Лабораторная работа №1.

# Создание процесса

## Цель работы

Научиться создавать процессы средствами .NET.

## Задания

## Вариант №1.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, откройте для редактирования документ Microsoft Word и дождитесь окончания его редактирования.

## Вариант №2.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, откройте для редактирования таблицу Microsoft Excel и дождитесь окончания её редактирования.

## Вариант №3.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, покажите на консоли содержимое текущего каталога, после чего выведите сообщение об окончании исполнения задания.

## Вариант №4.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, выполните указанный первым параметром командный файл, передав ему все последующие параметры, после чего выведите сообщение об окончании исполнения задания.

## Вариант №5.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, выполните указанный первым параметром .js-файл, передав ему все последующие параметры, после чего выведите сообщение об окончании исполнения задания.

## Вариант №6.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, выполните указанный первым параметром .vbs-файл, передав ему все последующие параметры, после чего выведите сообщение об окончании исполнения задания.

## Вариант №7.

Используя класс Process (System.Diagnostics) и его свойство StartInfo, откройте указанный первым параметром .htm-файл в браузере Internet Explorer, дождитесь окончания просмотра, после чего выведите сообщение об окончании исполнения задания.

## Теоретическая часть

Для создания процесса используется класс Process из пространства имён System.Diagnostics. Сам процесс создаётся непосредственно при вызове метода Start, который имеет 5 перегруженных статических вариантов его использования и 1 вариант метода экземпляра. Самая простая перегрузка не имеет аргументов и во всём полагается на поля свойства StartInfo, которое мы рассмотрим поподробнее.

Свойство **UseShellExecute** позволяет «запускать» не бинарные файлы, такие как doc, xls, js, vbs, cmd и другие. При этом при запуске процесс создаётся посредством WinAPI функции ShellExecute. Недостатком использования данного свойства является невозможность перенаправления потоков ввода, вывода и ошибок.

Свойство **FileName** определяет имя бинарного файла для запуска при **UseShellExecute**= **false** либо имя открываемого файла при **UseShellExecute**= **true**.

Строковое свойство **Arguments** используется для задания дополнительных параметров запускаемому процессу.

Для запуска процесса от лица другого пользователя используются свойства **UserName** и **Password**. При этом, рекомендуется задавать также свойство **WorkingDirectory**, которое по умолчанию буде принято за %SYSTEMROOT%\system32.

Если процесс успешно создан, то метод (экземпляра) возвращает значение **true** и заполняет необходимые поля экземпляра. Статические методы в случае успеха, возвращают новый экземпляр класса **Process** с заполненными полями. В противном случае они возвращают значение **null**.

## Пример

using System;

using System.Diagnostics;

public class Пример

{

public static void Main(string[] args)

{

if (args.Length!=1)

{

Console.WriteLine("Использование:");

Console.WriteLine(string.Format("\t{0} [команда]",

System.Reflection.Assembly.GetEntryAssembly().Location

));

return;

}

Process myProcess = new Process();

try

{

myProcess.StartInfo.UseShellExecute = false;

myProcess.StartInfo.FileName = args[0];

myProcess.Start();

Console.WriteLine(

"Дочерний процесс запущен. Ждём его окончания...");

// Ждём окончания дочернего процесса.

myProcess.WaitForExit();

Console.WriteLine("Дочерний процесс закончен.");

} catch (Exception e) {

Console.WriteLine(e.Message);

}

}

}