



GEODATASOURCE .NET COMPONENT

Developed by Hexasoft Development Sdn. Bhd.

Copyright © 2008 GeoDataSource.com. All Rights Reserved.

Visit our Web site at
<http://www.geodatasource.com>

~ Geographic Data for Everyone ~

INTRODUCTION

GeoDataSource™ offers the official gazetteer database of 3 million cities, land features, water features, structural landmarks and other geographical features for immediate online download.

All databases complete with official feature names with variant spellings, latitude and longitude, region and country information and additional information span across 260+ countries or territories in 6 world continents.

Developers use this .NET component for projects related to:

- [Look Up GeoDataSource World Cities Database \(Basic, Premium and Gold Edition\)](#)
- [Measure the Distance between Two Cities](#)
- [Wildcard Searching](#)
- [Free Support](#)

The GeoDataSource .NET Control works with Microsoft(r) Access, SQL Server and mySQL to query the complete GeoDataSource World Cities Database (Basic, Premium and Gold Edition). You can use the software object to query database by city and also calculate the distance between two cities or two latitude and longitude sets. This component will help developers to speed up the integration of GeoDataSource into their projects. Furthermore, it comes with general distance calculation function to reduce the programming time.

GeoDataSource™ .NET component is written for .NET Framework and optimized for Windows® operating system performance.

SYSTEM REQUIREMENT

GeoDataSource™ .NET Component has been developed to run exclusively under Windows based machines with support for both WinForm and ASP.NET (for web form). This component required a local GeoDataSource™ database for lookup purpose. No Internet connection required to execute this component.

Minimum Configuration Requirements

This section describes the minimum configuration requirements for a computer where the .NET Framework 1.1 or later redistributable package is to be installed. If the minimum requirements are not met, GeoDataSource™ setup file might block the installation of the redistributable package. Specifically, note that you cannot install the .NET Framework redistributable package on a computer running the Microsoft Windows 95 operating system.

Minimum Platform Requirements

- Microsoft® Windows 98
- Microsoft® Windows 98 Second Edition
- Microsoft® Windows Millennium Edition (Windows Me)
- Microsoft® Windows NT 4 (Workstation or Server) with Service Pack 6a
- Microsoft® Windows 2000 (Professional, Server, or Advanced Server) with the latest Windows service pack and critical updates available from the Microsoft Security Web site.
- Microsoft® Windows XP (Home or Professional)
- Microsoft® Windows Server 2003 family
- Microsoft® Windows Vista

Minimum Configuration Requirements

Scenario	CPU Required	RAM Required
Client	Pentium 90 MHz*	32 MB**
Server	Pentium 133 MHz*	128 MB**

* Or the minimum CPU required running the operating system, whichever is higher.

** Or the minimum RAM required running the operating system, whichever is higher.

Quick Start Guide

In this document, you will go through these basic steps to use GeoDataSource™ .NET component:

1. Download and installing GeoDataSource™ .NET Component.
2. Test your .NET Component installation.
3. Start using GeoDataSource™ in your projects.
4. Sample codes for GeoDataSource™ .NET Component client.
5. Purchase license to remove random 5 second delay in demo version.
6. Update GeoDataSource™ Component database,
7. Upgrade or uninstall GeoDataSource™ .NET Component.

Note: For demo version, you will have a random 5-second delay. Registered version has unlimited real-time query without delay.

1. DOWNLOAD AND INSTALLING GEODATASOURCE™ .NET COMPONENT

1.1. Download Setup File

Please use the following URL to download the latest GeoDataSource™ .NET Component together with sample database.

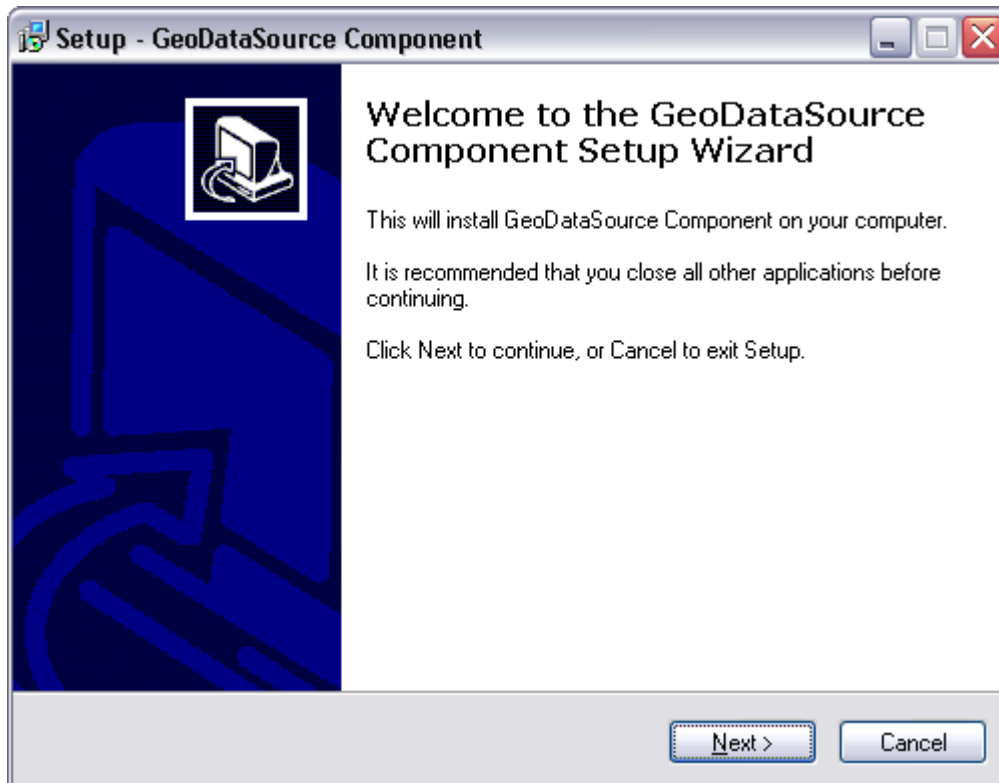
<http://www.geodatasource.com/download/GeoDataSourceDotNetComponent.ZIP>

1.2. Component Installation

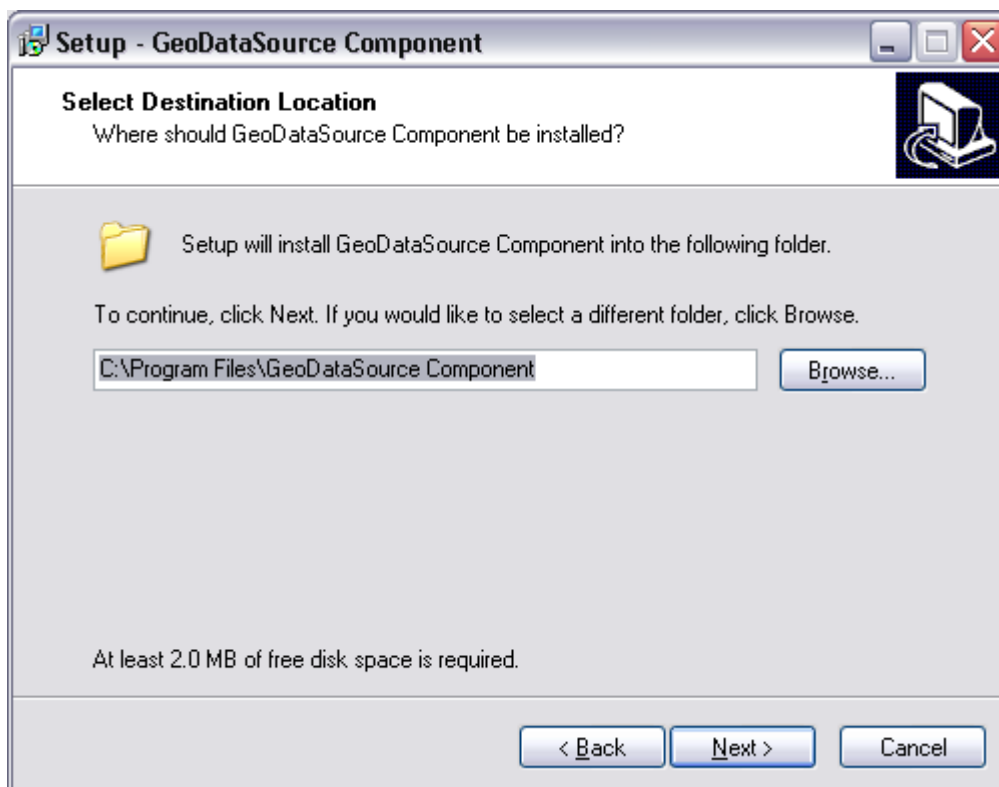
Installing the GeoDataSource™ .NET component system is very easy. You just need to execute the "Setup.exe".

General steps in installing GeoDataSource™ .NET component.

1. Unzip the "GeoDataSourceDotNetComponent.ZIP" into a temporary directory.
2. Execute the "Setup.exe".
3. Click Next button to continue.



4. Select the path you want to install. After selected click Next button to continue.



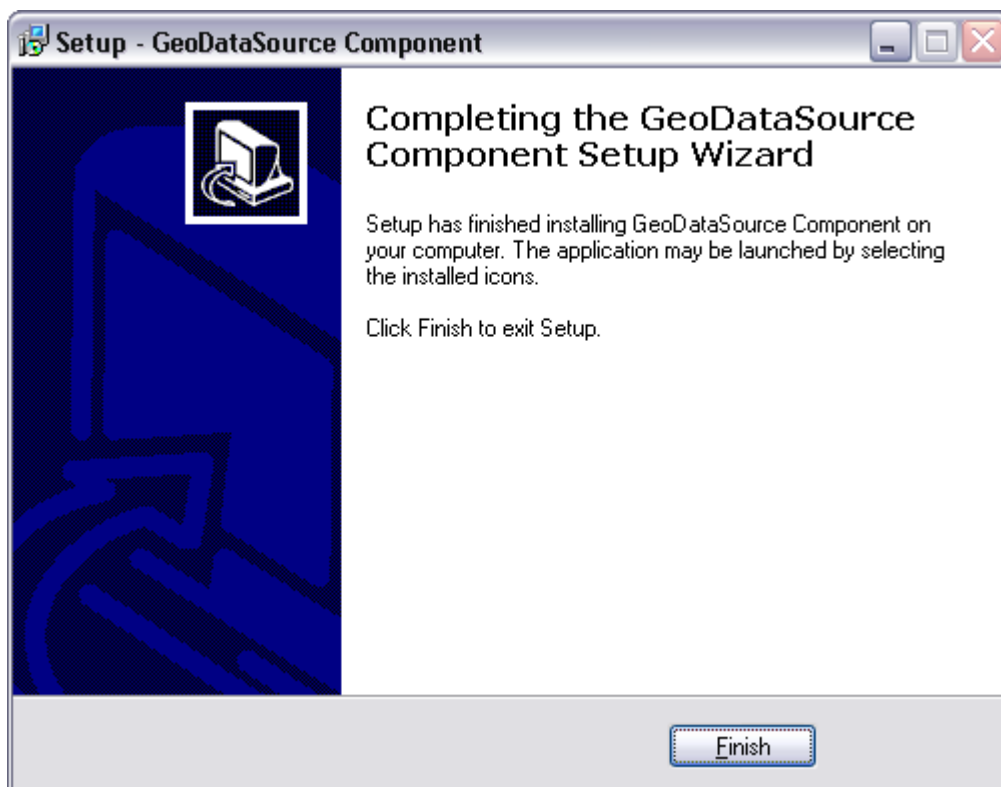
5. Click Next button to continue.



6. Click Install button to start installation.



7. Click Finish button to complete installation.



Note: ASP.NET Applications can be fully deployed through FTP. While this application comes with a DLL, you don't need to access the server in order to register (using REGSVR) it as it was required with previous ASP applications.

Just make sure to place the application's DLL in the /bin folder of your web site. Your hosting provider will be able to provide you with more information about the bin folder. **DO NOT TRY TO REGISTER THE DLL USING REGSVR.** Just place a copy of the DLL inside the /BIN folder at the root of your web site. If there is no /BIN folder at the root of your site, create it and place the DLL in there.

2. TESTING GEODATASOURCE™ .NET COMPONENT

2.1 Folders and Files Structure

You should find the following folders and files in the installation directory. The default installation directory is C:\Program Files\GeoDataSource Component\.

```
\SampleCodes\dotNET1.1\WinForm\VBnet\  
\SampleCodes\dotNET1.1\WinForm\C#\  
\SampleCodes\dotNET1.1\WebForm\VBnet\  
\SampleCodes\dotNET1.1\WebForm\C#\  
\SampleCodes\dotNET2.0_3.0_3.5\WinForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WinForm\C#\  
\SampleCodes\dotNET2.0_3.0_3.5\WebForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WebForm\C#\  
\Database\  
\DBScript\  
\GeoDataSourceComponent.dll  
\License_Agreement.txt  
\GeoDataSource_dotNET_Component.pdf  
\README.txt
```

2.2 Component Testing using WinForm

For desktop-based sample applications, please open the project or solution file to rebuild and execute the application.

Sample WinForm Written in VB.NET

```
\SampleCodes\dotNET1.1\WinForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WinForm\VBnet\
```

Sample WinForm Written in C#

```
\SampleCodes\dotNET1.1\WinForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WinForm\VBnet\
```

2.3 Component Testing using WebForm

For web-based sample applications, please copy the sample to a web directory that has configured to run web scripts. Then run these samples using your browser from local web server. Please make sure you have copied the GeoDataSourceComponent.DLL and sample database to the web application directory /bin.

Sample WebForm Written in VB.NET

```
\SampleCodes\dotNET1.1\WebForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WebForm\VBnet\
```

Sample WebForm Written in C#

```
\SampleCodes\dotNET1.1\WebForm\VBnet\  
\SampleCodes\dotNET2.0_3.0_3.5\WebForm\VBnet\
```

3. IMPLEMENTING GEODATASOURCE™ .NET COMPONENT

3.1. GeoDataSource™ Objects Reference

Object: `GeoDataSourceComponent.LookupClass.GeoDataSource`

Parameter	Description
REGION	Region Code. A code that determines the character mapping used in the full name field.
SUB_REGION	Sub-Region Code. A sub division of Region code.
UFI	Unique Feature Identifier. A number which uniquely identifies the feature. A feature with multiple names with share a same feature identifier.
UNI	Unique Name Identifier. A number which uniquely identifies a name.
DSG	Feature Designation Code. A two to five-character code used to identify the type of feature a name is applied to.
CC_FIPS	FIPS 10-4 Primary Country Code. A two alphabetic character FIPS 10-4 Primary Country Code uniquely identifying a geopolitical entity (countries, dependencies, and areas of special sovereignty).
CC_ISO	ISO 3166 Primary Country Code. A two alphabetic character ISO 3166 Primary Country Code uniquely identifying a geopolitical entity (countries, dependencies, and areas of special sovereignty).
FULL_NAME	Feature's Full Name. The full name is a complete name which identifies the named feature. It is comprised of the specific name, generic name, and any articles or prepositions.
FULL_NAME_ND	Feature's No Diacritics Full Name. Same as the full name but the diacritics and special characters are substituted with Roman characters.
SORT_NAME	Feature's Sort Name. A form of the full name which allows for easy sorting of the name into alpha-numeric sequence. It is comprised of the specific name, generic name, and any articles or prepositions. This field is all upper case with spaces, diacritics, and hyphens removed and numbers are substituted with lower case alphabetic characters.
ADM1_CODE	First-Order Administrative Division Code. A two alphanumeric character code uniquely identifying a primary administrative division of a country, such as a state in the United States.
ADM1_FULL_NAME	First-Order Administrative Division's Full Name. The full name is a complete name which identifies the first-order administrative division. It is comprised of the specific name, generic name, and any articles or prepositions.
ADM1_FULL_NAME_ND	First-Order Administrative Division's No Diacritics Full Name. Same as the division's full name but the diacritics and special characters are substituted with Roman characters.
ADM2_CODE	Second-Order Administrative Division Code. The name of a subdivision of a first-order administrative division, such as a county in the United States.
ADM2_FULL_NAME	Second-Order Administrative Division's Full Name. The full name is a complete name which identifies the second-order administrative division. It is comprised of the specific name, generic name, and any articles or prepositions.
ADM2_FULL_NAME_ND	Second-Order Administrative Division's No Diacritics Full Name. Same as the division's full name but the diacritics and special characters are substituted with Roman characters.

Parameter	Description
LATITUDE	Latitude in Decimal Degree: Latitude of the feature in \pm decimal degrees (WGS84). no sign (+) = North; negative sign (-) = South.
LONGITUDE	Longitude in Decimal Degree: Longitude of the feature in \pm decimal degrees (WGS84). no sign (+) = East; negative sign (-) = West.
DMS_LATITUDE	Latitude in Degree Minute Second: Latitude of the feature in \pm degrees, minutes, and seconds (WGS84). no sign (+) = North; negative sign (-) = South.
DMS_LONGITUDE	Longitude in Degree Minute Second: Longitude of the feature in \pm degrees, minutes, and seconds (WGS84). no sign (+) = East; negative sign (-) = West.
UTM	Universal Transverse Mercator: Universal Transverse Mercator coordinate grid reference.
JOG	Joint Operations Graphic: Joint Operations Graphic reference.

Method	Description	Return Type
Length()	Returns the number of records.	Integer
REGION(ByVal index As Integer)	Returns the region code according to record index.	Integer
SUB_REGION(ByVal index As Integer)	Returns the sub region code according to record index.	String
UFI(ByVal index As Integer)	Returns the Unique Feature Identifier value according to record index.	Integer
UNI(ByVal index As Integer)	Returns the Unique Name Identifier value according to record index.	Integer
DSG(ByVal index As Integer)	Returns the Feature Designation Code according to record index.	String
CC_FIPS(ByVal index As Integer)	Returns the FIPS 10-4 Primary Country Code according to record index.	String
CC_ISO(ByVal index As Integer)	Returns the ISO 3166 Primary Country Code according to record index.	String
FULL_NAME(ByVal index As Integer)	Returns the Feature's Full Name value according to record index.	String
FULL_NAME_ND(ByVal index As Integer)	Returns the Feature's No Diacritics Full Name value according to record index.	String
SORT_NAME(ByVal index As Integer)	Returns the Feature's Sort Name value according to record index.	String
ADM1_CODE(ByVal index As Integer)	Returns the First-Order Administrative Division Code according to record index.	String
ADM1_FULL_NAME(ByVal index As Integer)	Returns the counties First-Order Administrative Division's Full Name value according to record index.	String
ADM1_FULL_NAME_ND(ByVal index As Integer)	Returns the First-Order Administrative Division's No Diacritics Full Name value according to record index.	String
ADM2_CODE(ByVal index As Integer)	Returns the Second-Order Administrative Division Code according to record index.	String
ADM2_FULL_NAME(ByVal index As Integer)	Returns the Second-Order Administrative Division's Full Name value according to record index.	String
ADM2_FULL_NAME_ND(ByVal index As Integer)	Returns the Second-Order Administrative Division's No Diacritics Full Name value according to record index.	String
LATITUDE(ByVal index As Integer)	Returns the Latitude in Decimal Degree value according to record index.	String
LONGITUDE(ByVal index As Integer)	Returns the Longitude in Decimal Degree value according to record index.	Double

Methods	Description	Return Type
DMS_LATITUDE (ByVal index As Integer)	Returns the Latitude in Degree Minute Second value according to record index.	Double
DMS_LONGITUDE (ByVal index As Integer)	Returns the Longitude in Degree Minute Second value according to record index.	Double
UTM (ByVal index As Integer)	Returns the Universal Transverse Mercator value according to record index.	String
JOG (ByVal index As Integer)	Returns the Joint Operations Graphic value according to record index.	String
ErrMsg()	Returns the error message.	Integer

Object: GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet

Methods	Description	Return Type
DS_GeoDataSource()	Returns the Result of GeoDataSource value in dataset.	DataSet
ErrMsg()	Returns the error message.	Integer

Object: GeoDataSourceComponent.LookupClass.Distance

Methods	Description	Return Type
Distance()	Returns the Distance value.	Double
ErrMsg()	Returns the error message.	Integer

Note : Some of the results returned might return more than one result for a single field in a single record. In this case the result is appended in a single String separated by ‘/’.

In case of development using C#, the above documented methods can be assessed by adding get_ in front of the above documented methods.

App.config / Web.config

Parameter	Description
StrGeoDataSourceConnType	Define connection type for Microsoft Access, Microsoft SQL Server or MySQL. Example: access, mssql, mysql
StrGeoDataSourceConnAccess	Define Microsoft Access connection string for US database.
StrGeoDataSourceConnMySQL	Define MySQL connection string for US database.
StrGeoDataSourceConnMssql	Define Microsoft SQL Server connection string for US database.
StrGeoDataSourceLicenseKey	Define License key path.

Object: GeoDataSourceComponent.Component

Method	Description
<code>Query(ByVal strCityName As String, ByVal strCountry As String)</code>	Query GeoDataSource database by city name and country code. This method returns results in GeoDataSourceComponent.LookupClass.GeoDataSource object.
<code>QueryDataSet(ByVal strCityName As String, ByVal strCountry As String)</code>	Query GeoDataSource database by city name and country code. This method returns results in GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet object.
<code>DistanceByCity(ByVal strCityNameFrom As String, ByVal strCountryFrom As String, ByVal strCityNameTo As String, ByVal strCountryTo As String, Optional ByVal strUnit As Char = "K")</code>	Calculate distance between two cities. This method returns results in GeoDataSourceComponent.LookupClass.Distance object.
<code>DistanceByLatLong(ByVal dblLatitudeFrom As Object, ByVal dblLongitudeFrom As Object, ByVal dblLatitudeTo As Object, ByVal dblLongitudeTo As Object, Optional ByVal strUnit As Char = "K")</code>	Calculate distance between locations. This method returns results in GeoDataSourceComponent.LookupClass.Distance object.

3.2. GeoDataSource™ Database Setup

Microsoft SQL Server

1. Open Microsoft SQL Server Query Analyzer.
2. Enter the script below
CREATE DATABASE GEODATASOURCE
GO
3. Execute.
4. From Microsoft SQL Server Query Analyzer open the database script you want to import.
 - For GeoDataSource Basic
Open <your application path>\dbScript\MsSQL-CitiesBasic.sql
 - For GeoDataSource Premium
Open <your application path>\dbScript\MsSQL-CitiesPremium.sql
 - For GeoDataSource Gold
Open <your application path>\dbScript\MsSQL-CitiesGold.sql
5. Replace <your application path> to your local drive. (default: "C:\Program Files\GeoDataSource Component")
6. Execute.

MySQL

1. Open MySQL Command Line Client.
2. Enter the script below
CREATE DATABASE GEODATASOURCE;
3. Execute.
4. Open database script you want to import by NOTEPAD.
 - For GeoDataSource Basic
Open <your application path>\dbScript\MySQL-CitiesBasic.sql
 - For GeoDataSource Premium
Open <your application path>\dbScript\MySQL-CitiesPremium.sql
 - For GeoDataSource Gold
Open <your application path>\dbScript\MySQL-CitiesGold.sql
5. Replace <your application path> to your local drive (default: "C:\Program Files\GeoDataSource Component")
6. Copy the script and paste into MySQL Command Line Client.
7. Execute.

3.3. GeoDataSource™ Configuring Database Connection

- i. Open Web.Config/ App.Config
- ii. Add either ODBC-DBQ , ODBC-DSN, MySQL or MS SQL setting codes to Web.Config after </system.web> tag. Remember to update the database path or DSN in order to connect to the databases correctly

ODBC – DBQ (Database access through File Path)

You may change the database folder, but remember to point the database path correctly in the Web.config.

```
<appSettings>
  <add key="StrGeoDataSourceConnType" value="access" />
  <add key="StrGeoDataSourceConnAccess" value="Driver={Microsoft Access Driver (*.mdb)};
Dbq=[Your Database Path]\database\GEODATASOURCE-CITIES-GOLD-SAMPLES.MDB;Uid=;Pwd=" />
  <add key="StrGeoDataSourceLicenseKey" value="[Your Application Path]\License.key"/>
</appSettings>
```

ODBC – DSN (Database access through Data Source Name)

You need to create System DSN and ODBC Data Source for the following files:

- a) GEODATASOURCE-CITIES-GOLD.SAMPLE.MDB

```
<appSettings>
  <add key="StrGeoDataSourceConnType" value="access"/>
  <add key="StrGeoDataSourceConnAccess" value="DSN=GeoDataSource;Uid=;Pwd=" />
  <add key="StrLicenseKey" value="[Your Application Path]\License.key"/>
</appSettings>
```

MySQL

```
<appSettings>
  <add key="StrGeoDataSourceConnType" value="mysql" />
  <add key="StrGeoDataSourceConnMySQL" value="DRIVER={MySQL ODBC 3.51 Driver};SERVER=
localhost;DATABASE=geodatasource;USER=[User ID];PASSWORD=[Password];OPTION=3;" />
  <add key="StrLicenseKey" value="[Your Application Path]\License.key"/>
</appSettings>
```

Microsoft SQL Server

```
<appSettings>
  <add key="StrGeoDataSourceConnType" value="mssql" />
  <add key="StrGeoDataSourceConnMsSQL" value="Data Source=127.0.0.1;Initial Catalog=
geodatasource;User Id=[User ID];Password=[Password];" />
  <add key="StrLicenseKey" value="[Your Application Path]\License.key"/>
</appSettings>
```

4. GEODATASOURCE™ .NET COMPONENT SAMPLE CODES

4.1. Sample Codes in Visual Basic.NET (WinForm)

```
Imports System.Text
...
Private Sub btnQuery_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnQuery.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oGeoDataSource As GeoDataSourceComponent.LookupClass.GeoDataSource
    Dim strResult As New StringBuilder
    Dim i As Integer
    oGeoDataSource = oGeoDataSourceComponent.Query(Me.txtCity.Text, Me.cboCountry.SelectedItem)
    If oGeoDataSource.ErrMsg <> "" Then
        MessageBox.Show(oGeoDataSource.ErrMsg)
    Else
        For i = 0 To oGeoDataSource.Length - 1
            strResult.Append("REGION: " & oGeoDataSource.REGION(i) & vbNewLine)
            strResult.Append("SUB_REGION: " & oGeoDataSource.SUB_REGION(i) & vbNewLine)
            strResult.Append("UFI: " & oGeoDataSource.UFI(i) & vbNewLine)
            strResult.Append("UNI: " & oGeoDataSource.UNI(i) & vbNewLine)
            strResult.Append("DSG: " & oGeoDataSource.DSG(i) & vbNewLine)
            strResult.Append("CC_FIPS: " & oGeoDataSource.CC_FIPS(i) & vbNewLine)
            strResult.Append("CC_ISO: " & oGeoDataSource.CC_ISO(i) & vbNewLine)
            strResult.Append("FULL_NAME: " & oGeoDataSource.FULL_NAME(i) & vbNewLine)
            strResult.Append("FULL_NAME_ND: " & oGeoDataSource.FULL_NAME_ND(i) & vbNewLine)
            strResult.Append("SORT_NAME: " & oGeoDataSource.SORT_NAME(i) & vbNewLine)
            strResult.Append("ADM1_CODE: " & oGeoDataSource.ADM1_CODE(i) & vbNewLine)
            strResult.Append("ADM1_FULL_NAME: " & oGeoDataSource.ADM1_FULL_NAME(i) & vbNewLine)
            strResult.Append("ADM1_FULL_NAME_ND: " & oGeoDataSource.ADM1_FULL_NAME_ND(i) & vbNewLine)
            strResult.Append("ADM2_CODE: " & oGeoDataSource.ADM2_CODE(i) & vbNewLine)
            strResult.Append("ADM2_FULL_NAME: " & oGeoDataSource.ADM2_FULL_NAME(i) & vbNewLine)
            strResult.Append("ADM2_FULL_NAME_ND: " & oGeoDataSource.ADM2_FULL_NAME_ND(i) & vbNewLine)
            strResult.Append("LATITUDE: " & oGeoDataSource.LATITUDE(i) & vbNewLine)
            strResult.Append("LONGITUDE: " & oGeoDataSource.LONGITUDE(i) & vbNewLine)
            strResult.Append("DMS_LATITUDE: " & oGeoDataSource.DMS_LATITUDE(i) & vbNewLine)
            strResult.Append("DMS_LONGITUDE: " & oGeoDataSource.DMS_LONGITUDE(i) & vbNewLine)
            strResult.Append("UTM: " & oGeoDataSource.UTM(i) & vbNewLine)
            strResult.Append("JOG: " & oGeoDataSource.JOG(i) & vbNewLine & vbNewLine)
        Next
        If strResult.ToString <> "" Then
            Me.txtResult.Text = strResult.ToString
        End If
    End Sub

Private Sub btnQueryDataSet_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnQueryDataSet.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oGeoDataSource As GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet
    oGeoDataSource = oGeoDataSourceComponent.QueryDataSet(Me.txtCityDataSet.Text,
Me.cboCountryDataSet.SelectedItem)
    If oGeoDataSource.ErrMsg <> "" Then
        MessageBox.Show(oGeoDataSource.ErrMsg)
    Else
        Me.dgResult.DataSource = oGeoDataSource.DS_GeoDataSource.Tables(0)
    End If
End Sub
```

```
Private Sub btnCalculateByCity_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnCalculateByCity.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oDistance As GeoDataSourceComponent.LookupClass.Distance
    oDistance = oGeoDataSourceComponent.DistanceByCity(Me.txtCityFrom.Text,
Me.cboCountryFrom.SelectedItem, Me.txtCityTo.Text, Me.cboCountryTo.SelectedItem,
CStr(Me.cboUnitByCity.SelectedItem).Substring(0, 1))
    If oDistance.ErrMsg <> "" Then
        MessageBox.Show(oDistance.ErrMsg)
    Else
        Me.lblDistanceByCity.Text = "Distance between " & Me.txtCityFrom.Text.Trim & " (" &
Me.cboCountryFrom.SelectedItem & ") and " & Me.txtCityTo.Text.Trim & " (" &
Me.cboCountryTo.SelectedItem & ") is " & oDistance.Distance.ToString("0.000") & " " &
Me.cboUnitByCity.SelectedItem
    End If
End Sub

Private Sub btnCalculateByLatLong_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnCalculateByLatLong.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oDistance As GeoDataSourceComponent.LookupClass.Distance
    oDistance = oGeoDataSourceComponent.DistanceByLatLong(Me.txtLatitudeFrom.Text,
Me.txtLongitudeFrom.Text, Me.txtLatitudeTo.Text, Me.txtLongitudeTo.Text,
CStr(Me.cboUnitByLatLong.SelectedItem).Substring(0, 1))
    If oDistance.ErrMsg <> "" Then
        MessageBox.Show(oDistance.ErrMsg)
    Else
        Me.lblDistanceByLatLong.Text = "Distance between " & Me.txtLatitudeFrom.Text.Trim & ", " &
Me.txtLongitudeFrom.Text.Trim & " and " & Me.txtLatitudeTo.Text.Trim & ", " &
Me.txtLongitudeTo.Text.Trim & " is " & oDistance.Distance.ToString("0.000") & " " &
Me.cboUnitByLatLong.SelectedItem
    End If
End Sub
```

4.2. Sample Codes in C# (WinForm)

```
Using System.Text;
private void btnQuery_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.GeoDataSource oGeoDataSource;
    StringBuilder strResult = new StringBuilder();
    int i;
    oGeoDataSource = oGeoDataSourceComponent.Query(this.txtCity.Text.ToString(),
this.cboCountry.SelectedItem.ToString());
    if (oGeoDataSource.ErrMsg != "")
    {
        MessageBox.Show(oGeoDataSource.ErrMsg);
    }
    else
    {
        for (i = 0; i <= oGeoDataSource.Length - 1; i++)
        {
            strResult.Append("REGION: " + oGeoDataSource.get_REGION(i) +
Environment.NewLine);
            strResult.Append("SUB_REGION: " + oGeoDataSource.get_SUB_REGION(i) +
Environment.NewLine);
            strResult.Append("UFI: " + oGeoDataSource.get_UFI(i) + Environment.NewLine);
            strResult.Append("UNI: " + oGeoDataSource.get_UNI(i) + Environment.NewLine);
            strResult.Append("DSG: " + oGeoDataSource.get_DSG(i) + Environment.NewLine);
            strResult.Append("CC_FIPS: " + oGeoDataSource.get_CC_FIPS(i) +
Environment.NewLine);
            strResult.Append("CC_ISO: " + oGeoDataSource.get_CC_ISO(i) +
Environment.NewLine);
            strResult.Append("FULL_NAME: " + oGeoDataSource.get_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("FULL_NAME_ND: " + oGeoDataSource.get_FULL_NAME_ND(i) +
Environment.NewLine);
            strResult.Append("SORT_NAME: " + oGeoDataSource.get_SORT_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM1_CODE: " + oGeoDataSource.get_ADM1_CODE(i) +
Environment.NewLine);
            strResult.Append("ADM1_FULL_NAME: " + oGeoDataSource.get_ADM1_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM1_FULL_NAME_ND: " + oGeoDataSource.get_ADM1_FULL_NAME_ND(i)
+ Environment.NewLine);
            strResult.Append("ADM2_CODE: " + oGeoDataSource.get_ADM2_CODE(i) +
Environment.NewLine);
            strResult.Append("ADM2_FULL_NAME: " + oGeoDataSource.get_ADM2_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM2_FULL_NAME_ND: " + oGeoDataSource.get_ADM2_FULL_NAME_ND(i)
+ Environment.NewLine);
            strResult.Append("LATITUDE: " + oGeoDataSource.get_LATITUDE(i) +
Environment.NewLine);
            strResult.Append("LONGITUDE: " + oGeoDataSource.get_LONGITUDE(i) +
Environment.NewLine);
            strResult.Append("DMS_LATITUDE: " + oGeoDataSource.get_DMS_LATITUDE(i) +
Environment.NewLine);
            strResult.Append("DMS_LONGITUDE: " + oGeoDataSource.get_DMS_LONGITUDE(i) +
Environment.NewLine);
            strResult.Append("UTM: " + oGeoDataSource.get_UTM(i) + Environment.NewLine);
            strResult.Append("JOG: " + oGeoDataSource.get_JOG(i) + Environment.NewLine +
Environment.NewLine);
        }
        if (strResult.ToString() != "")
        {
            this.txtResult.Text = strResult.ToString();
        }
    }
}
```

```
private void btnQueryDataSet_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet oGeoDataSource;
    StringBuilder strResult = new StringBuilder();
    oGeoDataSource = oGeoDataSourceComponent.QueryDataSet(this.txtCityDataSet.Text.ToString(),
this.cboCountryDataSet.SelectedItem.ToString());
    if (oGeoDataSource.ErrMsg != "")
    {
        MessageBox.Show(oGeoDataSource.ErrMsg);
    }
    else
    {
        this.dgResult.DataSource = oGeoDataSource.DS_GeoDataSource.Tables[0];
    }
}

private void btnCalculateByCity_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.Distance oDistance;
    oDistance = oGeoDataSourceComponent.DistanceByCity(this.txtCityFrom.Text,
this.cboCountryFrom.SelectedItem.ToString(), this.txtCityTo.Text,
this.cboCountryTo.SelectedItem.ToString(),
char.Parse(this.cboUnitByCity.SelectedItem.ToString().Substring(0, 1)));
    if (oDistance.ErrMsg != "")
    {
        MessageBox.Show(oDistance.ErrMsg);
    }
    else
    {
        this.lblDistanceByCity.Text = "Distance between " + this.txtCityFrom.Text.Trim() + " ("
+ this.cboCountryFrom.SelectedItem + ") and " + this.txtCityTo.Text.Trim() + " (" +
this.cboCountryTo.SelectedItem + ") is " + oDistance.Distance.ToString("0.000") + " " +
this.cboUnitByCity.SelectedItem;
    }
}

private void btnCalculateByLatLong_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.Distance oDistance;
    oDistance = oGeoDataSourceComponent.DistanceByLatLong(this.txtLatitudeFrom.Text.ToString(),
this.txtLongitudeFrom.Text.ToString(), (object)this.txtLatitudeTo.Text,
(object)this.txtLongitudeTo.Text, char.Parse(this.cboUnitByLatLong.SelectedItem.ToString().Substring(0,
1)));
    if (oDistance.ErrMsg != "")
    {
        MessageBox.Show(oDistance.ErrMsg);
    }
    else
    {
        this.lblDistanceByLatLong.Text = "Distance between " + this.txtLatitudeFrom.Text.Trim()
+ ", " + this.txtLongitudeFrom.Text.Trim() + " and " + this.txtLatitudeTo.Text.Trim() + ", " +
this.txtLongitudeTo.Text.Trim() + " is " + oDistance.Distance.ToString("0.000") + " " +
this.cboUnitByLatLong.SelectedItem;
    }
}
```

4.3. Sample Codes in Visual Basic.Net (WebForm)

```
Imports GeoDataSourceComponent
Imports System.Text
...
Private Sub btnQuery_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnQuery.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oGeoDataSource As GeoDataSourceComponent.LookupClass.GeoDataSource
    Dim strResult As New StringBuilder
    Dim i As Integer
    Me.txtResult.Text = ""
    Me.txtResult.Visible = False
    Me.dgResult.Visible = False
    oGeoDataSource = oGeoDataSourceComponent.Query(Me.txtCity.Text,
Me.ddlCountry.SelectedItem.Value)
    If oGeoDataSource.ErrMsg <> "" Then
        Response.Write(oGeoDataSource.ErrMsg)
    Else
        For i = 0 To oGeoDataSource.Length - 1
            strResult.Append("REGION: " & oGeoDataSource.REGION(i) & vbNewLine)
            strResult.Append("SUB_REGION: " & oGeoDataSource.SUB_REGION(i) & vbNewLine)
            strResult.Append("UFI: " & oGeoDataSource.UFI(i) & vbNewLine)
            strResult.Append("UNI: " & oGeoDataSource.UNI(i) & vbNewLine)
            strResult.Append("DSG: " & oGeoDataSource.DSG(i) & vbNewLine)
            strResult.Append("CC_FIPS: " & oGeoDataSource.CC_FIPS(i) & vbNewLine)
            strResult.Append("CC_ISO: " & oGeoDataSource.CC_ISO(i) & vbNewLine)
            strResult.Append("FULL_NAME: " & oGeoDataSource.FULL_NAME(i) & vbNewLine)
            strResult.Append("FULL_NAME_ND: " & oGeoDataSource.FULL_NAME_ND(i) & vbNewLine)
            strResult.Append("SORT_NAME: " & oGeoDataSource.SORT_NAME(i) & vbNewLine)
            strResult.Append("ADM1_CODE: " & oGeoDataSource.ADM1_CODE(i) & vbNewLine)
            strResult.Append("ADM1_FULL_NAME: " & oGeoDataSource.ADM1_FULL_NAME(i) & vbNewLine)
            strResult.Append("ADM1_FULL_NAME_ND: " & oGeoDataSource.ADM1_FULL_NAME_ND(i) &
vbNewLine)
            strResult.Append("ADM2_CODE: " & oGeoDataSource.ADM2_CODE(i) & vbNewLine)
            strResult.Append("ADM2_FULL_NAME: " & oGeoDataSource.ADM2_FULL_NAME(i) & vbNewLine)
            strResult.Append("ADM2_FULL_NAME_ND: " & oGeoDataSource.ADM2_FULL_NAME_ND(i) &
vbNewLine)
            strResult.Append("LATITUDE: " & oGeoDataSource.LATITUDE(i) & vbNewLine)
            strResult.Append("LONGITUDE: " & oGeoDataSource.LONGITUDE(i) & vbNewLine)
            strResult.Append("DMS_LATITUDE: " & oGeoDataSource.DMS_LATITUDE(i) & vbNewLine)
            strResult.Append("DMS_LONGITUDE: " & oGeoDataSource.DMS_LONGITUDE(i) & vbNewLine)
            strResult.Append("UTM: " & oGeoDataSource.UTM(i) & vbNewLine)
            strResult.Append("JOG: " & oGeoDataSource.JOG(i) & vbNewLine & vbNewLine)
        Next
        If strResult.ToString <> "" Then
            Me.dgResult.Visible = False
            Me.txtResult.Visible = True
            Me.txtResult.Text = strResult.ToString
        End If
    End If
End Sub
```

```
Private Sub btnQueryDataSet_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
btnQueryDataSet.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oGeoDataSource As GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet
    Dim strResult As New StringBuilder
    Me.dgResult.DataSource = Nothing
    Me.dgResult.DataBind()
    Me.txtResult.Visible = False
    Me.dgResult.Visible = False
    oGeoDataSource = oGeoDataSourceComponent.QueryDataSet(Me.txtCityDataSet.Text,
Me.ddlCountryDataSet.SelectedItem.Value)
    If oGeoDataSource.ErrMsg <> "" Then
        Response.Write(oGeoDataSource.ErrMsg)
    Else
        Me.dgResult.Visible = True
        Me.txtResult.Visible = False
        Me.dgResult.DataSource = oGeoDataSource.DS_GeoDataSource
        Me.dgResult.DataBind()
    End If
End Sub
Private Sub btnDistanceByCity_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnDistanceByCity.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oDistance As GeoDataSourceComponent.LookupClass.Distance
    Me.txtResult.Text = ""
    Me.txtResult.Visible = False
    Me.dgResult.Visible = False
    oDistance = oGeoDataSourceComponent.DistanceByCity(Me.txtCityFrom.Text,
Me.ddlCountryFrom.SelectedItem.Value, Me.txtCityTo.Text, Me.ddlCountryTo.SelectedItem.Value,
CStr(Me.ddlUnitByCity.SelectedItem.Value).Substring(0, 1))
    If oDistance.ErrMsg <> "" Then
        Response.Write(oDistance.ErrMsg)
    Else
        Me.txtResult.Visible = True
        Me.dgResult.Visible = False
        Me.txtResult.Text = "Distance between " & Me.txtCityFrom.Text.Trim & " (" &
Me.ddlCountryFrom.SelectedItem.Value & ") and " & Me.txtCityTo.Text.Trim & " (" &
Me.ddlCountryTo.SelectedItem.Value & ") is " & oDistance.Distance.ToString("0.000") & " " &
Me.ddlUnitByCity.SelectedItem.Value
    End If
End Sub
Private Sub btnDistanceByLatLong_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles btnDistanceByLatLong.Click
    Dim oGeoDataSourceComponent As New GeoDataSourceComponent.Component
    Dim oDistance As GeoDataSourceComponent.LookupClass.Distance
    Me.txtResult.Text = ""
    Me.txtResult.Visible = False
    Me.dgResult.Visible = False
    oDistance = oGeoDataSourceComponent.DistanceByLatLong(Me.txtLatitudeFrom.Text,
Me.txtLongitudeFrom.Text, Me.txtLatitudeTo.Text, Me.txtLongitudeTo.Text,
CStr(Me.ddlUnitByLatLong.SelectedItem.Value).Substring(0, 1))
    If oDistance.ErrMsg <> "" Then
        Response.Write(oDistance.ErrMsg)
    Else
        Me.txtResult.Visible = True
        Me.txtResult.Text = "Distance between " & Me.txtLatitudeFrom.Text.Trim & ", " &
Me.txtLongitudeFrom.Text.Trim & " and " & Me.txtLatitudeTo.Text.Trim & ", " &
Me.txtLongitudeTo.Text.Trim & " is " & oDistance.Distance.ToString("0.000") & " " &
Me.ddlUnitByLatLong.SelectedItem.Value
    End If
End Sub
```

4.4. Sample Codes in C# (WebForm)

```
using System.Text;
...
private void btnQuery_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.GeoDataSource oGeoDataSource;
    StringBuilder strResult = new StringBuilder();
    int i;
    this.txtResult.Text = "";
    this.txtResult.Visible = false;
    this.dgResult.Visible = false;
    oGeoDataSource = oGeoDataSourceComponent.Query(this.txtCity.Text,
this.ddlCountry.SelectedItem.Value);
    if (oGeoDataSource.ErrMsg != "")
    {
        Response.Write(oGeoDataSource.ErrMsg);
    }
    else
    {
        for (i = 0; i <= oGeoDataSource.Length - 1; i++)
        {
            strResult.Append("REGION: " + oGeoDataSource.get_REGION(i) + Environment.NewLine);
            strResult.Append("SUB_REGION: " + oGeoDataSource.get_SUB_REGION(i) +
Environment.NewLine);
            strResult.Append("UFI: " + oGeoDataSource.get_UFI(i) + Environment.NewLine);
            strResult.Append("UNI: " + oGeoDataSource.get_UNI(i) + Environment.NewLine);
            strResult.Append("DSG: " + oGeoDataSource.get_DSG(i) + Environment.NewLine);
            strResult.Append("CC_FIPS: " + oGeoDataSource.get_CC_FIPS(i) +
Environment.NewLine);
            strResult.Append("CC_ISO: " + oGeoDataSource.get_CC_ISO(i) + Environment.NewLine);
            strResult.Append("FULL_NAME: " + oGeoDataSource.get_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("FULL_NAME_ND: " + oGeoDataSource.get_FULL_NAME_ND(i) +
Environment.NewLine);
            strResult.Append("SORT_NAME: " + oGeoDataSource.get_SORT_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM1_CODE: " + oGeoDataSource.get_ADM1_CODE(i) +
Environment.NewLine);
            strResult.Append("ADM1_FULL_NAME: " + oGeoDataSource.get_ADM1_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM1_FULL_NAME_ND: " + oGeoDataSource.get_ADM1_FULL_NAME_ND(i) +
Environment.NewLine);
            strResult.Append("ADM2_CODE: " + oGeoDataSource.get_ADM2_CODE(i) +
Environment.NewLine);
            strResult.Append("ADM2_FULL_NAME: " + oGeoDataSource.get_ADM2_FULL_NAME(i) +
Environment.NewLine);
            strResult.Append("ADM2_FULL_NAME_ND: " + oGeoDataSource.get_ADM2_FULL_NAME_ND(i) +
Environment.NewLine);
            strResult.Append("LATITUDE: " + oGeoDataSource.get_LATITUDE(i) +
Environment.NewLine);
            strResult.Append("LONGITUDE: " + oGeoDataSource.get_LONGITUDE(i) +
Environment.NewLine);
            strResult.Append("DMS_LATITUDE: " + oGeoDataSource.get_DMS_LATITUDE(i) +
Environment.NewLine);
            strResult.Append("DMS_LONGITUDE: " + oGeoDataSource.get_DMS_LONGITUDE(i) +
Environment.NewLine);
            strResult.Append("UTM: " + oGeoDataSource.get_UTM(i) + Environment.NewLine);
            strResult.Append("JOG: " + oGeoDataSource.get_JOG(i) + Environment.NewLine +
Environment.NewLine);
        }
        if (strResult.ToString() != "")
        {
            this.dgResult.Visible = false;
            this.txtResult.Visible = true;
            this.txtResult.Text = strResult.ToString();
        }
    }
}
```

```
private void btnQueryDataSet_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.GeoDataSourceDataSet oGeoDataSource;
    StringBuilder strResult = new StringBuilder();
    this.dgResult.DataSource = null;
    this.dgResult.DataBind();
    this.txtResult.Visible = false;
    this.dgResult.Visible = false;
    oGeoDataSource = oGeoDataSourceComponent.QueryDataSet(this.txtCityDataSet.Text,
this.ddlCountryDataSet.SelectedItem.Value);
    if (oGeoDataSource.ErrMsg != "")
    {
        Response.Write(oGeoDataSource.ErrMsg);
    }
    else
    {
        this.dgResult.Visible = true;
        this.txtResult.Visible = false;
        this.dgResult.DataSource = oGeoDataSource.DS_GeoDataSource;
        this.dgResult.DataBind();
    }
}
private void btnDistanceByCity_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.Distance oDistance;
    this.txtResult.Text = "";
    this.txtResult.Visible = false;
    this.dgResult.Visible = false;
    oDistance = oGeoDataSourceComponent.DistanceByCity(this.txtCityFrom.Text,
this.ddlCountryFrom.SelectedItem.Value, this.txtCityTo.Text, this.ddlCountryTo.SelectedItem.Value,
char.Parse(this.ddlUnitByLatLong.SelectedItem.Value.ToString().Substring(0, 1)));
    if (oDistance.ErrMsg != "")
    {
        Response.Write(oDistance.ErrMsg);
    }
    else
    {
        this.txtResult.Visible = true;
        this.dgResult.Visible = false;
        this.txtResult.Text = "Distance between " + this.txtCityFrom.Text.Trim() + " (" +
this.ddlCountryFrom.SelectedItem.Value + ") and " + this.txtCityTo.Text.Trim() + " (" +
this.ddlCountryTo.SelectedItem.Value + ") is " + oDistance.Distance.ToString("0.000") + " " +
this.ddlUnitByCity.SelectedItem.Value;
    }
}
private void btnDistanceByLatLong_Click(object sender, System.EventArgs e)
{
    GeoDataSourceComponent.Component oGeoDataSourceComponent = new
GeoDataSourceComponent.Component();
    GeoDataSourceComponent.LookupClass.Distance oDistance;
    this.txtResult.Text = "";
    this.txtResult.Visible = false;
    this.dgResult.Visible = false;
    oDistance = oGeoDataSourceComponent.DistanceByLatLong(this.txtLatitudeFrom.Text,
this.txtLongitudeFrom.Text, this.txtLatitudeTo.Text, this.txtLongitudeTo.Text,
char.Parse(this.ddlUnitByLatLong.SelectedItem.Value.ToString().Substring(0, 1)));
    if (oDistance.ErrMsg != "")
    {
        Response.Write(oDistance.ErrMsg);
    }
    else
    {
        this.txtResult.Visible = true;
        this.txtResult.Text = "Distance between " + this.txtLatitudeFrom.Text.Trim() + ", " +
this.txtLongitudeFrom.Text.Trim() + " and " + this.txtLatitudeTo.Text.Trim() + ", " +
this.txtLongitudeTo.Text.Trim() + " is " + oDistance.Distance.ToString("0.000") + " " +
this.ddlUnitByLatLong.SelectedItem.Value;
    }
}
}
```

5. PURCHASE GEODATASOURCE™ .NET COMPONENT

5.1. License Agreement

A license is required for each machine the product is installed on, including development or staging machines. Please refer to the end of this document for the complete license agreement.

5.2. Purchase Instructions

Proceed to our order page. Fill out the online form and choose the correct number of licenses. Once your order has been approved, you will receive your license file (License.Key) immediately through email. Save the license file to the same directory as GeoDataSourceComponent.dll in WinForm or /bin directory for WebForm to remove the random 5-second query delay.

Please visit <http://www.geodatasource.com> for online order.

6. UPDATE COMPONENT DATABASE

6.1. Update Component Internal Database

The GeoDataSource™ .NET Component depends to an internal database for lookup purpose. All users with valid license will be allowed to download the monthly updates from the GeoDataSource™ download area during subscription period. To update, users need to replace the database with the latest one from the download area. The component will use the latest database once the local copy has been updated. Please refer to the welcome email for more information regarding download updates and account information.

7. UPGRADE OR UNINSTALL COMPONENT

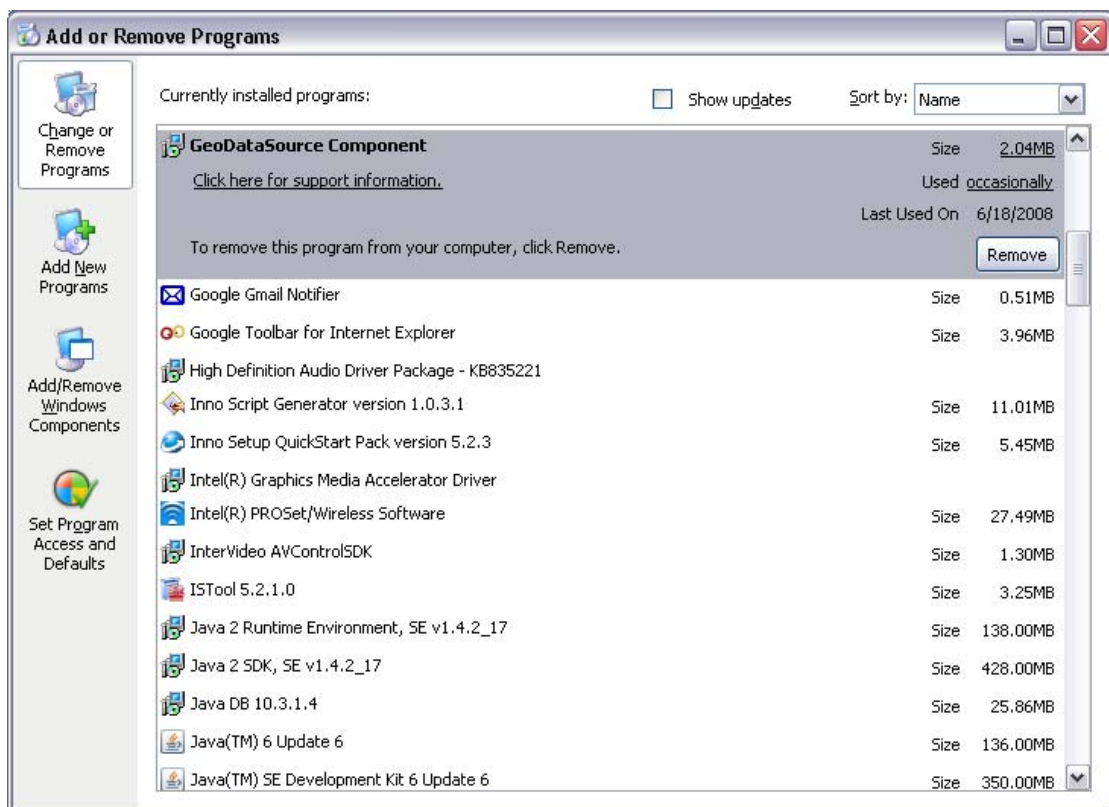
7.1. Upgrade Component

To upgrade a new version of GeoDataSource .NET Component, you need to download and execute the new setup file.

7.2. Uninstall Component

To uninstall GeoDataSource™ .NET Component, you need to remove from add/remove programs

- 7.2.1. Start => Control Panel => Add/Remove Programs =>
Select GeoDataSource Component => Click Remove



- 7.3.2. Click Yes to confirm uninstall, click No to cancel uninstall.

APPENDIX I: ISO3166 COUNTRY CODE

This table lists all valid ISO3166 two characters country codes that returns from IP2Location™ .NET component and explain the full country names for these country codes.

Country Code	Country Name
AD	ANDORRA
AE	UNITED ARAB EMIRATES
AF	AFGHANISTAN
AG	ANTIGUA AND BARBUDA
AI	ANGUILLA
AL	ALBANIA
AM	ARMENIA
AN	NETHERLANDS ANTILLES
AO	ANGOLA
AP	ASIA PACIFIC
AQ	ANTARCTICA
AR	ARGENTINA
AS	AMERICAN SAMOA
AT	AUSTRIA
AU	AUSTRALIA
AW	ARUBA
AZ	AZERBAIJAN
BA	BOSNIA AND HERZEGOWINA
BB	BARBADOS
BD	BANGLADESH
BE	BELGIUM
BF	BURKINA FASO
BG	BULGARIA
BH	BAHRAIN
BI	BURUNDI
BJ	BENIN
BM	BERMUDA
BN	BRUNEI DARUSSALAM
BO	BOLIVIA
BR	BRAZIL
BS	BAHAMAS
BT	BHUTAN
BV	BOUVET ISLAND
BW	BOTSWANA
BY	BELARUS
BZ	BELIZE
CA	CANADA
CC	COCOS (KEELING) ISLANDS
CD	CONGO, THE DEMOCRATIC REPUBLIC OF THE
CF	CENTRAL AFRICAN REPUBLIC
CG	CONGO

Country Code	Country Name
CH	SWITZERLAND
CI	COTE D'IVOIRE
CK	COOK ISLANDS
CL	CHILE
CM	CAMEROON
CN	CHINA
CO	COLOMBIA
CR	COSTA RICA
CS	CZECHOSLOVAKIA (FORMER)
CU	CUBA
CV	CAPE VERDE
CX	CHRISTMAS ISLAND
CY	CYPRUS
CZ	CZECH REPUBLIC
DE	GERMANY
DJ	DJIBOUTI
DK	DENMARK
DM	DOMINICA
DO	DOMINICAN REPUBLIC
DZ	ALGERIA
EC	ECUADOR
EE	ESTONIA
EG	EGYPT
EH	WESTERN SAHARA
ER	ERITREA
ES	SPAIN
ET	ETHIOPIA
EU	EUROPEAN UNION
FI	FINLAND
FJ	FIJI
FK	FALKLAND ISLANDS (MALVINAS)
FM	MICRONESIA, FEDERATED STATES OF
FO	FAROE ISLANDS
FR	FRANCE
FX	FRANCE, METROPOLITAN
GA	GABON
GB	GREAT BRITAIN
GD	GRENADA
GE	GEORGIA
GF	FRENCH GUIANA
GH	GHANA
GI	GIBRALTAR
GL	GREENLAND
GM	GAMBIA
GN	GUINEA
GP	GUADELOUPE
GQ	EQUATORIAL GUINEA
GR	GREECE

Country Code	Country Name
GS	SOUTH GEORGIA & SOUTH SANDWICH ISLANDS
GT	GUATEMALA
GU	GUAM
GW	GUINEA-BISSAU
GY	GUYANA
HK	HONG KONG
HM	HEARD ISLAND AND MCDONALD ISLANDS
HN	HONDURAS
HR	CROATIA
HT	HAITI
HU	HUNGARY
ID	INDONESIA
IE	IRELAND
IL	ISRAEL
IN	INDIA
IO	BRITISH INDIAN OCEAN TERRITORY
IQ	IRAQ
IR	IRAN, ISLAMIC REPUBLIC OF
IS	ICELAND
IT	ITALY
JM	JAMAICA
JO	JORDAN
JP	JAPAN
KE	KENYA
KG	KYRGYZSTAN
KH	CAMBODIA
KI	KIRIBATI
KM	COMOROS
KN	SAINT KITTS AND NEVIS
KP	KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF
KR	KOREA, REPUBLIC OF
KW	KUWAIT
KY	CAYMAN ISLANDS
KZ	KAZAKSTAN
LA	LAO PEOPLE'S DEMOCRATIC REPUBLIC
LB	LEBANON
LC	SAINT LUCIA
LI	LIECHTENSTEIN
LK	SRI LANKA
LR	LIBERIA
LS	LESOTHO
LT	LITHUANIA
LU	LUXEMBOURG
LV	LATVIA
LY	LIBYAN ARAB JAMAHIRIYA
MA	MOROCCO
MC	MONACO
MD	MOLDOVA, REPUBLIC OF

Country Code	Country Name
RU	RUSSIAN FEDERATION
RW	RWANDA
SA	SAUDI ARABIA
SB	SOLOMON ISLANDS
SC	SEYCHELLES
SD	SUDAN
SE	SWEDEN
SG	SINGAPORE
SH	SAINT HELENA
SI	SLOVENIA
SJ	SVALBARD AND JAN MAYEN
SK	SLOVAKIA
SL	SIERRA LEONE
SM	SAN MARINO
SN	SENEGAL
SO	SOMALIA
SR	SURINAME
ST	SAO TOME AND PRINCIPE
SU	RUSSIAN FEDERATION
SV	EL SALVADOR
SY	SYRIAN ARAB REPUBLIC
SZ	SWAZILAND
TC	TURKS AND CAICOS ISLANDS
TD	CHAD
TF	FRENCH SOUTHERN TERRITORIES
TG	TOGO
TH	THAILAND
TJ	TAJIKISTAN
TK	TOKELAU
TM	TURKMENISTAN
TN	TUNISIA
TO	TONGA
TP	EAST TIMOR
TR	TURKEY
TT	TRINIDAD AND TOBAGO
TV	TUVALU
TW	TAIWAN, PROVINCE OF CHINA
TZ	TANZANIA, UNITED REPUBLIC OF
UA	UKRAINE
UG	UGANDA
UK	UNITED KINGDOM
UM	UNITED STATES MINOR OUTLYING ISLANDS
US	UNITED STATES
UY	URUGUAY
UZ	UZBEKISTAN
VA	HOLY SEE (VATICAN CITY STATE)
VC	SAINT VINCENT AND THE GRENADINES
VE	VENEZUELA

Country Code	Country Name
VG	VIRGIN ISLANDS, BRITISH
VI	VIRGIN ISLANDS, U.S.
VN	VIET NAM
VU	VANUATU
WF	WALLIS AND FUTUNA
WS	SAMOA
YE	YEMEN
YT	MAYOTTE
YU	YUGOSLAVIA
ZA	SOUTH AFRICA
ZM	ZAMBIA
ZW	ZIMBABWE

GEODATASOURCE™ .NET COMPONENT LICENSE AGREEMENT

IMPORTANT-READ CAREFULLY:

This License Agreement is a legal agreement between you (either an individual or a single entity) and Hexasoft Development Sdn. Bhd., owner of GeoDataSource™ trademark, (“Hexasoft” or “we”) for the Hexasoft developed GeoDataSource™ .NET Component (hereafter referred to as the SOFTWARE PRODUCT) accompanying this License Agreement, which includes web service routines and data result(s). By exercising your rights to make and use copies of the SOFTWARE PRODUCT, you agree to be bound by the terms of this License Agreement. If you do not agree to the terms of this License Agreement, you may not use the SOFTWARE PRODUCT.

GRANT OF LICENSE.

This License Agreement grants the following rights: You are granted the right to use the SOFTWARE PRODUCT files on one computer in the Internet or Local Area Network (“LAN”). You may not use the SOFTWARE PRODUCT files on multiple computers without matching number of licenses.

DESCRIPTION OF LIMITATIONS.

You may not reverse engineer except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation. Without prejudice to any other rights, Hexasoft may terminate this License Agreement if you fail to comply with the terms and conditions of this License Agreement. In such event, you must destroy all copies of the SOFTWARE PRODUCT and all of its parts.

COPYRIGHT.

All title and copyrights in and to the SOFTWARE PRODUCT and any copies of the SOFTWARE PRODUCT are owned by Hexasoft. The SOFTWARE PRODUCT is protected by copyright laws and international treaty provisions.

NO WARRANTIES.

Hexasoft expressly disclaims any warranty for the SOFTWARE PRODUCT. The SOFTWARE PRODUCT and any related documentation is provided "as is" without warranty of any kind, either express or implied, including, without limitation, the implied warranties of merchantability or fitness for a particular purpose. The entire risk arising out of use or performance of the SOFTWARE PRODUCT remains with you.

LIMITATION OF LIABILITY.

Hexasoft’s entire liability and your exclusive remedy under this Agreement shall not exceed fifteen dollars (US \$15.00).

NO LIABILITY FOR CONSEQUENTIAL DAMAGES.

In no event shall Hexasoft nor anyone else who has been involved in the creation, production, or delivery of the SOFTWARE PRODUCT be liable for any damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this Hexasoft product, even if Hexasoft has been advised of the possibility of such damages. Because some states and jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

MISCELLANEOUS

“GeoDataSource” is a trademark of Hexasoft Development Sdn. Bhd..

“Microsoft” is a registered trademark of Microsoft Corporation.

”Windows” is a registered trademark of Microsoft Corporation.

All other trademarks are the property of their respective owners.

Copyright © 2008 Hexasoft Development Sdn. Bhd., All Rights Reserved.

Hexasoft Development Sdn. Bhd.

1-2-15 Mayang Mall Kompleks,

Jalan Mayang Pasir 1,

11950 Bayan Baru,

Pulau Pinang,

Malaysia. Tel: (6)-04-640-2380

Fax: (6)-04-640-2381

Email: sales@geodatasource.com