**Тема 4.2 Описание интерфейса среды. Компоненты визуализации данных**

Вопросы:

* 1. Этапы создания приложения.
  2. Интерфейс информационных систем.
  3. Описание интерфейса среды для разработки пользовательского приложения. Обзор компонентов для визуализации данных.

**Цель**: сформировать знания основных этапов создания приложения для работы с базой данных, описать основные виды интерфейсов информационных систем.

**Этапы создания приложения**

Если необходимо разработать приложение, данные которого будут храниться на источнике данных (в базе данных MS SQL Server или на клиент-серверной системе), то есть смысл последовательно выполнить все рекомендованные этапы проектирования и создания таких приложений.

Первый этап — *сбор информации о предметной области*: какие унаследованные системы на нем работают, с чем нужно обеспечивать совместимость, как организованы информационные потоки, какова изменчивость системы и т.п. Часто при этом используются специальные средства, такие, как Visio и ERWin.

Второй этап — *выбор архитектуры приложения, выбор подходящей системы управления базами данных и проектирование СУБД*. На этом этапе определяется, будет ли информация храниться с использованием ядра Jet или клиент-серверной системы, проектируется система таблиц для хранения информации и отношений между ними, проектируются некоторые другие объекты базы данных. Кроме того, определяется архитектура приложения — сколько в ней будет уровней, будут ли использоваться терминальные или Web-технологии, будет ли применяться репликация и т.п.

Третий этап — *реализация СУБД и бизнес-логики приложения*. На этом этапе проектируются, создаются, настраиваются, заполняются исходными данными объекты базы данных — таблицы, представления, хранимые процедуры, функции, триггеры, отчеты и т.п. При создании приложений в MS SQL Server большая часть этих операций выполняется при помощи графического интерфейса разработчика. Код языка программирования используется для проверки вводимых пользователем значений, для работы с элементами управления на форме, переключения между формами, отчетами, другими элементами управления, обращения к внешним объектным моделям и т.п. На этом этапе опять-таки могут помочь Visio и ERWin.

Четвертый этап — *оптимизация производительности базы данных*, этап, который часто упускается разработчиками. Задача эта комплексная, но включает в себя в том числе и оптимизацию кода.

Пятый этап — *тестирование и отладка приложения*.

Шестой этап — *развертывание приложения*.

**Интерфейс информационных систем.**

В системах построенных по *технологии клиент-сервер* существует два вида интерфейса:

* Интерфейс, реализуемый при помощи клиентского приложения;
* Web -интерфейс.

*Интерфейс*, реализуемый при помощи клиентского приложения - это компьютерная *программа*, устанавливаемая на клиентские компьютеры, предназначенная для работы с файлами данных через *сеть*. Основными элементами клиентских приложений являются формы (окно программы) и отчёты.

*Элементы управления* на форме называется объектами. Каждый *объект* обладает своим набором свойств, событий и методов.

* **Свойства объекта** - это его характеристики (высота, ширина и т.д.);
* **События объекта** - это события операционных систем или события инициируемые пользователем, на которые может *реагировать объект* (нажатие кнопки);
* **Методы объекта** - действия, которые можно производить с объектом в ходе выполнения программ.

В *БД* все объекты форм делятся на два класса:

* Объекты управления - объекты, осуществляющие управление БД (Например: Кнопка или Выпадающий список);
* Объекты для отображения информации - элементы, отображающие содержимое таблиц, запросов или фильтров, позволяющие добавлять изменять и удалять информацию, и проводить ее анализ.

Все формы в клиентском приложении делятся на три группы:

1. **Формы для работы с данными** - формы, содержащие как объекты управления, так и объекты просмотра данных. Такие формы предназначены для отображения, изменения, удаления и анализа данных;
2. **Кнопочные формы** - формы, содержащие только объекты управления, предназначаются для открытия всех других форм. Кнопочная форма, которая появляется первой после запуска программы, называется, *главной кнопочной формой*.
3. **Информационные и служебные формы** - формы, содержащие только элементы управления, предназначены для отображения служебной информации (справки), несвязанной с таблицами, запросами и фильтрами, либо для выполнения служебных операций не связанных с данными (Например: форма с калькулятором)

Существует два вида дизайна форм:

1. Ленточные формы - формы, выводящие информацию по одной записи;
2. Табличные формы - формы выводящие информацию в виде таблицы.

Объекты связи используются только в клиентском интерфейсе. В web-интерфейса функции объекта связи выполняет *сервер*.

Основой web-интерфейса являются страницы (*файл* с расширенным htm или html). Работа со страницами осуществляется с помощью программы - браузера. Изначально страницы находятся на сервере, *пользователь* сначала загружает их на свой *компьютер* с сервера, а затем с помощью страниц *пользователь* работает с файлом данных.

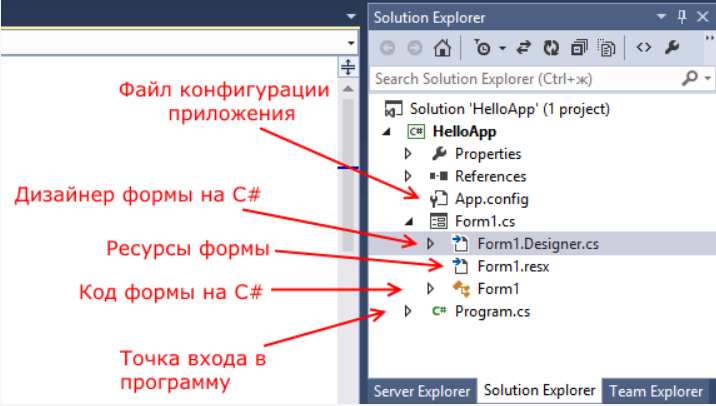
В web- интерфейсе отсутствуют отчёты, их роль выполняют сами страницы.

**Описание интерфейса среды для разработки пользовательского приложения. Обзор компонентов для визуализации данных**

Внешний вид приложения является преимущественно через формы. Формы являются основными строительными блоками. Они предоставляют контейнер для различных элементов управления. А механизм событий позволяет элементам формы отзываться на ввод пользователя, и, таким образом, взаимодействовать с пользователем.

При открытии проекта в Visual Studio в графическом редакторе можем увидеть визуальную часть формы - ту часть, которую мы видим после запуска приложения и куда мы переносим элементы с панели управления. Но на самом деле форма скрывает мощный функционал, состоящий из методов, свойств, событий и прочее. Рассмотрим основные свойства форм.

Если запустим приложение, то отобразится одна пустая форма. Однако даже такой простой проект с пустой формой имеет несколько компонентов:



Несмотря на то, что видим только форму, но стартовой точкой входа в графическое приложение является класс Program, расположенный в файле Program.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HelloApp

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Сначала программой запускается данный класс, затем с помощью выражения Application.Run(new Form1()) он запускает форму Form1. Если вдруг захотим изменить стартовую форму в приложении на какую-нибудь другую, то надо изменить в этом выражении Form1 на соответствующий класс формы.

Сама форма сложна по содержанию. Она делится на ряд компонентов. Так, в структуре проекта есть файл Form1.Designer.cs, который выглядит примерно так:

namespace HelloApp

{

partial class Form1

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

this.SuspendLayout();

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(284, 261);

this.Name = "Form1";

this.Text = "Привет мир!";

this.ResumeLayout(false);

}

#endregion

}

}

Здесь объявляется частичный класс формы Form1, которая имеет два метода: Dispose(), который выполняет роль деструктора объекта, и InitializeComponent(), который устанавливает начальные значения свойств формы.

При добавлении элементов управления, например, кнопок, их описание также добавляется в этот файл.

Но на практике редко будем сталкиваться с этим классом, так как они выполняет в основном дизайнерские функции - установка свойств объектов, установка переменных.

Еще один файл - Form1.resx - хранит ресурсы формы. Как правило, ресурсы используются для создания однообразных форм сразу для нескольких языковых культур.

И более важный файл - Form1.cs, который в структуре проекта называется просто Form1, содержит код или программную логику формы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace HelloApp

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent(); } }}

По умолчанию здесь есть только конструктор формы, в котором просто вызывается метод InitializeComponent(), объявленный в файле дизайнера Form1.Designer.cs. Именно с этим файлом и будем больше работать.

Ниже рассмотрены элементы управления согласно общему функциональному назначению.

|  |  |  |
| --- | --- | --- |
| **Элемент управления** | **Функция** | **Описание** |
| TextBox | Редактирование текста | Отображает текст, введенный во время разработки, который может редактироваться пользователями во время выполнения, а также может быть изменен программными средствами. |
| DataGridView | Вывод данных в таблицу и их редактирование. |  |
| Label | Отображение текста только для чтения | Отображает текст, недоступный для непосредственного редактирования пользователем. |
| StatusStrip | Отображает сведения о текущем состоянии приложения в окне, заключенном в рамку, обычно в нижней части родительской формы. |  |
| ListBox | Выбор из списка | Отображает список текстовых и графических элементов (значков). |
| CheckedListBox | Отображает список с полосой прокрутки, состоящий из элементов с флажками. |  |
| ComboBox | Отображает раскрывающийся список. |  |
| DomainUpDown | Отображает список текстовых элементов, который можно прокручивать с помощью кнопок со стрелками. |  |
| NumericUpDown | Отображает список чисел, который можно прокручивать с помощью кнопок со стрелками. |  |
| ListView | Отображает элементы в одном из четырех представлений: только текст, текст с маленькими значками, текст с большими значками и подробности. |  |
| TreeView | Отображает иерархическую структуру объектов с узлами, которые кроме текста могут включать флажки и значки. |  |
| PictureBox | Вывод и хранение графики | Отображает в рамке графические файлы, например точечные рисунки или значки. |
| ImageList | Компонент. Служит местом хранения изображений. Элементы управления ImageList и хранящиеся в них рисунки могут повторно использоваться в других приложениях. |  |
| CheckBox | Задание значений | Отображает флажок и надпись для текста. В основном используется для задания параметров. |
| RadioButton | Выводит кнопку, которая может быть включена или выключена. |  |
| Trackbar | Позволяет задавать значения на шкале, перемещая по ней ползунок. |  |
| HScrollBar | Горизонтальная линейка прокрутки. |  |
| VScrollBar | Вертикальная линейка прокрутки. |  |
| DateTimePicker | Установка даты | Выводит графический календарь, позволяющий пользователю выбрать дату или время. |