Using 3GL with Uniface on Windows

Thomas S. Shore III June 3, 2015

This article will discuss and demonstrate the construction and integration of C on a Windows platform with Uniface 9.7. For this example we will call a DLL that has an implementation of the Metaphone phonetic algorithm for indexing words by their English pronunciation. (See http://en.wikipedia.org/wiki/Metaphone)

Introduction

The topics covered in this article are:

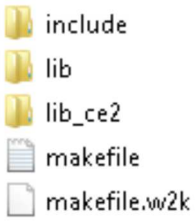
- Visual Studio 2008 construction of a DLL used with Uniface
- Configuring Uniface to call a DLL
- Entering a C signature into Uniface
- Creating a test form to show the integration

All of the files necessary to develop this exercise are included in the distribution. You will need access to the Visual Studio environment and its utilities, which is **NOT** included. This article assumes that the example files are installed in 'E:\Projects\3GL\Metaphone' adjust the instructions accordingly based upon your environment.

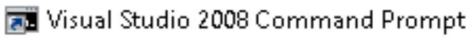
Part I: Creating a simple DLL in Visual Studio 2008

We will not be creating a project per se in Visual Studio, rather we will utilize the C/C++ compiler to build a DLL that can be integrated with Uniface.

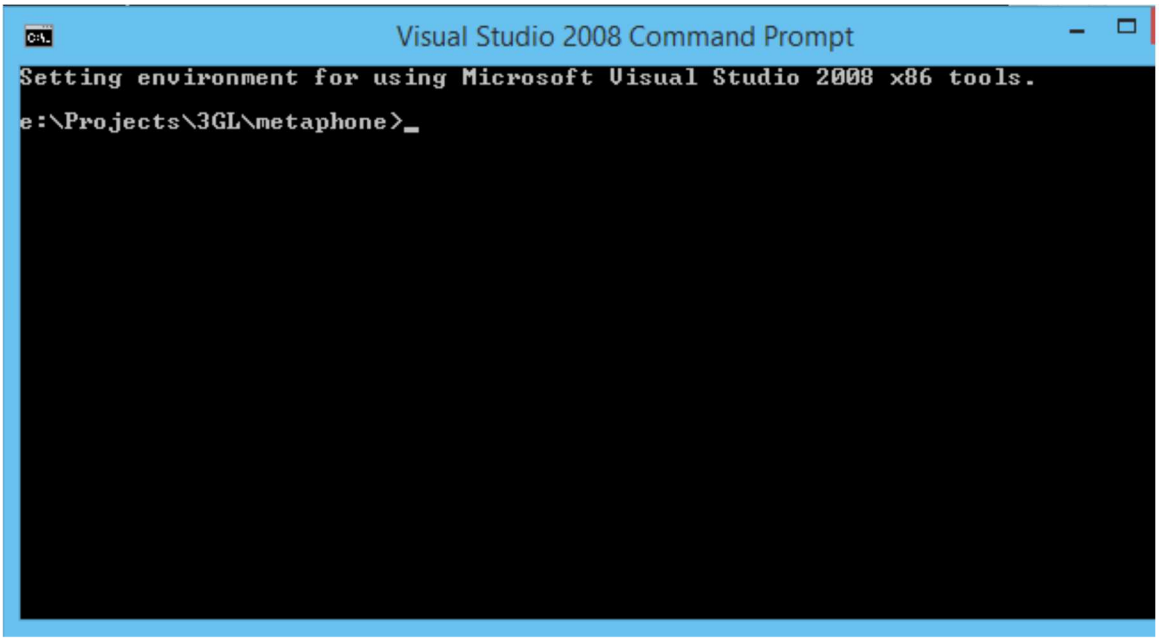
1. Ensure that you have access to the UNIFACE 3gl directory and all of its contents as shown here:



For simplicity sake these files have been included with this article in the 3gl folder. These files contain the Uniface libraries for integration as well as their definitions in C. You will need to assign an environment variable named U3GL to point to this example. From explorer execute the shortcut named :



(This will setup the necessary paths to Visual Studio 2008.)



Enter these commands *:*

```
E:\
SET U3GL = E:\projects\3gl\metaphone\3gl
CD \projects\3gl\metaphone\src\3gl
```

This will point to the Uniface required libraries and includes.

2. Building the DLL

Enter this: **NMAKE /F Metaphone.mak**

This will execute the included Makefile to compile the sources and create the output DLL.

```
Microsoft (R) Program Maintenance Utility Version 9.00.21022.08
Copyright (C) Microsoft Corporation.  All rights reserved.

        cl /c    metaphone.c

Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 15.00.21022.08 for 80x86
Copyright (C) Microsoft Corporation.  All rights reserved.

metaphone.c
        LINK -DLL -out:metaphone.dll  -def:metaphone.def -NODEFAULTLIB:library /libpath:e:\projects\3gl\3gl
urtl.lib ulib.lib yrtl.lib metaphone.obj
Microsoft (R) Incremental Linker Version 9.00.21022.08
Copyright (C) Microsoft Corporation.  All rights reserved.

   Creating library metaphone.lib and object metaphone.exp
```

3. Copy the newly created DLL to your application based directory or from the included **bin** directory.
4. Examine the Metaphone.c [1]sources.

```
XEXPORT(long)   metaphone(char *Word, char *Metaph) {

….

}
```

This declares a method as a DLL export (allowing the method to be exposed) named ‘metaphone’ that accepts 2 strings.

Part II: Configuring Uniface to call a DLL.

You will need to modify your application or IDF.asn to notify Uniface that there is/are 3GL routines that can be called.

1. Open the assigned file
2. Locate the [3GL] section or add it as necessary
3. Enter the name of the DLL and it’s method that are going to be utilized.

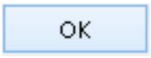
For example:

```
[USER_3GL]
METAPHONE(metaphone)
```

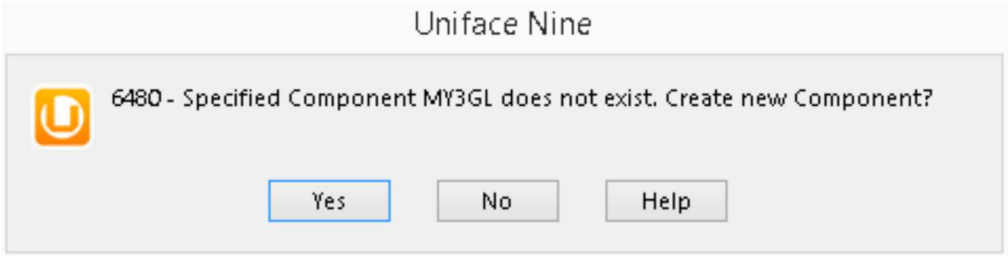
[1] This source code has been obtained from http://aspell.net/metaphone/ and all rights remain there.

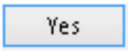
Part III: Entering a signature into Uniface

From the Main screen : **Editors -> Signature**
Enter '`MY3GL`' as the component

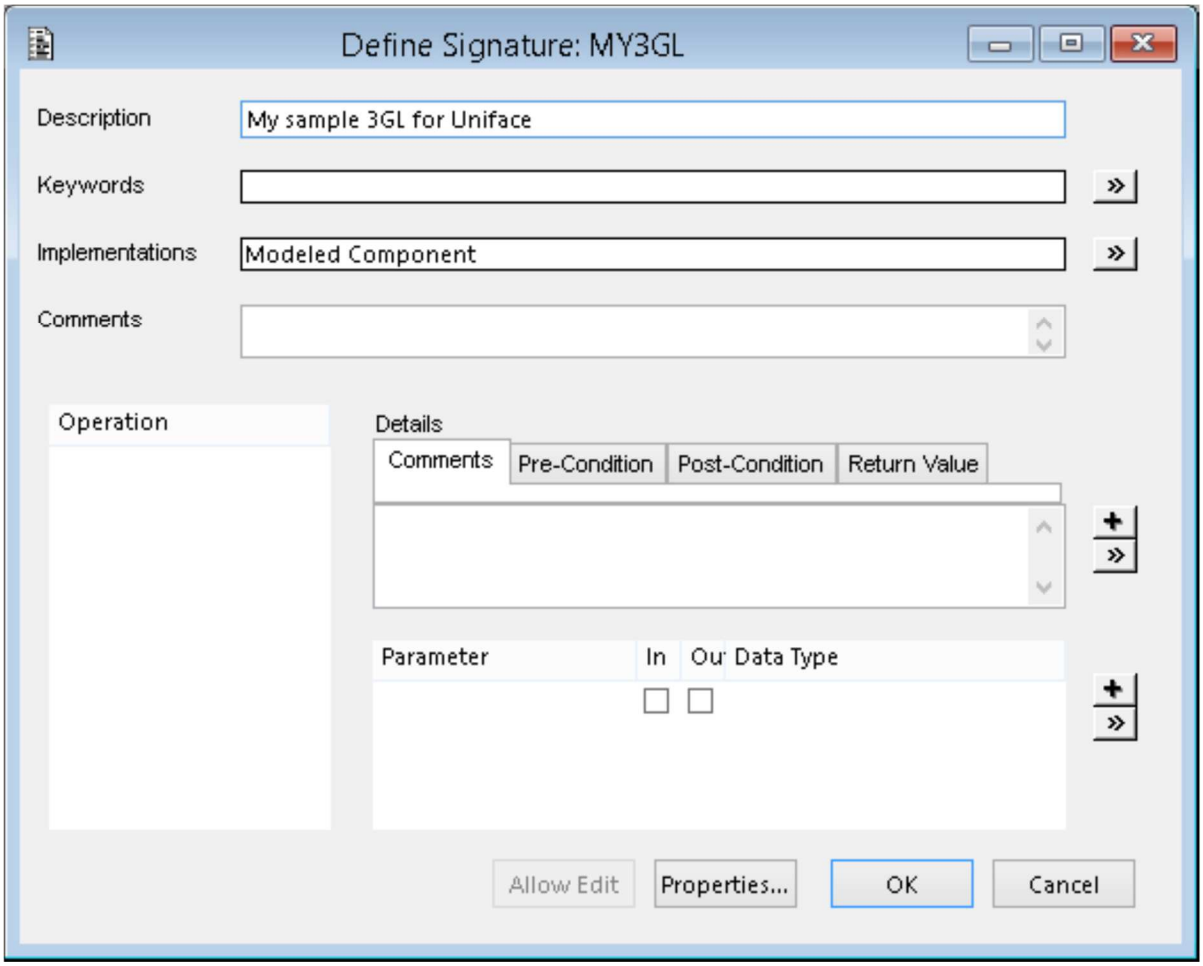
Click 

When prompted

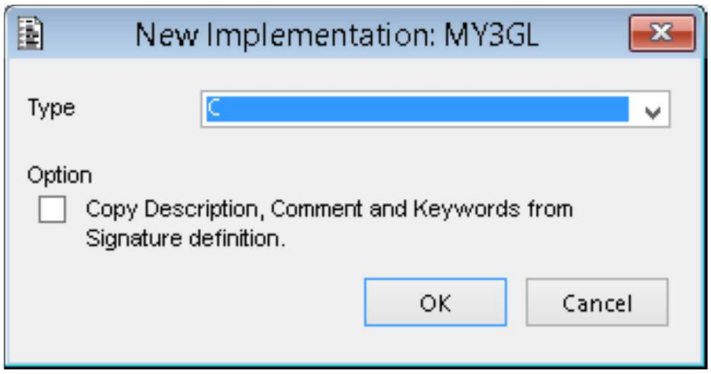


Click 

Enter '`My sample 3GL for Uniface`" as the Description

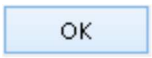


For Implementations Click »

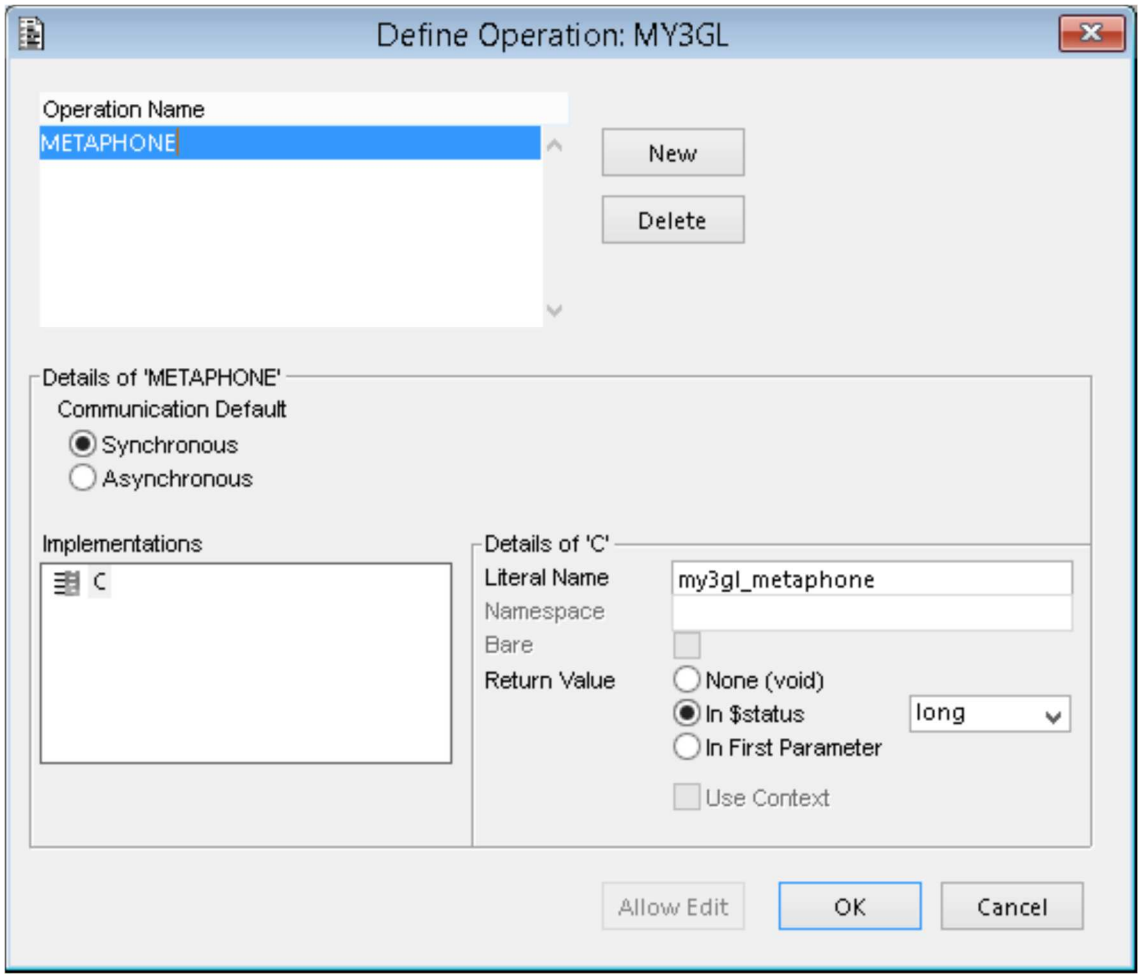


Select Type of '**C**'

Click OK

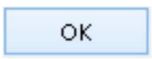
Click 

Enter '**METAPHONE**' as the operation.
On the Details side Click 

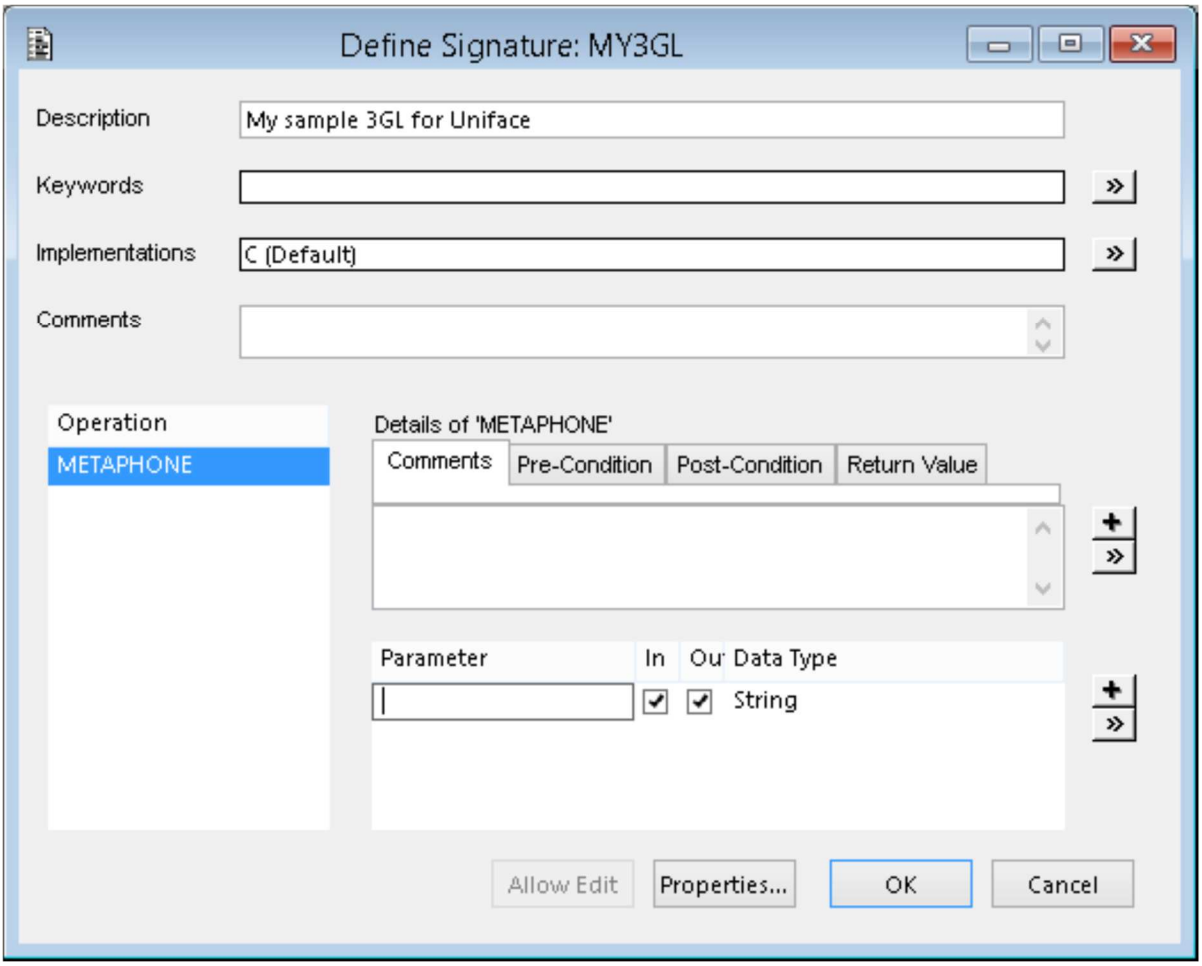


Make sure that you perform this step.

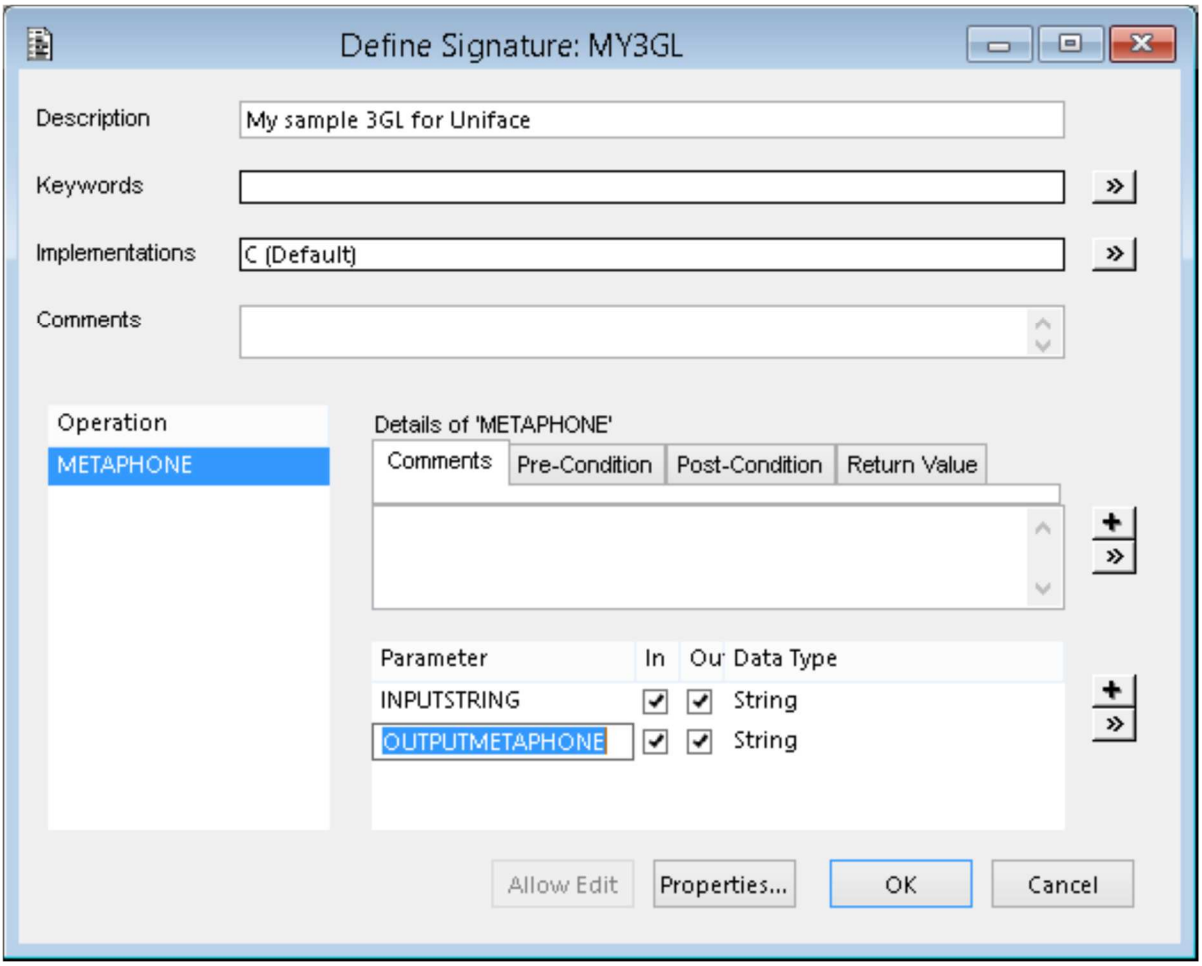
Change the **Literal Name** from **`my3gl_metaphone`** to **`metaphone`**. This is the actual name of the method that was created in our C Source.

Click 

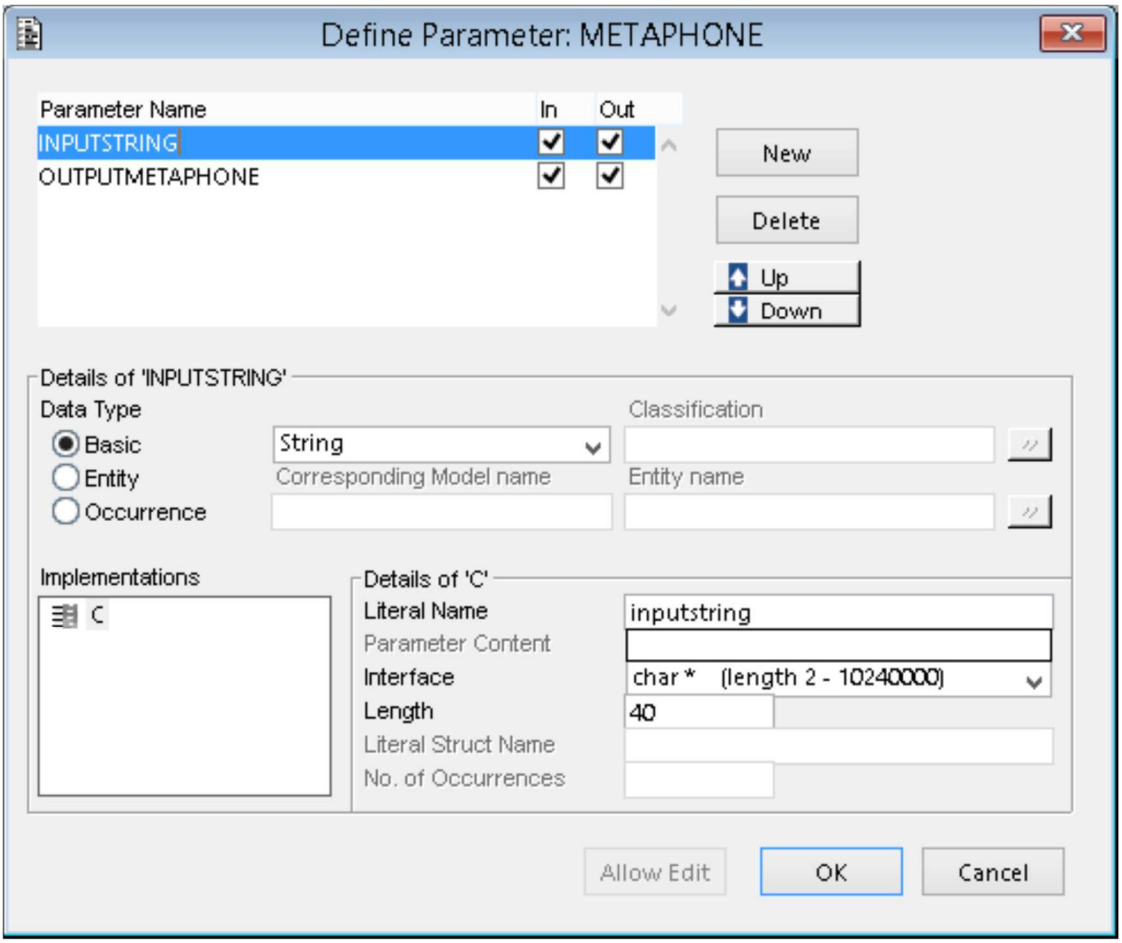
Click the  on the Parameters section



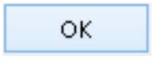
Enter '**`INPUTSTRING`**' as the Parameter name, Select **`In`** and **`Out`** and verify the datatype is set to '**`String`**'.
Click the +
Enter '**`OUTPUTMETAPHONE`**' as the Parameter name, Select **`In`** and **`Out`** and verify the datatype is set to '**`String`**'.



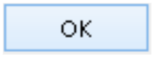
Click » to view more detail about the parameters.



Please note that the length of the strings must be specified. For this example 40 characters for the **`INPUTSTRING`** is sufficient. The **`OUTPUTMETAPHONE`** parameter really only needs to be 4 characters.

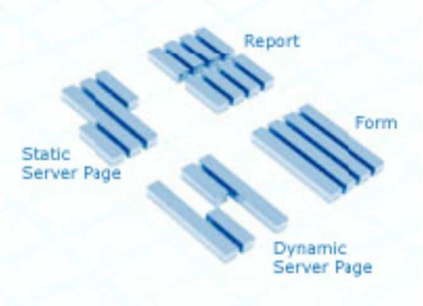
Click  to exit the parameter details screen.

From the Menu: **File -> Compile Signature**.

Click 

Part IV: Develop a test form

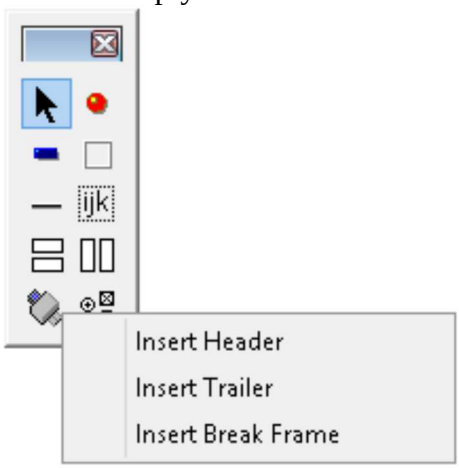
We need to create a new form to test our 3GL functionality. From the main Uniface window:



Either click on the Form glyph or **Editors -> Form**

For this example enter '**TEST3GL**' as the component/form name.

Insert an empty HEADER into the Structure. Click on the  icon and select 'Insert Header'

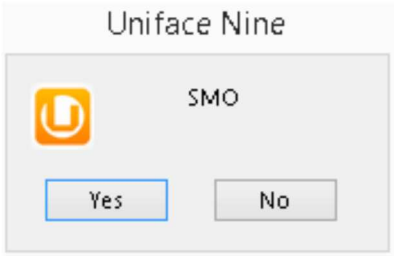


Enter the following code into the EXECUTE Trigger of the form:

```
variables
    string inputstr
    string metaphone
endvariables

inputstr = "SMITH"
activate "my3gl".metaphone ( inputstr, metaphone)
askmess metaphone

inputstr = "SMYTH"

activate "my3gl".metaphone ( inputstr, metaphone)
askmess metaphone


inputstr = "SMYTHE"

activate "my3gl".metaphone ( inputstr, metaphone)
askmess metaphone
```

File -> Test or press CTRL-F5

You should see 3 message boxes that all display :



Notice that SMITH, SMYTH, and SMYTHE all return the same metaphone string.

If the Transcript window says:

```
Unable to locate: my3gl_metaphone
*** The application has tried to dynamically activate this 3GL function but could not find it.
*** Check if you have an entry for this function in the [USER_3GL] section of your assignment file.
```

You didn't update the Literal Name in Part III: Entering a signature into Uniface.

Conclusion

This is only an example of some of the functionality that can be incorporated into Uniface. While the Metaphone functionality can be implemented in the Uniface proc language this shows how to leverage existing code assets into your application.

License

About the Author


Thomas S. Shore III

Subject Matter Expert

United States 

Thomas S. Shore is a Subject Matter Expert Uniface. Mr. Shore joined Uniface in 1993 and has held various positions in the UNIFACE product line and also focusing on J2EE and related technologies. Mr. Shore has also held various software engineering and consulting positions in the manufacturing, Oil & Gas exploration and database software markets. Mr. Shore holds a bachelor's degree in Entrepreneurial Studies from Babson College in Wellesley, Massachusetts.

.