, как и любым методам класса, обязательно неявно передается ссылка на вызвавший их объект (переменная Self), а во-вторых, их параметры содержат важнейшие сопутствующие вызову метода вещи:

---

[4] «Обычными» в том смысле, что они не являются методами-обработчиками событий. В нашей программе они вообще не являются методами класса (хотя и могли бы).

[5] И вообще: как только вам что-то незнакомо или непонятно, первым делом загляните в систему помощи.

непосредственный источник события, код нажатой клавиши, позицию мыши и т.п. В нашем случае все «кликовые» методы имеют параметр Sender, указывающий на объект, на котором был сделан щелчок мышью. Для метода TFormCalc.Button0Click, например, Self – это форма FormCalc, а Sender – это кнопка Button0.

Кроме «кликовых» методов, мы создадим обработчики событий OnCreate (при создании), OnKeyPress (при нажатии клавиши), OnActivate (при активации) для самой формы FormCalc. Заготовки для обработчиков задаются с помощью двойных щелчков мышью напротив соответствующих событий в окне Инспектора Объектов на закладке Events. Событие OnCreate возникает после создания формы; обработчик этого события TFormCalc.FormCreate запускает процедуру Sbros. OnActivate происходит, когда форма (ее окно) активна (у нас она активизируется сразу после создания и так и остается активной вплоть до выхода из программы ввиду того, что она единственная). EditTablo.SetFocus устанавливает фокус на табло калькулятора. Обработчик события OnKeyPress TFormCalc.FormKeyPress имеет два формальных параметра: Sender и Key. Key – это нажатая пользователем клавиша. В обработчике она анализируется, и, если все нормально, вызывается процедура input с фактическим параметром – соответствующим нажатой клавише символом. TFormCalc.FormKeyPress дает возможность пользоваться для вычислений не только клавишами нашего калькулятора, но и клавиатурой компьютера. Чтобы этот обработчик работал корректно, на стадии конструирования формы в ее свойство KeyPreview (предварительный просмотр клавиши) было помещено значение True. Это значит, что именно форма будет первой «отлавливать» нажатия пользователем клавиш (а не те ее объекты, которые в момент нажатия имели фокус).

Все «кликовые» процедуры очень просты и не требуют каких-либо дополнительных пояснений.

«Дельфийская» процедура ShowMessage (она встречается в процедуре input) показывает на экране модальное (блокирующее при своем появлении все другие окна приложения) окно с соответствующим сообщением.

Вот, пожалуй, и все, что необходимо пояснить касательно проекта «Калькулятор». Остальное написано в самой программе. Разберитесь с ней – и у вас наверняка появится желание что-то модернизировать, усовершенствовать – и может быть, улучшить Windows-калькулятор от Microsoft.[6]

[6] А почему бы и нет?