console.cm

```
/*
    ========================================================================

    Copyright (c) 2012-2016 Seppo Laakko
    http://sourceforge.net/projects/cmajor/

    Distributed under the GNU General Public License, version 3 (GPLv3).
    (See accompanying LICENSE.txt or http://www.gnu.org/licenses/gpl.html
        )

    ========================================================================
    */

// Copyright (c) 1994
// Hewlett-Packard Company
// Copyright (c) 1996
// Silicon Graphics Computer Systems, Inc.
// Copyright (c) 2009 Alexander Stepanov and Paul McJones

using System.IO;

namespace System
{
    public static class Console
    {
        static nothrow Console(): in(new InputFileStream()), out(new
            OutputFileStream(stdout)), err(new OutputFileStream(stderr))
        {
        }
        public static string ReadLine()
        {
            return in->ReadLine();
        }
        public static string ReadToEnd()
        {
            return in->ReadToEnd();
        }
        public static void Write(const char* s)
        {
            out->Write(s);
        }
        public static void Write(const string& s)
        {
            out->Write(s);
        }
        public static void Write(const wstring& s)
        {
            out->Write(s);
        }
        public static void Write(const ustring& s)
        {
```

```
        out->Write(s);
    }
    public static void Write(char c)
    {
        out->Write(c);
    }
    public static void Write(wchar c)
    {
        out->Write(c);
    }
    public static void Write(uchar c)
    {
        out->Write(c);
    }
    public static void Write(byte b)
    {
        out->Write(b);
    }
    public static void Write(sbyte s)
    {
        out->Write(s);
    }
    public static void Write(short s)
    {
        out->Write(s);
    }
    public static void Write(ushort u)
    {
        out->Write(u);
    }
    public static void Write(int i)
    {
        out->Write(i);
    }
    public static void Write(uint u)
    {
        out->Write(u);
    }
    public static void Write(long l)
    {
        out->Write(l);
    }
    public static void Write(ulong u)
    {
        out->Write(u);
    }
    public static void Write(bool b)
    {
        out->Write(b);
    }
    public static void Write(float f)
    {
        out->Write(f);
```

```
    }
    public static void Write(double d)
    {
        out->Write(d);
    }
    public static void WriteLine()
    {
        out->WriteLine();
    }
    public static void WriteLine(const char* s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(const wchar* s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(const uchar* s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(const string& s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(const wstring& s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(const ustring& s)
    {
        out->WriteLine(s);
    }
    public static void WriteLine(char c)
    {
        out->WriteLine(c);
    }
    public static void WriteLine(wchar c)
    {
        out->WriteLine(c);
    }
    public static void WriteLine(uchar c)
    {
        out->WriteLine(c);
    }
    public static void WriteLine(byte b)
    {
        out->WriteLine(b);
    }
    public static void WriteLine(sbyte s)
    {
        out->WriteLine(s);
    }
```

```
public static void WriteLine(short s)
{
    out->WriteLine(s);
}
public static void WriteLine(ushort u)
{
    out->WriteLine(u);
}
public static void WriteLine(int i)
{
    out->WriteLine(i);
}
public static void WriteLine(uint u)
{
    out->WriteLine(u);
}
public static void WriteLine(long l)
{
    out->WriteLine(l);
}
public static void WriteLine(ulong u)
{
    out->WriteLine(u);
}
public static void WriteLine(bool b)
{
    out->WriteLine(b);
}
public static void WriteLine(float f)
{
    out->WriteLine(f);
}
public static void WriteLine(double d)
{
    out->WriteLine(d);
}
public static InputStream& In()
{
    return *in;
}
public static OutputStream& Out()
{
    return *out;
}
public static OutputStream& Error()
{
    return *err;
}
public static void SetIn(UniquePtr<InputStream>&& in_)
{
    in = in_;
}
public static void SetOut(UniquePtr<OutputStream>&& out_)
```

```
        {
            out = out_;
        }
        public static void SetError(UniquePtr<OutputStream>&& err_)
        {
            err = err_;
        }
        private static UniquePtr<InputStream> in;
        private static UniquePtr<OutputStream> out;
        private static UniquePtr<OutputStream> err;
    }
}
```