



Using ClearImage in Delphi

Using ClearImage COM Server

These examples discuss barcode recognition. However, similar considerations apply to ClearImage [Repair](#) and [Tools](#) which are engines that modify the images for various purposes.

To import ClearImage into Delphi project:

1. In **Project** menu select **Import Type Library**
2. Select **ClearImage COM Server** in list box
3. Click on the **Create Unit** button

To use ClearImage in Delphi project

1. Add the ClearImage_TLB unit to the **uses** clause
2. Initialize the COM interface:
 - in GUI: Call `Application.Initialize;`
 - in DOS:
 - Add to `uses` `ActiveX, Windows`
 - Call `CoInitialize(nil);` before any ClearImage calls
 - Call `CoUnInitialize;` before exiting application
3. Implement ClearImage processing in a procedure or a function
 - A. Create the ClearImage server. This can be done once in your project:

```
Ci: ICiServer;  
Ci := CoCiServer.Create;
```
 - B. Create the barcode reading object. The example below is for reading 1D barcodes and uses ClearImage 1D Pro. Use `ICiPdf417` to read PDF417 barcodes or `ICiDataMatrix` to read DataMatrix barcodes.

```
Barcode: ICiBarcodePro;  
Barcode := Ci.CreateBarcodePro;
```
 - C. Load the image from a file (or other source):

```
Barcode.Image.Open(aFileName, 1);
```
 - D. Read the barcode values. Process the results in `Barcode.Barcodes`:

```
Barcode.Find(0);
....
```

4. Exception handling

Any error detected by ClearImage COM (e.g. Image.Open tries to open file that does not exist) generates a COM exception. Use **try - except** to trap and handle such errors.

5. If the image will be shared among multiple processing modules, the image object should be created separately and then attached to the processing object:

```
Image: ICiImage;
Image := Ci.CreateImage;
Image.Open(aFileName, 1);
Barcode.Image := Image;
```

Code Example

The following example demonstrates a common use case for ClearImage. It reads Code 39 barcodes from an image file.

- You may 'OR' the type values in the `Barcode.Type` property to detect multiple barcode symbologies. Alternatively, set the `AutoDetect1D` property to `ciTrue` to detect any barcodes of the most popular symbologies (recommended).
- Also see how you can search for either 1 or many barcodes with one call

```
uses
  ...,ClearImage_TLB, comobj;

.....
function DecodeBarcode(const aFileName: string): AnsiString;
var
  Ci: ICiServer;
  Image: ICiImage;
  Barcode: ICiBarcodePro;
begin
  Result := '';
  try
    try
      begin
        Ci := CoCiServer.Create;
        // Open image
        Image := Ci.CreateImage;
        Image.Open(aFileName, 1);
        // Create Barcode recognition object
        Barcode := Ci.CreateBarcodePro;
        Barcode.Image := Image;
        Barcode.Type_ := cibCode39;
        // To automatically detect barcode type
        // Barcode.AutoDetect1D := ciTrue;
        // Read one barcode. To find all barcodes call Barcode.Find(0)
        Barcode.Find(1);
```

```

        if Barcode.Barcodes.Count > 0 then
            Result := Barcode.Barcodes.Item[1].Text;
        end;
    except
        // Process errors
        on E: Exception do ShowMessage (Format('Error:%s.File:%s',
                                                [E.Message,FileName]));
    else
        ShowMessage ('Unknown Error');
    end;
finally
    end;
end;
end;

```

Using ClearMICR Reader

This object reads MICR lines from check images and can also extract the check image (from a larger image which may have other documents), crop and deskew it for presentation.

To import ClearMICR into Delphi project:

1. In **Project** menu select **Import Type Library**
2. Select **ClearCheck MICR Reader** in list box
3. Click on the **Create Unit** button

To use ClearMICR in Delphi project

1. Add the ClearMICR_TLB unit to the **uses** clause
2. Initialize COM interface:
 - in GUI: Call Application.Initialize;
 - in DOS:
 - Add to **uses ActiveX, Windows**
 - Call **CoInitialize(nil)**; before any ClearMICR calls
 - Call **CoUnInitialize**; before exiting application
3. Implement ClearMICR processing in a procedure or a function. Use the Code Sample below.
4. Exception handling

Any error detected by ClearMICR generates a COM exception. Use **try - except** to trap and process such errors.

Code Example

The following example demonstrates the use of ClearMICR.

```
uses
    ...,ClearMICR_TLB, comobj;

.....
var
    reader: TccMicrReader;

procedure TfmMain.FormCreate(Sender: TObject); Begin
    inherited;
    reader := TccMicrReader.Create(self); end;

procedure TfmMain.FormDestroy(Sender: TObject); Begin
    inherited;
    reader.Free;
end;

procedure FindMicr(Const FileName: String,
    var Route,Checksum,OnUs,AuxOnUs,EPC,Amount,Account: String);
var
    Cnt: Integer;
    Micr: ccMicr;
Begin
    Route := '';
    CheckSum := '';
    OnUs := '';
    AuxOnUs := '';
    EPC := '';
    Amount := '';
    Account := '';
    reader.Image.Open(FileName,1);
    Cnt := reader.FindMICR;
    If Cnt > 0 Then
        Begin
            Micr := reader.MicrLine[1];
            If Micr.Routing.IsRead <> 0 Then
                Route := Micr.Routing.TextANSI;
            If Micr.RoutingChecksum.IsRead <> 0 Then
                CheckSum := Micr.RoutingChecksum.TextANSI;
            If Micr.OnUs.IsRead <> 0 Then
                OnUs := Micr.OnUs.TextANSI;
            If Micr.AuxOnUs.IsRead <> 0 Then
                AuxOnUs := Micr.AuxOnUs.TextANSI;
            If Micr.EPC.IsRead <> 0 Then
                EPC := Micr.EPC.TextANSI;
            If Micr.Amount.IsRead <> 0 Then
                Amount := Micr.Amount.TextANSI;
            If Micr.Account.IsRead <> 0 Then
                Account := Micr.Account.TextANSI;
        end;
    End;
```