ИНТЕГРИРОВАННАЯ СРЕДА РАЗРАБОТКИ LAZARUS, КАК СВОБОДНО РАСПРОСТРАНЯЕМЫЙ ИНСТРУМЕНТ ДЛЯ СОЗДАНИЯ ПЕДАГОГИЧЕСКИХ ПРОГРАММНЫХ СРЕДСТВ С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ

Костюченко Андрей Александрович,
ассистент кафедры информатики и вычислительной техники Черниговского национального педагогического университета им. Т. Г. Шевченко, Украина.

Почти одновременно с появлением в школе компьютеров начали создаваться компьютерные программы, предназначенные для обучения школьников программированию. Затем появились программы для поддержки обучения другим предметам. За такими программами и инструментальными средствами закрепился термин «педагогические программные средства». Педагогические программные средства (ППС) – это совокупность компьютерных программ, предназначенных для компьютерной поддержки обучения и достижения конкретных учебных целей [2, с. 5].

Понятно, что на современном этапе программные средства и в частности педагогические программные средства, которые ориентированы на работу с пользователем, должны содержать графический интерфейс GUI (Graphics User Interface). Поэтому возникает задача выбора среды для создания таких педагогических программных средств.

Различные аспекты выбора и использования среды программирования исследуют ученые: Г.В. Белова, Н.В. Макарова, М.И. Жалдак, Р.И. Заболотный, И.А. Завадський, Т.П. Караванова, В.Д. Руденко, Ю.В. Горошко, З.С. Сейдаметова, С.О. Семериков, А.В. Спиваковський, О.М. Спирин, Ю.В. Триус. Так Спиваковський А.В. в своей работе [5] касается вопроса выбора среды программирования, которое тесно связано с языком программирования, говорит «В процессе рассмотрения сред программирования будем учитывать средства разработки Windows-приложений, которые имеют значительное распространение в учебных заведениях. На современном этапе наиболее популярны Delphi, Visual Basic, Visual C ++».

Проанализировав современные среды создания приложений с графическим интерфейсом наиболее популярными можно назвать: Lazarus, Delphi, Visual Basic, Microsoft Visual Studio, Qt Creator, Eclipse, SharpDevelop. Данные среды разработки - это продукты одного класса, снабжены в целом достаточным набором средств и компонент для создания развитых программ с графическим интерфейсом.

На мой взгляд, основываясь на использования языка Pascal при обучении алгоритмизации и программирования в школах Украины и России, свободной распространяемости и бесплатности среды разработки, что немаловажно для государственных учебных заведений, как среду разработки для создания педагогических программных средств можно выбрать Lazarus, который использует язык программирования Object Pascal.

Интегрированная среда разработки (Integrated Development Environment) IDE Lazarus – это свободная среда разработки программного обеспечения для компилятора Free Pascal на языке Object Pascal и распространяется под свободными лицензиями GNU General Public License (GNU GPL) и GNU Lesser General Public License (GNU LGPL). Она представляет собой среду с графическим интерфейсом для разработки программ, похожая на Delphi, и базируется на кроссплатформенной библиотеке визуальных и не визуальных компонентов LCL (Lazarus Component Library), совместимых с VCL Delphi. Такого набора компонентов Lazarus достаточно для создания приложений с графическим интерфейсом, приложений работающих с базами данных и Интернетом. В настоящее время Lazarus работает под управлением операционных систем (ОС) Linux, Mac OS X, UNIX-подобных, Windows, Android, в результате чего приложения, созданные под одну ОС, без труда могут быть перекомпилированы под другую ОС. То есть, используя среду Lazarus, можно создавать кроссплатформенные приложения. Кроме того среду Lazarus имеет довольно много локализаций, в том числе русскую и украинскую.

IDE Lazarus состоит из нескольких, физически не связанных между собой, «плавающих» окон (рис. 1).

Главное окно – выполняет основные функции управления проектом создаваемой программы. Окно разбито на три функциональные блоки: *главное меню* – в котором размещены команды управления средой и компиляцией, *панель инструментов* – предоставляет быстрый доступ к основным команд главного меню, *палитра компонентов* – предоставляет доступ к основным компонентам среды разработки.

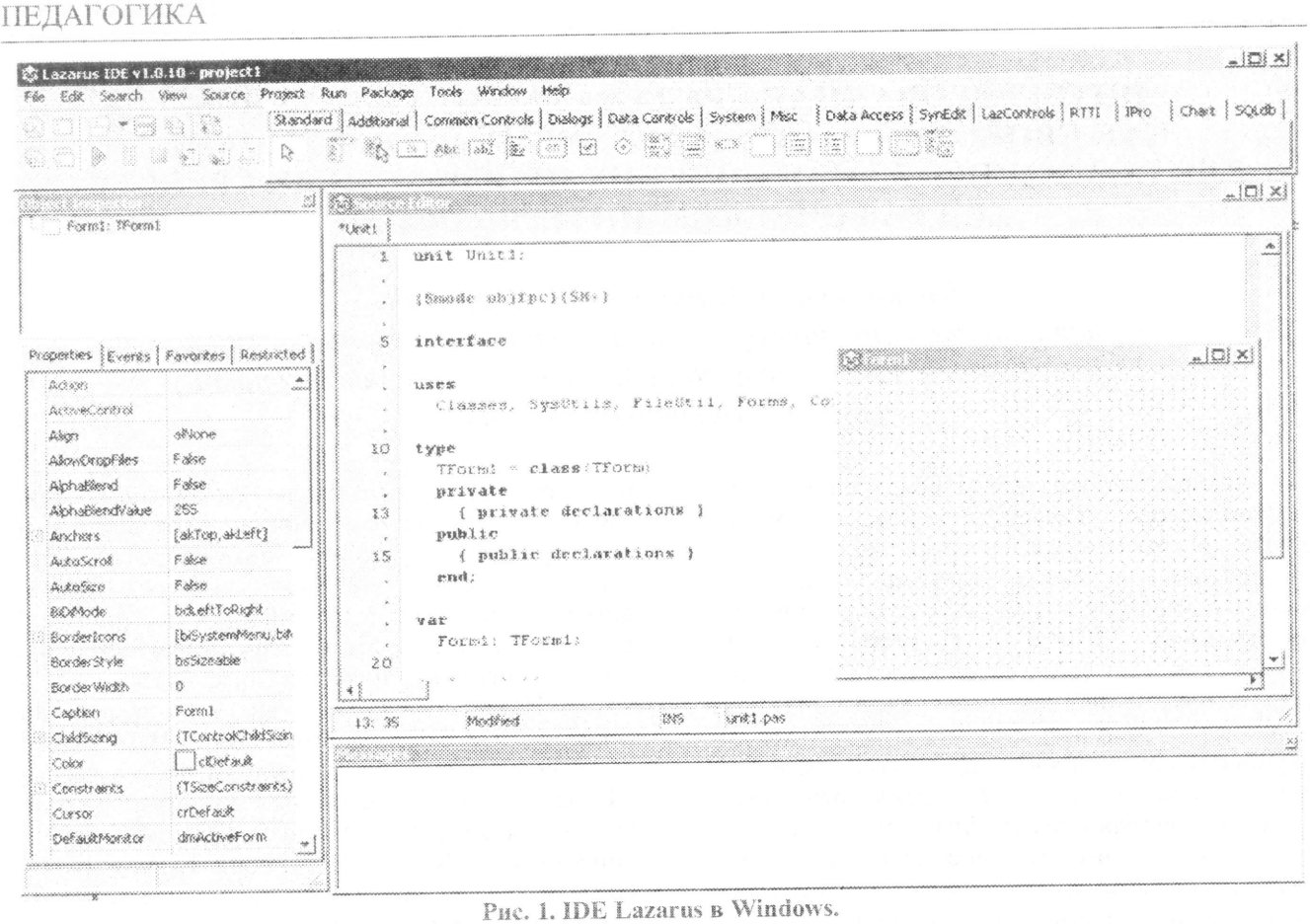

Рис. 1. IDE Lazarus в Windows.

Окно инспектора объектов (Object Inspector) – содержит параметры активного компонента размещенного на форме, каторые программист может изменить в процессе разработки программы. В верхней части находится область дерева объектов, в которой отображаются компоненты, размещенные на форме в виде дерева, то есть отражается принадлежность компонентов своим контейнерам. Основную часть окна инспектора объектов занимают 4 закладки: *Properties (Свойства)* – содержит свойства активного компонента, *Events (События)* – содержит события, которые может обработать активный компонент, *Favorites (Избранное)* – содержит избранные свойства и события, *Restricted (Ограничение)* – содержит ограничения свойств и событий активного компонента относительно используемой операционной системы.

Окно редактора кода программы (Source Editor) – предназначено для ввода и редактирования текста программы. Многие функции данного редактора совпадают с возможностями обычных текстовых редакторов: выделение, копирование, вставка, выполнение поиска и замена фрагмента текста. Однако данный редактор имеет ряд дополнительных возможностей для облегчения разработки программы. Редактор позволяет подсвечивать синтаксис, и не только языка Pascal. Фрагменты текста можно сдвинуть вправо или влево с целью структурирования кода. Кроме того, данное окно имеет возможности, которые смогут понадобиться при написании программного кода. *Поиск в редакторе*, если поместить курсор мыши над любым идентификатором, то в контекстной подсказке, которая появиться, будет указано, где этот идентификатор был объявлен. Также есть другое использование этого поиска, удерживая кнопку Ctrl и нажав левую кнопку мыши на подключенном модуле, компоненте, ее свойстве или методе произойдет переход в соответствующий модуль или на соответствующий класс компонента. Также, если стать на объявление некоторого метода и нажать комбинацию кнопок Ctrl + Shift + стрелка вниз, то произойдет переход на определение данного метода, а для перехода из определения метода к месту его объявления используется комбинация клавиш Ctrl + Shift+ стрелка вверх. *Завершение идентификаторов или завершения кода* позволяет выбрать объект, свойство или метод объекта из списка по введенным первым буквам. Для активации данного списка с перечнем возможных значений, необходимо нажать комбинацию кнопок Ctrl + Space, если введенное количество символов будет отвечать только единственному возможному варианту, то он автоматически подставлено в редактор кода. *Подсказка параметров* позволяет показать список необходимых параметров подпрограммы (процедуры или функции) в всплывающем окне. Вызвать эту подсказку можно нажав комбинацию кнопок Ctrl + Shift +

Space в области описания фактических параметров подпрограммы. В окне подсказки будет содержаться описание соответствующей подпрограммы с ее формальными параметрами, причем формальный параметр, для которого в данный момент должно происходить введение фактического параметра, будет подсвечен жирным шрифтом. *Завершение классов* используется для автоматического генерирования завершения кода, оно активизируется нажатием комбинации кнопок Ctrl + Shift + C. То есть, если вы объявите некоторое метод в классе и нажмете указанную комбинацию кнопок, то в разделе реализации модуля будет сгенерирован каркас соответствующего метода.

Окно формы – проект окна будущей программы. На данной форме происходит проектирование внешнего вида программы путем размещения на ней элементов управления (компонентов).

Окно сообщений (Messages) – в нем выводятся сообщения компилятора.

При создании программных средств с графическим интерфейсом работа программиста делится на две взаимосвязанные задачи: работа с созданием интерфейса и работа с программным кодом.

Создание интерфейса пользователя заключается в том, что из набора компонент библиотеки формируется дизайн вашей программы. В программах с графическим интерфейсом управления окнами более-менее стандартизировано даже для различных ОС и платформ. Почти все окна имеют одни и те же элементы, причем размещены в большинстве случаев, в одних и тех же местах. Понятно, что не все элементы не являются обязательными даже для окна главной программы, не говоря уже о второстепенных или диалоговых окнах, так как это зависит от необходимости управления программой. Конечно, можно и не соблюдать таких стандартов и создать собственный, инновационный интерфейс, но необходимо помнить, что созданный интерфейс может показаться неудобным, «недружественным» пользователю с привычками работы в стандартном интерфейсе, и в результате вызвать нежелание работать с таким программным средством.

Работа с программным кодом программы с графическим интерфейсом базируется на необходимости программы реагировать на события. Событие может вызвать действие пользователя (например, нажатия кнопки в приложении), операционная система, или событие может создать само приложение. Все события отслеживает ОС, формирует на каждый тип события соответствующее сообщение. Это сообщение передается в приложение, которое реагирует на него в соответствии с алгоритмом своей работы. То есть GUI-программы должны уметь обрабатывать сообщения операционной системы и уметь генерировать и отправлять собственные сообщения.

Приложение с графическим интерфейсом после запуска создает собственное окно и запускает так называемый цикл сообщений, т.е. ждет сообщений от операционной системы. Роль программиста заключается в разработке кода по обработке сообщений при возникновении событий. Сообщение передается операционной системой именно тому приложения, которому оно предназначалось. Таким образом, реализуется мультипрограммность операционной системы, то есть возможность одновременно запускать и работать нескольким программам.

В Lazarus обработка сообщений заменена на обработку событий. Программисту достаточно выбрать те события, на которые должна реагировать его приложение, и написать процедуры обработки соответствующих событий [1]. Понятно, что не обязательно описывать обработку всех возможных событий. В случае отсутствия описания некоторого события, она просто не будет обработана вашим приложением.

Таким образом, можно заметить, что для создания программных средств и педагогических программных средств с графическим интерфейсом может быть использована как удобная и свободная, среда разработки Lazarus.

Литература

1. Алексеев Е.Р., Чеснокова О.В., Кучер Т.В. Free Pascal и Lazarus: Учебник по программированию. – М.: ALT Linux; Издательский дом ДМК-прес, 2010. – 440 с.

2. Волинський В.П., Козлакова Г.О. Методичні рекомендації до використання педагогічних програмних засобів у навчальному процесі. – К.: НПУ ім. М. П. Драгоманова, 2007. – 59 с

3. Горошко Ю.В., Костюченко А.О. Теорія і методика розробки педагогічних програмних засобів. – Чернігів: Виготовлення Єрмоленко О.М., 2011. – 144 с.

4. Мансуров К.Т. Основы программирования в среде Lazarus, 2010. – 772 с. – [Электронный ресурс]. – Режим доступа: http://www.freepascal.ru/download/pdf/osnovy_programmirovanija_v_srede_lazarus.pdf.

5. Співаковський О.В., Гуржій А.М., Львов М.С. Основи програмування: Навчальний посібник. – К.: Наукова думка, 2004. – 355 с.

Поступила в редакцию 20.06.2013 г.