

## charclass.cm

```
/*  
  
    Copyright (c) 2012–2015 Seppo Laakko  
    http://sourceforge.net/projects/cmajors/  
  
    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.Collections;  
  
namespace System  
{  
    public enum CharClass  
    {  
        none = 0,  
        lower = 1,  
        upper = 2,  
        alpha = lower | upper,  
        digit = 4,  
        alnum = alpha | digit,  
        xdigit = 8,  
        cntrl = 16,  
        graph = 32,  
        print = 64,  
        punct = 128,  
        space = 256  
    }  
  
    public const int EXIT_CHAR_CLASS_TABLE_ALLOCATE = 249;  
  
    public static class CharClassTable  
    {  
        static nothrow CharClassTable()  
        {  
            try  
            {  
                table.Resize(256);  
            }  
            catch (const Exception& ex)  
            {  

```

```

        exit(EXIT.CHAR.CLASS.TABLE.ALLOCATE);
    }
    SetCharacterClass(cast<byte>('a'), cast<byte>('z'), CharClass
        .lower);
    SetCharacterClass(cast<byte>('A'), cast<byte>('Z'), CharClass
        .upper);
    SetCharacterClass(cast<byte>('0'), cast<byte>('9'), CharClass
        .digit);
    SetCharacterClass(cast<byte>('0'), cast<byte>('9'), CharClass
        .xdigit);
    SetCharacterClass(cast<byte>('a'), cast<byte>('f'), CharClass
        .xdigit);
    SetCharacterClass(cast<byte>('A'), cast<byte>('F'), CharClass
        .xdigit);
    SetCharacterClass(0u, 31u, CharClass.cntrl);
    SetCharacterClass(127u, 127u, CharClass.cntrl);
    SetCharacterClass(33u, 126u, CharClass.graph);
    SetCharacterClass(32u, 126u, CharClass.print);
    SetCharacterClass(33u, 47u, CharClass.punct);
    SetCharacterClass(58u, 64u, CharClass.punct);
    SetCharacterClass(91u, 96u, CharClass.punct);
    SetCharacterClass(123u, 126u, CharClass.punct);
    SetCharacterClass(9u, 13u, CharClass.space);
    SetCharacterClass(32u, 32u, CharClass.space);
}
public static nothrow CharClass GetCharacterClass(char c)
{
    return table[cast<byte>(c)];
}
private static nothrow void SetCharacterClass(byte first, byte
    last, CharClass c)
{
    for (byte i = first; i <= last; ++i)
    {
        table[i] = cast<CharClass>(table[i] | c);
    }
}
private static List<CharClass> table;
}

public inline nothrow bool IsLower(char c)
{
    return (CharClassTable.GetCharacterClass(c) & CharClass.lower) !=
        0;
}

public inline nothrow bool IsUpper(char c)
{
    return (CharClassTable.GetCharacterClass(c) & CharClass.upper) !=
        0;
}

public inline nothrow bool IsAlpha(char c)

```

```

    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.alpha) !=
            0;
    }

    public inline nothrow bool IsDigit(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.digit) !=
            0;
    }

    public inline nothrow bool IsAlphanumeric(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.alnum) !=
            0;
    }

    public inline nothrow bool IsHexDigit(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.xdigit)
            != 0;
    }

    public inline nothrow bool IsControl(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.cntrl) !=
            0;
    }

    public inline nothrow bool IsGraphic(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.graph) !=
            0;
    }

    public inline nothrow bool IsPrintable(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.print) !=
            0;
    }

    public inline nothrow bool IsPunctuation(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.punct) !=
            0;
    }

    public inline nothrow bool IsSpace(char c)
    {
        return (CharClassTable.GetCharacterClass(c) & CharClass.space) !=
            0;
    }
}

```

