Лабораторная работа №2

Конфигурационные файлы. Режимы design-time и run-time

Определение понятия КОМПОНЕНТ

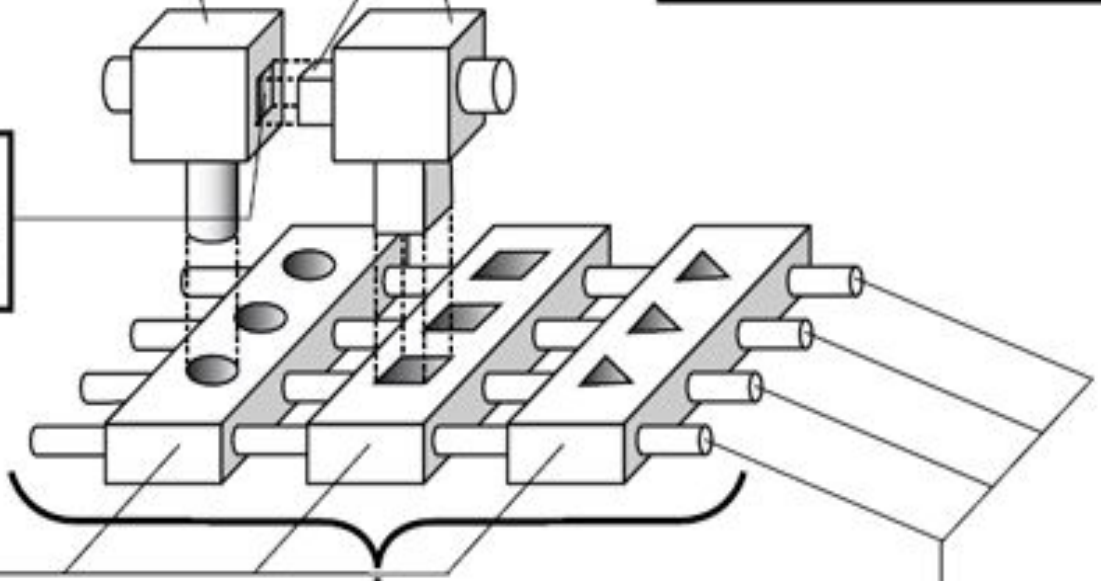
Компонент – это структурная единица программной системы, обладающая четко определенным интерфейсом, который полностью описывает ее зависимости от окружения.

Такой компонент может быть независимо поставлен или не поставлен, добавлен в состав некоторой системы или удален из нее, в том числе, может включаться в состав систем других поставщиков.

Компоненты
Предоставляют или требуют интерфейсы
Соответствуют компонентной модели
Развертываются в компонентной среде

Интерфейсы
Предоставляемые компонентами
Описываются контрактами

Интерфейс
Требуемый компонентом
Описывается контрактом



Компонентная модель
Определяет требования к работающим в ее рамках компонентам, виды компонентов
Иногда требует реализовать определенные интерфейсы

Компонентная среда
В ней развертываются компоненты
Предоставляет компонентную модель и набор базовых служб

Базовые службы
Обеспечивают работоспособность набора компонентов

- С точки зрения среды визуальной разработки приложений *компоненты* (**взгляд снаружи**) – это самодостаточные строительные блоки, которыми необходимо пользоваться при создании приложений.

- С точки зрения программы на языке Delphi (**взгляд изнутри**) компоненты – это классы, порожденные прямо или косвенно от класса TComponent**.**

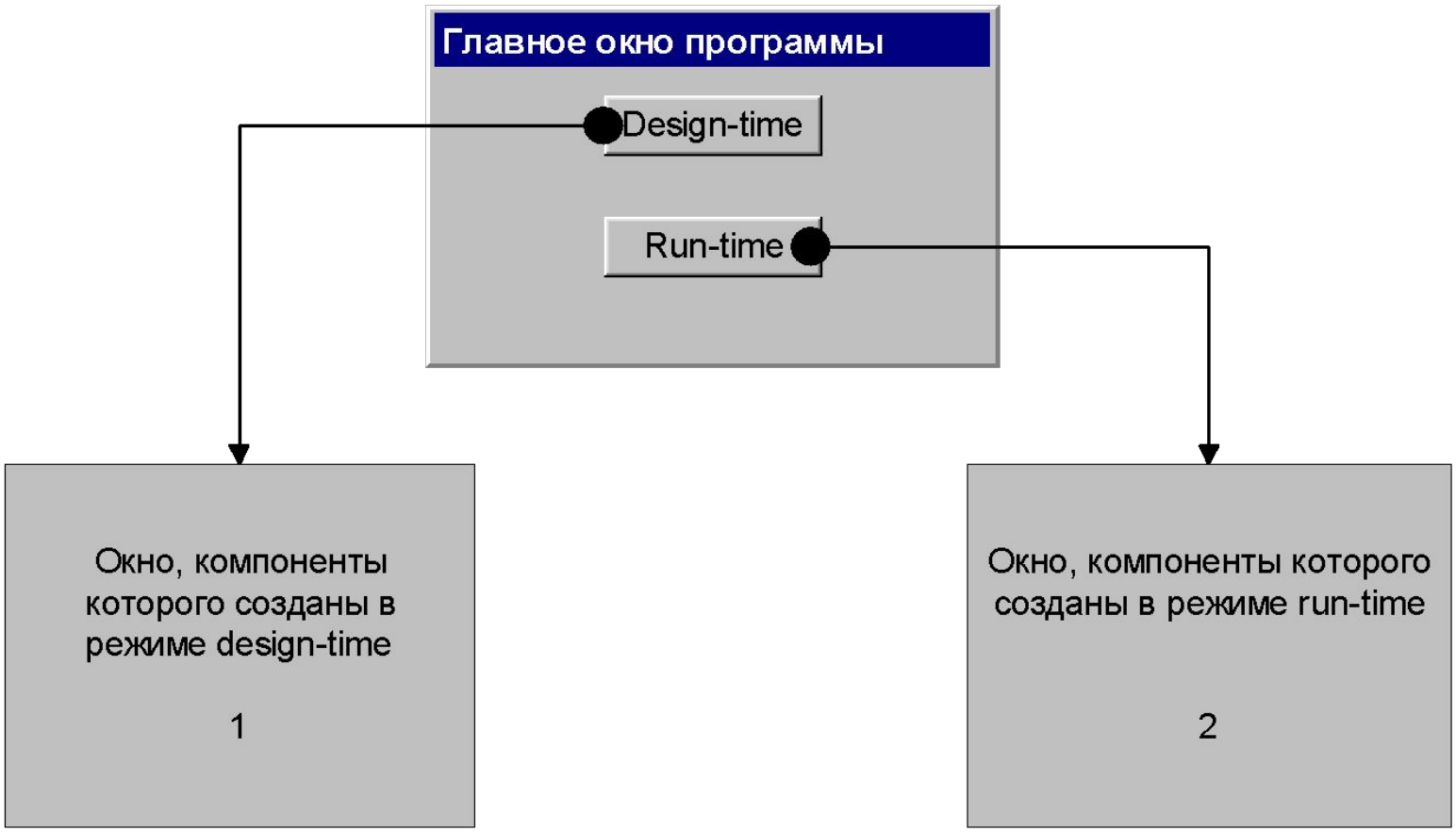
Природа компонента троична

Компонент = состояние (свойства) +
поведение (методы) +
обратная реакция (события)

Постановка задачи

Разработать приложение, структура интерфейса которого приведена на рисунке 1. Форма 1 должна быть создана в режиме design-time. Для создания формы 2, необходимо, прочитать структуру компонентов из ini-файла; создать данные компоненты на форме 2; определить их свойства, руководствуясь информацией, полученной из файла; переопределить событие OnClick для некоторых компонентов. При выходе из формы 2 все объекты, созданные динамически, должны быть уничтожены.

Структура интерфейса



Общая структура INI-файла:

< [Секция 1] >

<Поле 1> = <Значение 1>

…..

<Поле n> = <Значение n>

…..

< [Секция k] >

<Поле n+1> = <Значение n+1>

…

<Поле n+m> = <Значение n+m>

Пример:

```
[BASE]
BASEPATH = .\Resources\dbs\pi.fdb

[MENU]
SIZE = 12
COLOR = CLMENUHIGHLIGHT
```

Создание объекта TIniFile:

```
var
  Ini: Tinifile;
  …
  Ini:=TiniFile.Create(extractfilepath(Application.ExeName)+'Name.ini');
```

Запись данныхв ini-файл:

```
  WriteInteger(const Section: string, const Ident:string, Value: Integer)
  WriteStringInteger(const Section: string, const Ident:string, Value: String)
  WriteBool(const Section: string, const Ident:string, Value: Boolean)
```

Чтение данных из ini-файла:

```
ReadInteger(const Section: string, const Ident:string, DefaultValue:   Integer)
ReadStringInteger(const Section: string, const Ident:string, DefaultValue: String)
ReadBool(const Section: string, const Ident:string, DefaultValue: Boolean)
```

Процедурные типы (procedural types) и объектные процедурные типы (method pointers)

Основные этапы создания компонентов в динамическом режиме (Run-time)

- Объявить ссылку на компонент;
- Выделить память под компонент (вызов конструктора);
- Задать свойства компонента;
- Подключение обработчиков событий.

Пример

```
var
   Object: TSampleClass;
    …
   Object :=
   TSample.Create(…);
    …
    Object.Free;
    Object := nil;
```