using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Eyespy\_Core\_v1

{

/// <summary>

/// This stack data structure is used to create a stack frame of size x within which each current-context URL is stored.

/// This class will be dynamically-instantiated by the Systray class.

/// </summary>

/// <remarks></remarks>

public class Stack

{

/// <summary>

/// Maximum size of the stack before it needs resetting. this variable is assigned in the constructor by the 'size' parameter.

/// </summary>

/// <remarks></remarks>

private int stackSize;

/// <summary>

/// Like in a real-world stack, a stack pointed variable is required. this integer hold the index of the last added item. (LIFO principle)

/// </summary>

/// <remarks></remarks>

private int stackPointer;

/// <summary>

/// Number of times has the stack reached its maximum size and been reset - used for the 'About' context-menu item.

/// </summary>

/// <remarks></remarks>

public int stackFrameCount;

/// <summary>

/// Each member of this array is a URL corresponding to an item of the stack, where the index is the address.

/// </summary>

/// <remarks></remarks>

private string[] stackItems;

/// <summary>

/// Constructor for the class inside which are set the class-member variables.</see> namespace.

/// </summary>

/// <param name="size">The maximum size of the stack before it is reset.</param>

/// <remarks></remarks>

public Stack(int size)

{

stackSize = size;

stackPointer = 1;

stackItems = new string[size];

}

/// <summary>

/// Check if a URL is defined within the current stack frame.

/// </summary>

/// <param name="stackItem">The URL to be checked.</param>

/// <returns>Returns true if the URL is found within the stack frame, false otherwise.</returns>

/// <remarks></remarks>

public bool isMemorized(string stackItem)

{

int loc = Array.IndexOf(stackItems, stackItem);

if (loc > -1)

return true;

else

{

return false;

}

}

/// <summary>

/// Add a new item to the stack frame and ensure the stack is not in overflow (and reset it if it's the case).

/// </summary>

/// <param name="stackItem">URL to be pushed to the top of the stack.</param>

/// <remarks></remarks>

public void push(string stackItem)

{

if (stackPointer == stackSize)

{

stackPointer = 1;

Array.Clear(stackItems, 1, stackSize);

push(stackItem);

stackFrameCount++;

}

else

{

stackItems[stackPointer] = stackItem;

stackPointer++;

}

}

}

}