Invoking methods of a Dynamic Link Library (DLL) built with C# in Blue Prism

Smart Process Automation

Scott Ivinza

Cognitive Automation Technical Consultant at BrightKnight (Belfius Group)

scott.ivinza@brightknight.eu

Prerequisite

In this article, we provide to Blue Prism developers with a library containing methods already implemented in C# (see appendix). The implementation of that library is a **blackbox**. It doesn't matter if you don't have any programming background. You just need to use the object actions based on their specification (preconditions & postconditions).

Context

Given that the native functions provided by Blue Prism are not exhaustive. When it comes to develop more advanced processes that make use of powerful and efficient functions, Blue Prism developers can therefore feel limited by the scope of the tools they have.

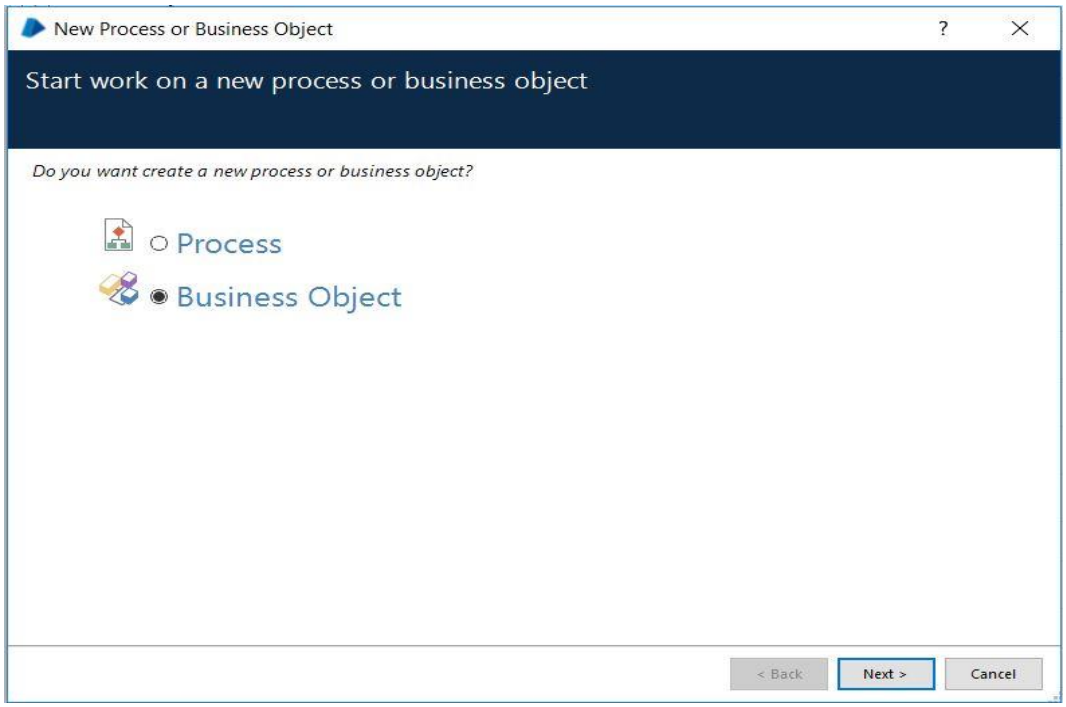
That's why, we can create C# components called **Dynamic Link Libraries** (DLLs) that we can embed to Blue Prism in order to create added value functions to integrate in Blue Prism objects. Then, use these objects inside processes to implement.

In this article, firstly we are going to focus on how to call methods of a DLL built with C# in Blue Prism. Secondly, we will discuss some **Visual Business Objects** (VBOs) that may require external C# components in order to achieve advanced processes with Blue Prism.

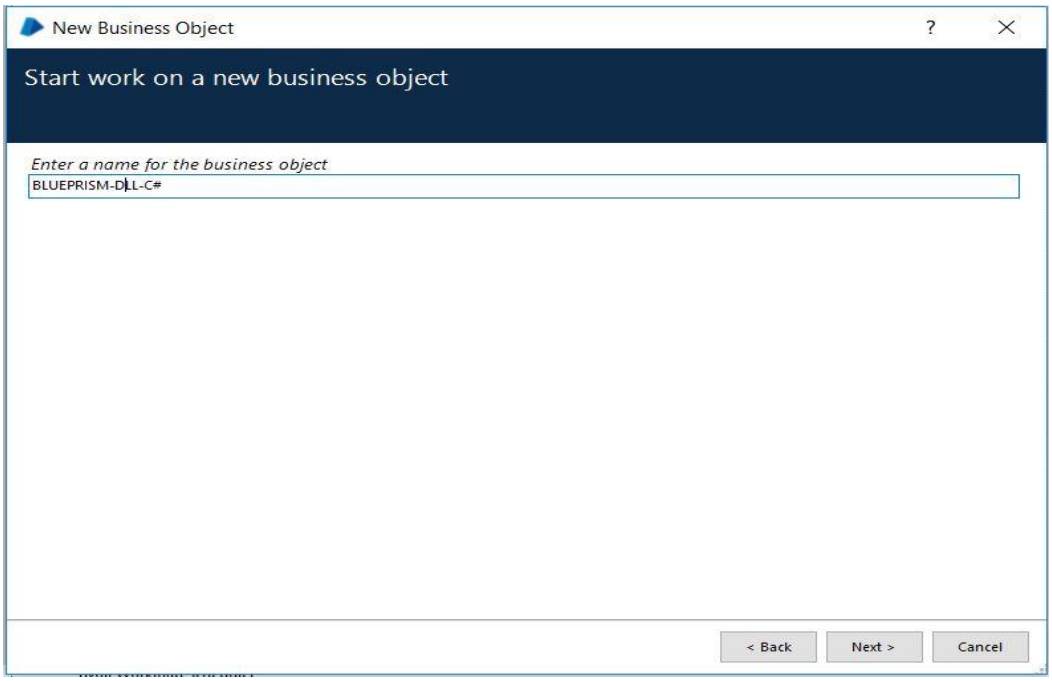
Calling the methods of a DLL in Blue Prism

In order to invoke static methods of the library (DLL) within Blue Prism, we proceed as follows:

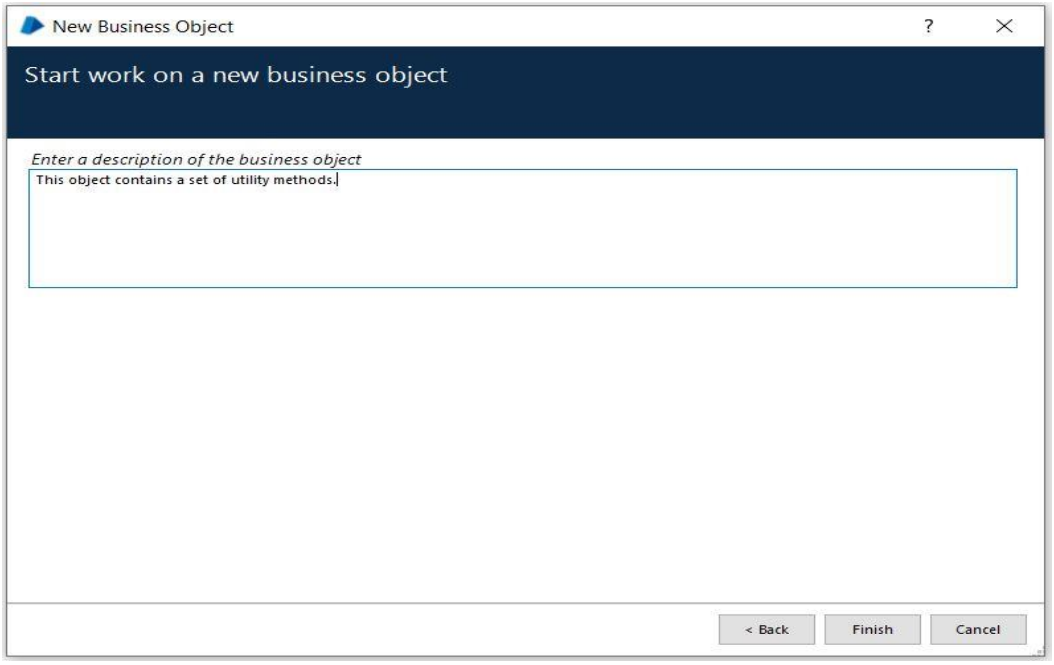
1. We need to create an object in Blue Prism. To create a new object, go to File tab, click New and choose Business Object as follows:



2. After pressing next, we can enter the name of the object as follows:

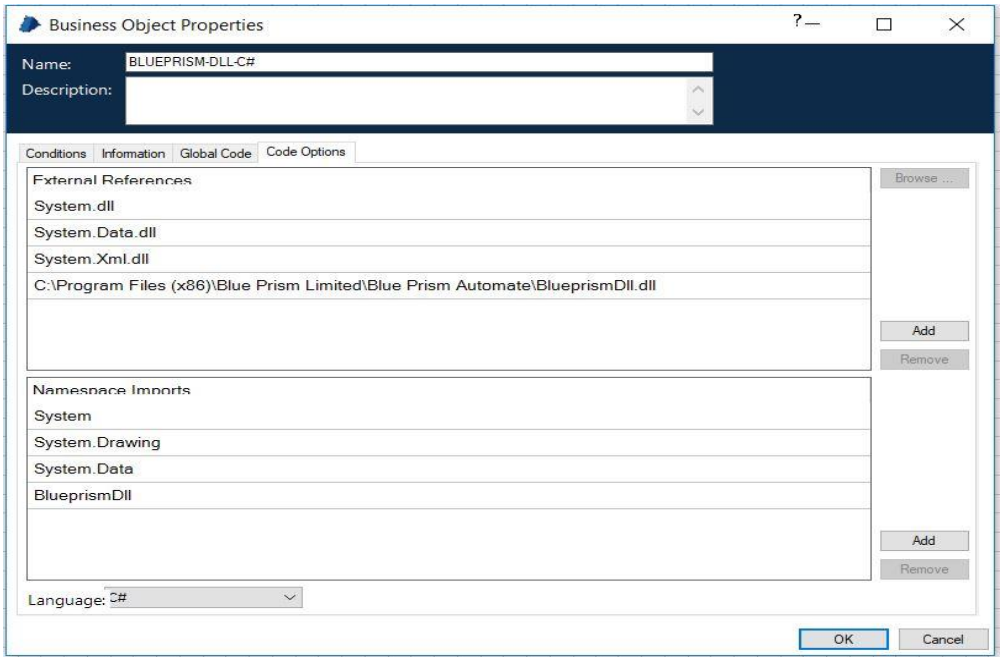


3. After pressing next again, we can enter the description of the object and pressing Finish to create the object as shown below:

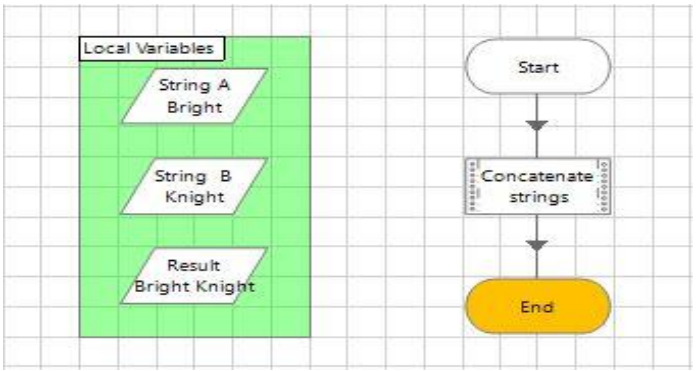


4. On the Initialise page of the object created, let's open up the Business Object Properties window and go to the tab Code Options. Below the title "External References", we can add the path towards our dll file. In our case the path towards the dll file is **C:\Program Files (x86)\Blue Prism Limited\Blue Prism Automate\BlueprismDll.dll**. We can notice that it's important to put the dll file inside the folder Blue Prism Automate. Otherwise, the code in the object can be compiled without errors but the execution can fail.

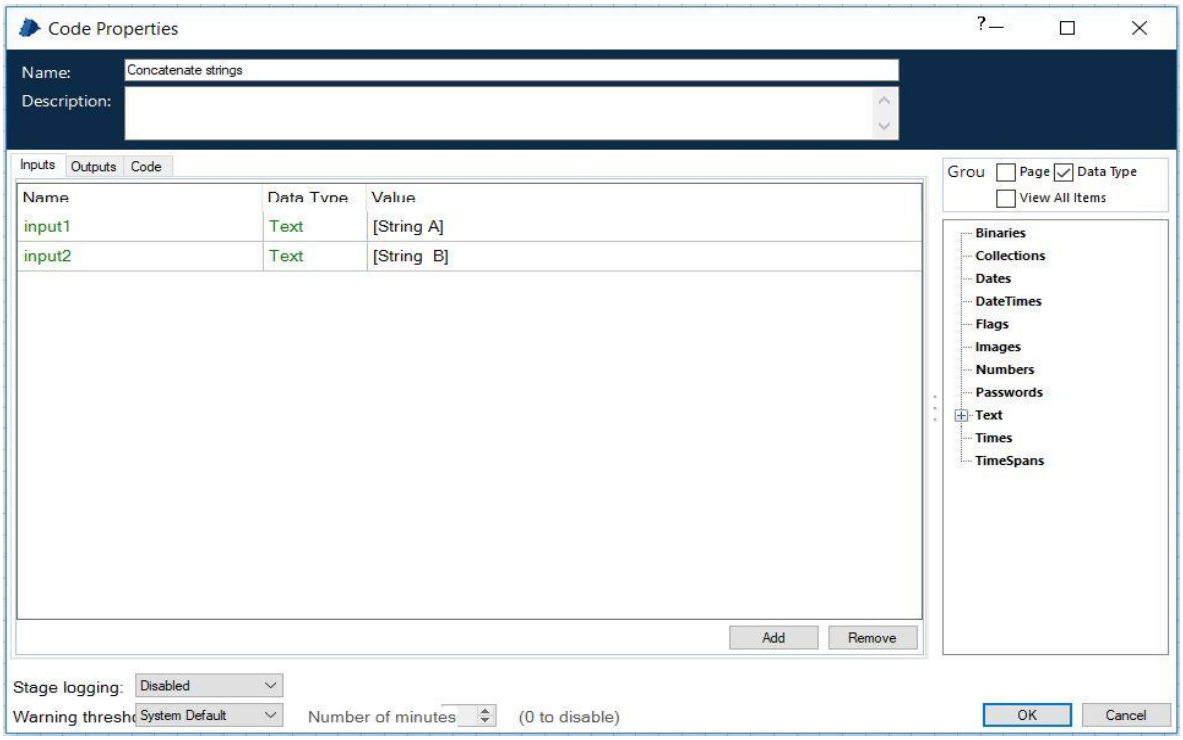
Then, below the title "Namespace Imports" we can import the namespace which is a container of the class containing the methods to use. Our namespace is **BlueprismDll**. Moreover, the choice of the programming language must be **C#** because it's the one we are going to use to call functions. The setup described will look like on the screenshot below:



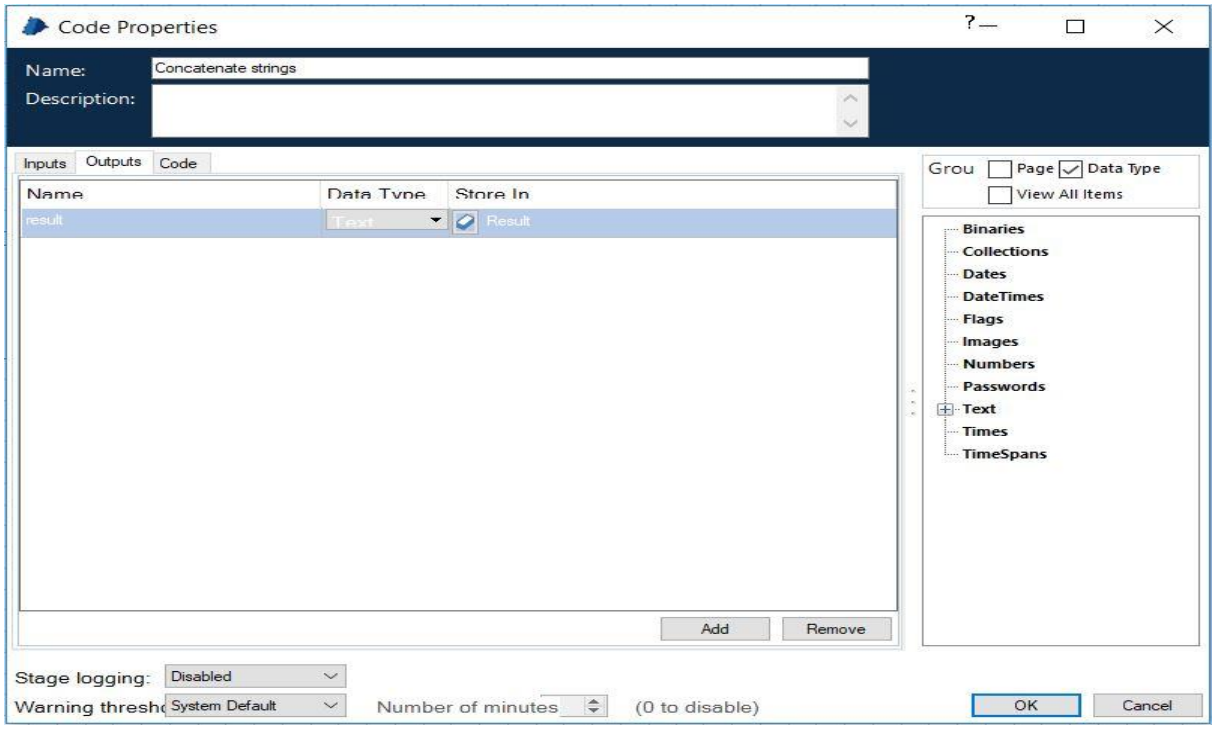
5. Now we can create a new page called "Concatenate Strings" in the object. On this page, we create a code stage to invoke the method **ConcatenateStrings** of the library BlueprismDll and also create three data items of Text data type. The first one is **String A** and contains the initial value **Bright**. The second one is **String B** and contains the initial value **Knight**. The last one is **Result**, his value is the concatenation of strings. The process is illustrated on the screenshot below:



By double clicking on the code stage, we open up Code Properties window on which we can define input arguments of the function as shown on the screenshot below:

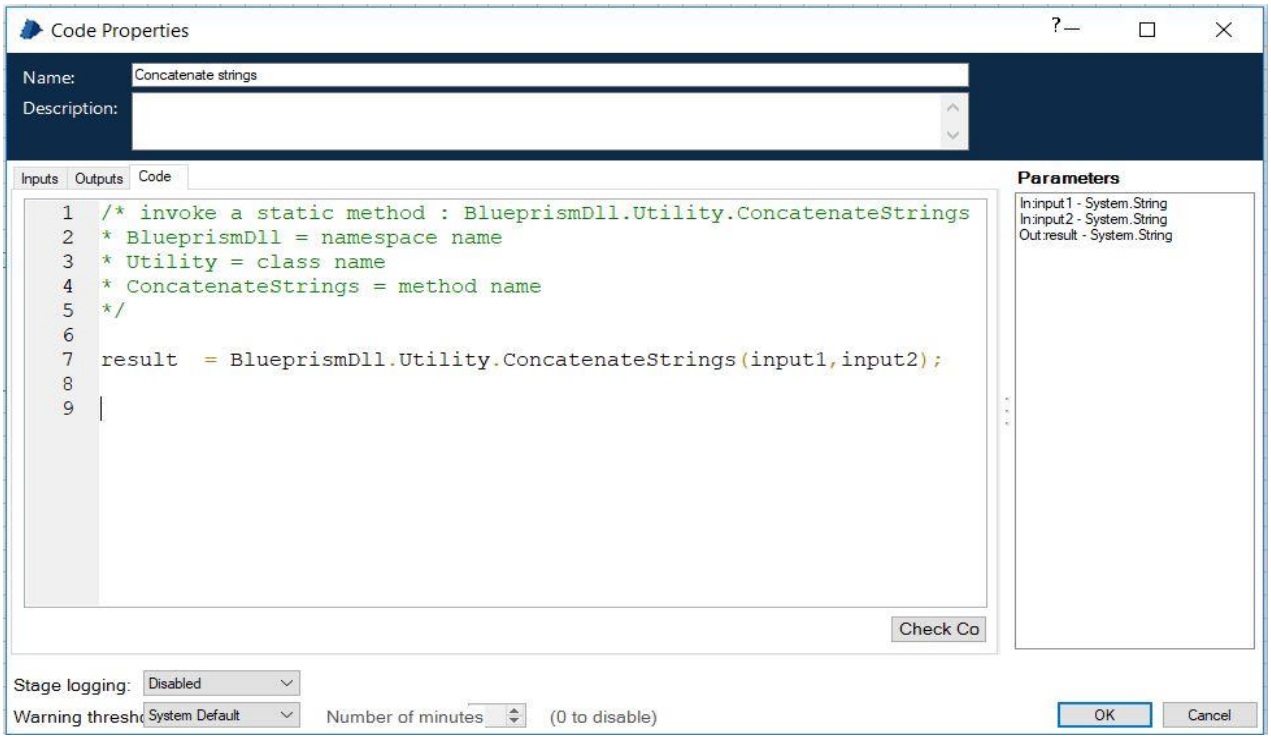


Similarly, on the outputs tab we define the output parameter as shown on the screenshot below:



Once all input and output arguments are defined, the call of the method ConcatenateStrings is performed on the Code tab. On the code editor, the C# program manipulates parameter names that we have defined that is **input1**, **input2** and **result**.

To invoke a static method in C#, we write the line **BlueprismDll.Utility.ConcatenateStrings (….,….)** where BlueprismDll is the namespace, Utility is the static class containing static methods and ConcatenateStrings is the static method to invoke. The C# code of the method call is shown below:



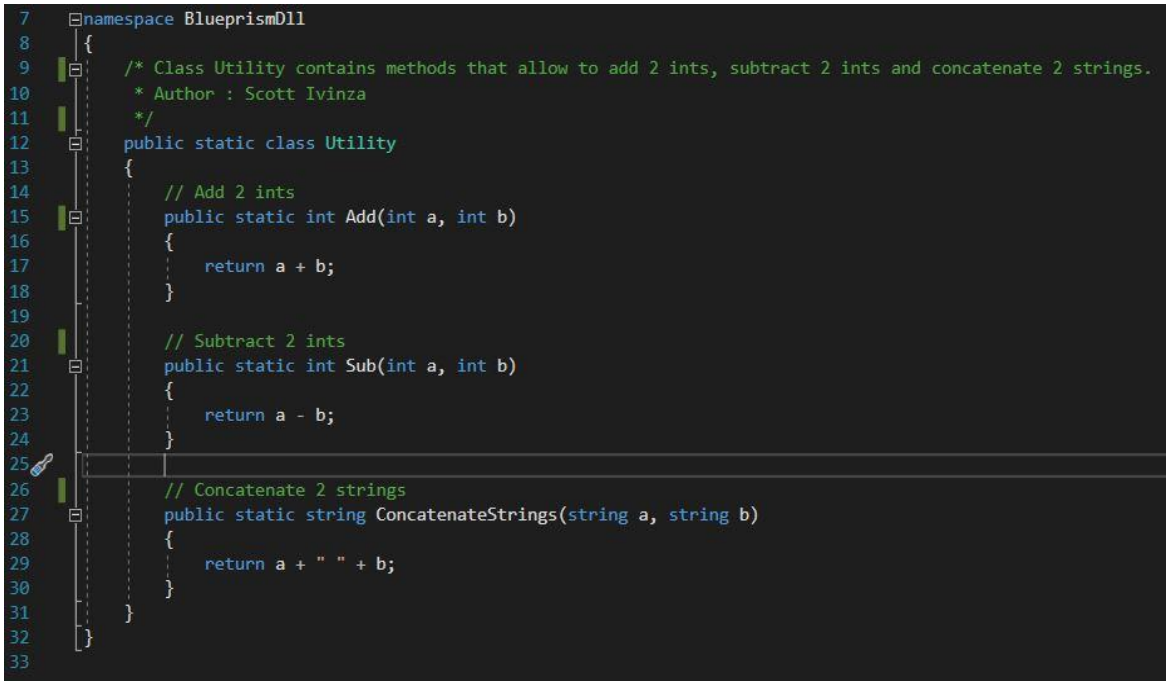
Conclusion

We can extend Blue Prism native objects such as Utility-Strings, Utility-File Management and MS Excel VBO with new custom actions using C#. For instance, we can build the following components:

- **String Manipulation Utilities** VBO can provide advanced functionalities for parsing, ordering and extracting a sequence of characters from a given text.
- **Regex Utilities** VBO can provide advanced functionalities for performing a search for text matches using regular expressions.
- **MS Excel Utilities** VBO can provide custom Excel functionalities to perform calculations on rows and columns.

Appendix

The implementation of the dynamic link library provided in this article is described in the following C# code sample.



```
namespace BlueprismDll
{
    /* Class Utility contains methods that allow to add 2 ints, subtract 2 ints and concatenate 2 strings.
     * Author : Scott Ivinza
     */
    public static class Utility
    {
        // Add 2 ints
        public static int Add(int a, int b)
        {
            return a + b;
        }

        // Subtract 2 ints
        public static int Sub(int a, int b)
        {
            return a - b;
        }

        // Concatenate 2 strings
        public static string ConcatenateStrings(string a, string b)
        {
            return a + " " + b;
        }
    }
}
```