Министерство образования и науки Российской Федерации  
Южно-Уральский государственный университет  
Кафедра Информатики

Пояснительная записка по курсовому проекту  
по курсу «Высокоуровневые методы информатики и программирования»

«Обучающая программа – переводчик текста» ОО ПЗ

Руководитель  
доц. каф. информатики Конова Е.А.  
« »\_\_\_\_\_\_\_\_2007г.

Автор проекта  
студент группы  
ЭиУ-264  
Лаврова Екатерина  
«10» декабря 2007г.

Проект защищён с оценкой  
\_\_\_\_\_\_\_\_\_\_  
« »\_\_\_\_\_\_\_\_2007г.

Челябинск  
2007

# Оглавление

1 Введение 1

2 Постановка задачи 2

3 Структура программы 3

4 Спецификация 4

5 Структурная схема отношений в спецификации классов 6

6 Реализация 7

7 Алгоритмы реализации 8

8 Сценарий работы проекта 10

9 Реализация сценария 11

10 Документация проекта 17

11 Заключение 18

12 Источники информации 19

Оглавление 20

# 1 Введение

Данная курсовая работа велась на основе объектно-ориентированного программирования. Суть объектно-ориентированного программирования состоит в том, что задачу можно разложить на составные части, каждая из которых становится самостоятельным объектом. Каждая составляющая содержит свои собственные данные и методы их обработки, также, возможно, методы отношений с другими объектами. В итоге получаем некую совокупность относительно независимых объектов в их взаимосвязи, не жёсткой, но основанной определённых концепциях (без которых теряется суть ООП) – инкапсуляции, полиморфизме и наследовании.

Поэтому для того, чтобы максимально полно раскрыть все возможности разрабатываемой программы, и использовалось объектно-ориентированное программирование.

# 2 Постановка задачи

Темой проекта является составление программы – переводчика текста (по типу Lingvo).

Так как данная задача относится к разделу обучающих программ, то, кроме собственно переводчика, в программу была включена дополнительная функция – Tutor («Учитель»), позволяющая проверить словарный запас пользователя и по возможности его расширить.

Для реализации поставленной задачи потребовалось обеспечить следующую функциональность:

* Возможность перевода отдельных слов;
* Возможность перевода предложений (текстов);
* Исключение из перевода не алфавитных знаков (знаков пунктуации, чисел и пр.);
* Возможность открытия файла для перевода;
* Возможность сохранения полученного перевода;
* Возможность распечатки перевода;
* Система Tutor – «Учитель» – для проверки знаний пользователя;
* Наличие необходимой справочной системы;
* Обеспечить приложение стандартным интерфейсом: основным меню, горячими клавишами, панелями инструментов.

# 3 Структура программы

Для реализации задачи были сформированы следующие классы:

* class TWord, предназначенный для хранения пар слово-перевод;
* class TWords, класс-контейнер, данным которого является вектор классов TWord, также содержащий методы работы с вектором.

Эти классы описаны в файле Lexicon.h, реализация – в файле Lexicon.cpp.

Файлы данных – En-Ru.csv и Ru-En.csv. Файлы проекта – EasyTranslator.cpp, EasyTranslator.h, EasyTranslator.bpr (установки опций проекта), EasyTranslator.res (ресурсы проекта), EasyTranslator.tds (файл таблицы символов – для отладки).

Интерфейс программы представлен формами:

* TBody, представляющая собой тело переводчика (реализовано меню с горячими клавишами, панели инструментов…), заголовочный файл – Bodys.h, реализация – Bodys.cpp, файл формы – Bodys.dfm; Bodys.ddp;
* TETHlp – помощь и TTAbout – о программе. Соответственно файлы: Hlp.cpp, Hlp.h, Hlp.ddp, Hlp.dfm (файл формы); ETAbout.cpp, ETAbout.h, ETAbout.ddp, ETAbout.dfm;
* TTutor, для проверки и пополнения словарного запаса пользователя. Файлы: ETutor.cpp, ETutor.h, ETutor .ddp, ETutor.dfm;

Файл справки – Help.rtf.

Исполняемый файл приложения – EasyTranslator.exe.

# 4 Спецификация

Разработаны объекты TWord и TWords:

* class TWord – класс, который хранит только данные «слово-перевод».

Параметры класса:

AnsiString One, Two;

One содержит «вводимое» слово, Two – возможные варианты перевода слова One. Для максимальной функциональности используется тип AnsiString – динамическая строка с нулевым символом в конце.

* class TWords – класс-контейнер, используется для хранения в оперативной памяти содержимого файлов данных (слов с переводами), что значительно ускоряет работу программы.

Параметры класса: vector <TWord> Words;

//контейнер типа vector для хранения классов TWord;

Прототипы методов класса:void Clear();

//Очищает вектор для адекватной работы в случае изменения направления перевода, для этого используется стандартная функция-член класса vector – pop\_back();

void OpenFile(char \*Name);

//Открывает файл данных слов для заполнения классов TWord, и, соответственно, вектора TWords. Для адекватной работы вектор «обнуляется» с помощью Clear(). В функцию передаётся указатель на имя открываемого файла. Функция использует поток ifstream для доступа к файлу данных, считывая в вектор с помощью push\_back() попарно слова-значения (соответственно классам TWord). TWord GetWord(int Index);

//Определяется оператор индекса [] для обеспечения доступа к элементам вектора путём обычной индексации, соответственно, передаваемое значение – целое число, возвращаемое – класс TWord. unsigned int GetSize();

/Определяется текущий размер вектора. Возвращает беззнаковое целое.AnsiString Translate(AnsiString W);

//Обеспечивает перевод одного слова (или устойчивого словосочетания). В функцию передаётся строка (то есть слово для перевода), в векторе ищется совпадение по данному One объектов TWord, и, если находится, функция возвращает данное Two того элемента, в котором найдено совпадение. AnsiString TranslateString(AnsiString Str);

//Обеспечивает пословный перевод введённой фразы с помощью функции Translate(AnsiString W). В функцию передаётся строка (слово или текст), которая посимвольно анализируется и, в зависимости от символа, заносится в строковую переменную. Если в переданной строке встречаются определённые символы (например, знак возврата каретки или пробел), значит, слово закончилось, и его можно перевести. Перевод слов заносится в переменную-строку, которая при окончании работы возвращается функцией.

# 5 Структурная схема отношений в спецификации классов

Так как основными данными класса TWord являются строковые переменные One и Two, а вектор TWords состоит из объектов TWord (и методы вектора работают как с самим вектором, так и с данными-классами), то, в общем, схему отношений между классами можно описать так:

Контейнер-вектор, хранящий классы TWord

Класс, хранящий данные слово-перевод

Данное класса TWord, хранящее слово для перевода

Данное класса TWord, хранящее варианты перевода

Методы класса-контейнера TWords

TWords

TWord

One

Two

TWord

One

Two

Рисунок 1 – Схема отношений в спецификации классов

# 6 Реализация

Так как особенностью поставленной задачи было обеспечение пословного перевода, то в качестве основного объекта, предназначенного для хранения данных, был выбран контейнер – вектор. Преимуществом использования вектора стало то, что в этом классе поддерживаются динамические массивы, а это значит, что размер контейнера может изменяться по мере необходимости, и одновременно с этим доступ к его элементам осуществляется с помощью обычной индексной нотации (оператора []).

Получаем, что для хранения относительно небольшого количества информации о слове-переводе используется класс TWord, а для хранения и обработки неизвестного на момент компиляции количества слов (соответственно, и классов TWord) используется вектор TWords, что делает все преимущества применения этого типа контейнера очевидными.

# 7 Алгоритмы реализации

Код программы снабжён комментариями на русском языке, поэтому суть работы программы можно понять, особо не вдаваясь в подробности механизма работы отдельных функций. Однако рассмотрим некоторые из методов.

Перевод одного слова (или устойчивого словосочетания) осуществляется с помощью довольно простого метода Translate:

AnsiString TWords::Translate(AnsiString W)

{ for (unsigned int r=0; r<GetSize();++r) //пробегаем по вектору слов

if (GetWord(r).One==W) //если первое слово "подкласса"

//\_вектора равно искомому

return GetWord(r).Two; //вернуть второе слово "подкласса"

return ""; //иначе возвращается пустая строка}

Функция Translate непосредственно вызывается только из функции TranslateString, ответственной за пословный перевод текста.

AnsiString TWords::TranslateString(AnsiString Str) //передаётся введённый текст

{ Str=Str+'\n'; //добавление перевода строки

AnsiString W; //отдельное слово из введённого текста

AnsiString outp; //строка-перевод

for(int i=1;i<=Str.Length();i++) //пробегаем по длине введённого текста

{ f(Str[i]!=' ' && Str[i]!='\r' && Str[i]!='\n' && ((Str[i] >= 'a' && Str[i]<='z') || (Str[i] >= 'а' && Str[i]<='я')))

W=W+Str[i]; //посимвольно формируем слово (исключая знаки-небуквы)

else //если символ-пробел или строка закончилась

if(W!="") //если слово непустое

{

//в строку перевода пишем исходное слово + перевод + возврат каретки

outp=outp+W.UpperCase() + " : "+Translate(W)+"\n";

W=""; //обнуляем переводимое слово

}

}

return outp;}

Работа Tutor, описанная в файле реализации формы TTutor, построена практически аналогично работе основного переводчика – на основании функции TranslateString:

void \_\_fastcall TTutor::CheckClick(TObject \*Sender)

{

AnsiString W = TwoW->Text.LowerCase();

if(Wordz.Translate(W)==OneW->Text.LowerCase())

//если перевод введённого слова совпадает с данным пользователю описанием

{ Result->Caption="That's right!! =) " + Wordz.GetWord(k).One;//правильно!

// выбор соответствующе картинки }

else {//если перевод введённого слова не совпадает

Result->Caption="That's wrong! =( "+Wordz.GetWord(k).One;//неправильно!

// выбор соответствующе картинки

# 8 Сценарий работы проекта

После инсталляции программы из самораспаковывающегося архива, который содержит все необходимые для адекватной работы приложения файлы, необходимо запустить исполняемый файл EasyTranslator.exe, после чего откроется главное окно переводчика.

После ввода текста в верхнее текстовое поле, нажав кнопку Translate или Enter, получаем пословный перевод вида ВВЕДЁННОЕ\_СЛОВО : перевод.

Изменение направления перевода – с помощью списка на главной форме.

Пользователю доступны все разделы меню, поэтому можно открыть текстовый файл изменить содержимое текста, а потом сохранить или распечатать полученный пословный перевод. Некоторые разделы меню (например, Правка) представлены в виде панелей инструментов, которые можно отобразить или скрыть.

Приступить к работе с Tutor можно, нажав кнопку Tutor главного окна программы, закончить работу – нажав Back to Translator или красный крест.

По нажатию Contents раздела меню Help или клавиши Ctrl+F1 можно ознакомиться со справкой по программе.

Основными правилами, предъявляемыми для пользователя, являются:

* Не удалять и не перемещать файлы, поставляемые с программой (любые файлы программы после распаковки архива являются неприкосновенными и не подлежат коррекции);
* Не пытаться использовать Easy Translator в не предназначенных и не рассчитанных для него целях;
* Относиться с уважением к работе автора программы

# 9 Реализация сценария

Верхнее текстовое поле главной формы предназначено для ввода переводимого текста, нижнее поле – для полученного перевода (поле перевода не подлежит правке – для этого при создании формы установлен параметр полей ReadOnly=True. Также устанавливается внешний вид полей, и загружается первоначальная база слов). Для получения перевода достаточно нажать большую и заметную кнопку Translate! в окне или Enter на клавиатуре (обрабатывается нажатие кнопки или анализируется нажатая кнопка клавиатуры при фокусе в поле ввода).

Кнопка для перевода, аналогична нажатию Enter

Полученный перевод

Поле ввода текста

Направление перевода

Переход к Tutor (Учителю)

Главное меню программы

Панели инструментов

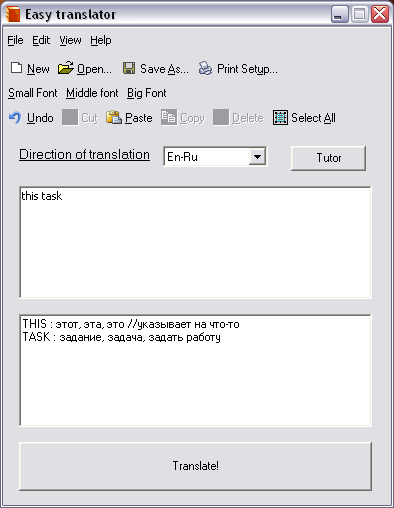


Рисунок 2 – Главное окно переводчика

Пользователь может изменять направление перевода (с русского на английский или с английского на русский) с помощью списка Direction of translation. При этом ранее переведённые слова (если они были противоположной направленности) теряют перевод. Реализовано с помощью методов класса TWords – содержание вектора меняется в соответствии с заданным направлением перевода:

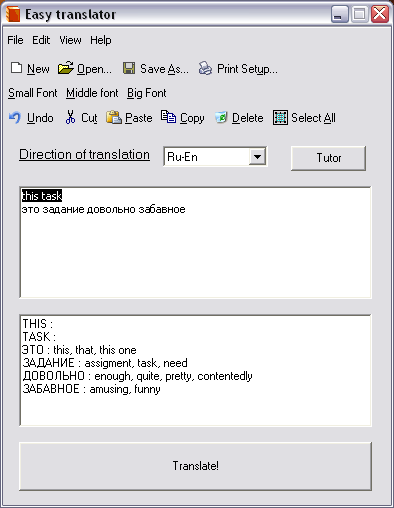


Рисунок 3 – Главное окно переводчика. Направление перевода

Программа была рассчитана на пользователя с хотя бы минимальными знаниями английского языка, поэтому практически весь интерфейс (названия кнопок, пунктов меню) реализован на английском. Однако для пользователей, совсем не знающих этого языка, предусмотрены подсказки на русском, они становятся видны при наведении курсора мыши на интересующую часть окна (надписи обеспечены свойством Hint, видимость надписей – свойством ShowHint=True).

В программе предусмотрено стандартное меню, оснащённое горячими клавишами, что позволяет пользователю открывать текстовые файлы для перевода, сохранять и распечатывать перевод, а также редактировать текущий текст для перевода. Меню реализовано с помощью компонента ActionManager, с помощью которого легко создаются и реализуются разделы меню (по необходимости снабжённые горячими клавишами и встроенными диалогами), к которым могут быть привязаны иконки действий по индексу компонента Images (ImageList). Пользователь может менять внешний вид приложения, например, изменять размер шрифта. Также возможно скрывать или отображать панели инструментов (которые были созданы также с помощью ActionManager) – с помощью контекстного меню (компонент PopupTools – PopupMenu, реализация действий взята из ActionManager):



Рисунок 4 – Изменение состояния панелей инструментов с помощью контекстного меню

или с помощью главного меню:



Рисунок 5 – Изменение состояния панелей инструментов с помощью главного меню

Контекстные меню предусмотрены и у обоих текстовых полей, действия также взяты из ActionManager.

Для перехода к Tutor необходимо нажать кнопку Tutor главного окна. Для этого было необходимо включить заголовочный файл TTutor.h в файл описания главного окна, после чего возможно спрятать главное окно и показать окно Tutor. При каждом обращении к этому окну случайным образом генерируется новое задание:

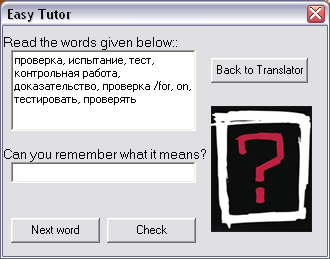


Рисунок 6 – Окно Tutor

Tutor предлагает ознакомится с сочетанием слов, данным в верхнем текстовом поле, и предложить свой вариант их перевода, записав его в нижнее поле, и затем нажав кнопку Check! Программа сравнивает перевод введённого слова с данными в верхнем поле словосочетаниями. Код обработки нажатия этой кнопки можно посмотреть в разделе 7 настоящей записки.

Если результат перевода верный, появится соответствующая надпись и забавная картинка (все картинки этой формы были загружены программно и их поставка с программой не требуется):

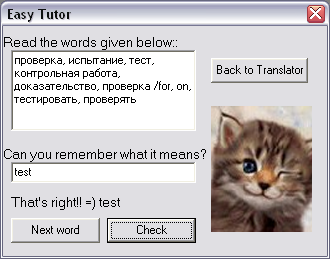


Рисунок 7 – Окно Tutor. Слово переведено верно

Если пользователь ошибся, он сразу же об этом узнает по соответствующей надписи и картинке:

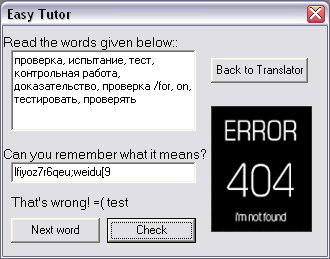


Рисунок 8 – Окно Tutor. Слово переведено неверно

Для возврата в Easy Translator достаточно нажать кнопку Back… или кнопку закрытия окна.

Также проект содержит небольшую, но информативную справку по программе, которую можно вызвать из раздела меню Help – Contents или нажав Ctrl+F1. Для раздела справки предназначена специальная форма ETHlp:

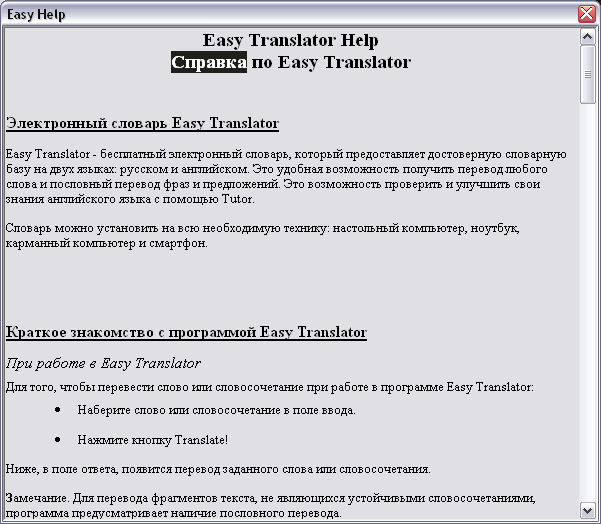


Рисунок 9 – Окно помощи

Также пользователю доступна информация о программе и составителе (Help – About…):



Рисунок 10 – Окно О программе

Некоторые особенности программы:

* Отключается возможность запускать несколько копий программы. При попытке запустить более одной версии выдаётся сообщение о невозможности выполнения операции;
* Из всех заголовков окон проекта была убрана кнопка Maximize, для запрещения изменения размеров с целью сохранения задуманных при дизайне пропорций;
* Для всех форм была добавлена иконка, которая видна в полосе задач рядом с именем программы или при просмотре пользователем каталога, в котором расположен выполняемый файл приложения, также эта иконка может отображаться в левом верхнем углу окна приложения.
* На данный момент программа имеет довольно узкие возможности перевода (около 15 слов в каждом направлении, список слов прилагается в текстовом файле), так как добавление слов в базу данных – процесс очень трудоёмкий и занимает сравнительно много времени

# 10 Документация проекта

Так как программа была реализована с параметрами, обеспечивающими независимость программного кода от среды разработки, то Easy Translator обладает полной переносимостью.

Для установки программы требуется самораспаковывающийся архив EasyTranslator.exe (никаких дополнительных программ для распаковки не требуется).

Требования для установки и работы с программой:

* Наличие настольного компьютера, ноутбука или КПК
* Версия Windows не ниже 95
* Нежелание перемещать и/или удалять файлы, образующиеся после установки программы

# 11 Заключение

Данный проект полностью реализует возможности использования средств объектно-ориентированного программирования на примере создания программного продукта – переводчика, а также иллюстрирует возможности использования среды разработки – Borland C++ Builder 6.

В качестве развития проекта можно предположить расширение словарной базы переводчика (как программного, так и пользовательского), возможность интеграции с другими приложениями (вызов программы из открытых документов), а также возможность использования словарей для тематического перевода.

# 12 Источники информации

1. Г. Шилдт «Самоучитель С++, 3е издание», Издательство «БХВ-Петербург» 2006

2. А.Я. Архангельский «C++ Builder 6, Справочное пособие, Книга 2 – Классы и компоненты», Издательство «Бином» 2002

3. Учащиеся групп Эиу-264 и Эиу-263 ЮУрГУ

4. Лекции по курсу «Высокоуровневые методы информатики и программирования»

# Оглавление

1 Введение 1

2 Постановка задачи 2

3 Структура программы 3

4 Спецификация 4

5 Структурная схема отношений в спецификации классов 6

6 Реализация 7

7 Алгоритмы реализации 8

8 Сценарий работы проекта 10

9 Реализация сценария 11

10 Документация проекта 17

11 Заключение 18

12 Источники информации 19

Оглавление 20