Основы ООП в Delphi

Дополнительные лекции
ОАиП

Предпосылки возникновения ООП

- Рост сложности кода:
  - следование/ветвление/цикл;
  - процедуры;
  - модули;
  - …
- Рост сложности данных:
  - скалярные типы;
  - агрегированные (составные) типы;
  - …

Основы ООП

- ООП — парадигма (идея).

Программа состоит из объектов, которые обмениваются сообщениями

Эволюция ООП

```
type
  TButton = record
    X, Y, W, H: Integer;
    Visible: Boolean;
    Caption: String;
  end;

procedure SetPos(var Button: TButton; X, Y: Integer);
procedure SetSize(var Button: TButton; W, H: Integer);
procedure Hide(var Button: TButton);
procedure Show(var Button: TButton);
procedure Draw(const Button: TButton);
...
```

Эволюция ООП

```
type
  TListBox = record
    X, Y, W, H: Integer;
    Visible: Boolean;
    Items: array of String;
  end;

procedure SetPos(var ListBox: TListBox; X, Y: Integer);
procedure SetSize(var ListBox: TListBox; W, H: Integer);
procedure Hide(var ListBox: TListBox);
procedure Show(var ListBox: TListBox);
procedure Draw(const ListBox: TListBox);
...
```

Эволюция ООП

```
procedure Button_SetPos(var Button: TButton; ... );
procedure Button_SetSize(var Button: TButton; ... );
procedure Button_Hide(var Button: TButton);
procedure Button_Show(var Button: TButton);
procedure Button_Draw(const Button: TButton);
...

procedure ListBox_SetPos(var ListBox: TListBox; ... );
procedure ListBox_SetSize(var ListBox: TListBox; ... );
procedure ListBox_Hide(var ListBox: TListBox);
procedure ListBox_Show(var ListBox: TListBox);
procedure ListBox_Draw(const ListBox: TListBox);
...
```

Эволюция ООП

```
type
  TControl = record
    X, Y, W, H: Integer;
    Visible: Boolean;
  end;
  TButton = record
    ControlData: TControl;
    Caption: String;
  end;
  TListBox = record
    ControlData: TControl;
    Items: array of String;
  end;
  ...
```

Эволюция ООП

```
procedure Control_SetPos(var Control: TControl; ... );
procedure Control_SetSize(var Control: TControl; ... );
procedure Control_Hide(var Control: TControl);
procedure Control_Show(var Control: TControl);
...

procedure Button_Draw(const Button: TButton);
procedure ListBox_Draw(const ListBox: TListBox);
...

Control_SetPos(Button.ControlData, 100, 200);
Button_Draw(Button);
```

Основы ООП

- «Три кита» ООП:
  - инкапсуляция;
  - наследование;
  - полиморфизм.

Инкапсуляция

Объединение данных и кода,
который их обрабатывает

Наследование

Возможность создавать
новые типы данных
путём расширения существующих

Полиморфизм

Возможность использовать различные типы данных взаимозаменяемо

Эволюция ООП

```
type
  TSetPosProc = procedure (var Control: TControl; ... );
  TDrawProc = procedure (var Control);
  ...

  TButton = record
    ControlData: TControl;
    Caption: String;

    SetPos: TSetPosProc;
    Draw: TDrawProc;
  end;
  ...
```

Эволюция ООП

```
type
  TButton = record
    ControlData: TControl;
    Caption: String;

    SetPos: TSetPosProc;
    Draw: TDrawProc;
  end;
  ...

MyButton.SetPos(MyButton, 100, 200);
MyButton.Draw(MyButton);
```

Эволюция ООП

- ООП не привязан к ЯП:
  - Писать в ООП-стиле можно на любом языке программирования.
  - Но проще, если в ЯП есть поддержка ООП (на уровне синтаксиса).

ООП в Delphi

```
type
  TControl = class
    X, Y, W, H: Integer;
    Visible: Boolean;
    procedure SetPos(X, Y: Integer);
    procedure SetSize(W, H: Integer);
    procedure Hide;
    procedure Show;
    procedure Draw;
  end;
```

ООП в Delphi

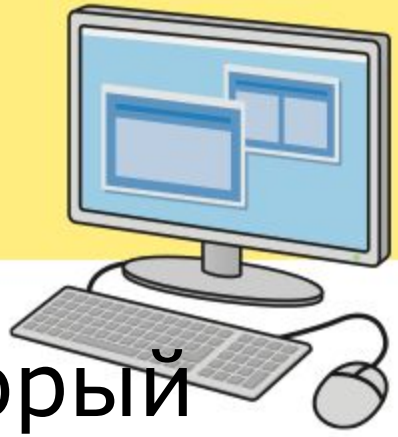
- **Класс** — тип данных, элементами (членами) которого являются:
  - поля (данные);
  - методы (код).

- **Объект** — экземпляр класса, т.е. конкретное значение этого типа.

ООП в Delphi

- Методы могут быть:
  - статическими;
  - виртуальными;
  - динамическими;
  - методами класса.
- Особо выделяют:
  - конструкторы;
  - деструкторы.

ООП в Delphi



- **Конструктор** — метод, который вызывается при создании объекта
  - Его задача — инициализация полей.
  - Т.е. приведение полей объекта в согласованное состояние.
- **Деструктор** — метод, который вызывается при удалении объекта
  - Его задача — освободить выделенные объектом ресурсы.

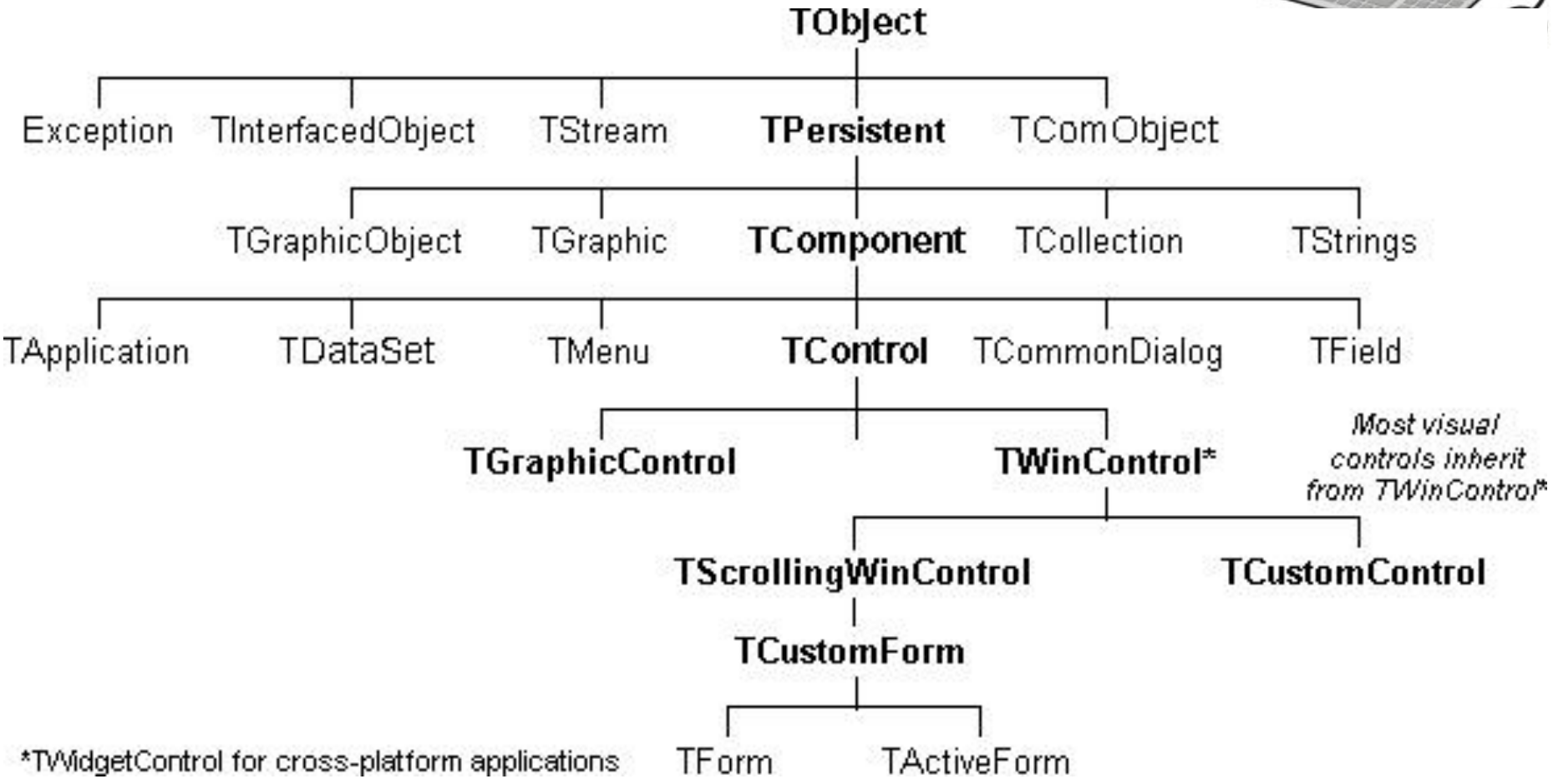
ООП в Delphi

- Все объекты в Delphi выделяются в динамической памяти.
  - Переменная хранит адрес объекта.
- В теле каждого метода доступна переменная Self.
  - Содержит адрес объекта, у которого был вызван метод.
  - (Кроме методов класса)

ООП в Delphi

- Класс может наследоваться от другого класса.
  - При этом наследуемый класс будет содержать все члены родительского.
  - Но сможет добавить новые…
  - … или изменить видимость существующих.
  - Если не указать родительский класс, им будет TObject.

Классы и объекты в Delphi



ООП в Delphi

- Членам класса могут задаваться различные уровни видимости:
  - private
  - protected
  - public
  - published

ООП в Delphi

- **private**
  - внутри методов этого же класса.
- **protected**
  - внутри методов этого же класса…
  - … или наследуемых от него.
- **public**
  - везде.

ООП в Delphi

- В Delphi есть некоторые особенности:
  - **private**-члены доступны всем внутри модуля, в котором объявлен класс;
  - есть **published**-видимость (используется для интеграции со средой);
  - по умолчанию — **public**/**published**.

ООП в Delphi

```
type
  TfrmMain = class(TForm)
    pbField: TPaintBox;
    procedure pbFieldPaint(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure OnGameUpdate(Sender: TObject);
  private
    Game: TGame;
  public
    { Public declarations }
  end;

var
  frmMain: TfrmMain;
```

ООП в Delphi

- **Свойство** — способ доступа к внутреннему состоянию объекта, имитирующий поведение переменной.
  - Внутреннее состояние объекта задаётся значениями его полей.
- Свойство может быть реализовано:
  - прямым доступом к какому-либо полю;
  - в виде метода, вызываемого при обращении к свойству.

Свойства

```
TMyClass = class(TMyBaseClass)
private
  FColor: TColor;
protected
  function GetColor: TColor;
  procedure SetColor(const Value:
    TColor);
public
  property Color: TColor read FColor write SetColor;
end;
```

Свойства-массивы

```
TMyClass = class
private
  ...
protected
  ...
public
  ...
  property Items[Index: Integer]: TObject
    read GetObject write SetObject;
  property Values[const Name: String]: String
    read GetValue write SetValue;
  ...
end;
```

Индексированные свойства

```
TRectangle = class
private
  FCoordinates: array [0..3] of LongInt;
  function GetCoordinate(Index: Integer): LongInt;
  procedure SetCoordinate(Index: Integer; Value: LongInt);
public
  property Left: LongInt   index 0 read GetCoordinate
                                    write SetCoordinate;
  property Top: LongInt    index 1 read GetCoordinate
                                    write SetCoordinate;
  property Right: LongInt  index 1 read GetCoordinate
                                    write SetCoordinate;
  property Bottom: LongInt index 1 read GetCoordinate
                                    write SetCoordinate;
end;
```

События

- **Событие** — сообщение, возникающее в различных местах программы при выполнении определённых условий:
  - нажатие на кнопку;
  - выбор элемента в списке;
  - и т.д.
- Обработчик события — ~~процедура~~ метод, определяющая реакцию программы на то или иное событие.

События

```
TOnClickEvent = procedure(X, Y: Integer) of object;

TMyClass = class
private
  FOnClick: TOnCickEvent;
public
  property OnClick: TOnClickEvent read FOnClick
                                  write FOnClick;
end;
```

Интерфейсы

- **Интерфейс** — тип данных, определяющий только интерфейсную часть объекта.
  - Один класс может *реализовывать* несколько интерфейсов.
  - Для переменных типа «интерфейс» управление памятью производится автоматически.

Интерфейсы

```
IMyInterface = interface(IInterface)
  ['{00000002-0000-0000-C000-000000000046}']
  function DoSomething(Size: Integer): Pointer;
  procedure DoAnything(const S: String);
  property MyProp read GetMyProp write SetMyProp;
end;

TMyCoolClass = class(TBaseClass, IMyInterface)
  ...
end;

TAnotherClass = class(IInterface1, IInterface2)
  function IInterface1.Jump = Jump;
  function IInterface2.Jump = JumpHigher;
end;
```

Вопросы?