|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Мытищинский филиал**  **федерального государственного бюджетного образовательного**  **учреждения высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МФ МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ: Космический

КАФЕДРА: «Прикладная математика, информатика и вычислительная техника» К3-МФ

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

**Конвертер валют**

Студент К3-45Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_ **А. М. Шехаб**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ **А. М. Ветошкин**

(Подпись, дата) (И.О.Фамилия)

*2021 г.*

**Министерство науки и высшего образования Российской Федерации**

**Мытищинский филиал**

**федерального государственного бюджетного образовательного**

**учреждения высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МФ МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине: Методы разработки ПО (программного обеспечения)

Студент группы: К3-45Б

Шехаб Адам Мохамад

(Фамилия, имя, отчество)

Тема курсовой работы: Конвертер валют

Индивидуальное задание: Создать окно перевода из одной валюты в другую, используя библиотеку Turbo Vision.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на 16 листе формата А4.

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**Руководитель курсовой работы**  **А. М. Ветошкин**

(И.О.Фамилия)

**Студент**  **А. М. Шехаб**

(И.О.Фамилия)

Оглавление

[**Введение** 4](#_Toc75131793)

[**Описание принципа работы программы** 5](#_Toc75131794)

[1. Main (Converter.pas) 5](#_Toc75131795)

[2. ModuleApp (ModuleApp.pas) 6](#_Toc75131796)

[3. ModuleTasks (ModuleTasks.pas) 9](#_Toc75131797)

[**Заключение.** 15](#_Toc75131798)

[**Литература:** 16](#_Toc75131799)

[**Ссылки на источники:** 16](#_Toc75131800)

# **Введение**

Цель работы: написать программу, которая конвертирует валюту. Написать программу нужно на языке программирования Turbo Pascal с использованием библиотеки Turbo Vision.

Для перевода я выбрал родную валюту – рубли (RUB), а также американский доллар (USD) и ливанскую лиру (LBP).

Программа получает на входе первоначальную валюту и само число. Выводиться будет денежный эквивалент этой же валюты и двух других. В своей курсовой работе я использовал наследование и виртуальные методы.

# **Описание принципа работы программы**

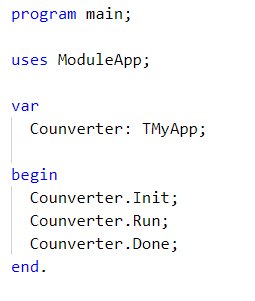
## Main (Converter.pas)

В основной программе компилятора я использую модуль **ModuleApp**, который описывается ниже.

Далее мы применяем инкапсуляцию (инкапсулируем все данные и функции-члены вместе, чтобы сформировать объект) полей и алгоритмов, которую применяем не только к графическим объектам, но и ко всей программе в целом.

Мы создаем объект-программу и даем ей 3 основных действия: инициации (**Init**), выполнению основной работы (**Run**) и завершению (**Done**).

* На этапе инициации (**Init**) экран переводится в графический режим работы и создаются, и отображаются графические объекты.
* На этапе **Run** осуществляется сканирование клавиатуры и перемещение графических объектов.
* На этапе **Done** экран переводится в текстовый режим и завершается работа всей программы.



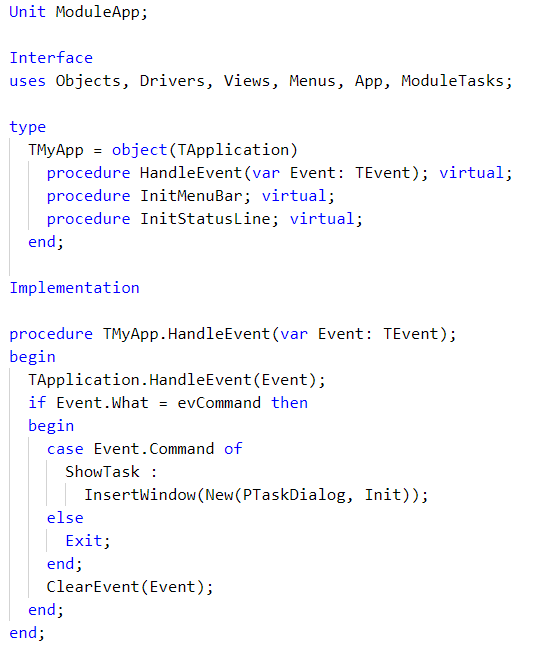
## ModuleApp (ModuleApp.pas)

Первый из двух модулей это **ModuleApp**. Он отвечает за создание и инициализацию нашего окна и добавление кнопок меню.

Окно Turbo Vision – это объект со встроенной в него возможностью реагировать на ввод пользователя без написания специального кода. Окна Turbo Vision уже знают, как открываться, изменять размер, перемещаться и закрываться. Окно Turbo Vision содержит то, что содержат и чем управляют другие объекты: эти объекты отображают себя на экране. Окно управляет видимыми элементами и функциями программы – это видимые элементы, которые окно содержит, и которыми оно управляет.

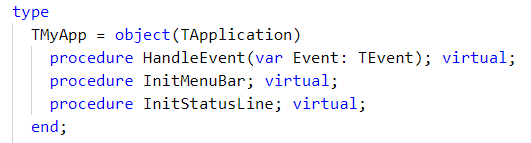
**Библиотеки**

Подключаем библиотеки дополнительных процедур и функций для управления экраном: **Objects**, **Drivers**, **Views**, **Menus**, **App**, **ModuleTasks**.

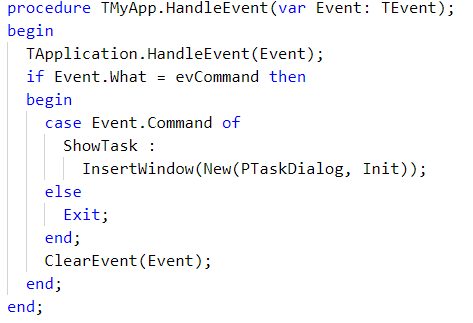


**Инициализация окна**

В наших процедурах мы будем использовать виртуальные методы (**virtual**). Виртуальный метод — это метод, чей вызов встроен в его код во время выполнения, процессом, называемым процессом поздней связи. Объявление метода виртуальным делает возможным для методов с тем же именем выполнение различными способами внутри иерархии объектных типов.

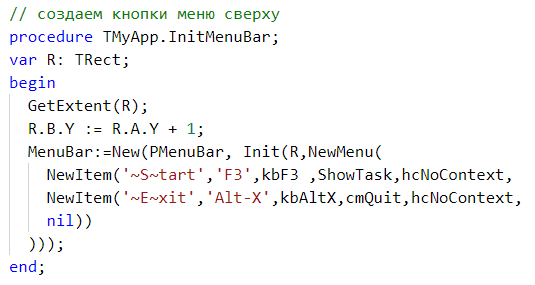


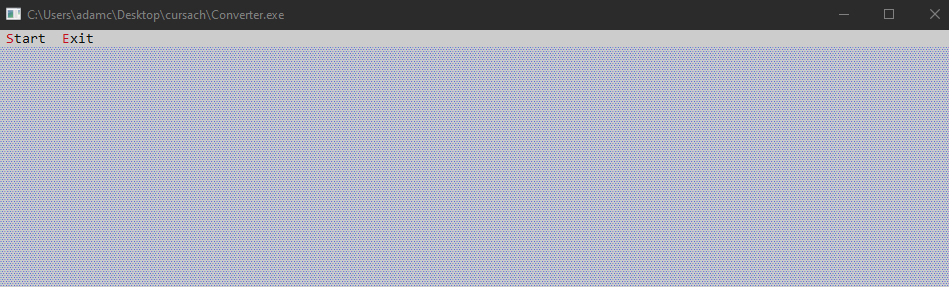
Первая процедура **TMyApp.HandleEvent** ответственна за создание окна. **TApplication.HandleEvent(Event)** в данном случае действует как предок, но откликается на дополнительные команды. **ClearEvent(Event)** очищает событие после обработки. Данный код инициализирует окно и подсоединяет его к панели экрана.

****

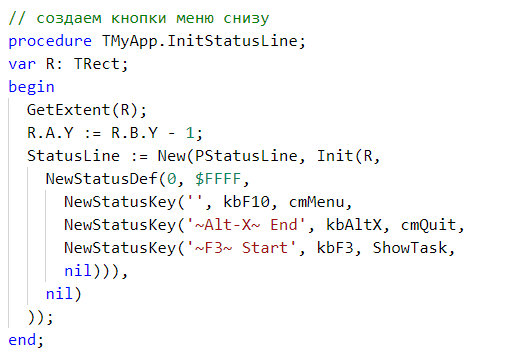
**Кнопки меню**

Добавим кнопки меню сверху. Для этого заведем другую процедуру **TMyApp.InitMenuBar**. Вверху окна мы создаем две кнопки: **Start** и **Exit**. Так же мы привязываем к ним кнопку «**F3**» и комбинацию клавиш «**Alt+X**» соответственно. Теперь при нажатии на кнопки или на клавиши у нас будет запускаться и закрываться наша программа.





Аналогично создаем такие же кнопки меню снизу. Делаем это с помощью процедуры **procedure TMyApp.InitStatusLine**.





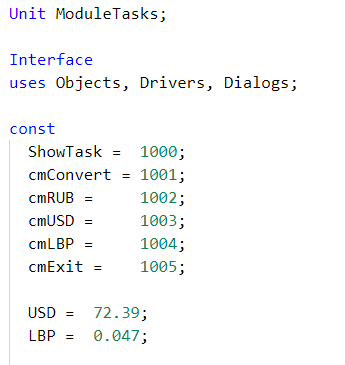
В результате у нас получилось инициализировать окно с кнопками (После того как мы зададим размеры окна во втором модуле **ModuleTasks**.

## ModuleTasks (ModuleTasks.pas)

Второй модуль **ModuleTasks** отвечает за размер окон, их положение, а также за расчет и вывод валют.

**Библиотеки и константы**

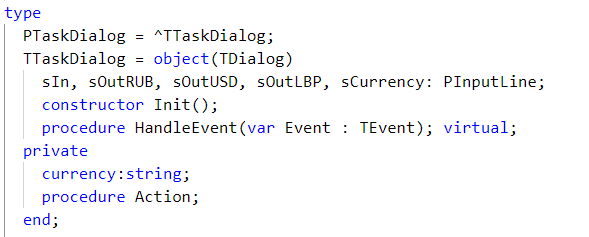
Вначале подключаем 3 библиотеки: **Objects**, **Drivers** и **Dialogs**. Задаем команды для обработчиков событий: ShowTask, cmConvert, cmRUB, cmUSD, cmLBP, cmExit. А также указываем курсы валют.



**Наследование**

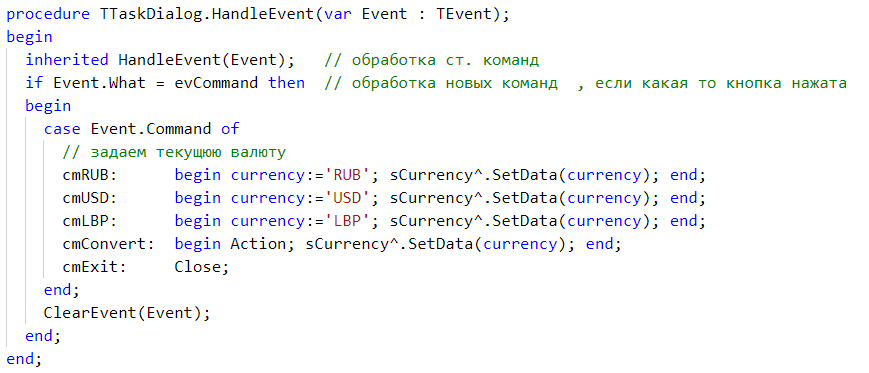
Если методы сделать виртуальными и инкапсулировать в объект-предок, каждый из его потомков сможет осуществить нужные действия одинаковым способом. Поэтому мы будем пользоваться предками.

Например в нашем случае объект **TTaskDialog** наследуется от **TDialog**.



**Выбор валюты**

Процедура **procedure TTaskDialog.HandleEvent** отвечает за обработку команд и выбор валюты.



**Inherited** используется для того, чтобы обозначить непосредственного предка вложенного метода объектного типа. В данном случае мы обрабатываем старые команды.

Условие  обрабатывает новые команды, если нажата какая-либо кнопка. При нажатии кнопки (в зависимости на какую кнопку мы нажали) мы либо выбираем текущую валюту, либо конвертируем валюту, либо закрываем окно конвертатора.

При нажатии на кнопку какой-либо валюты, мы выбираем текущую валюту.

При нажатии на кнопку **Convert**, мы переводим из данной валюты в остальные.

При нажатии на кнопку **Close**,мызакрываем окно конвертатора.

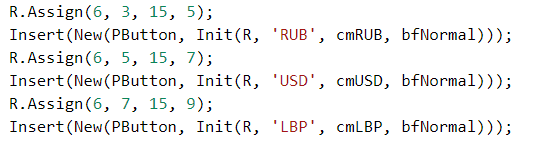
**Конструктор**

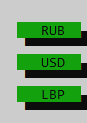
Далее мы вводим конструктор **constructor TTaskDialog.Init**. Конструктор – это специальный метод, который инициализирует объект, содержащий виртуальные методы. Тут мы будем описывать размеры и положение кнопок и окон.

Вначале мы вводим размер основного окна конвертера и пишем наверху «Converter». (Первые две цифры – координаты верхней левой точки, последние две цифры – координаты нижней правой)

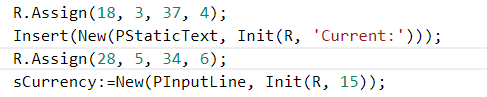


Ниже описываем положение и размеры трех кнопок выбора валют.



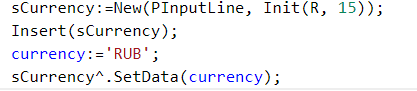


Далее вводим положение фразы «Current». Так же указываем координаты строки, в которой написана данная валюта (из которой мы переводим).

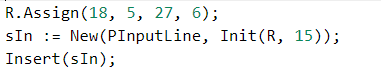




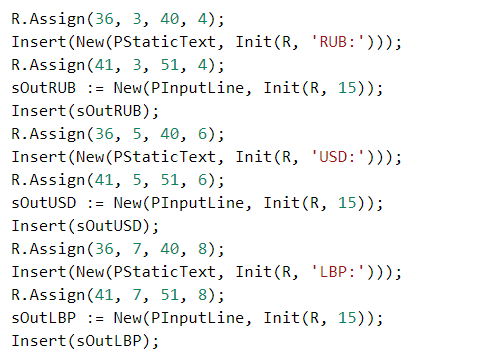
Пусть первоначальная валюта всегда будет равна рублям.

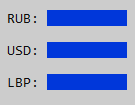


Потом идут координаты строки ввода.

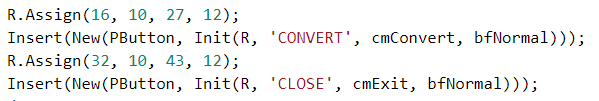


Ниже описаны окна, в которые программа будет передавать значения переведенных валют.



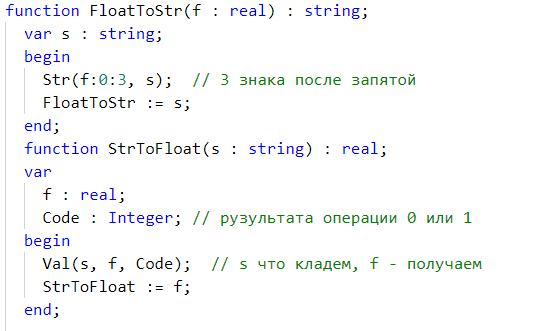


Аналогично с кнопками «**Close**» и «**Convert**», при нажатии на которые будет запускаться событие описанные выше.



**Перевод типов данных**

Описываем процедуру **procedure TTaskDialog.Action**. В ней находится две функции: **function FloatToStr** переводит число в строку, а **function StrToFloat** переводит строку в число. Нам нужны эти функции, чтобы взаимодействовать с данными, потому что, когда мы вписываем в диалоговое окно число для перевода, оно считается не как число, а как строка. Поэтому для того, чтобы выполнить какие-либо математические действия нам надо перевести эту строку в число. А когда мы все сделали, надо перевести его обратно в строку, чтобы вывести на экран. В функции **function FloatToStr** мы так же указываем количество знаков после запятой (в данный момент 3 знака после запятой).

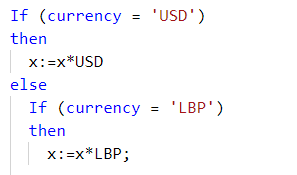


**Перевод валют**

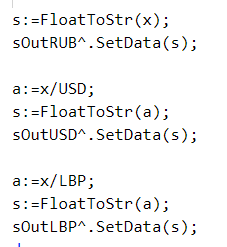
Далее идет тело программы. Мы получаем на вход строку для перевода и переводим его в число.



Так как мы считаем все ориентируясь на рубли, то сначала определяем валюту, а потом переводим все в рубли.



Потом мы выводим рубли и считаем другие валюты. Делим рубли на курс и получаем доллары и ливанскую лиру.



# **Заключение.**

В своей курсовой работе я решил все поставленные перед собой задачи. Мною был создан простой в оформлении, но очень эффективный конвертер валют. Он имеет такие функции, как выбор валюты (из предложенных), ввод числа, кнопка конвертации и кнопка закрытия окна. С помощью библиотеки Turbo Vision я смог реализовать все методы, которые хотел.

# **Литература:**

1. Фаронов В.В.Ф24 Турбо Паскаль 7.0. Начальный курс. Учебное пособие. -М.: Издательство «ОМД Групп», 2003. -616 с

# **Ссылки на источники:**

1. http://citforum.ru/programming/tv\_doc/

2. http://www.pascal.helpov.net/index/

3. https://ru.wikipedia.org/wiki/Turbo\_Pascal

4. http://mif.vspu.ru/books/pascal/