КАЗАНСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. А.Н.Туполева - КАИ

(КНИТУ-КАИ)

Отделение среднего профессионального образования института

Компьютерных технологий и защиты информации

«Колледж информационных технологий»

Лабораторная работа №3

по предмету:

Наладчик технологического оборудования

Выполнил:

студент 4 курса

Группы 4441

Зарипов Ислам Радифович

Казань 2019 г.

**Цель работы:**

Разработка программы позволяющий пользователю создавать, редактировать, сохранять документы формата «DOC».

**Решение:**

Были установлены слушатели на кнопки «Новый», «Открыть», «Сохранить», «Закрыть». При нажатии на копку «Новый» в памяти инициализируется пустой документ, открывается возможность редактировать документ (см. Рис. 3). Если было вызвано событие открытия файла, то отображается окно выбора файла (см. Рис. 4), затем – содержание файла можно редактировать в текстовом поле программы. После изменения, файл можно сохранить, нажав на кнопку «Сохранить». Так же файл можно закрыть нажатие кнопки «Закрыть». Предусмотрена проверка изменения файла при закрытии файла (см. Рис. 5). Для чтения данных из .doc фала была выбрана библиотека OpenXML, подключение через NuGet:

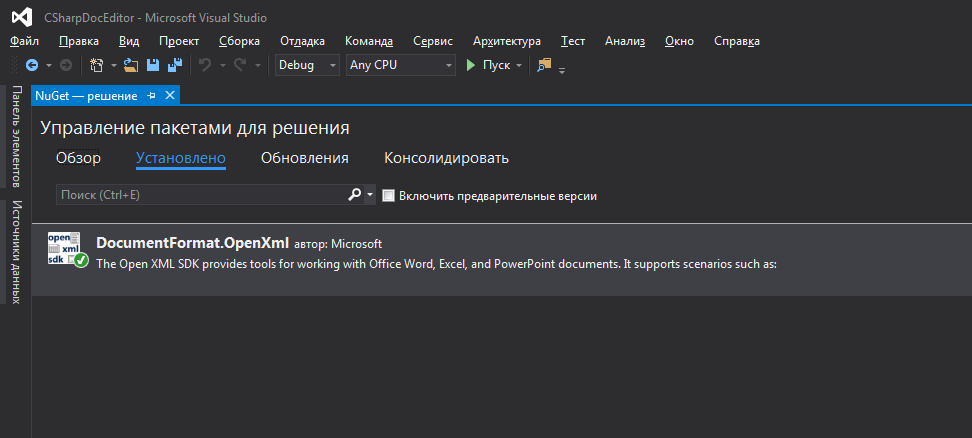


Рис. 1. Подключение OpenXML

Добавлена возможность смены размера текста (см. Рис. 3 – верхняя панель) и пример (см. Рис. 7).

**Скриншоты работы программы на C#:**

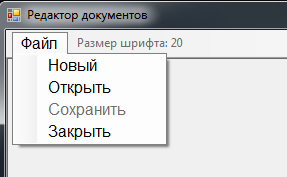
****

Рис. 2. Контекстное меню программы

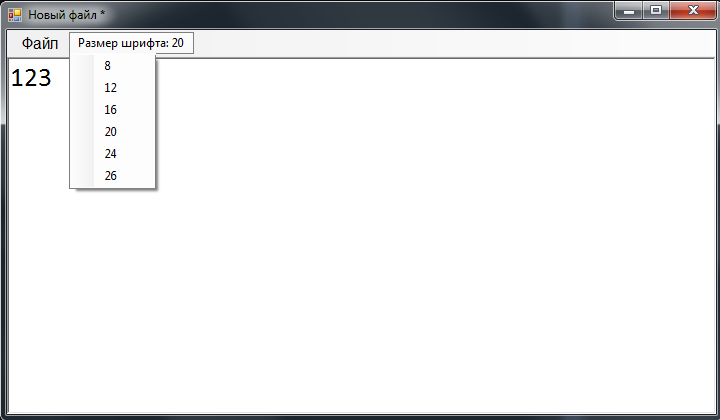


Рис. 3. Окно редактирования нового файла

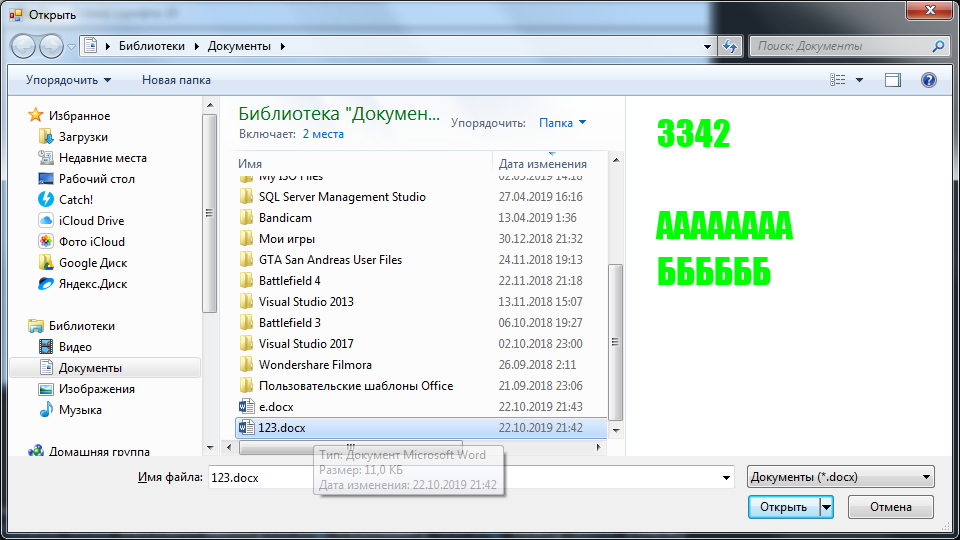


Рис. 4. Окно выбора файла для открытия

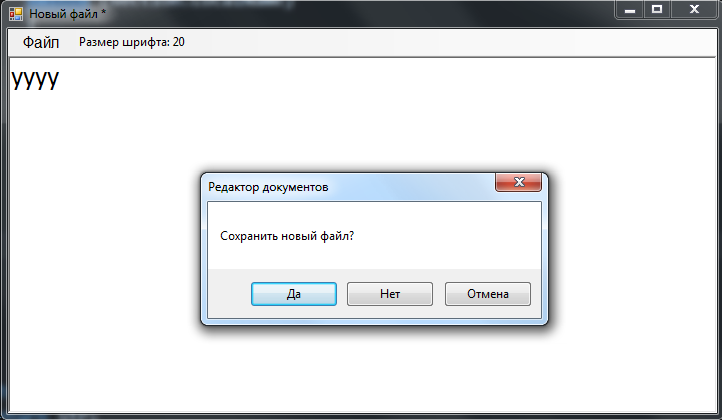


Рис. 5. Окно подтверждения «сохранить или нет»

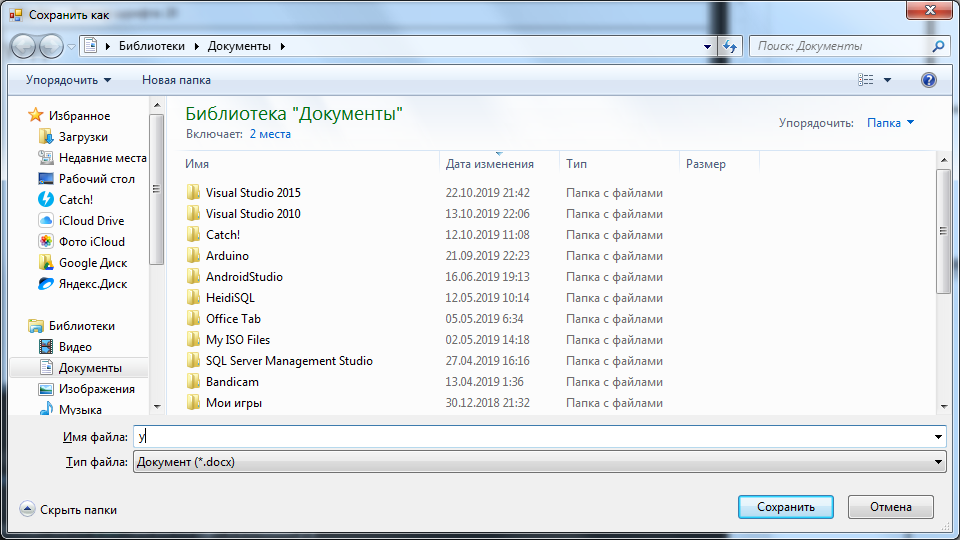


Рис. 6. Окно сохранения нового файла

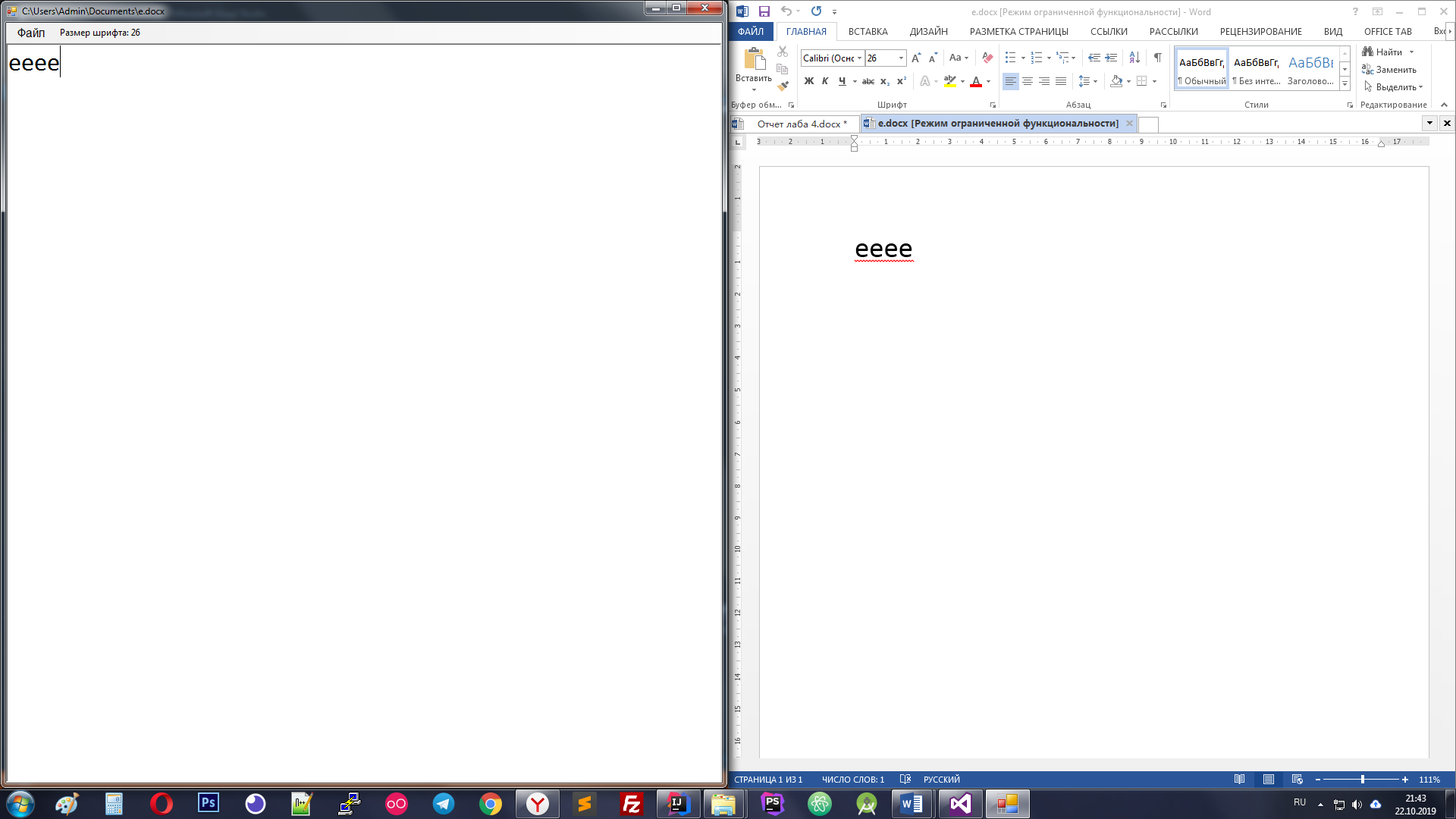


Рис. 7. Пример изменения параметров шрифта в документе

**Вывод:**

В ходе лабораторной работы была разработана программа, позволяющая пользователю создавать, редактировать, сохранять документы формата «DOC».

**Приложение**

**Файл Form1.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO.Packaging;

using DocumentFormat.OpenXml.Packaging;

using DocumentFormat.OpenXml.Wordprocessing;

using System.Xml;

using System.Xml.Linq;

using DocumentFormat.OpenXml;

namespace CSharpDocEditor

{

public partial class Form1 : Form

{

string currentFilePath = null;

string loadedFileText = null;

int loadedFileTextSize = 20;

bool edited = false;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

this.Resize += new EventHandler(Form1\_OnResize);

this.FormClosing += Form1\_FormClosing;

this.textBox.KeyUp += textBox\_KeyUp;

}

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

if (!checkIsSaved()) e.Cancel = true;

}

private void Form1\_OnResize(object sender, System.EventArgs e)

{

textBox.Size = new Size(this.Size.Width - 16, this.Size.Height - 65);

}

private void newMenuItem\_Click(object sender, EventArgs e)

{

if (!checkIsSaved()) return;

currentFilePath = null;

loadedFileText = "";

loadedFileTextSize = 20;

textBox.Text = "";

this.Text = "Новый файл";

edited = false;

textBox.Enabled = true;

saveMenuItem.Enabled = true;

sizeMenuItem.Enabled = true;

textBox.ForeColor = System.Drawing.ColorTranslator.FromHtml("#000000");

textBox.Font = new System.Drawing.Font("Calibri", loadedFileTextSize, FontStyle.Regular);

}

private void openMenuItem\_Click(object sender, EventArgs e)

{

if (!checkIsSaved()) return;

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.InitialDirectory = ".";

openFileDialog.Filter = "Документы (\*.docx)|\*.docx";

openFileDialog.FilterIndex = 0;

openFileDialog.RestoreDirectory = true;

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

string filepath = openFileDialog.FileName;

DocumentFormat.OpenXml.Wordprocessing.Color docColor = null;

string docFont = "Calibri";

loadedFileTextSize = 20;

loadedFileText = "";

try

{

using (WordprocessingDocument wordDocument = WordprocessingDocument.Open(filepath, true))

{

currentFilePath = filepath;

this.Text = currentFilePath;

Body body = wordDocument.MainDocumentPart.Document.Body;

var text = getPlainText(body);

loadedFileText = text;

Run r = wordDocument.MainDocumentPart.Document.Descendants<Run>().First();

int size = 0;

Int32.TryParse(r.RunProperties.FontSize.Val, out size);

size /= 2;

loadedFileTextSize = size;

sizeMenuItem.Text = "Размер шрифта: " + size;

docColor = r.RunProperties.Color;

docFont = r.RunProperties.RunFonts.Ascii;

}

} catch { }

if (docColor != null)

textBox.ForeColor = System.Drawing.ColorTranslator.FromHtml("#" + docColor.Val.ToString());

else

textBox.ForeColor = System.Drawing.Color.Black;

textBox.Font = new System.Drawing.Font(docFont, loadedFileTextSize, FontStyle.Regular);

textBox.Text = loadedFileText;

edited = false;

textBox.Enabled = true;

saveMenuItem.Enabled = true;

sizeMenuItem.Enabled = true;

}

}

public string getPlainText(OpenXmlElement element)

{

StringBuilder text = new StringBuilder();

foreach (OpenXmlElement section in element.Elements())

{

switch (section.LocalName)

{

case "t":

text.Append(section.InnerText);

break;

case "br":

text.Append("\n");

break;

case "tab":

text.Append("\t");

break;

default:

text.Append(getPlainText(section));

break;

}

}

string str = text.ToString();

return str;

}

private void buildDocument(string fileName, string text, bool isNew)

{

System.IO.File.Delete(@fileName);

using (var wordDoc = WordprocessingDocument.Create(fileName, WordprocessingDocumentType.Document, true))

{

List<string> resultList = new List<string>();

bool previousEmpty = false;

foreach (string split in text.Split(new[] { Environment.NewLine, "\v" }, StringSplitOptions.None))

{

if (!string.IsNullOrEmpty(split))

previousEmpty = false;

else if (!previousEmpty)

{

previousEmpty = true;

continue;

}

resultList.Add(split);

}

var mainPart = wordDoc.AddMainDocumentPart();

mainPart.Document = new Document();

var run = new Run();

var paragraph = new Paragraph(run);

var body = new Body(paragraph);

mainPart.Document.Append(body);

var runProp = new RunProperties();

var runFont = new RunFonts { Ascii = textBox.Font.FontFamily.Name };

var size = new FontSize { Val = new StringValue((textBox.Font.Size \* 2).ToString()) };

var color = new DocumentFormat.OpenXml.Wordprocessing.Color { Val = HexConverter(textBox.ForeColor) };

runProp.AppendChild(runFont);

runProp.AppendChild(size);

runProp.AppendChild(color);

Break lastBreak = new Break();

Run r = mainPart.Document.Descendants<Run>().First();

r.PrependChild<RunProperties>(runProp);

foreach (var line in text.Split(new string[] { "\n" }, StringSplitOptions.None))

{

run.AppendChild(new Text(line));

run.AppendChild(lastBreak = new Break());

}

run.RemoveChild(lastBreak);

//MessageBox.Show(HexConverter(textBox.ForeColor));

mainPart.Document.Save();

wordDoc.Close();

}

}

private String HexConverter(System.Drawing.Color c)

{

String rtn = String.Empty;

try

{

rtn = c.R.ToString("X2") + c.G.ToString("X2") + c.B.ToString("X2");

}

catch (Exception ex) { }

return rtn;

}

private void saveMenuItem\_Click(object sender, EventArgs e)

{

saveCurrentFile(true);

}

// return: true - продолжить; false - отмена

private bool checkIsSaved()

{

if (!edited) return true;

string what = currentFilePath == null ? "новый файл" : "\"" + currentFilePath + "\"";

DialogResult dialogResult = MessageBox.Show("Сохранить " + what + "?", "Редактор документов", MessageBoxButtons.YesNoCancel);

if (dialogResult == DialogResult.Yes)

{

saveCurrentFile(false);

return true;

}

else if (dialogResult == DialogResult.No) return true;

else return false;

}

private void saveCurrentFile(bool andOpen)

{

string text = textBox.Text;

float size = textBox.Font.Size;

if (currentFilePath != null)

{

buildDocument(currentFilePath, text, false);

loadedFileText = text;

loadedFileTextSize = (int) size;

this.Text = currentFilePath;

edited = false;

}

else

{

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.InitialDirectory = ".";

saveFileDialog.Filter = "Документ (\*.docx)|\*.docx";

if (saveFileDialog.ShowDialog() == DialogResult.OK)

{

string filename = saveFileDialog.FileName;

buildDocument(filename, text, true);

currentFilePath = filename;

this.Text = currentFilePath;

loadedFileText = text;

loadedFileTextSize = (int)size;

edited = false;

}

}

}

private void textBox\_KeyUp(object sender, EventArgs e)

{

if (loadedFileText != null)

{

if (!loadedFileText.Equals(textBox.Text)) setEdited(true);

else setEdited(false);

}

}

private void setEdited(bool edited)

{

edited = edited || ! loadedFileText.Equals(textBox.Text);

this.edited = edited;

if (edited)

{

if (currentFilePath != null) this.Text = currentFilePath + " \*";

else this.Text = "Новый файл \*";

}

else {

if (currentFilePath != null) this.Text = currentFilePath;

else this.Text = "Новый файл";

}

}

private void closeMenuItem\_Click(object sender, EventArgs e)

{

if (!checkIsSaved()) return;

this.Text = "Редактор документов";

sizeMenuItem.Text = "Размер шрифта: 20";

currentFilePath = null;

loadedFileText = null;

textBox.Text = "";

textBox.Enabled = false;

saveMenuItem.Enabled = false;

sizeMenuItem.Enabled = false;

edited = false;

}

private void setSize(int size)

{

if (textBox.Font.Size == size)

{

setEdited(false);

return;

}

textBox.Font = new System.Drawing.Font(textBox.Font.FontFamily, size, FontStyle.Regular);

sizeMenuItem.Text = "Размер шрифта: " + size;

if (loadedFileTextSize != size) setEdited(true);

else setEdited(false);

}

private void size\_8\_item\_Click(object sender, EventArgs e)

{

setSize(8);

}

private void size\_12\_item\_Click(object sender, EventArgs e)

{

setSize(12);

}

private void size\_16\_item\_Click(object sender, EventArgs e)

{

setSize(16);

}

private void size\_20\_item\_Click(object sender, EventArgs e)

{

setSize(20);

}

private void size\_24\_item\_Click(object sender, EventArgs e)

{

setSize(24);

}

private void size\_26\_item\_Click(object sender, EventArgs e)

{

setSize(26);

}

}

}