ProvideX

Version 7.5

Installation and Configuration

sage software

Publication Release: V7

Installation and Configuration

The ProvideX Installation and Configuration guide describes the requirements and procedures for downloading, installing, and configuring the ProvideX base system and related products for use under Windows and UNIX/Linux platforms.

This document should be read by system administrators who are responsible for setting up and maintaining ProvideX as well as by application developers who wish to incorporate the ProvideX system configuration into the design of their proprietary applications. Rather than reproduce existing material, references to related documents are supplied where applicable.

ProvideX Base System, Add-Ons, and Bundles

ProvideX products can be installed and configured in many different ways, depending on the license and the platform; however, all ProvideX installations and activations begin with a ***Base System*** that includes:

- *ProvideX.* Interpreter and application development tools.
- *NOMADS.* Toolset for developing GUI-based applications.
- *COM Support, SSL Support, Report Writer* and *Views* runtime modules, etc.

Add-Ons

Once the ProvideX base system is installed, its functionality can be extended using a variety of application development/deployment tools — or ***Add-ons***. Some of these products are delivered within the base system installation but are only available for use under a specific *license* and *activation*. Most of the following software may be purchased separately, or as part of a comprehensive product bundle (explained in the next section):

- *WindX and JavX*. Thin-client interfaces for accessing GUI ProvideX applications.
- *Local and Client/Server ODBC*. Open DataBase Connectivity (ODBC) access to ProvideX databases.
- *RPC*. Remote Processing Capability for distributed processing of ProvideX applications across the Internet.
- *OCI*. Interface for accessing ORACLE databases directly.

- *Internet Toolkit*. Utilities for developing e-mail/Web-enabled applications.
- *Multiple Image Support*. Extended image support.
- *Smart Controls*. Auto-load capability for list boxes/grids.
- *Charting Control*. Control object feature for creating various chart illustrations in your applications.
- *Customizer*. Customize panel information dynamically without changing source.
- *Views*. End-user "viewing" of application data for simplified extraction and reporting.
- *OLE Server*. Interface allowing external applications to access ProvideX objects directly.
- *Web Server*. Interface for producing ProvideX-coded websites that allow browser access to ProvideX and ODBC data sources.
- *Application Server*. Software to establish client/server TCP-IP connections for ProvideX-based applications.
- *Report Writer*. Powerful interface for designing and generating reports.

Professional and eCommerce Bundles

A ***Bundle*** license incorporates several ProvideX add-on packages into one convenient activation that is more cost-effective than the separate licenses combined. The Professional bundle extends the base system with an integrated set of development and deployment tools. The eCommerce bundle augments the Professional suite with modules for enhanced Web functionality.

Note: Contact your dealer/distributor or visit the ProvideX website at www.pvx.com for more information on the above product options and licensing.

Licensing and Activation Keys

Every licensed copy of ProvideX requires a unique *serial number* and *activation key* that is generated specifically for that installation. The ProvideX download includes an activation utility that helps with the entry and recording of this information (see **Installation Instructions**, *p.7* and **Activation Procedures**, *p.10*).

The various licensing and activation modes are categorized as follows:

Demo Activation

A *demo* activation is invoked automatically with every installation of ProvideX.

This category of activation allows you to "test drive" a complete copy of the ProvideX eCommerce bundle for a period of 30 days. However, demo mode displays a "nag" message every 30 minutes, is limited to a single user, and allows no more than 5000 records per file. When the demo expires in 30 days, a *temporary* or *permanent* activation key must be obtained in order to fully activate ProvideX. Refer to **Sales** on the ProvideX website for more information.

If necessary, demo mode can be re-activated for an additional 30 days (see **Command Line Activation Parameters**, *p.15*.).

Trial Activation

A *trial* activation allows you to try specific add-on packages for a period of 25 days. Trial *keys* are issued upon request and will work under any current level of ProvideX installation. For more information, see **Primary and Secondary-Level Activations**, *p.6*

Temporary Activation

This activation is issued with every new license and it allows you to run a ProvideX product for a transitional period of up to 30 days.

When you purchase a ProvideX license, you receive a serial number and *temporary key* for a specific product and a specific number of users. The temporary activation is fully functional for 30 days or until a permanent key is issued. It is also useful for emergency re-installation of your ProvideX software in case of system failure.

Warning messages are displayed 7 days before the expiry date. When the date expires, you will not be able to access the software until you re-enter the temporary key (for another 30-day cycle), or obtain a permanent key (explained below).

Permanent Activation

This activation establishes a licensed copy of a ProvideX product on a specific machine. Depending on the license, this activation may or may not be issued with an expiry date.

Every permanent activation requires a new *permanent key*, which is derived from system information that is generated at the time of installation. All permanent activations are dependent on the following criteria to be operational:

- system ID
- machine class number
- product serial number
- specific number of users
- specific expiry date (or blank, if no expiry is issued).

This information also serves to restrict a specific activation to its original system configuration. Therefore, once a permanent activation is recorded, the software cannot be transferred to any other location unless the license specifically allows for this. If the activation file is moved or deleted, a new permanent key must be obtained to re-establish the activation at its new location.

To obtain a permanent key, record your system information at the time of installation, then contact your local ProvideX distributor. Refer to **Sales** on the ProvideX website for purchasing information.

Runtime Activation

On occasion, a *runtime* activation key may be issued under a special non-development licence that limits ProvideX to be run without the ability to list, edit, save, etc. This type of activation may or may not be issued with an expiry date.

Primary and Secondary-Level Activations

Every instance of ProvideX requires a base system or *primary-level* activation that enables use of the language itself, program editing, listing, saving, and access to console mode. This level of activation is the basis for all ProvideX products and applications. A primary-level activation must be established before additional ProvideX software can be activated.

Add-ons are either included in a bundle-type activation or activated separately. These products require package or *secondary-level* activations on top of the base system activation. The software for add-on packages is typically loaded along with the ProvideX installation; however, the ability to *use* most add-on packages is dependant on an additional secondary-level activation.

For example, in order to use the Internet Toolkit, you must first install and activate a licensed copy of ProvideX. Then, you have to invoke the activation utility again to enter a *package key* for the Internet Toolkit add-on. Each add-on must be activated separately with its own key (unless it is included in a bundle-type activation). Each package key is created for the same serial number as the base system.

To remove an expired package from a Windows system, simply remove its activation key from the activation dialog. (This does not affect other activations.) To remove an expired package from a UNIX/Linux system, delete the `ACTIVATE.PVX` file, then re-activate ProvideX with the expired package omitted. For details, see **Activation Procedures**, *p.10*.

Proprietary Activations

Some development shops maintain their own key generation system to control user access to the software they sell with ProvideX. These applications are encoded with a special *owner code* (issued by Sage Software Canada Ltd.) that permits the generation of activation keys by the software developer. The ProvideX activation utility is used to process the proprietary activation keys, which are recorded as *secondary-level* package activations.

However, the majority of ProvideX-built applications are sold without the need for secondary activations. Most development shops implement distribution controls within their applications; i.e., *password protection*.

System Requirements

ProvideX does not impose further hardware prerequisites beyond what is generally needed to run supported systems. Your ProvideX applications may necessitate specific resources to be run efficiently; however, the base system itself runs under a minimal configuration:

Windows	2-3 MB of RAM and 15MB of hard disk space
UNIX / Linux	1-1.5 MB of RAM and 10MB of hard disk space

The ProvideX website maintains an up-to-date list of supported operating systems. Refer to the ***Product Downloads*** link to see if a ProvideX installation is available for your specific Windows or UNIX/Linux system.

Installation Instructions

Before downloading and installing ProvideX, ensure that you have read and understood the information discussed under **Licensing and Activation Keys** and **System Requirements**.

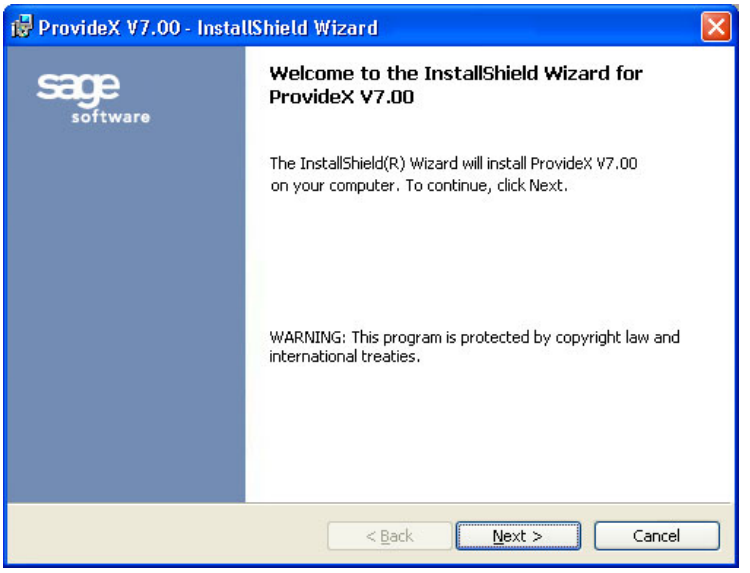
ProvideX products can be obtained from your dealer/distributor or downloaded for direct installation from the ProvideX website, **www.pvx.com**. Refer to the **Downloads** page for a complete list of available products. Each link allows you to download the appropriate installation file and save it to disk.

This section describes procedures for the installation of ProvideX on different platforms: **Windows Setup** and **UNIX/Linux Setup**. For product options that require separate installation/activation procedures, see **ODBC, WindX, & JavX Installations**, *p.37*.

Windows Setup

The instructions below apply to all installations and upgrades of ProvideX for Windows (95, 98, ME, NT4, 2000, XP, and Server 2003). After you download the installation program (e.g., `ProvideX V7.00.exe`), follow these steps to install ProvideX on your computer:

1. If possible, remain connected to the Internet. The installation process may include some options to download additional components.
2. Double-click on the ProvideX installation program that was downloaded to your computer to begin the installation process. The installation program launches an *InstallShield Wizard* that will take you through a series of dialogue boxes:



3. Follow the wizard instructions and click **Next >** to complete each step:
   - **License Agreement** explains terms for the use of ProvideX. *You must acknowledge the terms of this license agreement before proceeding*.
   - **Customer Information** provides fields for changing ownership information and to apply user restrictions (on operating systems that allow multiple users).
   - **Installation Type** allows you to **Upgrade** from an older version of ProvideX or **Install New** to set up ProvideX for the first time. Select the **Install New** option if you wish to retain your older version; e.g., install a trial copy for testing purposes. *This dialogue is not displayed during a first-time installation*.
   - **Destination Folder** specifies where the ProvideX executive and associated libraries are to be copied. Click **Change..** to choose a location other than the default.
   - **Setup Type** gives you the option to copy the **Complete** package or to select specific features for a **Custom** setup.
   - The final step installs ProvideX components onto your hard disk and displays a progress bar to indicate the current installation status. *This process may take several minutes*.
4. When all components are copied to disk, the *activation utility* may be invoked (automatically or through a dialogue box) depending on the circumstances:
   - If this is a new installation, a ***demo*** activation is applied automatically.
   - If this is a maintenance update for an existing copy of ProvideX, the activation is transferred automatically from the original version.
   - If this a purchased upgrade, you must establish a *new* activation for ProvideX.

   For complete activation instructions, see **Windows Activation**, *p.10*.
5. When the activation is recorded, close the final installation status window. The **Sage Software Canada Ltd** menu (with **ProvideX** shortcuts) will appear under **Programs** in the Windows **Start** menu.
6. To start ProvideX, click the new **ProvideX** shortcut under the **Start > Programs > Sage Software Canada Ltd** menu.

Note: *Activations* are verified by ProvideX at startup, not by the activation utility. Therefore, if an activation is recorded incorrectly, the session will fail and return the error message: `NO VALID ACTIVATION FOUND!!.`

You can review or change your activation status at any time by clicking the **Activation** shortcut via **Programs > Sage Software Canada Ltd > ProvideX** under the Windows **Start** menu.

ProvideX AutoUpdater (Windows)

As of Version 7, ProvideX/WindX client installations may be configured to check for and perform *automatic* installation updates when connected to ProvideX on the server. For system requirements, and a complete list of features, refer to the ProvideX ***AutoUpdater*** documentation.

UNIX/Linux Setup

The instructions below apply to all installations and upgrades of ProvideX for UNIX and Linux systems.

ProvideX UNIX/Linux distribution files have a `.taz` extension, which is a short form for `.tar.Z`. This file is a compressed version of a UNIX `.tar` file that contains a complete set of ProvideX software and directories. After you download the pv*xxxxxx*`.taz` file to a `/tmp` directory, follow these steps to expand, install, and activate ProvideX on your computer:

1. Change directories to the `/tmp` directory and rename the pv*xxxxxx*`.taz` with a `.tar.Z` extension so that it can be uncompressed:

```
umask 0
cd /tmp
mv pvxxxxxx.taz pvxxxxxx.tar.Z
uncompress pvxxxxxx.tar.Z
```

2. Create a new directory to receive the ProvideX software and move into it. We *recommend* that you use `/pvx` for the directory name:

```
mkdir /pvx
cd /pvx
```

3. Use the `tar` command to copy the software files into the `/pvx` directory:

```
tar xvf /tmp/pvxxxxxx.tar
```

ProvideX is fully installed once these files are copied from pv*xxxxxx*`.tar`; however, an *activation* is required in order to run the software.

Warning: Please read the license agreement in the `LICENSE.TXT` file (included in the pv*xxxxxx*`.tar`) before proceeding with the ProvideX for UNIX/Linux activation. By activating ProvideX, you acknowledge that you have read the license agreement, understand it, and agree to be bound by its terms and conditions.

4. Invoke the `pvxactv` activation program to establish a demo, temporary, or permanent activation. For further instructions, refer to the **UNIX/Linux Activation**.

5. Once the activation is recorded, enter the following commands to start ProvideX:

```
cd /pvx
/pvx/pvx
```

Always use the full path name when running `pvx` or `pvxactv`. Failing to use the full path name to the executable may result in `NO VALID ACTIVATION FOUND!!`

Important: The *permissions* for files and directories installed with ProvideX must be correct for the users and groups that require access. A minimum of read access is required. For activation and permission issues, see **Trouble-Shooting**, *p.44*.

You can review or change your activation status at any time by running the UNIX `pvxactv` program.

Activation Procedures

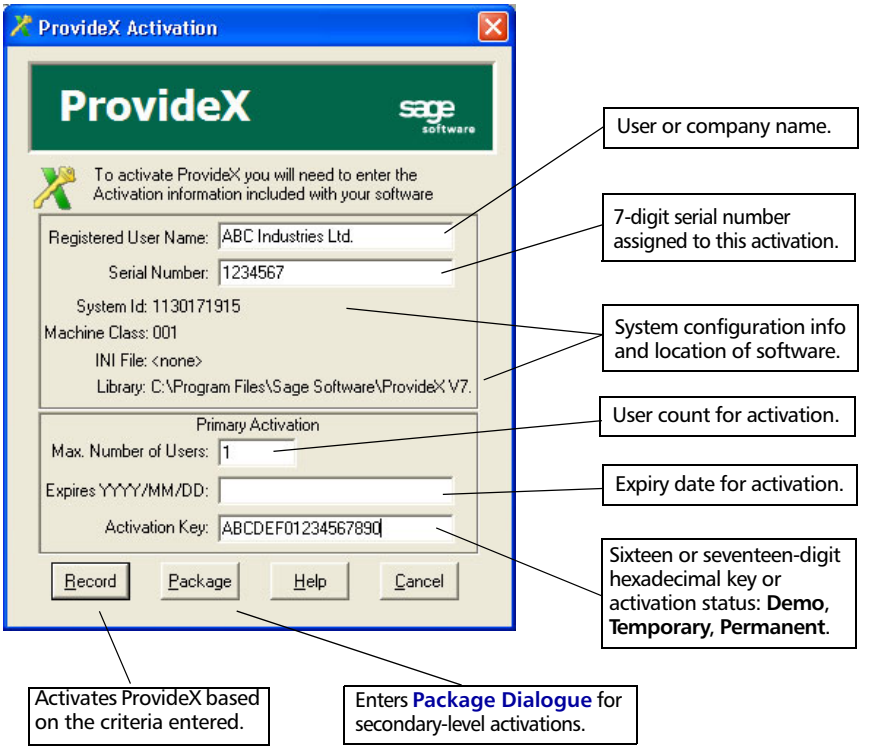
The final stage of the installation process requires that valid activation information be recorded. This is a prerequisite to using any ProvideX software.

A ProvideX *demo* activation is recorded with a default serial number of 079999, a user count of 1, and an expiry date set to 30 days from the date of installation. In order to record a *temporary* or *permanent* activation, you must obtain a license and be issued a unique serial number, user count, expiry date (if any), plus an activation key — the existing demo activation must be *completely removed* and replaced by these new activation values. Activation categories are described under **Licensing and Activation Keys**, *p.4*.

The utilities for performing a ProvideX **Windows Activation** and a ProvideX **UNIX/Linux Activation** are described in the sections below.

Windows Activation

The Windows **ProvideX Activation** interface appears as follows:



This utility (`pvxwactv.exe`) runs as part of the ProvideX installation. It can also be used to review/alter your activation status at any time; i.e., click the **Activation** shortcut via **Programs > Sage Software Canada Ltd > ProvideX** under the Windows **Start** menu. Activation information recorded by this utility is stored in the file `ACTIVATE.PVX`, which is located in the `PVX\LIB\KEYS` folder. The `ACTIVATE.PVX` file will not exist until the first primary-level activation is recorded.

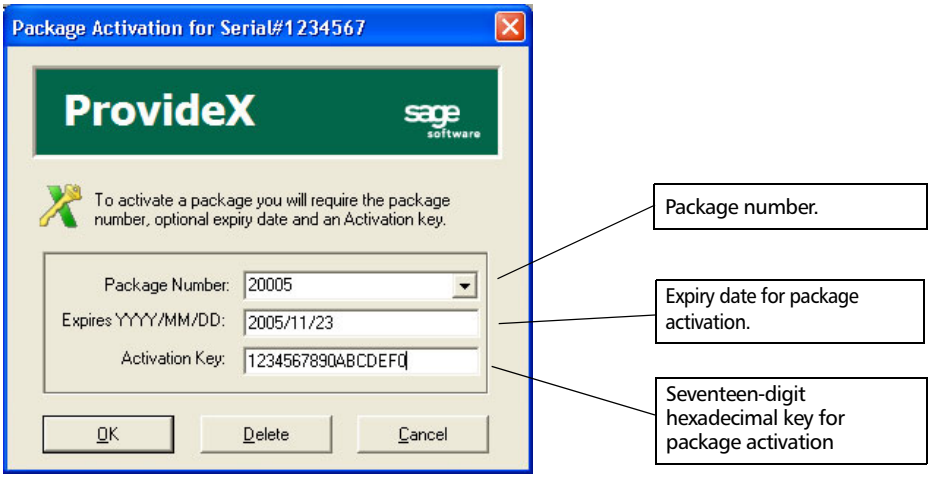
The following elements are displayed in the **ProvideX Activation** dialogue:

Registered User Name:	***Field***. User or company name associated with the installed copy of ProvideX.
Serial Number:	***Field***. Seven-digit serial number. This value is issued under license along with the temporary/permanent key. For a demo activation, the serial number is always **`0799999`**.
System ID:	***Derived Information***. Generated value based on the system where ProvideX is installed. This value is unique to the current system configuration and the location of installed components. If ProvideX is moved, this value changes.
Machine Class:	***Derived Information.*** Value determined by the operating system and processor type.
INI File:	***Derived Information.*** Path to the `PVX.INI` file used to load and maintain ProvideX environment settings. By default, ProvideX creates a new `PVX.INI` file in the folder where ProvideX is first installed. Refer the section **INI Files (Windows)** for more information on specifying and using ProvideX INI files.
Library:	***Derived Information.*** Path to ProvideX library files.
Max. Number of Users:	***Field***. Number of simultaneous users that can run this copy of ProvideX. This field must match the exact user count registered for the license, otherwise the activation will fail.
Expires YYYY/MM/DD:	***Field***. This date must match the expiry date for the license, otherwise the activation will fail. If a license has no expiry date, this field must be left *blank* or cleared manually before the permanent activation is recorded.
Activation Key:	***Field***. Hexadecimal value. Sixteen-digit temporary key *or* seventeen-digit permanent key issued under license along with a serial number. Once the activation is recorded, this field displays the words **`Demo`**, **`Temporary`**, **`Permanent`** to indicate current activation status. ProvideX is automatically activated in demo mode when it is installed on a Windows system for the first time. Refer to **Licensing and Activation Keys** for more information.

Important: When installing a ProvideX upgrade, ensure that all fields contain the correct values for the new activation; e.g., the expiry date field must match the expiry date supplied for the permanent activation – if no expiry date is given, then ensure that this field is left blank.

Package Dialogue

The secondary-level **Package Activation** dialogue can be reached via the initial **ProvideX Activation** dialogue.



The following elements are displayed in this dialogue:

Package Number:	***Field-Dropdown***. The product/add-on package number. This field also provides a dropdown list of all package numbers associated with the current ProvideX activation.
Expires YYYY/MM/DD:	***Field***. This date must match the expiry date for the product license, otherwise the activation will fail. If a license has no expiry date, this field must be left blank.
Activation Key:	***Field***. Seventeen-digit hexadecimal key generated for this product. Refer to **Licensing and Activation Keys** for information on the various types of activation keys.

Note: Activations are verified by ProvideX at startup, not by the activation utility. Therefore, if an activation is recorded incorrectly, the session will fail and return the error message: `NO VALID ACTIVATION FOUND!!` See **Trouble-Shooting**, *p.44*.

UNIX/Linux Activation

The pvxactv program provides an interface for entering and recording activation keys and querying activation status. Activation information recorded by the pvxactv program is stored in a file called ACTIVATE.PVX, which is located in the /pvx/lib directory. This file does not exist until the first primary-level activation is recorded. For a description of *demo*, *temporary*, and *permanent* activation types, refer to **Licensing and Activation Keys**, *p.4*.

Demo Activation

If you wish to use ProvideX in demo mode, enter the following commands to run the activation utility directly without prompts:

```
cd /pvx
/pvx/pvxactv -d
```

The -d parameter automatically records a demo activation. This allows you to run ProvideX for a *trial period* of 30 days, or until a new temporary or permanent key is recorded. For a description of all available parameters, refer to the section **Command Line Activation Parameters**, *p.15*.

New Activation (Prompted)

A new activation must be recorded for every new installation, whenever installed software has been moved to a new location, or if the ACTIVATE.PVX file has been deleted. Run pvxactv without arguments to launch a series of prompts and options:

```
cd /pvx
/pvx/pvxactv
```

Match your activation information to the prompts as they appear:

`Enter name of registered user:`	User or company name associated with the installed copy of ProvideX.
`Enter serial number:`	Seven-digit serial number supplied with the purchase of a ProvideX license.

After the first two prompts, a partial summary of the Activation info is displayed:

```
Activation info
    User identification: ABC Industries Ltd.
          Serial number: 1234567
          System number: 1059068746
          Machine Class: 686
```

If you receive a PACKAGE# prompt, refer to **Activation Status Options**, *p.14*.

The System number and Machine Class are derived from the current system configuration based on the location of installed components. This information must be supplied to your dealer/distributor in order to obtain a permanent activation key. As mentioned earlier, each activation is locked to the original system configuration. If the System number and Machine Class values change (i.e., the software is moved), a new activation key will be required.

The remaining prompts appear below the `Activation info`:

`Maximum user count:` Number of simultaneous users allowed to run under this copy of ProvideX.

You must enter the exact user count registered under this license, otherwise the activation will fail.

`Expiry date (YYYY/MM/DD or press <ENTER>):` Expiry date for the current license.

The exact expiry date for the license must be entered, otherwise the activation will fail. If the license has *no expiry date*, press Enter (with no text entered) to skip.

`Enter activation key:` Sixteen-digit hexadecimal temporary key *or* seventeen-digit hexadecimal permanent key.

Once the activation key is entered, the primary-level activation will be recorded, and a final prompt will present further activation options (described below).

Activation Status Options

Once an activation is recorded (an `ACTIVATE.PVX` file is created) you can review or change your activation status at any time by running `pvxactv`. This displays a summary of the current activation followed by a prompt that gives you the opportunity to make some changes:

```
ProvideX Activation utility
---------------------------
Activation info
    User identification: Demonstration System
          Serial number: 0799999
          System number: 3415516501
          Machine Class: 686
          Maximum users: 1
             Activation: ProvideX Expires on 2006/12/20 *Temporary*
                  Flags: 40010001
Please enter one of the following:
 A - To enter/update ProvideX activation keys
 P - To enter/update Package activation keys
 U - To record an Update key
 Q - To save changes and quit
Enter A, P, U, or Q:
```

The available options are outlined with examples in the sections below:

Option A. This prompts you to record a ProvideX Base system activation; e.g.,

```
ProvideX activation
  Maximum user count:5
  Expiry date (YYYY/MM/DD or press <ENTER>):
  Enter activation key:1234567890123456
Activation recorded
```

Option P. This prompts you for a secondary-level activation (add-on package); e.g.,

```
Enter the number for the Package you wish to change:200001
*Maximum Package number is 32000
Enter the number for the Package you wish to change:20001
Activation for PACKAGE#20001:
  Expiry date (YYYY/MM/DD or press <ENTER>):
  Enter activation key:1234567890123456
Activation recorded
```

Option U. This allows you to enter an update key for your current activation; e.g.,

```
Update key entry
  Enter activation key:1234567890123456
* Update key must start with a 'U'
  Enter activation key:U1234567890123456
Activation recorded
```

Deleting an Activation. Entering an exisiting package activation, via ***Option P,*** will result in a prompt that allows you to remove or change the activation; e.g.,

```
Enter A, P, U, or Q:p
Enter the number for the Package you wish to change:20001
There is a key on file for this package already.
Enter D to delete it, C to change it, or press ENTER to skip: d
```

Command Line Activation Parameters

The ProvideX activation for Windows and UNIX/Linux can be set up to run directly from the command line. The activation utility supports various parameters that can be used in place of the interface/prompts described earlier.

Important: When performing an activation from the command line, ensure that *all* key-related parameters are supplied together, and contain valid information. These specific parameters are listed with an asterisk * below. Arguments that include spaces must be enclosed in quotes; e.g., `-n "ABC Industries Ltd."`

UNIX/Linux prompt PVXACTV *parameters:*

`? or -a`		Display current activation information.
`-?`		Display help message.
`-d`		Demo activation.
`-e` *date*	*	Expiry (YYYYMMDD or 0) - 0 with a temporary key yields 30 days.
`-i`		Initialize activation file before starting.
`-k` *key*	*	Activation key.
`-l` *lib_path*		Location of the ProvideX `LIB` directory
`-m #`		Magic # to activate Run-Time PVX.
`-n` *name*	*	Registered name of user.
`-o #`		Owner ID (assigned by ProvideX).
`-s #`	*	Software serial number.
`-u #`	*	Number of users.
`-v`		Verbose (the Auto Activation operates quietly by default).

Windows command prompt `pvxwactv.exe` *parameters:*

`-a` *filename*		Capture current activation information and copy into *filename*.
`-c` *ini_file*		INI configuration file to affect (default `pvx.ini`).
`-d`		Demo activation.
`-e` *date*	*	Expiry (YYYYMMDD or 0) - 0 with a temporary key yields 30 days.
`-h`		Run hidden - suppress `Activation Recorded` message.
`-i`		Initialize activation file before starting.
`-k` *key*	*	Activation key.
`-l` *lib_path*		Location of the ProvideX `LIB` directory.
`-n` *name*	*	Registered name of user.
`-q`		Run the activation automatically.
`-s` *#*	*	Software serial number.
`-u` *#*	*	Number of users.

Note: Activations are verified by ProvideX at startup, not by the activation utility. Therefore, if an activation is recorded incorrectly, the session will fail and return the error message: `NO VALID ACTIVATION FOUND!!` See **Trouble-Shooting**, *p.44*.

Launching ProvideX

Formats

1. *Windows 32-Bit:* *path*\pvx\pvxwin32.exe [*winprms*][*ini*][*prog*][*overrides*][*pvxprms*][-ARG *list*]
2. *UNIX:* *path*/pvx/pvx [*prog*][*overrides*][*pvxparms*][-ARG *list*]

Where

path\pvx\pvxwin32.exe *path*/pvx/pvx	Specific Windows or UNIX/Linux command to start ProvideX in Command mode. Use the *full pathname*; e.g., in Windows: `C:\MY_DIRECTORY\pvx\pvxwin32.exe` The path and filename of the ProvideX executable may be retrieved during execution using `ARG(0)`.

Note: The total length of all Windows 32-Bit command elements (path, parameters, and arguments included) cannot exceed 512 bytes.

-ARG *list*	The dash and keyword (`-ARG`) mark the start of a list of optional arguments you can pass to the program on startup. The list of arguments (*arg1 arg2* ...) is space-separated. If an argument contains spaces, enclose it in quotation marks. For more information on arguments, see the **NAR** system variable and the **ARG( )** function in the *ProvideX Language Reference* manual.
ini	Name of an INI file to use (*for Windows only*). Optional. String expression. You can have different startup commands with different INI files so that each application can be set up with it's own defaults for fonts, window size, etc. ***Default:*** *windows dir path*`\pvx.ini`; e.g., `c:\pvx\pvxwin32.exe` *this.ini that_program* To return the current value, use `ARG(-1)`. ProvideX accepts quotes enclosing your INI filename and spaces within the INI filename in the command line argument. See **INI Files (Windows)**, *p.21*.
overrides	Optional arguments that override startup. Valid overrides: `-ID=`*FID val* **FID(0)** value the session is to start with. `-DR=`*path* Directory ProvideX will use as its HWD (Home Working Directory) e.g., `C:\pvx\pvxwin32.exe THAT_PROGRAM -ID=T0 -DR=MARY`

prog	Optional name of a program to run at startup (your lead program name, limited to 64 bytes); e.g.,
	`C:\pvx\pvxwin32.exe -PC=10`
	`c:\my_program\to_run_first`

pvxprms Optional ProvideX system parameters to be set from the command line. Most ProvideX system parameters are valid, including:

`-SZ=`*integer* Maximum memory (e.g., `-SZ=16000`).
`-XT=0` or `1` Exit direct to OS (e.g., `-XT=1`).

The dash is mandatory and indicates that the argument is a parameter. Do not use single quotes. However, it is better programming practice to set/reset parameters from within ProvideX applications via the **SET_PARAM** directive. See **System Parameters**, *p.19*.

winprms Optional window parameters (*for Windows only*):

`-HD` Force start with initial window hidden.
`-MN` Force start with initial window minimized.
`-MX` Force start with initial window maximized.
`-RS` Force start with initial window restored.

Note: The command line options `-MN`, `-MX`, `-HD`, and `-RS` will override all other initial start window considerations. Normally, if the ProvideX icon is set to be minimized or maximized then ProvideX will honour that setting for the initial window. If the icon is set to "Normal Window", then ProvideX will look at its saved window location from the INI file to determine what state to bring the window up in.

Description

The formats above describe the various syntax elements you can include in your command line to start ProvideX. We strongly recommend that you always include the full (absolute) pathname of the executable file in this command.

To Return Command-Line Values

As noted above, you can use the **ARG( )** and **FID( )** functions as well as the **HWD**, **NAR** and **LPG** system variables to retrieve the values in current use. For more information, refer to the *ProvideX Language Reference* manual.

Examples

The following example illustrates the basic command in Windows:

```
c:\my_directory\pvx\pvxwin32.exe
```

This command includes a lead program:

```
c:\my_directory\pvx\pvxwin32.exe my_program
```

Other syntax elements appear as follows:

```
c:\pvx\pvxwin32.exe -mn program_1 -dr=h:\acct_pay
c:\pvx\pvxwin32.exe myapp.ini my_program -id=t0 -dr=mary -xt=1
c:\my_directory\pvx\pvxwin32.exe myapp.ini windx
c:\my_directory\pvx\pvxwin32.exe myapp.ini *nthost
```

Customizing ProvideX

The ProvideX environment is dynamic. There are many ways to customize ProvideX to suit your application's requirements. You can add/change devices and modify other aspects of the underlying operating system, as part of the general launch procedure or during a ProvideX session.

Your ProvideX environment can be configured in several ways for different purposes:

System Parameters
Environment Variables
INI Files (Windows)
START_UP Initialization Program
Windows Services (Windows NT/2000/XP).

System Parameters

System parameters can be used at startup to define system operations under ProvideX, including system settings, default operating modes, and environmental emulation modes. Each parameter consists of a two-character code enclosed in single quotes. Some are Boolean switches (0 or negative sign to indicate *off*, 1 or "no sign" to indicate *on*), and some require specific values in order to be set.

System parameters are usually set once at the beginning of an application; e.g., in a lead program following the START_UP procedure. They can also be specified directly on the PVX command line; in which case, they do not require single quotes and must be prefixed with a dash or slash (See **Launching ProvideX**, *p.17*.)

Use the **SET_PARAM** directive to set/reset parameters within your ProvideX applications. The *UCP utility can also be used to display and change parameters in the system. Use the **PRM** system variable to return a string of all current system parameters. Use the **PRM** function to return the settings for specific parameters.

Example:

```
PRINT PRM
-'3D',-'AD',-'AH','AI'=10,-'B0','BF'=10,-'BT',-'BX','BY'=1970,-'CD','CS',
'CT'=0,'CU'=36,-'D0',-'DC','DF'=0,'DL'=0,'DP'=46,'DT'=0,'DW'=0,-'EG','EL'=0,
-'EO',-'ES',-'EX',-'F4','FB'=5,-'FC','FF'=0,-'FI','FO'=0,-'FU',-'FL','FP',
'FS'=138,-'FT',-'FX',-'F,',-'I0',-'I2','IC',-'IM','IR','IS'=5,-'IZ',-'KR',
'LB'=4,-'LC',-'LD',-'LE','LS'=1,-'LU',-'LZ','MB'=0,-'MC','MF'=50,-'MP','NE',
-'NI',-'NK',-'NL',-'NN',-'NR',-'OC','OL'=25,'OM',-'OP','OR','OW'=0,'PC'=0,
'PD'=2,-'PO',-'PU','PW'=36,-'PZ','QF'=1,'Q_'=2,'Q^'=2,-'QS',-'QT',-'RI',
'RN'=1,'RP',-'RR',-'RS',-'SC',-'SD',-'SF',-'SK','SL'=32,-'SP',-'SR','SV'=1,
'SZ'=32000,-'TB','TC'=0,'TH'=44,-'TL',-'TN',-'TT',-'TU',-'TX','VP'=48,'VR'=0,
'VW'=0,'WB','WD'=10000,-'WF','WH'=0,'WI'=1000,-'WK','WT'=2,-'XC',-'XF',-'XI',
-'XT',-'ZP',-'DD','!B'=3,'!U'=0,-'1U'

PRINT PRM('LC')
0
```

A complete and detailed list of ProvideX system parameters is available in Chapter 6 of the *ProvideX Language Reference*.

START_UP Initialization Program

The START_UP initialization program allows for the preloading of variables and/or files, the setting of various ProvideX system parameters, and the implementation of a security system. During the initialization process for an application, ProvideX executes the system startup program `*start.up`, which in turn calls the user-developed program START_UP, if it exists.

START_UP must be physically located in the *Start-In* (*HWD)* directory in order for it to be read during initialization. If it is not defined (in the correct directory) no initialization will be performed. However, an initialization program other than START_UP can be assigned using PVXSTART, see **Environment Variables**, *p.29*.

Some of the typical uses of a START_UP program include:

- Setting system parameters
- Opening global files
- Defining global variables/functions
- Setting **PREFIX**
- Assigning FID value.

Example:

```
0010 ! Sample START_UP program
0020 PREFIX "PROG/ DATA/"
0030 ERROR_HANDLER "*ERROR"
0040 ADDR "DATCHK"
0050 SET_PARAM 'SZ'=400,'CE','PC'=10
0060 LET %CUSTDB = GFN; OPEN (%CUSTDB) "CUSTDB"
0070 END
```

Running Applications from Initialization

Once the initialization program has completed execution, ProvideX will load and run a *lead program*, if specified on the command line. The **LPG** system variable can be used to determine the name of the lead program.

Note: The initialization procedure should not execute the application directly.

Attempting to run an application within START_UP causes ProvideX to incorrectly report application errors. ProvideX internally does a `CALL "START_UP"` when initializing ProvideX. Doing a RUN *within* the initialization procedure will prevent START_UP from terminating and will not let the initialization process conclude. It also causes ProvideX programs to run 1 level deeper than normal.

This also affects use of WindX/JavX thin clients to connect via *NTHOST/*NTSLAVE or the Application Server, as well as spawning processes via *WINDX.UTL;SPAWN or *SERVER;SPAWN.

START_UP initialization issues are discussed in the section **Trouble-Shooting**, *p.44*

INI Files (Windows)

Various properties of the ProvideX Windows environment can be specified through the use of flat text files identified by a .INI (or .PVX) suffix. The contents of these files must adhere to the standard format for Windows initialization files: headers are enclosed in square brackets and parameters are indicated by name, equal sign, and associated value. INI parameters can be used to load/maintain several environment settings, including:

- Font and font size (used in calculating column/row sizes for the screen display).
- Location of the initial window and its state (minimized, maximized, or normal).
- Title of the initial window.

The ProvideX INI file is passed is an argument to PVXWIN32.EXE on the command line. For more information, refer to the command line syntax described under **Launching ProvideX**, *p.17*.

Specifying an INI File

If an INI file is not specified, then ProvideX resorts to the default PVX.INI; however, there are many reasons to create and use application-specific INI files:

- Control resources between different ProvideX applications running on the same system. Without controls, there can be severe problems due to conflicting libraries, resource files, and even activation keys.
- Specify unique icons and other resources to identify your application and extend its functionality.
- Launch different types of programs from within your application; i.e., you may have one INI for starting your application, and a different one to run hidden tasks.
- Lock users out of modifying basic properties, such as font and font size, which could have an adverse effect in your application.
- Control resources between ProvideX-built applications and applications from different companies running on the same system.

The INI file that ProvideX uses when starting is determined on the command line; e.g,

```
c:\pvx\pvxwin32.exe c:\myapp\myapp.ini mystartprog -ARG myarguments
c:\pvx\pvxwin32.exe c:\myapp\myapp.ini windx
c:\pvx\pvxwin32.exe c:\myapp\myapp.ini *nthost
```

Note: If the INI file is specified on the WindX, ***NTSLAVE** or ***NTHOST** command line, then any instances of ProvideX started from within the ProvideX session will make use of the same INI file automatically.

File Locations

The following search rules apply to the default `pvx.ini` file and for any INI file specified on the command line that does not have an explicit path:

1. Windows directory (`\Windows` or `\Winnt`).
2. Current directory.
3. Directory where `PVXWIN32.EXE` is currently being run.

When ProvideX needs to update an INI for items such as the last window state, it will update the INI file found during ProvideX initialization, based on the search pattern above. If no INI is found during initialization and ProvideX needs to update some INI values, then a new `PVX.INI` file will be automatically created/updated in the directory where `PVXWIN32.EXE` was executed.

INI files can also be set to *read-only*, which prevents ProvideX from updating window properties on exit. This ensures that no matter the location, or in what state, the window is in when ProvideX is exited, the window will be restored to default properties when ProvideX is restarted.

INI Contents

The sections and parameters supported for use in a ProvideX `.INI` file are listed below:

`[Config]` ***Section***

`Library=C:\PVX\LIB`

Pathname of the ProvideX library that contains the `ACTIVATE.PVX`. The user count is controlled from this location.

`Fid0=`*text*

Override setting of **FID(0)**.

`Icon=`*text*

Name of the icon to be used by ProvideX. This can be the name of an icon contained in a specified `resourcelib` file or it can be an icon file name (e.g., *path/myicon.ico*).

`ResourceLib=lib.dll`

DLL file that contains bitmaps and icons that extend the internal bitmaps `{!name}` available in your ProvideX session. If you require both a 16-bit DLL and a 32-bit DLL, then specify the 16-bit DLL here.

`ResourceLib32=lib.dll`

32-bit DLL file that contains bitmaps and icons that extend the internal bitmaps `{!name}` available in your ProvideX session. This need only be used to specify the 32-bit DLL if you have different DLLs for 16-bit and 32-bit ProvideX and wish to maintain a single INI file.

`CommandLine=`*text -prm -prm* `....`

Optional argument used to define the command line to be used when running ProvideX. This argument is used only if an INI file is specified. Its contents replace the INI file name in the original command sequence.

Warning: The `CommandLine=` parameter overrides any command line used to launch ProvideX. This can have adverse effects on programs such as `*NTHOST/*NTSLAVE`, the Application Server, `*WINDX.UTL;SPAWN` and `*Server;Spawn`.

`ShutdownMsg=`*text*

There are three possible options for this parameter:

`Text`	Specifies text to appear in a message box when the user attempts a shutdown and a PVX session is still running.
`^Text`	Same as `=Text`, except that the user is not able to exit ProvideX.
`*ignore*`	Allows ProvideX to be exited while a program is running without asking if the user really wants to shutdown. This is for applications that run in the background or as NT services.

`Debug=`*n*

This enables access to the debugging environment windows. Values are:

`0`	***Default.*** Debugging available if no lead program specified – no debugging windows if a lead program is given.
`1`	Debugging windows are always available.
`-1`	Debugging windows are never available.

`Directory=`*text*

Starting directory name.

`SystemPalette=`*n*

When set to `1`, ProvideX uses the Windows colour palette. When set to `0`, ProvideX does not use the Windows colour palette. By default, *n*=`1`.

`ButtonCursor=`*n*

Cursor number to be used when the pointer is over a button.

`ButtonDisableCursor=`*n*

Cursor number to be used the when the pointer is over a disabled button.

`GrayMapping=`*n*

When set to `1`, ProvideX substitutes the light and dark gray with the colours in the Windows desktop. When set to `0`, then light and dark gray are used. By default, *n*=`1`.

`Disabled3Detch=`*n*

Controls the use of 3D etching on disabled objects, where `0` is off and `1` is on. By default, *n*=`0`

`RetryDeviceError=`*n*

Controls what happens on a timeout from a physical device such as an LPT port. System parameters **'DT'** (Device Timeout) and **'WT'** (Number of Retries) also affect the behaviour of device timeouts. When *n* is set to 0, then an err=0 is reported immediately after a **DT** time has occurred. When *n* is set to 1 or 2, then a `Device Timed Out` message box appears and asks the user to retry or abort. If the user aborts, then an err=0 is reported immediately. When set to 2, then a **BREAK** is also issued. By default, *n*=0.

`Cursor`*n*`=`*text*

Overrides settings for the graphic display of the mouse pointer (as controlled via the **'CURSOR'** mnemonic). The *n* identifies a cursor number to override and *text* specifies the pointer to use. The *text* value can be the name of a valid resource within an associated resource library or an *asterisk* ***** followed by a standard system cursor name.

These are represented as follows:

`*`	Standard arrow (`IDC_ARROW`).
`*appstart`	Standard arrow and small hourglass (`IDC_APPSTARTING`).
`*arrow`	Standard arrow (`IDC_ARROW`).
`*cross`	Crosshair (`IDC_CROSS`).
`*ibeam`	I-beam (`IDC_IBEAM`).
`*hand`	Windows 98/Me, Windows 2000/XP: Hand (`IDC_HAND`).
`*help`	Arrow and question mark (`IDC_HELP`).
`*no`	Slashed circle (`IDC_NO`).
`*size`	Four-pointed arrow pointing north, south, east, and west (`IDC_SIZEALL`).
`*sizenesw`	Double-pointed arrow pointing northeast and southwest (`IDC_SIZENESW`).
`*sizenwse`	Double-pointed arrow pointing northwest and southeast (`IDC_SIZENWSE`).
`*sizewe`	Double-pointed arrow pointing west and east (`IDC_SIZEWE`).
`*sizens`	Double-pointed arrow pointing north and south (`IDC_SIZENS`).
`*uparrow`	Vertical arrow (`IDC_UPARROW`).
`*wait`	Hourglass (`IDC_WAIT`).

The `IDC` value, shown in parenthesis, represents the corresponding Windows LoadCursor API resource identifier.

In the the following example, cursor #5 (`'CURSOR'(5)` mnemonic) is changed to the standard Windows Hand mouse pointer:

```
Cursor5=*hand
```

The following INI entries can be used to reset cursors #5 and #6 to their pre-V7.50 states:

```
Cursor5=PvxPush
Cursor6=PvxNoPush
```

`ForceMessageBoxOnTop= -1 | 1`

When set to 1, this forces a message box to display on the desk top when running as a service. A -1 does not display the message box and automatically selects the OK button.

`[WindowFrame]` *Section*

`TypeSizeLoc=1,649,467,66,66`

Defines the screen mode, size and position. It is updated automatically by ProvideX when the session terminates. The first value represents screen mode: 1 (normal), 2 (minimized), or 3 (maximized). The second and third values represent window width and height in graphical units. The fourth and fifth values represent the window position, X and Y coordinates for the upper left corner in graphical units.

`Caption=`*text*

Default caption line to appear on the ProvideX main window.

`ToolBar=`*nnn*

Toolbar size. By default, the toolbar is set to the same size as the windows caption boxes.

`MessageBar=`*nnn*

Message bar size. By default, the message bar is set to the same size as the windows caption boxes plus a few pixels to allow for the 3D effect.

`[Font]` *Section*

`Name=`*text*

Name of the default fixed-width font (e.g., `Courier New`). The fixed-width font name and size are used by ProvideX to determine the size of a column or row in a Window.

`Points=12`

Default fixed-width font size. The fixed-width font name and size are used by ProvideX to determine the size of a column or row in a Window.

`Charset=0`

Character set (0=ANSI) -- ***Do not change!!!***

`Bold=0 | 1`

Bold toggle. When set to 1, all foreground data is displayed as Bold.

`LockFont=0 | 1`

Font locking. When set to 1, the user may not change the fixed-width font from the system menu (button in top left corner of the screen).

`TipFontName=`*text*

Name of the font to use in floating tips (e.g., `Courier New`). The default is 'MS Sans Serif'.

`TipFontPoints=10`

Tip font size. The default is 10.

`SystemFontName=`*text*

Name of the default graphical font to use (e.g., `Times New Roman`). If this entry is omitted, then the system uses the Windows default system font.

`SystemFontPoints=12`

Default graphical font size.

`[ODBC]`, `[OCI]`, *and* `[DB2]` *Sections*

Each of these section headings can be inserted into the `PVX.ini` file to manually override specific defaults. The **`[ODBC]`**, **`[OCI]`** and **`[DB2]`** INI parameters and settings are described along with corresponding SQL keywords (if available).

`ACCESS= READ | WRITE`

ODBC & DB2. Determines type of file access required. Default is `WRITE`. SQL keywords: `SQL_ACCESS_MODE`, `SQL_MODE_READ_ONLY`, `SQL_MODE_READ_WRITE`.

`AUTOCOMMIT= ON | OFF`

ODBC & DB2. Determines auto commit functionality of the database driver. It is applicable only if the driver supports transactions. SQL keywords: `SQL_AUTOCOMMIT`, `SQL_AUTOCOMMIT_ON`, `SQL_AUTOCOMMIT_OFF`.

`CURSOR_USE= DRIVER | ODBC | IF_NEEDED`

ODBC & DB2. Controls the type of cursor to be used within the ODBC connection. `DRIVER` (default) assumes the specific driver's own cursors. `ODBC` causes the ODBC interface to use the "Driver Managers" cursor library that may provide additional functionality not available within the database driver. `IF_NEEDED` tells the system to use the specific database driver's own cursor functionality unless the additional functionality is requested specifically. SQL keywords: `SQL_ODBC_CURSORS`, `SQL_CUR_USE_DRIVER`, `SQL_CUR_USE_ODBC`, `SQL_CUR_USE_IF_NEEDED`.

ISOLATION= UNCOMMITED | COMMITED | REPEATABLE | SERIAL | VERSIONING

ODBC & DB2. Controls the isolation that this connection will have relative to other processes on the same database. In particular, it controls *D*irty reads (reading data that may be rolled back), Non-*R*epeatable reads (reading data after being changed by other transactions), and *P*hantom reads (reading data newly added to file).

Settings include: UNCOMMITED (*D, R, P* possible), COMMITED (*D* possible, *R* & *P* not possible), REPEATABLE (*P* possible, *D* & *R* not possible), SERIAL (*D, R,* & *P* not possible), VERSIONING (*D, R,* & *P* not possible, but uses versioning as opposed to record locks).

SQL keywords: SQL_TXN_ISOLATION, SQL_TXN_READ_UNCOMMITTED, SQL_TXN_REPEATABLE_READ, SQL_TXN_SERIALIZABLE, SQL_TXN_VERSIONING.

CONCURRENCY= READONLY | LOCK | OPT_VERSION | OPT_VALUE

ODBC & DB2. Determines the type of con-current access control/locking to be used. READONLY sets the cursor is set to read only - no updates allowed. LOCK applies low-level record locking. OPT_VERSION causes optimistic locking with the database version control to be used. OPT_VALUE causes optimistic locking with comparing record/column values to be used. SQL keywords: SQL_CONCURRENCY, SQL_CONCUR_READ_ONLY, SQL_CONCUR_LOCK, SQL_CONCUR_ROWVER, SQL_CONCUR_VALUES.

CURSOR_TYPE= FORWARD | STATIC | KEYSET | DYNAMIC

ODBC & DB2. Defines the type of cursor that is to be used. FORWARD indicates that any result sets can be read in a forward only direction. STATIC indicates that the result set is static. KEYSET forces the cursor to use/maintain record keys in a keyset. DYNAMIC indicates that the cursor is effective in the current rowset only. SQL keywords: SQL_CURSOR_TYPE, SQL_CURSOR_FORWARD_ONLY, SQL_CURSOR_STATIC, SQL_CURSOR_KEYSET_DRIVEN, SQL_CURSOR_DYNAMIC.

KEYSET_SIZE=*n*

ODBC & DB2. Size of the keyset for use with the cursor. SQL keyword: SQL_KEYSET_SIZE.

MAXROWS=*n*

ODBC & DB2. Maximum number of rows/records returned. SQL keyword: SQL_MAX_ROWS.

ROWSET_SIZE=*n*

ODBC & DB2. Size of the rowset used by the cursor. SQL keyword: SQL_ROWSET_SIZE.

TIMEOUT=*n*

ODBC & DB2. Timeout value for any SQL operation (time before error 0 returned). SQL keyword: SQL_QUERY_TIMEOUT.

DB=*dbname* *or* QUALIFIER=*dbname*

ODBC only. Qualifies the specific database that you wish to use when using a driver to service multiple databases. SQL keyword: SQL_CURRENT_QUALIFER.

TOP=*n*

ODBC, OCI & DB2. If specified, then ProvideX uses the TOP keyword in all SELECT statements, where possible. If *n* is non-zero, then the **KEF( )** and **KEL( )** functions issue a SELECT TOP 1... SQL statement, which improves system performance. If *n* > 0, then PVX issues SELECT TOP *n* to reduce the data transferred. TOP= -1 indicates the driver supports TOP, however normal reading should not use it. Default is zero (TOP *not* supported).

ORACLE= Y | N

ODBC & OCI. Indicates if the database uses ORACLE SQL sequence. If ORACLE= and TOP= are used, then SELECT commands are generated as SELECT * FROM (SELECT * FROM TABLE) WHERE ROWNUM < 1. Defaults: ORACLE=Y (under [OCI]), ORACLE=N (under [ODBC]).

EXTROPT=*text*

ODBC, OCI & DB2. Controls the format of the SELECT statement used to process an EXTRACT. By default, PVX generates a SELECT * FROM *table* FOR UPDATE WHERE... If specified, then *text* is substituted in place of FOR UPDATE. In addition, if the first character of *text* is $, then the remaining characters of *text* are placed at the end of the SELECT statement rather than after the file name. This allows for different variations of SQL to be supported.

USER=*text*

ODBC, OCI & DB2. Default user Id. Used if no user id is supplied on **OPEN**.

PSWD=*text*

ODBC, OCI & DB2. Default password. Used if no password supplied on **OPEN**.

Warning: The above setting is not secure. Anyone with access to the INI can read the password.

DATEFMT=*text*

ODBC, OCI & DB2. String mapping of date format to be returned. Applies to all date fields in table.

POSUPDATE=*x*

ODBC & DB2. Use one of the following: M (must use positioned update), O (default, optionally use positioned update), N (never use positioned update).

DEBUGIT=*text*

ODBC, OCI & DB2. String to append to SQL statement along with program name and line number for debugging purposes. *text* must be the comment character(s) appropriate to the database. For example, "–" is the comment identifier for Microsoft SQL Server; anything after -- is ignored by SQL Server when compiling the SQL statement.

PREPARE=*x*

ODBC, OCI & DB2. Set to use prepared statements. *x* can be `1`, `Y` or `y`. Use `PREPARE=N` to turn off this behavior. Prepared statements are pre-compiled SQL that may improve performance.

NULLPADKEY=*x*

ODBC, OCI & DB2. Set to force keys to be padded to full length with the null character, `$00$`. *x* can be `1`, `Y` or `y`. Use `NULLPADKEY=N` to turn off this behavior.

UNIQUE=*x*

ODBC, OCI & DB2. Set for new opens to be on a unique connection. *x* can be `1`, `Y` or `y`. Use `UNIQUE=N` to turn off this behavior.

TEXTMAX=*n*

ODBC & DB2. Defines the maximum length of a returned column. The default is `TEXTMAX=4096`.

Debug Sections

Debug windows and their respective settings from ProvideX's Windows Debug Environment are saved in the INI file under the following section headings:

```
[TraceWindow]
[WatchWindow]
[BreakWindow]
[CommandWindow]
```

Environment Variables

ProvideX maintains a set of OS-level environment variables that can be used to define and control various aspects of its current working environment. If your application requires changes from the default values, you can initialize the required environment variables (using OS-specific mechanisms) prior to launching ProvideX.

Note: Only set environment variables if you absolutely need them. It is better programming practice to control these actions from inside the ProvideX environment than from outside (where possible).

The following environment variables can be manipulated and accessed for use in your ProvideX applications:

`LANG` Default language code. The ProvideX default is `EN` (English).

`PVXEDIT` Name of editor used to maintain on-line program help information. The defaults are `edit` for DOS, and `vi` for UNIX.

`PVXFID0` Value for **FID(0)**. Many ProvideX applications use the logical device name of the terminal; i.e., file 0 (zero) as a unique identifier. Under DOS or Windows, the value for **FID(0)** defaults to `T0` (T-zero) unless you override it by setting this environment variable.

To avoid the problem of duplicate **FID(0)** values in a DOS or Windows multi-user environment or network, this value must be different for each PC user. One of the simplest ways to uniquely identify each PC is to place a different `SET PVXFID0=`*xx* command in each `AUTOEXEC.BAT`. *xx* is the value for **FID(0)**.

On systems that support NetBIOS, another solution is to use the system variable `NID` (which contains the network station name) or `UID` (user ID) as a key to a file which contains one record for each station. The record in the file contains the **FID(0)** value that your `START_UP` passes to the **SETFID** directive.

In the following example, `START_UP` using NetBIOS node name for **FID(0)**, terminates if the `NID` is not on file.

```
0010 ! START_UP - xxxxx
......
0100 X=HFN; OPEN (X) "Z:\PVXAPP\DATA\FIDFILE"
0110 READ (X,KEY=NID,DOM=0190) F$
0120 SETFID F$
0130 CLOSE (X)
......
0190 PRINT 'CS','RB',"System restricted!!"
0200 WAIT 2; CLOSE(X)
0210 QUIT
```

Note: ProvideX does not require unique **FID(0)** values. Only applications that were originally designed to have unique **FID(0)** values need change or control the **FID(0)**.

PVXFLMAP File name change map. Use this variable when migrating files from one host environment to another, to translate invalid filename characters to valid ones for the new environment. To create a map, use one or more two-character codes as values. The first character is the filename character as it appears in your program; the second is the character ProvideX will pass to the OS as a substitute.

For example, in UNIX the file name \\AB\\ is valid, but to DOS the backslash is a directory delimiter. To translate the filename to a valid DOS filename:

```
SET PVXFLMAP=\_
1000 OPEN(1)"\\AB\\" REM would open "__AB__"
```

PVXLS `ls` Command - ***UNIX only.***

To read a directory under UNIX / Linux, ProvideX internally uses an `ls -a` command. Output from this OS command is parsed and a file name list is built from this output.

Only set PVXLS if your OS does not have a `ls -a` command — in this case, set it to whatever command the operating system has that matches the output of an `ls -a` command on a typical UNIX/Linux system.

PVXLIB Location of PVX library directory. By default, ProvideX searches for its libraries in the directory where the PVX.EXE program is located. *The library must have the name LIB.* The default directory is \PVX\LIB on the current disk drive. You can override the location of the library by adding the following directive to your AUTOEXEC.BAT or to a ProvideX startup batch file; e.g.,

`SET PVXLIB=`*pathname*

where *pathname* is the path to your directory containing the ProvideX library. However, if ProvideX can't locate the library, it will not start. Instead, it generates `Error #100 - No driver for terminal type or library missing.`

PVXSTART Path and file name of initialization file to be used in place of the default START_UP. See **START_UP Initialization Program** for further details.

PVXTMP Name of the directory where temporary files will be placed. Default: current directory.

USER UserID - ***DOS only.***

DOS does not fully support the concept of user registration. To establish a user name to make applications work properly, create and/or modify a .BAT file to include the directive:

`SET USER=`*`userID`*

where *userID* is the user's name. The ProvideX system variables `UID` and `WHO` return whatever value is in the `USER` environment variable or an asterisk (`*`) when this variable is not set.

Example:

```
SET USER=Jones
PRINT UID
Jones
```

TERM Terminal Type - ***UNIX only.***

This environment variable is normally set automatically by your UNIX/Linux operating system and matches the type of terminal you are using such as `ansi`, `wyse60`, `linux`, and so on.

ProvideX uses the current value of the `TERM` environment variable, to determine which ProvideX device driver in the `*dev` directory, and which keyboard map, to **LOAD/RUN** during initialization.

Windows Services (Windows NT/2000/XP)

ProvideX can be installed as a ***Windows Service***, which is a special process designed to start and run automatically under Windows (NT/2000/XP) directly from start up; e.g., a networking or remote access procedure. The three primary methods for setting up service entry points to ProvideX (`pvxwin32.exe`) are outlined below.

Microsoft Windows Resource Kit Utilities

The Windows *Resource Kit* includes two utilities for creating a service: `Instsrv.exe`, which installs/removes system services from Windows, and `Srvany.exe`, which allows an application to run as a service. The following steps describe how to use these utilities:

1. Install `instsrv.exe` and `srvany.exe` in a directory; e.g., create `d:\tools\` and copy the files to this location.
2. Create the service using the command line:
   `instsrv` *`ServiceName path`*`\srvany.exe`

   ServiceName Name that will appear in the Services control panel.

   Path Full path to the `srvany.exe` executable

   Example:

   ```
   D:\tools\instsrv MyApp d:\tools\srvany.exe
   ```

3. From your desktop, select Start > Parameters > Control Panel > Services. Select the service name created in Step 2, then select the Start-Up button.

4. Set up the service as Auto-Start, and select a *user account* and password, or a LocalSystem account. If a user account is selected, then the ***NTHOST** program and any spawn tasks will be hidden from the desktop.

   If the LocalSystem account is selected, then Allow Service to Interact with Desktop can be enabled. With this option enabled the ***NTHOST** program and any spawned tasks will be visible on the desktop. Note that LocalSystem account does not have network access by default.

Important: The remaining steps involve editing the registry. Before you change the registry, make sure you understand how to restore it if a problem occurs. For information about how to do this, view the Restoring the Registry *Help* topic in `Regedit.exe` or the Restoring a Registry Key *Help* topic in `Regedt32.exe`.

5. From your desktop select Start > Run > regedt32. Under `HKEY_LOCAL_MACHINE\SYSTEM\Current\ControlSet\Services\`*ServiceName* create a key named `'Parameters'` . Under this key, create the following values:

```
Name : Application
Type  : REG_SZ
String : ProvideX_Installation_Path\Pvxwin32.exe
```

```
Name : AppParameters
Type  : REG_SZ
String : *NTHOST parameters
```

```
Name : AppDirectory
Type  : REG_SZ
String : ProvideX_Installation_Path\lib
```

   Due to a restriction enforced by Windows NT on services, an application can either be interactive (have a console, read keyboard input, etc.) or have network access - not both at the same time.

6. To allow LocalSystem services on any machine in the domain to access a specific share on a server, use the Registry Editor to add the name of that share to: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\LanmanServer\ Parameters\NullSessionShare`.

   To allow access to named pipes add an entry to: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\LanmanServer\ Parameters\NullSessionPipes`.

To allow ***all*** shares and pipes on the server to be accessed by LocalSystem services, add a value:
`HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\LanmanServer\ Parameters\RestrictNullSessAccess.`

```
Type  : REG_DWORD
Value : 0
```

The above entry effectively allows everyone in the domain to access that server.

7. Quit `regedt32.exe` and reboot the server.

   The service should start automatically and begin to listen on the socket specified in the `AppParameters` key, or 10000 if no socket was specified.

Note: The parameters associated with NTHOST should be tested using a shortcut to ensure they are correct prior to creating the service.

ProvideX Command Line Options

As of Version 7.00, ProvideX supports command line options for installation as a true Windows Service. Currently, only *Start* and *Stop* functions are supported – *Pause* and *Continue* functions may be added in a future release.

Install Service:
`path\pvx\pvxwin32.exe -i [pvx.ini]` *ServiceName ServiceDisplayName StartType StartInDirectory CommandLine*[*ServiceDescription*][*Account*][*Password*]

Where:

ServiceName	Name that will appear in the Services control panel.
ServiceDisplayName	Name that appears in the Services control panel.
StartType	Service startup option, value between one and three: 1- auto start service 2 - manually start service 3 - service is disabled
StartInDirectory	Directory used as the startup directory for the service.
CommandLine	Command-line arguments to start ProvideX with.
ServiceDescription	Description that appears in the Services control panel.
Account	Domain and account to run the service as. If the account is set to "interactive" (case insensitive), then the service will be setup to use the Local System Account and interact with the desktop.
Password	Password for the above account.

Start Service:

```
path\pvx\pvxwin32.exe -s [pvx.ini] ServiceName
```

Where:

ServiceName Name that will appear in the Services control panel.

Un-install Service

```
path\pvx\pvxwin32.exe -u [pvx.ini] ServiceName
```

Where:

ServiceName Name that will appear in the Services control panel.

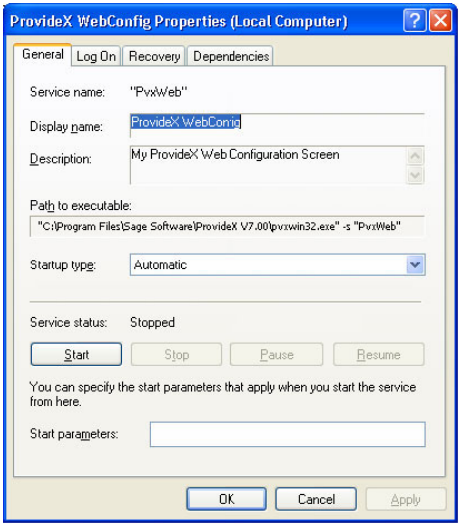
The `-s` option will not start the service unless it is executed by the Service Manager. It is documented here for completeness and it will appear as the command line argument in the Service Properties panel. If the `pvx.ini` file used by the executable contains a `LogFile=` entry, then any errors that occur during the install, un-install, or start of the service will be logged.

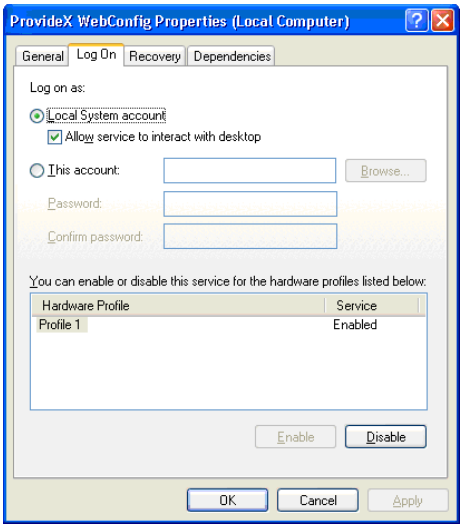
If a Windows function fails, then the last error code will be logged. These can be found in MSDN under "System Error Codes" (index entry "error codes [Win32]"). Any parameter containing spaces requires a delimiter character of either *quote* " or *apostrophe* '. Leading/trailing apostrophes are always stripped. Leading/trailing quotes are stripped from *StartType*, *ServiceDisplayName*, *ServiceDescription*, *Account*, and *Password*.

`TCB(170)` returns the service handle. The value will be zero if the program is not running as a service. This only applies to a ProvideX executable installed and started using the `-i` and `-s` options above.

Examples:

```
>pvxwin32.exe -i "PvxWeb" "ProvideX WebConfig" 1 "C:\Program
      Files\Sage\ProvideX V7.00" "*web\webcfg" "My ProvideX Web
      Configuration Screen" "interactive"
>pvxwin32.exe -u "PvxWeb"
```





ProvideX Object *obj\ntservice

As of Version 7.00, the ProvideX object *obj\ntservice can be used to manipulate Windows Services. The following methods are currently available:

CreateProvideXService( ) **DeleteProvideXService( )** **StartService( )**
StopService( ) **ServiceState( )** **OpenService( )**
CloseService( )

Example:

```
00010 let a=new("*obj/ntservice")
00020 let servicename$="pvxapp"
00030 let displayname$="providex appcfg"
00040 let description$="providex application server config"
00050 let commandline$="*appserv\config"
00060 let startindirectory$="c:\program files\sage software\providex v7.00"
00070 let starttype=1
00080 let useraccount$=""
00090 let userpassword$=""
00100 let interactwithdesktop=1
00110 print a'createprovidexservice(servicename$,displayname$,
      description$,commandline$,startindirectory$,starttype,useraccount$,
      userpassword$,interactwithdesktop)
00120 let sh=a'openservice(servicename$)
00130 print a'servicestate(sh)
00140 print a'startservice(sh)
00150 print a'servicestate(sh)
00155 escape
00160 print a'stopservice(sh)
00170 print a'deleteprovidexservice(servicename$)
00180 end
```

ODBC, WindX, and JavX Installations

While most add-ons and bundles can be installed, activated, and configured from within the initial ProvideX download, separate procedures may be required for the installation of certain product options. This product category includes the ProvideX **ODBC Local and Client/Server** drivers, the **WindX Thin Client**, and the **JavX Thin Client** — each have their own download procedures for installing components on your system.

The following sections discuss the basic installation and activation only. Refer to each product's documentation for *system requirements* and specific *configuration* and *operational* procedures.

ODBC Local and Client/Server

ProvideX ODBC allows your ProvideX database to be accessed using standard SQL commands. Two types of ProvideX ODBC driver can be obtained from your dealer/distributor or downloaded from the ProvideX website, **www.pvx.com**:

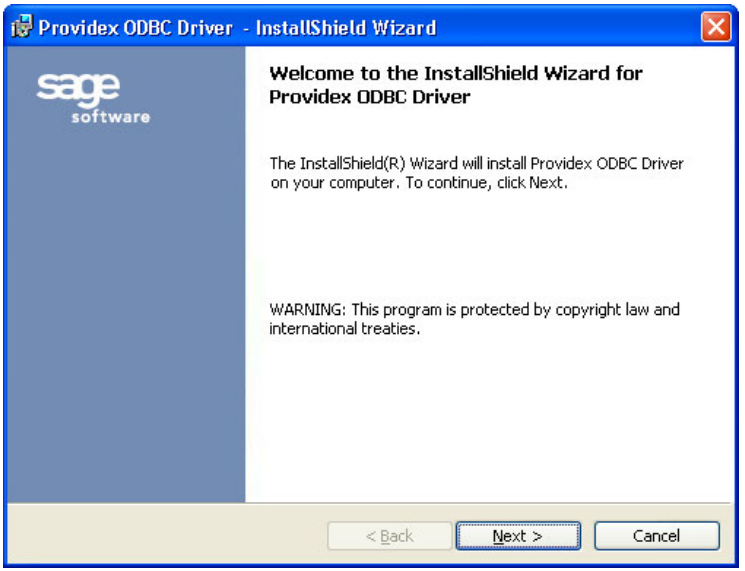
- ODBC ***Local/Standalone*** with read only and read/write capabilities. Each local license requires its own *serial number*, *user count*, and *activation key*.
- ODBC ***Client/Server***. The Client version of the driver is freely distributable (Windows only), but it must be connected to a fully installed and activated ***ProvideX File Server*** (included with the Professional and eCommerce bundles).

Note: ProvideX ODBC installations are available with or without Microsoft Data Access Components. If you choose not to install MDAC, the installation automatically verifies if your current version of MDAC (if any) is compatible with ProvideX ODBC.

ODBC Installation and Activation for Windows

The ODBC installation files (e.g., `odbc-client-4.10-windows-32bit-x86.exe` and `odbc-server-4.10-windows-32bit-x86.exe`) may be obtained from your dealer/distributor or from the ProvideX website. The installation process is virtually identical for all ODBC components. After downloading the appropriate installation program, follow these basic steps to install an ODBC driver or the ProvideX File Server on your computer:

1. If possible, remain connected to the Internet. The installation process may include some options to download additional MDAC components.
2. Double-click on the installation program that was downloaded to your computer to begin the installation process. This launches a series of ***InstallShield Wizard*** panels similar to the ProvideX installation.
3. Click **Next >** to continue. The installation program searches for existing ProvideX ODBC components. The Wizard then displays one of several different dialogue windows, depending on whether it is a completely new install, or if older/newer components exist on your machine.



- If upgrading from an earlier driver, you have the option to update System DSN entries – User and File DSN entries are not updated and should be removed or adjusted prior to the upgrade.
- If identical ODBC components exist on your machine, you will also be given the option to modify, repair, or remove existing driver/server components.
- If you are installing the ProvideX File Server while a ProvideX Windows Service is currently running on your computer, you will be warned that the existing service must be stopped before the installation can continue.

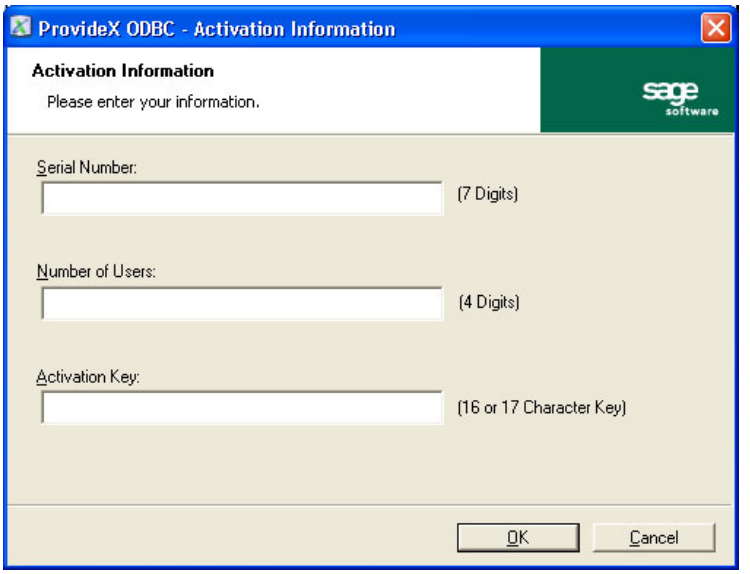
When the installation wizard has checked the above criteria and is cleared to proceed, it takes you through a series of dialogues. Follow the wizard instructions and click **Next >** to complete each step. The final step installs driver/server components onto your hard disk and displays a progress bar to indicate the current installation status. *This process may take several minutes.*

4. When all components are copied to disk, a ProvideX ODBC *activation* dialogue will appear (for new installations) otherwise the Wizard simply indicates that the driver or server has been updated successfully.

   A valid serial number, number of users, and activation key are required in order to set up and run a permanent version of the ODBC driver. The necessary activation information is issued with every purchase of a product package from Sage Software Canada Ltd or authorized dealer/distributor.

Note: When installing the ODBC driver as part of an eCommerce or Professional licence, use your *temporary key* for permanent ODBC activation. Permanent keys that are generated for bundled activations do not apply to ODBC components.

The ProvideX ODBC activation dialogue appears as follows:



If you press OK and the activation is invalid, you will be given the option to enter your information again. If you press Cancel, the activation utility automatically records a *demo mode* activation for the ODBC; in this case, the activation dialogue pops up for every ODBC connection and a "nag" message appears on every execute. For configuration details, refer to the *ODBC Local & Client/Server* manual.

ProvideX File Server Installation and Activation for UNIX/Linux

Obtain the ProvideX File Server distribution file from your dealer/distributor or via the ProvideX website. Ensure that you download the correct version for your specific UNIX/Linux operating system. The ODBC distribution file is named with a `.taz` extension, which is short for `.tar.Z`, a compressed version of a UNIX `.tar` file:

`po`*`xxxvvv`*`.taz`

Where:

xxx	identifies a specific operating system; e.g., `RH6` for Redhat versions 6 to 7.1.
vv	identifies the version of the File server; e.g., `312` for version 3.12.

The `po`*`xxxvvv`*`.taz` distribution file contains the following installation components:

`pvxodbs`	File Server executable.
`pvxodbs.conf.sample`	File Server configuration file (sample).
`install.txt`	Installation readme file.
`license.txt`	License agreement.
`podbcs`*`xx`*`.txt`	ODBC version readme file, containing current change information.

After you download the po*xxxvvv*.taz file to a /tmp directory, follow these steps to expand, install, and activate the ProvideX ODBC Server program on your computer:

1. Change directories to the /tmp directory and rename the po*xxxvvv*.taz with a .tar.Z extension so that it can be uncompressed:

```
umask 0
cd /tmp
mv poxxxvvv.taz poxxxvvv.tar.Z
uncompress poxxxvvv.tar.Z
```

2. Create the new directory to receive the ProvideX software and move into it. Use /usr/pvxodbc for the directory name:

```
mkdir /usr/pvxodbc
cd /usr/pvxodbc
```

3. Use the tar command to copy the software into the /usr/pvxodbc directory:

```
tar xvf /tmp/poxxxvvv.tar
```

4. Set the file permissions on the pvxodbs executable and configuration files to whatever is necessary depending on the *username* who will be running the server daemon (typically root):

```
chmod 500 pvxodbs
chmod 600 pvxodbs.conf.sample
chown root pvx*
chgrp root pvx*
```

5. If this is the first time the ODBC Server has been installed on this system, then rename the pvxodbs.conf.sample file to pvxodbs.conf.

```
mv pvxodbs.conf.sample pvxodbs.conf
```

   At this point, the installation of the ODBC Server is complete; however, the pvxodbs.conf file may require updated settings. Refer to the *ODBC Local & Client/Server* manual for more information on the UNIX/Linux configuration file and command line arguments.

6. The ProvideX ODBC Server must be activated in order to run the software. Enter your activation information in the pvxodbs.conf file as follows:

```
serial=xxxxx-y-zzzzzzzzzzzzzzzz
```

   Where:

xxxxx	Serial number.
y	User count.
zzzzzzzzzzzzzzzz	Activation key.

Note: To run the ODBC Server in demo mode, leave the serial= blank. When installing the ODBC as part of an eCommerce or Professional licence, use your *temporary key* for permanent ODBC activation. Permanent keys that are generated for bundled activations do not apply to ODBC components.

WindX Thin Client

WindX thin client technology allows you to create a remote graphics interface that will run ProvideX applications from any host. It provides convenient Windows access on the client side of your application, while maintaining processing and data storage on your UNIX, Linux, or Windows server (even if the server does not support GUI).

Two versions of WindX can be obtained from your dealer/distributor or downloaded from the ProvideX website, **www.pvx.com**:

- WindX ***Standalone***, which requires an individual license per client installation and is able to interact with any ProvideX application on any server. Each license has its own *serial number, user count, expiry date* and *activation key.*
- WindX ***Plugin***, which is freely distributable for clients but must connect with a ProvideX application on a server that maintains a multi-user Professional or eCommerce license. If the server is not licensed for plug-in access, server file/directory permissions are incorrect, or a ProvideX session cannot be established within 2 minutes of startup, then the plug-in will terminate automatically.

Note: The Standalone and Plugin downloads for WindX are for installation on Windows 95/98/ME/NT4/2000/XP client machines only.

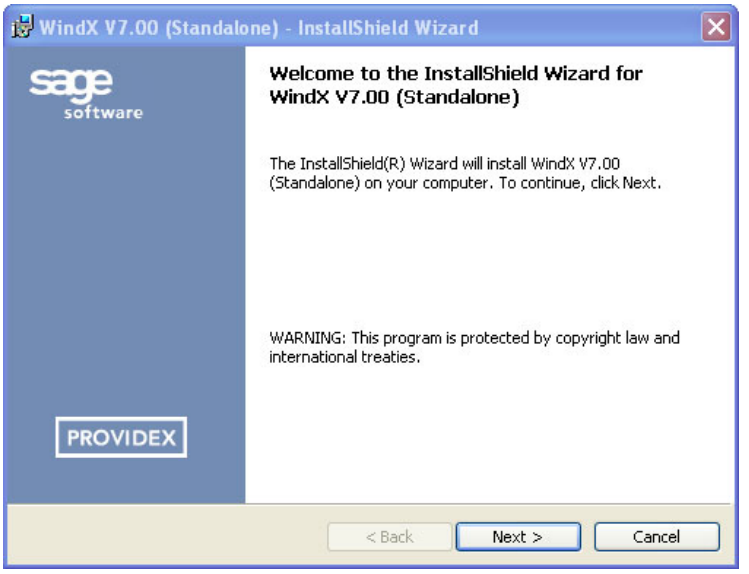
Client Installation

Click on a link to download the appropriate WindX installation file (i.e., Standalone or Plugin) and save it to disk. Once you download the installation program, follow these steps to install and activate WindX on your client computer:

1. If possible, remain connected to the Internet. The installation process may include some options to download additional components.
2. Double-click on the installation program that was downloaded to your computer to begin the installation process. The installation program launches an ***InstallShield Wizard*** and immediately checks for any existing versions of WindX.

   If an older version of WindX is detected, you will be given the option to upgrade/overwrite your existing version, or to install the new version in a different location. If your current version of WindX is identical to the new install, you will be given the option to modify, repair, or remove the existing components.

The wizard takes you through a series of dialogue boxes:



3. Follow the wizard instructions and click **Next >** to complete each step. The final step installs the driver onto your hard disk and displays a progress bar to indicate the current installation status. *This process may take several minutes.*

4. The ***InstallShield Wizard*** will be completed when all components are copied to disk. If you downloaded the ***Plugin***, WindX is fully installed at this point and will *skip further steps* in the installation process.

   However, if you downloaded a ***Standalone*** version of WindX, the ***InstallShield Wizard*** automatically invokes the ProvideX Activation utility:

   - If this is a ***new*** installation, a Demo Mode activation is applied automatically.
   - If this is a ***maintenance update*** for an existing copy of WindX, the activation is transferred automatically from the original version.
   - If this a ***purchased*** upgrade, you must establish a ***new*** activation for WindX.

   The activation for WindX Standalone is performed in the same manner as the ProvideX Windows Activation. The necessary activation information will be issued to you when you purchase a product package from Sage Software Canada Ltd. or your authorized dealer/distributor.

   For detailed activation instructions refer to procedures explained under ProvideX **Windows Activation**, *p.10*.

JavX Thin Client

The ProvideX Java-based thin-client, JavX, offers a platform-independent solution for displaying and interacting with your server-based ProvideX applications. ***Three editions of JavX are available for download:*** JavX SE (for PC/workstations), JavX AE (for portable devices), and JavX LE (for task-specific devices).

Depending on the implementation and on the client requirements, JavX may be deployed either as a Java application (permanently installed) or as a Java applet (downloaded as needed). In either case, the client side of a JavX implementation must be equipped with a Java2-compliant Runtime Environment (JRE). A Professional or eCommerce activation of ProvideX is required on the server side.

The following JavX components and utilities are downloadable for direct installation from the ProvideX website at **www.pvx.com**:

- ***JavX Developer's Kit***, which includes all the tools necessary to develop applications for JavX. The kit installs documentation, utilities, a tutorial, the most recent JavX [SE/AE/LE] JAR files, as well as an optional Java2-compliant Runtime Environment (JRE).
- `JavX.zip`, which contains the most recent versions of JavX [SE/AE/LE] in JAR (Java ARchive) format, a readme with notes on the latest enhancements, and a license file. For a JAR file to be run as an application, you will need to install the Java JRE, a Java JIT (Just-In-Time) compiler, or the full Java SDK.

The only installation file required to run JavX itself is the JAR file. However, if you are programming with JavX for the first time, we recommend that you take advantage of the tools provided in the JavX Developer's Kit.

JavX as an Application

When JavX is run as an application, the distributed JAR is installed directly on the workstation or device, and JavX has local access to the client system with no imposed security restrictions. Utilities such as InstallShield or InstallAnywhere can be used to create a custom routine for automating the installation procedure.

JavX as an Applet

Currently this option is only available for JavX SE. For this implementation, the JAR is not installed permanently on the client workstation, but is delivered (as needed) via web server to the client's web browser. Certain settings within the web page identifies the applet and its location then loads and runs the applet. If a copy of the applet is not cached on the local system then it automatically downloads it from the website.

Note: The *JavX Manual* explains all of the necessary configuration, HTML, and command line syntax for installing the different JavX editions.

Trouble-Shooting

The following trouble-shooting procedures are intended to help you quickly identify and resolve problems involving the installation, startup, and configuration of ProvideX: **Activation Issues**, **Permission Issues**, and **START_UP Initialization Failure**.

Activation Issues

While ProvideX activation is normally a straightforward process, some issues may arise because of missing files/directories, incorrect licensing, permission difficulties etc. Be aware that activations are verified during the ProvideX startup, not by the activation utility itself. You can review your activation status at any time by re-running the activation utility. For more information, see **Activation Procedures**, *p.10*.

The ProvideX version and serial number are displayed when starting a new session; e.g.,

```
ProvideX eCommerce (Ver:7.00/MS-WINDOWS) Serial Number:0700-001-1234567
(c) Copyright 1987-2005 Sage Software Canada Ltd. (All rights reserved)
  Website: http://www.pvx.com
```

The example above describes an activation for ProvideX eCommerce Version 7 for Windows. The machine class is 001 and the serial number is 1234567. This information can also be obtained programmatically as follows:

```
10 a$=hta(bin(tcb(0,2),4))
20 if mid(a$,3,1)="1" then print "Professional"
30 if mid(a$,3,1)="2" then print "eCommerce" else print "Base"
40 print ssn
```

The most common activation issues are discussed in the following sections:

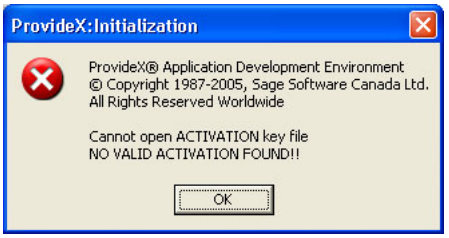
Removing an Activation in Windows and UNIX/Linux

Sometimes the best way to correct an activation problem is to completely remove the activation file `activate.pvx` and then restart the activation process, as follows:

Windows	Delete the directory `pvx\lib\keys`. This removes all contents related to activation keys, including the `activate.pvx` file.
UNIX / Linux	Delete the activation file `pvx/lib/ACTIVATE.PVX`.

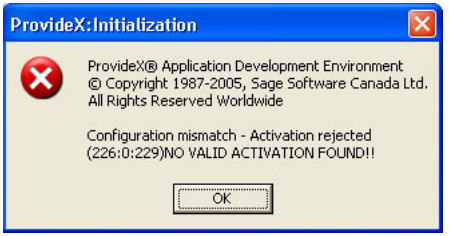
At this point you can re-activate your demo or temporary activation — permanent activations require *new activation values*. See **Activation Procedures**, *p.10*.

Windows Activation Messages



The above message indicates that ProvideX cannot locate or open the `activate.pvx` file. Possible causes of this problem include:

- Incorrect permissions on the directories leading to the activation file, or on `activate.pvx` itself. See **Permission Issues** below.
- The `Library=` setting in the INI file points to an invalid directory.
- The `PVXLIB` environment variable references an invalid directory.
- The `activate.pvx` file does not exist which means that the activation program was not run correctly (or at all).



The above message indicates that the activation data does not agree with the current machine configuration. This can happen for many different reasons, including:

- The pathname, `Library=`, in the `ini` file points to the wrong directory.
- The activation file was copied from another machine.
- ProvideX was moved from a different location on disk or from another file system.
- The system information has changed since the activation key was generated.
- An attempt was made to upgrade or install a permanent activation over a demo activation. When purchasing a license, the demo activation must be completely removed and replaced by the new temporary or permanent activation values you were issued; i.e., new serial number, user count, expiry date (if any), and activation key.
- The activation file was restored from a backup (i.e., system information has changed).

UNIX/Linux Activation Messages

Message: `Cannot open ACTIVATION key file`
`NO VALID ACTIVATION FOUND!!`

This message indicates that ProvideX cannot locate the `ACTIVATE.PVX` file. Possible causes of this problem include:

- Incorrect permissions on the directories leading to the activation file, or on `ACTIVATE.PVX` itself. See **Permission Issues** below.
- The `PVXLIB` environment variable has not been set or it references an invalid directory.
- The `ACTIVATE.PVX` file does not exist which means that the activation program was not run correctly (or at all).
- Use of `./` (dot-slash) instead the full pathname to launch the executable.

Message: `Configuration Mismatch - Activation Rejected`
`NO VALID ACTIVATION FOUND!!`

This message indicates that the activation data does not agree with the current machine configuration. Possible causes of this problem include:

- The activation file was copied from another machine.
- ProvideX was moved from a different location on disk or from another file system.
- The system information has changed since the activation key was generated.
- An attempt was made to upgrade or install a permanent activation over a demo activation. When purchasing a license, the demo activation must be completely removed and replaced by the new temporary or permanent activation values you were issued; i.e., new serial number, user count, expiry date (if any), and activation key.
- The activation file was restored from a backup (i.e., system information has changed).

Problems with Add-On Activations

Add-on packages require separate activations on top of the base system activation. Any attempts to **LOAD**, **RUN**, **CALL**, or **PERFORM** an add-on package without a proper activation, or after a trial activation has expired, will result in one of the following:
`Error #63: Not activated for this software package`
`Error #17: Invalid file type or contents`.

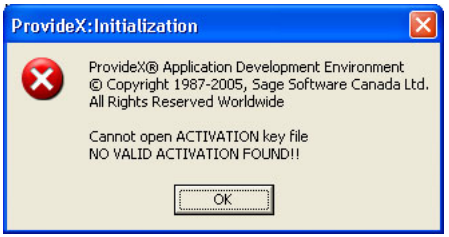
To avoid these messages, ensure that you obtain and record the necessary trial, temporary, or permanent activation information for the intended software packages.

Permission Issues

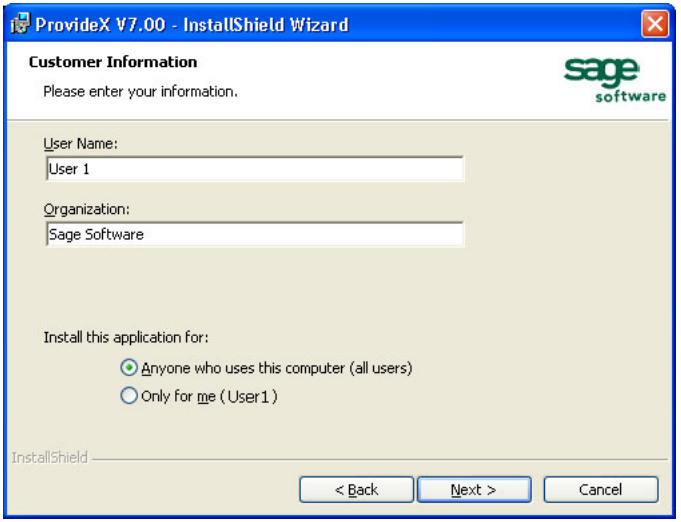
It is important that the files and directories installed with ProvideX have the correct permissions for the users and groups that require access. Incorrect permissions can result in a system's root/administrator user being the only one able to launch ProvideX. Attempts by other users to launch ProvideX will result in an activation error. For further information see **Activation Issues**, *p.44*.

Incorrect Permissions in Windows 2000/XP

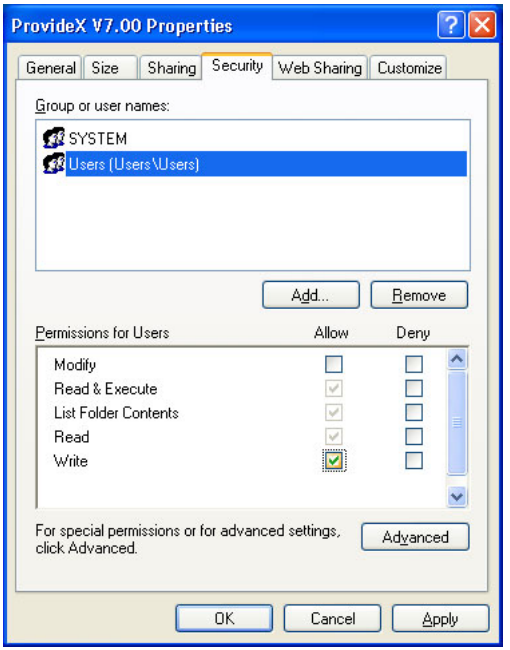
The following activation message is reported when there are incorrect permissions on directories that lead to the activation file. ProvideX will not run if it does not have direct access to the activation file.



This usually occurs when ProvideX is installed for a specific user only, which ultimately disables write permissions for anyone who is not the owner of the file. You can prevent this from happening by setting correct permissions during installation. In the Customer Information dialogue box, select Anyone who uses this computer (all users) to ensure that anyone who logs on to the computer will also be able to access ProvideX.



To resolve this issue after the software has been installed, use Windows Explorer to modify the properties of the path to where ProvideX is installed:



Incorrect Permissions in UNIX/Linux

Message: `Cannot open ACTIVATION key file`
`NO VALID ACTIVATION FOUND!!`

The above activation message will be reported when there are incorrect permissions on directories that lead to the activation file. ProvideX will not run if it does not have direct access to the activation file.

This problem is usually caused by the default `umask` setting of the user who installed ProvideX (normally root). Most UNIX/Linux machines assign a default `umask` value of 0022 for the root user, which ultimately disables the write permissions for all but the owner of the file. This problem can be prevented by installing ProvideX *after* the user changes their `umask` to a value of `0`; e.g.,

```
umask 0
```

To resolve the issue after the software has been installed, use the UNIX `chmod` command to add write permission to the activation file; e.g.,

```
chmod a+w /pvx/lib/ACTIVATE.PVX
```

START_UP Initialization Failure

If the initialization fails, consider the following reasons that START_UP may not be executing correctly:

- START_UP does not exist in the directory that the application launches ProvideX from. Prior to executing a **PREFIX** directive, ProvideX does not know where to search for any program or file, aside from the current directory.
- START_UP has incorrect file permissions (primarily in UNIX).
- START_UP is not a valid program file. The START_UP file is not a ProvideX program or a recognizable Serial program.
- The environment variable PVXSTART is set, but points to an invalid program.
- The START_UP file name is not completely upper case on a UNIX machine. ProvideX will be unable to locate the start up routine if the name is not completely in capitals; i.e., START_UP.
- START_UP stops executing prematurely due to an un-trapped error.