Санкт-Петербургский Государственный политехнический университет

Факультет технической кибернетики

Кафедра компьютерных систем и программных технологий

# Курсовая работа по курсу «Программирование»

# Тема работы: «Разработка приложения с графическим интерфейсом на языке программирования С++ с использованием библиотеки QT»

Студент: Баратынский Александр

Преподаватель: Пышкин Евгений Валерьевич

2012 г.

# Образ и границы проекта

В качестве курсовой работы по программированию предлагается реализация игры «Виселица». Приложение должно быть реализовано на языке программирования C++ с использованием библиотеки QT. Данное приложение должно быть с графическим интерфейсом.

Основные моменты игры:

Суть игры состоит в том, чтобы отгадать предоставленное слово за определенное количество попыток. Графический интерфейс должен выглядеть следующим образом: в верхней части экрана должен появиться набор символов, определяющий количество букв в слове, снизу же должен быть набор кнопок с буквами на них. Пользователь должен нажимать на букву, которую выбрал, она будет зачеркиваться, и, если он ее угадал, то буква должна появиться сверху, если же нет, то слева появится виселица. По мере прохождения, если пользователь плохо отгадывает буквы, на виселице вырисовывается человечек. Победа в игре присуждается, если пользователь угадал слово, поражение – если на виселице полностью вырисовался человечек.

Также, я рассматриваю возможность сделать игру на нескольких языках.

# Техническое задание

Реализовать игру, описанную в первом пункте, на языке программирования С++ с использованием библиотеки QT.

2 этапа разработки игры:

1. Реализация проекта в консольном режиме
2. Последующая интеграция кода с библиотекой QT. Реализация графического интерфейса.

Требования к проекту:

Главное меню должно содержать следующие пункты:

Выбор языка: русский или английский.

Сценарии выполнения программы:

При запуске приложения на экран выводится окно, где предлагается выбрать язык. Когда пользователь выбрал язык, подгружается соответствующий текстовый файл со словами. Оттуда выбирается случайное слово и начинается игра.

# Требования к консольному режиму

1. Выбор языка
2. На экран должна выводиться последовательность \_, показывающая количество букв в слове. По мере отгадывания, буквы появляются вместо звездочек.
3. Должно выводиться число, показывающее число оставшихся попыток.

# Требования к графическому интерфейсу

1. Пункты меню в виде кнопок
2. Поле игры.
3. Вместо числа попыток, должна отображаться виселица
4. Остальные пункты идентичны.

# Системная архитектура

## Классы консольного приложения

В консольном приложении будут реализованы следующие классы:

1. class BasicWord

{

protected:

char letter;

string word;

public:

BasicWord();

char mistakes[6];

string getWord();

friend class Library;

};

Класс BasicWord отвечает за представление слова в целом.

1. class DictionaryWord : public BasicWord

{

public:

DictionaryWord();

void setWord(string dword);

};

Класс DictionaryWord – производный от BasicWord, отвечает за представление словарного слова.

1. class UserWord : public BasicWord

{

void enterLetter();

public:

UserWord();

void makeUserWord(DictionaryWord dword);

void putMistake(int curMisses);

bool checkLetter(DictionaryWord dword);

void putLetter(DictionaryWord dword);

};

Класс UserWord – производный от BasicWord, отвечает за представление конструируемого пользователем слова.

1. class Library

{

int randNum;

int var;

DictionaryWord \*word;

int numberOfWords(ifstream &in);

DictionaryWord\* takeWord(ifstream &in);

public:

Library();

~Library();

DictionaryWord\* takeWordOfLang();

int chooseTheme();

};

Класс Library отвечает за взятие слова из текстового файла.

1. class Man

{

void createMan();

void drawMan();

public:

char man[6];

int maxMisses;

int curMisses;

Man();

void makeMan();

};

Класс Man отвечает за конструирование человечка на виселице.

1. class Game

{

Man \*man;

Library \*lib;

DictionaryWord \*dword;

UserWord \*uword;

bool isWon();

bool isLose();

bool playAgain();

void reset();

~Game();

public:

Game();

int play();

};

Класс Game отвечает за сам игровой процесс.

class Test

{

DictionaryWord \*word;

UserWord \*uword;

Game \*game;

Man \*man;

DictionaryWord\* takeTheThirdWord();

bool checkTheWord();

bool checkLetter();

bool testCheckLetterMethod();

bool checkPutLetter();

bool checkIsWon();

bool checkIsLose();

public:

Test();

~Test();

bool fullTest();

};

Процесс тестирования приложения:

1. Метод checkTheWord() проверяет было ли взято то слово, которое планировалось взять.
2. Метод checkLetter() проверяет работоспособность метода enterLetter() класса UserWord.
3. Метод testCheckLetterMethod() проверяет работоспособность метода checkLetter(DictionaryWord word) класса UserWord.
4. Метод checkPutLetter() проверяет, вставилась ли угаданная буква на нужное место.
5. Метод checkIsWon() проверяет метод isWon() класса Game.
6. Метод checkIsLose() проверяет метод isLose() класса Game.

Функциональность консольного приложения

1. В данном консольном приложении реализовано взятие случайного слова из файла. Причем пользователь может выбрать язык слов.
2. Затем появляется изображение виселицы, составленное из определенных символов.
3. Ниже виселицы появляется маска слова.
4. Затем пользователю предлагается ввести букву, которая, по его мнению, должна быть в слове.
5. Если буква есть в слове, она появляется, если же нет – то на виселице появляются части тела человечка, реализованные в классе Man.
6. Игра заканчивается, когда заканчивается количество попыток угадать букву или же, когда пользователь отгадывает слово.
7. Далее пользователю предлагается начало новой игры.

## Классы графического приложения

К вышеизложенным классам консольного интерфейса добавляются следующие классы:

1. class MainWidget : public QWidget

{

Q\_OBJECT

public:

explicit MainWidget(QWidget \*parent = 0);

~*MainWidget*();

Game \*game;

IfLoseWidget \*iflose;

IfWinWidget \*ifwin;

void connectButtons();

void makeLabel();

void addWidgets(QGridLayout \*layout);

void checkLetter();

public:

void play();

protected:

void *paintEvent*(QPaintEvent \*event);

private slots:

void on\_a\_clicked();

void on\_b\_clicked();

void on\_c\_clicked();

void on\_d\_clicked();

void on\_e\_clicked();

void on\_f\_clicked();

void on\_g\_clicked();

void on\_h\_clicked();

void on\_i\_clicked();

void on\_j\_clicked();

void on\_k\_clicked();

void on\_l\_clicked();

void on\_m\_clicked();

void on\_n\_clicked();

void on\_o\_clicked();

void on\_p\_clicked();

void on\_q\_clicked();

void on\_r\_clicked();

void on\_s\_clicked();

void on\_t\_clicked();

void on\_u\_clicked();

void on\_v\_clicked();

void on\_w\_clicked();

void on\_x\_clicked();

void on\_y\_clicked();

void on\_z\_clicked();

private:

Ui::MainWidget \*ui;

};

Этот класс отвечает за представление английского интерфейса.

1. class RussianWidget : public QWidget

{

Q\_OBJECT

public:

explicit RussianWidget(QWidget \*parent = 0);

~*RussianWidget*();

Game \*game;

RusLoseWidget \*iflose;

RusWinWidget \*ifwin;

void connectButtons();

void makeLabel();

void addWidgets(QGridLayout \*layout);

void checkLetter();

public:

void play();

protected:

void *paintEvent*(QPaintEvent \*event);

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

void on\_pushButton\_3\_clicked();

void on\_pushButton\_4\_clicked();

void on\_pushButton\_5\_clicked();

void on\_pushButton\_6\_clicked();

void on\_pushButton\_7\_clicked();

void on\_pushButton\_8\_clicked();

void on\_pushButton\_9\_clicked();

void on\_pushButton\_10\_clicked();

void on\_pushButton\_11\_clicked();

void on\_pushButton\_12\_clicked();

void on\_pushButton\_13\_clicked();

void on\_pushButton\_14\_clicked();

void on\_pushButton\_15\_clicked();

void on\_pushButton\_16\_clicked();

void on\_pushButton\_24\_clicked();

void on\_pushButton\_23\_clicked();

void on\_pushButton\_22\_clicked();

void on\_pushButton\_21\_clicked();

void on\_pushButton\_20\_clicked();

void on\_pushButton\_19\_clicked();

void on\_pushButton\_18\_clicked();

void on\_pushButton\_17\_clicked();

void on\_pushButton\_32\_clicked();

void on\_pushButton\_31\_clicked();

void on\_pushButton\_30\_clicked();

void on\_pushButton\_29\_clicked();

void on\_pushButton\_28\_clicked();

void on\_pushButton\_27\_clicked();

void on\_pushButton\_26\_clicked();

void on\_pushButton\_25\_clicked();

private:

Ui::RussianWidget \*ui;

};

Этот класс отвечает за представление интерфейса на русском языке.

1. class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~*MainWindow*();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_2\_clicked();

private:

Ui::MainWindow \*ui;

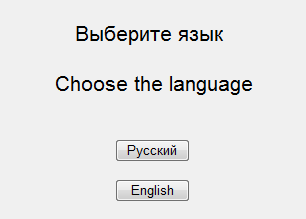
};

Этот класс отвечает за начальное окно.

Также имеются вспомогательные классы, реализующие взаимодействие с пользователем.

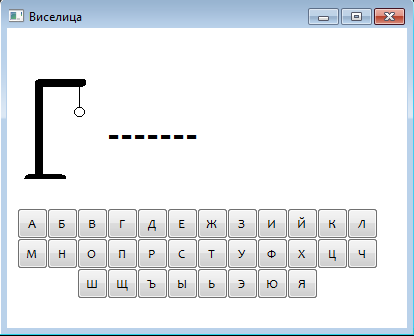
# Графический интерфейс приложения

Главное окно приложения:

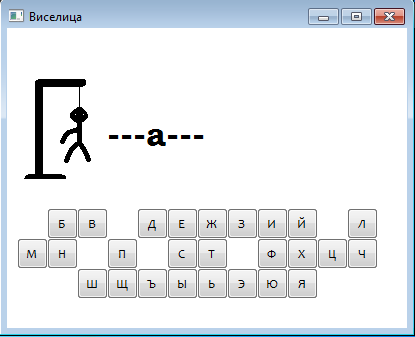


При нажатии на одну из кнопок, вызывается один из виджетов – MainWidget для английского языка и RussianWidget для русского.

При выборе русского языка появляется следующее окно:

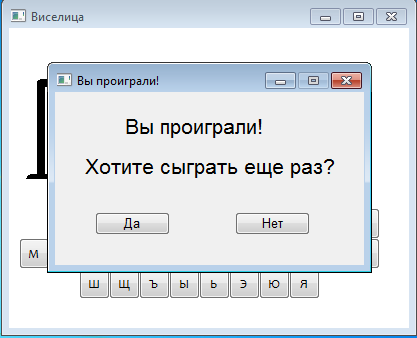


Середина игры:



При нажатии на кнопку с буквой, она исчезает.

Конец игры:



Если на виселице полностью прорисовался человечек, то открывается виджет IfLose, при нажатии на кнопку “Да”, заново создается основной виджет, а открытый закрывается. При нажатии на кнопку “Нет”, приложение закрывается.