+

El Kibambo v. 2.1

Official Manual

El CONCEPT

Content table

Technical details

Creator	Michael KYUNGU ILUNGA
Developed by	El CONCEPT
First version	Beta (v. 1.0) / 2021
Latest version	January 30, 2022
Deposit	https://github.com/elconcept15/elkibambo/tree/main/winforms
Written in	C#
Type	.NET Class Library
Target .NET Framework	>= 4.5
Supported Languages	French and English

Overview

El Kibambo is a .NET library for quick and easy implementation of search features on data grids in Windows Forms programs.

With El Kibambo, the search can be done on several columns of the data grid (`DataGridView`) and the comparison is not limited only to equality.

El Kibambo is currently in version 2.1, this is the first alpha version.

In this manual, we will illustrate it with the programming languages C# and VB.NET.

El Kibambo currently supports the following comparison operators:

- == (equality);
- != (Different) ;
- > (strictly superior);
- < (strictly lower);
- >= (higher or equal);
- <= (lower or equal);
- LIKE: an attempt by El Kibambo to use the legendary token search as the SQL operator « LIKE ».

The library El Kibambo has as a classes:

- **`Kibambo`**: this is the main class of the library. It is thanks to it that research is implemented.
- **`KibamboColumn`**: represents each column of the grid containing the data.
- **`KibamboException`**: in case of error, El Kibambo triggers exceptions of this type. This class has a `KibamboErrorCode` type attribute that allows us to better know the type of error.

El Kibambo use the following enumerations:

- **`KibamboErrorCode`**: each `KibamboException` type object has the attribute `ErrorCode` of type `KibamboErrorCode` that gives us more information about the type of error.
- **`KibamboColumnType`**: each object of `KibamboColumn` type has a `KibamboColumnType` property indicating the value type of the column. By the way, a column can have numeric values or character strings, and the way the comparison is made for numeric values and strings is not the same.
- **`KibamboComparisonType`**: allows you to specify the type of comparison (sensitive or not case sensitive, etc.). Each `Kibambo` object has a `ComparisonType` attribute that allows to specify the type of comparison.

El Kibambo also defines a delegate, the delegate `KibamboDelegate`. [...] :

```
public delegate void KibamboDelegate(DataGridViewRow row);
```

```
Public Delegate Sub KibamboDelegate(row As DataGridViewRow)
```

1. Enumerations

1.1. KibamboErrorCode

Overview:

```
public enum KibamboErrorCode
{
    ColumnNotRecognized = 0,
    ColumnsNotSpecified = 1,
    ComparisonTypeNotRecognized = 2,
    DgvOfDataMustBeDifferentThanDgvOfSearch = 3,
    MoreColumnsThanDgvData = 4,
    OperatorNotRecognized = 5,
    ValueMustBeNumeric = 6,
    YouMustSpecifyAtLeastOneOperator = 7
}
```

```
Public Enum KibamboErrorCode
    ColumnNotRecognized = 0
    ColumnsNotSpecified = 1
    ComparisonTypeNotRecognized = 2
    DgvOfDataMustBeDifferentThanDgvOfSearch = 3
    MoreColumnsThanDgvData = 4
    OperatorNotRecognized = 5
    ValueMustBeNumeric = 6
    YouMustSpecifyAtLeastOneOperator = 7
End Enum
```

This enumeration has as values:

- **`ColumnNotRecognized`**: with El Kibambo, each column of the data grid has passed in the form of an object of type `KibamboColumn` when creating the `Kibambo` object. To be accurate, we must give to the constructor of `Kibambo` class, a table of `KibamboColumn` objects representing the columns of the grid containing the data. When searching, if such a column is used, El Kibambo will search all the lines of which such column satisfies the search. This error is triggered in case where El Kibambo not found a column. This is the case, for example, of a column that does not exist in the grid containing the data, or deleted/renamed (`Name` property of a `DataGridViewColumn` object) after instancing the `Kibambo` object.
- **`ColumnsNotSpecified`:** this error is due to the fact that we have not specified columns when the `Kibambo` object is instantiated, and that the `Kibambo.AutoCompleteColumns` property is false.
- **`ComparisonTypeNotRecognized`**: this type of error is internal to the library. By default, the comparison is of type not case sensitive and ignoring diacritics signs (`KibamboComparisonType.IgnoreDiacriticSignsAndCase`). However, in the event of

a serious internal error and El Kibambo cannot determine the type of comparison (which is really less likely), this error can then occur.

- **`DgvOfDataMustBeDifferentThanDgvOfSearch`**: occurs if we pass the same `DataGridView` as a grid containing the data and also as a search grid when instancing the `Kibambo` object.
- **`MoreColumnsThanDgvData`**: happens if we pass more `KibamboColumn` objects to the `Kibambo` object than actually contains the grid containing the data.
- **`OperatorNotRecognized`**: this error arises in the event that, for one reason or another, El Kibambo fail to know or recognize the comparison operator during the search (which is still very much less likely).
- **`ValueMustBeNumeric`**: This error is due to the fact that the user is entering a non-numerical value for a numeric column.
- **`YouMustSpecifyAtLeastOneOperator`**: occurs if we pass an empty array for the operators when building a `KibamboColumn` object.

1.2. KibamboColumnType

Overview:

```
public enum KibamboColumnType
{
    String = 0,
    Numeric = 1
}
```

```
Public Enum KibamboColumnType
    [String] = 0
    Numeric = 1
End Enum
```

For the moment, El Kibambo only supports two types of values:

- **`String`**: for strings;
- **`Numeric`**: for numerical values.

So, El Kibambo can be used with other types using `KibamboColumnType.String`.

But why ...? It should be known that the comparison between numeric and non-numeric (strings) is diametrically opposite. If with the numerics it is the order of magnitude that is used, with the strings it is not the case. So, « ''10'' < ''2'' » give true but « 10 < 2 » give false!

1.3. KibamboComparisonType

Overview:

```
public enum KibamboComparisonType
{
    CaseSensitive = 0,
    IgnoreCase = 1,
    IgnoreDiacriticSigns = 2,
```

```
    IgnoreDiacriticSignsAndCase = 3
}
```

```
Public Enum KibamboComparisonType
    CaseSensitive = 0
    IgnoreCase = 1
    IgnoreDiacriticSigns = 2
    IgnoreDiacriticSignsAndCase = 3
End Enum
```

This enumeration has as values:

- **`CaseSensititve`**: for an ordinal comparison of strings that respect the case.
- **`IgnoreCase`**: for an ordinal comparison of strings not sensitive to case.
- **`IgnoreDiacriticSigns`**: for a comparison ignoring the diacritic signs that mainly accents.
- **`IgnoreDiacriticSignsAndCase`**: works as `IgnoreDiacriticSigns`, with this advantage that the comparison is not sensitive to case.

2. Class

2.1. KibamboException

Overview:

```
public class KibamboException : Exception
{
    public KibamboException(KibamboErrorCode errorCode);

    public KibamboErrorCode ErrorCode { get; }
    public string Message { get; }

    public override string ToString();
}
```

```
Public Class KibamboException Inherits Exception

    Public Sub New(errorCode As KibamboErrorCode)

    Public ReadOnly Property ErrorCode As KibamboErrorCode
    Public ReadOnly Property Message As String

    Public Overrides Function ToString() As String
End Class
```

Inheriting the base class of the exceptions `Exception`, this class defines two properties:

- **`Message`**: accessible read-only, it provides a message detailing the error. This property redefines the one defined in the base class.

```
string KibamboException.Message { get; }
```

```
ReadOnly Property KibamboException.Message As String
```

- **ErrorCode**: contains the error code. It is of type `KibamboErrorCode`.

```
KibamboErrorCode KibamboException.ErrorCode { get; }
```

```
ReadOnly Property KibamboException.ErrorCode As KibamboErrorCode
```

The `KibamboException` class redefines the `ToString()` base method by adding the error message at the end of the string:

```
public override string ToString()
{
    return base.ToString() + Environment.NewLine + "Kibambo Error Message : " + Message;
}
```

```
Public Overrides Function ToString() As String
    Return MyBase.ToString() + Environment.NewLine + "Kibambo Error Message : " + Message
End Function
```

2.2. KibamboColumn

The `KibamboColumn` class is used to represent each column of the grid containing the data.

Overview:

```
public class KibamboColumn
{
    public static readonly string[] SupportedOperators;

    public KibamboColumn(string name, KibamboColumnType columnType);
    public KibamboColumn(string name, string header);
    public KibamboColumn(string name, string header, KibamboColumnType columnType);
    public KibamboColumn(string name, string header, KibamboColumnType columnType, string
headerColumnType);
    public KibamboColumn(string name, string header, KibamboColumnType columnType, string
headerColumnType, string[] operators);

    public KibamboColumnType ColumnType { get; set; }
    public string Header { get; set; }
    public string HeaderColumnType { get; set; }
    public string Name { get; set; }
    public string[] Operators { get; set; }

    public override string ToString();
}
```

```
Public Class KibamboColumn
    Public Shared ReadOnly SupportedOperators As String()

    Public Sub New(name As String, columnType As KibamboColumnType)
    Public Sub New(name As String, header As String)
    Public Sub New(name As String, header As String, columnType As KibamboColumnType)
    Public Sub New(name As String, header As String, columnType As KibamboColumnType,
headerColumnType As String)
    Public Sub New(name As String, header As String, columnType As KibamboColumnType,
headerColumnType As String, operators() As String)
```

```
    Public Property ColumnType As KibamboColumnType
    Public Property Header As String
    Public Property HeaderColumnType As String
    Public Property Name As String
    Public Property Operators As String()

    Public Overrides Function ToString() As String
End Class
```

a. Properties

KibamboColumn has as properties: Name, Header, ColumnType, HeaderColumnType, Operators.

- **Name**: refers to the DataGridViewColumn.Name property of the grid column that contains the data that the KibamboColumn object represents.

```
string KibamboColumn.Name { get; set; }
```

```
Property KibamboColumn.Name As String
```

- **Header**: this is the title of the column to be displayed by El Kibambo.

```
string KibamboColumn.Header { get; set; }
```

```
Property KibamboColumn.Header As String
```

- **ColumnType**: represents the type of column.

```
KibamboColumnType KibamboColumn.ColumnType { get; set; }
```

```
Property KibamboColumn.ColumnType As KibamboColumnType
```

- **HeaderColumnType**: the type of column to display by El Kibambo in the search grid.

```
string KibamboColumn.HeaderColumnType { get; set; }
```

```
Property KibamboColumn.HeaderColumnType As String
```

- **Operators**: represents a table of signs of comparison operators to be used for the column. As a reminder, El Kibambo only supports eight operators (==, !=, <, <=, >, >=, !=, LIKE). You must specify that the operators are supported by El Kibambo. The presence of an unsupported operator throws a KibamboException with an error code that worth KibamboErrorCode.OperatorNotRecognized. The fact of specifying an empty table triggers also a KibamboException with an error code that worth KibamboErrorCode.YouMustSpecifyAtLeastOneOperator.

```
string[] KibamboColumn.Operators { get; set; }
```

```
Property KibamboColumn.Operators As String()
```

- **`SupportedOperators`**: constant containing a table of comparison operators supported by El Kibambo library.

```
string[] KibamboColumn.SupportedOperators
```

```
KibamboColumn.SupportedOperators As String()
```

b. Constructors

`KibamboColumn` class has the following constructors:

```
KibamboColumn(string name, KibamboColumnType columnType)

KibamboColumn(string name, string header)

KibamboColumn(string name, string header, KibamboColumnType columnType)

KibamboColumn(string name, string header, KibamboColumnType columnType, string
headerColumnType)

KibamboColumn(string name, string header, KibamboColumnType columnType, string
headerColumnType, string[] operators)
```

```
KibamboColumn(name As String, columnType As KibamboColumnType)

KibamboColumn(name As String, header As String)

KibamboColumn(name As String, header As String, columnType As KibamboColuymnType)

KibamboColumn(name As String, header As String, columnType As KibamboColuymnType,
headerColumnType As String)

KibamboColumn(name As String, header As String, columnType As KibamboColuymnType,
headerColumnType As String, operators As String())
```

Let's detail each parameter:

- `name`: value to be assigned to the property `KibamboColumn.Name`;
- `header:` value to be assigned to the property `KibamboColumn.Header`;
- `columnType`: value to be assigned to the property `KibamboColumn.ColumnType`;
- `headerColumnType`: value to be assigned to the property `KibamboColumn.HeaderColumnType`;
- `operators`: value to be assigned to the property `KibamboColumn.Operators`.

By using a constructor not taking a table of comparison operators, all the comparison operators supported by El Kibambo will be automatically used.

c. Methods

The `KibamboColumn` class redefines just one method: « `ToString()` ».

```
public override string ToString()
{
    return Header;
}
```

```
Public Overrides Function ToString() As String
    Return Header
End Function
```

2.3. Kibambo

The `Kibambo` class is the very heart of the El Kibambo Library. So, how do we proceed to set up search with El Kibambo?

By the way, do not forget that El Kibambo currently works only with data grids.

During the instanciation of `Kibambo` object, we must pass to the constructor the grid containing data and the one allowing to implement the search.

Overview:

```
public class Kibambo
{
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, bool autoCompleteColumns =
false);
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns, bool
autoCompleteColumns = false);
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboDelegate
searchHandler, bool autoCompleteColumns = false);
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboDelegate
searchHandler, KibamboDelegate cancelSearchHandler, bool autoCompleteColumns = false);
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns,
KibamboDelegate searchHandler, bool autoCompleteColumns = false);
    public Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns,
KibamboDelegate searchHandler, KibamboDelegate cancelSearchHandler, bool autoCompleteColumns =
false);

    public bool AutoCompleteColumns { get; set; }
    public KibamboDelegate CancelSearchHandler { get; set; }
    public KibamboColumn[] Columns { get; set; }
    public KibamboComparisonType ComparisonType { get; set; }
    public List<DataGridViewRow> RowsSatisfied { get; }
    public List<DataGridViewRow> RowsUnsatisfied { get; }
    public KibamboDelegate SearchHandler { get; set; }
    public string TitleField { get; set; }
    public string TitleType { get; set; }
    public string TitleOperator { get; set; }
    public string TitleValue { get; set; }

    public void CancelSearch();
    public void Initialize();
    public void Search();
```

```
}
```

```
Public Class Kibambo
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, Optional
autoCompleteColumns As Boolean = False)
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, columns() As
KibamboColumn, Optional autoCompleteColumns As Boolean = False)
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, searchHandler As
KibamboDelegate, Optional autoCompleteColumns As Boolean = False)
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, searchHandler As
KibamboDelegate, cancelSearchHandler As KibamboDelegate, Optional autoCompleteColumns As
Boolean = False)
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, columns() As
KibamboColumn, searchHandler As KibamboDelegate, Optional autoCompleteColumns As Boolean =
False)
    Public Sub New(dgvData As DataGridView, dgvSearch As DataGridView, columns() As
KibamboColumn, searchHandler As KibamboDelegate, cancelSearchHandler As KibamboDelegate,
Optional autoCompleteColumns As Boolean = False)

    Public Property AutoCompleteColumns As Boolean
    Public Property CancelSearchHandler As KibamboDelegate
    Public Property Columns As KibamboColumn()
    Public Property ComparisonType As KibamboComparisonType
    Public ReadOnly Property RowsSatisfied As List(Of DataGridViewRow)
    Public ReadOnly Property RowsUnsatisfied As List(Of DataGridViewRow)
    Public Property SearchHandler As KibamboDelegate
    Public Property TitleHead As String
    Public Property TitleType As String
    Public Property TitleOperator As String
    Public Property TitleValue As String

    Public Sub CancelSearch()
    Public Sub Initialize()
    Public Sub Search()
End Class
```

a. Constructors

The `Kibambo` class has the following constructors:

```
Kibambo(DataGridView dgvData, DataGridView dgvSearch, [bool autoCompleteColumns = false])

Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns, [bool
autoCompleteColumns = false])

Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboDelegate searchHandler,
[bool autoCompleteColumns = false])

Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns,
KibamboDelegate searchHandler, [bool autoCompleteColumns = false])

Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboDelegate searchHandler,
KibamboDelegate cancelSearchHandler, [bool autoCompleteColumns = false])

Kibambo(DataGridView dgvData, DataGridView dgvSearch, KibamboColumn[] columns,
KibamboDelegate searchHandler, KibamboDelegate cancelSearchHandler, [bool
autoCompleteColumns = false])
```

```
Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, [autoCompleteColumns As
Boolean = False])

Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, columns As KibamboColumn(),
[autoCompleteColumns As Boolean = False])

Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, searchHandler As
KibamboDelegate, [autoCompleteColumns As Boolean = False])

Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, columns As KibamboColumn(),
searchHandler As KibamboDelegate, [autoCompleteColumns As Boolean = False])

Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, searchHandler As
KibamboDelegate, cancelSearchHandler As KibamboDelegate, [autoCompleteColumns As Boolean
= False])

Kibambo(dgvData As DataGridView, dgvSearch As DataGridView, columns As KibamboColumn(),
searchHandler As KibamboDelegate, cancelSearchHandler As KibamboDelegate,
[autoCompleteColumns As Boolean = False])
```

Let's detail each parameter:

- `dgvData`: `DataGridView` containing the data;
- `dgvSearch`: `DataGridView` that will allow to set up the search features;
- `autoCompleteColumns`: indicates whether we want to adding automatically or not the unspecified columns;
- `columns`: table of `KibamboColumn` objects referring to each column of the grid containing the data;
- `searchHandler`: delegate of type `KibamboDelegate` to call when searching, i.e. when calling the `Kibambo.Search()` method;
- `cancelSearchHandler`: `KibamboDelegate` type delegate called by El Kibambo when we want to cancel a search done, that is to say when calling the method `Kibambo.CancelSearch()`.

Note the presence of a new type: the delegate `KibamboDelegate`. His statement in the meanders of El Kibambo is the following:

```
delegate void ElKibambo.KibamboDelegate(System.Windows.Forms.DataGridViewRow row)
```

```
Delegate Sub ElKibambo.KibamboDelegate(row As System.Windows.Forms.DataGridViewRow)
```

Quite simply a function that does not return anything and that receives as the only parameter a `DataGridViewRow` object representing of course a grid line containing the data.

But why? See you when studying `Kibambo.Search()` and `Kibambo.CancelSearch()` methods!

It should be noted that:

- when we use a constructor who don't take a table of `KibamboColumn` objects, `Kibambo` class use default settings of each column of a grid containing data. Also note that El Kibambo will consider that the column is of type `KibamboColumnType.String`. This is so and only if the `Kibambo.AutoCompleteColumns` property is `true`. However, if `Kibambo.AutoCompleteColumns` is false and we do not specify columns, El Kibambo triggers a `KibamboException` error with the `KibamboErrorCode.ColumnsNotSpecified` error code.
- If the value of `dgvData` and the value of `dgvSearch` is the same, that is to say that we pass to the constructor the same data grid, and for data and research, El Kibambo throw a `KibamboException` type exception with the error code `KibamboErrorCode.DgvOfDataMustBeDifferentThanDgvOfSearch`.
- When initializing (`Kibambo.Initialize()`), El Kibambo is setting up the search grid. For example, the following properties are disabled (value passed to `false`): `AllowDrop`, `AllowUserToAddRows`, `AllowUserToDeleteRows`, `AllowUserToOrderColumns`, `AllowUserToResizeRows`, `RowHeadersVisible`.

b. Properties

AutoCompleteColumns

```
bool Kibambo.AutoCompleteColumns { get; set; }
```

```
8Property Kibambo.AutoCompleteColumns As Boolean
```

This property tells El Kibambo to automatically complete or not the unspecified columns.

Thus, you can not specify columns and leave the task at El Kibambo automatically generate them on the basis of the columns of the grid containing the data. You can also specify only part of columns (if necessary, for example if some columns require some comparison operators and others not), and let El Kibambo complete the unspecified columns.

Columns

```
KibamboColumn[] Kibambo.Columns { get; set; }
```

```
Property Kibambo.Columns As KibamboColumn()
```

The `Columns` property allows you access in read and writing to `KibamboColumn` objects representing each column of the grid containing the data.

Thanks to this property, you can specify only certain lines of the grid containing the data if you want to search for example on some columns and others not (of course, the `Kibambo.AutoCompleteColumns` property must be `false` in this case).

ComparisonType

```
KibamboComparisonType Kibambo.ComparisonType { get; set; }
```

```
Property Kibambo.ComparisonType As KibamboComparisonType
```

Specify the type of comparison to use during the search. The default value of this property is « `KibamboComparisonType.IgnoreDiacriticSignsAndCase` ».

RowsSatisfied et RowsUnsatisfied

```
List<DataGridViewRow> Kibambo.RowsSatisfied { get; }
List<DataGridViewRow> Kibambo.RowsUnsatisfied { get; }
```

```
ReadOnly Property Kibambo.RowsSatisfied As List (Of DataGridView)
ReadOnly Property Kibambo.RowsUnsatisfied As List (Of DataGridView)
```

After a search with the `Kibambo.Search()` method, the property `Kibambo.RowsSatisfied` will contain all the lines that have met the search criteria and `Kibambo.RowsUnsatisfied` all lines that have not satisfied the search criteria.

SearchHandler et CancelSearchHandler

```
KibamboDelegate Kibambo.SearchHandler { get; set; }
KibamboDelegate Kibambo.CancelSearchHandler { get; set; }
```

```
Property Kibambo.SearchHandler As KibamboDelegate
Property Kibambo.CancelSearchHandler As KibamboDelegate
```

These two properties are of type `KibamboDelegate` and represent the functions called respectively (if specified) when calling `Kibambo.Search()` and `Kibambo.CancelSearch()`.

TitleField, TitleType, TitleOperator, TitleValue

These properties represent the four headers of the data grid that will implement the search. It is respectively: "field", "type", "operator", and "value". They just allow to change the default values as needed.

```
string Kibambo.TitleField { get ; set; }
string Kibambo.TitleType { get ; set; }
string Kibambo.TitleOperator { get ; set; }
string Kibambo.TitleValue { get ; set; }
```

```
Property Kibambo.TitleField As String
Property Kibambo.TitleType As String
Property Kibambo.TitleOperator As String
Property Kibambo.TitleValue As String
```

c. Methods

Search()

```
void Kibambo.Search()
```

```
Sub Kibambo.Search()
```

This method is called to perform the search with El Kibambo. When searching, all rows that satisfied the condition are added to the accessible collection using the `Kibambo.RowsSatisfied` property.

When searching, if the search delegate, `Kibambo.SearchHandler` has been specified, each line that has not satisfied the search criteria will be passed to the processing function.

CancelSearch()

```
void Kibambo.CancelSearch()
```

```
Sub Kibambo.CancelSearch()
```

When this function is called, if the `Kibambo.CancelSearchHandler` property is not null (i.e., a reminder function has been specified) and the collection `Kibambo.RowsUnsatisfied` is not empty, lines that have not satisfied the search criteria are passed in turn to the delegate `Kibambo.CancelSearchHandler`.

It is thus reasonable to reverse what has been done in the research delegate. For example, if in the search we have hidden lines that have not met the search criteria, it seems logical that in the `Kibambo.CancelSearchHandler` delegate, they should be re-displayed.

It should be noted that this method empties collections `Kibambo.RowsSatisfied` and `Kibambo.RowsUnsatisfied`.

Initialize()

```
void Kibambo.Initialize()
```

```
Sub Kibambo.Initialize()
```

This method is called to initialize/reset El Kibambo. This method is to be called just after building the `Kibambo` object or after modification of its properties.

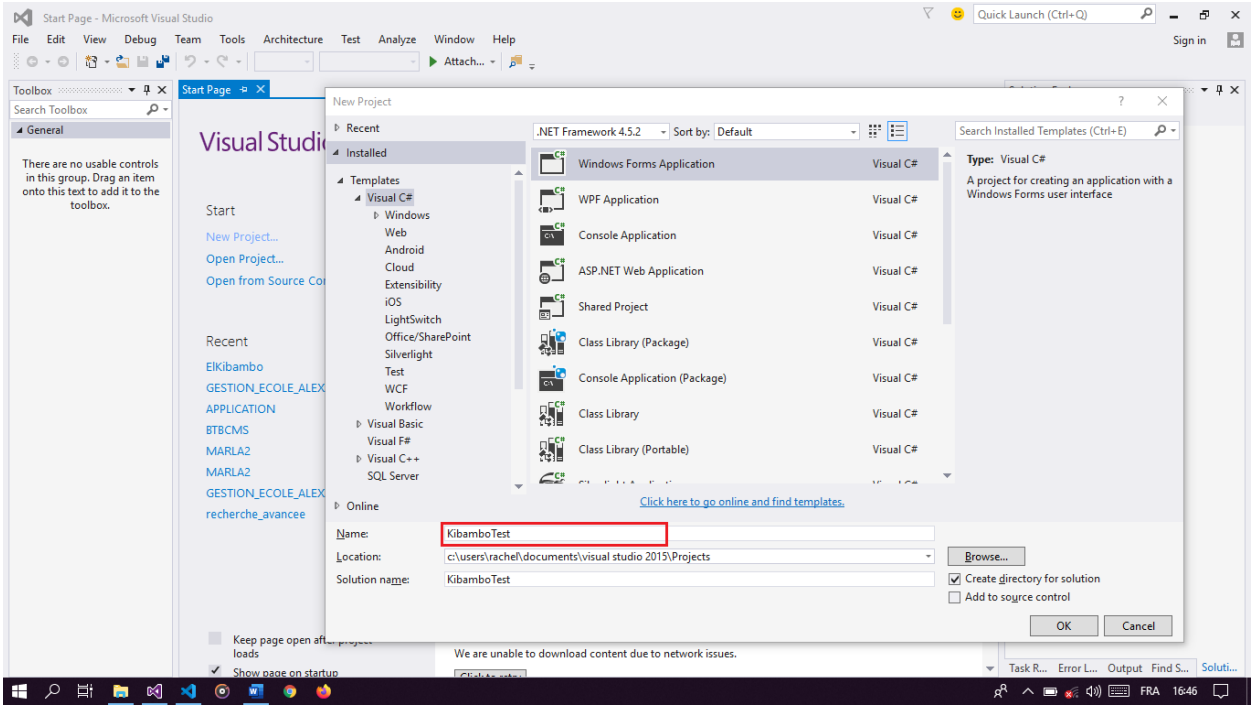
☞ You can find examples of use on El Kibambo's playlist on the YouTube channel of EL CONCEPT: *https://youtube.com/playlist?list=PL7i4WkgFSFMlT75YwyJVNduON6Tzu8upT*

- ☞ Library download link: *https://github.com/elconcept15/elkibambo/blob/main/winforms/elkibambo.dll*
- ☞ If needed, [...]:
  - Email: elconcept15@gmail.com
  - Contact: +243 82 34 23 495 / +243 85 57 99 756
  - Facebook page : https://web.facebook.com/elconcept15

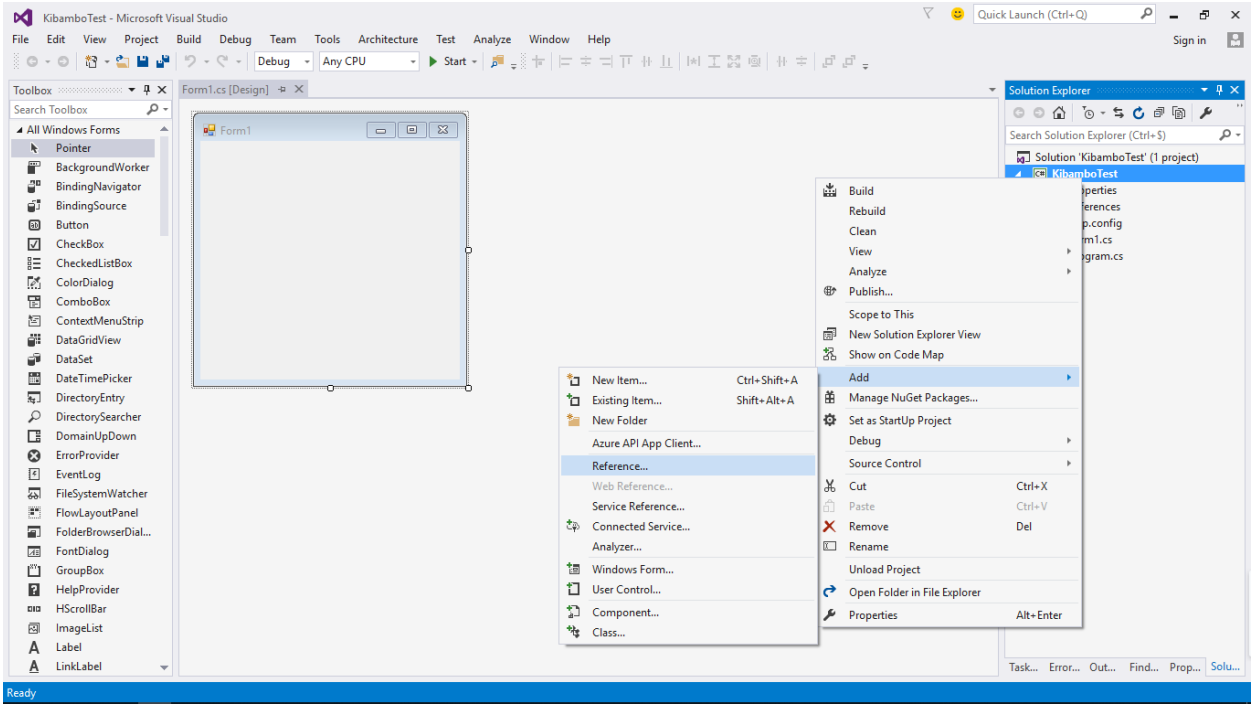
Implementation (C#)

In this section, we will create a C# Windows Forms program illustrating the implementation of the El Kibambo library. We will use as IDE "Microsoft Visual Studio Enterprise 2015".

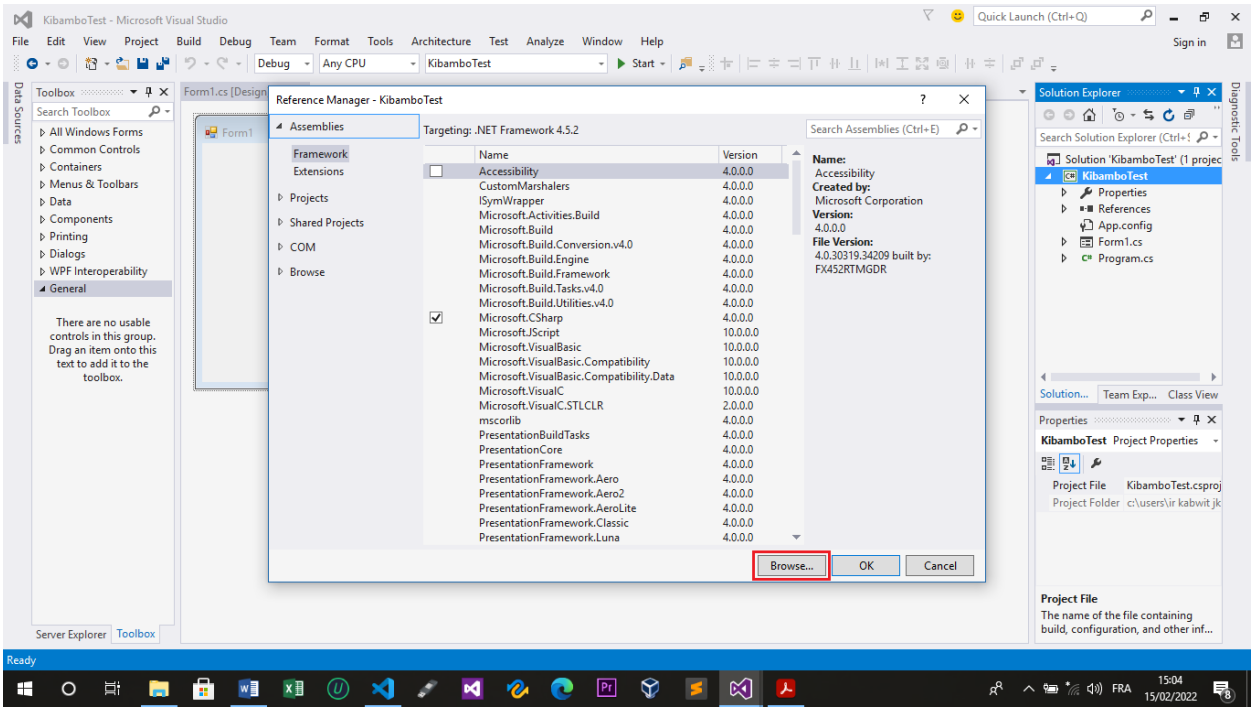
Create a new C# Windows Forms project that we call “KibamboTest”.



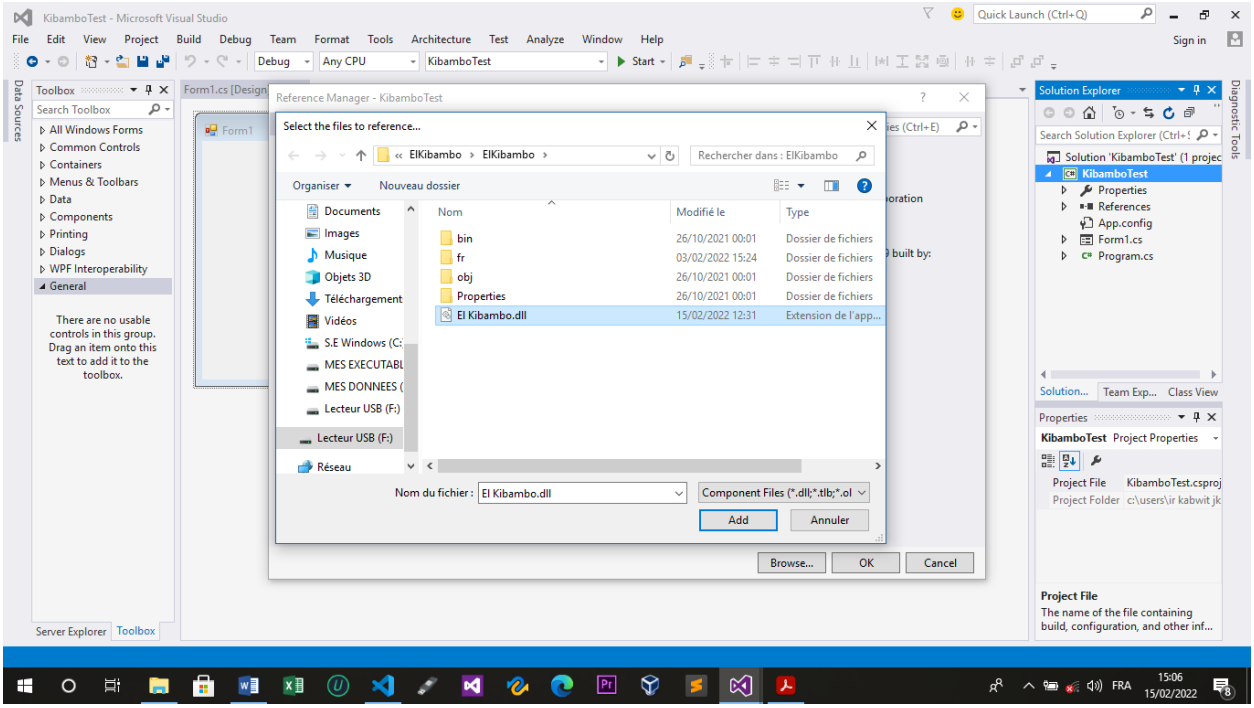
Let's now add the reference to the El Kibambo Library. For this, right-click on the project name, choose the "Add" option, then "Reference ...".



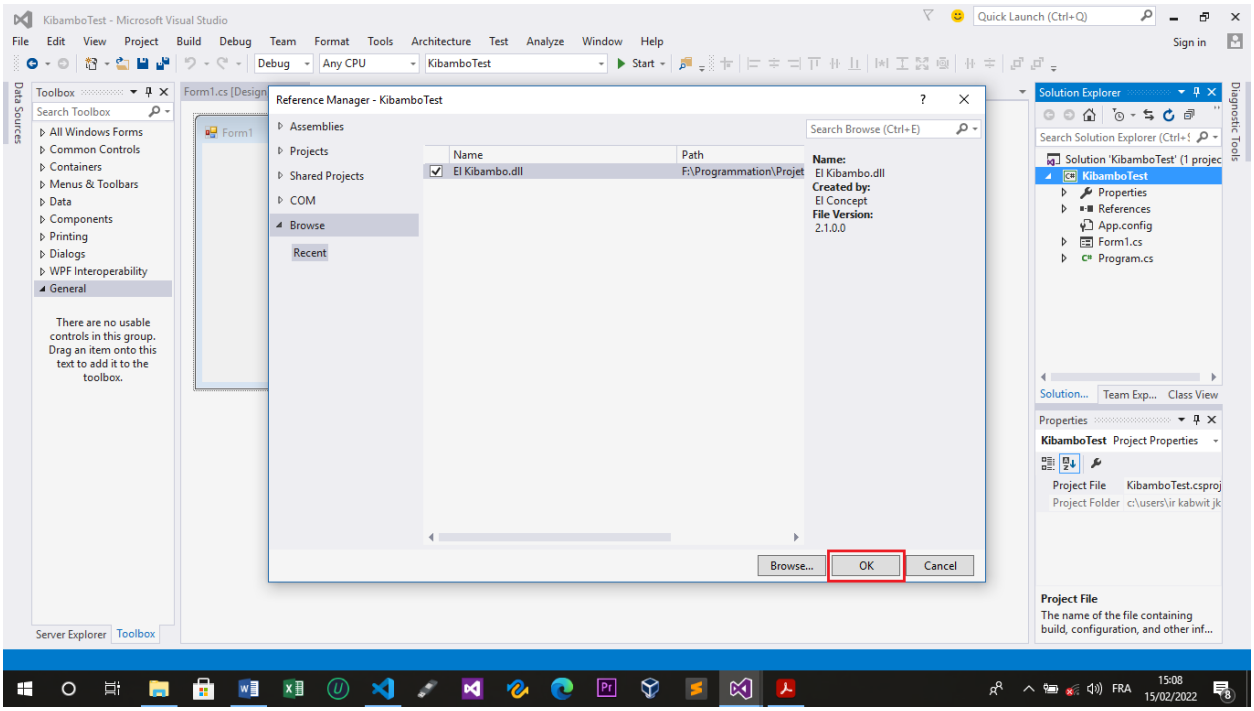
By clicking on the "Reference ..." option, Visual Studio will show you an interface similar to:



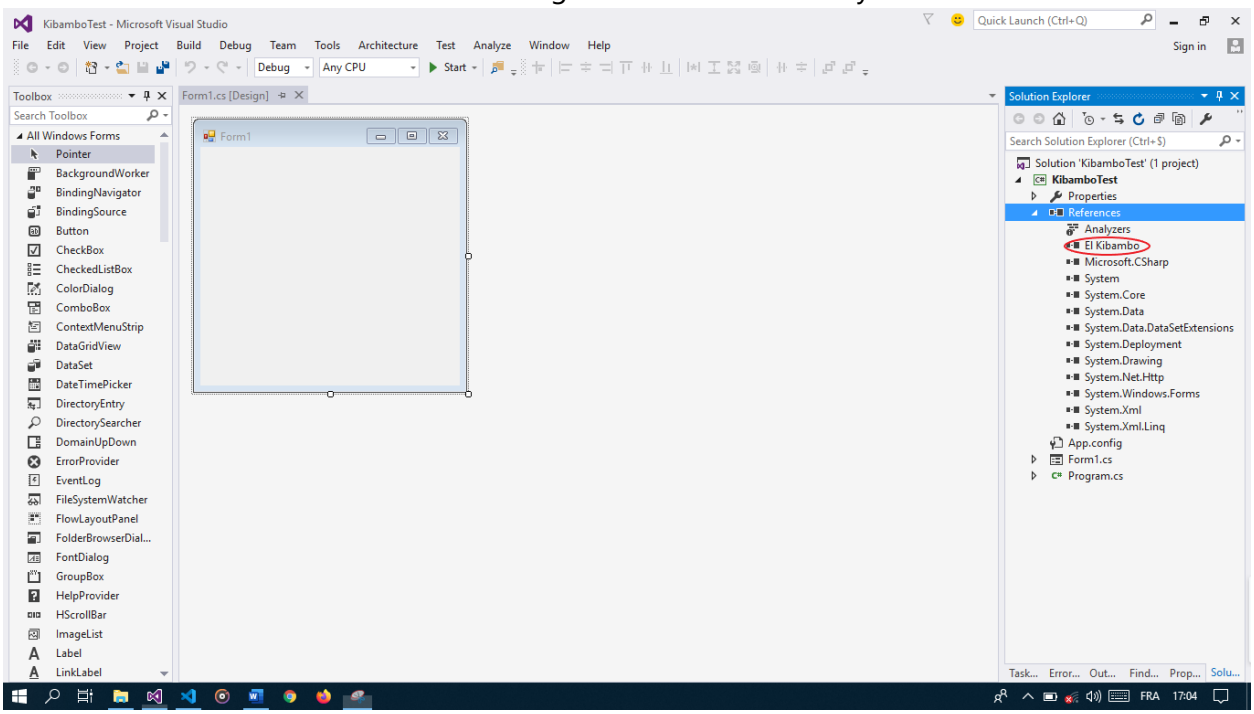
Press the "Browse ..." button, this will display the file explorer in order to choose the .dll file of the library.



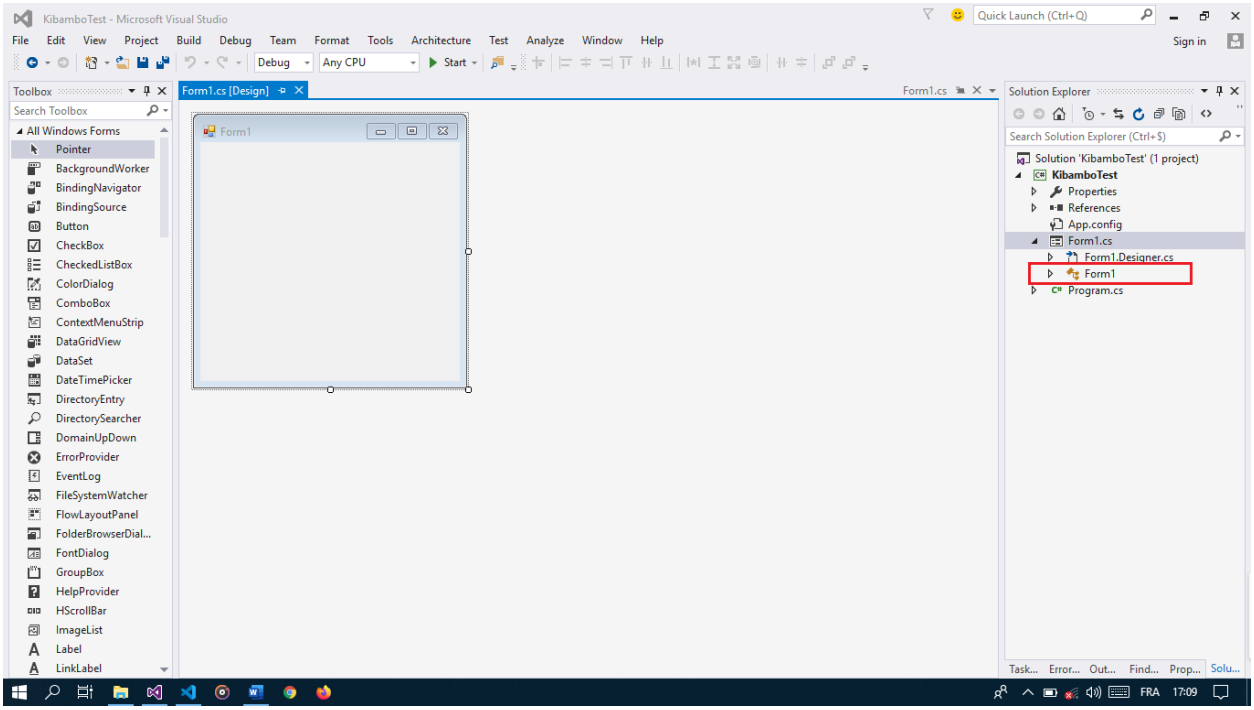
Press the "Add" button, you will have an interface similar to:



Press the "OK" button to finalize the adding reference to the library.



Now open the file "Form1.cs":



Replace the code in the "Form1.cs" file with the following:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

```
using System.Drawing;
using System.Windows.Forms;
using ElKibambo;

namespace KibamboTest
{
    public partial class Form1 : Form
    {
        DataGridView dgvPersons;
        DataGridView dgvSearch;

        GroupBox gbSearch;

        Button btnShowAll;
        Button btnSearch;

        Kibambo kibambo;

        public Form1()
        {
            InitializeComponent();
            InitializeControls();
        }

        private void InitializeControls()
        {
            dgvPersons = new DataGridView();
            dgvSearch = new DataGridView();

            gbSearch = new GroupBox();

            btnShowAll = new Button();
            btnSearch = new Button();

            ((ISupportInitialize)(dgvPersons)).BeginInit();
            ((ISupportInitialize)(dgvSearch)).BeginInit();

            gbSearch.SuspendLayout();
            SuspendLayout();

            Controls.Clear();

            //
            // dgvcs1
            //
            var dgvcs1 = new DataGridViewCellStyle()
            {
                Alignment = DataGridViewContentAlignment.MiddleCenter,
                BackColor = SystemColors.Control,
                Font = new Font("Trebuchet MS", 9F, FontStyle.Bold, GraphicsUnit.Point, 0),
                ForeColor = SystemColors.WindowText,
                SelectionBackColor = SystemColors.Highlight,
                SelectionForeColor = SystemColors.HighlightText,
                WrapMode = DataGridViewTriState.True
            };

            //
            // dgvcs2
            //
            var dgvcs2 = new DataGridViewCellStyle()
            {
```

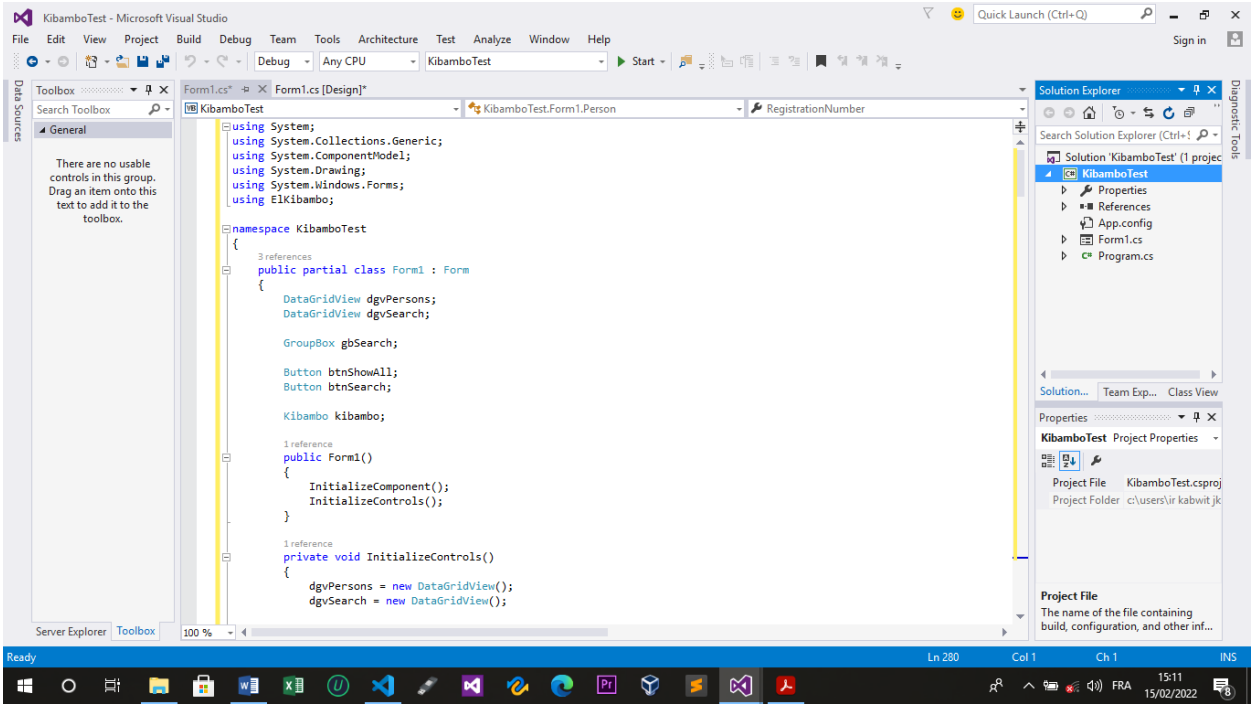
```
                Font = new Font("Segoe UI", 9F, FontStyle.Regular, GraphicsUnit.Point, 0)
            };

            //
            // dgvPersons
            //
            dgvPersons.AllowUserToAddRows = false;
            dgvPersons.AllowUserToDeleteRows = false;
            dgvPersons.AllowUserToResizeColumns = false;
            dgvPersons.AllowUserToResizeRows = false;
            dgvPersons.Anchor = ((AnchorStyles.Top | AnchorStyles.Bottom) | AnchorStyles.Left)
| AnchorStyles.Right;
            dgvPersons.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
            dgvPersons.BackgroundColor = Color.White;
            dgvPersons.BorderStyle = BorderStyle.Fixed3D;
            dgvPersons.ColumnHeadersDefaultCellStyle = dgvcs1;
            dgvPersons.ColumnHeadersHeightSizeMode =
DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            dgvPersons.EditMode = DataGridViewEditMode.EditProgrammatically;
            dgvPersons.Location = new Point(12, 12);
            dgvPersons.MultiSelect = false;
            dgvPersons.RowHeadersVisible = false;
            dgvPersons.RowsDefaultCellStyle = dgvcs2;
            dgvPersons.ScrollBars = ScrollBars.Vertical;
            dgvPersons.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
            dgvPersons.Size = new Size(470, 358);
            dgvPersons.TabIndex = 0;

            //
            // dgvSearch
            //
            dgvSearch.Anchor = (AnchorStyles.Top | AnchorStyles.Left) | AnchorStyles.Right;
            dgvSearch.BackgroundColor = Color.White;
            dgvSearch.BorderStyle = BorderStyle.Fixed3D;
            dgvSearch.ColumnHeadersHeightSizeMode =
DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            dgvSearch.Location = new Point(10, 19);
            dgvSearch.Size = new Size(268, 111);
            dgvSearch.TabIndex = 1;

            //
            // btnShowAll
            //
            btnShowAll.Anchor = AnchorStyles.Top | AnchorStyles.Right;
            btnShowAll.Location = new Point(106, 136);
            btnShowAll.Size = new Size(83, 23);
            btnShowAll.TabIndex = 3;
            btnShowAll.Text = "Show all";
            btnShowAll.UseVisualStyleBackColor = true;
            btnShowAll.Click += btnShowAll_Click;

            //
            // btnSearch
            //
            btnSearch.Anchor = AnchorStyles.Top | AnchorStyles.Right;
            btnSearch.Location = new Point(195, 136);
            btnSearch.Size = new Size(83, 23);
            btnSearch.TabIndex = 2;
            btnSearch.Text = "Search";
            btnSearch.UseVisualStyleBackColor = true;
            btnSearch.Click += btnSearch_Click;
```

```
            //
            // gbSearch
            //
            gbSearch.Anchor = AnchorStyles.Top | AnchorStyles.Right;
            gbSearch.Controls.Add(btnShowAll);
            gbSearch.Controls.Add(btnSearch);
            gbSearch.Controls.Add(dgvSearch);
            gbSearch.Font = new Font("Segoe UI", 8.25F, FontStyle.Regular, GraphicsUnit.Point,
0);
            gbSearch.Location = new Point(490, 12);
            gbSearch.Size = new Size(286, 170);
            gbSearch.TabIndex = 2;
            gbSearch.TabStop = false;
            gbSearch.Text = "Research";

            //
            // Form1
            //
            AutoScaleDimensions = new SizeF(6F, 13F);
            AutoScaleMode = AutoScaleMode.Font;
            BackColor = Color.White;
            ClientSize = new Size(789, 385);
            Controls.Add(gbSearch);
            Controls.Add(dgvPersons);
            Text = "El Kibambo - Illustration";
            Load += new EventHandler(Form1_Load);
            StartPosition = FormStartPosition.CenterScreen;

            ((ISupportInitialize)(dgvPersons)).EndInit();
            ((ISupportInitialize)(dgvSearch)).EndInit();

            gbSearch.ResumeLayout(false);
            ResumeLayout(false);
        }

        private void InitializeDgvPersonnes()
        {
            // Ajout des colonnes
            dgvPersons.Columns.Add("registration_number", "Reg. number");
            dgvPersons.Columns.Add("first_name", "First name");
            dgvPersons.Columns.Add("last_name", "Last name");
            dgvPersons.Columns.Add("age", "Age");

            // "Person" objects to display in the data grid
            var persons = new List<Person>
            {
                new Person("0001", "Michael", "Kyungu Ilunga", 21),
                new Person("0002", "Emmanuel", "Ilunga Ndalamba", 25),
                new Person("0003", "Jonathan", "Mulimbi Somwe", 22),
                new Person("0004", "Gloria", "Ngombe Kibambo", 20),
                new Person("0005", "Achille", "Mutombo Mubakilay", 22),
                new Person("0006", "Marc", "Kyalika Musomena", 23),
                new Person("0007", "Martin", "Manama Kabeya", 26),
                new Person("0008", "Ornellah", "Masengo Mutoni", 20),
                new Person("0009", "Elie", "Ebukeya Tshombe", 22),
                new Person("0010", "Françoise", "Ebukeya Lukonde", 15),
                new Person("0011", "Hansvané", "Kashala Kahilu", 23),
                new Person("0012", "Benoit", "Kamona Bunake", 23),
                new Person("0013", "Samuel", "Ngandu Mwepu", 21),
                new Person("0014", "Jules", "Malemo Miyayo", 23),
```

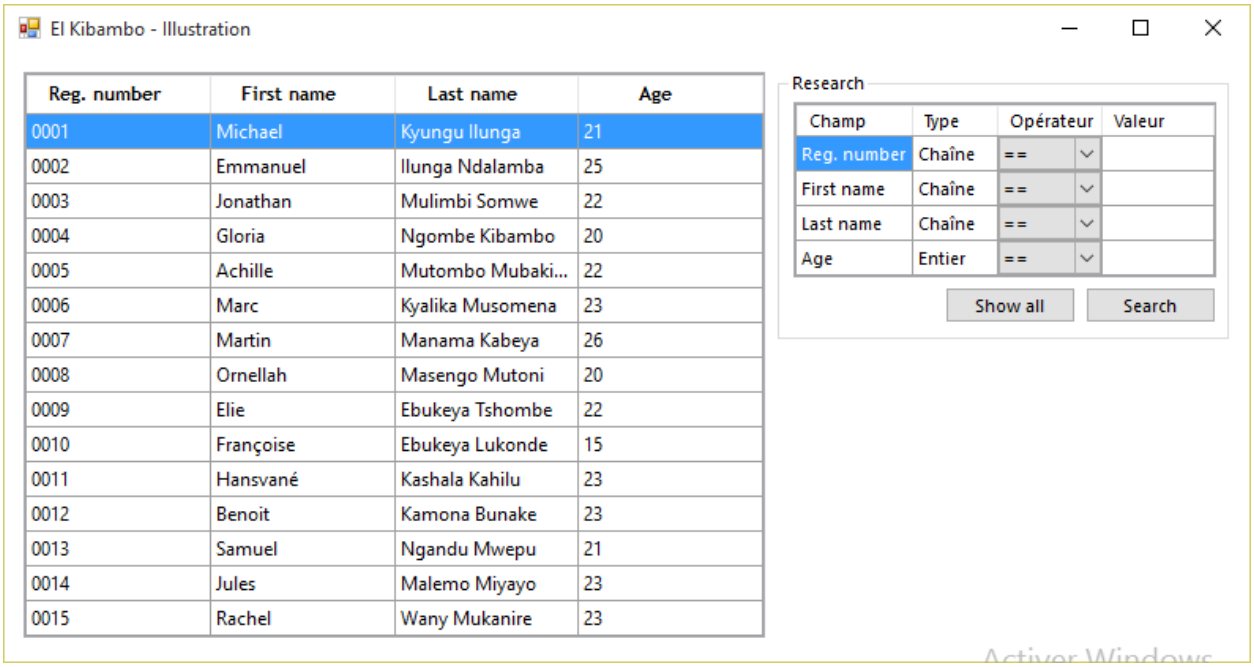
```
                new Person("0015", "Rachel", "Wany Mukanire", 23)
            };

            // Adding data in the data grid
            foreach (Person person in persons)
            {
                dgvPersons.Rows.Add(
                    person.RegistrationNumber,
                    person.FirstName,
                    person.LastName,
                    person.Age
                );
            }
        }

        private void InitializeKibambo()
        {
            kibambo = new Kibambo(dgvPersons, dgvSearch, true)
            {
                Columns = new KibamboColumn[]
                {
                    new KibamboColumn("registration_number", "Reg. number")
                    {
                        Operators = new[] { "==", "!=" }
                    },

                    new KibamboColumn("age", "Age", KibamboColumnType.Numeric, "Entier", new
string[] { "==", "<", ">", "<=", ">=", "!=" })
                },

                SearchHandler = delegate (DataGridViewRow row)
                {
                    // We choose to make the lines that have not satisfied the search criteria
invisible.
                    // But you can nevertheless do something else, for example changing the
color, etc.
                    row.Visible = false;
                },

                CancelSearchHandler = delegate (DataGridViewRow row)
                {
                    // As we have masked the lines that do not meet the search criteria during
the research, it seems logical that we can re-display them when we want to cancel the search
made.
                    // We must therefore put the line to its initial state before the search.
                    row.Visible = true;
                }
            };

            kibambo.Initialize();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            InitializeDgvPersonnes();
            InitializeKibambo();
        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            try
```

```
            {
                kibambo.Search();
            }

            catch (KibamboException ex)
            {
                switch (ex.ErrorCode)
                {
                    case KibamboErrorCode.ValueMustBeNumeric:
                        MessageBox.Show("You must enter a digital type value for the digital
column!", " Input error ", MessageBoxButtons.OK, MessageBoxIcon.Error);
                        break;

                    default:
                        MessageBox.Show("An error occurred during the search. The error
message is " + ex.Message + " !", " Error during search ", MessageBoxButtons.OK,
MessageBoxIcon.Error);
                        break;
                }
            }
        }

        private void btnShowAll_Click(object sender, EventArgs e)
        {
            kibambo.CancelSearch();
        }

        public class Person
        {
            public string RegistrationNumber { get; set; }
            public string FirstName { get; set; }
            public string LastName { get; set; }
            public int Age { get; set; }

            public Person(string registrationNumber, string firstName, string lastName, int
age)
            {
                RegistrationNumber = registrationNumber;
                FirstName = firstName;
                LastName = lastName;
                Age = age;
            }
        }
    }
}
```

Run the application by pressing the "Start" button or with the F5 function key:



Result:



A concise syntax could be used with lambda expressions when instantiating the Kibambo object:

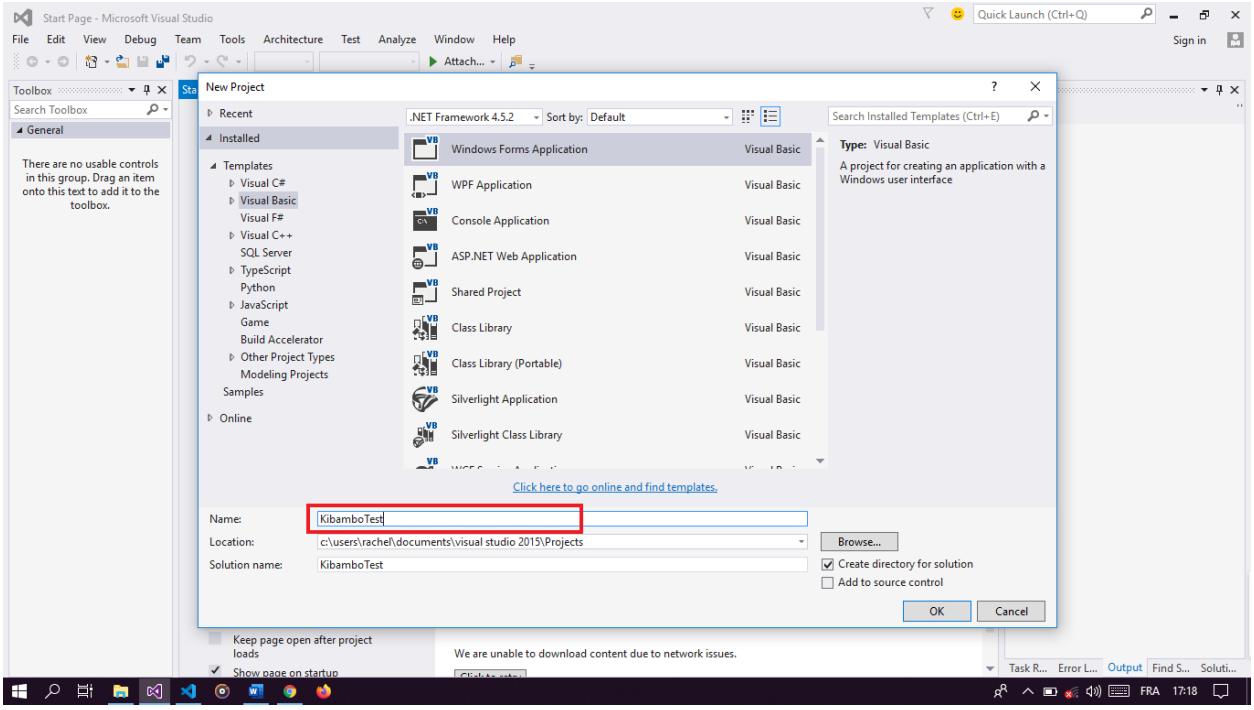
```
kibambo = new Kibambo(dgvPersons, dgvSearch, true)
{
    Columns = new KibamboColumn[]
    {
        new KibamboColumn("registration_number", "Reg. number")
        {
```

```
            Operators = new[] { "==", "!=" }
        },

        new KibamboColumn("age", "Age", KibamboColumnType.Numeric, "Entier", new string[] {
"==", "<", ">", "<=", ">=", "!=" })
    },

    SearchHandler = (row) => row.Visible = false,
    CancelSearchHandler = (row) => row.Visible = true
};
```

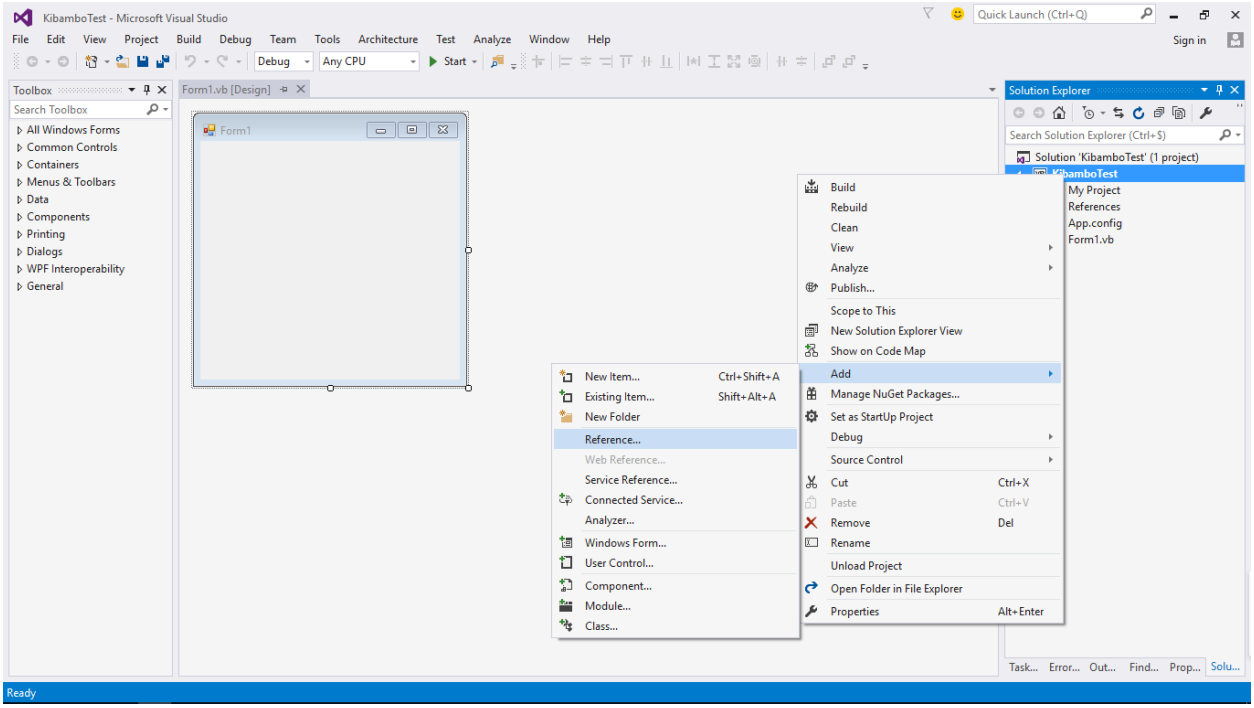
Implementation (VB.NET)

In this section, we will create a VB.NET Windows Forms program illustrating the implementation of the El Kibambo Library. We will use as IDE "Microsoft Visual Studio Enterprise 2015".

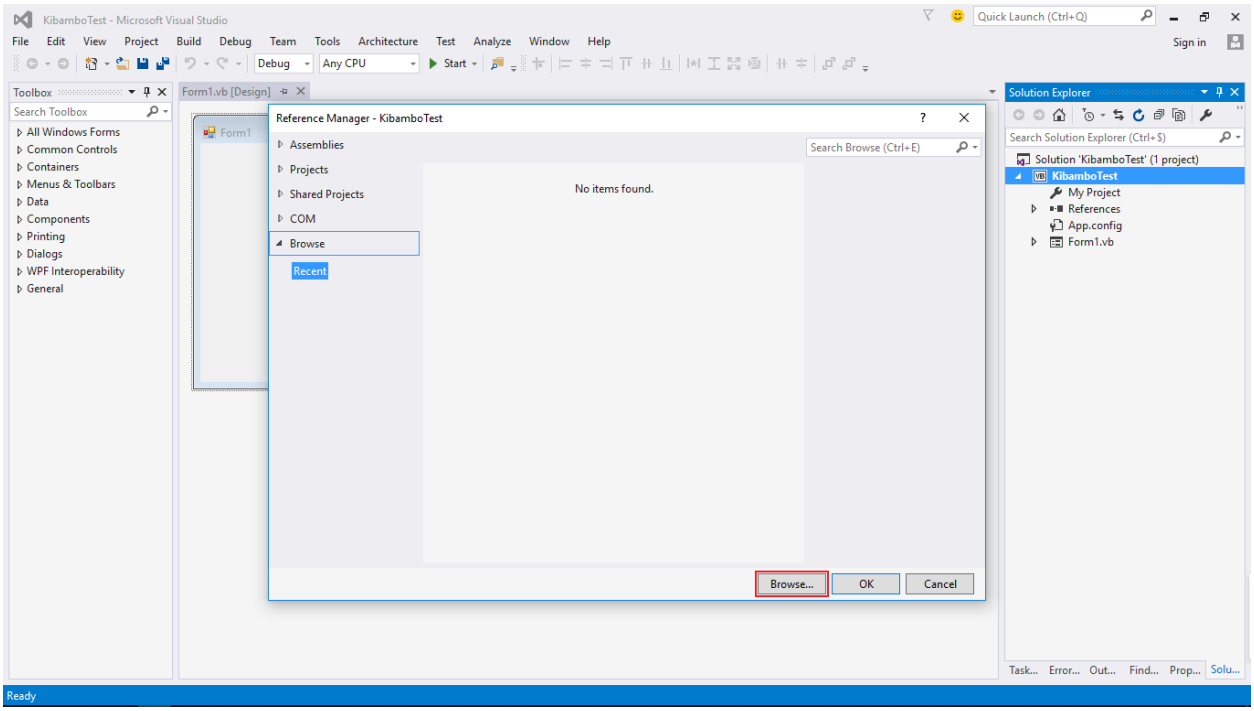
Let's create a new VB.NET Windows Forms project that we name "KibamboTest".



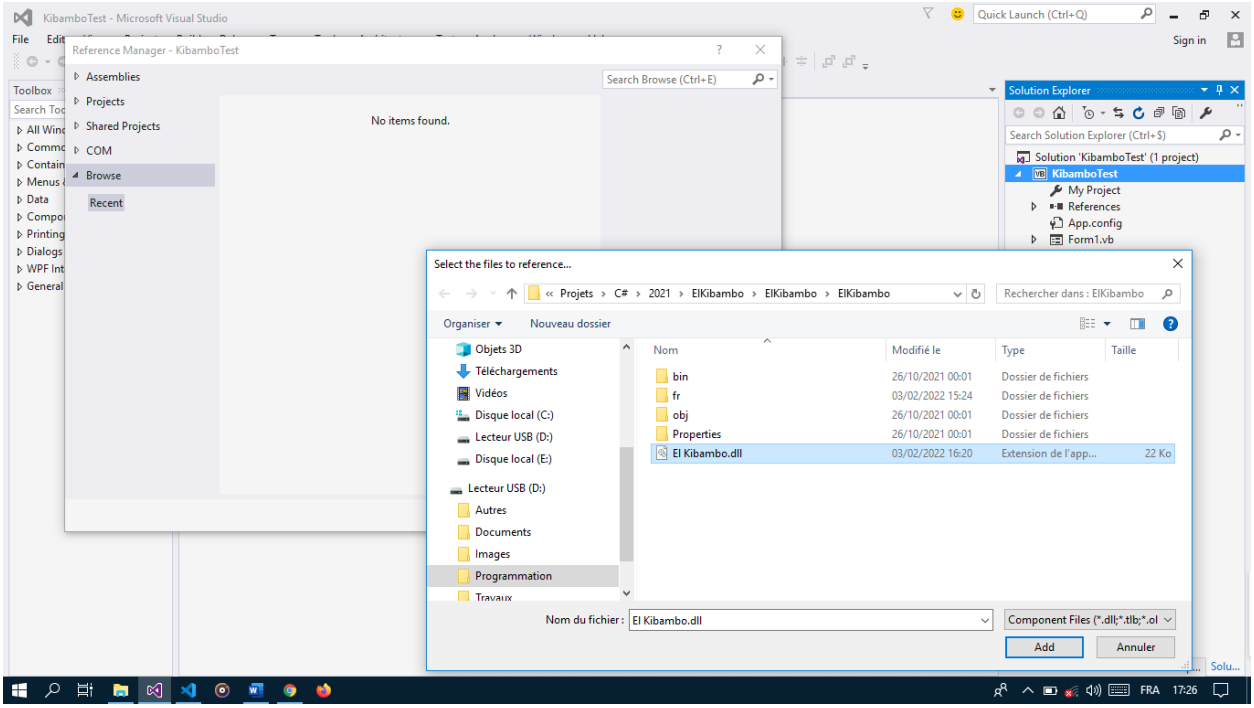
Now let us add the reference to the El Kibambo library. For this, right-click on the project name, choose the "Add" option, then "Reference ...".



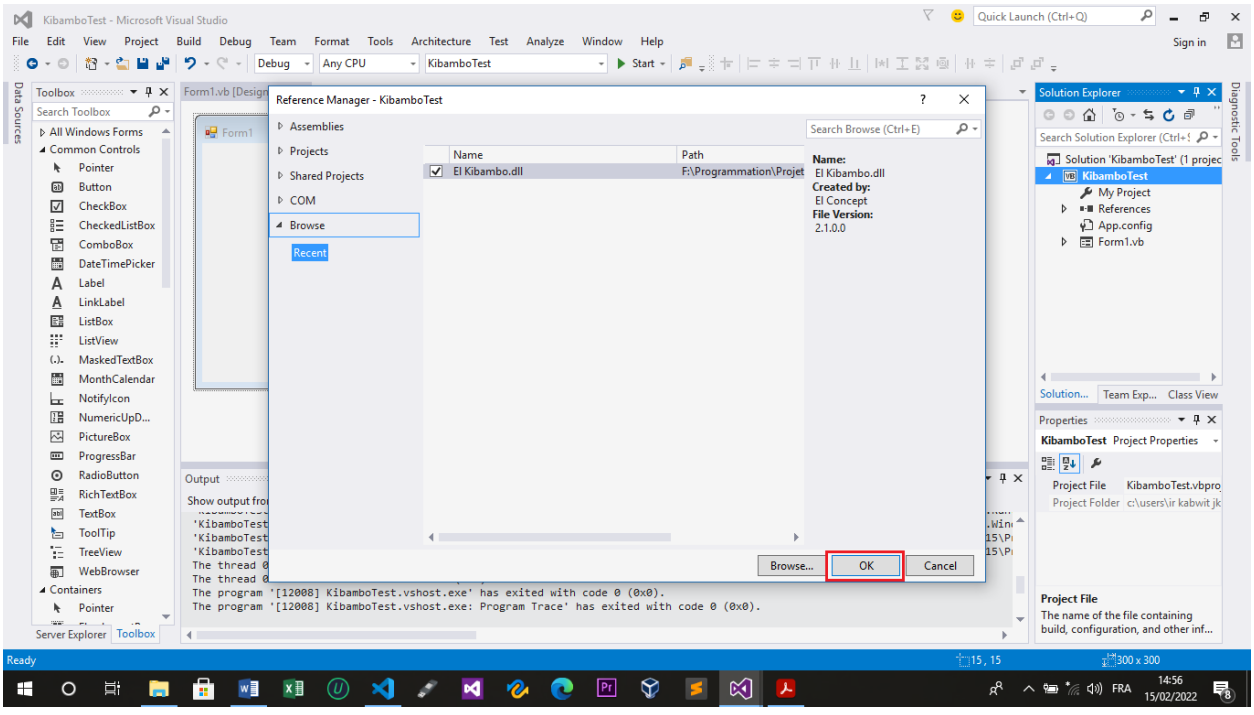
By clicking on the "Reference ..." option, Visual Studio will show you an interface similar to:



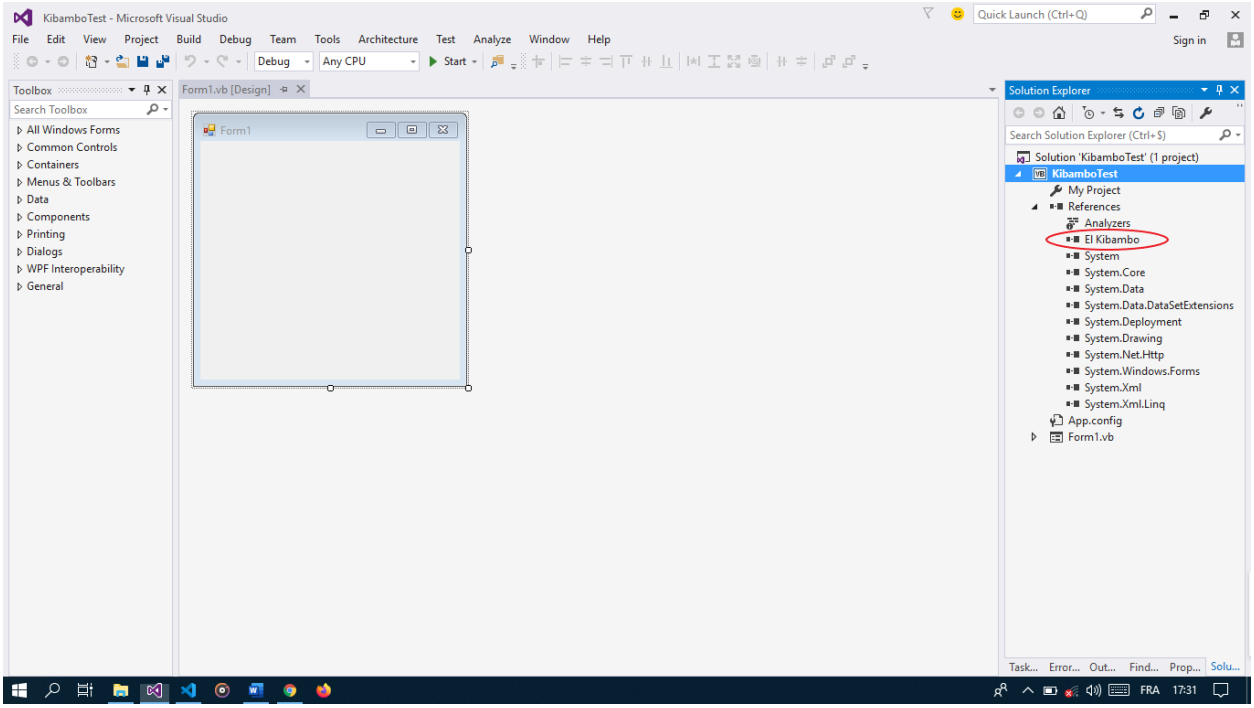
Press the "Browse ..." button, this will display the file explorer in order to choose the .dll file of the library.



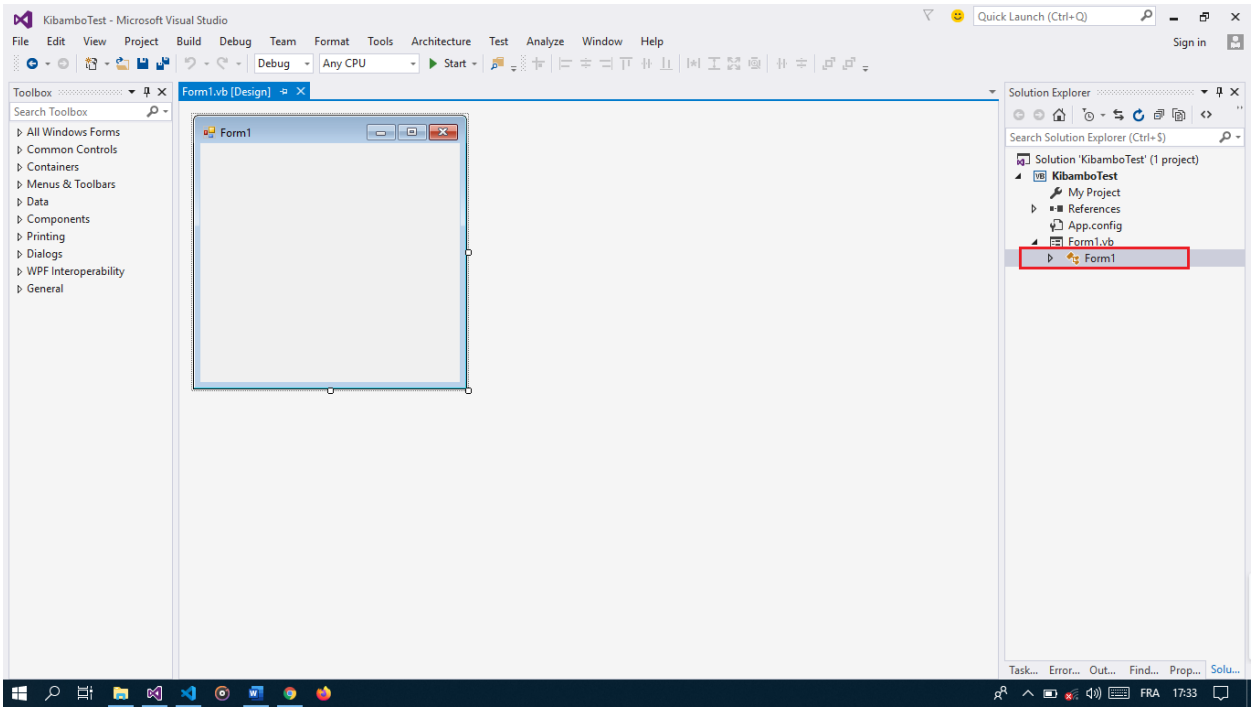
Press the "Add" button, you will have the following interface:



Press the "OK" button to finalize the adding reference to the library.



Now open the "Form1.VB" file:



Replace the code in the "Form1.vb" file with the following:

```
Imports ElKibambo
Imports System.ComponentModel

Public Class Form1
    Dim dgvPersons As DataGridView
```

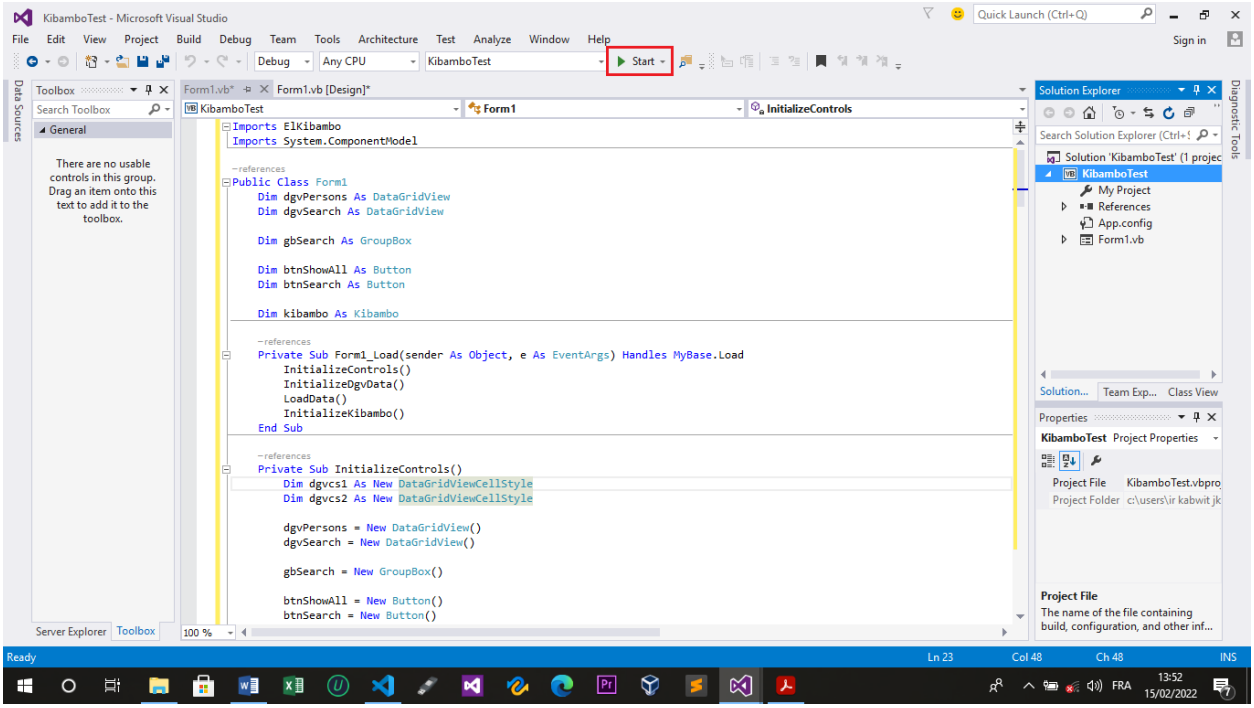
```
    Dim dgvSearch As DataGridView

    Dim gbSearch As GroupBox

    Dim btnShowAll As Button
    Dim btnSearch As Button

    Dim kibambo As Kibambo

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        InitializeControls()
        InitializeDgvData()
        LoadData()
        InitializeKibambo()
    End Sub

    Private Sub InitializeControls()
        Dim dgvcs1 As New DataGridViewCellStyle
        Dim dgvcs2 As New DataGridViewCellStyle

        dgvPersons = New DataGridView()
        dgvSearch = New DataGridView()

        gbSearch = New GroupBox()

        btnShowAll = New Button()
        btnSearch = New Button()

        CType(dgvPersons, ISupportInitialize).BeginInit()
        CType(dgvSearch, ISupportInitialize).BeginInit()

        gbSearch.SuspendLayout()
        SuspendLayout()

        Controls.Clear()

        '
        ' dgvcs1
        '
        dgvcs1.Alignment = DataGridViewContentAlignment.MiddleCenter
        dgvcs1.BackColor = SystemColors.Control
        dgvcs1.Font = New Font("Trebuchet MS", 9.0!, FontStyle.Bold, GraphicsUnit.Point,
CType(0, Byte))
        dgvcs1.ForeColor = SystemColors.WindowText
        dgvcs1.SelectionBackColor = SystemColors.Highlight
        dgvcs1.SelectionForeColor = SystemColors.HighlightText
        dgvcs1.WrapMode = DataGridViewTriState.True

        '
        ' dgvcs2
        '
        dgvcs2.Font = New Font("Segoe UI", 9.0!, FontStyle.Regular, GraphicsUnit.Point,
CType(0, Byte))

        '
        ' dgvPersons
        '
        dgvPersons.AllowUserToAddRows = False
        dgvPersons.AllowUserToDeleteRows = False
        dgvPersons.AllowUserToResizeColumns = False
        dgvPersons.AllowUserToResizeRows = False
```

```
        dgvPersons.Anchor = CType((((AnchorStyles.Top Or AnchorStyles.Bottom) _
            Or AnchorStyles.Left) _
            Or AnchorStyles.Right), AnchorStyles)
        dgvPersons.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill
        dgvPersons.BackgroundColor = Color.White
        dgvPersons.BorderStyle = BorderStyle.Fixed3D
        dgvPersons.ColumnHeadersDefaultCellStyle = dgvcs1
        dgvPersons.ColumnHeadersHeightSizeMode =
DataGridViewColumnHeadersHeightSizeMode.AutoSize
        dgvPersons.EditMode = DataGridViewEditMode.EditProgrammatically
        dgvPersons.Location = New Point(12, 12)
        dgvPersons.MultiSelect = False
        dgvPersons.RowHeadersVisible = False
        dgvPersons.RowsDefaultCellStyle = dgvcs2
        dgvPersons.ScrollBars = ScrollBars.Vertical
        dgvPersons.SelectionMode = DataGridViewSelectionMode.FullRowSelect
        dgvPersons.Size = New Size(433, 336)
        dgvPersons.TabIndex = 0

        '
        ' dgvSearch
        '
        dgvSearch.Anchor = CType(((AnchorStyles.Top Or AnchorStyles.Left) _
            Or AnchorStyles.Right), AnchorStyles)
        dgvSearch.BackgroundColor = Color.White
        dgvSearch.BorderStyle = BorderStyle.Fixed3D
        dgvSearch.ColumnHeadersHeightSizeMode =
DataGridViewColumnHeadersHeightSizeMode.AutoSize
        dgvSearch.Location = New Point(10, 19)
        dgvSearch.Size = New Size(268, 111)
        dgvSearch.TabIndex = 1

        '
        ' btnShowAll
        '
        btnShowAll.Anchor = CType((AnchorStyles.Top Or AnchorStyles.Right), AnchorStyles)
        btnShowAll.Location = New Point(106, 136)
        btnShowAll.Size = New Size(83, 23)
        btnShowAll.TabIndex = 3
        btnShowAll.Text = "Show all"
        btnShowAll.UseVisualStyleBackColor = True
        AddHandler btnShowAll.Click, AddressOf btnShowAll_Click

        '
        ' btnSearch
        '
        btnSearch.Anchor = CType((AnchorStyles.Top Or AnchorStyles.Right), AnchorStyles)
        btnSearch.Location = New Point(195, 136)
        btnSearch.Size = New Size(83, 23)
        btnSearch.TabIndex = 2
        btnSearch.Text = "Search"
        btnSearch.UseVisualStyleBackColor = True
        AddHandler btnSearch.Click, AddressOf btnSearch_Click

        '
        ' gbSearch
        '
        gbSearch.Anchor = CType((AnchorStyles.Top Or AnchorStyles.Right), AnchorStyles)
        gbSearch.Controls.Add(btnShowAll)
        gbSearch.Controls.Add(btnSearch)
        gbSearch.Controls.Add(dgvSearch)
```

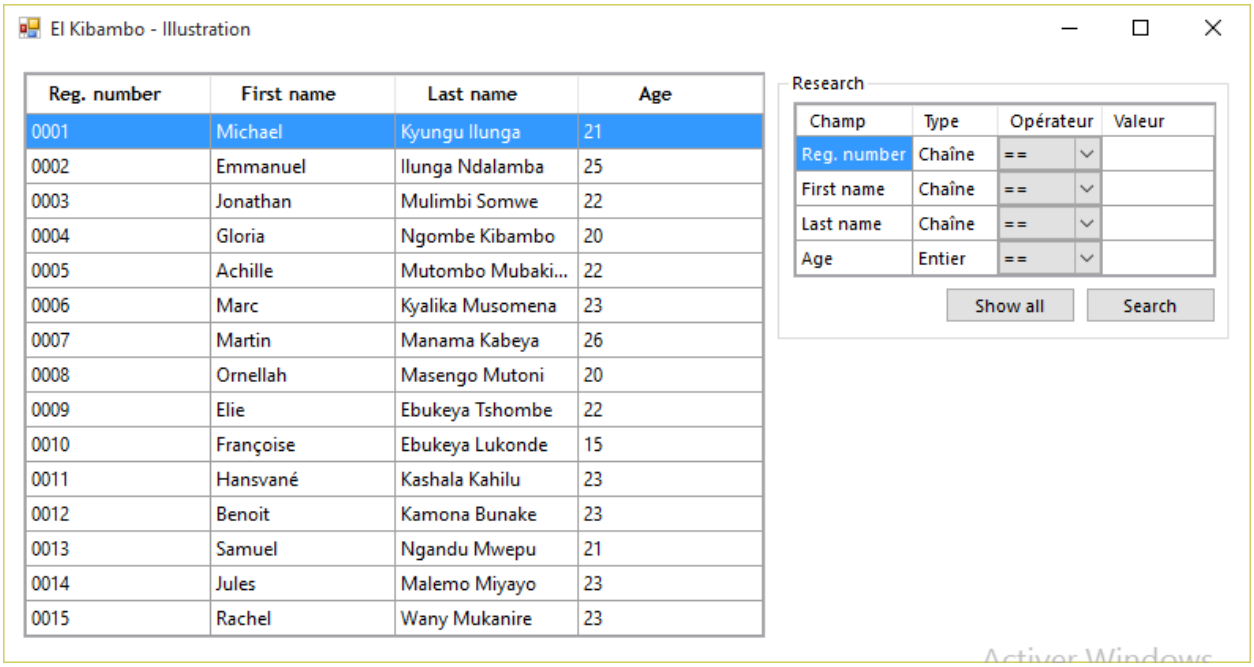
```
        gbSearch.Font = New Font("Segoe UI", 8.25!, FontStyle.Regular, GraphicsUnit.Point,
CType(0, Byte))
        gbSearch.Location = New Point(451, 12)
        gbSearch.Size = New Size(286, 170)
        gbSearch.TabIndex = 2
        gbSearch.TabStop = False
        gbSearch.Text = "Research"

        '
        ' Form1
        '
        AutoScaleDimensions = New SizeF(6.0!, 13.0!)
        AutoScaleMode = AutoScaleMode.Font
        BackColor = Color.White
        ClientSize = New Size(750, 360)
        Controls.Add(gbSearch)
        Controls.Add(dgvPersons)
        Text = "El Kibambo - Illustration"
        StartPosition = FormStartPosition.CenterScreen
        CType(dgvPersons, ISupportInitialize).EndInit()
        CType(dgvSearch, ISupportInitialize).EndInit()
        gbSearch.ResumeLayout(False)
        ResumeLayout(False)

    End Sub

    Private Sub InitializeDgvData()
        dgvPersons.Columns.Add("registration_number", "Reg. number")
        dgvPersons.Columns.Add("first_name", "First name")
        dgvPersons.Columns.Add("last_name", "Last name")
        dgvPersons.Columns.Add("age", "Age")
    End Sub

    Private Sub LoadData()
        dgvPersons.Rows.Add("0001", "Michael", "Kyungu Ilunga", 21)
        dgvPersons.Rows.Add("0002", "Emmanuel", "Ilunga Ndalamba", 25)
        dgvPersons.Rows.Add("0003", "Jonathan", "Mulimbi Somwe", 22)
        dgvPersons.Rows.Add("0004", "Gloria", "Ngombe Kibambo", 20)
        dgvPersons.Rows.Add("0005", "Achille", "Mutombo Mubakilay", 22)
        dgvPersons.Rows.Add("0006", "Marc", "Kyalika Musomena", 23)
        dgvPersons.Rows.Add("0007", "Martin", "Manama Kabeya", 26)
        dgvPersons.Rows.Add("0008", "Ornellah", "Masengo Mutoni", 20)
        dgvPersons.Rows.Add("0009", "Elie", "Ebukeya Tshombe", 22)
        dgvPersons.Rows.Add("0010", "Françoise", "Ebukeya Lukonde", 15)
        dgvPersons.Rows.Add("0011", "Hansvané", "Kashala Kahilu", 23)
        dgvPersons.Rows.Add("0012", "Benoit", "Kamona Bunake", 23)
        dgvPersons.Rows.Add("0013", "Samuel", "Ngandu Mwepu", 21)
        dgvPersons.Rows.Add("0014", "Jules", "Malemo Miyayo", 23)
        dgvPersons.Rows.Add("0015", "Rachel", "Wany Mukanire", 23)
    End Sub

    Private Sub Search(row As DataGridViewRow)
        ' We choose to make the lines that have not satisfied the search criteria invisible.
        ' But you can also do something else, for example change color, etc.
        row.Visible = False
    End Sub

    Private Sub ShowAll(row As DataGridViewRow)
        ' As we have masked the lines that do not meet the search criteria during the
research, it seems logical that we can re-display them when we want to cancel the search made.
        ' We must therefore put the line to its initial state before the search.
```

```
        row.Visible = True
    End Sub

    Private Sub InitializeKibambo()
        Dim kcm As New KibamboColumn("registration_number", "Reg. number")
        kcm.Operators = {"==", "!="}

        Dim kca As New KibamboColumn("age", "Age", KibamboColumnType.Numeric, "Entier", {"==",
"<", ">", "<=", ">=", "!="})

        kibambo = New Kibambo(
            dgvPersons,
            dgvSearch,
            New KibamboColumn() {
                kcm,
                kca
            },
            AddressOf Search,
            AddressOf ShowAll,
            True
        )

        kibambo.Initialize()
    End Sub

    Private Sub btnSearch_Click(sender As Object, e As EventArgs)
        Try
            kibambo.Search()
        Catch ex As KibamboException
            Select Case ex.ErrorCode
                Case KibamboErrorCode.ValueMustBeNumeric
                    MessageBox.Show("You must enter a digital type value for the digital
column!", " Input error ", MessageBoxButtons.OK, MessageBoxIcon.Error)

                Case Else
                    MessageBox.Show("An error occurred during the search. The error message is
" + ex.Message + " !", " Error during search ", MessageBoxButtons.OK, MessageBoxIcon.Error)
            End Select
        End Try

    End Sub

    Private Sub btnShowAll_Click(sender As Object, e As EventArgs)
        kibambo.CancelSearch()
    End Sub
End Class
```

Run the application by pressing the "Start" button or with the "F5" key:


KibamboTest - Microsoft Visual Studio
Imports ElKibambo
Imports System.ComponentModel
Public Class Form1
Dim dgvPersons As DataGridView
Dim dgvSearch As DataGridView
Dim gbSearch As GroupBox
Dim btnShowAll As Button
Dim btnSearch As Button
Dim kibambo As Kibambo
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
InitializeControls()
InitializeDgvData()
LoadData()
InitializeKibambo()
End Sub
Private Sub InitializeControls()
Dim dgvcs1 As New DataGridViewCellStyle
Dim dgvcs2 As New DataGridViewCellStyle
dgvPersons = New DataGridView()
dgvSearch = New DataGridView()
gbSearch = New GroupBox()
btnShowAll = New Button()
btnSearch = New Button()

Result:


El Kibambo - Illustration
Reg. number
First name
Last name
Age
0001 Michael Kyungu Ilunga 21
0002 Emmanuel Ilunga Ndalamba 25
0003 Jonathan Mulimbi Somwe 22
0004 Gloria Ngombe Kibambo 20
0005 Achille Mutombo Mubaki... 22
0006 Marc Kyalika Musomena 23
0007 Martin Manama Kabeya 26
0008 Ornellah Masengo Mutoni 20
0009 Elie Ebukeya Tshombe 22
0010 Françoise Ebukeya Lukonde 15
0011 Hansvané Kashala Kahilu 23
0012 Benoit Kamona Bunake 23
0013 Samuel Ngandu Mwepu 21
0014 Jules Malemo Miyayo 23
0015 Rachel Wany Mukanire 23
Research
Champ
Type
Opérateur
Valeur
Reg. number Chaîne ==
First name Chaîne ==
Last name Chaîne ==
Age Entier ==
Show all
Search

History

The El Kibambo library was born from an observation made by Michael KYUNGU ILUNGA, then a second graduate of the Pedagogical Institute of Lubumbashi. Its observation was the way in which the search features were implemented in Windows Forms programs. He has seen how his colleagues and promotional students (superior also) managed to implement the research features, and how they do not give the end user the possibility of performing multicolumn searches on data grids in Windows Forms .NET programs with the VB.NET programming language, but also C#.

By the way, imagine a data grid containing the data on people, for example "number", "name", "first name". At the ISP/L'SHI, the search features allowed only one search on the column serving as an identifier, such as the "number" column in the example above. For others, you can search on any column, however on one and only one column. So in the example above, it's up to you to choose, do you want to search by "number"? by "name"? or by "first name"? But never on more than one column at a time, and the search tests were limited to pure equality.

For Michael Kyungu Ilunga, the idea was to allow the end user to perform multicolumn searches, and whose type of comparison is not limited to equality.

During a practical work with the purpose of creating a complete management program, practical work of the VB.NET course then provided by the Kitenge Kalume Kitenge Assistant, Michael Kyungu Ilunga decided to take action but the result was not praise.

He decided to appeal to two of his colleagues to meet the challenge together: Benoit KAMONA BUNAKE and Samuel NGANDU MWEPU. For two days, Michael Kyungu Ilunga and Benoit Kamona Bunake (Samuel Ngandu Mwepu did not respond favorably) worked on a project so named "Advanced Search".

This was a Windows Forms program written in C# that would allow multicolumn search and with several comparison tests on data from relational databases (with the MySQL database management system). At startup, it asks you to choose a table from a database, database that has been hard-specified in the code. You choose a table, it displays a grid containing all the data in the chosen table and another grid allowing you to perform the search operations. To be able to search, the program generated an SQL request to perfect search. In this SQL query, a simple "where" clause was specified for the columns in which the user has entered a value but also the selected comparison operator. This strong dependence on SQL was the biggest difficulty.

This is so, Michael KYUNGU ILUNGA, based on this program, worked on a .NET library project that could be used without necessarily using SQL, a project which he called then "El Search.

The first versions were never released in alpha version, it was just beta versions limited to personal uses of Michael KYUNGU ILUNGA, and of one of his close peers of promotion, Hans KASHALA KAHILU. However, version 2.1 is the first alpha version of the El Kibambo library.

The library was officially announced on January 5, 2021 under the name "El Search", the library has undergone several test phases and was renamed "El Kibambo" mid 2021.

The word "Kibambo" comes from Gloria NGOMBE KIBAMBO, a leading colleague at Michael KYUNGU ILUNGA. Why this name choice? Well, for a thousand and one reason [...] =)