ComponentOne XapOptimizer Overview

Size matters in Silverlight. In one click you can reduce the size of your Silverlight apps up to 70% and secure your code with obfuscation. Add **ComponentOne XapOptimizer** to your dev cycle with build automations, backups, and limitless options. Optimize your Silverlight applications (XAP files) with **ComponentOne XapOptimizer**. Without any loss in functionality, you get an application with improved start-up time, reducing network traffic.

Getting Started

Get started with the following topics:

- Key Features

- XAP Files|tag=About XAP Files;document=C1XapOptimizer.doc

- Getting Started|tag=Getting Started with ComponentOne XapOptimizer;document=C1XapOptimizer.doc

Installing ComponentOne XapOptimizer

The following sections provide helpful information on installing **ComponentOne XapOptimizer**.

ComponentOne XapOptimizer Setup Files

The **ComponentOne XapOptimizer** installation program will create the following directory: **C:\Program Files\ComponentOne\XapOptimizer**. This directory includes the EULA (End-user License Agreement), help file and additional licensing information.

System Requirements

System requirements for **ComponentOne XapOptimizer** include the following:

- .NET 3.5 Framework or later
- SN.exe (Optional; The Strong Name Tool, typically installed with Visual Studio, is used to create strong-named assemblies and is only required when signing optimized assemblies.)

Installing Demonstration Versions

If you wish to try **ComponentOne XapOptimizer** and do not have a serial number, follow the steps through the installation wizard and use the default serial number.

The only difference between unregistered (demonstration) and registered (purchased) versions of our products is that the registered version will stamp every application you compile so a ComponentOne banner will not appear when your users run the applications.

Activating ComponentOne Licensing

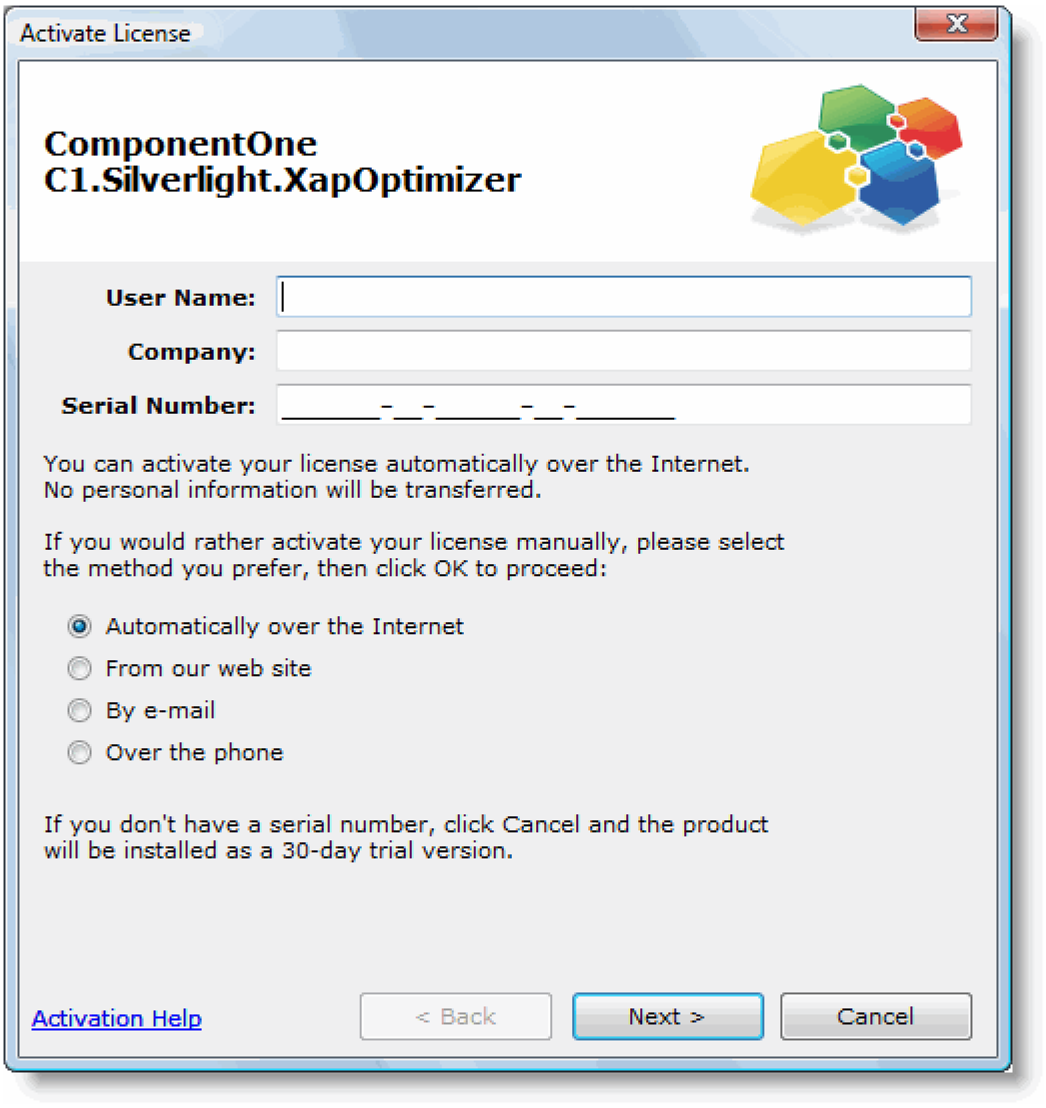
The **ComponentOne XapOptimizer** trial period begins when the product is installed, and it is available for 30 days. Prior to the end of the trial, you must activate **ComponentOne XapOptimizer** in order to continue using them. If you have installed the trial version and then purchased **ComponentOne XapOptimizer**, follow these steps in order to activate it:

1. Select the **About** button in the upper-right corner of the **XapOptimizer** application.

   The **About XapOptimizer** screen will open.

2. Click the **Activate** button.

   The **Activate License** dialog box will appear:



Activate License

ComponentOne
C1.Silverlight.XapOptimizer

User Name:
Company:
Serial Number: ______-__-______-__-______

You can activate your license automatically over the Internet.
No personal information will be transferred.

If you would rather activate your license manually, please select the method you prefer, then click OK to proceed:

- [✓ Automatically over the Internet]
- From our web site
- By e-mail
- Over the phone

If you don't have a serial number, click Cancel and the product will be installed as a 30-day trial version.

3. In the **Activate License** dialog box:
   a. Enter your name in the **User Name** text box.
   b. Enter your company name in the **Company** dialog box.
   c. Enter the serial number you received when you purchased the product in the **Serial Number** text box.
   d. Select an activation method:

Method	Description
Automatically over the	This is the default method. If you have an Internet connection,

Internet	**ComponentOne XapOptimizer** will be automatically activated.
From our web site	The activation wizard will give you an authentication number. Browse to the Web site and enter your serial number and authentication number to receive an Activation Code.
By e-mail	The activation wizard will give you an authentication number. Email your serial number and authentication number to activate@componentone.com and a customer service representative will provide you with an Activation Code.
Over the phone	The activation wizard will give you an authentication number. Call the Activation hotline at 866.379.0274 (U.S. and Canada) or 412.681.0711 (International) 9:00 a.m. to 5:00 p.m. EST Monday through Friday, provide your serial and authentication numbers, and a customer service representative will provide you with an Activation Code.

e. Click **Next** and follow the steps in the **Activate License** dialog box to receive an Activation Code. Note that if you activate **Automatically over the Internet**, you do not have to do anything else and the process is complete. If you choose another method, click **Next** again to continue.

4. Once the license is activated, the dialog box will ask if you want to register the product. Click **Yes** or **No**; however, we recommend clicking **Yes** so we can notify you of product updates and upgrades.

   You can also choose to register the product later from the **About** screen of the **ComponentOne XapOptimizer** application.

Deactivating ComponentOne Licensing

If you need to deactivate ComponentOne licensing for any reason, click open **ComponentOne XapOptimizer**, open the **About** screen, and choose **License Deactivation**.

Uninstalling ComponentOne XapOptimizer

To uninstall ComponentOne XapOptimizer:

1. Open the **Control Panel** and select the **Add or Remove Programs**.
2. Select **ComponentOne XapOptimizer** and click the **Remove** button.
3. Click **Yes** to remove the program.

Technical Support

ComponentOne offers various support options. For a complete list and a description of each, visit the ComponentOne Web site at http://www.componentone.com/Support.

Some methods for obtaining technical support include:

- **Online Support via HelpCentral**
  ComponentOne HelpCentral provides customers with a comprehensive set of technical resources in the form of FAQs, samples, Version Release History, Articles, searchable Knowledge Base, searchable Online Help and more. We recommend this as the first place to look for answers to your technical questions.

- **Online Support via our Incident Submission Form**
  This online support service provides you with direct access to our Technical Support staff via an online incident submission form. When you submit an incident, you'll immediately receive a response via e-mail confirming that you've successfully created an incident. This email will provide you with an Issue Reference ID and will provide you with a set of possible answers to your question from our Knowledgebase. You will receive a response from one of the ComponentOne staff members via e-mail in 2 business days or less.

- **Peer-to-Peer Product Forums and Newsgroups**
  ComponentOne peer-to-peer product forums and newsgroups are available to exchange information, tips, and techniques regarding ComponentOne products. ComponentOne sponsors these areas as a forum for users to share information. While ComponentOne does not provide direct support in the forums and

newsgroups, we periodically monitor them to ensure accuracy of information and provide comments when appropriate. Please note that a ComponentOne User Account is required to participate in the ComponentOne Product Forums.

- **Installation Issues**
  Registered users can obtain help with problems installing ComponentOne products. Contact technical support by using the online incident submission form or by phone (412.681.4738). Please note that this does not include issues related to distributing a product to end-users in an application.

- **Documentation**
  ComponentOne documentation is installed with each of our products and is also available online at HelpCentral. If you have suggestions on how we can improve our documentation, please email the Documentation team. Please note that e-mail sent to the Documentation team is for documentation feedback only. Technical Support and Sales issues should be sent directly to their respective departments.

Note: You must create a ComponentOne Account and register your product with a valid serial number to obtain support using some of the above methods.

About This Documentation

Acknowledgements

Microsoft, Windows, Windows Vista, and Visual Studio, are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Red Gate and .NET Reflector are trademarks or registered trademarks of Red Gate Software Ltd. Firefox is a registered trademark of the Mozilla Foundation. Safari is a trademark of Apple Inc., registered in the U.S. and other countries.

ComponentOne

If you have any suggestions or ideas for new features or controls, please call us or write:

Corporate Headquarters

ComponentOne LLC
201 South Highland Avenue
3rd Floor
Pittsburgh, PA 15206 • USA
412.681.4343
412.681.4384 (Fax)

http://www.componentone.com

ComponentOne Doc-To-Help

This documentation was produced using ComponentOne Doc-To-Help® Enterprise.

Key Features

Size matters in Silverlight. In one click you can reduce the size of your Silverlight apps up to 70% and secure your code with obfuscation. Add **ComponentOne XapOptimizer** to your dev cycle with build automations, backups, and limitless options. With **XapOptimizer**, you can:

- **Customize the Optimization Output**

  See what's going to be removed from the assembly before it is actually removed. You can also manually "Pin" some elements to keep them after the optimization process.

- **Create Backups**

  XapOptimizer has the ability to automatically create a back up version of the file to be optimized.
- **Obfuscation**

  Choose to enable or disable obfuscation and choose the specific elements to obfuscate.
- **Sign Assemblies**

  Choose whether to sign elements in the assembly and which specific elements to sign.
- **Save Projects**

  XapOptimizer allows you to save projects allowing you to reopen the project at a later date or easily try different options.
- **Integrate with Your Build Process**

  Integrate **XapOptimizer** as part of the build process in Microsoft Visual Studio to automatically optimize your XAP files.
- **Use with Any Third Party Controls**

  Use **XapOptimizer** with any Silverlight assembly including third party controls; is not restricted to ComponentOne controls.

About ComponentOne XapOptimizer

ComponentOne XapOptimizer is a utility that optimizes Silverlight applications by making XAP files smaller and more difficult to reverse engineer.

Reducing application size is always a good thing because small applications load faster and are easier to distribute and install. This is especially important in Internet scenarios, where applications are deployed as part of Web pages and are constantly downloaded and updated.

Preventing reverse engineering is also a common concern, especially for .NET applications which are easy to disassemble using popular tools such as the Red Gate .NET Reflector.

ComponentOne XapOptimizer accomplishes these optimizations using two techniques:

- **Pruning**: Pruning (also known as dead-code elimination) consists of analyzing the application and removing classes and resources that are not used. Dead code is common in applications that use libraries. Libraries typically include many controls and classes of which each client application only uses small subsets.
  For example, if your application uses a **C1HyperPanel**, then it needs a reference to the **C1.Silverlight.dll** assembly, which contains the **C1HyperPanel** as well as many other controls your application does not use or need. In this case, **XapOptimizer** will create a new version of the **C1.Silverlight.dll** assembly that contains only the **C1HyperPanel** class and its dependencies.

  ComponentOne XapOptimizer will also remove unused resources (Styles, Templates, Brushes, and so on) defined in resource dictionaries in XAML files
- **Obfuscation**: Obfuscation consists of changing the names of classes and fields in order to deter reverse engineering. This process also reduces the size of assemblies by using obfuscated names that are shorter than the original names.

For example, if your application has a public class called **PasswordManager**, the **XapOptimizer** obfuscator will rename that class to something more like **x**. This makes the new assembly smaller and harder to understand for anyone who opens it in a disassembler.

About XAP Files

To understand how **ComponentOne XapOptimizer** works, consider the structure of a XAP file. A XAP file is a Silverlight application file. It's basically a file compressed in the ZIP format that includes all of the files needed by your Silverlight application. To see exactly what's inside a XAP file, you can re-name the file with a .zip extension and expand the file.

For example, if you examine the contents of a XAP file you'll see that it might contain a collection of files similar to the following:

- AppManifest.xaml
- ServiceReferences.ClientConfig
- StockPortfolio.dll
- System.Windows.Controls.dll
- C1.Silverlight.Chart.dll
- C1.Silverlight.Data.dll
- C1.Silverlight.DataGrid.dll
- C1.Silverlight.dll
- C1.Silverlight.Extended.dll

The XAP file always contains an application manifest file and copies of the assemblies and resources referenced by the application:

- **An application manifest file (AppManifest.xaml)**

  The application manifest file is always present and includes deployment information for your application. For example, it will list the application's entry point and a list of the files that comprise the application.

- **A YourProject.dll assembly**

  This file is the compiled version of your Silverlight application. It should have the same file name as your project.

- **Any additional assemblies need for your project**

  The XAP file will include additional assemblies needed for your project, if any. So, for example, if you're using the ComponentOne **C1RadialGauge** gauge control in your project, the XAP file will include the C1.Silverlight.Gauges.dll assembly.

- **External resources**

  If you embedded images or other external resources in your Silverlight application, they may also be included in the XAP file.

While the XAP file is initially compressed to minimize deployment size, it can include unnecessary elements that can be pruned and it can typically be further compressed – that's where **ComponentOne XapOptimizer** can help. **XapOptimizer** further reduces the size of your Silverlight application to reduce load time.

How XapOptimizer Works

ComponentOne XapOptimizer works by extracting the individual files from the zipped archive, then using reflection to build a call tree. The call tree starts with the application's entry point and expands to determine all the classes that are used by the application. At the end of this process, any classes not used by the application are removed from the assemblies, which are then obfuscated and repackaged into a new XAP file.

ComponentOne XapOptimizer works by extracting the original XAP file, then inspecting MSIL instructions and XAML content to build a dependency tree. The dependency tree starts with the application's entry point and expands to determine all the classes and XAML resources that are used by the application. At the end of this process, any classes and XAML resources not used by the application are removed from the assemblies, which are then obfuscated, resigned and repackaged into a new XAP file.

This brief description skips over some important details. Most important is that the dependency analysis process performed by **XapOptimizer** cannot resolve dependencies when indirect instantiation mechanisms are used (using Reflection for example). For more information, see Classes Used Through Reflection.

Command Line Arguments

When using **ComponentOne XapOptimizer** from the command line, you can specify several command line arguments. These include:

- **/p**: Specifies the project name.
- **/i**: Specifies the location of the input XAP file.
- **/o**: Specifies the location of the output XAP file.
- **/cmd**: Runs **ComponentOne XapOptimizer** in command mode (not using the user interface).

Note that generally there are two exclusive modes: specifying just the input/output file locations or the **ComponentOne XapOptimizer** project. So, for example, the following arguments are valid:

- XapOptimizer.exe /i: input.xap /o: output.xap
- XapOptimizer.exe /p:project.xoproj

See Adding XapOptimizer to the Build Process for an example.

Classes Used Through Reflection

Instead of creating objects using the **new** operator, some classes may use reflection instead, as shown below:

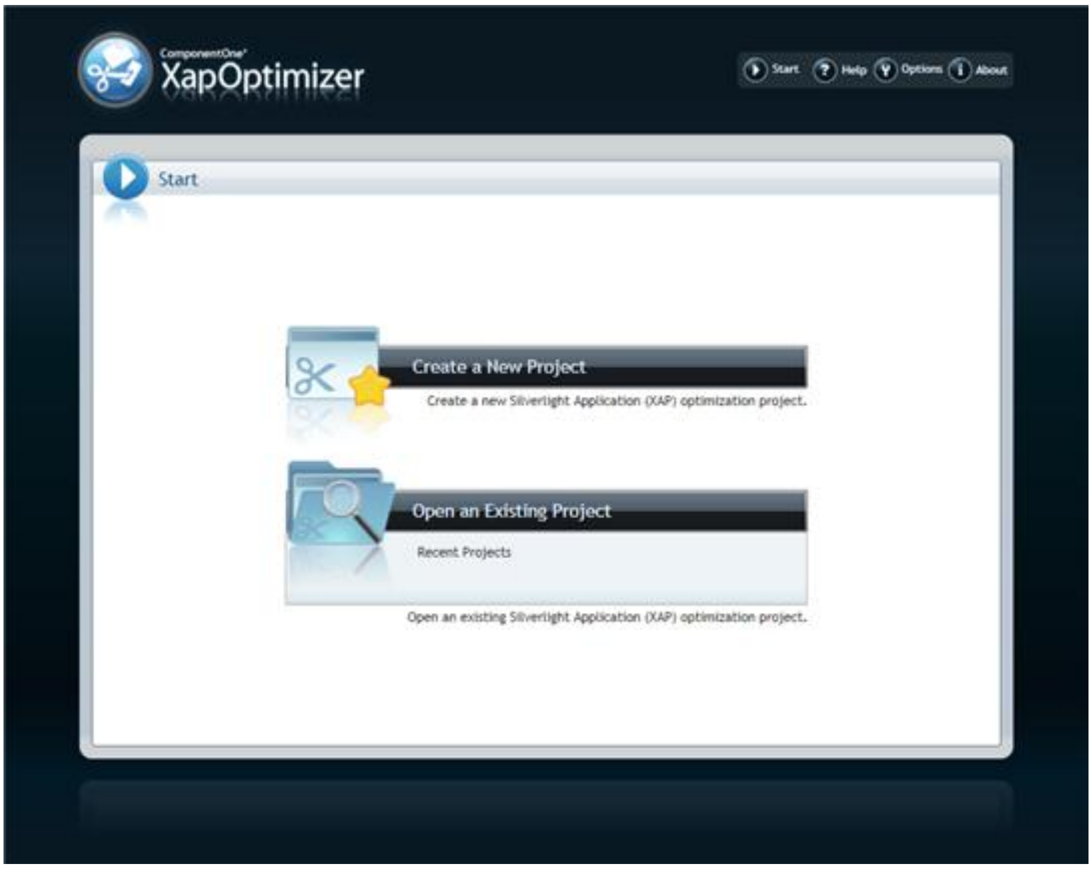
```
// Create the control as usual
// var ctl = new SilverlightControl1();

// Create the control using reflection
var asm = System.Reflection.Assembly.GetExecutingAssembly();
var type = asm.GetType("XapOptimizerTest.SilverlightControl1");
var ctor = type.GetConstructor(System.Type.EmptyTypes);
Control ctl = ctor.Invoke(null) as Control;
```

In this example, **XapOptimizer** will not be able to determine that the **SilverlightControl1** class is used by the project and will prune it from the application, causing it to fail.

In cases such as this, you need to tell the **XapOptimizer** not to prune or obfuscate the **SilverlightControl1** class. This is done by adding options to the **XapOptimizer** project, see Working with XapOptimizer for more information.

Getting Started with ComponentOne XapOptimizer

When you initially open **ComponentOne XapOptimizer**, the **Start** screen should appear:

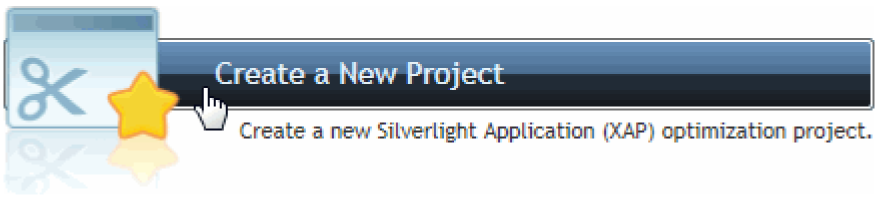

ComponentOne XapOptimizer
Start
Help
Options
About
Start
Create a New Project
Create a new Silverlight Application (XAP) optimization project.
Open an Existing Project
Recent Projects
Open an existing Silverlight Application (XAP) optimization project.

The following topics describe how to create, open, and save a **XapOptimzer** project, as well as how to customize the **ComponentOne XapOptimizer** application's settings.

Creating a New XapOptimizer Project

Creating a new **ComponentOne XapOptimizer** project is simple – complete the following steps to create a new project:

1. Launch the **XapOptimizer** application.
2. If the **Start** screen is not open, select the **Start** button from the navigation bar in the top-right corner of the application.
3. On the **Start** screen, click the **Create a New Project** option.


Create a New Project
Create a new Silverlight Application (XAP) optimization project.

The **Open** dialog box will appear.

4. In the **Open** dialog box, browse to where the XAP file you plan to optimize is located, select the XAP file, and click **Open**.

   If your Silverlight solution contains multiple projects, make sure you build the solution first, then select the XAP file in the **ClientBin** folder of the server project. This will make it easier for you to test and deploy the optimized XAP file. For example, if your application is called "MyApp", then you should choose the XAP file located at: **MyApp\MayAppWeb\ClientBin\MyApp.xap**.

   Once you've opened a file a progress bar will appear indicating that the file is being analyzed:

   Once the file has been analyzed, the Files tab will appear. At this point, you can save your project (see Saving a XapOptimizer Project), optimize the file by selecting the **Optimize** button (see Optimizing a File), or you can prune, obfuscate, or sign assemblies before optimizing (see Working with XapOptimizer).

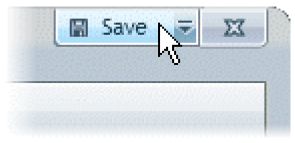
Saving a XapOptimizer Project

You can save your **ComponentOne XapOptimizer** project if you plan on returning to your **XapOptimizer** project later, re-optimizing your XAP file, or if you want to save your settings to optimize another XAP file. Note that **ComponentOne XapOptimizer** saves project files with a .xoproj extension.

Saving the Project

To save your project, complete the following steps:

1. Click the **Save** button in the top right-hand corner of any project screen:



   If you have previously saved the file it will be saved in the same location. If you have not previously saved the file the **Save** dialog box will appear.

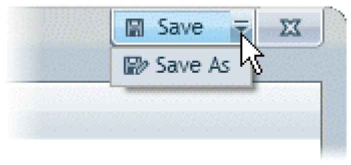
2. If the **Save** dialog box appears, browse to a location to save the file, enter a name in the **File name** text box, and click the **Save** button.

   The file will be saved to the location indicated.

Saving the Project to a New Location

To save your project to a new location or with a different file name, complete the following steps:

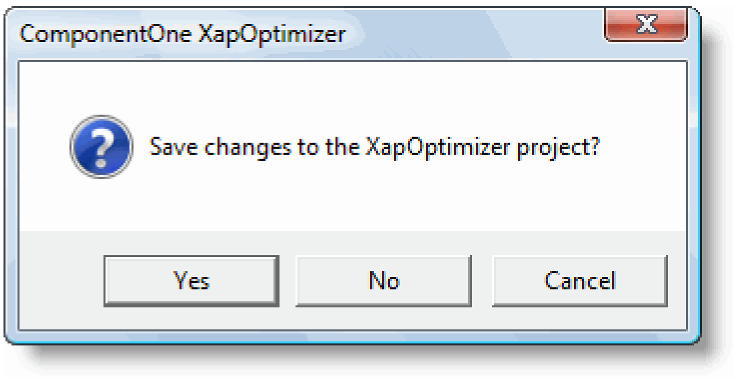
1. Click the drop-down arrow on the **Save** button in the top right-hand corner of any project screen:



2. Click the **Save As** option.
3. In the **Save As** dialog box, browse to a location to save the file, enter a name in the **File name** text box, and click the **Save** button.

   The file will be saved to the location indicated.

Note that **ComponentOne XapOptimizer** will also prompt you to save the project when closing the project:

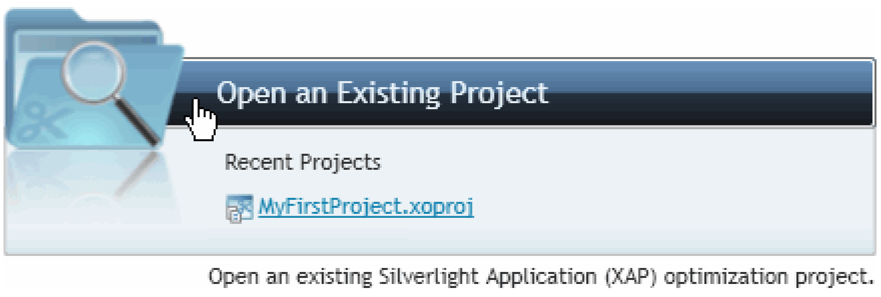

ComponentOne XapOptimizer
Save changes to the XapOptimizer project?
Yes
No
Cancel

You can click **Yes** in this dialog box to save your project.

Opening an Existing XapOptimizer Project

You can open an existing project file if you have previously saved your **ComponentOne XapOptimizer** project (see Saving a XapOptimizer Project for more information). To open an existing **XapOptimizer** file, complete the following steps:

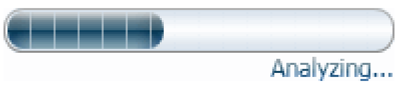
1. Launch the **XapOptimizer** application.
2. If the **Start** screen is not open, select the **Start** button from the navigation bar in the top-right corner of the application.
3. On the **Start** screen, click the **Open an Existing Project** option.


Open an Existing Project
Recent Projects
MyFirstProject.xoproj
Open an existing Silverlight Application (XAP) optimization project.

   The **Open** dialog box will appear. Note that you can also select your project in the **Recent Projects** list on the **Start** screen if it is listed.

4. In the **Open** dialog box, browse to where the XAP file you plan to optimize is located, select the XAP file, and click **Open**.

   A progress bar will appear indicating that the file is being analyzed:



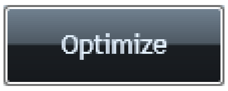
   Once the file has been analyzed, the **Files** tab will appear.

Optimizing a File

Once you have created a new XapOptimizer project or opened an existing project you can optimize your XAP file. Note that you can customize options before optimization, see Working with XapOptimizer for more information.

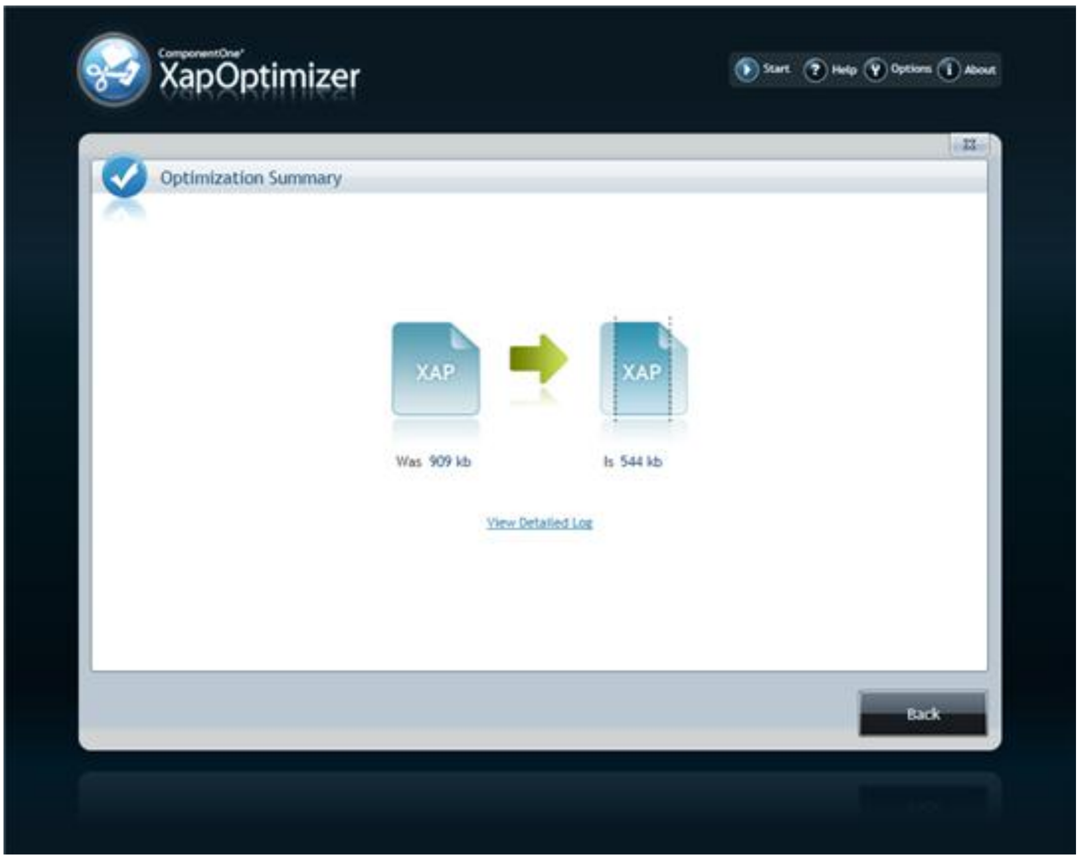
To optimize a XAP file, complete the following steps:

1. Click the **Optimize** button located in the lower right corner on any project screen:

   A progress bar will appear, indicating that the file is being optimized:

   When the file has finished optimizing, an **Optimization Summary** screen will appear indicating the size of the initial file compared to the size of the optimized file:

2. Click **View Detailed Log** to view more information about the optimization process including a summary of pruned, obfuscated, and signed elements.
3. Click the **Back** or **Close** ("X") button on this screen to return to the project.

   Your XAP file will be optimized. If you choose, you can change settings and re-optimize your file. For more information see Working with XapOptimizer.

Testing an Optimized XAP File

You don't have to deploy your optimized XAP file to test it. One way you can test your optimized XAP files is by adding it to the build process and then running the project from Visual Studio. For more information, see Adding XapOptimizer to the Build Process.

You could also embed your XAP file in an HTML page that you can open and then refresh after optimization (see Adding Silverlight to a Web Page by Using HTML or JavaScript at MSDN for more information: http://msdn.microsoft.com/en-us/library/cc838217(VS.95).aspx) or reopen the TestPage.html file that Visual Studio typically creates when debugging or building a Silverlight application and that should have your XAP file already embedded (generally located in the ProjectName\ProjectName\Bin\Debug directory).

Adding XapOptimizer to the Build Process

Once you've created a **XapOptimizer** project, you can add it to your build process in Visual Studio. This is easy to do using a post-build event.

Complete the following steps:

1. Open the Silverlight project in Visual Studio.
2. Select the **Release** configuration (you probably will not want to optimize the project after each debug build).
3. Right-click the main project node in the Solution Explorer and select **Properties**.
4. Click the **Build Events** tab on the left side of the Properties window.
5. Enter the following command in the **Post-build event command line**:
   ```
   $Program Files\ComponentOne\XapOptimizer\XapOptimizer.exe /cmd
   /p:$(ProjectDir)$(ProjectName).xoproj
   ```

   Note that the command line statement in step 5 assumes that the command line version of **ComponentOne XapOptimizer** is located in the **Program Files\ComponentOne\XapOptimizer\bin** folder. You may change this line if the application is installed in a different directory. See Command Line Arguments for more information about the arguments used in the above statement.

   Once this step is finished, the XAP file will be optimized automatically every time you perform a release build. You can easily test the optimization by running the release configuration from Visual Studio.

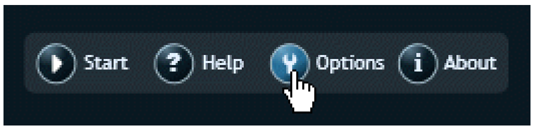
If your application does not use reflection, it will probably work right away without any customizations to the project file. If you do use reflection, you may need to customize the **XapOptimizer** project. See Working with XapOptimizer for more information.

Customizing XapOptimizer Options

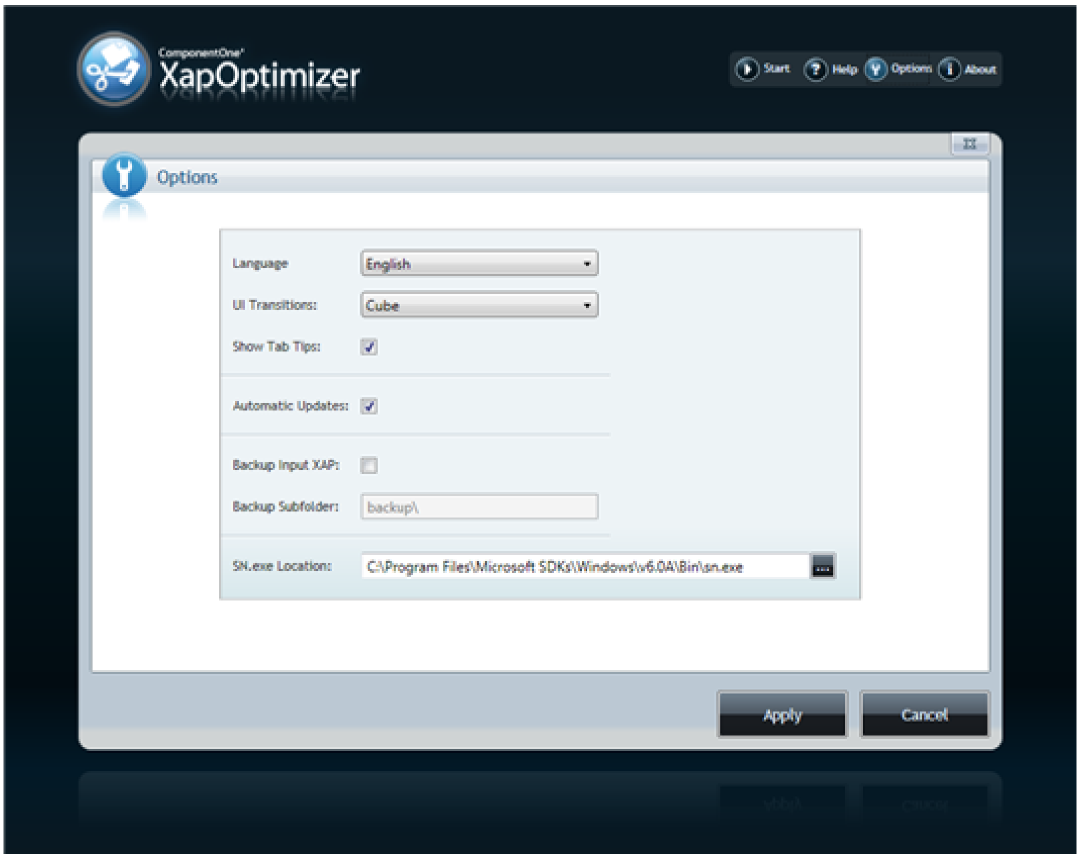
ComponentOne XapOptimizer includes an **Options** screen allowing you to customize the **XapOptimizer** application's settings. Note that these are global options and are not project specific.

Accessing the Option Screen

To access the **Options** screen, select the **Options** button located in the upper right-hand corner of the application:



The **Options** screen appears similar to the following image:



Language: English
UI Transitions: Cube
Show Tab Tips: ✓
Automatic Updates: ✓
Backup Input XAP: ☐
Backup Subfolder: backup\
SN.exe Location: C:\Program Files\Microsoft SDKs\Windows\v6.0A\Bin\sn.exe

The **Options** screen includes the following items:

- **Language**

  Select the default language for the **ComponentOne XapOptimizer** application's user interface. By default **English** is selected.

- **UI Transitions**

  Choose the transition effect used in the **XapOptimizer** application when showing and hiding the **Options** and **About** screens. By default the **Cube** effect is selected.

- **Show Tab Tips**

  Check this check box to display ToolTips in the **XapOptimizer** application when you hover the mouse over tabs in the main window.

- **Automatic Updates**

  Use this check box to determine if the **ComponentOne XapOptimizer** application should automatically connect to the ComponentOne Web site on startup to look for updates. By default this item is checked and the application is automatically updated. Uncheck this item to disable automatic updates.

- **Backup Input XAP**

  Use this check box to determine if your original unoptimized XAP file is automatically backed up when the file is optimized. By default this option is not checked, and the file is not backed up.

- **Backup Subfolder**

  Select a subfolder location to back up your XAP file to. This option is only customizable when the **Backup Input XAP** check box is checked. By default, the folder name is "backup". The folder location is relative to the location of the original XAP file.

- **SN.exe location**

  This option allows you to specify the location of the SN.exe utility (Strong Name Tool) that ships with Visual Studio and is used to strong name assemblies. This utility is required when strong naming the optimized assemblies.

- **Apply Button**

  Once you have made changes to the **XapOptimizer**'s options, select the **Apply** button to apply those changes.

- **Cancel Button**

  Click the **Cancel** button to cancel any changes you have made to the **XapOptimizer**'s options.

Working with XapOptimizer

The following sections describe various options you can set when using **ComponentOne XapOptimizer** to reduce the file size of your XAP files. By default, **ComponentOne XapOptimizer** will try to prune and obfuscate as much of your application as possible and it will not strong-name the assemblies it generates.

In some cases, you may have to customize your **XapOptimizer** projects to prevent pruning and obfuscation of classes that are used in reflection scenarios. You may also want to strong-name the output assemblies. This is done using three tabs in the **XapOptimizer** application: Pruning, Obfuscation, and Signing.

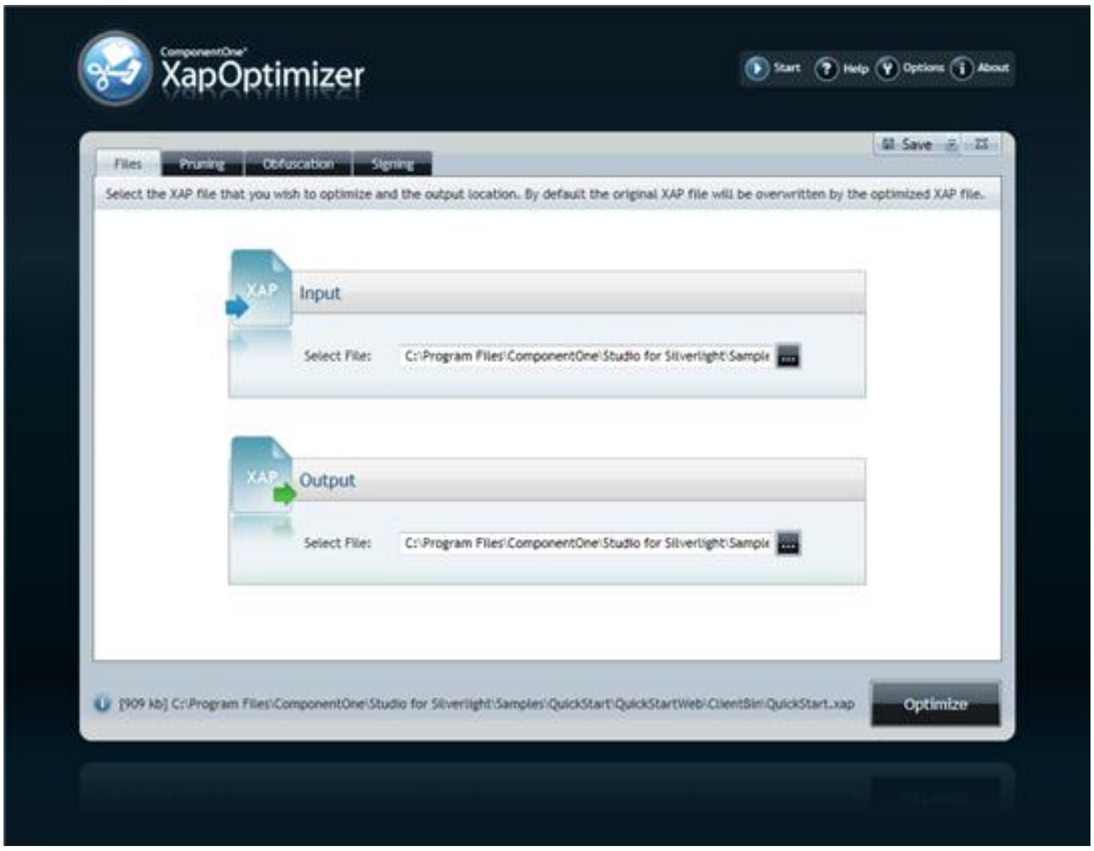
Once you have completed customizing the project, you can test the customizations by rebuilding the application and running it from Visual Studio (provided you have added the post-build event step described in the Adding XapOptimizer to the Build Process topic).

Note that **XapOptimizer** project files are XML files, and you can edit them manually or programmatically if you prefer to do so.

Files Tab

The **Files** tab is the first screen to open after creating a new project or opening an existing **XapOptimizer** project. In the **Files** tab you can select the XAP file that you wish to optimize and select the output location for the optimized file. By default the output file location will be the same as the input file location and the original file will be overwritten by the optimized file.

The **Files** tab looks similar to the following:



Select the XAP file that you wish to optimize and the output location. By default the original XAP file will be overwritten by the optimized XAP file.

Select File: C:\Program Files\ComponentOne\Studio for Silverlight\Sample

Select File: C:\Program Files\ComponentOne\Studio for Silverlight\Sample

[909 kb] C:\Program Files\ComponentOne\Studio for Silverlight\Samples\QuickStart\QuickStartWeb\ClientBin\QuickStart.xap

The **Files** tab includes two sections:

- **XAP Input**

  Select a file here to optimize. The file that you initially selected when you created a new project will be displayed here by default.

- **XAP Output**

  Choose a location for your optimized file to be saved. By default, the file that you initially selected when you created a new project will be displayed here.

Selecting an Input File

To select a different input file to optimize, complete the following steps:

1. Click the ellipsis button in the **XAP Input** section to open the **Open** dialog box.
2. In the **Open** dialog box, browse to the location of your file and select **Open** to close the dialog box and select your file.

   The file you selected will appear in the **XAP Input** text box.

Selecting an Output File

To change the file output location or the name of the outputted file, complete the following steps:

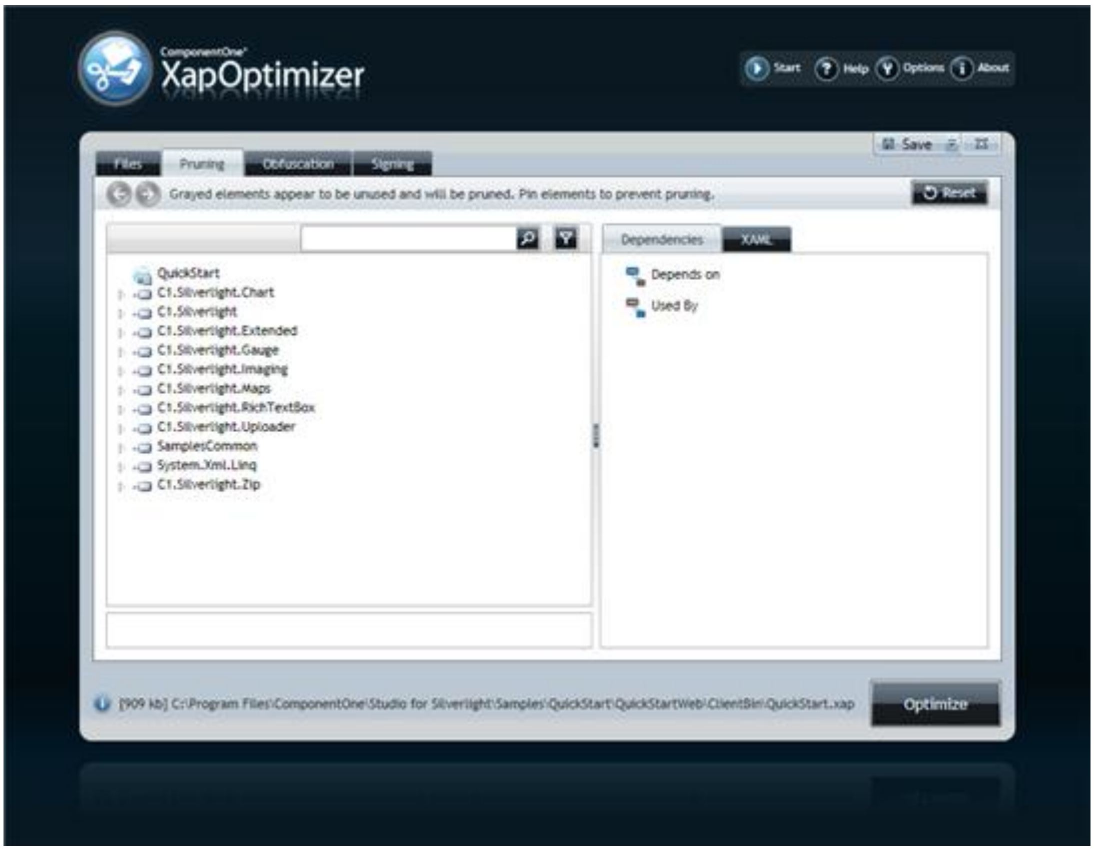
1. Click the ellipsis button in the **XAP Output** section to open the **Open** dialog box.
2. In the **Open** dialog box, browse to the location where you plan to save your outputted optimized file and select **Open** to close the dialog box and select your file.

   The file you selected will appear in the **XAP Output** text box.

Pruning Tab

On the **Pruning** tab, you can choose elements to be included in your optimized file. Pinned elements will be included and unpinned elements will not be included in the final optimized file.

The **Pruning** tab appears similar to the following image:



Grayed elements appear to be unused and will be pruned. Pin elements to prevent pruning.

[909 kb] C:\Program Files\ComponentOne\Studio for Silverlight\Samples\QuickStart\QuickStartWeb\ClientBin\QuickStart.xap

The **Pruning** tab includes the following elements:

- **Navigation Arrows**

  Click the left and right navigation arrows to move through the tree list below. You can also navigate through the list by using they keyboard arrow keys or by expanding nodes and selecting items with the mouse.

- **Reset Button**

  The **Reset** button returns all of the items in the list to the default settings. If you have pinned or unpinned elements manually, you will lose your changed settings. If you want to restore the default pruning settings, click the **Reset** button on the top-right corner of the screen. This will unpin all elements so you can restart your customization.

- **Search Bar**

  Type text in the search bar to locate items that match your term. You can enter one letter or a partial term to have all elements including that letter or term appear.

- **Filter Button**

  Click the **Filter** button to filter the tree list by element. You can filter the tree to display only elements that are pinned (included in the optimized file), items that will be pruned, will not be pruned, and cannot be pruned (and must stay in the optimized file).

- **Element Tree**

  The element tree allows you to view the elements that will be included in the optimized file and to pin or unpin elements to change the default settings. The element tree on the left shows all the assemblies and classes in the project. Grayed out classes are ones that don't seem to be used in the project and will be pruned by **XapOptimizer**. If you know that some classes will be invoked via reflection, click the pin icon next to the class name. This will pin the class, preventing it from being pruned. Any classes that depend on the pinned class will also be excluded from the pruning process.

- **Description Window**

The Description window appears below the element tree window and displays a description of the selected element.

- **Dependencies Window**

  The Dependencies tab is selected and its window visible by default. This pane shows a dependency analysis for the currently selected class. This consists of two lists, one showing the classes that **Depend on** the selected class and one showing the classes that are **Used by** the selected class. You can browse through the dependencies in the application by double-clicking items under either branch. If you want to retrace your steps through the dependency tree, use the arrow buttons at the top left corner of the screen.

- **XAML Window**

  The XAML window lets you see the actual XAML markup for the selected element. This will give you a better idea of how that element is used and whether it can be pruned or not.

- **Splitter Bar**

  A splitter bar appears between the element tree and Dependencies/XAML windows. Drag the splitter bar to resize the windows if needed.

Once you are done selecting classes that should not be pruned, you can click the **Save** button on the top right of the screen to save your options. See Saving a XapOptimizer Project for more information.

Pinned and Unpinned Elements

ComponentOne XapOptimizer performs a dependency analysis and automatically detects elements that can be pruned from the application. This automatic analysis works in most cases, but sometimes you may want to exclude additional elements from the pruning process. To do this, you should "pin" the elements that you want to keep in the application.

Pinned and unpinned elements in the element tree appear in differing fonts and with icons to distinguish them. Elements appear similar to items in the following table:

Image	Type	Description
ColorPalette	Pinned.	Pinned elements will not be pruned because they are marked to stay in an assembly. When you pin an element, all elements used by that element also stay in the assembly. If you pin an element with subelements, all subelements are pinned. You can expand the item and pin or unpin subelements manually, as well.
{ } C1.Silverlight	Partially pinned.	Partially pinned elements are elements with subelements, some of which have been pinned, and some which have not been pinned.
C1PropertyGrid	Will be pruned.	Grayed out elements will be pruned because **XapOptimizer** could not find any dependency to those elements. You can avoid this by pinning items manually.
C1Accordion	Will not be pruned.	Items that appear in a normal font will not be pruned and will stay in the final assembly after optimization.
QuickStart	Cannot be pruned.	Some items cannot be pruned at all, due to limitations (for example, if they are part of the Silverlight core). Items that cannot be pruned will appear in a normal font. The top level project assembly pictured to the left, for example, cannot be pruned.

Pinning Elements

To pin an element, complete the following steps:

1. Expand nodes in the element tree and locate the item you wish to pin.

   Note that only grayed out elements (items that would normally be pruned) can be pinned.

2. Click the **Pin** icon next to the item you wish to include. The item will be pinned and will no longer appear grayed out. If you pin an element with subelements, all subelements will be pinned.

Note that you can also pin elements by pressing the SPACEBAR key when navigating the tree with the keyboard.

Unpinning Elements

To unpin a pinned element, complete the following steps:

1. Expand nodes in the element tree and locate the item you wish to unpin.

   Note that only pinned elements can be unpinned.

2. Click the **Pin** icon next to the item you wish to no longer include. The item will be unpinned and will appear grayed out. If you unpin an element with subelements, all subelements will be unpinned.

   Note that you can also unpin elements by pressing the SPACEBAR key when navigating the tree with the keyboard.

Resetting Pinned Items

If you need to, you can reset pinned and unpinned items. Click the **Reset** button to return to the default settings. Note that you will lose any changes that you have made.

Note that all elements defined in the main application assembly are kept in the final application. The pruning process is only applied to auxiliary assemblies included in the application, which are often libraries.

Navigating the Element Tree

You can navigate through the element tree by using the navigation buttons or selecting items in the tree. You can also navigate items using the XAML or Dependencies window.

Using the Navigation Buttons

To navigate through the element tree using the navigation buttons, complete the following steps:

1. Click an element in the element tree to activate navigation buttons, if the navigation buttons are not active.
2. Click the **Previous** arrow button to return to the previous item or the **Next** arrow button to move to the next item.

   Note that clicking the arrows buttons will move to each node and then items in that node before moving to the next node.

Using the Element Tree

You can use the mouse to navigate the element tree. In the element tree, click an item to select it. You can expand and contract nodes by clicking the "+" and "-" icons next to a node.

Using the Dependencies Window

You can navigate through the element tree by using the Dependencies window. Complete the following steps:

1. Select the **Dependencies** tab if the Dependencies window is not visible.
2. Using the mouse, expand nodes in the element tree and select an item.

   The Dependencies window will display items that use and are used by the selected element.

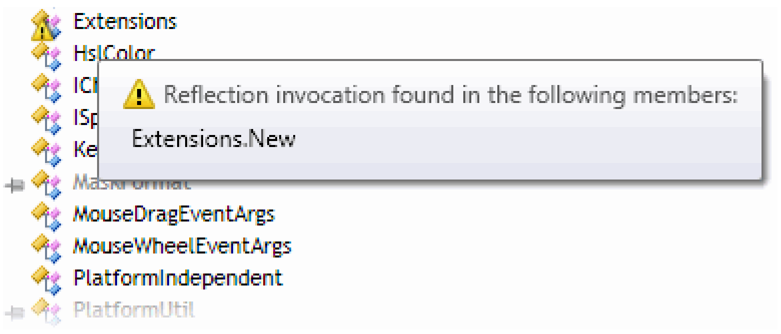
3. In the Dependencies window, double-click an item to move to it in the element tree.

Reflection Warnings

ComponentOne XapOptimizer cannot determine the actual Type dependency in scenarios where the Reflection API is used to instantiate or invoke an object (see Classes Used Through Reflection for more information). In those cases, the Type may be pruned, causing the application to fail.

XapOptimizer will detect the usage of the Reflection API and it will notify you by displaying a warning icon on the classes (or Silverlight application) where Reflection is being used. **XapOptimizer** will also display a ToolTip on specific members where the API is invoked.

In the image below the Reflection API is being invoked inside the **New** method of the **Extensions** class:


Extensions
HslColor
ICI
ISp
Ke
MaskFormat
MouseDragEventArgs
MouseWheelEventArgs
PlatformIndependent
PlatformUtil

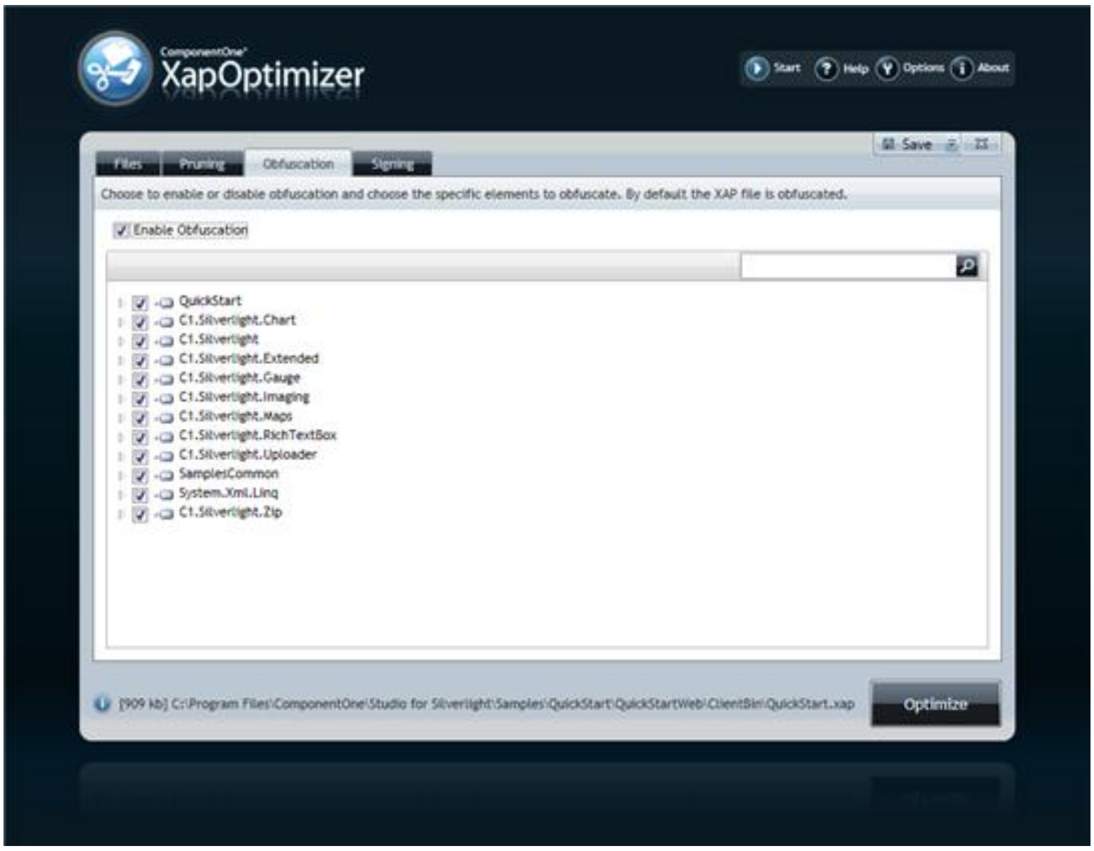
Reflection invocation found in the following members:
Extensions.New

If you receive a Reflection warning, you can either replace the Reflection invocation in code (if possible) or pin the class that's being indirectly instantiated or invoked so it will not be pruned (see Pinned and Unpinned Elements for more information).

Obfuscation Tab

By default, **ComponentOne XapOptimizer** obfuscates your XAP file as it optimizes it. The **Obfuscation** tab lets you choose whether obfuscation is enabled or disabled and, if enabled, what elements should be obfuscated.

The **Obfuscation** tab appears similar to the following image:



Choose to enable or disable obfuscation and choose the specific elements to obfuscate. By default the XAP file is obfuscated.

[✓] Enable Obfuscation

- [✓] QuickStart
- [✓] C1.Silverlight.Chart
- [✓] C1.Silverlight
- [✓] C1.Silverlight.Extended
- [✓] C1.Silverlight.Gauge
- [✓] C1.Silverlight.Imaging
- [✓] C1.Silverlight.Maps
- [✓] C1.Silverlight.RichTextBox
- [✓] C1.Silverlight.Uploader
- [✓] SamplesCommon
- [✓] System.Xml.Linq
- [✓] C1.Silverlight.Zip

[909 kb] C:\Program Files\ComponentOne\Studio for Silverlight\Samples\QuickStart\QuickStartWeb\ClientBin\QuickStart.xap

The **Obfuscation** tab includes several elements, such as:

- **Enable Obfuscation Check Box**

  The **Enable Obfuscation** check box lets you determine if code in the XAP file should be obfuscated or not. By default this check box is checked and the file is obfuscated.

- **Search Bar**

Type text in the search bar to locate items that match your term. You can enter one letter or a partial term to have all elements including that letter or term appear.

- **Element Tree**

  The element tree allows you to view the elements that are included in the optimized file, and to select whether or not elements are obfuscated. By default all elements are selected and obfuscated.

Once you are done configuring obfuscation, you can click the **Save** button on the top right of the screen to save your options. See Saving a XapOptimizer Project for more information.

Disable Obfuscation

To disable obfuscation, uncheck the **Enable Obfuscation** check box.

Disable Obfuscation for Specific Elements

To disable obfuscation for specific elements, complete the following steps:

1. Expand nodes in the element tree and locate the item you wish to disable obfuscation for.
2. Uncheck the check box next to the item you wish to keep unobfuscated. When the file is optimized, the element will not be obfuscated.

Enable Obfuscation

To enable obfuscation when disabled, check the **Enable Obfuscation** check box.

Enable Obfuscation for Specific Elements

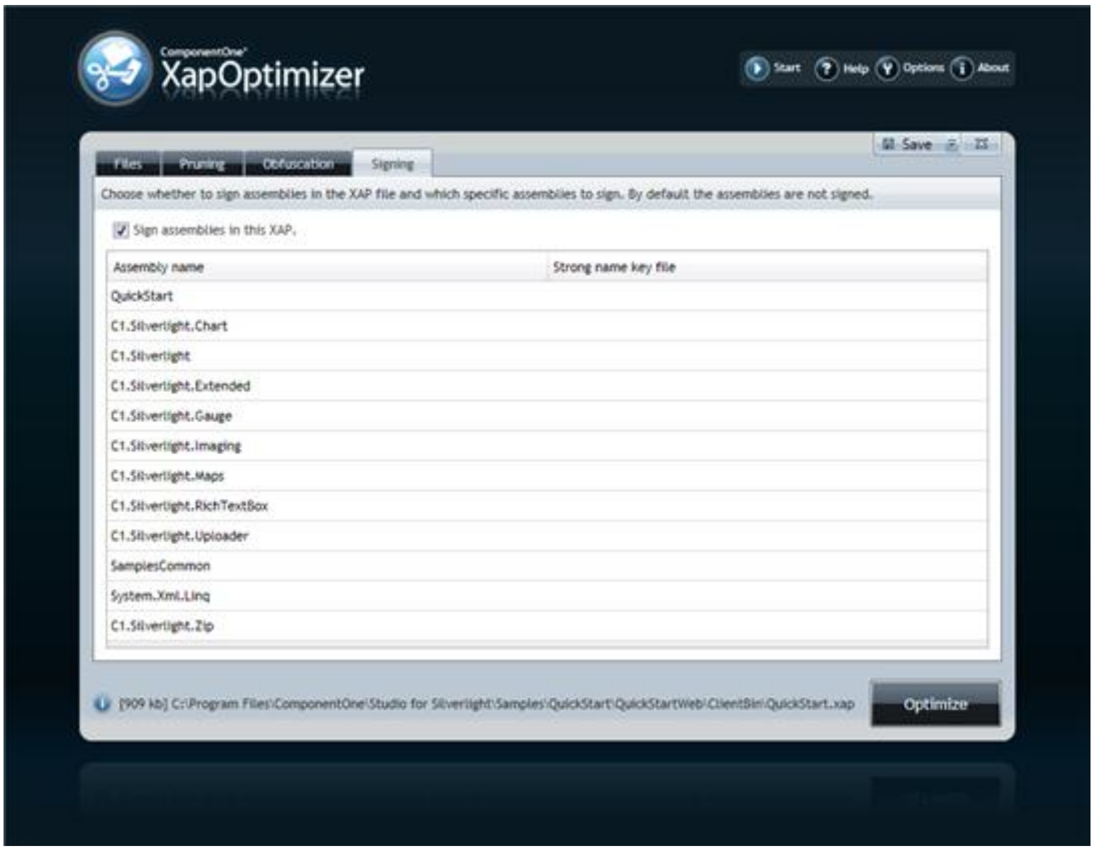
To enable obfuscation for specific disabled elements, complete the following steps:

1. Expand nodes in the element tree and locate the item you wish to enable obfuscation for.
2. Check the check box next to the item you wish to obfuscate. When the file is optimized, the element will now be obfuscated.

Signing Tab

On the **Signing** tab you can specify strong-names for the assemblies in the application. You can choose whether to sign elements in the assembly and which specific elements to sign. By default all elements are unsigned. If you want your assemblies to be strong-named, you have to check them on the list and then specify the "snk" file that will be used to sign the assemblies.

The **Signing** tab appears similar to the following:



Choose whether to sign assemblies in the XAP file and which specific assemblies to sign. By default the assemblies are not signed.

[✓] Sign assemblies in this XAP.

Assembly name	Strong name key file
QuickStart	
C1.Silverlight.Chart	
C1.Silverlight	
C1.Silverlight.Extended	
C1.Silverlight.Gauge	
C1.Silverlight.Imaging	
C1.Silverlight.Maps	
C1.Silverlight.RichTextBox	
C1.Silverlight.Uploader	
SamplesCommon	
System.Xml.Linq	
C1.Silverlight.Zip	

[909 kb] C:\Program Files\ComponentOne\Studio for Silverlight\Samples\QuickStart\QuickStartWeb\ClientBin\QuickStart.xap

The **Signing** tab includes elements, such as:

- **Sign assemblies in this XAP check box**

  The **Sign assemblies in this XAP** check box lets you determine if assemblies in the XAP file should be signed or not. By default this check box is not checked and assemblies are not signed.

- **Assembly grid**

  The assembly grid lists each assembly included in the XAP file. You can interact with this grid only if the **Sign assemblies in this XAP** check box is checked. To specify the SNK file that will be used to sign the assemblies, click the drop-down arrow in the **Strong name key file** field for an assembly and create a new file or open an existing one.

To sign assemblies in the XAP file, you must enable signing assemblies and choose a strong name key file for signed assemblies.

Note: For assemblies to be signed, on **XapOptimizer**'s **Option** screen you must have chosen a location for the Strong Name application included with Visual Studio (SN.exe). For more information, see Customizing XapOptimizer Options.

Signing Assemblies

Complete the following steps to sign assemblies in your XAP file:

1. Check the **Sign assemblies in this XAP** check box to enable signing assemblies in the XAP file (by default assemblies in your XAP file are not signed).
2. Click the drop-down arrow in the **Strong name key file** field for each assembly that you want to sign and:
   - Choose **New** to create a new strong name key file. Browse to a location and enter a name in the **Save As** dialog box and click **Save** to create your file.

     OR
   - Choose **Browse** to open existing new strong name key file. In the **Open** dialog box browse to your file's location, select your file, and click **Open** to open your file.

   The strong name key file name and location will appear next to the assembly name.

Once you are done configuring signing, you can click the **Save** button on the top right of the screen to save your options. See Saving a XapOptimizer Project for more information. Note that signatures in third-party signed assemblies will be removed when using **XapOptimizer**. For details see the Limitations and Troubleshooting topic.

> **Note:** The .NET Framework security policy forbids singed assemblies to load unsigned ones. So, when you manually choose to sign an assembly in **XapOptimizer**, you must ensure that all the assemblies it references are going to be signed as well. If you do not, a **System.IO.FileLoadException** exception will be thrown with the following message: "A strongly-named assembly is required'.

XapOptimizer Top Tips

Occasionally you may run into issues with **ComponentOne XapOptimizer** or you may have questions about using the application. This section outlines some solutions and tips for using **ComponentOne XapOptimizer**.

The following list details some tips for using **ComponentOne XapOptimizer**:

- **Reflection:** If you are using the System.Reflection API in your code to dynamically create objects in the application, make sure to you "Pin" all the corresponding Types you may instantiate using this technique. For more information, see Reflection Warnings.
- **Navigation APIs:** Several third-party navigation libraries use System.Reflection API to dynamically instantiate the visual components to be displayed to the users. So, the third-party library may be using System.Reflection API even when you are not using it explicitly. If that's the case, make sure to "Pin" all the visual components (Pages and UserControls) you know that are being used in the application
- **Themes:** Themes are usually loaded dynamically at run time, so **XapOptimizer** may not find any dependencies to your Themes. Make sure to "Pin" any XAML resource that your application may be using for the Themes you are using in the application.
- **Obfuscation:** If the application is not working correctly, the Web browser will display an error message. Sometimes this error message can help you determine what's missing in the optimized assembly. Disabling obfuscation can be helpful when trying to interpret these messages. Also, it's recommended that you not obfuscate System assemblies (for example System.Xml.Linq).

Limitations and Troubleshooting

ComponentOne XapOptimizer does have some limitations. This section describes these limitations and offers some suggestions for dealing with them and other issues.

XapOptimizer does not offer assembly cache support.

ComponentOne XapOptimizer does not include support for assembly caching. One solution when working with **ComponentOne XapOptimizer** is to disable assembly caching from the Silverlight XAP file that you are building.

XapOptimizer removes signatures in third-party signed assemblies.

If you are using third-party signed assemblies in your application, the key will be removed by **ComponentOne XapOptimizer** when the XAP file is being optimized. To solve this issue, locate the XAP file after optimization and complete the following steps:

1. Select the XAP file and rename the file so that it has a .zip extension. So for example, Application.xap would become Application.zip.
2. Unzip the renamed XAP file to a folder and remove the modified third-party signed assemblies.
3. Add the original third-party signed assemblies (corresponding to those you removed) to the folder.
4. Zip the XAP up again with the original assemblies.
5. Rename the XAP file so that it again has a .xap file extension. So for example, Application.zip would become Application.xap.

The file should now be in the same state as before, except it will contain the original assembly for the third-party signed assemblies in your project.

XapOptimizer does not work as expected.

Sometimes you may run into issues or error messages and be unsure how to proceed. For example, in some cases **XapOptimizer** may be unable to determine the actual dependencies between classes or XAML resources. In these scenarios, you should pin elements to ensure that required elements and resources will remain in the optimized assembly. But it's not always easy to find the correct elements to be pinned so the application can continue working as expected.

If you run into issues, try completing the following steps:

1. Pin SDK libraries (such as, System.Xml.Linq).
2. Pin all the resources and try again.
3. Disable Obfuscation and try again.
4. Pin all the elements and try again. If this works, you can start unpinning elements to determine where the problem is.

If these steps do not work or you have any other questions, please contact support at support@componentone.com or submit a support incident at http://www.componentone.com/Support/.