Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**Иркутский национальный исследовательский технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |
|  |
|  |

|  |
| --- |
| **Отчет** |
| по лабораторной работе №3 по дисциплине «Операционные системы» |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | ЭВМб-21-1 |  |  |  | Алексеев Л.Г. |
|  |  | шифр |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Лебедева Е.Н. |
|  |  |  |  | подпись |  | И.О. Фамилия |
| Работа защищена с оценкой | | | |  | | |

Иркутск 2024 г.

**Постановка задачи.**

1.      Выберите любую пару кодировок текста

2.      Напишите файл сценария Windows Script Host, который считывает содержимое текстового файла в одной кодировке и записывает содержимое в другой кодировке в новый текстовый файл. Написать обратный скрипт.

3.      Написать html-приложение (\*.hta), которое переводит текст из одной кодировки в другую и наоборот. Для ввода текста в исходной кодировке использовать поля ввода.

4.      Написать приложение на любом выбранном языке программирования, которое также переводит текст из одной кодировки в другую. Для ввода текста использовать поля ввода.

5.      Напишите приложение для выполнения под ОС Android, которое переводит текст в одной кодировке и записывает содержимое в другой кодировке и обратную задачу. Приложение должно содержать текстовые поля для ввода, а также иметь возможность загружать текстовый файл с текстом в исходной кодировке и сохранять в текстовый файл переведенный текст на ваше мобильное устройство.

**Задание 1. Сценарий Windows Script Host**

В качестве примера выбрана пара кодировок UTF-8 и UTF-16, однако скрипт посредством диалога с пользователем позволяет выбрать и другие кодировки.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* Имя: ConverCodecs.js \*/

/\* Язык: JScript \*/

/\* Описание: Преобразование кодировок \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

var WshShell, BasePath, inputFile, outputFile, fso;

function Init() {

WshShell = WScript.CreateObject("WScript.Shell");

BasePath = WshShell.CurrentDirectory+"\\";

fso = new ActiveXObject("Scripting.FileSystemObject");

}

function CheckFile(file) {

if (fso.FileExists(file)) return true;

else {

WScript.StdOut.WriteLine("There is no such file!");

return false;

}

}

function GetFileName(needCheck) {

var file;

do {

if (needCheck) WScript.StdOut.WriteLine("Enter the name of the input file:");

else WScript.StdOut.WriteLine("Enter the name of the output file:");

file = WScript.StdIn.ReadLine();

} while (needCheck && !CheckFile(file));

return file;

}

function convert(from, to) {

var ADODBStream = WScript.CreateObject("ADODB.Stream");

// Чтение из файла в исходной кодировке

ADODBStream.Type = 2; // Текстовый режим

ADODBStream.Charset = from;

ADODBStream.Open();

ADODBStream.LoadFromFile(BasePath + inputFile);

var text = ADODBStream.ReadText();

ADODBStream.Close();

ADODBStream = WScript.CreateObject("ADODB.Stream");

ADODBStream.Type = 2; // Текстовый режим

ADODBStream.Charset = to;

ADODBStream.Open();

ADODBStream.WriteText(text);

ADODBStream.SaveToFile(BasePath + outputFile, 2);

ADODBStream.Close();

}

Init();

inputFile = GetFileName(true);

outputFile = GetFileName(false);

WScript.StdOut.WriteLine("Enter the input encoding (e.g., 'UTF-8'):");

var from = WScript.StdIn.ReadLine();

WScript.StdOut.WriteLine("Enter the output encoding (e.g., 'UTF-16'):");

var to = WScript.StdIn.ReadLine();

convert(from, to);

WScript.StdOut.WriteLine("Conversion completed");

WScript.StdIn.ReadLine();

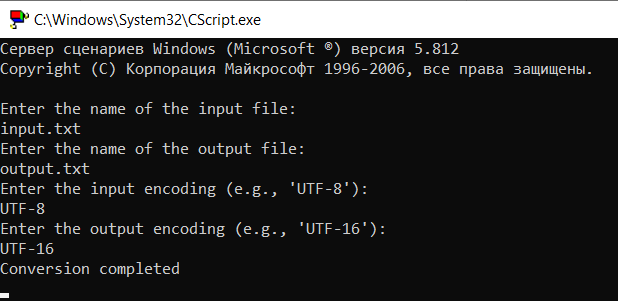


Рисунок 1 - Демонстрация работы скрипта

|  |  |
| --- | --- |
| Рисунок 2 - Исходный файл в кодировке UTF-8 | Рисунок 3 - выходной файл в кодировке UTF-16 |

В данном случае изменений в тексте не видно, но справа внизу файла указано, что теперь он записан в другой кодировке, а значит скрипт работает.

Теперь попробуем из уже созданный файл output.txt конвертировать из UTF-16 обратно в UTF-8 в файл output-back.txt

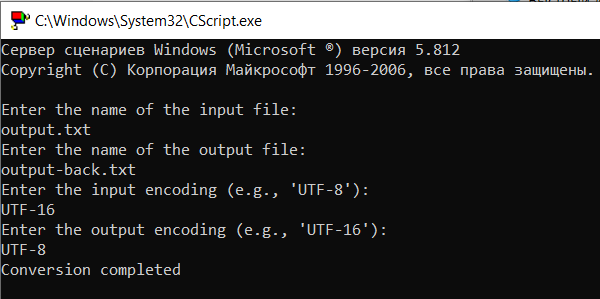


Рисунок 4 - демонстрация работы скрипта в обратную сторону

|  |  |
| --- | --- |
| Рисунок 5 - Исходный файл в кодировке UTF-16 | Рисунок 6 -- выходной файл в кодировке UTF-8 |

**Задание 2. html-приложение (HTA).**

Для ввода используется текстовое поле, из которого текст сохраняется в файл в выбранной кодировке (UTF-8 или Windows-1251).

<!DOCTYPE HTML>

<html>

<head>

<meta charset="utf-8">

<title>HTA</title>

</head>

<body>

<form name="hta">

<p>Введите текст<Br>

<textarea name="Text" cols="50" rows="5" id="in"></textarea>

</p>

<p>

<input type="button" value="Сохранить в UTF-8" onclick="convert('UTF-8')" >

<input type="button" value="Сохранить в Windows-1251" onclick="convert('Windows-1251')">

</p>

</form>

</body>

</html>

<script type="text/javascript">

window.resizeTo(640,360);

</script>

<script language="javascript">

var WshShell = new ActiveXObject("WScript.Shell");

var BasePath = WshShell.CurrentDirectory + "\\";

function convert(to) {

text = document.getElementById("in").value;

if (text.length) {

var ADODBStream = new ActiveXObject("ADODB.Stream");

ADODBStream.Type = 2; // Текстовый режим

ADODBStream.Charset = to;

ADODBStream.Open();

ADODBStream.WriteText(text);

ADODBStream.SaveToFile(BasePath + "text\_in\_" + to + ".txt", 2); // 2 - перезапись существующего файла

ADODBStream.Close();

alert("Файл сохранен как text\_in\_" + to + ".txt в папке " + BasePath);

} else {

alert("Введите текст!");

}

}

</script>

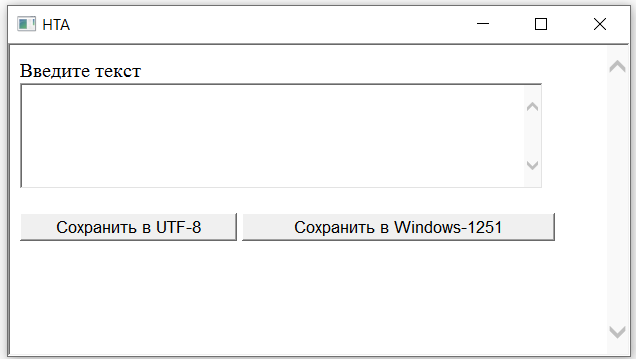


Рисунок 7 - интерфейс HTA-приложения

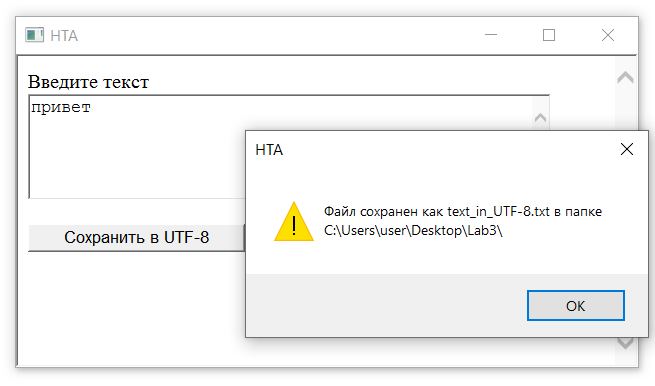


Рисунок 8 - Сохранение файла в кодировке UTF-8

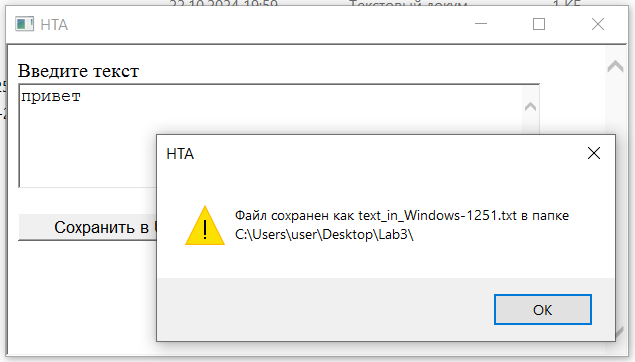


Рисунок 9 - Сохранение файла в кодировке Windows-1251

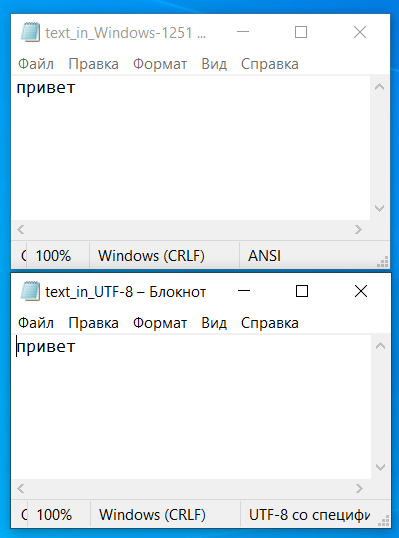


Рисунок 10 - сохраненные в выбранных кодировках файлы

**Задание 3. Приложение на языке Python**

Для ввода текста исользуется одно текстовое поле, а для вывода текста второе, также в приложении есть две кнопки с помощью которых можно перевести текст в требуемую кодировку (UTF-8 либо Windows-1251).

import sys

from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout, QTextEdit, QPushButton, QMenu

from PyQt5.QtGui import QClipboard

class PyConverter(QWidget):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("PyConverter")

self.setGeometry(100, 100, 550, 300)

layout = QVBoxLayout()

self.editor\_in = QTextEdit()

self.editor\_in.setPlaceholderText("Введите текст для преобразования...")

layout.addWidget(self.editor\_in)

self.editor\_out = QTextEdit()

self.editor\_out.setReadOnly(True)

self.editor\_out.setPlaceholderText("Результат преобразования появится здесь")

layout.addWidget(self.editor\_out)

self.btn\_to\_utf8 = QPushButton("Перевести в кодировку UTF-8")

self.btn\_to\_utf8.clicked.connect(lambda: self.convert(0))

layout.addWidget(self.btn\_to\_utf8)

self.btn\_to\_windows1251 = QPushButton("Перевести в кодировку Windows-1251")

self.btn\_to\_windows1251.clicked.connect(lambda: self.convert(1))

layout.addWidget(self.btn\_to\_windows1251)

self.setLayout(layout)

self.editor\_in.setContextMenuPolicy(3) # Включаем кастомное контекстное меню

self.editor\_in.customContextMenuRequested.connect(self.show\_context\_menu)

self.clipboard = QApplication.clipboard()

def show\_context\_menu(self, position):

menu = QMenu()

copy\_action = menu.addAction("Копировать")

paste\_action = menu.addAction("Вставить")

action = menu.exec\_(self.editor\_in.mapToGlobal(position))

if action == copy\_action:

self.editor\_in.copy()

elif action == paste\_action:

self.editor\_in.paste()

def convert(self, index):

charsets = ("UTF-8", "Windows-1251")

try:

txt = self.editor\_in.toPlainText()

converted = txt.encode(charsets[index], 'replace').decode(charsets[index - 1], 'replace')

self.editor\_out.setPlainText(converted)

except Exception as e:

self.editor\_out.setPlainText(f"Ошибка преобразования: {str(e)}")

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

converter = PyConverter()

converter.show()

sys.exit(app.exec\_())

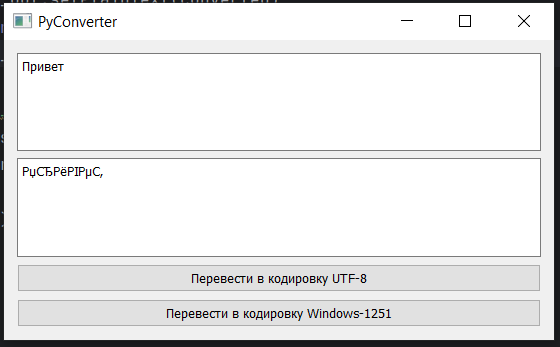


Рисунок 11 - Перевод из кодировки UTF-8 в кодировку Windows-1251

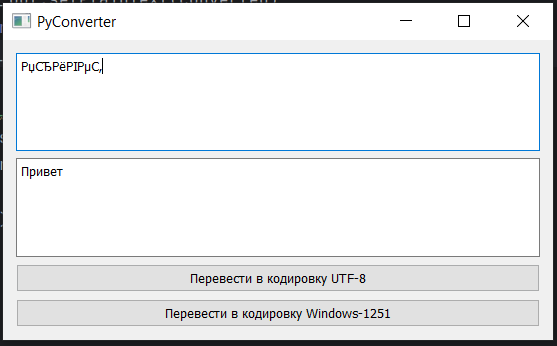


Рисунок 12 - Перевод из кодировки Windows-1251 в кодировку UTF-8

**Задание 4. Мобильное приложение.**

* Разметка
* <?xml version="1.0" encoding="utf-8"?>  
  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
   android:layout\_width="match\_parent"  
   android:layout\_height="match\_parent"  
   android:padding="16dp">  
    
   <EditText  
   android:id="@+id/inputText"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:hint="Введите текст для конвертации" />  
    
   <EditText  
   android:id="@+id/outputText"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_below="@id/inputText"  
   android:layout\_marginTop="16dp"  
   android:hint="Результат конвертации"  
   android:focusable="false" />  
    
   <Button  
   android:id="@+id/convertToUtf8Button"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_below="@id/outputText"  
   android:layout\_marginTop="16dp"  
   android:text="Конвертировать в UTF-8" />  
    
   <Button  
   android:id="@+id/convertToUtf16Button"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_below="@id/convertToUtf8Button"  
   android:layout\_marginTop="16dp"  
   android:text="Конвертировать в UTF-16" />  
    
   <Button  
   android:id="@+id/loadFileButton"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_below="@id/convertToUtf16Button"  
   android:layout\_marginTop="16dp"  
   android:text="Загрузить файл" />  
    
   <Button  
   android:id="@+id/saveFileButton"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_below="@id/loadFileButton"  
   android:layout\_marginTop="16dp"  
   android:text="Сохранить файл" />  
  </RelativeLayout>
* Код Main Activity

package com.example.encodingconverterapp  
  
import android.app.Activity  
import android.content.Intent  
import android.net.Uri  
import android.os.Bundle  
import android.provider.OpenableColumns  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import java.io.BufferedReader  
import java.io.InputStreamReader  
import java.io.OutputStreamWriter  
  
class MainActivity : AppCompatActivity() {  
  
 private lateinit var inputText: EditText  
 private lateinit var outputText: EditText  
 private val PICK\_FILE\_REQUEST\_CODE = 1  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 inputText = findViewById(R.id.*inputText*)  
 outputText = findViewById(R.id.*outputText*)  
  
 val convertToUtf8Button = findViewById<Button>(R.id.*convertToUtf8Button*)  
 val convertToUtf16Button = findViewById<Button>(R.id.*convertToUtf16Button*)  
 val loadFileButton = findViewById<Button>(R.id.*loadFileButton*)  
 val saveFileButton = findViewById<Button>(R.id.*saveFileButton*)  
  
 // Конвертация в UTF-8  
 convertToUtf8Button.setOnClickListener **{** val text = inputText.*text*.toString()  
 val convertedText = convertToUtf8(text)  
 outputText.setText(convertedText)  
 Toast.makeText(this, "Конвертация в UTF-8", Toast.*LENGTH\_SHORT*).show()  
 **}** // Конвертация в UTF-16  
 convertToUtf16Button.setOnClickListener **{** val text = inputText.*text*.toString()  
 val convertedText = convertToUtf16(text)  
 outputText.setText(convertedText)  
 Toast.makeText(this, "Конвертация в UTF-16", Toast.*LENGTH\_SHORT*).show()  
 **}** // Загрузка файла  
 loadFileButton.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_OPEN\_DOCUMENT*)  
 intent.addCategory(Intent.*CATEGORY\_OPENABLE*)  
 intent.*type* = "text/\*"  
 startActivityForResult(intent, PICK\_FILE\_REQUEST\_CODE)  
 **}** // Сохранение файла  
 saveFileButton.setOnClickListener **{** val intent = Intent(Intent.*ACTION\_CREATE\_DOCUMENT*)  
 intent.addCategory(Intent.*CATEGORY\_OPENABLE*)  
 intent.*type* = "text/plain"  
 intent.putExtra(Intent.*EXTRA\_TITLE*, "converted\_text.txt")  
 startActivityForResult(intent, PICK\_FILE\_REQUEST\_CODE + 1)  
 **}** }  
  
 override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  
 super.onActivityResult(requestCode, resultCode, data)  
 if (resultCode == Activity.*RESULT\_OK* && data != null) {  
 val uri = data.*data* when (requestCode) {  
 PICK\_FILE\_REQUEST\_CODE -> uri?.*let* **{** loadFileContent(**it**) **}** PICK\_FILE\_REQUEST\_CODE + 1 -> uri?.*let* **{** saveFileContent(**it**) **}** }  
 }  
 }  
  
 private fun loadFileContent(uri: Uri) {  
 try {  
 val inputStream = *contentResolver*.openInputStream(uri)  
 val reader = BufferedReader(InputStreamReader(inputStream, Charsets.UTF\_8))  
 val fileContent = reader.*readText*()  
 inputText.setText(fileContent)  
 reader.close()  
 Toast.makeText(this, "Файл загружен", Toast.*LENGTH\_SHORT*).show()  
 } catch (e: Exception) {  
 Toast.makeText(this, "Ошибка загрузки файла", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
  
 private fun saveFileContent(uri: Uri) {  
 try {  
 val outputStream = *contentResolver*.openOutputStream(uri)  
 val writer = OutputStreamWriter(outputStream, Charsets.UTF\_16) // Сохранение файла в UTF-16  
 writer.write(outputText.*text*.toString())  
 writer.close()  
 Toast.makeText(this, "Файл сохранен в UTF-16", Toast.*LENGTH\_SHORT*).show()  
 } catch (e: Exception) {  
 Toast.makeText(this, "Ошибка сохранения файла", Toast.*LENGTH\_SHORT*).show()  
 }  
 }  
  
 private fun convertToUtf8(text: String): String {  
 return try {  
 val utf8Bytes = text.*toByteArray*(Charsets.UTF\_8)  
 *String*(utf8Bytes, Charsets.UTF\_8)  
 } catch (e: Exception) {  
 "Ошибка конвертации в UTF-8"  
 }  
 }  
  
 private fun convertToUtf16(text: String): String {  
 return try {  
 val utf16Bytes = text.*toByteArray*(Charsets.UTF\_16)  
 *String*(utf16Bytes, Charsets.UTF\_16)  
 } catch (e: Exception) {  
 "Ошибка конвертации в UTF-16"  
 }  
 }  
}

* AndroidMainfest.xml

<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.encodingconverterapp">  
  
 <application  
 android:allowBackup="true"  
 android:icon="@android:drawable/btn\_star\_big\_on"  
 android:label="@string/app\_name"  
 android:roundIcon="@android:drawable/btn\_star\_big\_on"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.EncodingConverterApp">  
  
 <activity android:name=".MainActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

|  |  |  |
| --- | --- | --- |
| Рисунок 13 - окно Приложения | Рисунок 14 - сохранение файла | Рисунок 15 - загрузка файла |