Object Pascal Wrapper for Shapefile C Library

Description

Object Pascal Wrapper for Shapefile C Library (OPW4SHPLIB) offers an elegant way to work with the Enviromental Systems Research Institute (ESRI)'s ShapeFiles using the Frank Warmerdam's well-known library released under the MIT license.

The project page is: http://coderesearchlabs.com/opw4shplib

To learn more about how to use it, check the Usage section of this documentation.

NOTES:
This library was created in Windows but nothing prevents you to compile it under other FPC or Delphi supported platforms with small adjustments. Btw, remember to adjust the ShapeLib const in ShpWrapper unit to fit your needs.

Just in case you are new to the GIS programming world, I recommend you to take a look at Open Source Geospatial Foundation (OSGeo) to know about other open source projects like this and obtain more detailed information about geographic information systems.

Enjoy,
Javier Santo Domingo
j-a-s-d@coderesearchlabs.com

Usage

The intended and most elegant way to use **OPW4SHPLIB** is by calling the provided automated factories to load a Shapefile or a whole directory of Shapefiles.

Some simple examples showing an automated file info report:

```
procedure ShowShapeFileInfo(const AFile: string);
begin
   with ShapeFile(AFile, [loCaptureDetailedFileInfo,
   loDBFAttributesGetNativeFieldType]) do
      if FileLoadResult = lrOk then
         WriteLn(FileInfo);
end;

procedure ShowDirectoryShapeFilesInfo(const ADirectory: string);
var shpFile: TShapeFile;
begin
   with ShapeFilesDirectory(ADirectory, [loCaptureDetailedFileInfo,
   loDBFAttributesGetNativeFieldType]) do
      if not HasErrorsLoading then
         for shpFile in ShapeFiles do
            WriteLn(shpFile.FileInfo);
end;
```

Alternatively, for the purist, you can use it by instantiating the classes manually in a traditional approach when we talk about an "Object Pascal Wrapper".

```
procedure ShowShapeFileInfo(const AFile: string);
var shpFile: TShapeFile;
begin
   shpFile := TShapeFile.Create;
   try
      if shpFile.LoadFromFile(AFile, [loCaptureDetailedFileInfo,
      loDBFAttributesGetNativeFieldType]) then
         WriteLn(shpFile.FileInfo);
   finally
      shpFile.Free;
   end;
end;
```

For more detailed examples, you can check the demos included (for example, ShpDump and ShpDumpDir), all tested under FPC 2.4.2 and Turbo Delphi 2006.

In any of the both cases explained, you must import the **ShpFiles** unit in your uses clause.

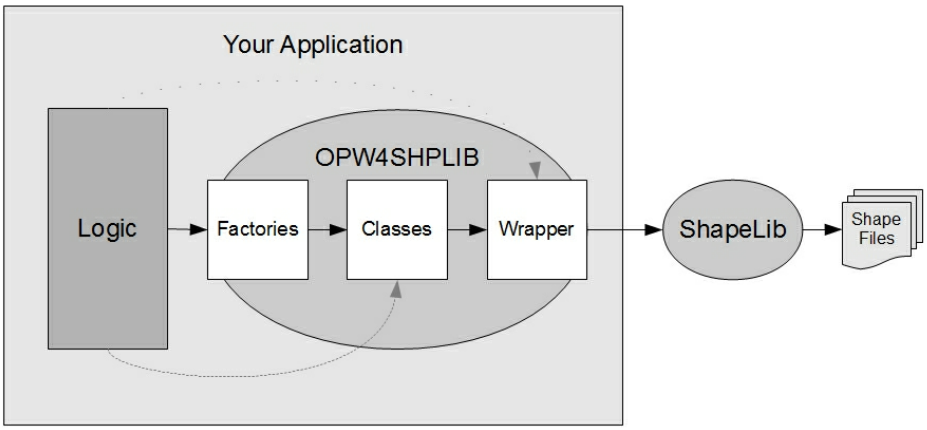
And finally, if for any reason you want a more raw way (for example if you just want to upgrade to the most updated version you can find of the ShapeLib bindings) you can directly access the wrapper. Just import the unit **ShpWrapper** in your uses clause.

Usage

To get the full picture of all this, see the following diagram:

Object Pascal Wrapper for Shapefile C Library

Usage Diagram



http://coderesearchlabs.com/opw4shplib

The Factories

The factories, contained in the **ShpFiles** unit, allow you to interact with Shapefiles more easily by providing interfaces to work with them.

- **File factory**

To load a Shapefile just call the following automated factory:

```
function ShapeFile(const AFilename: AnsiString; ALoadOptions:
TShapeFileLoadOptions = []): IShapeFile;
```

that returns the following interface:

```
IShapeFile = interface
   function GetFileLoadResult: TShapeFileLoadResult;
   function GetFileInfo: AnsiString;
   function GetFileName: AnsiString;
   function GetFileBounds: TShapeBounds;
   function GetShapeTypeName: AnsiString;
   function GetShapeType: LongInt;
   function GetEntities: LongInt;
   function GetShapeObjectList: TShapeObjectList;
   function LoadFromFile(const AFileName: AnsiString; ALoadOptions:
   TShapeFileLoadOptions = []): boolean;
   procedure AddToFileInfo(const AText: AnsiString);
   property ShapeObjects: TShapeObjectList read GetShapeObjectList;
   property ShapeObjectsCount: LongInt read GetEntities;
   property ShapeType: LongInt read GetShapeType;
   property ShapeTypeName: AnsiString read GetShapeTypeName;
   property FileLoadResult: TShapeFileLoadResult read GetFileLoadResult;
   property FileInfo: AnsiString read GetFileInfo;
   property FileName: AnsiString read GetFileName;
   property FileBounds: TShapeBounds read GetFileBounds;
end;
```

- **Directory factory**

And if you want to load a whole directory:

```
function ShapeFilesDirectory(const APath: AnsiString = '.'; ALoadOptions:
TShapeFileLoadOptions = []): IShapeFilesDirectory;
```

that returns the following interface:

```
IShapeFilesDirectory = interface
   function GetShapeFiles: TShapeFiles;
   procedure LoadShapeFiles(const APath: AnsiString = '.'; AShapeFileLoadOptions:
   TShapeFileLoadOptions = []);
   procedure RetrieveLoadingErrorsReport(var AStringList: TStringList);
   function HasErrorsLoading: boolean;
   procedure GetShapeFileNames(var AStringList: TStringList);
   function GetShapeFileByFileName(const AFileName: AnsiString): TShapeFile;
   property ShapeFiles: TShapeFiles read GetShapeFiles;
end;
```

The Classes

The classes, contained in the **ShpFiles** unit, follow the ESRI's Shapefiles Technical Description, mostly mapping the whole structure in a direct way.

Additionally, **OPW4SHPLIB** incorporates special facilities to those classes to improve dealing with them, such as: a collection based filiation of parts and vertices, properties and methods indicating is the end part or the last vertex, etc.

- **dBASE File related classes**
  - TDBFAttribute
  - TDBFAttributes
- **Main File related classes**
  - TShapeBounds
  - TShapeVertex
  - TShapeVertices
  - TShapePart
  - TShapeParts
  - TShapeObject
  - TShapeObjectList
- **Shapefile & Shapefile Directory related classes**
  - TShapeFile
  - TShapeFiles
  - TShapeFilesDirectory

The Wrapper

The **ShpWrapper** unit contains the direct wrapper to the Shapefile C Library (v. 1.3.0b2).

It is based on the previous works of Keven Meyer and Alexander Weidauer, which are dated back from 2002 and 2003 respectively, and both are released under the licenses MIT and LGPL. After the mix up, the FreePascal support was added, the source code was formatted and the new routines of ShapeLib were wrapped, amongst other enhacements.

- **dBASE File related routine prototypes**

  - DBFOpen
  - DBFCreate
  - DBFCreateEx
  - DBFGetFieldCount
  - DBFGetRecordCount
  - DBFAddField
  - DBFGetFieldInfo
  - DBFGetFieldIndex
  - DBFReadIntegerAttribute
  - DBFReadDoubleAttribute
  - DBFReadStringAttribute
  - DBFIsAttributeNULL
  - DBFWriteIntegerAttribute
  - DBFWriteDoubleAttribute
  - DBFWriteStringAttribute
  - DBFWriteNULLAttribute
  - DBFReadTuple
  - DBFWriteTuple
  - DBFCloneEmpty
  - DBFClose
  - DBFGetNativeFieldType
  - DBFWriteLogicalAttribute
  - DBFReadLogicalAttribute
  - DBFUpdateHeader
  - DBFIsRecordDeleted
  - DBFMarkRecordDeleted
  - DBFGetCodePage
  - DBFWriteAttributeDirectly

- **Main File related routine prototypes**

  - SHPOpen
  - SHPCreate
  - SHPGetInfo
  - SHPReadObject
  - SHPWriteObject
  - SHPDestroyObject
  - SHPComputeExtents
  - SHPCreateObject
  - SHPCreateSimpleObject
  - SHPClose
  - SHPTypeName
  - SHPPartTypeName
  - SHPWriteHeader
  - SHPRewindObject
  - SHPCreateTree
  - SHPDestroyTree
  - SHPWriteTree
  - SHPReadTree
  - SHPTreeAddObject
  - SHPTreeAddShapeId
  - SHPTreeRemoveShapeId
  - SHPTreeTrimExtraNodes
  - SHPTreeFindLikelyShapes
  - SHPCheckBoundsOverlap

License

Tools

In the development of **Object Pascal Wrapper for Shapefile C Library** the following tools were specifically involved:

for source code compiling
Florian Kaempfl's FreePascal 2.4.2
http://freepascal.org

for source code compiling
CodeGear's Turbo Delphi 2006
http://www.turboexplorer.com/delphi

for source code editing
Tabitha
http://coderesearchlabs.com/tabitha

When working at **Code Research Laboratories** the following tools are used:

in the test management field
Gurock Software's TestRail
http://www.gurock.com/testrail/

in the version control field
VisualSVN Ltd's VisualSVN
http://www.visualsvn.com/

in the similarity analysing field
RedHill Consulting's Simian
http://www.redhillconsulting.com.au/products/simian/

in the application lifecycle management field
Inedo's BuildMaster
http://www.inedo.com/buildmaster/

in the documentation field
EC Software's Help & Manual
http://www.ec-software.com/products_hm_overview.html

History

DATE	DESCRIPTION
2006.06.28	initial wrapper around ShapeLib 1.2.9
2006.07.10	improvements to the wrapper objects
2011.04.27	updated to ShapeLib 1.3.0b2
2011.04.28	addded TShapeFileLoadOptions
2011.04.30	added IShapeFile and IShapeFilesDirectory interfaces and their factories
2011.05.20	added Documentation