МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное ГОСУДАРСТВЕННОЕ бюджетное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»   
  
 Факультет информатики и вычислительной техники

Кафедра информационной безопасности

Отчёт

По лабораторной работе №1

“Система контроля версийна языке С#”

По дисциплине “Программирование на языке высокого уровня”

Выполнил: студент группы БИ-31

Анисимов В.В.

Проверил: доцент кафедры ИБ Сучков Д.С.

Йошкар-Ола

2019 г

**Цель работы:** Используя объектно-ориентированный подход, разработать упрощенную систему контроля версий на языке C#.

**Описание реализованных команд**

1. Функция **Status** – обновление данных о файлах активной директории.

static int Status(string active\_path\_way)

{

if (Directory.Exists(active\_path\_way) == true)

{

string[] new\_files = Directory.GetFiles(active\_path\_way);.

foreach (string s in new\_files)

{

FileInfo fileInf = new FileInfo(s);

int count = files.FindIndex(x => string.Equals(x.name, s, StringComparison.CurrentCultureIgnoreCase));

if (count >= 0)

{

if (files[count].size != Convert.ToString(fileInf.Length) || files[count].new\_modified != Convert.ToString(fileInf.LastWriteTime))

{

files.Add(new Control\_File() { path\_way = active\_path\_way, name = files[count].name, size = files[count].size, new\_size = Convert.ToString(fileInf.Length), created = files[count].created, modified = files[count].modified, new\_modified = Convert.ToString(fileInf.LastWriteTime)});

files.RemoveAt(count);

}

}

else

{

files.Add(new Control\_File() { path\_way = active\_path\_way, name = s, size = Convert.ToString(fileInf.Length), created = Convert.ToString(fileInf.CreationTime), modified = Convert.ToString(fileInf.LastWriteTime), new\_modified = Convert.ToString(fileInf.LastWriteTime), note = "new" });

}

}

for(int i = files.Count - 1; i >= 0; i--)

{

if (files[i].path\_way == active\_path\_way)

{

if (Array.FindIndex(new\_files, x => x.Equals(files[i].name)) < 0)

{

int count = files.IndexOf(files[i]);

files.Add(new Control\_File() { path\_way = active\_path\_way, name = files[i].name, size = files[i].size, new\_size = files[i].new\_size, created = files[i].created, modified = files[i].modified, new\_modified = files[i].new\_modified, note = "delete" });

files.RemoveAt(count);

}

}

}

return 0;

}

else

{

Console.WriteLine("Активная директория исчезла.");

active\_path\_way = null;

return 1;

}

}

1. **init [dir\_path]** – инициализация СКВ для папки с файлами (или без), путь к которой указан в **dir\_path**.

if (str.IndexOf("init ") == 0)

{

str = str.Remove(0,5);

if (Directory.Exists(str) == true)

{

if (folders.Exists(x => string.Equals(x.path\_way, str, StringComparison.CurrentCultureIgnoreCase)) != true)

{

folders.Add(new Control\_Folder() { path\_way = str });

active\_path\_way = str;

string[] new\_files = Directory.GetFiles(str);

foreach (string s in new\_files)

{

FileInfo fileInf = new FileInfo(s);

files.Add(new Control\_File() { path\_way = str, name = s, size = Convert.ToString(fileInf.Length), created = Convert.ToString(fileInf.CreationTime), modified = Convert.ToString(fileInf.LastWriteTime), new\_modified = Convert.ToString(fileInf.LastWriteTime) });

}

Console.WriteLine("Директория инициализирована.");

}

else

Console.WriteLine("Папка с введенным путём уже проинициализирована.");

}

else

Console.WriteLine("Директория с введенным путём не существует.");

Write(active\_path\_way);

goto link;

}

1. **status** – отображение статуса отслеживаемых файлов последней проинициализированной папки (какие файлы отслеживаются, краткая информация по ним) Note: красным выделяется измененный файл, зеленым, соответственно, нет.

if (str == "status")

{

if (active\_path\_way != null)

{

if (Status(active\_path\_way) == 0)

{

Console.WriteLine("Directory: {0}\\", active\_path\_way);

int mark = 0;

foreach (Control\_File u in files)

{

if (active\_path\_way == u.path\_way)

{

if (u.new\_size==null || u.modified == u.new\_modified)

Console.ForegroundColor = ConsoleColor.Green;

else

Console.ForegroundColor = ConsoleColor.Red;

if (String.IsNullOrEmpty(u.note))

Console.WriteLine("file: {0}", u.name.Replace(u.path\_way + "\\", ""));

else

Console.WriteLine("file: {0} <<-- {1}", u.name.Replace(u.path\_way + "\\",""), u.note);

int size = Convert.ToInt32(u.size);

string name\_size = "Byte";

if ((size / 1024) > 1)

{

size /= 1024;

name\_size = "Kb";

if ((size / 1024) > 1)

{

size /= 1024;

name\_size = "Mb";

}

}

if (!String.IsNullOrEmpty(u.new\_size))

{

int new\_size = Convert.ToInt32(u.new\_size);

string name\_new\_size = "Byte";

if ((new\_size / 1024) > 1)

{

new\_size /= 1024;

name\_new\_size = "Kb";

if ((new\_size / 1024) > 1)

{

new\_size /= 1024;

name\_new\_size = "Mb";

}

}

Console.WriteLine(" size: {0} {1} <<-- {2} {3}", size, name\_size, new\_size, name\_new\_size);

}

else

{

Console.WriteLine(" size: {0} {1}", size, name\_size);

}

Console.WriteLine(" created: {0}", u.created);

Console.WriteLine(" modified: {0}\n", u.new\_modified);

mark++;

}

}

Console.ResetColor();

if (mark == 0)

Console.WriteLine("Нет в файлов в активной директории.");

}

else

{

Console.WriteLine("Активная директория исчезла.");

active\_path\_way = null;

}

}

else

Console.WriteLine("Не выбрана активная директория(используйте команду checkout).");

Write(active\_path\_way);

goto link;

}

1. **add [file\_path] –** добавить файл под версионный контроль.

if (str.IndexOf("add ") == 0)

{

str = str.Remove(0, 4);

if (active\_path\_way != null)

{

Status(active\_path\_way);

if (File.Exists(active\_path\_way + "\\" + str) == true)

{

int count = files.FindIndex(s => string.Equals(s.name, active\_path\_way + "\\" + str, StringComparison.CurrentCultureIgnoreCase));

if (count >= 0)

{

files.RemoveAt(count);

FileInfo fileInf = new FileInfo(active\_path\_way + "\\" + str);

files.Add(new Control\_File() { path\_way = active\_path\_way, name = active\_path\_way + "\\" + str, size = Convert.ToString(fileInf.Length), created = Convert.ToString(fileInf.CreationTime), modified = Convert.ToString(fileInf.LastWriteTime), new\_modified = Convert.ToString(fileInf.LastWriteTime), note = "added" });

Console.WriteLine("Файл {0} добавлен под версионный контроль.", str);

}

else

Console.WriteLine("Файл не удалось добавить");

}

else

Console.WriteLine("Файл с таким названием в директории не существует.");

}

else

Console.WriteLine("Не выбрана активная директория(команда checkout).");

Write(active\_path\_way);

goto link;

}

1. **remove [file\_path] –** удалить файл из-под версионного контроля.

if (str.IndexOf("remove ") == 0)

{

str = str.Remove(0, 7);

if (active\_path\_way != null)

{

Status(active\_path\_way);

if (File.Exists(active\_path\_way + "\\" + str) == true)

{

int count = files.FindIndex(s => string.Equals(s.name, active\_path\_way + "\\" + str, StringComparison.CurrentCultureIgnoreCase));

string mod =files[count].modified;

files.RemoveAt(count);

FileInfo fileInf = new FileInfo(active\_path\_way + "\\" + str);

files.Add(new Control\_File() { path\_way = active\_path\_way, name = active\_path\_way + "\\" + str, size = Convert.ToString(fileInf.Length), created = Convert.ToString(fileInf.CreationTime), modified = mod, new\_modified = Convert.ToString(fileInf.LastWriteTime), note = "removed" });

Console.WriteLine("Файл {0} убран из под версионного контроля.", str);

}

else

Console.WriteLine("Файл с таким названием не существует.");

Write(active\_path\_way);

}

else

Console.WriteLine("Не выбрана активная директория(команда checkout).");

goto link;

}

1. **apply [dir\_path] –** сохранить все изменения в отслеживаемой папке (удалить все метки к файлам и сохранить изменения в них).

if (str.IndexOf("apply ") == 0)

{

str = str.Remove(0, 6);

if (Directory.Exists(str) == true)

{

if (folders.Exists(x => x.path\_way == str ) == true)

{

for(int i = files.Count()-1; i>=0; i--)

{

if (files[i].path\_way == str)

{

if(files[i].note != "delete")

{

string size = !String.IsNullOrEmpty(files[i].new\_size) ? files[i].new\_size : files[i].size;

files.Add(new Control\_File() { path\_way = str, name = files[i].name, size = size, created = files[i].created, modified = files[i].modified, new\_modified = files[i].modified, note = "" });

}

files.Remove(files[i]);

}

}

Console.WriteLine("Все изменения в данной директории сохранены.");

Write(active\_path\_way);

}

else

Console.WriteLine("Данная директория не отслеживается (для добавления используйте команду init).");

}

else

Console.WriteLine("Директория с введенным путём не существует.");

goto link;

}

1. **list branch -**  показать все отслеживаемые папки.

if (str == "list branch")

{

if (folders.Count > 0)

{

Console.WriteLine("Отслеживаемые директории:");

foreach (Control\_Folder u in folders)

{

Console.WriteLine(u.path\_way);

System.Diagnostics.Process.Start(u.path\_way);

}

}

else

Console.WriteLine("Отслеживаемые директории отсутствуют.");

goto link;

}

1. **checkout [dir\_path] OR [dir\_number] –** перейти к указанной отслеживаемой директории.

if (str.IndexOf("checkout ") == 0)

{

str = str.Remove(0, 9);

if (Directory.Exists(str) == true)

{

if (folders.Exists(x => x.path\_way == str) == true)

{

active\_path\_way = str;

Console.WriteLine("Директория {0} активна", active\_path\_way);

}

else

Console.WriteLine("Данная директория не отслеживается (для добавления используйте команду init).");

}

else

Console.WriteLine("Директория с введенным путём не существует.");

goto link;

}

if (Directory.Exists(str) == true)

{

if (Directory.Exists(str) == true)

{

if (folders.Exists(x => x.path\_way == str) == true)

{

active\_path\_way = str;

Console.WriteLine("Директория {0} активна", active\_path\_way);

}

else

Console.WriteLine("Данная директория не отслеживается (для добавления используйте команду init).");

}

else

Console.WriteLine("Директория с введенным путём не существует.");

goto link;

}

**Вывод:** В ходе работы была реализована упрощенная система контроля версий на языке C#.