# SNMP Pro Documentation

***Release 1.2***

**Lex Li**

April 21, 2018

Contents

CHAPTER 1

# Topics

## Getting Started

### #SNMP MIB Compiler Pro Features

By Lex Li

This page shows you the main features of the Compiler Pro Edition.

**In this article:**



#### Background

#SNMP Library used to ship a compiler sample project, which only provides limited features such as syntax highlighting. The Pro Edition of #SNMP MIB Compiler, however, is designed and developed to include advanced features that target SNMP professionals and enterprises. It also comes with professional consulting and support services.

#### Supported Platforms

The Compiler Pro which requires .NET Framework 4.5 and Windows Vista and above.

#### Features

##### Accurate Error Reporting and Rich Metadata Extraction

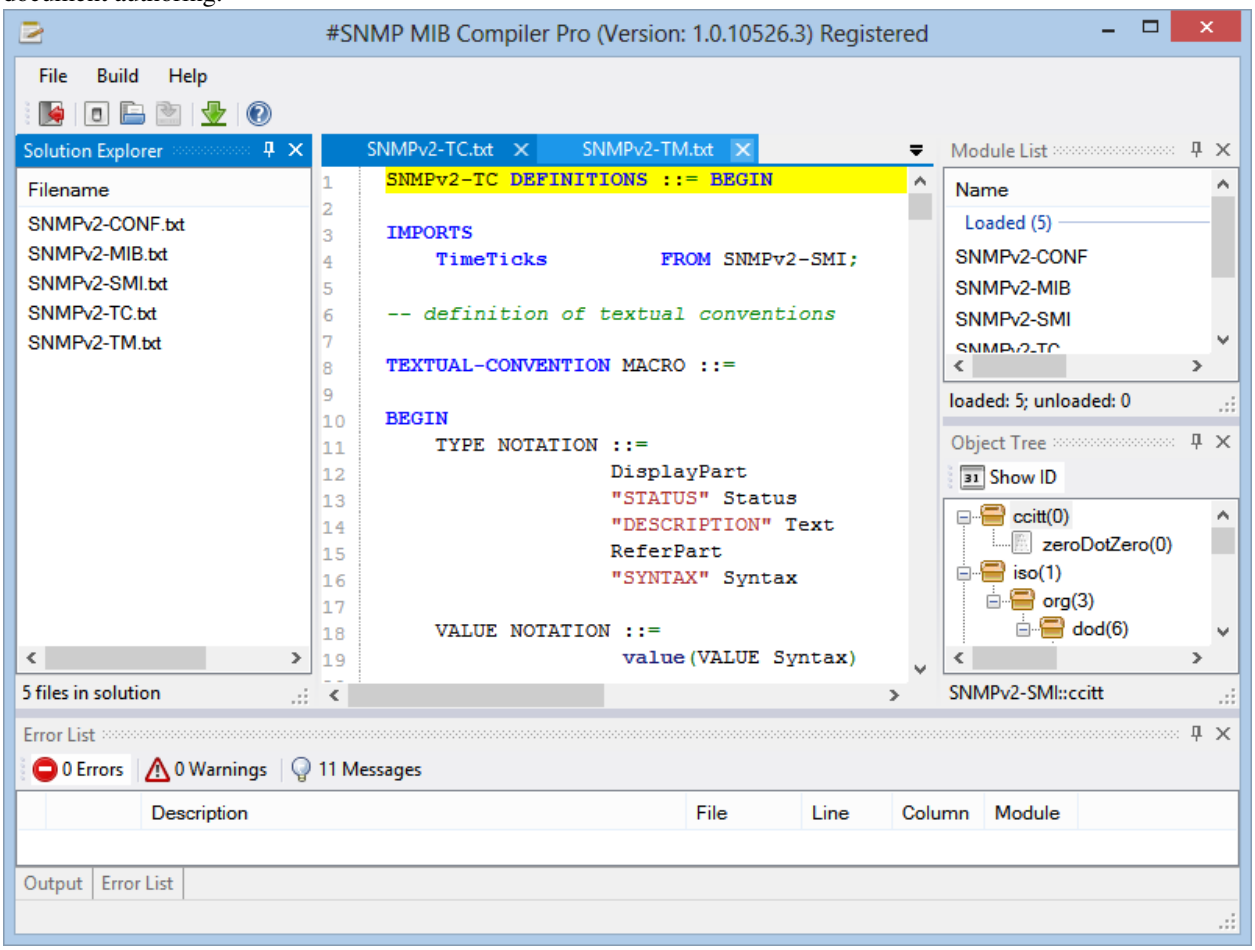
Type information extraction is a key requirement of a compiler that for every objects their basic types can be determined (Counter32, OCTET STRING, and so on). Even the intermediate types (such as DisplayString) are very useful for troubleshooting. In the meantime, as constraints can be added at each levels, such constraints play an important role in data validation.

#SNMP Compiler Pro can perform cross module dependency resolution, type resolution, and entity validation. Such tasks can provide more accurate error reporting and help identify broken MIB documents.

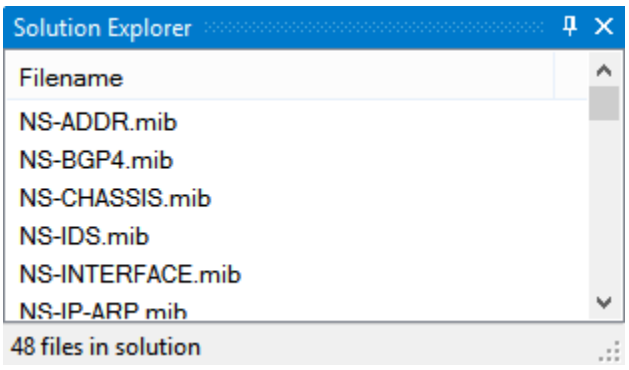
Based on the metadata collected from MIB documents, the compiler can generate appropriate C# code through compilation, which is similar to Net-SNMP's mib2c utility (who compiles MIB documents to C code). The output C# source files can be used to link with the snmpd sample in #SNMP Library so as to form a simulator for those MIB documents.

### Visual Studio Style Panels

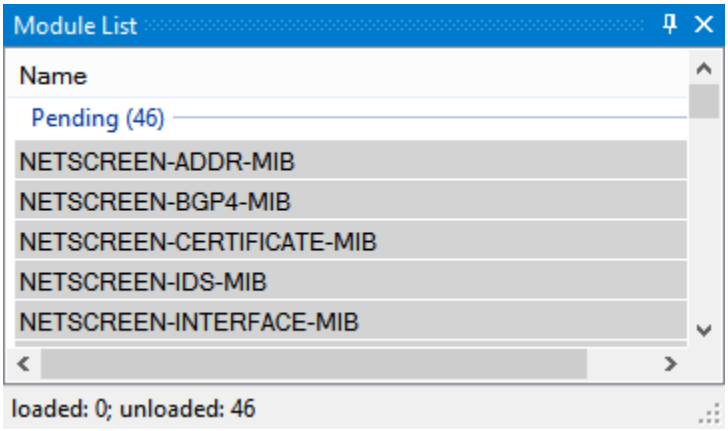
The compiler is designed to be similar to Visual Studio, with dock panels that show various information to assist MIB document authoring.



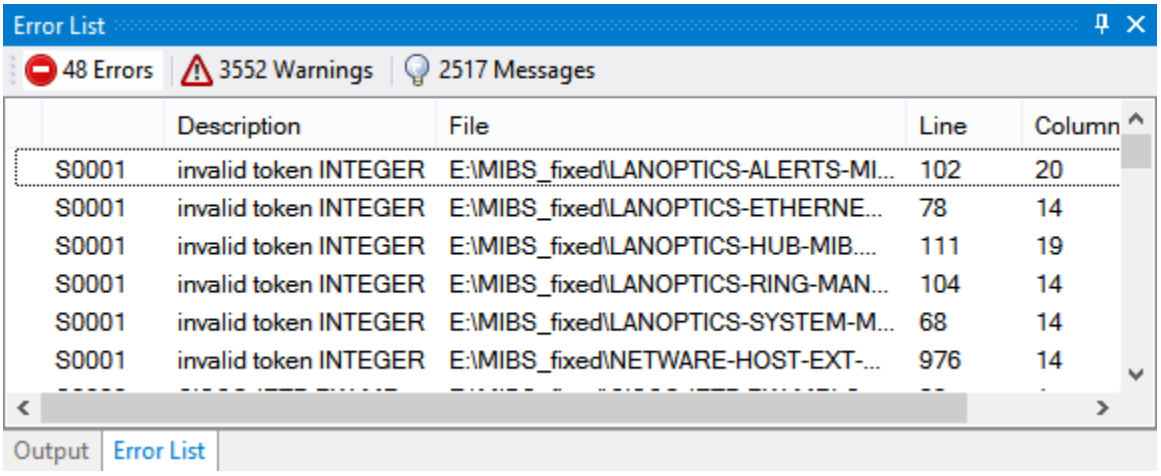

**Solution Explorer** MIB documents can be loaded here. Last used solution will be automatically loaded at startup.



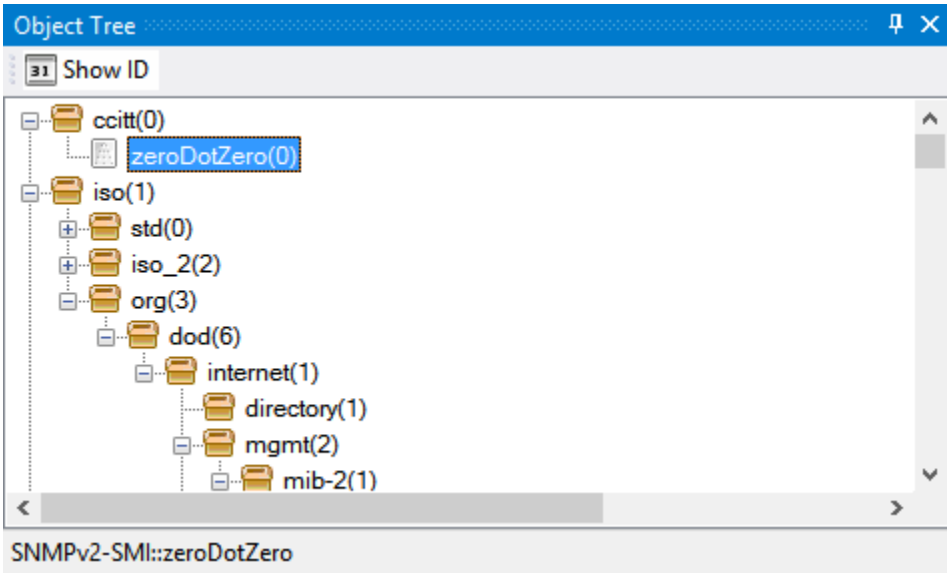
**Module List** Loaded and pending modules are displayed in this panel, so it is obvious which modules are not compiled.




**Error List** Compilation errors and warnings are displayed so it is easy to identify issues in documents.




**Object Tree** Objects from loaded modules are displayed so the whole object tree is one click away.




### Related Resources

- Purchase
- Requesting Trial
- SharpSnmpPro.Mib Assembly Features
- #SNMP Compiler Pro Trial Version Reviewers' Guide

## SharpSnmpPro.Mib Assembly Features

By Lex Li

This page shows you the main features of SharpSnmpPro.Mib assembly.

**In this article:**



### Background

#SNMP Suite ships with an assembly SharpSnmpLib.Mib which can compile MIB documents and extract some information from them. It only provides limited functionality and users ask for more advanced editions. Here it comes.

### The Brand New SharpSnmpPro.Mib Assembly

This assembly is the key component that empowers the Compiler Pro product.

### Supported Platforms

Unlike the Compiler Pro which requires .NET 4.5 and Windows, this assembly can be used on multiple platforms,

- .NET Framework 4.5.2 and above
- Mono 4.2.1 and above
- Xamarin.iOS Unified
- Xamarin.Android
- Xamarin.Mac

### Features

Now let us see a few common examples that reveal the power of this library.

#### MIB Document Compilation

The following code shows how to compile several essential MIB documents and load the metadata into memory,

```
var registry = new SimpleObjectRegistry();
var collector = new ErrorRegistry();
registry.Tree.Collector = collector;
registry.Import(Parser.Compile(new MemoryStream(Resources.SNMPv2_SMI), collector));
registry.Import(Parser.Compile(new MemoryStream(Resources.SNMPv2_CONF), collector));
registry.Import(Parser.Compile(new MemoryStream(Resources.SNMPv2_TC), collector));
registry.Import(Parser.Compile(new MemoryStream(Resources.SNMPv2_MIB), collector));
registry.Import(Parser.Compile(new MemoryStream(Resources.SNMPv2_TM), collector));
registry.Refresh();
```

`SimpleObjectRegistry` is a class that holds metadata in memory. It can be used to import metadata generated via `Parser.Compile`.

`ErrorRegistry` is the container of errors and warnings. When `SimpleObjectRegistry.Refresh` is called, all errors and warnings can be found in the `ErrorRegistry` instance, to help you identify why some MIB documents cannot be compiled or imported.

---

**Note:** The Trial version lacks of certain validation so it might not report all the issues, while the Full version does not have such limitations.

---

`SimpleObjectRegistry.Tree.PendingModules` can be queried to see a list of modules that fail to be loaded.

Similarly, `SimpleObjectRegistry.Tree.LoadedModules` is a list of loaded modules.

#### Name/OID Translation

Once the metadata are extracted from MIB documents, an obvious application of them is to enable OID translation. The translation is bi-directional (from names to OIDs or vice verse).

```
const string textual = "SNMPv2-SMI::zeroDotZero";
var number = new uint[] { 0, 0 };
Assert.AreEqual(textual, registry.Translate(number));
Assert.AreEqual(number, registry.Translate(textual));
```

Here the `Translate` method only provides a single textual form of the OID, while in some cases a single OID can have many textual forms (as it might be defined in multiple documents with different modules and names). Below is the code to get all the textual forms of the OID,

```
var number = new uint[] { 0, 0 };
var searchResult = registry.Tree.Search(number);
var definition = searchResult.Definition;
var textualForms = definition.TextualForms;
```

### Extract Object Identifier Metadata

Once all metadata are loaded in a `SimpleObjectRegistry` instance we can easily extract the information for individual objects,

```
Definition item = registry.Tree.Find("SNMPv2-MIB", "sysDescr");
IEntity entity = item.DisplayEntity;
Assert.AreEqual("A textual description of the entity.  This value should include the full name and ve
Assert.AreEqual(EntityStatus.Current, entity.Status);
Assert.AreEqual(string.Empty, entity.Reference);

var obj = entity as IObjectTypeMacro;
Assert.AreEqual(Access.ReadOnly, obj.MibAccess);
Assert.AreEqual(SnmpType.OctetString, obj.BaseSyntax);
```

We can see that if we are looking for `SNMPv2-MIB::sysDescr` (whose OID is `1.3.6.1.2.1.1.1`), we can use `SimpleObjectRegistry.Tree.Find` method to locate the `Definition` instance. Each such instance contains one or more `IEntity` instances to match their entity definition in MIB documents.

From `Definition.DisplayEntity` we can get one of the entities, and check its properties such as `IEntity.DescriptionFormatted`, `IEntity.Status`, and `IEntity.Reference`.

Since `SNMPv2-MIB::sysDescr` is an `OBJECT-TYPE` macro entity, we can further cast it to `IObjectTypeMacro` to access more properties, such as `IObjectTypeMacro.MibAccess` and `IObjectTypeMacro.BaseSyntax`. It is obvious that the data type of `SNMPv2-MIB::sysDescr` is `OCTET STRING`.

There are of course other properties you can review, which are documented online at the help site .

**Note:** The Trial version limits which attributes you can see, while the Full version does not have such limitations.

### Table Validation

With MIB documents, it is very easy to determine if an OID is a table, a table entry, or a table column.

```
var table = new ObjectIdentifier(new uint[] { 1, 3, 6, 1, 2, 1, 1, 9 });
var entry = new ObjectIdentifier(new uint[] { 1, 3, 6, 1, 2, 1, 1, 9, 1 });
var unknown = new ObjectIdentifier(new uint[] { 1, 3, 6, 8, 18579, 111111 });
Assert.IsTrue(registry.ValidateTable(table));
Assert.IsFalse(registry.ValidateTable(entry));
Assert.IsFalse(registry.ValidateTable(unknown));
```

By accessing `Children` property of a table object, the entry of that table can be queries.

Similarly, by accessing `Children` property of an entry object, the columns of the table can be queried easily.

#### Input Data Validation

In SNMP managers or agents, it is a common need to determine if a piece of data is valid for an OID. Various constraints can be defined at MIB document level, but it is often difficult to extract that from the files. With a few lines of code you can now do that

```
Assert.IsTrue(registry.Verify("SNMPv2-MIB", "sysDescr", new OctetString("test")));
Assert.IsTrue(registry.Verify("SNMPv2-MIB", "sysDescr", new OctetString(string.Empty)));
Assert.IsFalse(registry.Verify("SNMPv2-MIB", "sysDescr", new Integer32(2)));
```

We can easily test if the data is valid for `SNMPv2-MIB::sysDescr`.

---

**Note:** The Trial version does only support data validation against a limited set of default types (defined in core MIB documents), while the Full version supports even custom types such as `BITS`, `CiscoRowOperStatus`, and `CiscoPort`.

---

### Related Resources

- Purchase
- API Documentation
- Requesting Trial
- #SNMP MIB Compiler Pro Features
- SharpSnmpPro.Mib Assembly Trial Version Reviewers' Guide
- SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide

# Tutorials

## #SNMP Compiler Pro Trial Version Reviewers' Guide

By Lex Li

This page shows you a guide on #SNMP Compiler Pro Trial version.

**In this article:**



### Background

The Trial version can be requested here , and is packaged up with the latest #SNMP Library and SharpSnmpPro.Mib.

### Evaluation Steps

To test it out, the default test MIB documents can be found at GitHub . It can be cloned to a local folder, such as `D:\sharpsnmppro-mib` .

```
git clone https://github.com/lextm/sharpsnmppro-mib.git
```

Then the documents can be copied to that folder (`D:\sharpsnmppro-mib` for example).

Launch the compiler by executing `Compiler.exe` and then click File | Open menu item to navigate to `D:\sharpsnmppro-mib` folder in Open file dialog.

Change the file extension filter to All files (.), and then select all .txt files in this folder. Click Open button to open all files in the compiler. The file names should appear in Solution Explorer panel.

Click Build | Compile menu item to start compiling the files. In a few seconds, the Object Tree panel should be updated with objects extracted from the files, while Module List panel shows the loaded modules (as well as pending ones). The Error List panel should display any error or warnings. The Output panel contains the diagnostics logging entries.

### Related Resources

- Purchase
- Requesting Trial
- #SNMP MIB Compiler Pro Features
- #SNMP Compiler Pro Full Version Reviewers' Guide
- SharpSnmpPro.Mib Assembly Trial Version Reviewers' Guide
- SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide

## #SNMP Compiler Pro Full Version Reviewers' Guide

By Lex Li

This page shows you a guide on #SNMP Compiler Pro Full version.

**In this article:**



### Background

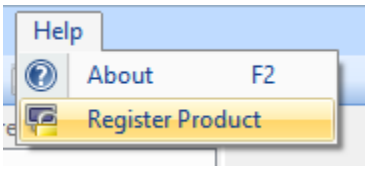
Once you finish the purchase, please follow the these steps to request a license file to activate the compiler. This license file converts the Trial version to Full version.

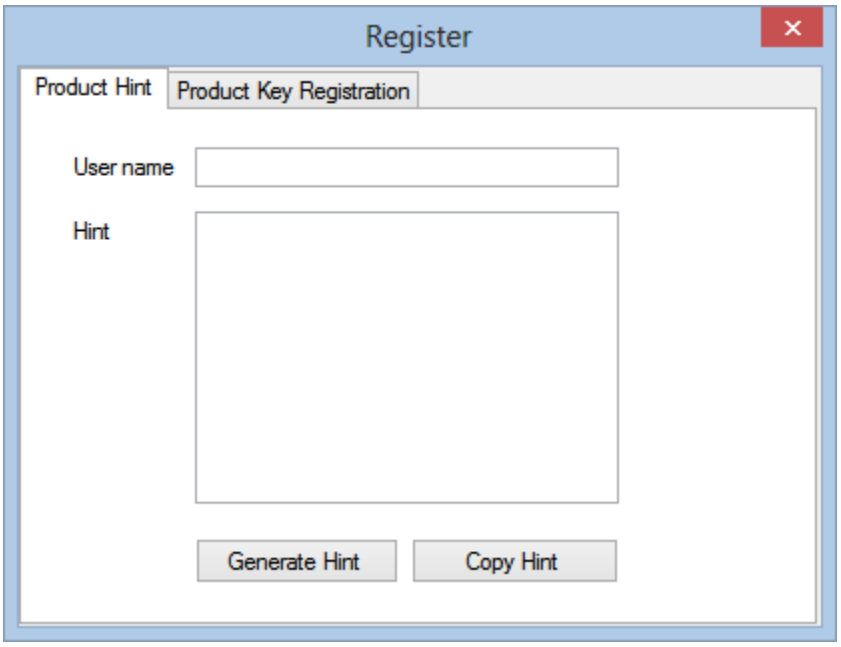
### Generate License Hint

1. Launch the compiler by double clicking Compiler.exe.
2. Go to Help | Register Product menu item.
3. In Register dialog, fill in the User name field and click Generate Hint button. The hint data will be generated in the Hint field.
4. Click Copy Hint button to copy the data to clipboard.

5. Exit the Register dialog by clicking the red cross on top right corner.

Please paste that data to an email and send to the support team .

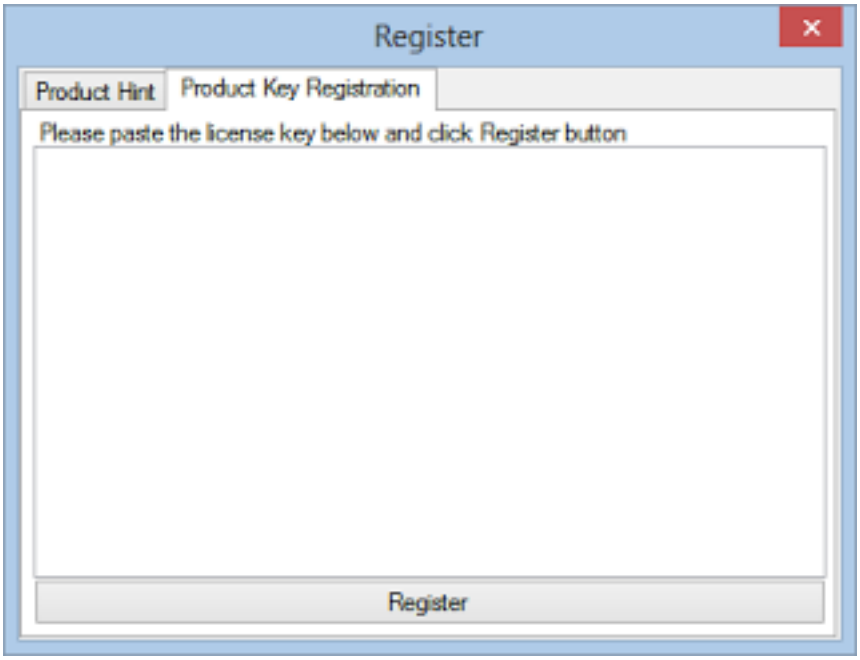



### License Activation

Once the license key arrives via email, it is time to perform the following steps to get the compiler activated,

1. Launch the compiler by double clicking Compiler.exe.
2. Go to Help | Register Product menu item.
3. In Register dialog, switch to Product Key Registration tab.
4. Paste the license key to the text box and click Register.
5. Exit the Register dialog by clicking the red cross on top right corner.




### Related Resources

- Purchase
- Requesting Trial
- #SNMP MIB Compiler Pro Features
- #SNMP Compiler Pro Trial Version Reviewers' Guide
- SharpSnmpPro.Mib Assembly Trial Version Reviewers' Guide
- SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide

## SharpSnmpPro.Mib Assembly Trial Version Reviewers' Guide

By Lex Li

This page shows you a guide on SharpSnmpPro.Mib assembly trial version.

**In this article:**



### Background

The Trial Edition can be requested here , and is packaged up with the latest #SNMP Library. So below are the assemblies in the ZIP package,

- SharpSnmpPro.Mib.Trial.dll

- SharpSnmpLib.dll
- SharpSnmpPro.Mib.Trial.2.0.0.nupkg

---

**Note:** For 1.2 release, the following are included in the ZIP package,

- SharpSnmpPro.Mib.Trial.dll
- SharpSnmpLib.Full.dll
- SharpSnmpLib.Portable.dll

---

### Supported Platforms

Unlike the Compiler Pro which requires .NET 4.5 and Windows, this assembly can be used on multiple platforms,

- .NET Framework 4.5.2 and above
- Mono 5.0 and above (via .NET Standard 2.0)
- Xamarin.iOS Unified (via .NET Standard 2.0)
- Xamarin.Android (via .NET Standard 2.0)
- Xamarin.Mac (via .NET Standard 2.0)

### Evaluation Steps

To test it out, the default test projects can be found at GitHub . It can be cloned to a local folder, such as `D:\sharpsnmppro-sample`.

```
git clone https://github.com/lextm/sharpsnmppro-sample.git
git checkout release_2.0
```

---

**Note:** For 1.2 release, execute the following commands,

```
git clone https://github.com/lextm/sharpsnmppro-sample.git
git checkout release_1.2
```

---

Then extract all the files from the ZIP package to that folder (`D:\sharpsnmppro-sample` for example).

Lastly, execute an extra script to prepare the NuGet local feed,

```
    install.nuget.bat
```

---

**Note:** For 1.2 release, there is no `install.nuget.bat`.

---

In Visual Studio you can analyze the two projects in `Tests.sln`.

`Tests.csproj` is an NUnit project that shows the below,

- How to compile and load MIB documents.
- How to query entity by name.
- How to check description of entities.
- How to verify data against entities.
- How to check `OBJECT-TYPE` macro specific properties.

**Note:** The trial edition only support simple entities, while the full edition supports all entities.

`snmptranslate.csproj` is a console application project that illustrates how to translate OIDs to strings (and vice versa) based on compiled MIB documents.

If new projects are going to be created to test out the Trial Edition, please note that you need to follow the provided samples to,

1. Include a text file named `sharpsnmppro.txt`.
2. The text file must contain exactly "#SNMP MIB Compiler Pro" (without quotes).
3. Build Action for this text file must be set as Embedded Resource.

Without this text file, the Trial Edition should give you an exception with error message "This assembly is not licensed to you. Please buy a license from LeXtudio...".

The API reference documentation can be found on the help site .

### Related Resources

- Purchase
- API Documentation
- Requesting Trial
- SharpSnmpPro.Mib Assembly Features
- SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide
- #SNMP Compiler Pro Trial Version Reviewers' Guide
- #SNMP Compiler Pro Full Version Reviewers' Guide

## SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide

By Lex Li

This page shows you a guide on SharpSnmpPro.Mib assembly full version.

**In this article:**



### Background

The Full version is sent to registered users only via emails, and is packaged up with latest #SNMP Library. So below are the assemblies in the ZIP package,

- SharpSnmpPro.Mib.Trial.dll
- SharpSnmpLib.dll
- SharpSnmpPro.Mib.Trial.2.0.0.nupkg

**Note:** For 1.2 release, the following are included in the ZIP package,

- SharpSnmpPro.Mib.Trial.dll
- SharpSnmpLib.Full.dll
- SharpSnmpLib.Portable.dll

### Supported Platforms

Unlike the Compiler Pro which requires .NET 4.5 and Windows, this product can be used on multiple platforms,

- .NET Framework 4.5.2 and above
- Mono 5.0 and above (via .NET Standard 2.0)
- Xamarin.iOS Unified (via .NET Standard 2.0)
- Xamarin.Android (via .NET Standard 2.0)
- Xamarin.Mac (via .NET Standard 2.0)
- Other platforms that are compliant to .NET Standard 2.0.

**Note:** More information about .NET Standard 2.0 can be found from Microsoft .

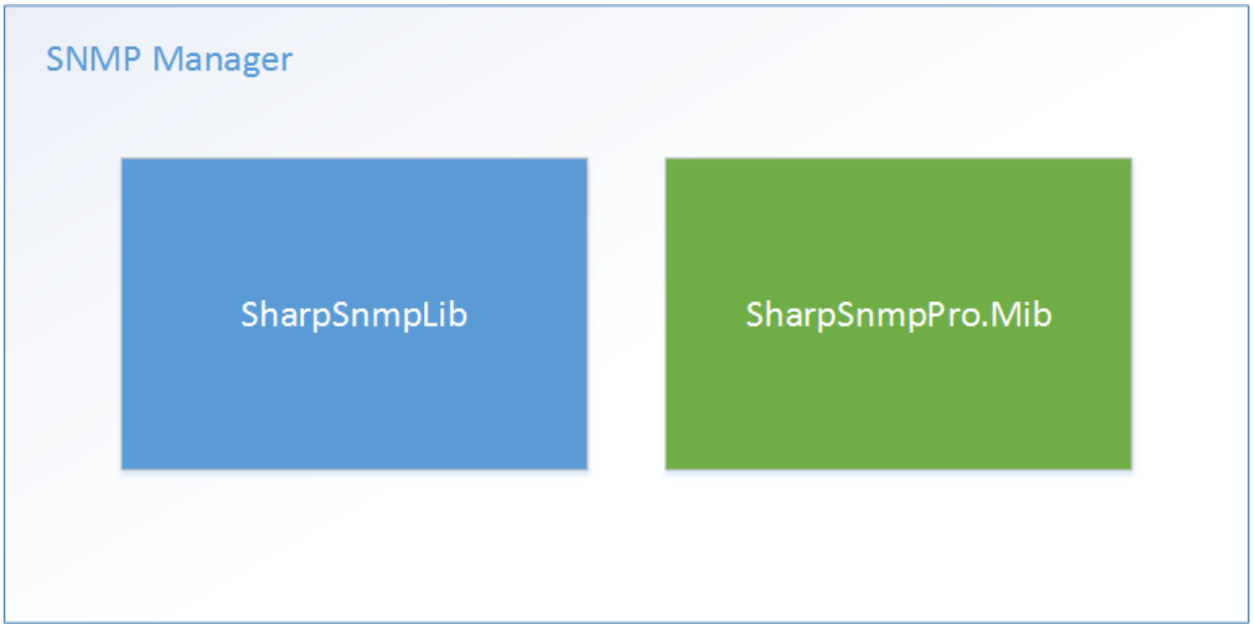
### Complete Sample Project

You learn how SNMP operations can be done by consuming the open source SNMP API. A question then is what values MIB documents provide, as they are said to be an important part of SNMP protocol but not seem to be utilized anywhere if we solely use #SNMP Library.

Well, a rough answer is MIB documents mean everything,

- They tell what each object identifiers (OID) mean.
- They tell which OID is for a table, a row, and a column.
- They tell which kind of data we should expect for an object, `OCTET STRING` or any other valid types.

Thus, a MIB specific library such as SharpSnmpPro.Mib can help build a much more powerful SNMP manager.




To test it out, we reuse the default test projects for Trial version, which can be found at GitHub. It can be cloned to a local folder, such as `D:\sharpsnmppro-sample` .

```
git clone https://github.com/lextm/sharpsnmppro-sample.git
git checkout release_2.0
```

**Note:** For 1.2 release, execute the following commands,

```
git clone https://github.com/lextm/sharpsnmppro-sample.git
git checkout release_1.2
```

Then extract all the files from the ZIP package to that folder (`D:\sharpsnmppro-sample` for example).

Lastly, execute an extra script to prepare the NuGet local feed,

```
    install.nuget.bat
```

**Note:** For 1.2 release, there is no `install.nuget.bat`.

`Tests.csproj` is an NUnit project that shows the below,

- How to compile and load MIB documents.
- How to query entity by name.
- How to check description of entities.
- How to verify data against entities. (note that the trial edition only support simple entities, while the full edition supports all entities).
- How to check OBJECT-TYPE macro specific properties.

To make the test project work with Full version, the following changes need to be made,

1. Modify `snmptranslate.csproj` and `Tests.csproj` to use `SharpSnmpPro.Mib` as package reference, instead of `SharpSnmpPro.Mib.Trial`.
2. Modify `sharpsnmppro.txt` following the instructions in the email.

3. Remove `TRIAL` from "Conditional compilation symbols", which then enables Full version only test cases.

---

**Note:** For 1.2 release, the following changes need to be made,

1. Remove the original reference to `SharpSnmpPro.Mib.Trial.dll`.
2. Add a new reference to `SharpSnmpPro.Mib.dll`.
3. Modify `sharpsnmppro.txt` following the instructions in the email.
4. Remove `TRIAL` from "Conditional compilation symbols", which then enables Full version only test cases.

---

#### Related Resources

- Purchase
- API Documentation
- Requesting Trial
- SharpSnmpPro.Mib Assembly Features
- SharpSnmpPro.Mib Assembly Trial Version Reviewers' Guide
- #SNMP Compiler Pro Trial Version Reviewers' Guide
- #SNMP Compiler Pro Full Version Reviewers' Guide

# Support Information

## SharpSnmpPro.Mib Release Notes

By Lex Li

This page documents major releases of SharpSnmpPro.Mib.

**In this article:**



### Releases

#### 2.0.0

This is a major release with new platform support. Major changes:

Bug fixes

- Compiler performance and stability is improved .

New Features

- .NET Standard 2.0 support is added .
- RFC 2578 descriptor prefix is supported .

- RFC 2578 restriction on imported items is applied .
- RFC 1155 support is improved .
- Moved a few classes to new namespaces .
- Added overloading methods to ObjectRegistryBase support OID input .

**Note:** This release depends on #SNMP Library 10.0.2 and above.

#### 1.2.0

This is a bug fix release with some new APIs. Major changes:

Bug fixes

- zeroDotZero DisplayEntity.GetObjectIdentifier is fixed .
- jmgt DisplayEntity.GetObjectIdentifier is fixed .
- Table index verification is improved to avoid false alarms .
- Special SMIv1 -> SMIv2 converted TRAP items are properly handled .
- OID resolution is revised to work better with multiple references to the same tree node .

New features

- A new Index type is added and implied flag is supported .
- A new Augments type is added.
- New types such as ContraintedType are added to provide initial constraint related support.

Tutorials on the new features will be added to related sections.

**Note:** This release depends on #SNMP Library 9.0.5 and above.

#### 1.1.3

This is a bug fix release. Major changes:

- Parser2.Compile exceptions are handled properly.

#### 1.1.2

This is a bug fix release. Major changes:

- Assistance OIDs are no longer generated.
- Duplicate module detection is added.

#### 1.1.1

This is a bug fix release. Major changes:

- .NET Framework 4.0 is no longer supported. .NET 4.5 and above is required.

#### 1.1.0

This is a bug fix release. Major change:

- More internal types are now exposed as public.

#### 1.0.0

Initial release.

### Product Lifecycle

The lifecycle of the releases are listed below,

| Version | Release Date | End-of-life Date |
|---|---|---|
| 2.0.0 | Dec 31, 2017 | N/A |
| 1.2.0 | Jan 16, 2017 | Mar 31, 2018 |
| 1.1.3 | Oct 1, 2016 | Apr 16, 2017 |
| 1.1.2 | Sep 15, 2016 | Oct 1, 2017 |
| 1.1.1 | Jul 14, 2016 | Sep 15, 2017 |
| 1.1.0 | Feb 22, 2015 | Jul 14, 2017 |
| 1.0.0 | Feb 3, 2014 | Feb 22, 2016 |

The old rule (1.0.0-1.1.2) is that any release will be actively supported till its next release becomes one year old.

The new rule (1.1.3 and above) is that any release will be actively supported till its next release becomes three months old. This change is to match our more frequent release cycles.

Users of expired releases must upgrade to an active supported release before contacing technical support team.

### Related Resources

- Purchase
- API Documentation
- Requesting Trial
- #SNMP MIB Compiler Pro Features
- SharpSnmpPro.Mib Assembly Features

## Purchase

By Lex Li

This page shows you how to purchase the Pro Edition.

**In this article:**

### #SNMP Pro Bundle

Buy #SNMP MIB Compiler Pro + SharpSnmpPro.Mib Assembly Bundle here at **799 USD**

**Note:** Starting from 1.1.2 release, the products are no longer sold individually.

### Source Code Availability

All above are binaries sold only. Source code is sold separately. Please contact the support team for more information.

### Related Resources

- Support Services
- Requesting Trial
- SharpSnmpPro.Mib Assembly Full Version Reviewers' Guide
- #SNMP Compiler Pro Full Version Reviewers' Guide

## Licensing Terms

By Lex Li

This page shows you how the licensing works for the Pro Edition.

**In this article:**



### Licensing Model for #SNMP MIB Compiler Pro

Lex Li issues a unique license file for each ordered software license. The user then applies the license file to the installed software. Once the software license is applied, the licensed software product displays 'Registered' in title. Each license entitles the user to install and use the licensed software on only one computer, i.e., each license file can be applied to at most one copy of the installed software.

Floating license is not supported. Due to the nature of the software, the software does not contain any functions for floating license checking. Many users install the software on their laptops and then use it somewhere in the field. It might be inconvenient for them if the software was trying to connect to some kind of licensing server in order to determine if it is eligible to start-up or not, depending on the current number of other users.

Without signing an appropriate software licensing agreement with Lex Li it is not permitted to bundle any purchased products and license files with your products and ship them to your customers or end-users.

The MIB documents authored or the C# source files generated are the intellectual properties of the licensees and Lex Li does not request any permission on them.

### Evaluation License for #SNMP MIB Compiler Pro

#SNMP MIB Compiler Pro is available for a 15-day evaluation, so that potential customers may try using the software before ordering the regular software licenses. Under the 15-day evaluation license the user should determine the suitability of the evaluated software product, its available functions, performance capabilities and similar features. Under the 15-day evaluation license the user is not permitted to use the evaluated product for any kind of commercial purposes. More specifically, with the product running under the evaluation license it is not permitted to test or debug equipment for commercial purposes, it is not permitted to provide commercial training services, nor any similar commercial activity. Such activities can be performed only after obtaining the licensed version of the software.

If the user determines the suitability of the evaluated software product, the regular software license should be obtained. Otherwise, the user must uninstall and stop using the evaluation software.

### Licensing Model for SharpSnmpPro.Mib Assembly

When purchasing a license for the SharpSnmpPro.Mib Assembly, you receive a licensed copy for one developer and unlimited number of redistributable run-time licenses.

This means that one software developer can install the licensed version of the SharpSnmpPro.Mib Assembly to his or her computer. The redistributable run-time licenses can be deployed in the development environment in order to test and debug the product through its development stages. After the product is fully developed, the redistributable run-time licenses can be deployed (bundled with your application) either within your company (i.e., if you were developing an in-house tool) and/or sold to your end users (again, bundled with your application). You are eligible to redistribute unlimited copies of SharpSnmpPro.Mib Assembly.

### Evaluation License for SharpSnmpPro.Mib Assembly

SharpSnmpPro.Mib is available for a 15-day evaluation, so that potential customers may try using the software before ordering the regular software licenses. Under the 15-day evaluation license the user should determine the suitability of the evaluated software product, its available functions, performance capabilities and similar features. Under the 15-day evaluation license the user is not permitted to use the evaluated product for any kind of commercial purposes. More specifically, with the product running under the evaluation license it is not permitted to test or debug equipment for commercial purposes, it is not permitted to provide commercial training services, nor any similar commercial activity. Such activities can be performed only after obtaining the licensed version of the software.

If the user determines the suitability of the evaluated software product, the regular software license should be obtained. Otherwise, the user must uninstall and stop using the evaluation software.

### Related Resources

- Purchase
- Requesting Trial
- #SNMP MIB Compiler Pro Features
- SharpSnmpPro.Mib Assembly Features

## Support Services

By Lex Li

This page shows you what are the support services provided by LeXtudio.

**In this article:**



## Professional Support Services

Professional Support helps you address problems encountered with the development, deployment and management of #SNMP products (#SNMP Library, its samples, and Pro Edition) in business environments.

Professional Support is available as a single "pay-per-incident" (PPI) or an annual contract with five incidents. Professional Support incidents focus on troubleshooting a specific problem, error message, or functionality that is not working as intended for #SNMP products. An incident is defined as a single support issue and the reasonable effort to resolve it. Incidents may be submitted online. Response time will be between 2 and 8 hours, depending on severity of incident.

| Contract Types | Price | To Purchase | Business hours | To submit incident |
|---|---|---|---|---|
| Professional Support Single Incident | $259 USD for one incident | Buy Single | 7:00 PM – 10:00 PM Eastern Time Zone, Monday through Friday, excluding public holidays | Get Started |
| Professional Support 5-Pack Annual Support Contract | $1289 USD for five incidents | Buy Multiple | 7:00 PM – 10:00 PM Eastern Time Zone, Monday through Friday, excluding public holidays | Get Started |

Business-critical after-hours support is available for $515 USD per incident and provides support only for business-critical issues. Service is available after business hours, on weekends, and during public holidays. To purchase after-hours support, please contact the support team .

## Advisory Services

Professional Advisory services are designed for developers, IT professionals and small and medium business customers to provide advice and information to enable customers to deliver their own solution, e.g. providing recommendations and best practices for design, development and deployment, leveraging #SNMP products and technologies. This offering does not include hands on configuration, writing custom code, onsite consulting, root cause analysis or account management.

Professional Advisory services are provided for the same price as a Professional Support incident, and the delivery time is expected be similar to a support incident, typically 2 to 3 hours. Requests may be rescoped to fit the limited delivery time or the service may be declined. Professional Advisory services are available during normal business hours: 7:00 PM – 10:00 PM Eastern Time Zone Monday to Friday, excluding public holidays. Response time will be 8 hours. Simply contact the support team , stating that you want Professional Advisory Services.