Кафедра: автоматика и инф. технологии

Лабораторная работа

На тему:

"Создание, компиляция и отладка простого приложения"

Екатеринбург, 2008

Программа, создаваемая в среде Delphi в процессе проектирования приложения, основана на модульном принципе. Главная программа состоит из объявления списка используемых модулей и нескольких операторов, создающих объекты для необходимых форм и запускающих приложение на выполнение. Модульность очень важна для создания надежных и относительно легко модифицируемых и сопровождаемых приложений. Четкое соблюдение принципов модульности в сочетании с принципом скрытия информации позволяет производить модификации внутри любого модуля, не затрагивая при этом остальных модулей и главную программу.

Все объекты компонентов размещаются в объектах - формах. Для каждой формы, проектируемой в приложении, Delphi создает отдельный модуль. Именно в модулях и осуществляется программирование задачи. В обработчиках событий объектов размещаются описания алгоритмов, которые в основном сводятся к обработке информации, содержащейся в свойствах одних объектов, и задании по результатам этой обработки свойств других объектов.

***Структура файла главной (головной) программы приложения Delphi.***

В процессе проектирования Delphi автоматически создает код головной программы и отдельных модулей. В модули вводятся собственные коды, создавая обработчики различных событий. Но головную программу, как правило, не приходится модифицировать и даже просматривать ее текст (только в исключительных случаях).

***Головной файл*** приложения Delphi имеет следующую структуру:

**Program < имя >;**

{ объявление подключаемых модулей, а также локальных типов, классов, констант, переменных, описание локальных функций и переменных }

**Begin**

{ операторы тела программы }

**End.**

*Типичная головная* программа приложения имеет следующий вид:

**Program Project 1;**

Uses

Forms,

Unit 1 in ‘Unit 1. pas’ {Form 1}, Unit 2 in ‘Unit 2. pas’ {Form 2 };

{$R \*. res }

{ *можно поместить описание констант, переменных, функций, процедур, доступных для использования только в пределах данного файла* }

**Begin**

**Application. Initialize;**

**Application. CreateForm (TForm 1, Form 1);**

**Application. CreateForm (TForm 2, Form 2);**

**Application.run;**

**End.**

1) Программа начинается с ключевого слова **program**, после которого указывается имя программы (оно совпадает с именем файла, в котором был сохранен проект). Это же имя присваивается исполняемому файлу приложения. По умолчанию используется имя **Project1.**

2) После заголовка в тексте программы располагается предложение:

**Uses**

**Forms,**

**Unit 1 in ‘Unit 1. pas’ {Form 1}, Unit 2 in ‘Unit 2. pas’ {Form 2 };**

В этом предложении перечисляются модули, загружаемые программой. Первый модуль Forms является системным, а следующие - модулями разработанными самостоятельно форм. Данный пример подразумевает, что в проекте были созданы две формы с именами Form1, Form2 в модулях Unit1, Unit2. Заключенные в фигурные скобки название форм представляют собой комментарии.

3) Следующая строка текста - **{$R \*. res }** - представляет собой директиву компилятора, связанную с использованием файлов ресурсов. Указанный файл должен быть файлом ресурсов Windows. По умолчанию используется расширение. RES для файлов ресурсов.

4) Первый оператор в теле программы **Application. Initialize**; - инициализирует приложение, следующий за ним оператор **Application. CreateForm** (TForm 1, Form 1); и Application. CreateForm (TForm 2, Form 2); - создают объекты формы1 и формы2, последний оператор **Application.run**; - начинает выполнение приложения.

***Общая структура файла модуля:***

**Unit < имя модуля >;**

**Interface** // Открытый интерфейс модуля (могут помещаться списки подключаемых модулей, объявление типов, констант, переменных, функций и процедур, к которым будет доступ из других модулей)

**Implementation** // Реализация модуля (могут помещаться списки подключаемых модулей, объявление типов, констант, переменных, функций и процедур, к которым не будет доступа из других модулей)

**Initialization // (**Необязательный раздел - операторы выполняются один раз при первом обращении к модулю)

**Finalization // (**Необязательный раздел - операторы выполняются при любом завершении работы модуля)

**End.**

Рассмотрим теперь текст модуля с пустой формой:

**Unit Unit1;**

**Interface //** Открытый интерфейс модуля

**Uses** *{ Список подключаемых модулей }*

**Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms, Dialogs;**

**Type** *{ Объявление класса формы }*

**TForm1 = class (TForm)**

**Private //** закрытый раздел класса

**Public //** открытый раздел класса

**End;**

**Var**

**Form 1: TForm1;**

**Implementation //** Реализация модуля

**{$R \*. dfm }**

**End.**

1) Модуль начинается с ключевого слова **UNIT**, после которого указывается имя модуля. Оно совпадает с именем файла, в котором был сохранен модуль. Текс модуля состоит из двух основных разделов: открытый интерфейс модуля и реализация модуля. Внешние модули не могут видеть типы, переменные, константы, функции и процедуры, размещенные в разделе реализации.

2) После предложения **USES**, содержащего список подключаемых модулей, располагается заготовка объявления класса формы. Имя класса формы - TForml. Класс содержит два раздела: открытый раздел класса (PUBLIC) - то, что объявлено в этом разделе доступно для других классов и модулей, закрытый раздел класса (PRIVATE) - то, что объявлено в этом разделе доступно в пределах данного модуля.

3) За объявлением класса формы следуют строки:

**Var**

**Form 1: TForm1;**

Таким образом, объявляется переменная Form1 класса TForm1.

4) Затем следует пустой раздел реализации **INPLEMENTATION**, в котором содержится только директива компилятора {$R \*. dfm }, обеспечивающая компоновку файлов ресурсов форм.

Задание 1. Создать приложение, которое обеспечивает ввод двух целых чисел, по щелчку на кнопке с символом "=" вычисляет их сумму и выводит значение результата.

Решение:

После запуска Delphi создайте новый проект при помощи команды File>New>Application (Файл>Создать>Приложение). В результате создания проекта приложения в окне будет раскрыто окно формы, на которой можно размещать визуальные компоненты проекта.

Далее выполните следующее:

Сохраните новый проект File>Save Project As (Файл>Сохранить проект как).

В окне Save Unit1 As (Сохранить Unit1 как) откройте диск S и создайте новую папку для файлов создаваемого проекта и назовите ее "Сумма двух целых чисел".

Затем в этом же окне откройте созданную папку и нажмите кнопку СОХРАНИТЬ.

После сохранения файла модуля Unit1. pas откроется окно Save Project As (Сохранить проект как). Задайте имя проекта "SUMMA" и нажмите кнопку СОХРАНИТЬ.

**Изменим свойства формы:**

Для изменения размеров формы достаточно захватить угол окна формы и, не отпуская левую кнопку мыши, перетащить мышь, задавая требуемый размер формы (или указать в свойствах размеры: ***Client Height*** - 489, ***Client Width*** - 730).

Измените надпись в заголовке формы с помощью Инспектора объектов. Для этого выберите свойство ***Caption*** (Заголовок) и задайте новое значение - "Сумма двух целых чисел".

Измените цвет формы с помощью Инспектора объектов. Для этого выберите свойство ***Color*** и измените его значение на clWindow (цвет окна), поверхность формы изменит свой цвет.

**Размещение на форме компонентов** Edit1, Edit2, Edit3, Label1, Button1. Для размещения компонентов на форме выполните следующее:

Откройте палитру компонентов Standard.

Затем щелкните на значке соответствующего компонента и перетащите его на форму.

Задайте положение и размер компонента при помощи мыши, как показано на рисунке (рис.1):

Edit 1

Edit 2

Edit 3

Label 1

Button1

Сумма двух целых чисел

Рис.1 Пример формы с компонентами

**Изменим свойства компонентов** на форме:

Выберите компонент Label1 и задайте свойство Caption компонента Label1 - "плюс".

Измените цвет, стиль и высоту шрифта компонента Label1. Для этого выберите свойство ***Font*** и щелкните на рядом расположенном знаке плюс. Откроется таблица свойств. Введите следующие значения: ***Color*** - clPurple, ***Size*** - 20.

Аналогичным образом задайте свойство ***Caption*** компонента Button1 - "равно" и размер и цвет символов.

Удалите текст Edit1, Edit2, Edit3 в соответствующих компонентах. Для этого выберите свойство ***Text*** и удалите текст.

Добавьте на форму три объекта Label, расположите их над объектами Edit и задайте их свойствам ***Caption*** значения "Слагаемое", "Слагаемое" и "Сумма" соответственно.

Измените цвет, стиль и высоту шрифта добавленных компонентов используя свойство ***Font*** и ***Style***.

Сохраните изменения, внесенные в проект, нажатием кнопки Save All (сохранить все) в стандартной панели инструментов.

Для просмотра содержимого проекта Delphi откройте окно Проводника и просмотрите папку проекта. Проект состоит из форм, модулей, установок параметров проекта, ресурсов и т.д. Вся эта информация размещается в файлах. Многие из этих файлов автоматически создаются Delphi, когда строится приложение. Главной частью приложения является файл summa. dpr - файл проекта, содержащий код на языке Object Pascal, с которого начинается выполнение программы и который обеспечивает инициализацию других модулей. Он создается и модифицируется Delphi автоматически в процессе разработки приложения. Имя, присваиваемое файлу проекта при его сохранении, становится именем исполняемого файла (в нашем примере - summa. exe). Другие файлы проекта:

Unit1. pas - файл модуля;

Unit1. dfm - файл формы;

Summa. dof - файл параметров проекта;

Summa. res - файл ресурсов;

Unit1. ~dfm, Unit1. ~pas - файлы резервных копий.

После просмотра списка файлов проекта закройте окно Проводника и активизируйте окно Delphi.

Запустите программу на выполнение при помощи команды **Run→Run** или нажатием кнопки **Run** на панели инструментов или клавиши **F9**. После этого на экране компьютера появляется окно созданной вами формы с компонентами, в которые можно вводить значения, но приложение не будет выполнять каких-либо вычислений, т.к выполняющий вычисления фрагмент программы не был создан. Завершите работу приложения нажатием клавиш **Alt + F4**.

**Создание кода - обработчика события**.

Для того чтобы приложение выполняло вычисление при щелчке на кнопке Button1 с изображением символа "равно", следует написать код обработки этого события. Для создания кода обработки события следует воспользоваться страницей EVENTS (Событие) в окне Инспектора объектов.

Активизируйте объект Button1 и выберите на странице Событий событие OnClick (это событие нажатия левой кнопки мыши по данному объекту).

Произведите двойной щелчок на пустом поле списка этого события. После этого курсор помещается в окно Редактора кода. В этом окне в разделе ***interface***находится запись процедуры обработчика события:

**Procedure TForm1. Button1Click (Sender: TObject);**

В разделе ***implementation*** располагается текст заготовки этой процедуры:

**Procedure TForm1. Button1Click (Sender: TObject);**

**Begin**

**…**

**End;**

Курсор будет находиться в пустой строке между ключевыми словами *begin* и *end*. Этот код является заготовкой для обработчика события, которую автоматически создала интегрированная среда Delphi.

Так как в задаче требуется выполнить сложение двух целых чисел, а результат их сложения также является целым числом, то в разделе описания переменных следует ввести следующее описание: **var a, b, c: integer; -** два слагаемых и сумма.

Так как в приложении для ввода чисел-слагаемых используются окна редактирования Edit1, Edit2, то необходимо при помощи процедуры **StrToInt** преобразовать строки в целые числа. Для вывода результата суммирования в окне редактирования Edit3 нужно преобразовать число в строку функцией **IntToStr**. Поэтому в основном теле процедуры обработки события следует ввести следующий текст:

**a: = StrToInt (Edit1. text);**

**b: = StrToInt (Edit2. text);**

**c: = a + b;**

**Edit3. text: = IntToStr (c);**

Целиком процедура обработки события щелчка на кнопке Button1 будет выглядеть следующим образом:

**Procedure TForm1. Button1Click (Sender: TObject);**

**var a, b, c: integer;**

**Begin**

a: = StrToInt (Edit1. text); {преобразование текстовой строки в целое число}

b: = StrToInt (Edit2. text);

c: = a + b;

Edit3. text: = IntToStr (c); {преобразование целого числа в текстовую строку}

**End;**

Запустите приложение на выполнение. Введите два слагаемых и, нажав кнопку с надписью "равно", запустите процедуру обработки события - вычисление суммы двух целых чисел.

Сохраните все изменения внесенные в проект и закройте его.