

## TProfile Help Contents

### **TProfile**

A Delphi component for working with .INI files

Copyright (c) 1995, Kenn Nesbitt

CompuServe: 76100, 57

Internet: kennn@netcom.com

The following help topics are available:

[What is TProfile?](#)

[Installing TProfile](#)

[Using TProfile](#)

## What is TProfile?

TProfile is a freeware component for Borland Delphi to simplify the task of working with .INI files.

Someone mentioned to me today that I was working too hard because I use `GetPrivateProfileString` and `WritePrivateProfileString` to work with .INI files. "There is a .VBX that does that and makes it a lot easier," he said. What a great idea, I thought. But why do I need a VBX when I can write my own Delphi component? So here it is.

I am releasing this component into the public domain, along with the source code in the hopes that others will find it useful, and that it help others learn how easy it is to write components. It took me about an hour to write this (and another hour of dinking around with the resource file for the component image).

You can do anything you like with this product except sell it. Feel free to copy and distribute it as much as you like as long as you don't charge anything for it. Just make sure you include everything in this zip file if you post it to an online service.

I'd love to hear your comments and/or suggestions. You can reach me at:

CompuServe: 76100,57  
Internet: [kennn@netcom.com](mailto:kennn@netcom.com)

## Installing TProfile

Follow these instructions to add the TProfile component to your Delphi programming environment:

1. Copy PROFILE.DCR and PROFILE.DCU into \delphi\lib.
2. Run Delphi and select Options|Install Components...
3. Select Add and type "Profile" in the Add Module dialog and press Enter.
4. Select OK.

The INI button should appear on the Additional page of your tool palette.

## Using TProfile

TProfile has the following properties:

[DefaultText](#)  
[Entry](#)  
[FileName](#)  
[Name](#)  
[Section](#)  
[State](#)  
[Tag](#)  
[Text](#)  
[Value](#)

All properties are read/write at design-time and runtime except the Name property which is read-only at runtime.

After setting the FileName, Section and Entry properties, you can change the INI file entry by setting the Text, State or Value properties.

You can read the INI file entry by reading the Text, State or Value properties. If the entry isn't found in the .INI file, the Text property will contain the same value as the DefaultText property.

Use the State property to read and write boolean (True/False) values. Also, if the Text property is set to 'True' or 'False', 'Yes' or 'No', 'On' or 'Off', '1' or '0', State will automatically be set to the corresponding boolean value.

Use the Text property to read and write string values. If the Value property is set, Text will contain a numeric integer string (e.g., '12345'). If the State property is set, Text will contain a corresponding value of 'True' or 'False'.

Use the Value property to read and write integer values. If the Text property is set to an integer numeric string (e.g., '12345'), the Value property will contain the corresponding integer value. If the State property is set, the Value property will contain a value of 1 for True or 0 for False.

That's about it. Here's some example code.

```
Profile1.FileName := 'MYPROG.INI';
Profile1.Section := 'Main';
Profile1.Entry := 'SomeEntry';
Profile1.DefaultText := 'Entry not found';
Profile1.Text := 'Here's some text';

{ This entry should be found }
MessageDlg(Profile1.Entry + '=' + Profile1.Text, mtInformation, [mbOK], 0);

{ This entry should not be found so Text
  should have the same value as DefaultText }
Profile1.Entry := 'AnotherEntry';
MessageDlg(Profile1.Entry + '=' + Profile1.Text, mtInformation, [mbOK], 0);

{ This entry should be found }
Profile1.Entry := 'SomeEntry';
MessageDlg(Profile1.Entry + '=' + Profile1.Text, mtInformation, [mbOK], 0);
```

## DefaultText Property

### Declaration

```
property DefaultText: string;
```

### Description

The DefaultText property indicates what value the Text property will contain if the specified entry is not found or the INI file does not exist.

## Entry Property

### Declaration

**property** Entry: **string**;

### Description

The Entry property indicates the name of the entry in the .INI file. In the following example, 'Options' is the [Section](#), the values to the left of the equals signs correspond to the Entry property (e.g., 'WordWrap', 'UserName', 'Level') and the values to the right of the equals sign correspond to the [State](#), [Text](#) and [Value](#) properties (e.g., True, 'Filbert McGillicutty', 5).

```
[Options]
WordWrap=True
UserName=Filbert McGillicutty
Level=5
```

**Note:** You must complete the Entry, FileName and Section properties before you can manipulate the State, Text for Value properties.

## FileName Property

### Declaration

**property** FileName: **string**;

### Description

The FileName property indicates which .INI file will be read to and written from. You should specify the entire file name (e.g., 'NOTEPAD.INI').

If FileName does not contain a fully qualified path and filename for the file, TProfile searches the Windows directory for the file. If the file does not exist, TProfile creates the file in the Windows directory when the State, Text or Value properties are set.

If FileName contains a fully qualified path and filename and the file does not exist, TProfile creates the file when the State, Text or Value properties are set. The specified directory must already exist.

An application should use a private (application-specific) initialization file to record information that affects only that application. This improves the performance of both the application and Windows itself by reducing the amount of information that Windows must read when it accesses the initialization file. The exception to this is that device drivers should use the SYSTEM.INI file, to reduce the number of initialization files Windows must open and read during the startup process.

**Note:** You must complete the Entry, FileName and Section properties before you can manipulate the State, Text or Value properties.

## Name Property

### Declaration

**property** Name: TComponentName;

### Description

The Name property contains the name of the component as referenced by other components. By default, Delphi assigns sequential names based on the type of the component, such as 'Button1', 'Button2', and so on. You may change these to suit your needs.

**Note:** Change component names only at design time.



## Section Property

### Declaration

**property** Section: **string**;

### Description

The Section property indicates the name of the section in the .INI file. In the following example, 'Options' is the Section, the values to the left of the equals signs correspond to the [Entry](#) property (e.g., 'WordWrap', 'UserName', 'Level') and the values to the right of the equals sign correspond to the [State](#), [Text](#) and [Value](#) properties (e.g., True, 'Filbert McGillicutty', 5).

```
[Options]
WordWrap=True
UserName=Filbert McGillicutty
Level=5
```

**Note:** You must complete the Entry, FileName and Section properties before you can manipulate the State, Text or Value properties.

## State Property

### Declaration

**property** State: boolean;

### Description

If the INI file entry is 'True', 'Yes', 'On' or '1', the State property will contain a value of True. If the INI file entry is 'False', 'No', 'Off' or '0', the State property will contain a value of False.

Setting the State property will place a value of 'True' or 'False' at the specified entry in the INI file.

**Note:** You must complete the [Entry](#), [FileName](#) and [Section](#) properties before you can manipulate the State, [Text](#) or [Value](#) properties.

## Tag Property

### Declaration

**property** Tag: Longint;

### Description

The Tag property is available to store an integer value as part of a component. While the Tag property has no meaning to Delphi, your application can use the property to store a value for its special needs.

## Text Property

### Declaration

**property** Text: **string**;

### Description

The Text property reads and sets the text value of the specified entry the INI file.

**Note:** You must complete the [Entry](#), [FileName](#) and [Section](#) properties before you can manipulate the [State](#), Text or [Value](#) properties.

## Value Property

### Declaration

**property** Value: Longint;

### Description

The Value property contains the numeric value of the specified entry in the INI file. If the entry has a value of 'True', 'Yes', or 'On', the Value property will contain a value of 1. If the entry has a value of 'False', 'No' or 'Off', the Value property will contain a value of 0.

**Note:** You must complete the [Entry](#), [FileName](#) and [Section](#) properties before you can manipulate the [State](#), [Text](#) or Value properties.

