

---

**Tutorial:**

Letzte Änderung:

Autor:

E-Mail:

Web:

Getestet mit:

Copyright:

Dank an:

**Registry**

25.05.2004

Marcel Kukelka (Chatfix)

software@KuKnet.de

www.KuKnet.de

Delphi 6, Delphi 7

© 2002-2004 KuKnet Software

MathiasSimmack (Delphi-Forum)

---

**1. Vorwort**

Die Registry ist ein wichtiger Bestandteil von Windows.

Alle Änderungen sollten nur mit Bedacht durchgeführt werden!

Für bestimmte Zugriffe benötigt man gegebenenfalls besondere Rechte.

**2. Erste Schritte**

Als erstes erweitert man die *Uses*-Liste:

```
uses {...}, Registry;
```

Danach benötigt man eine weitere Variable vom Typ *TRegistry*:

```
var  
  {...}  
  reg: TRegistry;
```



## 4. Aus der Registry lesen

Mit folgenden Anweisungen kann man Daten aus der Registry auslesen. Man benötigt hier allerdings Variablen, um die ausgelesenen Werte zu speichern.

Der Schlüssel wird in diesem Beispiel mit den Zugriffsrechten "Nur Lesen" geöffnet.

```
var
  var1: String;
  var2: Integer;

{...}

reg := TRegistry.Create(KEY_READ);
try
  reg.Rootkey := HKEY_LOCAL_MACHINE;
  //hier wird der Hauptschlüssel festgelegt
  if reg.OpenKey('SOFTWARE\DeinName\DeinProgramm', False) then
    //Schlüssel öffnen, bei True wird der Schlüssel erstellt, falls er noch nicht
    existiert
  begin
    var1 := reg.ReadString('Eigenschaft1');
    // Daten von Eigenschaft1 auslesen
    var2 := reg.ReadInteger('Eigenschaft2');
    // Daten von Eigenschaft2 auslesen
    reg.CloseKey;
    // Schlüssel schließen
  end else ShowMessage('Schlüssel konnte nicht geöffnet werden.');
```

```
finally
  reg.Free;
  // Variable reg freigeben
end;
```

Weitere Befehle zum Lesen aus der Registry:

- *ReadDate* Liest Daten vom Typ *TDateTime*
- *ReadDateTime* Liest Daten vom Typ *TDateTime*
- *ReadFloat* Liest Daten vom Typ *Double*
- *ReadTime* Liest Daten vom Typ *TDateTime*

## 5. Überprüfen ob ein Schlüssel / eine Eigenschaft existiert

Mit folgenden Anweisungen kann man überprüfen ob ein Schlüssel in der Registry oder eine Eigenschaft in dem geöffnetem Schlüssel existiert.

```
// existiert Schlüssel?
if reg.KeyExists('SOFTWARE\DeinName\DeinProgramm') then
  {...}
else
  {...}

// existiert Eigenschaft1 (Es muss ein Schlüssel geöffnet sein)?
if reg.ValueExists('Eigenschaft1') then
  {...}
else
  {...}
```

## **6. Werte aus der Registry löschen**

Mit folgender Anweisung kann man einen Schlüssel aus der Registry löschen:

```
reg.DeleteKey('SOFTWARE\DeinName\DeinProgramm');
```

Mit dieser Anweisung kann man eine Eigenschaft löschen:

```
reg.DeleteValue('Eigenschaft1');
```

Um Eigenschaften löschen zu können, muss der Schlüssel wie in Punkt 3 geöffnet werden.

Beide Funktionen geben folgende Werte zurück:

*True* - Löschen erfolgreich  
*False* - Löschen fehlgeschlagen

## **7. Schlüssel verschieben / kopieren**

Mit folgender Anweisung verschiebt man einen Schlüssel. Der erste Parameter gibt die Quelle, der zweite das Ziel an.

Um Werte verschieben zu können muss der Schlüssel wie in Punkt 3 geöffnet werden.

```
reg.MoveKey('SOFTWARE\DeinName\DeinProgramm', 'SOFTWARE\DeinProgramm', True);
```

Mit folgender Anweisung kopiert man einen Schlüssel. Der erste Parameter gibt die Quelle, der zweite das Ziel an.

```
reg.MoveKey('SOFTWARE\DeinName\DeinProgramm', 'SOFTWARE\DeinProgramm', False);
```