NOTE

This information is specific to DotImage at the time it was written and may change slightly in future versions.

Every OcrEngine has different requirements in terms of how it is deployed. Atalasoft has tried to formalize this process as much as possible as well as to provide guidelines on the mechanism for deployment. Licensing is covered in another topic. This topic covers how to ensure that an OcrEngine will be able to start and will be able to find its own resources.

In your SDK installation, you will find a folder named "OcrResources". This folder is the general folder for all supported OCR engines. Within it you will see a structure like this:

```
ngineManufacturer1 EngineManufacturer2 ... EngineManufacturerN ...
```

In general, most of the handling of loading and locating resources is managed by Atalasoft or by the engine itself and does not require work by the client, but in custom situations, there may be work to be done by the client to handle this.

To sort this out, let's start with a few definitions:

Definitions

- engine resources folder - this is the folder which contains the OCR Engine's resource files
- OCR resources folder - the top level folder of all OCR Engine resources, called "OcrResources"
- application folder - the folder where your application is installed
- assembly folder - the folder which contains the DotImage assembly files (ie., Atalasoft.dotImage.Ocr.dll), this may be the same as the application folder
- SDK folder - the folder which contains all the DotImage assembly files as installed as part of the DotImage SDK. This folder is typically C:\Program Files (x86)\Atalasoft\DotImage \bin
- engine module - an engine supplied dll that provides engine functionality

In order to run, some engines have two requirements: an engine module for some portion of OCR functionality and resource files that are used to configure the engine or otherwise

provide necessary data or services. This may include such things as dictionaries, grammar rules, glyph shapes, neural networks and so on.

Engines that require engine modules typically need to have those modules loaded before attempting to construct a class that requires them. This presents an interesting issue in that the assembly that uses the engine module should contain the knowledge of how to find the engine module, but the engine module needs to be loaded before the module that should be able to find it is loaded.

Atalasoft tries to handle this for you when possible so you don't have to worry about it, but there are some cases where this simply isn't possible.

Options for the Developer

The developer can choose to leave the engine module in the OCR resources folder as shipped. If this is the case, then the developer must put the OCR resources folder within the assembly folder. Alternately, the developer can put the OcrResources folder in any location, but it is the developer's responsibility to load the DLL.

The developer can choose to move the engine module out of the OCR resources folder. In this case, if the engine module is put into the application folder or the assembly folder, then it should be located automatically. If the engine module is located somewhere else it is the developer's responsibility to locate it and load it. If the OCR resources folder is within the assembly folder, the developer can pass in null to the engine constructor for the path, otherwise the developer must pass the location in.

OmniPage Engine (DotImage 11.3 and newer)

OmniPageEngine is new as of 11.3. OmniPage resources must be downloaded separately, we Don't ship them with the SDK be cause the resource zip is quite large. In order to deploy an app with OmniPageEngine, you must also deploy our OmniPage Resources

1. Download the OmniPage Resources zip file Atalasoft.OmniPage.Resources.zip
2. Create a folder where you will store the OmniPage Resources

   NOTE: the Atalasoft.OmniPage.Resources.zip does not create a container directory when you unzip, so make sure the folder you want to use exists, such as:

C:\Program FIles (x86)Atalasoft\DotImage 11.3\bin\OcrResources\OmniPage

3. Copy the Atalasoft.OmniPage.Resources.zip folder to that directory and unzip
4. Ensure that in your application code, your OmniPageLoader is pointing at the folder where you unzipped the OmniPage Resources

   EX:

```
string ocrResourcePath = @"C:\Program Files (x86)\Atalasoft\DotImage
11.3\bin\OcrResources\OmniPage"; OmniPageLoader loader = new
OmniPageLoader(ocrResourcePath);
```

Please see INFO: OmniPageEngine Overview for more details on OmniPageEngine

AbbyyEngine (DotImage 10.7 through 11.2)

AbbyyEngine OCR resources must be downloaded separately. We don't ship them with the SDK because the resources are rather large (about 3/4 of a GiB). In order to deploy an app with AbbyyEngine, you must also deploy our AbbyyResources

1. Download the AbbyyResources zip file Atalasoft.ABBYY.Resources.zip
2. Create a folder where you will store the Abbyy Resources

   NOTE: the Atalasoft.ABBYY.Resources.zip does not create a container directory when you unzip, so make the folder you want to use such as:

   C:\Program Files (x86)\Atalasoft\DotImage 11.1\bin\OcrResources\ABBYY
3. Copy the Atalasoft.ABBYY.Resources.zip folder into that directory and unzip
4. Ensure that in your application code, your AbbyyLoader is pointing at the folder where you have unzipped the Abbyy Resources

   EX:

```
tring ocrResourcePath= @"C:\Program Files (x86)\Atalasoft\DotImage
11.0\bin\OcrResources\ABBYY"; AbbyyLoader loader = new AbbyyLoader(ocrResourcePath);
```

   Please see INFO: AbbyyEngine - Overview for more details on the AbbyyEngine

GlyphReaderEngine (v5.0 for DotImage 11.2 and newer)

NOTE: this section is referring to GlyphReader v5.0 which is found in version 11.2 and newer. Please see the section on GlyphReader v4.0 for 10.3.1 through 11.1 and the section

HOWTO: Deploy a project using OCR

on GlyphReader v3.0 for older than 10.3.1

The 5.0 GlyphReader engine requires an OcrResources folder with a GlyphReader sub-directory which itself has a v5.0 sub-directory containing the following 13 files and two folders:

- TOCR50.qnp
- TOCR50.teh
- TOCR50de.gar
- TOCR50el.gar
- TOCR50en.gar
- TOCR50es.gar
- TOCR50fr.gar
- TOCR50it.gar
- TOCR50nl.gar
- TOCR50no.gar
- TOCR50ru.gar
- TOCR50sk.gar
- TOCR50tr.gar
- x86\
- x64\

the x86 and x64 folders each contain the following files (of the correct "bitness")

- GlyphReader.dll
- GlyphReader.ini
- GlyphReaderEngine.exe

You will need to create a folder called OcrResources in your bin directory of your application, and copy the GlyphReader folder from

```
:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\
```

into it (where x is the version of DotImage such as 11.2 etc...)

You will also need to have a reference in your project to

Atalasoft.dotImage.Ocr.GlyphReader.dll, but set the CopyLocal property to false, you then will need to physically place a copy of the correct version (x86/x64 and 3.5/4.5.2 for bitness and .NET framework respectively) of Atalasoft.dotImage.Ocr.GlyphReader.dll in the OCRResources folder

So, your application's bin folder will have

- bin\OCRResources\Atalasoft.dotImage.Ocr.GlyphReader.dll
- bin\OCRResources\GlyphReader\ (folder)
- bin\OCRResources\GlyphReader\v5.0\ (folder)
- bin\OCRResources\GlyphReader\v5.0\TOCR50.qnp
- bin\OCRResources\GlyphReader\v5.0\TOCR50.teh
- bin\OCRResources\GlyphReader\v5.0\TOCR50de.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50el.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50en.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50es.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50fr.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50it.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50nl.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50no.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50ru.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50sk.gar
- bin\OCRResources\GlyphReader\v5.0\TOCR50tr.gar
- bin\OCRResources\GlyphReader\v5.0\x64\ (folder)
- bin\OCRResources\GlyphReader\v5.0\x64\GlyphReader.dll
- bin\OCRResources\GlyphReader\v5.0\x64\GlyphReader.ini
- bin\OCRResources\GlyphReader\v5.0\x64\GlyphReaderEngine.exe
- bin\OCRResources\GlyphReader\v5.0\x86\ (folder)
- bin\OCRResources\GlyphReader\v5.0\x86\GlyphReader.dll
- bin\OCRResources\GlyphReader\v5.0\x86\GlyphReader.ini
- bin\OCRResources\GlyphReader\v5.0\x86\GlyphReaderEngine.exe

Due to the architecture of the GlyphReader engine, to specify a location other than a default search path such as System32, you'll need to create an instance of the OcrResourceLoader or GlyphReaderLoader in a static constructor before any OCR code is loaded. This is the case even if the resources are in the assembly folder. There you can specify an alternate location of the resources if desired.

```
lyphReaderLoader loader = new GlyphReaderLoader( "PathToFolderContainingGlyphReaderResources"
);
```

NOTE: When deploying GlyphReader to a Web Application, Web Service, or WCF service you need to load by Reflection.

Please see HOWTO: Load GlyphReaderEngine by Reflection

GlyphReaderEngine (v4.0 for DotImage 10.3.1 through 11.1)

NOTE:: this section is referring to GlyphReader v4.0 which is found in DotImage 10.3.1 through 11.1. Please see the section on GlyphReader Engine v3.0 for older version, and GlyphReader4Engine v5.0 for 11.2 and newer

The 4.0 GlyphReader engine requires an OcrResources folder with a GlyphReader sub-directory which itself has a v4.0 sub-directory containing the following 3 files and two folders:

- TOCR40.gar
- TOCR40.gnp
- TOCR40.teh
- \x64
- \x86

the x86 and x64 folders each contain the following files (of the correct "bitness")

- GlyphReader.dll
- GlyphReader.ini
- GlyphReaderEngine.exe

You will need to create a folder called OcrResources in your bin directory of your application, and copy the GlyphReader folder from

```
:\Program Files (x86)\Atalasoft\DotImage 10.x\bin\OCRResources\
```

into it (where x is the version of DotImage such as 10.3, 10.4, 10.5 and so on)

HOWTO: Deploy a project using OCR

You will also need to have a reference in your project to Atalasoft.dotImage.Ocr.GlyphReader.dll, but set the CopyLocal property to false, you then will need to physically place a copy of the correct version (x86/x64 and 2.0/4.0 for bitness and .NET framework respectively) of Atalasoft.dotImage.Ocr.GlyphReader.dll in the OCRResources folder

So, your application's bin folder will have

- bin\OCRResources\Atalasoft.dotImage.Ocr.GlyphReader.dll
- bin\OCRResources\GlyphReader\
- bin\OCRResources\GlyphReader\\OCRResources\GlyphReader\v4.0\
- bin\OCRResources\GlyphReader\v4.0\TOCR40.gar
- bin\OCRResources\GlyphReader\v4.0\TOCR40.gnp
- bin\OCRResources\GlyphReader\v4.0\TOCR40.teh
- bin\OCRResources\GlyphReader\v4.0\x64\
- bin\OCRResources\GlyphReader\v4.0\x64\GlyphReader.dll
- bin\OCRResources\GlyphReader\v4.0\x64\GlyphReader.ini
- bin\OCRResources\GlyphReader\v4.0\x64\GlyphReaderEngine.exe
- bin\OCRResources\GlyphReader\v4.0\x86\
- bin\OCRResources\GlyphReader\v4.0\x86\GlyphReader.dll
- bin\OCRResources\GlyphReader\v4.0\x86\GlyphReader.ini
- bin\OCRResources\GlyphReader\v4.0\x86\GlyphReaderEngine.exe

Due to the architecture of the GlyphReader engine, to specify a location other than a default search path such as System32, you'll need to create an instance of the OcrResourceLoader or GlyphReaderLoader in a static constructor before any OCR code is loaded. This is the case even if the resources are in the assembly folder. There you can specify an alternate location of the resources if desired.

```
lyphReaderLoader loader = new GlyphReaderLoader( "PathToFolderContainingGlyphReaderResources"
);
```

NOTE: When deploying GlyphReader to a Web Application, Web Service, or WCF service you need to load by Reflection.

Please see HOWTO: Load GlyphReaderEngine by Reflection

HOWTO: Deploy a project using OCR

GlyphReaderEngine (v3.0 for DotImage 10.3.0 and lower)

NOTE:: This section is referring to GlyphReader v3.0 which is found in DotImage 10.3.0 and older... as of 10.3.1 and newer, we use GlyphReader 4.0 which has a significantly different deployment profile.

The GlyphReaderEngine requires the following resource files:

- GlyphReader.dll
- GlyphReader.ini
- GlyphReaderEngine.exe
- TOCR32.gar
- TOCR32.n3s
- TOCR32.qnp
- TOCR32.teh

They are located by default in:

```
DK folder\OcrResources\GlyphReader\v3.0\
```

Due to the architecture of the GlyphReader engine, to specify a location other than a default search path such as System32, you'll need to create an instance of the OcrResourceLoader or GlyphReaderLoader in a static constructor before any OCR code is loaded. This is the case even if the resources are in the assembly folder. There you can specify an alternate location of the resources if desired.

```
lyphReaderLoader loader = new GlyphReaderLoader( "PathToFolderContainingGlyphReaderResources"
);
```

NOTE: When deploying GlyphReader to a Web Application, Web Service, or WCF service you need to load by Reflection.

Please see HOWTO: Load GlyphReaderEngine by Reflection

Tesseract5Engine (11.4.0.9 and newer)

The Tesseract5Engine replaces Tesseract3Engine

HOWTO: Deploy a project using OCR

For this engine, the default constructor will search the current bin directory for an OcrResources folder containing Tesseract\v5.3.0\Tessdata and containing one or more xxx.traineddata files. If it doesn't find them it will try searching the default Atalasoft OcrResources location:

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\

looking for the Tesseract folder and resources

You can also use the constructor overload that takes an OcrResources path and point it at a location where you've placed the Tesseract OCR files

example, you copy the CONTENTS OF

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\ folder to

d:\OcrResources\

then

Tesseract5Engine engine = new Tesseract5Engine(@"D:\OcrResources\");

If the engine.Initialize throws an exception you may need to check your path and ensure the full structure is present

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\deu.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\eng.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\fra.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\ita.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\nld.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\nor.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\por.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.4\bin\OCRResources\Tesseract\v5.3.0\Tessdata\spa.traineddata

Tesseract3Engine (Removed in 2026.2 - Consider moving to Tesseract5Engine available in 11.4.0.9 and newer)

The Tesseract3Engine replaces TesseractEngine

For this engine, the default constructor will search the current bin directory for an OcrResources folder containing Tesseract\v3.04\Tessdata and containing one or more xxx.traineddata files. If it doesn't find them it will try searching the default Atalasoft OcrResources location:

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\

looking for the Tesseract folder and resources

You can also use the constructor overload that takes an OcrResources path and point it at a location where you've placed the Tesseract OCR files

example, you copy the C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\ folder to d:\OcrResources\

then

Tesseract3Engine engine = new Tesseract3Engine(@"D:\OcrResources\");

if the engine.Initialize throws an exception you may need to check your path and ensure the full structure is present

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\deu.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\eng.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\fra.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\ita.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\nld.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\nor.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\por.traineddata

C:\Program Files (x86)\Atalasoft\DotImage 11.2\bin\OCRResources\Tesseract\v3.04\Tessdata\spa.traineddata

TesseractEngine (OBSOLETE)

NOTE: TesseractEngine was completely removed from our SDK in 11.1 - please move to Tesseract3Engine if you wish to continue using Tesseract

The TesseractEngine requires its resource files to be pointed at by an environment variable "TESSDATA_PREFIX". Install the "tessdata" folder to a location on the deployment machine and then in your project set the environment variable to the absolute path of "tessdata's" parent folder. You can use this call to accomplish this:

```
ystem.Environment.SetEnvironmentVariable("TESSDATA_PREFIX",
absolutepath,EnvironmentVariableTarget.User);
```

RecoStarEngine (OBSOLETE)

NOTE: RecoStarEngine was retired in 10.7 - this information provided for legacy support only

The RecoStar engine requires RecoStar Resources which are not distributed with DotImage by default, but which may be downloaded from here:

10.6.1 and newer:

http://www.atalasoft.com/download/Atalasoft.RecoStarResources.zip

10.6.0 and older:

http://www.atalasoft.com/download/DotImage-RecoStarResources.zip

Once downloaded, you can unzip (it will create a folder called RecoStar with a 7.2 (new as of 10.6.1 and newer... for older versions it will be 5.0) sub-directory and place several files and folders under there) and place the entire RecoStar folder into the default location (C:\Program Files (x86)\Atalasoft\DotImage 10.x\bin\OCRResources\ )

You'll need a RecoStar Loader as well:

```
RecoStarLoader loader = new RecoStarLoader( "PathToFolderContainingRecoStar" );
```

ExperVisionEngine (Obsolete)

ExperVision was in 4.0 and removed in 10.0

I.R.I.S. Ocr (Obsolete)

It was added in 10.0 and removed in 10.3

Original Article

Q10141 - HOWTO: Deploy a project using OCR

Last Update

2026-10-28 - TD - (Fixed old OCR removals)

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50303/HOWTO-Deploy-a-project-using-OCR