2017

Глава 9 Расширение объектной модели Delphi

МГТУ им. Н.Э. Баумана
Факультет Информатика и системы управления
Кафедра Компьютерные системы и сети
Лектор: д.т.н., проф.
Иванова Галина Сергеевна

9.1 Свойства

Свойство - это средство Pascal Delphi, позволяющее определять интерфейс доступа к полям и методам класса.

В Delphi различают:

- простые или скалярные свойства;
- свойства-массивы;
- индексируемые свойства или свойства со спецификацией index;
- процедурные свойства.

Простые свойства

Используются для ограничения доступа к полю и при необходимости выполнения дополнительных действий при чтении и записи.

Property <Имя свойства>:<Тип>

[read <Метод чтения или имя поля>]

[write <Метод записи или имя поля>]

[stored <Метод или булевское значение>]

[default <Константа>];

read - eсли метод чтения не определен, то свойство не доступно для чтения;

write - если метод записи не определен, то свойство не доступно для записи;

stored – для опубликованных свойств – хранить ли значение в файле формы .dfm;

default - – для опубликованных свойств – значение по умолчанию.

Простые свойства (2)

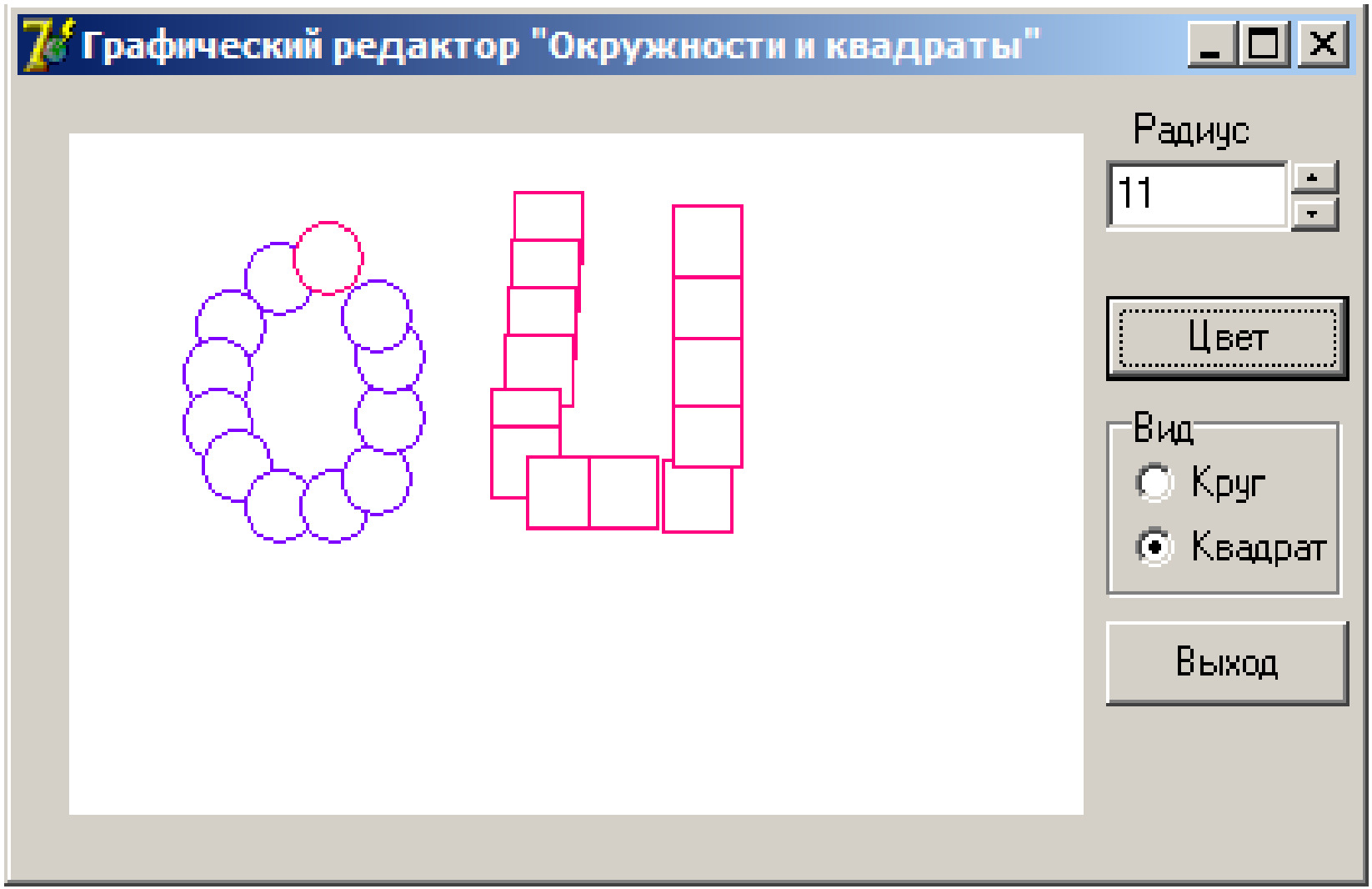
Пример:

```
private FValue:integer;
        procedure SetValue(AValue:integer);
        function StoreValue:boolean;
 published
     property Value:integer
         read FValue write SetValue
         stored StoreValue default 10; . . .
```

Обращение в программе:

```
A.Value := n;   {A.SetValue(n);}
K := A.Value;   {K := A.FValue;}
```

Примитивный графический редактор (Ex 9.1)



Объектная декомпозиция

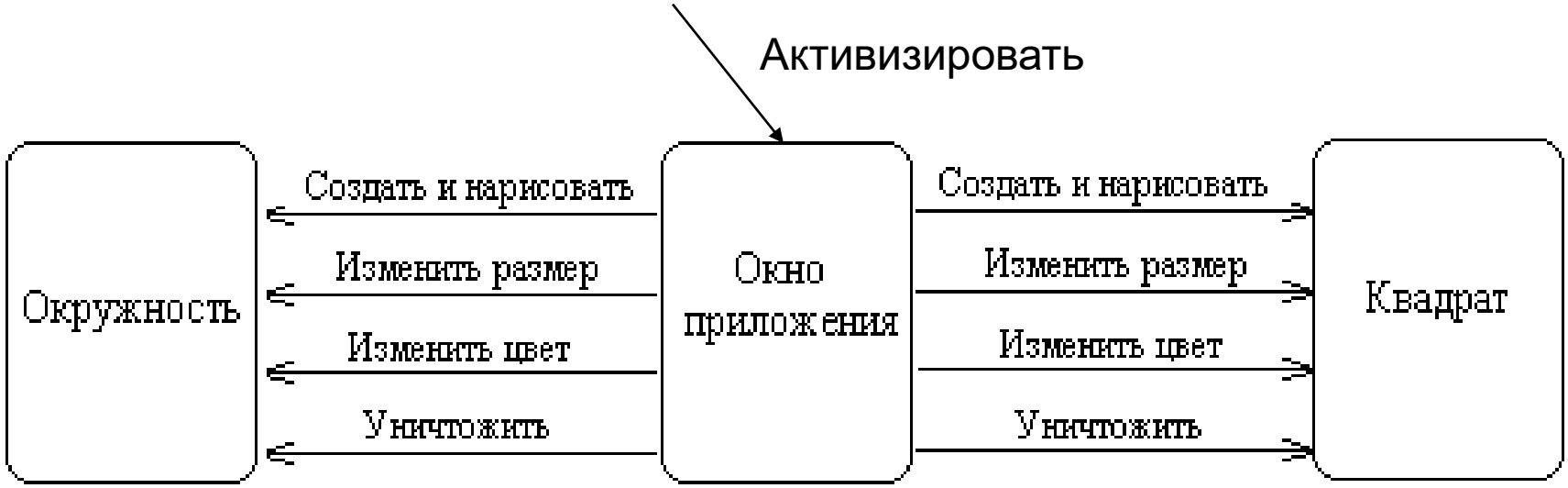
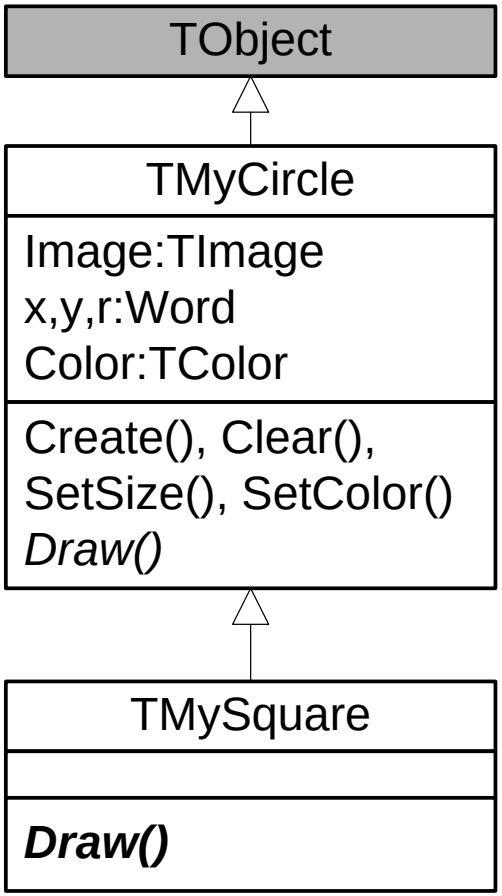
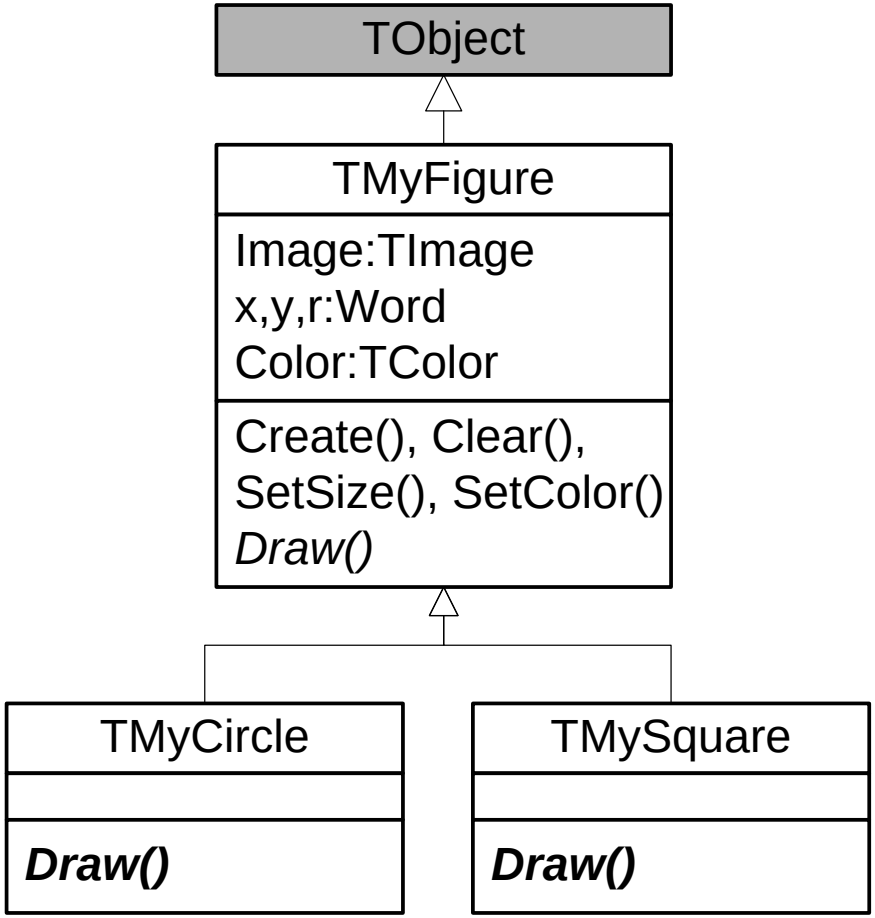


Диаграмма классов предметной области





Модуль Figure

```
Unit Figure;
 Interface
  Uses  extctrls,Graphics;
  Type TMyFigure=class
       private    x,y,FRadius:Word;
          FColor:TColor; Image:TImage;
          procedure Clear;
          procedure SetSize(ar:word);
          procedure SetColor(aColor:TColor);
       public
          Constructor Create(aImage:TImage;
                             ax,ay,ar:Word;aColor:TColor);
          Procedure Draw; virtual; abstract;
          Property Radius:Word write SetSize;
          Property Color:TColor write SetColor;
        end;
```

Модуль Figure (2)

```
 TMyCircle=class(TMyFigure)
      public   Procedure Draw; override;
 end;
 TMySquare=class(TMyFigure)
      public   Procedure Draw; override;
 end;
Implementation
   Constructor TMyFigure.Create;
      Begin
            inherited Create;
            Image:=aImage;
            x:=ax;    y:=ay;
            FRadius:=ar;
            FColor:=aColor;
            Draw;
      End;
```

Модуль Figure (3)

```
Procedure TMyFigure.Clear;
  Var TempColor:TColor;
  Begin      TempColor:=FColor;
             FColor:=Image.Canvas.Brush.Color;
             Draw;
             FColor:=TempColor;
  End;
Procedure TMyFigure.SetSize;
  Begin      Clear;
             FRadius:=ar;
             Draw;
  End;
Procedure TMyFigure.SetColor;
   Begin     Clear;
             FColor:=aColor;
             Draw;
   End;
```

Модуль Figure (4)

```
Procedure TMyCircle.Draw;
  Begin
      Image.Canvas.Pen.Color:=FColor;
      Image.Canvas.Ellipse(x-FRadius,y-FRadius,
                                   x+FRadius,y+FRadius);
  End;

Procedure TMySquare.Draw;
  Begin
     Image.Canvas.Pen.Color:=FColor;
     Image.Canvas.Rectangle(x-FRadius, y-FRadius,
                                   x+FRadius,y+FRadius);
  End;
End.
```

Модуль Main

```
unit Main;
interface
uses
    Windows,Messages,SysUtils,Variants,Classes,Graphics,
    Controls,Forms,Dialogs,ComCtrls, StdCtrls, ExtCtrls;
type
  TMainForm = class(TForm)
    Image: TImage;
    ColorButton: TButton;
    ExitButton: TButton;
    RadioGroup: TRadioGroup;
    rLabel: TLabel;
    rEdit: TEdit;
    UpDown: TUpDown;
    ColorDialog: TColorDialog;
```

Модуль Main (2)

```
    procedure FormActivate(Sender: TObject);
    procedure ImageMouseDown(Sender: TObject;… );
    procedure UpDownClick(Sender: TObject; …);
    procedure ColorButtonClick(Sender: TObject);
    procedure ExitButtonClick(Sender: TObject);
  end;
var MainForm: TMainForm;
implementation
uses Figure;
Var C:TMyFigure;
{$R *.dfm}
procedure TMainForm.FormActivate(Sender: TObject);
begin
    Image.Canvas.Brush.Color:=clWhite;
    Image.Canvas.Pen.Color:=clBlack;
end;
```

Модуль Main (3)

```
procedure TMainForm.ImageMouseDown(Sender: TObject;
    Button:TMouseButton;Shift:TShiftState;X,Y:Integer);
begin
  if Button=mbLeft then
    case RadioGroup.ItemIndex of
    0: begin
         C.Free;
         C:=TMyCircle.Create(Image,X,Y,
           strtoint(rEdit.Text),Image.Canvas.Pen.Color);
       end;
    1: begin
         C.Free;
         C:=TMySquare.Create(Image,X,Y,
           strtoint(rEdit.Text),Image.Canvas.Pen.Color);
       end;
    end;
end;
```

Модуль Main (4)

```
procedure TMainForm.UpDownClick(Sender:TObject;
                                Button:TUDBtnType);
    begin
       if C<>nil then C.Radius:=strtoint(rEdit.Text);
    end;
procedure TMainForm.ColorButtonClick(Sender: TObject);
    begin
       if ColorDialog.Execute then
          Image.Canvas.Pen.Color:=ColorDialog.Color;
       if C<>nil then C.Color:=Image.Canvas.Pen.Color;
    end;
procedure TMainForm.ExitButtonClick(Sender:TObject);
    begin   Close;  end;
initialization
finalization C.Free;
end.
```