

## Overview

**AddressLocate** is an easy-to-use and low-cost method of finding the approximate geographical position of a UK postcode.

It is accurate to an average value of +/- 2.7 kilometres (1.7 miles). For many applications, such as a "where's my nearest" search; delivery distance estimation or local depot notification, this is adequate.

Users who need more accuracy may wish to consider Arc en Ciel's **Address** and **Address+** programs which have an optional grid reference add-on that gives a position for each postcode that is accurate to about +/- 50 metres. Users that need pin-point accuracy for individual premises are referred to the **AddressPoint** product marketed by the Ordnance Survey. ( [www.ordnancesurvey.co.uk](http://www.ordnancesurvey.co.uk) )

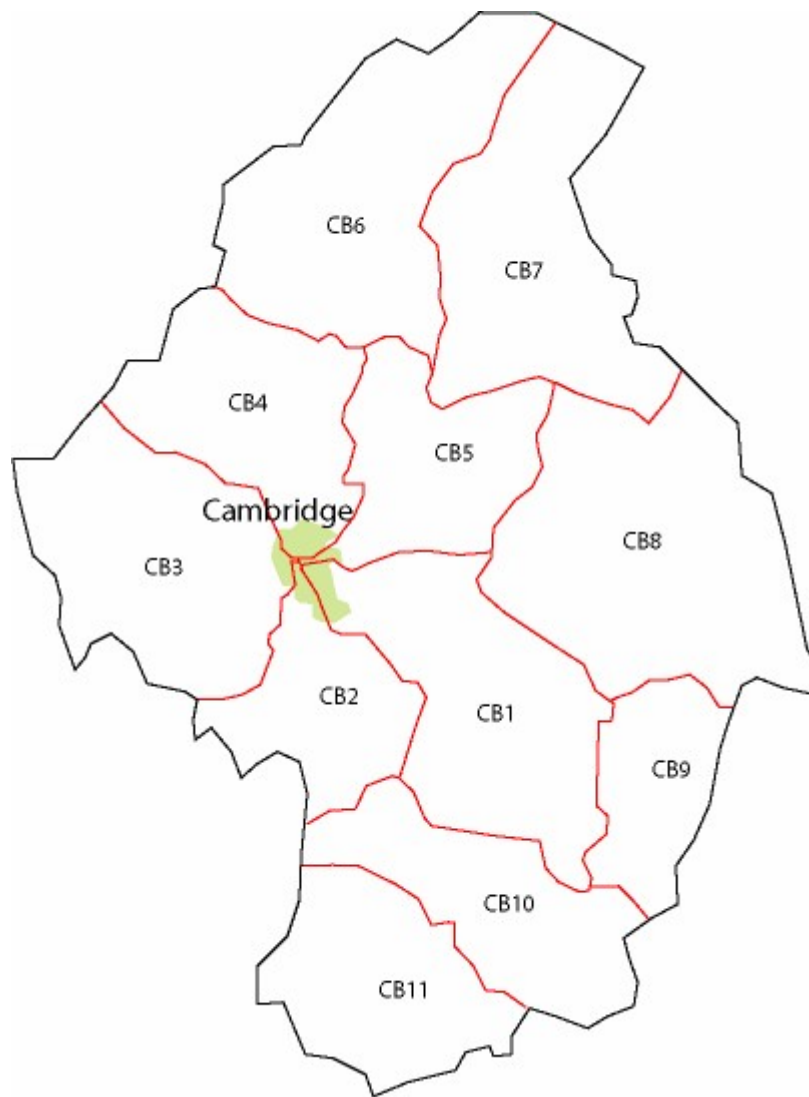
AddressLocate cannot run as a stand-alone application, but must be called from a host program. It is currently available as a COM object that can be conveniently used from Visual Basic and many other applications; a DLL file that can be called from C++ or Delphi applications and as a Java library. A .NET version is under active development.

AddressLocate is licensed for a single year, after which it must be renewed .

## Concepts

The Royal Mail has divided the UK into 124 *Areas* for the purpose of delivering mail. Each Area is named with a one- or two-letter code which is a mnemonic for the major town. For example, the Area around Cambridge is coded as "CB". For a full list of Areas, click [here](#).

In turn, each Area is divided into *Districts*. Each District has a one- or two-digit number; for example, the Cambridge Area has 11 Districts, coded as CB1 to CB11, as shown below. In central London this may alternatively be a digit and a letter; for example the W1D District encompasses most of Oxford Street. There are currently 2796 Districts.



An individual postcode is made up of the Area and District codes, then a space then a digit followed by two letters, as in for example CB1 3AA. There are about 2m individual postcodes.

AddressLocate incorporates data for each Area polygon and each District polygon. The data stored is the area in square kilometres and the (x, y) coordinate of the centroid of the polygon.

The centroid or centre of gravity of a shape is its centre of mass. One way of visualising this is that if you take a notional straight edge and place the polygon on it so that its centroid is over the straight edge, the polygon will always balance, no matter how you rotate the polygon. The figure below shows District CB1 with its centroid marked.



The area is supplied as an integer to the nearest square kilometre. therefore the area of a District which is less than 0.5 km<sup>2</sup> such as WC1A, will be given as zero.

The centroid is returned as the Easting and Northing of an Ordnance Survey grid reference. This is effectively an (x, y) co-ordinate in metres from an origin off the Scilly Isles. For consistency, areas such as Northern Ireland and the Channel Islands are also expressed as points on the Great Britain grid. The [GeoRefConvert](#) function can be used to change the datum as required.

The largest District is IV27 in the north of Scotland, which has an area of 3478 km<sup>2</sup>. However, the average size is 30 km<sup>2</sup>. This implies that taking a District centroid as an approximation for a particular postcode gives an average error of +/- 2.7 kilometres (1.7 miles).

## Dynamic Link Library (DLL) deployment

Programmers using languages such as C++ or Delphi will normally find it most convenient to deploy **AddressLocate** as a Dynamic Link Library, usually abbreviated to "DLL".

On installation, the DLL file, named AddressLocate.dll, was placed in your System directory, which will normally be C:\Windows\System32.

The AddressLocate.dll file must be added to your project.

You must then add the following declarations to your Header file: -

```

BOOL WINAPI OutcodeData(char *szCode, int *nArea, int *nEasting, int *nNorthing);
BOOL WINAPI GeoRefConvert(int projIn, double xIn, double yIn, int projOut,
                           double *xOut, double *yOut);
BOOL WINAPI CodesInRadius(int x, int y, int Radius, char *szCodes);

```

You are then able to make calls to the functions from within your code, such as: -

```

int nArea, xCoord, yCoord;
BOOL bOK = OutcodeData("CB1", &nArea, &xCoord, &yCoord);

```

This will return the area of District CB1 in square kilometres, and its centroid as an x,y coordinate pair in metres.

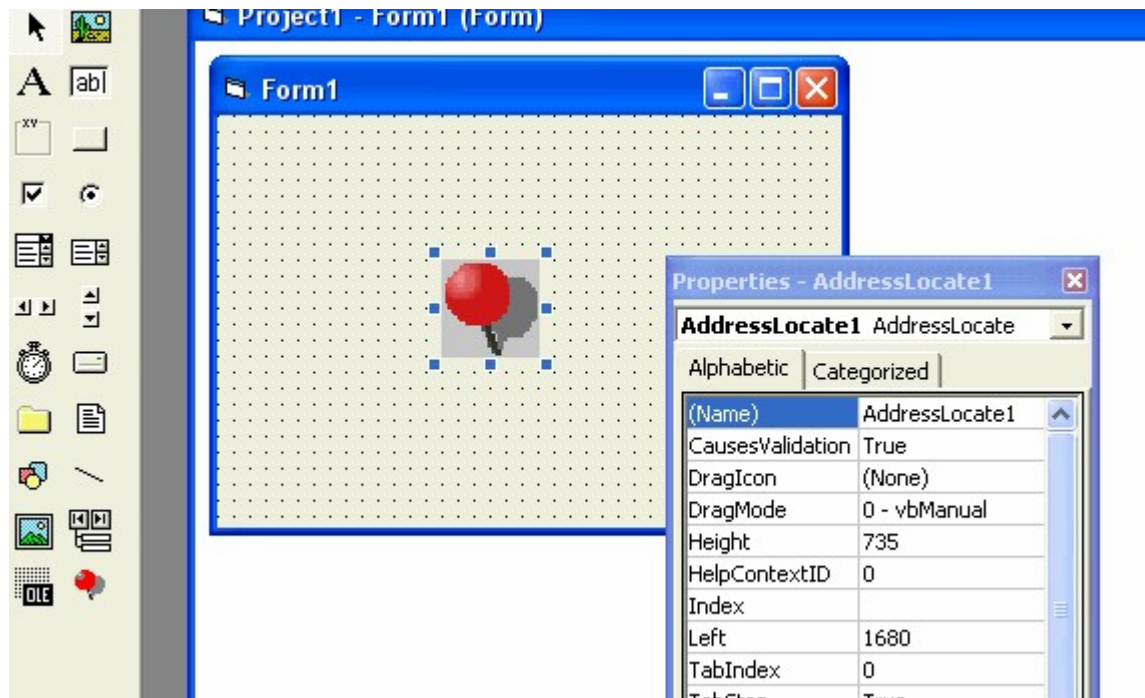
## COM object deployment

**AddressLocate** is available as a COM object that is registered with your system on installation.

In Visual Basic, select Components from the Project menu. This will show a list of all the currently registered COM objects, amongst which should be "Arc en Ciel Positioning System". Place a tick

against this and dismiss the dialog. You will then see a map-pin icon has been added to your toolbox.

You can now click and drag the icon onto your form, when you should see something like the following.



Note that the object added to your project has been called "AddressLocate1" by default.

Now swap to the Code view and you can place a line of code such as: -

```
Status = AddressLocate1.OutcodeData("CB1", Area, xCoord, yCoord)
```

On return, *Status* will be zero (False) on failure, or non-zero on success. *Area* will hold the area of the CB1 District in square kilometres and (*xCoord*, *yCoord*) the grid reference in metres of the centroid.

In systems other than Visual Basic it may not be possible to place objects graphically.

In this case the code below will create the object and call it: -

```
Set objLocate = CreateObject("AddressLocateX.AddressLocate")
Status = objLocate.OutcodeData("CB1", Area, xCoord, yCoord )
```

## Java deployment

**AddressLocate** is available as a set of Java classes. By default these are placed in C: \Program Files\AddressLocate\Java on installation.

The top-level class is called **AddressLocate** and has the public methods **OutcodeData**, **GeoRefConvert** and **CodesInRadius**.

The following code snippet will set up an instance of the class and call the method to get information for a District code.

```
// Set up the class
```

```

AddressLocate objLocate = new AddressLocate();
// Get data for the CB1 District
int Data[] = { 0, 0, 0};
boolean bOK = objLocate.OutcodeData("CB1", Data);

// Display the grid reference
if(bOK) System.err.println("Centroid is at " + Data[1] + "," + Data[2]) ;

```

## Functions

**AddressLocate** currently has only three functions (or methods as they are often called).

These are **OutcodeData**, **GeoRefConvert** and **CodesInRadius**.

OutcodeData finds information for an Area or a District.

GeoRefConvert converts a map reference into another coordinate system. For example, an Ordnance Survey grid reference can be converted to a Latitude and Longitude pair or vice versa.

CodesInRadius finds the District codes within a given radius of a point.

Calling details are as follows: -

### OutcodeData

COM Usage

```
bool objLocate.OutcodeData(VARIANT sCode, VARIANT nArea, VARIANT nEast, VARIANT nNorth)
```

DLL Usage

```
BOOL OutcodeData(char *szCode, int *nArea, int *nEast, int *nNorth);
```

Java Usage

```
boolean OutcodeData(String sCode, int Data[]);
```

Return value

True (non-zero) on success. False (zero) Licence has expired, or *sCode* is not recognised.

Parameters

*sCode/szCode* A string (zero-terminated in DLL usage) giving an Area or District code

*nArea* The area of the Area or District in square kilometres

*(nEast, nNorth)* The grid reference of the centroid of the Area or District in metres.

*Data* A 3-element integer array that is loaded with the area and grid reference.

Remarks

The code must be a valid Area code (e.g. "CB") or a valid District code (such as "CB1") The code is not case-sensitive. A full postcode may also be supplied, in which case its District code will be used.

The area value is rounded to the nearest square kilometre, so the area of a very small District such as WC1A may be given as zero.

### GeoRefConvert

#### COM Usage

```
bool objLocate.GeoRefConvert(VARIANT projIn, VARIANT xIn, VARIANT yIn,  
                             VARIANT projOut, VARIANT xOut, VARIANT yOut)
```

#### DLL Usage

```
BOOL GeoRefConvert(int projIn, double xIn, double yIn, double *xOut, double *yOut);
```

#### Java Usage

```
boolean GeoRefConvert(int projIn, double xIn, double yIn, int projOut, double Output[]);
```

#### Return value

True (non-zero) on success. False (zero) invalid coordinate system or invalid input geo-reference.

#### Parameters

*projIn/projOut* The type of input and output geo-reference. The following values are valid: -

- 1 - Ordnance Survey Easting/Northing grid reference.
- 2 - Grid reference on the Irish grid.
- 3 - Longitude/Latitude using the OSGB36 system (Great Britain).
- 4 - Longitude/Latitude using the Ireland 65 system (Ireland).
- 5 - Longitude/Latitude using the WSG84 system (world).

(*xIn*, *yIn*) The elements of the geo-reference in the form given by *projIn*.

For *projIn*=1 they are Easting and Northing and for other values of *projIn*, Longitude and Latitude.

(*xOut*,*yOut*) The transformed values of (*xIn*, *yIn*)

*Output* A 2-element array that is loaded with *xOut* and *yOut*.

#### Remarks

Longitude and Latitude can vary by up to 200m depending on the system used. When working with GPS recorders note that these devices usually use the World Geodetic System (WGS84)

### **CodesInRadius**

#### COM Usage

```
int objLocate.CodesInRadius(VARIANT x, VARIANT y, VARIANT Radius, VARIANT sCodes)
```

#### DLL Usage

```
int CodesInRadius(int x, int y, int Radius, char *szCodes);
```

#### Java Usage

```
String CodesInRadius(int x, int y, int Radius);
```

#### Return value

With COM and DLL, return -1 on internal error, otherwise the number of District codes found.

With Java return the *sCodes* string.

## Parameters

*(x, y)* The coordinates of a point, in metres.

*Radius* A radius about the point, in metres.

*sCodes/szCodes* A string containing the codes. Each code occupies 4 characters, blank padded if necessary.

## Remarks

Only District codes, such as "CB1" will be returned, not Area codes such as "CB" .

## Contact Details

You can contact us as follows: -

**Address:** **Arc en Ciel Ltd**  
13 Gresley Road  
LONDON  
N19 3LA

**Telephone:** **(020) 7281 9885**

**Fax:** **(020) 7281 9824**

**Email:** **info@arcenciel.com**



Also see our website at [www.arcenciel.com](http://www.arcenciel.com) for the latest information