УДК 372.862

# ПОЭТАПНОЕ УСЛОЖНЕНИЕ И РАЗРАБОТКА МОДЕЛИ УПРАВЛЕНИЯ ГРАФИЧЕСКИМИ ОБЪЕКТАМИ НА ОСНОВЕ КЛАССОВ

**Бочкарева Е.А.**
**научный руководитель канд. физ.-мат. наук Дмитриев В.Л.**
***Стерлитамакский филиал Башкирского государственного университета***
***Российская Федерация, г. Стерлитамак***

Около сорока лет назад большая часть программирования по своей природе была процедурной. Элементами программирования были процедура, функция, алгоритм. Описания данных зачастую были разбросаны по коду программы и не абстрагированы. В связи с этим было необходимо очень аккуратно документировать описываемые структуры данных, например, в комментариях.

В объектно-ориентированном программировании базовым понятием выступает понятие объекта, который имеет определенные свойства. Состояние объекта задается перечислением значений его признаков (полей). Кроме того, обычно объект располагает набором методов, призванных решать разнообразные задачи. При написании программы, содержащей объекты, можно организовать взаимодействие объектов между собой.

Главными достоинствами объектно-ориентированного программирования являются: во-первых, использование абстрагирования и инкапсуляции структуры; во-вторых, то, что объекты представляют собой хорошую модель для большинства сущностей реального мира. Вместе с тем иногда становится сложно понять и представить разрабатываемый алгоритм в целом, поскольку его фрагменты могут быть распределены по многим методам, ассоциированным с различными типами объектов. Поэтому в объектно-ориентированном программировании при написании больших программ в большинстве практически важных случаев необходимо крайне аккуратно документировать методы.

Для использования объектов необходимо объявить объектовый тип, который в объектно-ориентированном программировании чаще называют классом. Методы, определенные в объектовом типе, имеют доступ ко всем полям этого типа.

Событийно-ориентированное программирование – это парадигма программирования, в которой выполнение программы определяется событиями – действиями пользователя (сигналы с клавиатуры и мыши), сообщениями других программ и потоков (при организации взаимодействия между отдельными приложениями), событиями операционной системы. Как правило, событийно-ориентированное программирование применяется в трёх случаях:

- при построении пользовательских интерфейсов (в том числе графических);
- при создании серверных приложений в случае, если по тем или иным причинам нежелательно порождение обслуживающих процессов;
- при программировании игр, в которых осуществляется управление множеством объектов.

В современных визуальных языках программирования события и обработчики событий являются центральным звеном реализации графического интерфейса пользователя. Рассмотрим, например, принцип взаимодействия программы с событиями от мыши. Нажатие какой-либо клавиши мыши вызывает соответствующее системное прерывание, запускающее определенную процедуру операционной системы, в которой выполняется поиск окна, расположенного под курсором мыши. Если такое окно найдено, событие посылается в очередь обработки сообщений этого окна. Далее, в зависимости от типа окна, могут генерироваться дополнительные события.

В приведенной работе выполняется создание упрощенного графического интерфейса с элементами модели управления (реализовано движение объектов) в среде визуального программирования Delphi. Выделение нужного объекта для его передвижения основано на получении координат щелчка мыши, а затем поиске объекта с нужными координатами.

С целью упрощения логики рассуждений при реализации поставленной задачи, она была разбита на две основные части:

- рисование графических примитивов (в нашем случае был выбран объект "Прямоугольник"),
- организация движения и удаления созданных объектов (все они одного типа – "Прямоугольник").

Выбор Delphi обоснован тем что данная среда разработки предоставляет многообразные возможности во всех областях программирования прикладного программного обеспечения, включая данную. На этой мощной основе программист может создать многофункциональный программный продукт с большим спектром возможностей. Для реализации программы была выбрана платформа Microsoft Windows XP. Этот выбор обусловлен наличием данной операционной системы на ЭВМ в аудиториях университета, что позволит наглядно продемонстрировать полученный программный продукт для студентов, изучающих основы программирования.

Рассмотрим основные этапы разработки. Реализуем класс "Прямоугольник", методами которого являются процедура рисования прямоугольника, организация его движения в указанную точку, возможность создания прямоугольников различного размера, удаление выбранного прямоугольника.

Поставленную задачу разобьем на 4 части, которые логически связаны между собой, причем с каждым последующим шагом происходит усложнение работы программы, появляются новые процедуры, реализующие методы класса прямоугольник (здесь мы используем принцип, рассмотренный в работах [1, 2]). Работа с отдельными объектами-прямоугольниками будет происходить на компоненте Image.

На первом этапе работы создаем класс "Прямоугольник", основными свойствами которого являются цвет границы, цвет заливки, координаты и размеры (размеры можно задать на этом этапе разработки фиксированными). Обработчик на отпускание мыши напишем таким образом, чтобы он позволял создавать отдельные прямоугольники в любом месте компонента Image.

На следующем этапе добавим процедуру движения, которая основана на прибавлении к координатам объекта некоторого значения, рассчитываемого на основе коэффициентов передвижения по осям координат. Объект останавливается после того, как достигает пункта назначения, данный метод реализуется путем введения таймера как одного из полей разрабатываемого класса. На данном этапе придется столкнуться со следующей проблемой: при передвижении двух объектов одновременно процедура Paint осуществляет рисование, которое создает визуальный эффект взаимодействия двух прямоугольников путем наложения одного на другой (стирание одного объекта другим). Как только один из взаимодействующих объектов останавливается, а второй продолжает двигаться по нему, происходит стирание остановившегося объекта. Данная проблема была устранена на четвертом этапе путем написания соответствующей функции, которая проверяет совпадение двух объектов-прямоугольников.

На текущем этапе удобно ввести массив для хранения отдельных прямоугольников, тем самым каждый создаваемый объект будет элементом данного массива. Да данном этапе можно реализовать возможность рисования прямоугольников различного размера (задается пользователем как и в любом графическом редакторе – путем нажатия и перемещения мыши). Однако при таком способе хранения объектов скоро становится по-

нятным, что использование массива оказывается нерациональным при организации удаления выбранного прямоугольника (приходится сдвигать все элементы данного массива, тем самым перезаписывая отдельные объекты).

В связи с отмеченной выше проблемой использования массива, вводим список на основе TList. (точнее, организуем класс для работы со списком объектов-прямоугольников). Теперь каждый прямоугольник будет являться элементом списка.

Данный класс будет обладать своими полями и методами. Рассмотрим некоторые из них. Свойство Count – хранит число элементов (указателей) в списке. Свойство Items – позволяет обращаться к элементам в списке (например, обращение TRectangle(List.Items[2]) возвращает 3-ий элемент в списке). Метод Add – добавляет элемент в конец списка. Метод Delete – удаляет элемент из списка по его позиции в списке. Таким образом, получаем удобное и достаточно оптимальное решение поставленной задачи работы с прямоугольниками.

В результате мы получим упрощенный графический интерфейс, который позволяет осуществлять создание и удаление графических изображений, а также производить их перемещение. Теперь можно посмотреть результат работы программы, и убедиться, что объекты будут перемещаться так, как мы и хотели.

Написанная программа имеет прикладной характер и может использоваться для самостоятельного изучения студентами основ среды разработки Delphi.

Список литературы

1. Дмитриев В.Л. Развитие представлений об объектном программировании на примере разработки объектов в среде программирования Turbo Pascal // Информатика в школе. № 2 (95), 2014. – С. 54-59.
2. Поляков К.Ю., Еремин Е.А. Информатика. Углубленный уровень: учебник для 11 класса: в 2 ч., ч. 2. – М.: "БИНОМ. Лаборатория знаний", 2013. – 304 с.