

Методика разработки интерфейса для кроссплатформенных приложений

04, апрель 2015

Вишневская Т. И.^{1,*}

УДК: 004.413

¹Россия, МГТУ им. Н.Э. Баумана

^{*}tv_moscow@mail.ru

1. Введение

Наличие разнообразия операционных систем современных компьютеров и широкое распространение мобильных устройств сделало актуальным разработку кроссплатформенных приложений. Среда Borland Delphi и Turbo Delphi позволяют создавать только Windows приложения. Современная версия Delphi (Embarcadero Delphi XE7) - это среда разработки приложений для Windows, Mac OS X, iOS, Android, гаджетов и носимых устройств [1].

Для создания интерфейса Windows приложений в среде Turbo-Delphi используется объектно-ориентированная библиотека VCL (Visual Component Library). В среде Embarcadero Delphi XE7 основой разработки кроссплатформенных приложений стала библиотека FireMonkey или платформа FM [2]. Умение использовать возможности новой библиотеки FireMonkey и знание технологий программной инженерии в Delphi XE7 необходимы для создания кроссплатформенных приложений.

В работе рассмотрены особенности разработки кроссплатформенных приложений. Предложена и на модельном примере продемонстрирована методика создания кроссплатформенного приложения с использованием библиотеки FireMonkey.

2. Особенности разработки кроссплатформенных приложений

При разработке кроссплатформенных приложений необходимо учитывать следующие особенности:

- создаваемое приложение должно успешно работать на любом устройстве для которого оно предназначено;
- приложение должно максимально использовать возможности используемых операционных систем;

- инструменты кроссплатформенной разработки должны позволять создавать приложения сразу для нескольких платформ (единый код), который компилируется в нативное (родное) приложение для каждой целевой платформы. Нативное приложение предназначено для конкретной ОС и максимально использует ее возможности.

Примерами инструментов кроссплатформенной разработки могут служить среды Embarcadero Delphi XE7 и Microsoft Visual Studio 2015 [3].

3. Выбор компонент для разработки интерфейса нативных приложений

Библиотека FireMonkey из Embarcadero Delphi XE7, включает в себя многочисленный набор компонент и сервисных интерфейсов, написанных на языке Delphi. Возникает проблема выбора тех компонент, которые в кроссплатформенных приложениях отображаются наиболее естественным образом для каждой целевой платформы.

Одним из основных элементов пользовательского интерфейса приложения является меню приложения. В библиотеке VCL для разработки меню используются классы *TMainMenu* (главное меню) и *TPopupMenu* (контекстное меню). В библиотеке FireMonkey представлены три класса меню: *TMainMenu*, *TPopupMenu* и класс *TMenuBar* (планка меню) [2]. Если при создании приложения на платформе FM используется класс *TMainMenu*, то для Windows приложения главное меню размещается сразу под заголовком формы. В OS X поведение главного меню иное – главное меню активного приложения сливается с меню рабочего стола, а название первого пункта меню замещается названием исполняемого файла.

Планка меню (*TMenuBar*) отличается от главного меню (*TMainMenu*), универсальностью и может быть размещена в любом месте формы, может перемещаться и менять размеры. Использование этого класса позволяет создавать меню приложения, которое принимает традиционный вид той операционной системы в которой мы используем данное приложение.

В качестве модельного кроссплатформенного приложения Интеграл рассмотрим приложение для численного интегрирования с использованием одного из возможных методов (Метода прямоугольников, метода трапеций и метода Симпсона) [3]. Интерфейс приложения должен позволять задать значения пределов интегрирования, точность вычислений и делать выбор используемого метода и подинтегральной функции. При создании меню данного модельного приложения будем использовать компонент *TMenuBar*, предназначенный для хранения управляющих элементов, создаваемых на основе класса

TMenuItem и снабженный визуальным редактором *Items Designer*. На Рис.1 представлен пример разработки меню для модельного примера в визуальном редакторе меню Embarcadero Delphi XE7.

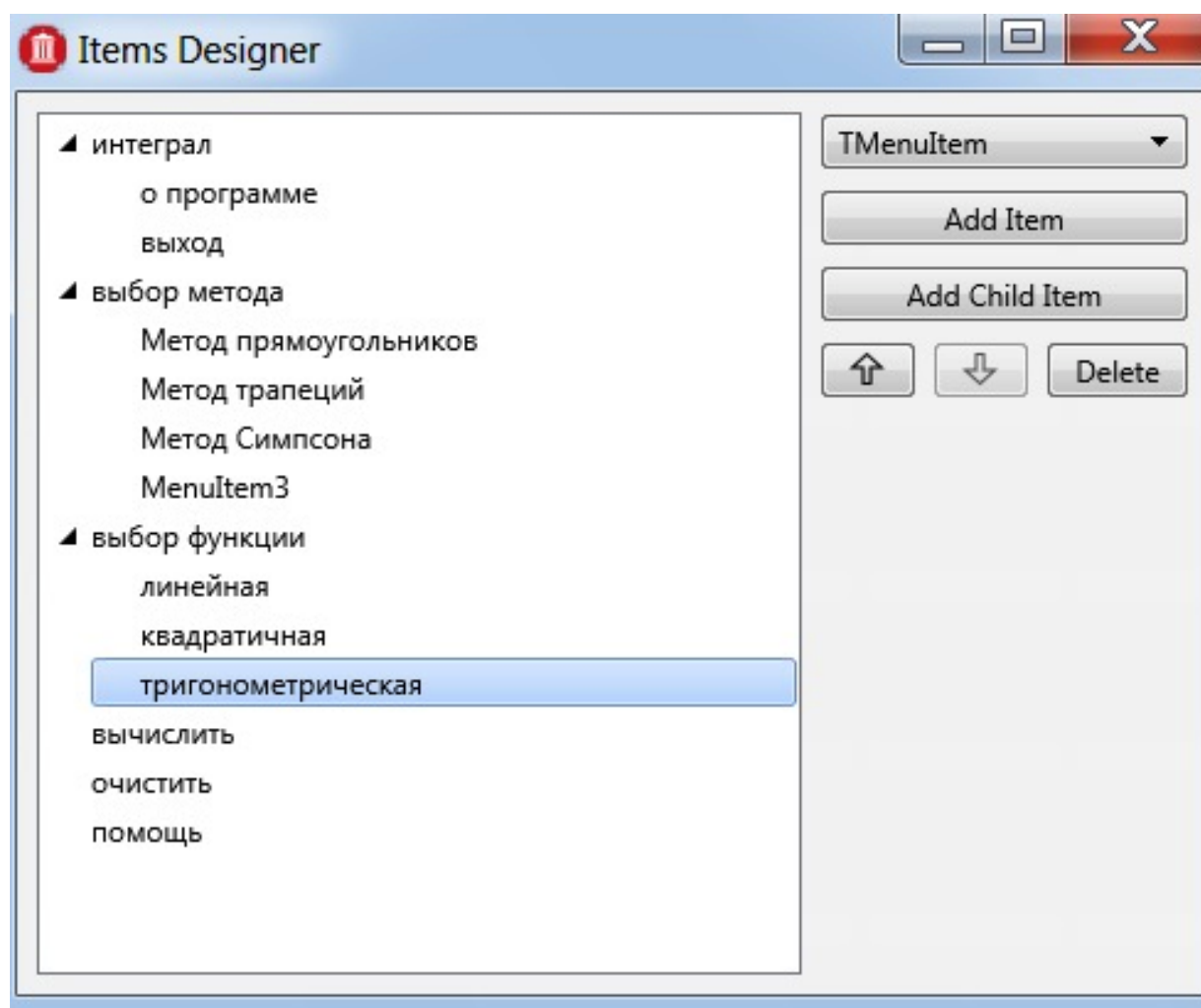


Рис. 1. Визуальный редактор меню

Библиотека FireMonkey предоставляет широкий набор компонент для отображения текстовых данных, которые можно использовать для ввода или вывода информации приложений [1]. В библиотеку вошли аналогичные уже известным из библиотеки VCL классы TEdit, TLabel, TMemo. Но эти классы являются потомками кроссплатформенного класса TControl и могут быть использованы для создания нативного приложения. Новые кроссплатформенные классы TNumberBox, TSpinBox и TComboTrackBar предназначены для ввода числовых данных. Появление этих компонент позволяет обойтись без постоянного перевода строковых данных в числовые и наоборот. Свойство этих компонент ValueType позволяет задать тип вводимых данных (целочисленный или вещественный). В

модельном примере будем использовать компонент TNumberBox для ввода пределов интегрирования и точности вычислений. Для выбора подинтегральной функции и метода интегрирования - комбинированные списки TComboBox. Этот компонент позволяет выбрать только один элемент и мало занимает места на форме. Пример формы при разработке интерфейса модельного приложения представлен на Рис. 2.

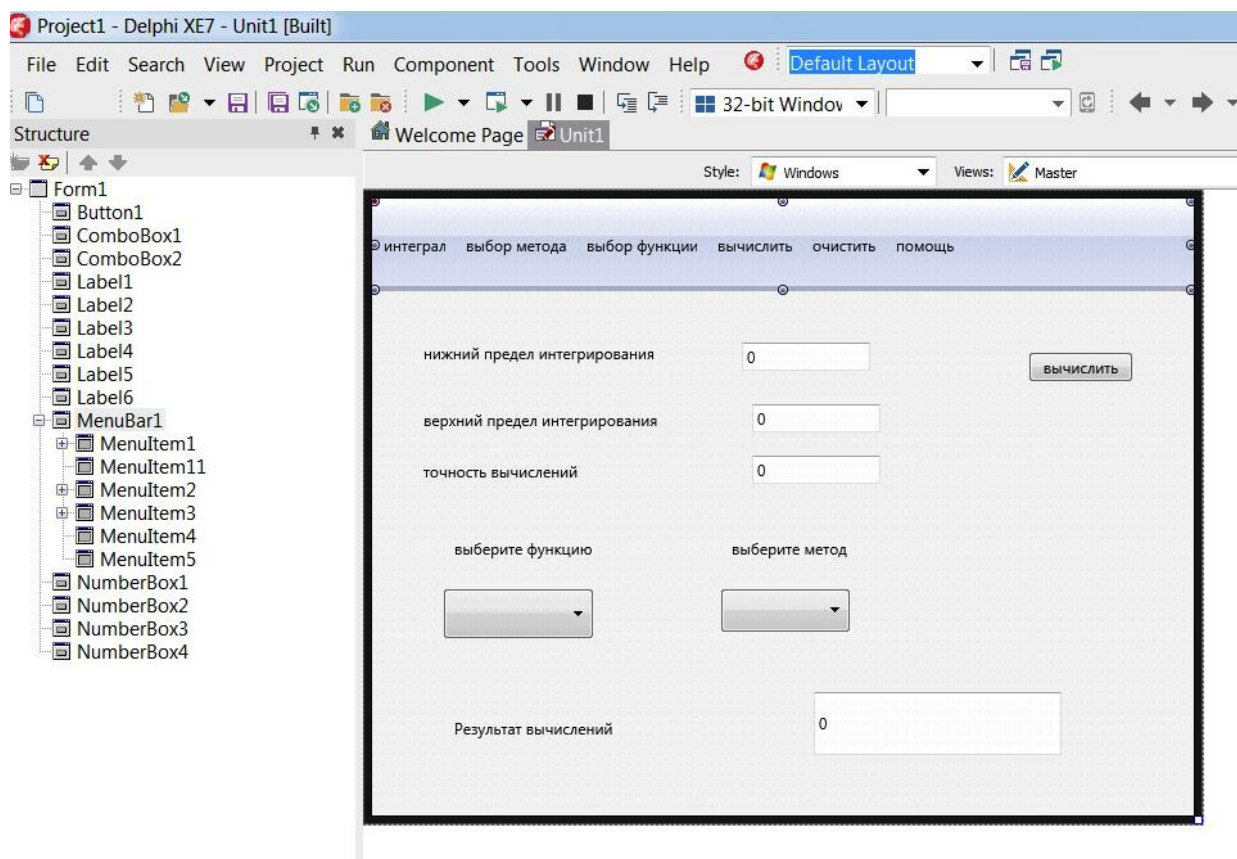


Рис. 2. Визуальная среда разработки Delphi XE7. Форма приложения Интеграл.

4. Методика разработки кроссплатформенных приложений

Рассмотрим методику создания приложений с высококачественным графическим интерфейсом для операционных систем Windows и Mac OS X (Multi-Device Application для Windows и OS X) с использованием среды Embarcadero Delphi XE7.

Начало работы. Установите на компьютер с операционной системой Windows и запустите Embarcadero Delphi XE7.

Выбор типа приложения. Для создания приложения на базе платформы FM воспользуйтесь пунктом меню **File | New | Multi-Device Application** (рис.3). Назовем новый проект **CrossApp.dproj**, а название приложения **Интеграл** (form1.caption).

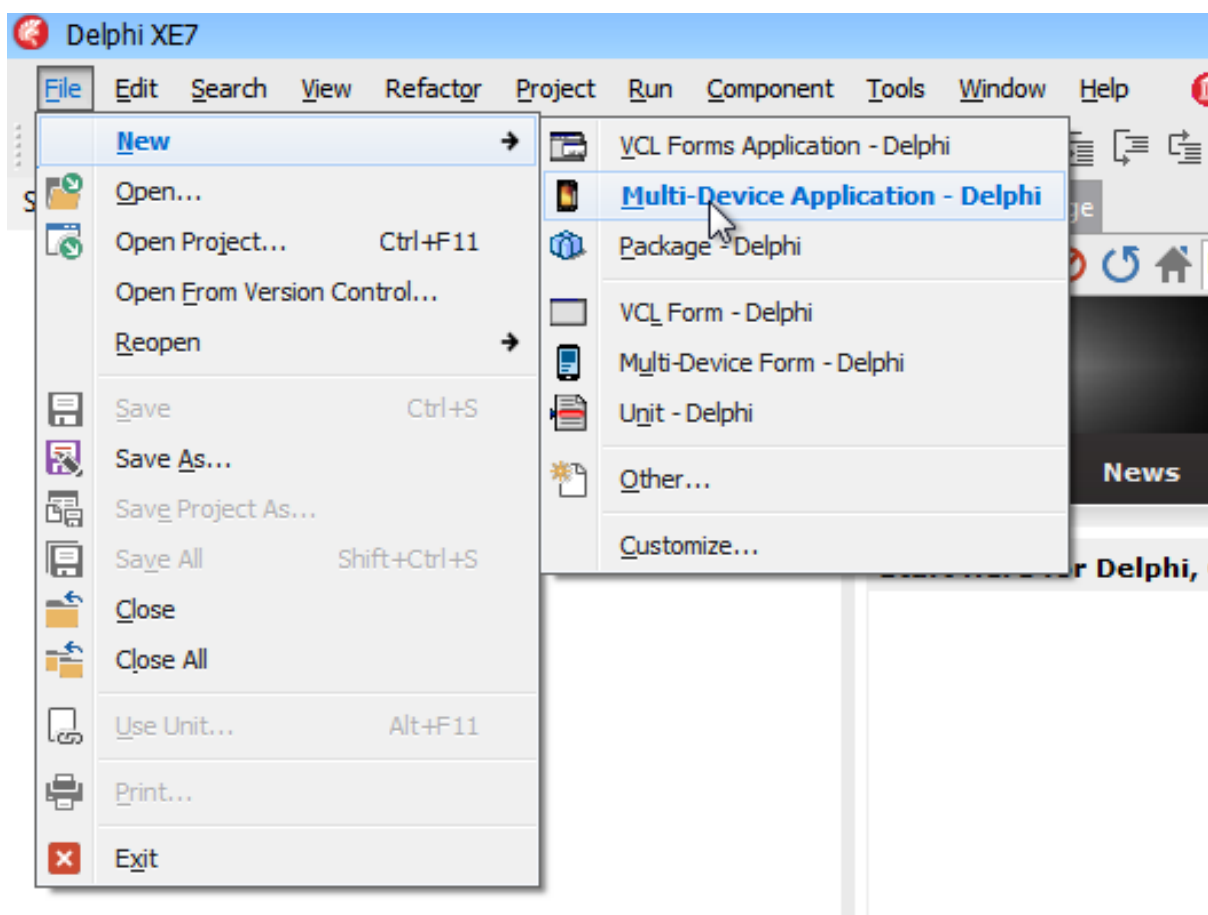


Рис. 3. Выбор типа приложения

Выбор целевой платформы для реализации приложения.

В качестве шаблона приложения в открывшемся окне Multi-Device Application выберите «**Blank Application**». В окне дизайнера отобразится пустая форма, без заголовков и каких либо элементов дизайна. При этом, в верхней части окна дизайнера во вкладке стиль **Style** добавьте целевую платформу (32 разрядная Windows), а режим просмотра **Views** — **Master** (Рис. 2). Данный режим просмотра позволит редактировать интерфейс формы.

Разработка приложения для Windows.

На этом этапе необходимо выполнить разработку, отладку и тестирование Windows приложения.

В п.3 данной статьи предложен модельный пример для разработки нативного приложения и выполнено создание интерфейса с использованием классов библиотеки FireMonkey. На Рис. 4 представлен интерфейс приложения Интеграл на платформе Windows.

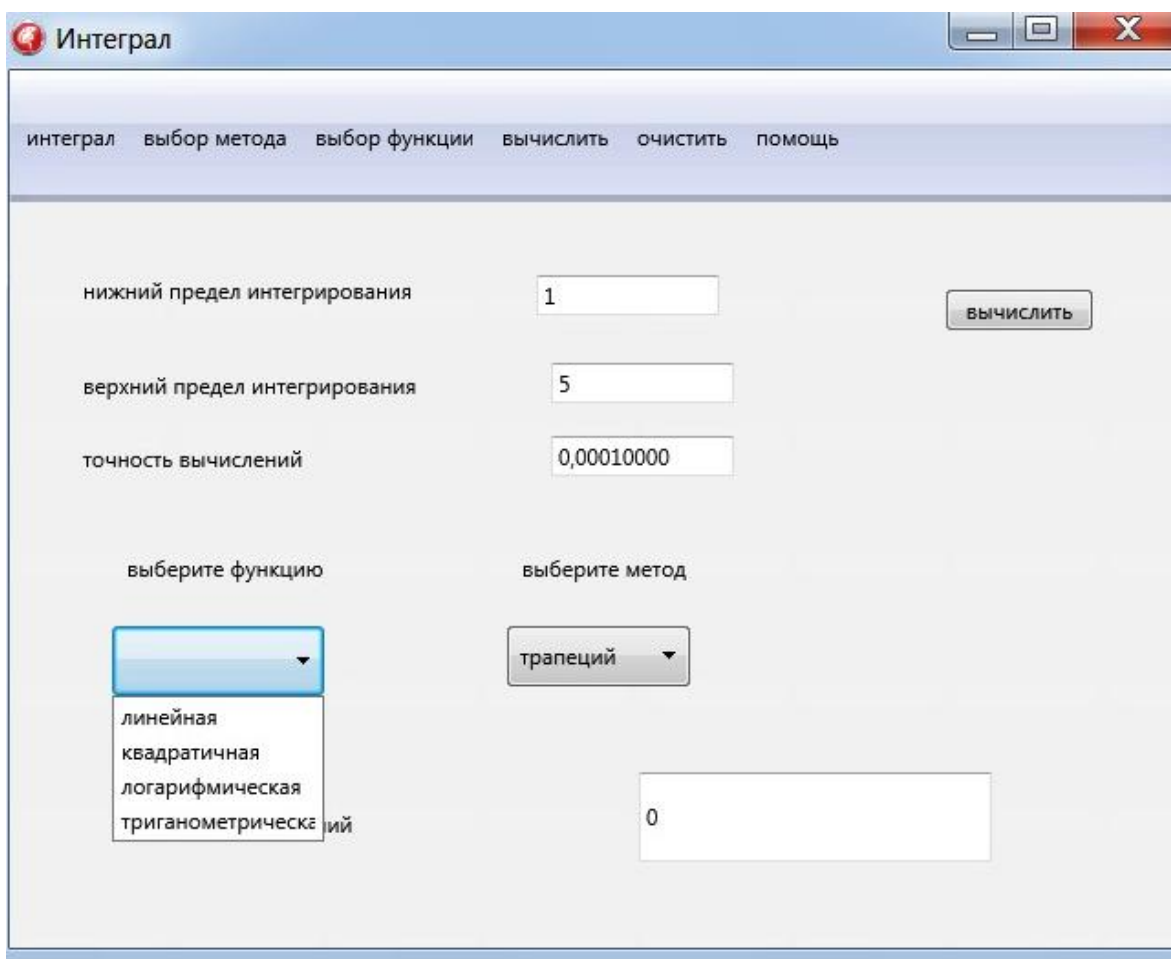


Рис. 4. Интерфейс приложения Интеграл на Windows

Следующий этап создания приложения – написание обработчиков событий для реализации пользовательского интерфейса доступных через компоненты меню (TMenuBar) и стандартные элементы управления (класс TButton). В библиотеку FireMonkey вошли командные классы, которые не только вызывают требуемую процедуру, но и отражают текущее состояние приложения, позволяют управлять последовательностью активных управляющих элементов (команд). Примером такой команды является класс TActionList – невидимый элемент управления. Для заполнения списка команд используется специализированный редактор и для каждой команды свой управляющий код в событиях OnExecute. Для модельного примера возможные команды – это **Выбор подинтегральной функции**, **Выбор численного метода** и **Вычисление**. При этом команда **Вычисление** активна только при выполнении первых двух, для контроля текущего состояния приложения используется метод OnUpdate.

Выбор дополнительной платформы кроссплатформенного приложения

Во вкладке просмотр **Views** выберите режим отображения OS X Desktop и вы увидите интерфейс Вашего приложения в MAC OS X (Рис. 5), но при этом все компоненты остались. При необходимости можно продолжить редактирование интерфейса формы.

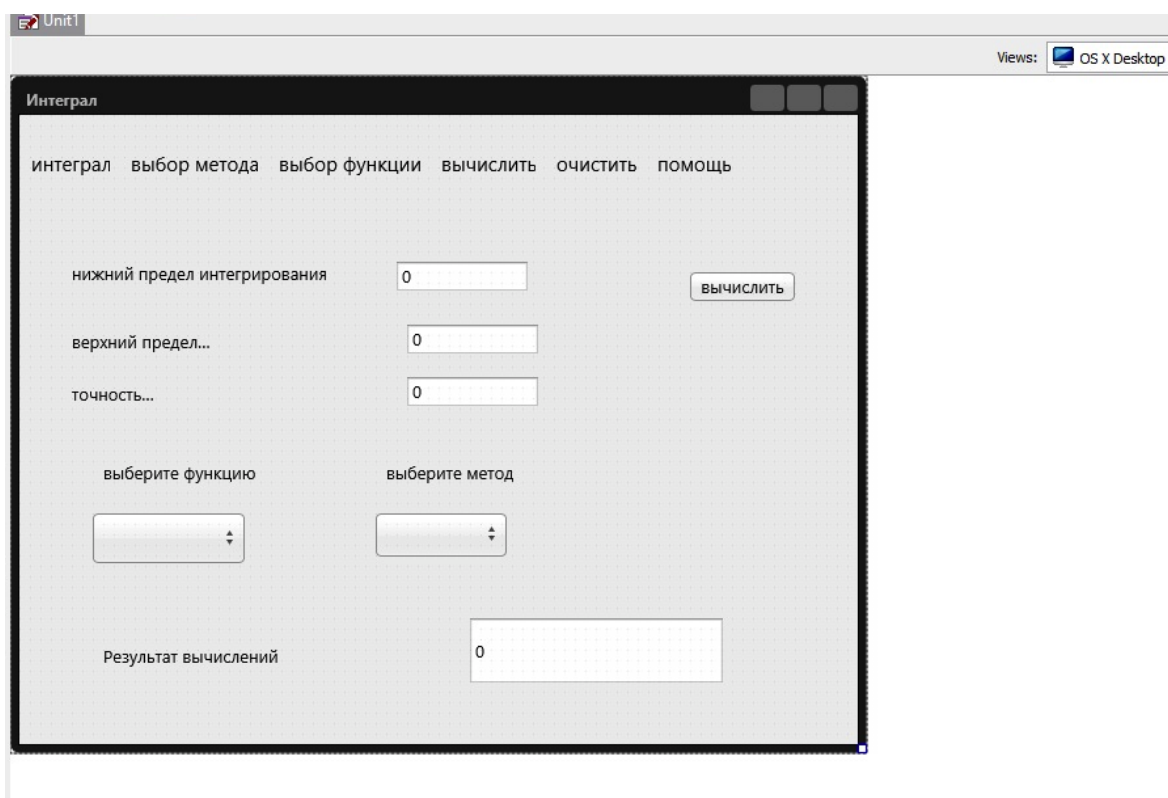


Рис. 5. Интерфейс приложения Интеграл для платформы OS X

Во вкладке стиль **Style** выберите целевую платформу (OS X) и вы не только увидите интерфейс Вашего приложения в MAC OS X, но и Delphi XE7 создаст для этого режима отдельный рас-файл. В результате получаем новую форму *TForm1_CrossApp* и, соответственно новый файл *Unit1CrossApp.pas*. В проект новая форма включаются как ресурсы:

```
implementation {$R *.Macintosh.fmx _MACOS}
```

При этом, весь исходный код (единый код) находится в одном файле — Unit1.pas.

Создание релиза для OS X.

1. Выпуск приложения для OS X потребует второй компьютер (либо виртуальную машину) с OS X, к которому мы можем получить доступ с Windows компьютера.
 - Соедините эти два компьютера в сеть с использованием технологии Wi-Fi. Для этого необходимо, чтобы оба компьютера находились в одной сети Wi-Fi.
 - Отключите брандмауэр на Mac, чтобы входящие подключения к компьютеру были разрешены.
 - Выполните настройку общего доступа на Mac OS X, для этого в окне **Общий доступ (Системные настройки)** выберите в списке **Общий доступ к файлам** и выберите вкладку **Параметры** (Рис. 6).

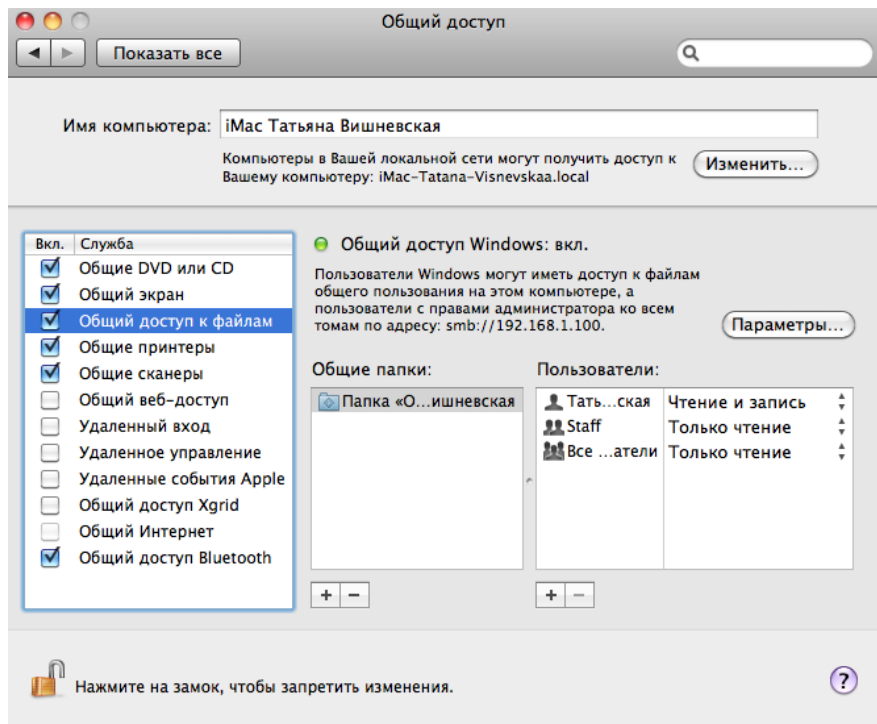


Рис. 6. Окно настройки общего доступа в OS X.

- В появившемся меню поставьте галочку возле **Предоставление общего доступа к файлам и папкам с помощью SMB (Windows)** и **Удаленный вход**. И выберите учетную запись, которая и будет разрешать связь между компьютерами (потребуется ввести логин и пароль учетной записи Mac OS) (Рис. 7).

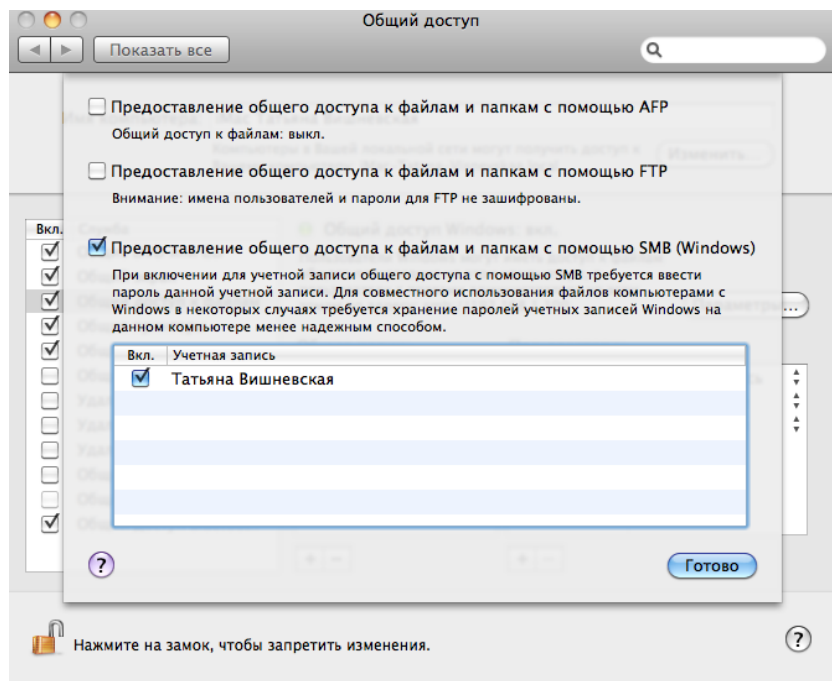


Рис. 7. Окно настройки общего доступа в OS X с помощью Windows.

- Чтобы увидеть сеть между двумя компьютерами, перезагрузите и откройте на Windows окно проводника и в списке слева выберите **Сеть** и увидите компьютеры рабочей группы (Рис. 8).

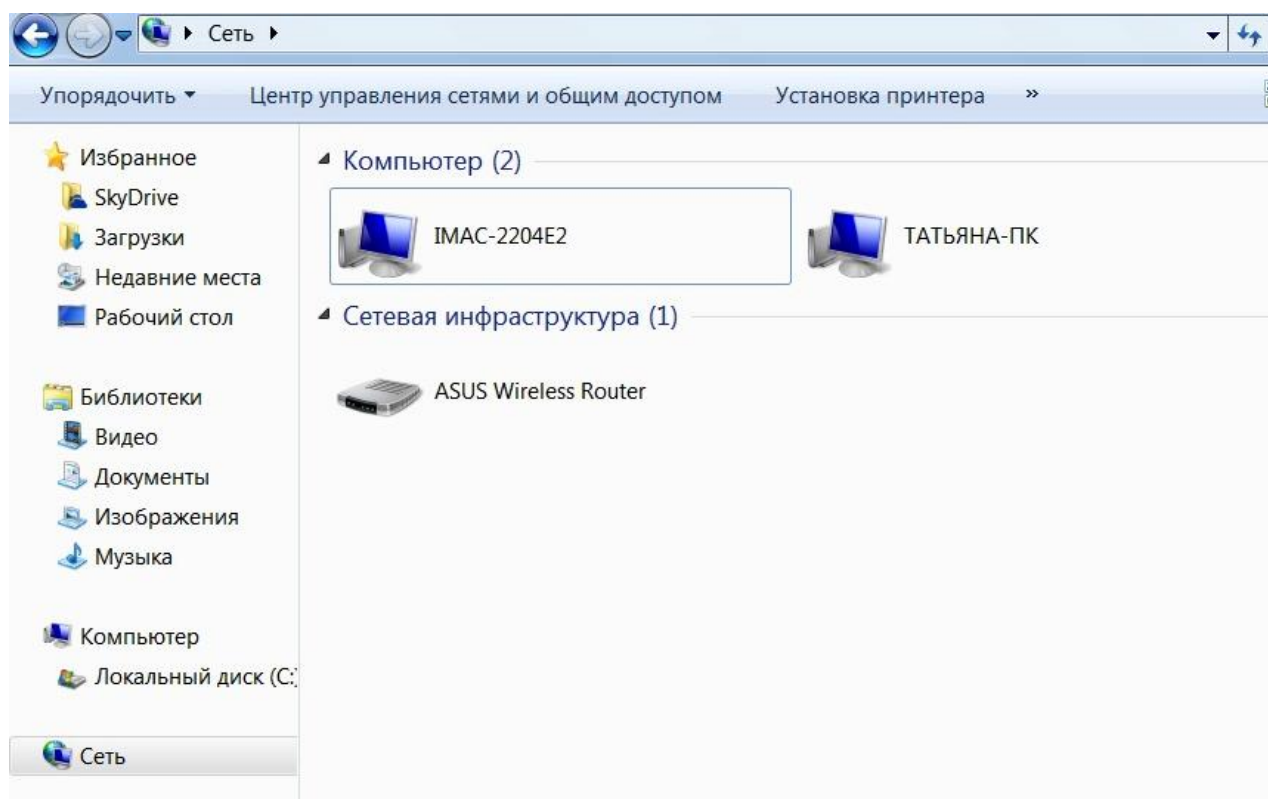


Рис. 8. Окно проводника Windows 7

2. В окне создания приложения (Рис. 2) выберите целевую платформу, на которую будем транслировать приложение, в нашем случае это OS X (была платформа 32 разрядная Windows). Приложение (в официальной документации именуемое **Platform Assistant**), которое выполнит компиляцию приложения FireMonkey на платформу OS X входит в пакет Embarcadero Delphi XE7 и называется **PAServer** [2]. Установочный файл этого приложения PAServer n.n.pkg, где n.n – версия Embarcadero, находится в директории: C:\Program Files\Embarcadero\Studio\n.n\PAServer
3. Установите приложение PAServer на компьютере с OS X, используя установочный файл PAServer n.n.pkg. После установки в окне Программы найдите приложение PAServer и запустите его. В окне диалога программы нажмите Enter (нет пароля), а затем i, чтобы узнать IP адрес компьютера с OS X.
4. Перейдите в окно разработки приложения на компьютер с Windows. Выполните созданное приложение Интеграл, используя команду Run. При попытке запустить приложение под OS X откроется окно создания профиля.

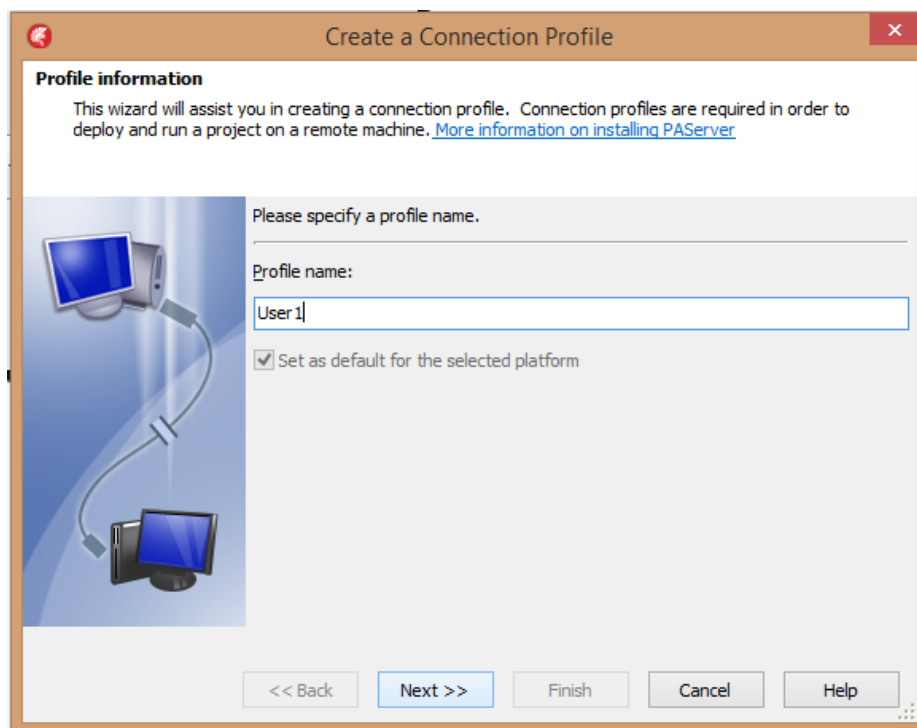


Рис. 9. Окно создания профиля

Введите новое имя профиля для Delphi XE7 (и нажмите далее, после чего будет нужно ввести IP адрес компьютера Mac (порт и пароль изменять не нужно). Если всё введено верно, то нажмите на Finish (Рис. 10).

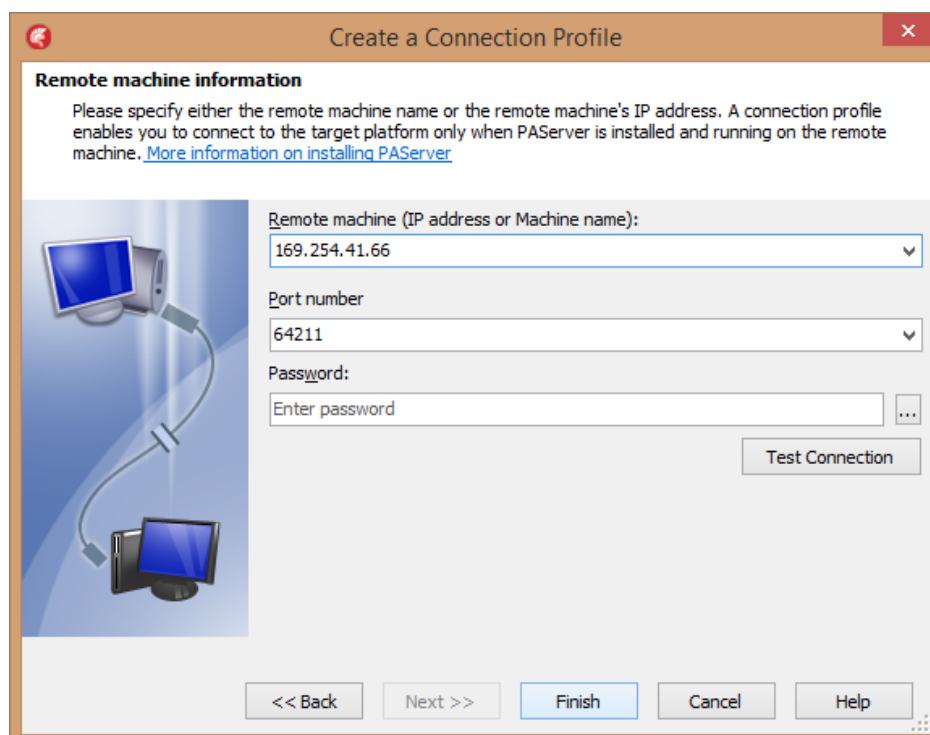


Рис. 10. Мастер создания удаленного профиля

При необходимости добавить или отредактировать профиль перейдите в Tools -> Options -> Connection Profile Manager.

После настройки профиля повторим команду Run - приложение скомпилируется и запустится на выбранном компьютере с OS X.

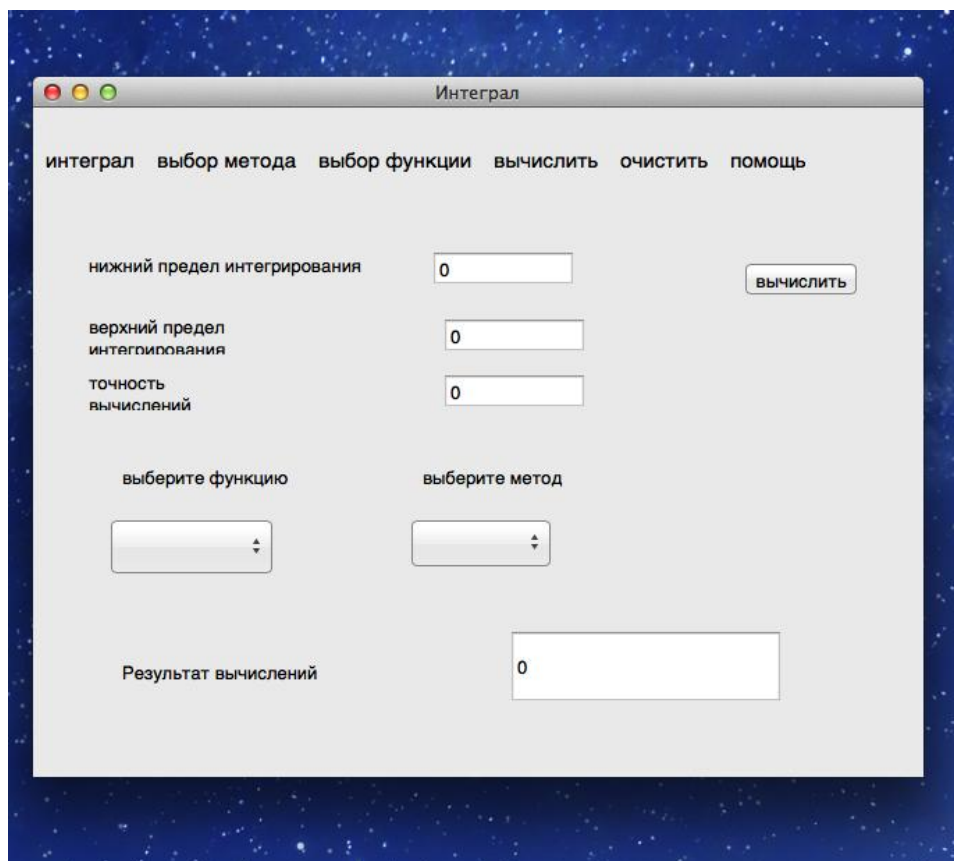


Рис.11. Интерфейс приложения Интеграл на Mac OS X

Исполнительный файл приложения для его дальнейшего использования как отдельной программы для MAC OS X находится в директории:

Macintosh/Users/UserOsXName/PAServer/scrath-dir/UserWinName-ProfileName,

где UserOsXName – имя пользователя в OS, XUserWinName – имя пользователя в Windows, ProfileName – имя профиля, созданного в Delphi XE7.

Заключение

В данной статье сформулированы основные этапы разработки интерфейса приложений для Windows и Mac OS X с использованием среды Embarcadero Delphi XE7 и приведен пример их реализации для модельного приложения. Продемонстрированы универсальные возможности библиотеки FireMonkey для создания нативных приложений, но необходи-

мо помнить, что и эта библиотека не может заменить все функции API целевой операционной системы.

Данный материал позволит студентам начать самостоятельную разработку кросс-платформенных приложений в том числе и для реализации распределенных систем [4] .

Список литературы

1. Delphi XE8. // Embarcadero. Режим доступа: <http://www.embarcadero.com/ru/products/delphi/> (дата обращения 12.03.2015.)
2. Осипов Д.Л. Delphi. Программирование для Windows, OS X, iOS и Android. / Серия: Профессиональное программирование. СПб.: БХВ-Петербург. 2014. 464 с.
3. Visual Studio 2015. // Visual Studio. Режим доступа: <http://www.visualstudio.com> (дата обращения: 12.03.2015).
4. Алексеев Ю.Е., Ваулин А.С., Куров А.В. Практикум по программированию. Обработка числовых данных. М.: Изд-во МГТУ им. Н.Э. Баумана. 2008. 288 с.
5. Вишневская Т.И. Романова Т.Н. Технология программирования: методические указания к лабораторному практикуму по дисциплинам "Технология программирования" и "Методология программной инженерии". Ч. 3. / Электронное учебное издание. ФГБОУ ВПО МГТУ им. Н.Э. Баумана. 2012. // ФГУП НТЦ "Информрегистр". Режим доступа: <http://catalog.inforeg.ru/Inet/GetEzineByID/293408> (дата обращения: 20.03.2015).