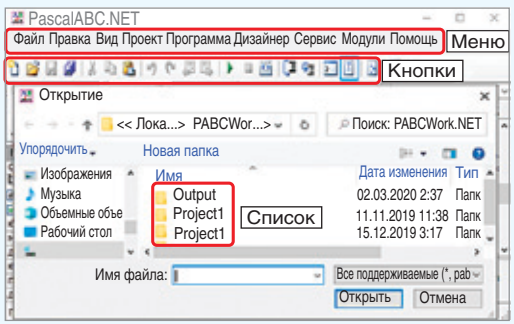
Глава 1
ВВЕДЕНИЕ В ОБЪЕКТНО-СОБЫТИЙНОЕ ПРОГРАММИРОВАНИЕ

§ 1. Объектно-событийная модель работы программы

Основателем RAD считается сотрудник IBM, британский консультант по информационным технологиям Джеймс Мартин (1933—2013), который в начале 1990-х гг. сформулировал основные принципы RAD, основываясь на идеях Барри Бойма и Скотта Шульца.

Пример 1.1. После загрузки какого-либо редактора пользователь может открыть файл для редактирования. При этом он выбирает меню **Файл**, находит в списке команду **Открыть**, выбирает нужный файл, нажимает кнопку **Открыть**. Как мы видим, чтобы открыть файл, пользователь взаимодействует с такими элементами управления, как меню, список, кнопка.



1.1. Элементы управления в приложениях с графическим интерфейсом

Современные программы, с которыми сегодня работают пользователи компьютера, отличаются от тех, которые вы создавали раньше. Основное отличие — взаимодействие пользователя с программой.

Программы, которые вы создавали в 7—10-м классах, взаимодействовали с пользователем посредством текстового интерфейса (часто его называют интерфейсом командной строки). После запуска программы вы вводили данные, программа выполнялась, и вы видели результат. И ввод, и вывод данных осуществлялся в алфавитно-цифровой форме.

Операционные системы с графическим оконным интерфейсом (например, Windows) предполагают общение пользователя с программой посредством элементов управления. К элементам управления относят: кнопки, разнообразные меню, текстовые сообщения, списки и др. При работе программы пользователь выбирает какой-либо элемент управления и совершает с ним определенное действие (пример 1.1). Если такое действие для выбранного элемента было определено, то программа его выполняет, иначе выдает сообщение об ошибке.

Многие системы программирования позволяют создавать программы с оконным интерфейсом. Такие программы называют **оконными приложениями** (Windows Form Application).

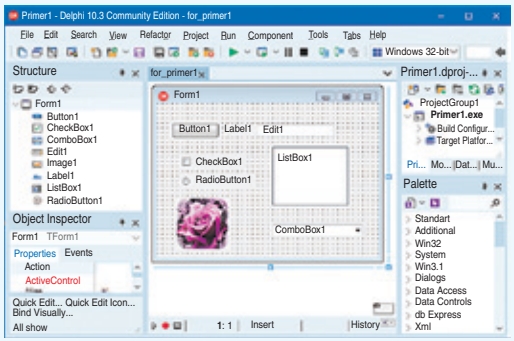
Проектирование интерфейса окна программы можно выполнять с использованием RAD-технологии (Rapid Application Development — **быстрая разработка приложений**). Технология RAD характерна для многих систем программирования. Быстрая разработка стала возможной за счет того, что элементы управления были визуализированы и собраны в специальные библиотеки — VCL (Visual Component Library — визуальная библиотека компонентов).

Различные элементы управления можно перетаскивать с палитры компонентов на форму с помощью мыши. Процесс создания интерфейса будущей программы представляется аналогом работы с неким конструктором. Программирование в RAD-средах является визуальным, поскольку код по созданию объекта не пишется, а генерируется средой. Задача программиста — написание кода по управлению готовыми компонентами.

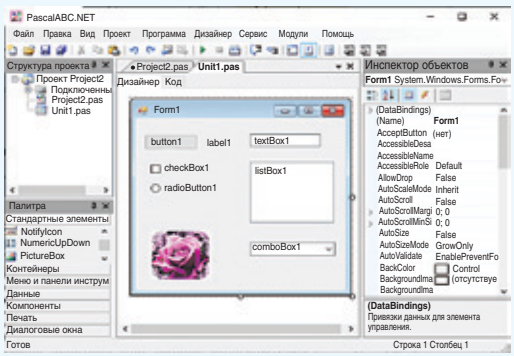
Визуальное программирование поддерживается в PascalABC и Delphi (код пишется на языке Pascal), VisualBasic, C# и др. (пример 1.2). Для обучения учащихся младших классов используется визуальное программирование в среде Скретч (Scratch).

Пример 1.2. Среды программирования, в которых реализована поддержка парадигмы визуального программирования.

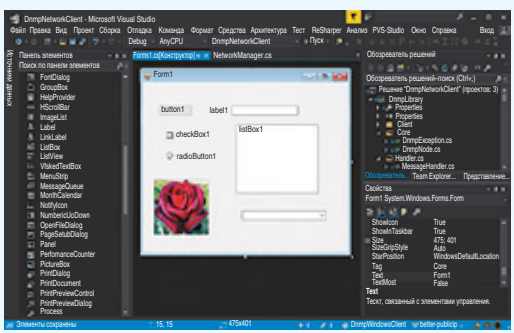
Delphi:



PascalABC:



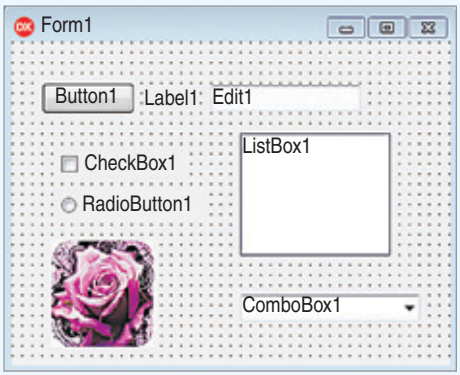
Visual Studio для языка C#:



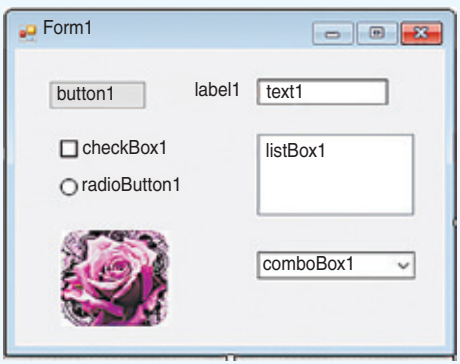
Пример 1.3. Основные элементы интерфейса:

Элемент управления	Имя
Кнопка	Button
Надпись	Label
Поле для ввода текста	TextBox (Edit)
Флажок	CheckBox
Радиокнопка	RadioButton
Список	ListBox
Выпадающий список	ComboBox
Рисунок	PictureBox (Image)

Элементы управления на форме в среде программирования Delphi:



Элементы управления на форме в среде программирования PascalABC:



Многие элементы управления в разных средах имеют одинаковые или синонимичные имена (пример 1.3).

Создаются оконные приложения как проект и состоят из нескольких файлов. Внешний вид окна будущего приложения строится на форме. Для формы сохраняются два файла — один содержит описание внешнего вида формы, другой — описание действий при выборе пользователем того или иного элемента управления. Главный файл проекта содержит описание его структуры, а также команды по созданию формы и запуску приложения.

Все элементы, размещенные на форме, и сама форма образуют систему взаимодействующих объектов. Способ их взаимодействия основан на объектно-ориентированном программировании.

Объектно-ориентированное программирование (ООП) — технология создания программ, основанная на использовании системы объектов. Каждый объект обладает набором **свойств**, которые описывают его состояние, и **методов**, характеризующих его поведение.

Объект — совокупность данных и методов работы с ними.

Организация данных внутри объекта скрыта от пользователя. Данные и способы их чтения и записи являются свойствами объекта, их можно изменять. Методы — процедуры и функции для обработки данных.

1.2. События

Организация взаимодействия между программой и пользователем управляется **событиями:** пользователь может нажать на клавишу мыши или клавиатуры, ввести текст и др.

Метод программирования, основанный на управлении событиями, называют **событийно-ориентированным программированием.**

Каждое событие связано с каким-либо объектом, которому передается управление в тот момент времени, когда происходит событие. Среди основных событий можно выделить три категории: события мыши, события клавиатуры и системные события (примеры 1.4—1.6).

Процедура (или функция), инициируемая событием, называется **обработчиком события.**

Запущенный на выполнение проект находится в ждущем режиме, реагируя на события, учтенные при его создании, вызываемые действиями пользователя или возникающими в самой программе.

Объектно-событийная модель программы предполагает следующее:

- создание объектов с присущими им свойствами и методами;
- описание событий, при которых объект может выполнять алгоритм обработки данных.

Пример 1.4. События мыши возникают в том случае, если пользователь производит какие-либо действия с мышью:

Click	Нажатие левой кнопки мыши
DblClick	Двойной щелчок левой кнопкой мыши
MouseDown	Нажатие на любую кнопку мыши. Параметры обработчика события позволяют определить, какая из кнопок была нажата и в какой точке
MouseUp	Освобождение кнопки мыши, которая была нажата
MouseMove	Перемещение указателя мыши

Пример 1.5. События клавиатуры происходят при нажатии клавиш на клавиатуре:

KeyPress	Нажатие клавиши с текстовым символом
KeyDown	Нажатие любой клавиши
KeyUp	Освобождение клавиши

Пример 1.6. Системные события управляются функциями операционной системы:

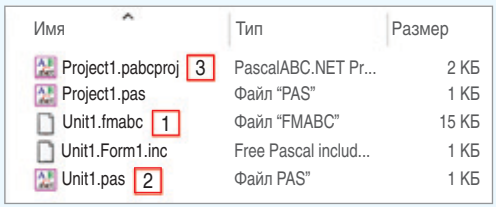
Paint	Возникает, когда элемент необходимо перерисовать
Resize	Происходит, когда размеры элемента изменяются
Enter	Возникает, когда элемент управления становится активным
Leave	Происходит, когда элемент управления перестает быть активным

? **1.** Какие программы называют оконными приложениями?
2. Что понимают под событийным программированием?
3. Какие типы событий вы можете назвать?

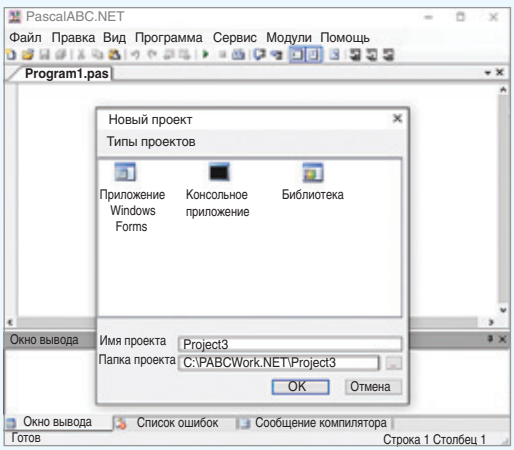
§ 2. Визуальная среда разработки программ

Работа по созданию оконных приложений рассматривается в среде программирования PascalABC.Net.

Пример 2.1. Файлы проекта:



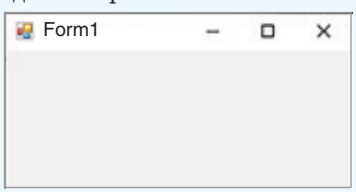
Пример 2.2. Создание проекта в PascalABC.Net:



Пример 2.3. Создание и выполнение проекта:

1. Создать папку с именем Primer1.
2. Создать проект.
3. Запустить проект на выполнение.

Вид окна приложения:



Для сохранения изменений используют команду Сохранить все (кнопка).

2.1. Структура проекта

При создании оконного приложения работают с проектом, состоящим из нескольких файлов. В разных средах программирования проект может состоять из различного количества файлов. Обязательными файлами являются следующие:

- файл формы (1), содержащий описание внешнего вида окна приложения;
- файл программного модуля (2), содержащий описание функций-обработчиков для объектов на форме;
- файл проекта (3), позволяющий связать структурные элементы проекта между собой.

(Рассмотрите пример 2.1.)

Файлы одного проекта обычно хранятся внутри отдельной папки. При компиляции приложения создается файл с расширением *.exe* и именем, совпадающим с именем проекта. Этот файл запускает работающее приложение без загрузки среды программирования. (Как скомпилировать приложение, чтобы файл с расширением *.exe* не удалялся после закрытия окна, см. в *Приложении*, с. 102).

Для создания проекта в среде PascalABC.Net нужно выполнить команды **Файл → Новый проект → Приложение Windows Form** (пример 2.2).

При создании проекта файлы сохраняются автоматически (пример 2.3).

2.2. Интерфейс среды программирования

Полное окно среды программирования PascalABC.Net при создании приложений Windows Form можно посмотреть в *Приложении* (с. 102).

Рассмотрим основные элементы.

Основное меню и **панель быстрого доступа** (пример 2.4) содержат команды для управления проектом: сохранение, загрузка, выполнение и др.

Форма (пример 2.5) служит для визуального отображения окна приложения. Во время проектирования приложения на форме отображается точечная сетка, позволяющая выравнивать помещаемые на форму компоненты.

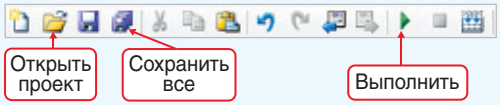
Инспектор объектов (пример 2.6) отображает свойства (или события) выбранного объекта.

В левом столбце вкладки **Свойства** перечислены все свойства объекта, которыми пользователь может управлять при проектировании приложения. В правом столбце указаны значения свойств, которые могут выбираться из списка или вводиться с клавиатуры.

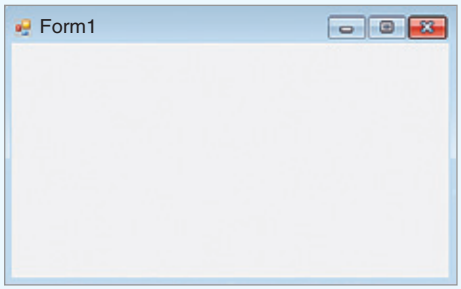
Вкладка **События** содержит список событий для объекта. Для каждого события может быть определен свой обработчик. Если обработчик для события определен, напротив события будет прописано имя процедуры (функции) обработчика.

В нижней части инспектора объектов размещено описание выбранного свойства или обработчика событий.

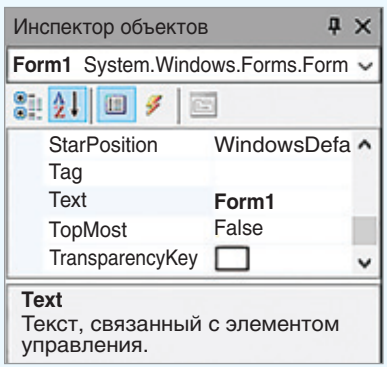
Пример 2.4. Меню и панель быстрого доступа:

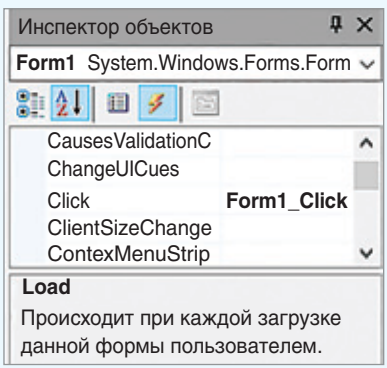


Пример 2.5. Форма:

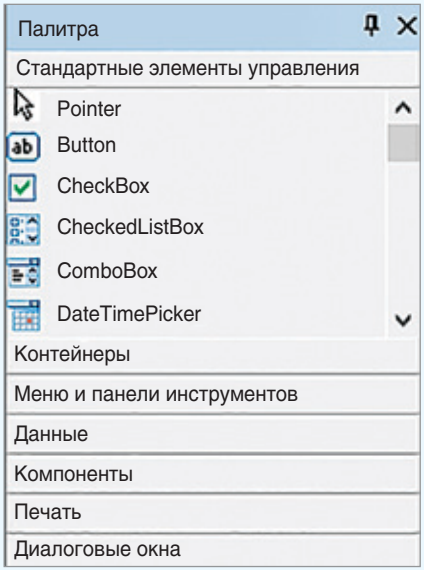


Пример 2.6. Инспектор объектов. Отображаются свойства формы:





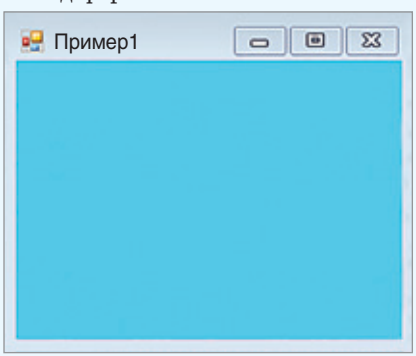
Пример 2.7. Палитра компонентов:



Пример 2.8. Изменение свойств формы в инспекторе объектов:

Свойство	Значение
Text	Пример 1
BackColor	clAqua (выбрать из списка на вкладке Интернет)
Size	250; 250
Location	200; 200

После изменения значений свойств в инспекторе объектов изменится внешний вид формы:



Палитра компонентов (пример 2.7) содержит список визуальных компонентов, объединенных в группы. Раскрытие группы происходит по щелчку с названием группы.

2.3. Работа с формой

Форма является объектом и служит для визуального отображения окна приложения. Как любой объект, форма обладает свойствами (пример 2.8).

Свойство	Назначение
Text	Заголовок формы отображается в строке заголовка окна при запуске приложения. По умолчанию — Form1
BackColor	Цвет формы. Может быть выбран один из стандартных (перечислены в списке) или задан вручную тремя числами, соответствующими RGB
Size	Высота и ширина формы. Можно указать два числа через «;» или развернуть свойство, нажав значок >, и получить возможность ввода значений Width и Height
Location	Горизонтальная и вертикальная координаты положения верхнего левого угла окна формы на экране. Можно указать два числа через «;» или развернуть свойство, нажав значок >, и получить возможность ввода значений X и Y
(Name)	Имя (внутреннее) формы. Используется в программном коде для обращения к объекту. Является идентификатором

Для создания обработчика событий формы нужно в инспекторе объектов перейти на вкладку **События** (), выбрать событие. Процедура генерируется автоматически при двойном клике мышью в пустой строке напротив выбранного события. После этого среда переключается на страницу, на которой пишется код (пример 2.9).

Имя процедуры-обработчика состоит из названия компонента, над которым происходит событие, и названия события (Form_Click).

Для каждого объекта определен обработчик по умолчанию, который создается при двойном клике по объекту. Для формы таким обработчиком будет `Form1_Load` — событие, которое происходит при загрузке формы.

Для переключения между окном программного кода и конструктором дизайна формы можно использовать вкладки **Дизайнер** и **Код** в верхней части окна приложения: Дизайнер Код.

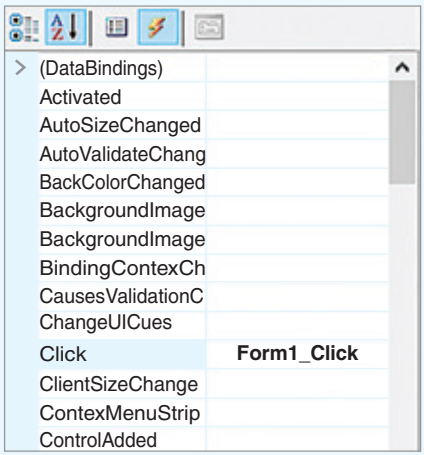
При создании процедур-обработчиков свойства объектов можно изменять программно. Для этого нужно обратиться к свойству по его имени и присвоить новое значение. Например, для изменения цвета формы нужно записать следующую команду:

```
BackColor := Color.Red;
```

Система Pascal позволяет упростить ввод сложных имен в код программы. После того как вы наберете часть сложного имени, на экране появится список со всеми свойствами и методами, которые относятся к этому объекту (пример 2.10).

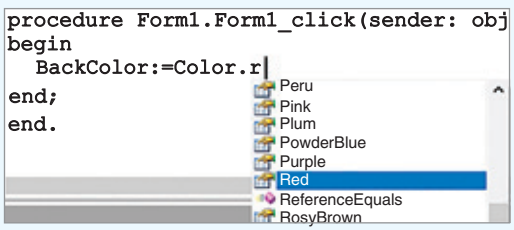
Пример 2.9. Создание обработчика события Click (клик левой клавишей мыши) для формы.

1. Выбор события в инспекторе объектов:



2. Окно программного кода со вставленным обработчиком:

Пример 2.10. Подсказка системы при вводе свойств объекта:



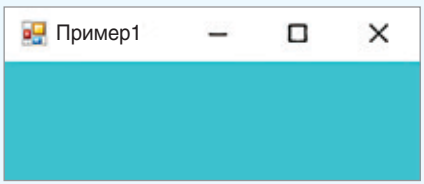
Если вы введете первые буквы названия свойства (метода), то курсор переместится в списке к тем свойствам и методам, названия которых начинаются на эти буквы. После этого нужное свойство можно вставить в программу щелчком мыши или нажатием клавиши Enter. Если список не появился, его можно вызвать комбинацией клавиш Ctrl + пробел.

Пример 2.11. Код процедуры-обработчика:

```
procedure Form1.Form1_Click
 (sender: Object; e: EventArgs);
begin
  BackColor := Color.Red;
end;
```

Чтобы убедиться в правильности работы программы, нужно запустить проект и проверить, что при выполнении щелчка мышью по форме ее цвет меняется на красный.

Форма после запуска проекта:



Форма после щелчка мыши по ней:

Пример 2.11. Создать обработчик события для щелчка левой клавишей мыши по форме, в результате которого цвет формы должен поменяться на красный (продолжить работу с примером 2.8).

Этапы выполнения задания

1. Перейти на вкладку **Events** в окне инспектора объектов.
2. Выполнить двойной щелчок в поле напротив события **OnClik**.
3. В окне редактора кода в процедуре

```
Form1.Form1_Click(sender: Object;
e: EventArgs);
```

вписать команду

```
BackColor := Color.Red;
```

Все изменения свойств формы, которые производили в примере 2.8, можно описать программно. Для этого создается обработчик события `Form1_Load`.

4. Сохранить изменения в проекте.

1. Какие элементы среды PascalABC отображаются на экране после создания проекта?
2. Какие файлы входят в состав приложения, создаваемого в PascalABC?
3. Для чего предназначена форма?
4. Для чего используют инспектор объектов?
5. Какие свойства форм вы можете назвать?
6. Как создать обработчик события?

Упражнения

1 Внесите изменения в проект из примера 2.11 так, чтобы цвет формы менялся случайно. Изменять цвет можно с помощью функции FromArgb. У этой функции четыре параметра: прозрачность (альфа-канал, интенсивность красного цвета, интенсивность зеленого цвета, интенсивность синего цвета). Генерация случайных чисел происходит следующим образом. Сначала создается переменная, являющаяся объектом класса Random (команда `var rnd: Random := new Random();`). Каждое новое случай-

ное число можно получить, обращаясь к методу `next(x)`, где `x` задает полуинтервал `[0, x)`. Команда смены цвета будет выглядеть следующим образом:

```
BackColor := Color.FromArgb(255, rnd.next(256), rnd.next(256), rnd.next(256));
```

2 Создайте проект, в котором при двойном клике мыши по форме ее размеры будут увеличиваться на 5.

1. Создайте и сохраните в новой папке проект.
2. Измените свойство Text формы на Упражнение 2.
3. Создайте обработчик события мыши `DblClick`.
4. Для изменения ширины и высоты формы можно воспользоваться командами:

```
Width := Width + 5;
Height := Height + 5;
```

5. Сохраните изменения в проекте.
6. Запустите проект и проверьте его работу.

3 Создайте проект, в котором цвет формы будет меняться при наведении на нее мыши, например с желтого на зеленый.

1. Измените свойство Text у формы на Упражнение 3.
2. Установите желтый цвет формы.
3. Создайте обработчики для двух событий мыши: `MouseEnter` и `MouseLeave`.
4. В коде события `MouseEnter` установите зеленый (Green) цвет формы, а коде события `MouseLeave` — желтый (Yellow).
5. Сохраните изменения в проекте.
6. Запустите проект и проверьте его работу.

§ 3. Проектирование интерфейса оконного приложения с использованием элементов управления

3.1. Основные элементы управления

Элементами управления называются объекты, которые используются для отображения данных или организации взаимодействия между пользователем и приложением с помощью мыши или клавиатуры. Они собраны в специальные библиотеки компонентов, которые ОС использует для обеспечения единообразного интерфейса

Для элементов управления используется и другое название — виджеты. Слово употребляется примерно с 1920-х гг. в американском английском для обозначения простой, но необходимой вещи, маленького изделия. Одним из вариантов происхождения этого слова считается словослияние «window gadget» (букв. «оконное приспособление»), также произошедшее в начале XX в.