

## User Manual

# CANdb ++

Database Editor

Version 3.0

English

## **Imprint**

Vector Informatik GmbH  
Ingersheimer Straße 24  
D-70499 Stuttgart

The information and data given in this user manual can be changed without prior notice. No part of this manual may be reproduced in any form or by any means without the written permission of the publisher, regardless of which method or which instruments, electronic or mechanical, are used. All technical information, drafts, etc. are liable to law of copyright protection.

© Copyright 2010, Vector Informatik GmbH. Printed in Germany.

All rights reserved.

80318

# Table of contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	About this user manual	4
1.1.1	Access helps and conventions	4
1.1.2	Certification	5
1.1.3	Warranty	5
1.1.4	Support	5
1.1.5	Registered trademarks	5
<b>2</b>	<b>Basics</b>	<b>7</b>
2.1	Overview of CANdb++	8
2.2	CANdb++ data model	10
2.3	Installing CANdb++	10
2.4	CANdb++ program window	11
<b>3</b>	<b>Tutorial</b>	<b>13</b>
3.1	Overview	14
3.2	Program start	14
3.3	Creating a new CAN database	15
3.4	Creating and modifying objects	16
3.4.1	Creating new objects	16
3.4.2	Copying existing objects	18
3.4.3	Modifying existing objects	19
3.5	Linking objects	20
3.6	Showing the communications matrix	22
3.7	Value tables	23
3.8	Assigning value tables	25
3.9	User-defined attributes	25
3.10	Modifying the value of an object's user-defined attribute	27
3.11	Consistency check	28
<b>4</b>	<b>Version administration</b>	<b>29</b>
4.1	Preconditions for version administration capability	30
4.2	Version administration for CAN databases and objects	30
<b>5</b>	<b>Appendix A: File name extensions</b>	<b>31</b>
<b>6</b>	<b>Appendix B: INI files</b>	<b>32</b>
6.1	Vector.ini	32
6.2	CANdb.ini	33
<b>7</b>	<b>Appendix C: Address table</b>	<b>34</b>
<b>8</b>	<b>Index</b>	<b>37</b>



# 1 Introduction

In this chapter you find the following information:

---

1.1	About this user manual	page 4
	Access helps and conventions	
	Certification	
	Warranty	
	Support	
	Registered trademarks	

---

## 1.1 About this user manual

### 1.1.1 Access helps and conventions

#### To find information quickly

The user manual provides you the following access helps:








- > At the beginning of each chapter you will find a summary of the contents,
- > In the header you can see in which chapter and paragraph you are,
- > In the footer you can see to which version the user manual replies,
- > At the end of the user manual you will find an index, with whose help you will quickly find information.

#### Conventions

In the two following charts you will find the conventions used in the user manual regarding utilized spellings and symbols.

Style	Utilization
<b>bold</b>	Blocks, surface elements, window- and dialog names of the software. Accentuation of warnings and advices. <b>[OK]</b> Push buttons in brackets <b>File   Save</b> Notation for menus and menu entries
CANdb++	Legally protected proper names and side notes.
Source code	File name and source code.
Hyperlink	Hyperlinks and references.
<STRG>+<S>	Notation for shortcuts.

Symbol	Utilization
	Here you can obtain supplemental information.
	This symbol calls your attention to warnings.
	Here you can find additional information.
	Here is an example that has been prepared for you.
	Step-by-step instructions provide assistance at these points.
	Instructions on editing files are found at these points.
	This symbol warns you not to edit the specified file.

### 1.1.2 Certification

#### Certified Quality Management System

Vector Informatik GmbH has ISO 9001:2000-12 certification.  
The ISO standard is a globally recognized quality standard.

### 1.1.3 Warranty

#### Restriction of warranty

We reserve the right to change the contents of the documentation and the software without notice. Vector Informatik GmbH assumes no liability for correct contents or damages which are resulted from the usage of the user manual. We are grateful for references to mistakes or for suggestions for improvement to be able to offer you even more efficient products in the future.

### 1.1.4 Support

#### You need support?

You can get through to our hotline at the phone number  
+49 711 80670-200 or by email under **CANdb++-Support**.

### 1.1.5 Registered trademarks

#### Registered trademarks

All trademarks mentioned in this user manual and if necessary third party registered are absolutely subject to the conditions of each valid label right and the rights of particular registered proprietor. All trademarks, trade names or company names are or can be trademarks or registered trademarks of their particular proprietors. All rights which are not expressly allowed, are reserved. If an explicit label of trademarks, which are used in this user manual, fails, should not mean that a name is free of third party rights.

> **Windows, Windows XP, Windows Vista and Windows 7** are trademarks of the Microsoft Corporation.





## 2 Basics

In this chapter you find the following information:

---

2.1	Overview of CANdb++	page 8
2.2	CANdb++ data model	page 10
2.3	Installing CANdb++	page 10
2.4	CANdb++ program window	page 11

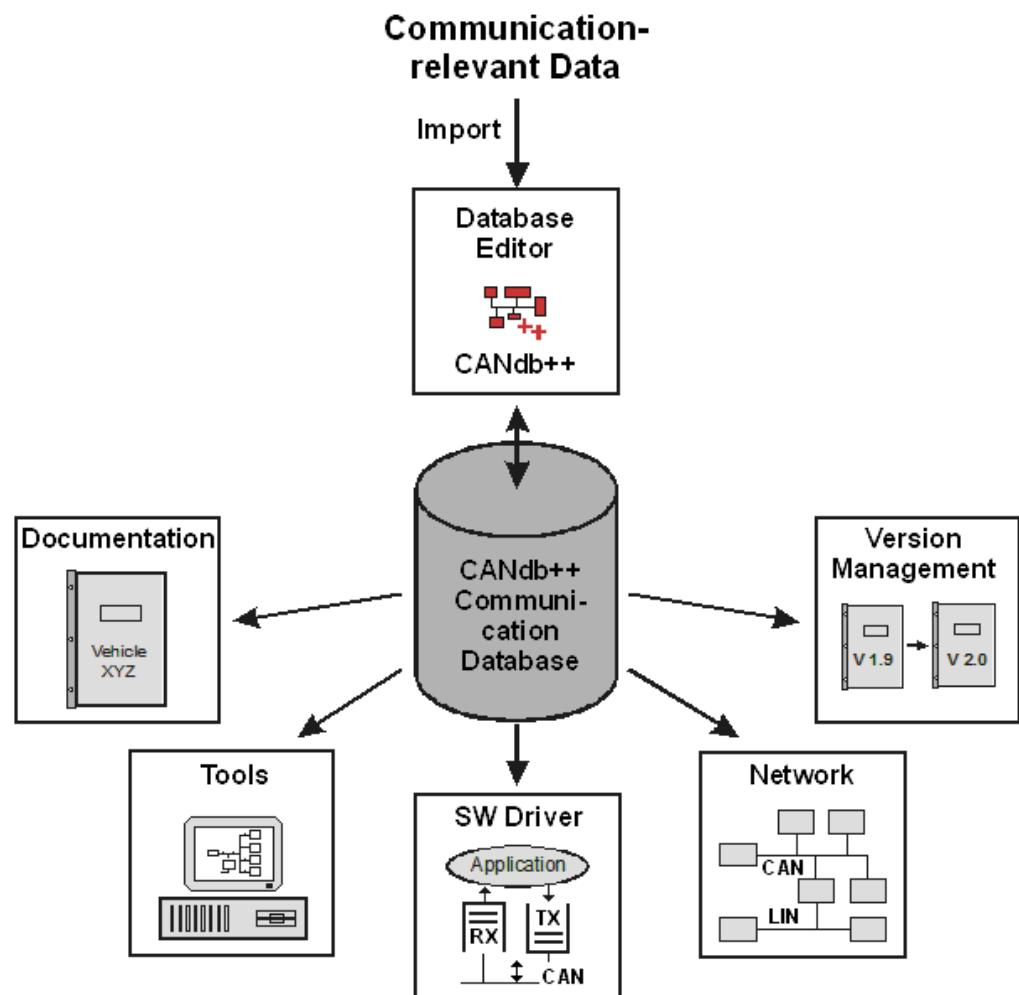
---

## 2.1 Overview of CANdb++

**What is CANdb++?** All communication-relevant data that are processed in a networked CAN bus system as well as their interrelationships are usually administered in a central communications database.

**CANdb++** is a data administration program with which these communication databases can be created and modified in the form of CAN databases.

CANdb++ as central tool for managing communication data



**Program versions** CANdb++ is available in the following program versions:

- > CANdb++
- > CANdb++ Admin.J1939

**Functional features** Comparison of the functional features of **CANdb++** and **CANdb++ Admin**:

Function	CANdb++	CANdb++ Admin
Creating and modifying CANdb++ databases (*.mdc)	—	X
Creating and modifying CANdb network files (*.dbc)	X	X
Creating and modifying user-defined attributes	X	X
Creating and modifying value tables	X	X
Showing the communication matrix	X	X
Consistency check of a CAN database	X	X
Creating and modifying objects of the "Vehicles" object type	—	X
Comparing objects	X	X
Comparing CAN databases Export of differences in text files (*.csv)	—	X
Import objects and CAN databases	—	X
Export objects and CAN databases	X	X
Import of attribute definitions	X	X
Version administration for CAN databases and objects	—	X
Generate reports	—	X
Timing analysis (estimation of bus load)	—	X



**Note:** To achieve Version Administration capabilities with **CANdb++ Admin** it is necessary to have **Microsoft® Visual Source Safe** installed. Moreover, all of the requirements named in chapter 2.3 **Installing CANdb++** (page 10) must also be satisfied.

#### Integration in the tool chain

**CANdb++** is a central tool in the Vector CAN tools toolbox, and it can be started directly from Vector CAN tools such as **CANalyzer**, **CANoe**, **CANape** and **CANscope**. This provides the Vector CAN tools with direct access to communication-relevant data via **CANdb++**.

Communication-relevant data are defined, modified and managed entirely within **CANdb++**; the Vector CAN tools only read-access these data. The communication-relevant data must exist in the form of CANdb network files (\*.dbc) so that the Vector CAN tools can access them.

To also permit the use of data from **CANdb++** databases (\*.mdc) **CANdb++** provides a data export option. **CANdb++** databases can be exported to one or more CANdb network files (\*.dbc). In this way the user can either export the data of an entire **CANdb++** database or only those relevant to a Vehicle, Network or Control Unit.

## 2.2 CANdb++ data model

### Object types

Various object types are available in the Overview Window for modeling the communications structure of a vehicle's CAN bus system.

### Links and relations

Adding a link establishes a connection (Relation) between two objects of different object types. By linking a signal with a message, for example, the user can define the message in which this signal should be transmitted.

Object type of the object to be linked	Object type of the object to which a link can be made						
	Vehicles	Networks	Control Units	Node Groups	Network Nodes	Messages	Signal Groups
Networks	X	—	—	—	—	—	—
Control Units (ECUs)	X	—	—	—	—	—	—
Environment Variables	—	—	X	—	—	—	—
Node Groups	—	X	—	—	—	—	—
Network Nodes	—	X	X	X	—	—	—
Messages	—	—	—	—	X	—	—
Message Signals	—	—	—	—	X	—	X
Signal Groups	—	—	—	—	—	X	—
Signals	—	—	—	—	—	X	—

## 2.3 Installing CANdb++

### Hardware

To install and run CANdb++ your hardware must satisfy the following requirements:

- > Processor: at least Pentium
- > Working memory (RAM): According to the minimum requirements of the respective operating system
- > Operating System: **Windows XP**, **Windows Vista** and **Windows 7**



**Note:** You must have administrator rights to install the software!  
You can switch the language of the menus and dialogs at any time after installation (see 6.1 Vector.ini, page 32).

### Procedure

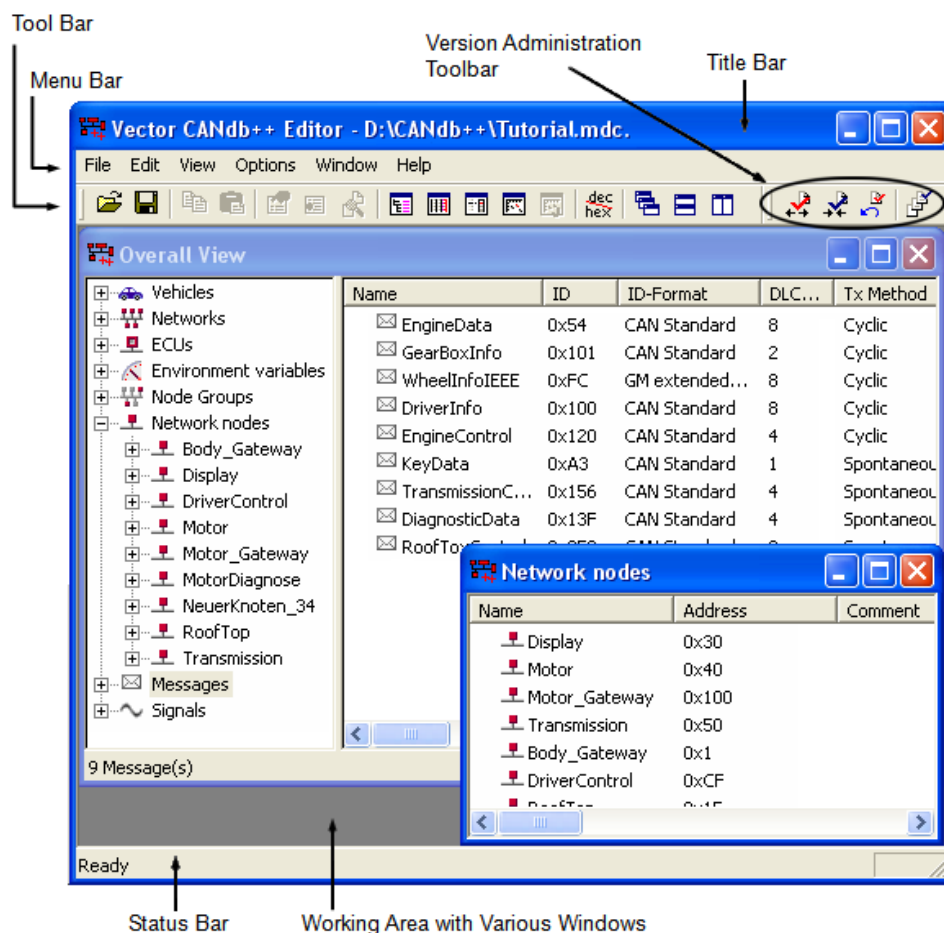


1. Insert the installation CD in your CD drive.
2. Call the installation program SETUP.EXE.
3. Follow the instructions of the installation program.

## 2.4 CANdb++ program window

**Elements of the main window** The following elements are arranged in the CANdb++ program window:

- > Title bar
- > Menu bar
- > Toolbar
- > Version Administration Toolbar  
(The Version Administration Toolbar is only available in the CANdb++ Admin program version!)
- > Working area with various windows
- > Status bar





## 3 Tutorial

In this chapter you find the following information:

---

3.1	Overview	page 14
3.2	Program start	page 14
3.3	Creating a new CAN database	page 15
3.4	Creating and modifying objects	page 16
	Creating new objects	
	Copying existing objects	
	Modifying existing objects	
3.5	Linking objects	page 20
3.6	Showing the communications matrix	page 22
3.7	Value tables	page 23
3.8	Assigning value tables	page 25
3.9	User-defined attributes	page 25
3.10	Modifying the value of an object's user-defined attribute	page 27
3.11	Consistency check	page 28

---

## 3.1 Overview

### Purpose

The purpose of this tutorial is to familiarize you with the **CANdb++** user interface concept and the most important of the **CANdb++** functions.

### Structure

At the beginning of each chapter are problems that you can solve with the help of the explanations that follow.



This symbol identifies the tasks.

### Setup of this tutorial

When creating a new CAN database the following steps are performed in the sequence shown below:



1. Starting the program (section 3.2, page 14).
2. Setting up a new CAN database (section 3.3, page 15).
3. Creating and modifying the necessary objects.
4. Creating new objects (section 3.4.1, page 16).
5. Copying existing objects (section 3.4.2, page 18).
6. Modifying existing objects (section 3.4.3, page 19).
7. Linking the objects (section 3.5, page 20).
8. Displaying the communications matrix (section 3.6, page 22).
9. Creating (section 3.7, page 23) and assigning (section 3.8, page 25) value tables.
10. Creating user-defined attributes (section 3.9, page 25) and modifying values of the user-defined attributes of individual objects (section 3.10, page 27).
11. Performing the consistency check (section 3.11, page 28) and making necessary corrections to the CAN database.


## 3.2 Program start



**Task:** Start CANdb++.

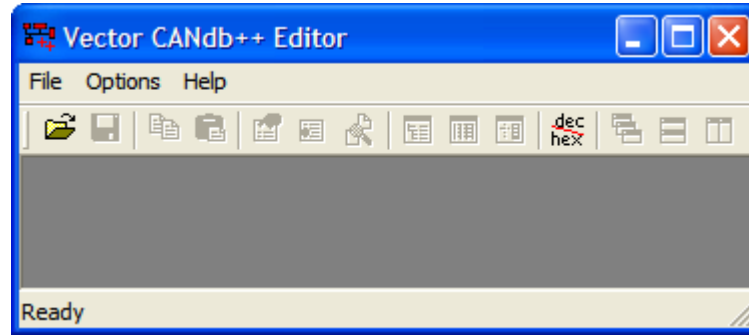


**CANdb++** can be started as follows:

- > Double click the program icon  in the **CANdb++** program group
- > Double click **CANDB.EXE** (e.g. in **Microsoft® Explorer**)
- > Click the program name **CANdb++** in the appropriate sub-folder of the **Microsoft® Windows®** Start menu.

After starting **CANdb++** the **CANdb++** program window appears. The working area is still empty, i.e. it does not contain any windows.





### 3.3 Creating a new CAN database



**Task:** Create a new **CANdb++** database with the name **Tutorial.mdc** or **Tutorial.dbc** in the Data directory of your **CANdb++** installation.



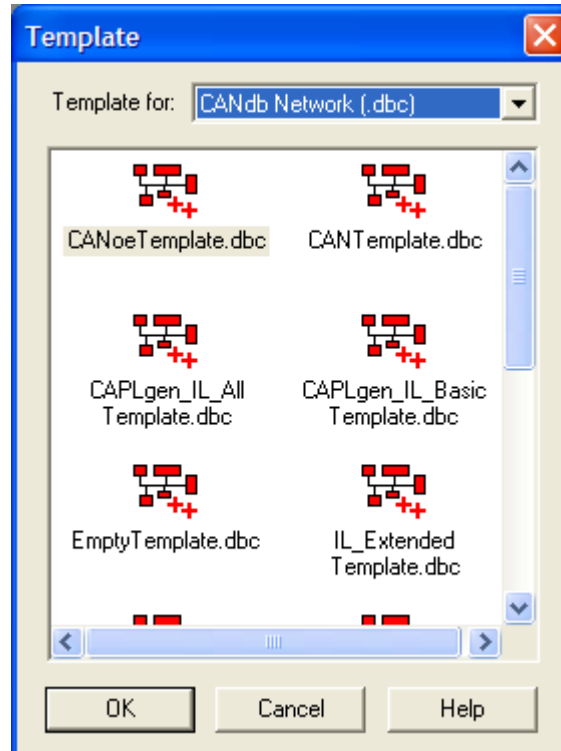
With **CANdb++** you can set up CAN databases of the following types:

- > **CANdb++ databases** (\*.mdc) (**CANdb++ Admin** only)
- > **CANdb network file** (\*.dbc)

Proceed as follows to create a new CAN database:

1. Choose the **File|Create Database** command.

First, the **Template** dialog is opened.



2. There you decide whether to choose a template for a **CANdb** network (.dbc) or for a **CANdb++** database (.mdc).

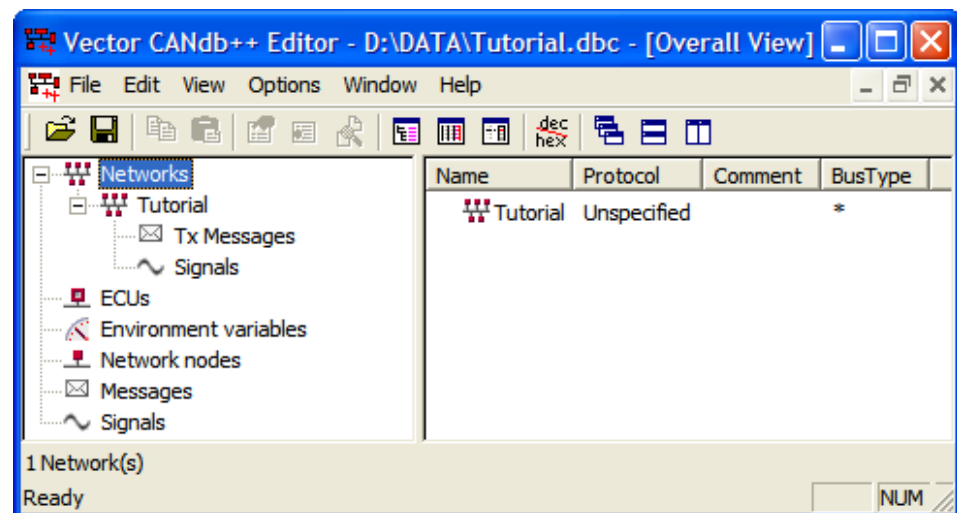
3. Next you can select one of the available templates by a double click or by a single click and choosing **[OK]**.
4. After you have selected a template, the **New Database File** dialog is opened, in which the memory location, file name and file type can be defined for the CAN database to be created.
5. Select the directory in which you wish to save the new CAN database (Data directory of your **CANdb++** installation or **CANdb** network).  
Select the desired file type (e.g. **CANdb++** database) and enter the desired file name (e.g. Tutorial).

It is not necessary to input a file name extension – it will be assigned automatically by **CANdb++** according to the selected file type.

6. Press the **[Save]** button.

The new CAN database (e.g. Tutorial.mdc or Tutorial.dbc) is set up and is shown in the Overview window.

7. Shown on the left side of the Overview window is a hierarchical overview of the available object types.



## 3.4 Creating and modifying objects

### 3.4.1 Creating new objects



**Task:** Create the following objects in the CAN database Tutorial.mdc or Tutorial.dbc:

- > Vehicles (Coupe) (**CANdb++ Admin** only)
- > Networks (Body, PowerTrain) (**CANdb++ Admin** only)
- > Control units (Combi, DriverControl, ECU\_Motor (**CANdb++ Admin** only)
- > Environment variables (EnvTemp)
- > Network nodes (Body\_Gateway, Display, Motor, Motor\_Gateway, SteeringLock)
- > Messages (DriverInfo, EngineControl, KeyData, TransmissionData)
- > Signals (DisplayTemp, GearSelect, RunMode, Information)



Proceed as follows to create a new object, e.g. a signal, in the Overview window:

1. Select the object type for which you wish to set up a new object.  
To do this click the object type on the left side of the Overview Window.

The selected object type gets a color background (Default: Blue).

Appearing on the right side of the Overview Window are all objects of this type that have already been created.

Using this method you would select, for example, the Signals object type if you wish to create a new signal.

2. Choose the **Edit|New** command.

This opens an object dialog.

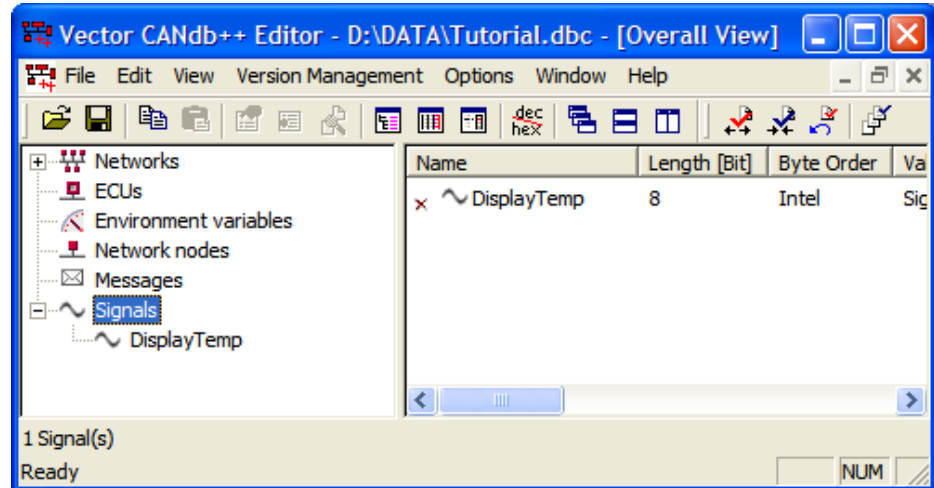
Appearing in the input boxes of the object dialog are suggestions for concrete values of system parameters.

For example, when you have created a new signal the **Signal...** object dialog appears.

3. Change the values of system parameters in the object dialog and press the **[OK]** button.

The newly created object appears in the Overview window.

Appearing in the table columns on the right side of the Overview Window are the values of the object's system parameters.



### 3.4.2 Copying existing objects



**Task:** Create a copy of the signal **RunMode** in the CAN database Tutorial.mdc or Tutorial.dbc and give it the name **OperateMode**.



Proceed as follows to create a copy of an existing object, e.g. of a signal:

1. Select the object that you wish to copy.  
To do this, click the object in the Overview window.  
The selected object has a color background (Default: Blue).
2. Copy the selected object with the **Edit|Copy** command to the Clipboard.
3. Create a copy of the object located in the Clipboard using the **Edit|Paste** command.
4. Modify the copy of the object.



**Note:** After creating a copy of an object this copy is completely independent of the original object!  
Changes to the original object are not mirrored in the copy, and if the changes are desired in the copy they must be made manually afterwards.

### 3.4.3 Modifying existing objects



**Task:** Modify the network nodes in the CAN database Tutorial.mdc or Tutorial.dbc such that each node has a different address.

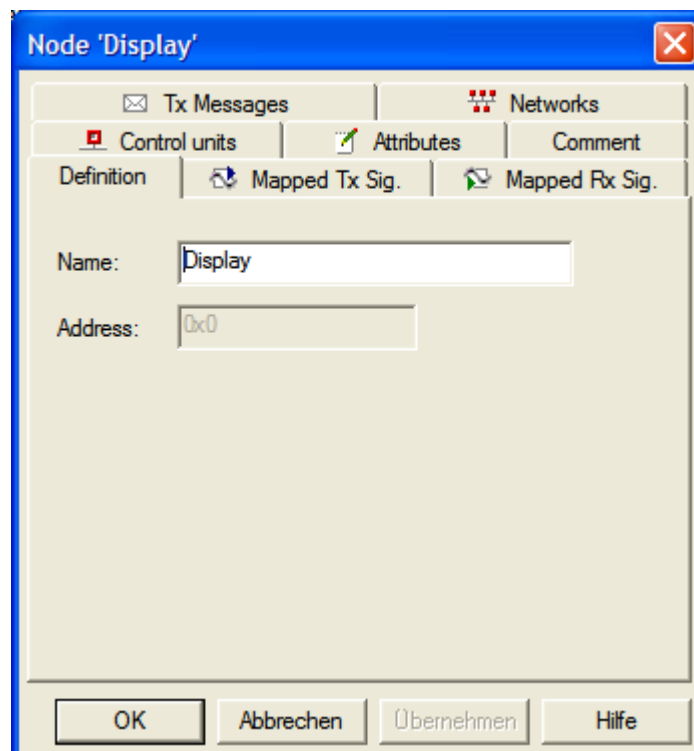
#### Solution 1

#### Modifying an object's parameters in the Object dialog



Proceed as follows to modify the parameters of an object:

1. Select the object to be modified.  
To do this, click the object in the Overview window.  
The selected object gets a color background (Default: Blue).  
In this manner you would select, for example, the network node **Display**.
2. Choose the **Edit|Edit** command.  
This opens an object dialog.  
Appearing in the input boxes of the object dialog are the concrete values for system parameters.  
For example, when you select the network node **Display** the **Node 'Display'** object dialog appears.



3. Modify the system parameters according to your requirements.
4. Press the **[OK]** button.  
The changed system parameters of the object appear in the table on the right side of the Overview window.

## Solution 2

## Modifying an object's parameters in the table in the Overview or List window



Proceed as follows to modify an object's parameters directly in the table in the Overview window or List window:

1. Click the table cell containing the parameter of the object to be modified.  
The corresponding table line gets a color background (Default: Blue).  
The activated cell is shaded with a different color (Default: Light gray).
2. Activate the editing mode for the activated cell by pressing the <F2> key or by clicking the cell again.  
The cell gets a frame and a color background (Default: Blue), and if applicable a list of possible parameters is shown.
3. Change the parameter by entering a different value or selecting a different option from the list that is shown.
4. Press the <Esc> key if you wish to **abort** the procedure, i.e. you wish to leave the parameter value unchanged.  
Press the <Return> key if you wish to **accept** the changed value.



**Note:** You can switch the selection of the active cell within the table using the arrow cursor keys <←>, <→>, <↑> and <↓>.

### 3.5 Linking objects



**Task:** Link the following objects in the CAN database Tutorial.mdc or Tutorial.dbc:

- > Signals with messages  
(GearSelect-TransmissionData, DisplayTemp-DriverInfo, Information-DriverInfo, OperateMode-EngineControl, RunMode-KeyData)
- > Messages with network nodes  
(TransmissionData-Display, EngineControl-Motor\_Gateway, KeyData-SteeringLock, DriverInfo-Body\_Gateway)
- > Message signals with network nodes  
(DisplayTemp-Display, GearSelect-Body\_Gateway, OperateMode-Motor, RunMode-Body\_Gateway)
- > Network nodes with node groups  
(Motor-PowerTrain\_Base, Motor\_Gateway-PowerTrain\_Base)  
(CANdb++ Admin only)
- > Network nodes with control units  
(Body\_Gateway-Combi, Display-Combi, Motor-ECU\_Motor, Motor\_Gateway-Combi, SteeringLock-DriverControl)  
(CANdb++ Admin only)
- > Network nodes with networks  
(Body\_Gateway-Body, SteeringLock-Body, Display-Body, Motor-PowerTrain, Motor\_Gateway-PowerTrain)  
(CANdb++ Admin only)
- > Node groups with networks  
(PowerTrain\_Base-PowerTrain)  
(CANdb++ Admin only)

- > Control units with vehicles  
(Combi-Coupe, DriverControl-Coupe, ECU\_Motor-Coupe)  
(CANdb++ Admin only)
- > Networks with vehicles  
(Body-Coupe, PowerTrain-Coupe)  
(CANdb++ Admin only)

By linking two objects of different object types you can establish a connection (Relation) between these two objects.


For example, by linking a signal with a message you can define the message in which this signal should be transmitted.

### Solution 1


#### Linking by 'Drag and Drop'



Proceed as follows to link two objects by 'Drag and Drop':

1. On the left side of the Overview window show the structure level that contains the object with which a link should be made.  
To do this, click the  symbol in front of the object type for this object.

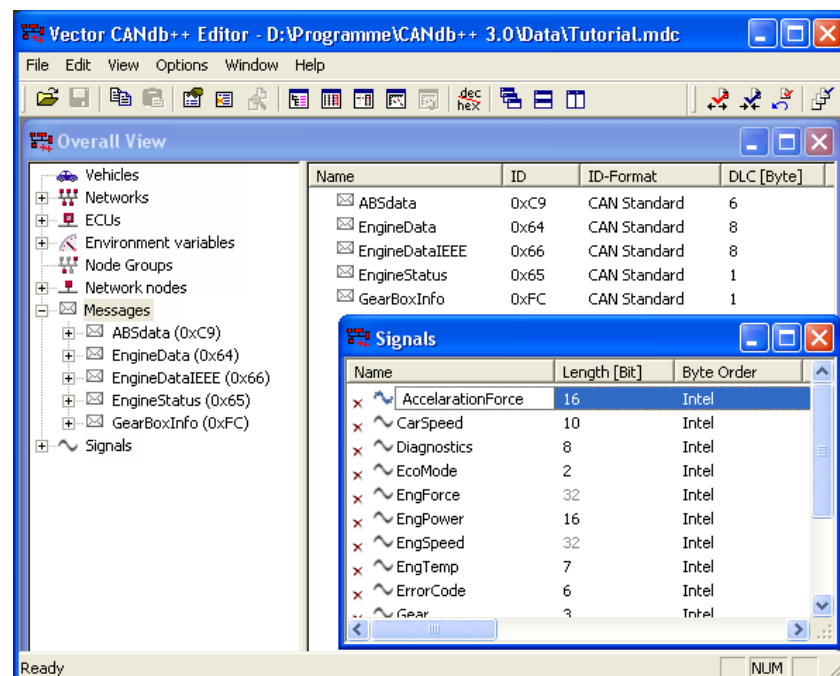
The subordinate structure level is then shown.

For example, if you wish to link a signal to a message click the  symbol in front of the **Messages** object type.

2. Open a List Window for the object type of the object you wish to link.  
To do this, choose the **View|List...** command for the relevant object type.

For example, if you wish to link a signal to a message choose the **List-Signals** command to open a List window with signals.

If necessary change the size of the Overview window and the List window so that the relevant areas are visible in both windows.



3. In the List window select the object that you wish to link.

4. Link the two objects.

To do this, drag the selected object from the List window to the Overview window while holding down the mouse button. Do not release the mouse button until the mouse pointer is located above the object in the Overview window with which the link is to be made.



When the mouse pointer assumes this shape the objects can be linked.



When the mouse pointer assumes this shape the objects cannot be linked.

- > The link is automatically added after releasing the mouse button.
- > The linked object appears in the Overview window below the object with which it was linked.
- > The object symbol of the linked object changes to display the link.

For example, the object symbol of a linked signal, i.e. a message signal would appear as follows:

## Solution 2

### Linking by menu commands



As an alternative to this you could link objects by menu commands as described below:

1. In the Overview window select the object that you wish to link.
2. Copy the selected object to the Clipboard with the **Edit|Copy** command.
3. In the Overview window select the object you would like to link to the object just copied.
4. Add the new link with the **Edit|Add link** command.

The link is then added.

## 3.6 Showing the communications matrix



**Task:** Open the Communications Matrix window that contains the communications matrix of the CAN database Tutorial.mdc or Tutorial.dbc.



A communications matrix shows the communications relationships between signals, messages and network nodes in table format.

Proceed as follows to have the communications matrix displayed:

1. Choose the **View|Communications matrix** command.

This automatically opens the Communications Matrix window with the communications matrix of the active CAN database.

Communication Matrix: All networks			
Signals/Node	DisplayEngine	Gateway	Motor
~ Status	EngineStatus (0x65)		<Tx> EngineStatus (0x65)
~ ShiftRequest			<Tx> GearBoxInfo (0xFC)
~ IdleRunning	EngineData (0x64)		<Tx> EngineData (0x64)
~ GearLock	AB5data (0xC9)	AB5data (0xC9)	<Tx> AB5data (0xC9)
~ Gear	GearBoxInfo (0xFC)		<Tx> GearBoxInfo (0xFC)
~ ErrorCode	EngineStatus (0x65)		<Tx> EngineStatus (0x65)



In the Communications Matrix window the signals are arranged by lines and the network nodes by columns. The table boxes with the signal names have gray shading. The messages shown in the remaining table boxes are the messages in which the signal is transmitted or received by the particular network node.

Transmitted messages are identified as follows:

- > Message name in blue font
- > "<TX>" in front of the message name



**Note:** When the Communications Matrix window remains open it is automatically updated with each modification to the CAN database.

### 3.7 Value tables



**Task:** In the CAN database Tutorial.mdc or Tutorial.dbc create the value table **Colors** which assigns the concrete values 0, 1 and 2 to the symbolic identifiers 'red', 'yellow' and 'green'.

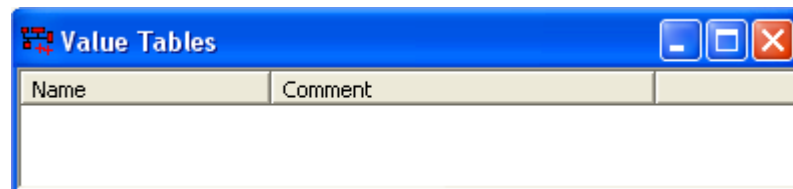


In a value table individual signal values or environment variable values can be assigned to symbolic identifiers.

Proceed as follows to create a value table:

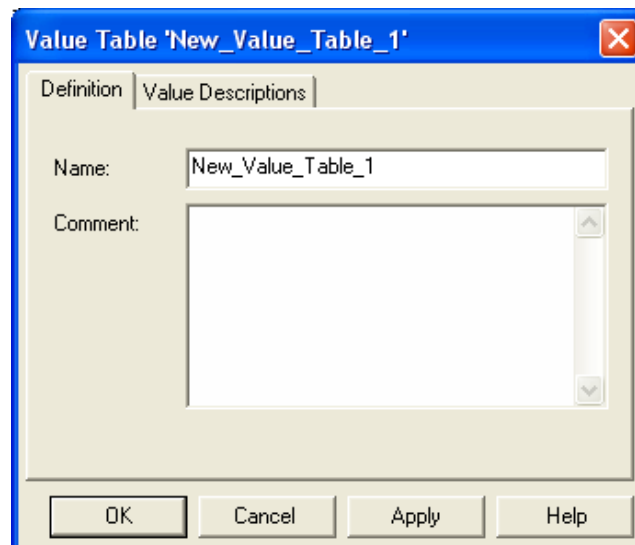
1. Choose the **View|Value Tables** command.

The Value Tables window is opened.

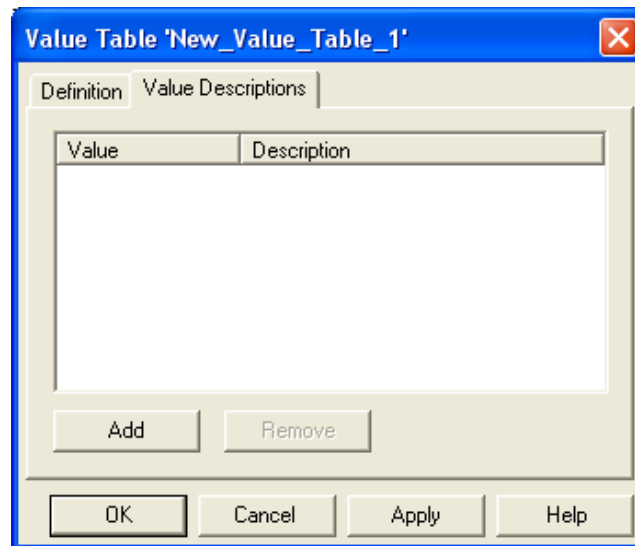


2. Choose the **Edit|New** command.

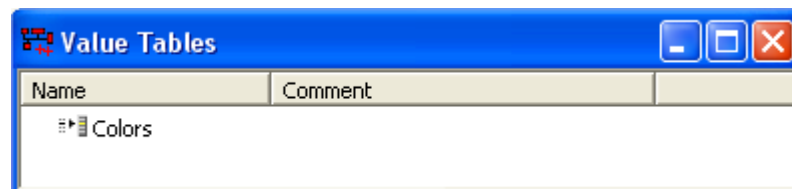
The **Value Table** object dialog is opened.



3. Change the name of the value table.  
Enter the new name in the **Name** input box.
4. Activate the **Value Description** page by clicking its tab.



5. Press the **[Add]** button.  
A new line is added to the table.  
Appearing in the **Value** column is a suggested concrete value for the signal or environment variable to which a symbolic identifier should be assigned.  
Appearing in the **Description** column is a suggested text for this symbolic identifier.
6. Click the cell whose text you wish to change.  
The cell is selected by a frame and can now be edited.
7. Change the values and symbolic identifiers.
8. Press the <Esc> key if you wish to **abort** the procedure, i.e. you wish to leave the value or symbolic identifier unchanged.  
Press the <Return> key if you wish to **accept** the changed value or changed identifier.
9. Press the **[OK]** button.  
The newly created value table appears in the Value Tables window.



### 3.8 Assigning value tables



**Task:** In the CAN database Tutorial.mdc or Tutorial.dbc assign the **Colors** value table to the **Information** signal.



The value table must be assigned to a signal or environment variable so that the symbolic identifiers that are assigned to individual values in a value table will indeed be assigned to the signal or environment variable values.

Proceed as follows to assign a value table to a signal or environment variable:

1. Select the signal or environment variable to which the value table should be assigned.

To do this, click the specific object in the Overview window or List window.

2. Choose the **Edit|Edit** command.

This opens an object dialog.

For example, if you select the **Information** signal the **Signal 'Information'** object dialog appears.

3. In the **Value Table** list box select the value table that you wish to assign to the signal or environment variable.
4. Press the **[OK]** button.

### 3.9 User-defined attributes



**Task:** In the CAN database Tutorial.mdc or Tutorial.dbc create the user-defined attribute **Release** for Vehicles. Concrete values available for the **Release** attribute should be: 'pending', 'confirmed' and 'denied'.

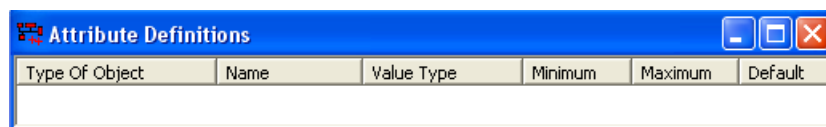


In addition to system parameters that must be defined when creating an object, objects can also be assigned user-defined attributes.

Proceed as follows to create a user-defined attribute:

1. Choose the **View|Attribute Definitions** command.

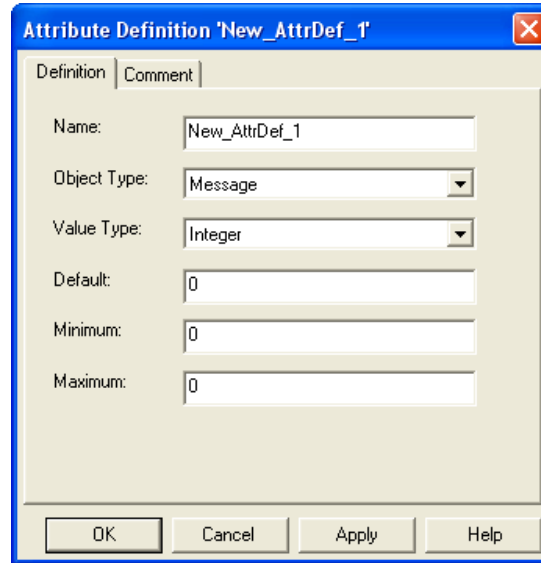
This opens the Attribute Definitions window.



2. Choose the **Edit|New** command.

The Attribute Definition object dialog is opened.

Appearing in the input boxes are suggested concrete values for parameters of the user-defined attribute.



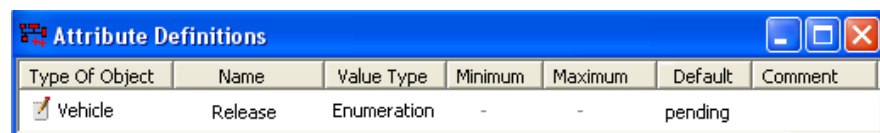
The dialog box 'Attribute Definition 'New\_AttrDef\_1'' has two tabs: 'Definition' and 'Comment'. The 'Definition' tab is active. It contains the following fields:

- Name: New\_AttrDef\_1
- Object Type: Message (dropdown)
- Value Type: Integer (dropdown)
- Default: 0
- Minimum: 0
- Maximum: 0

At the bottom are buttons for OK, Cancel, Apply, and Help.

3. Change the parameters of the user-defined attribute.  
Enter the new name and select the object type and value type.
4. To provide concrete texts as values you must select the **Enumeration** value type.  
These texts can then be entered in the **Value Range** input box that then appears.  
The value in the **Default** input box is assigned as the default value of the user-defined attribute for each object of the selected object type.
5. Press the [OK] button.

The newly created user-defined attribute now appears in the Attribute Definitions window.



Type Of Object	Name	Value Type	Minimum	Maximum	Default	Comment
Vehicle	Release	Enumeration	-	-	pending	

Like system parameters, the values of user-defined attributes also appear in the columns on the right side of the Overview window or in the corresponding List windows.



**Note:** User-defined attributes of objects which still have their default values are identified by a \* after the attribute value on the right side of the Overview window.

Overall View

Vehicles

Networks

ECUs

Environment variables

Groups

Network nodes

Messages

ABSDATA (0xC9)


EngineData (0x64)

EngineDataIEEE (0x66)

EngineStatus (0x65)

GearBoxInfo (0xFC)

Signals

Name	Comment	Release
 Coupe		pending*

1 Vehicle(s)

### 3.10 Modifying the value of an object's user-defined attribute



**Task:** In the CAN database Tutorial.mdc or Tutorial.dbc set the user-defined attribute Release for the Vehicle object Coupe to the value **denied**.



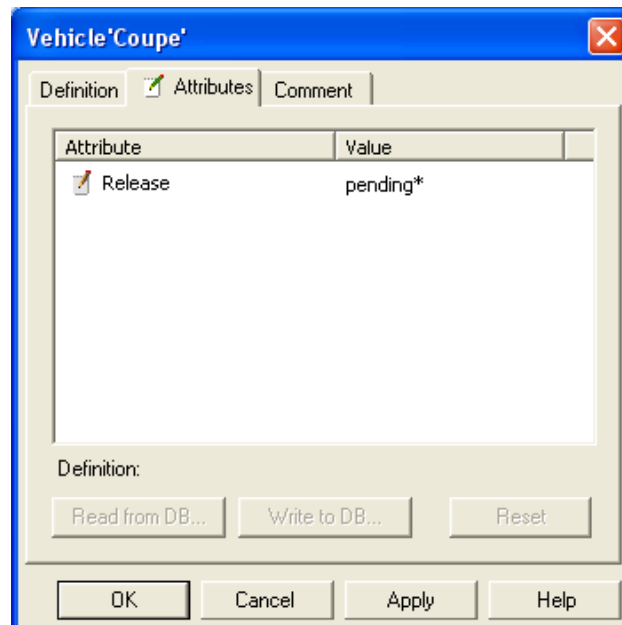
Like system parameters, the values of user-defined attributes can also be modified at any time.

Proceed as follows to modify the value of an object's user-defined attribute:

#### Solution 1

#### Modifying the values in the Object dialog

1. Select the object whose user-defined attribute value you wish to modify.
2. Choose the **Edit|Edit** command.  
The Object dialog is opened.
3. Activate the **Attribute** page by clicking its tab.



4. Activate the user-defined attribute whose value you wish to change by clicking the appropriate table cell.  
The corresponding table line is highlighted in color (Default: Blue).
5. Click the value of the attribute.
6. A frame appears around the cell and if applicable the available attribute values are shown.
7. Enter the desired value or select it from the list.
8. Press the <Esc> key if you wish to **abort** the procedure, i.e. you wish to leave the value unchanged.  
Press the <Return> key if you wish to **accept** the changed value.
9. Press the **[OK]** button.

The changed value of the object's user-defined attribute appears in the table on the right side of the Overview window.

## Solution 2

**Modifying the values in the table in the Overview or List window**

1. Click the table cell containing the object's user-defined attribute to be modified.  
The corresponding line in the table is highlighted in color (Default: Blue).  
The activated cell is shaded with a different color (Default: Light gray).
2. Activate the editing mode for the activated cell by pressing the <F2> key or clicking the cell again.  
A frame appears around the cell and it is shaded in color (Default: Blue); if applicable a list of possible values is shown.
3. Change the attribute value by entering a different value or selecting a different option from the list shown.
4. Press the <Esc> key if you wish to **abort** the procedure, i.e. you wish to leave the parameter value unchanged.  
Press the <Return> key if you wish to **accept** the changed value.

### 3.11 Consistency check



**Task:** Execute an automatic consistency check for the CAN database Tutorial.mdc or Tutorial.dbc with **CANdb++** and correct any inconsistencies that are found.



To determine whether the objects of a CAN database and their interrelationships are consistent with one another, an automatic consistency check can be performed with **CANdb++**.

Proceed as follows to perform a consistency check:

1. Choose the **File|Consistency check** command to start the automatic consistency check.

The results of the consistency check are displayed in a Consistency Check window.

- > If **CANdb++** does not find any inconsistencies in the CAN database, then there will be no entries in the Consistency Check window.
- > However, if inconsistencies are found in the CAN database they will be listed in the Consistency Check window.

Object Status	Type Of Object	Name	Note	Explanation
✗ Info	Signal	AccelerationForce	No Sender Node	This signal is not transmitt...
✗ Info	Signal	CarSpeed	No Sender Node	This signal is not transmitt...
✗ Info	Signal	Diagnostics	No Sender Node	This signal is not transmitt...
⚠ Warning	Node	DisplayEngine	Multiple Warnings.	-



**Note:** When the Consistency Check window is open it is updated automatically with each modification of the CAN database.

## 4 Version administration

In this chapter you find the following information:

4.1	Preconditions for version administration capability	page 30
4.2	Version administration for CAN databases and objects	page 30



**Note:** Version Administration for CAN databases and objects is only available in the **CANdb++ Admin** program version!

## 4.1 Preconditions for version administration capability

Preconditions	<p>To perform version administration the following preconditions must be satisfied:</p> <ul style="list-style-type: none"><li>&gt; Microsoft® Visual Source Safe must be installed.</li><li>&gt; The version administration archive in which the CAN database or object should be stored must have been created and must be accessible to the user.</li><li>&gt; The configuration file SRCSAFE.INI must exist for the CAN database archive.</li><li>&gt; The user must have read and write rights for all folders in which data are to be stored.</li></ul>
Configuration	<p>When these preconditions are satisfied <b>CANdb++ Admin</b> automatically establishes the connection to the Microsoft® Visual Source Safe archive, whose configuration file SRCSAFE.INI is entered in the <b>Archive Path</b> input box of the <b>Versioning page (Settings dialog)</b>. Use the <b>Options Settings</b> command to open the <b>Settings</b> dialog.</p>

## 4.2 Version administration for CAN databases and objects

Accessibility	<p><b>CANdb++ Admin</b> permits direct access to CAN databases and objects that have been saved in Microsoft® Visual Source Safe.</p>
Functional range	<p>The following actions can be executed directly, i.e. without leaving <b>CANdb++ Admin</b>:</p> <ul style="list-style-type: none"><li>&gt; Opening a database that was saved in the version administration archive</li><li>&gt; Reserving (Checking-out) a database</li><li>&gt; Versioning (Checking-in) a database</li><li>&gt; Displaying the version history of a database</li><li>&gt; Comparing the different versions of a database</li><li>&gt; Displaying the reservation status of a database</li><li>&gt; Versioning (Checking-in) an object</li><li>&gt; Displaying the version history of an object</li><li>&gt; Comparing the different versions of an object</li></ul>



## 5 Appendix A: File name extensions

### What is it?

The file name extension is understood to consist of the three characters located after the dot following the file name.

The file name extension identifies the file type.

<b>CHM</b>	Help file (Microsoft Compiled HTML Help)
<b>CSV</b>	Text file with <b>C</b> omma/ <b>C</b> haracter <b>S</b> eparated <b>V</b> alues
<b>DBC</b>	CANdb network file ( <b>D</b> ata <b>B</b> ase for <b>C</b> AN)
<b>DLL</b>	Run-time library ( <b>D</b> ynamic <b>L</b> ink <b>L</b> ibrary)
<b>EXE</b>	<b>E</b> xecutable program
<b>INI</b>	File with configuration settings.
<b>MDC</b>	<b>CANdb++</b> CAN database
<b>RPT</b>	<b>Crystal Report</b> file
<b>TXT</b>	Text file

## 6 Appendix B: INI files

### Location of the INI files

The INI files are located in the EXEC32 directory of your **CANdb++** installation and contain configuration options for **CANdb++**.

To make changes to options it is possible to edit the INI files with any ASCII editor.



**Note:** You should exit **CANdb++** before editing the INI files.  
The changed options in the INI files do not take effect until **CANdb++** is restarted.

### 6.1 Vector.ini

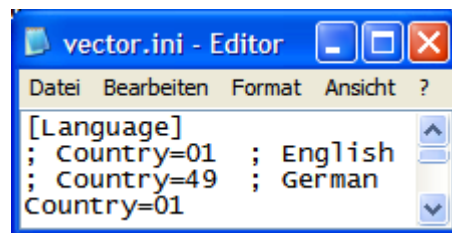
#### Settings for...

In VECTOR.INI you can set options for the language of the menus and dialogs.



Proceed as follows to change the language of the menus and dialogs:

1. Open the file VECTOR.INI with an ASCII editor.



2. In the [Language] section replace the language identifier entered in the Country= line by the identifier of the desired language.

For example, English is configured in the following entry:

```
[Language]
Country=01
```

Please refer to the comment lines (identified by // ) of the [Language] section for the available languages.

3. Save the VECTOR.INI file and close the editor.

## 6.2 CANdb.ini

### Settings for...

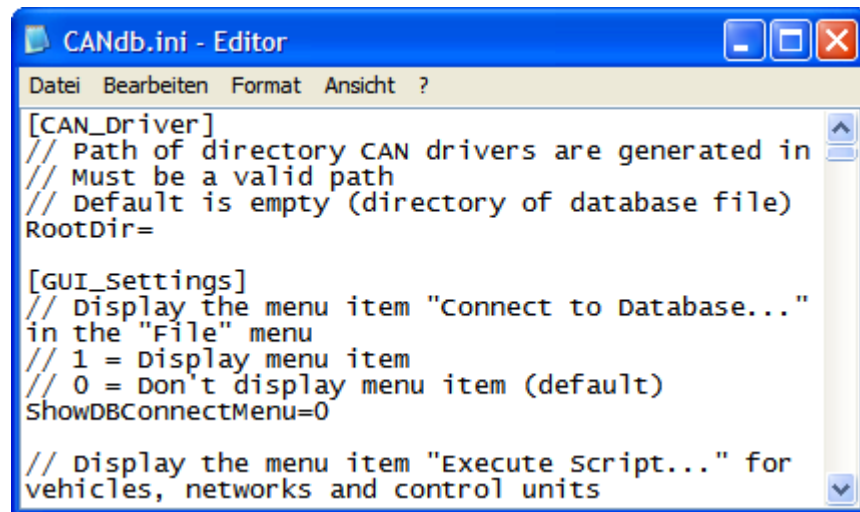
The following options can be configured for **CANdb++** in CANdb.INI:

- > Root directory for generated CAN drivers  
(section [CAN\_Driver], line "RootDir=")
- > Display format for attributes of type Integer  
(section [GUI\_Settings], line "DisplayIntAttrsAlwaysDecimal=")
- > Format to display signals with motorola byte ordering  
(section [Format], line "MotorolaFormat=")
- > Indexing of bits in layout dialog of messages  
(section [Format], line "UseInvertedLayoutIndexing=")



Proceed as follows to make changes to these options:

1. Open the CANdb.INI file with an ASCII editor.



```

CANdb.ini - Editor
Datei Bearbeiten Format Ansicht ?

[CAN_Driver]
// Path of directory CAN drivers are generated in
// Must be a valid path
// Default is empty (directory of database file)
RootDir=

[GUI_Settings]
// Display the menu item "Connect to Database..."
// in the "File" menu
// 1 = Display menu item
// 0 = Don't display menu item (default)
ShowDBConnectMenu=0

// Display the menu item "Execute Script..." for
// vehicles, networks and control units
```

2. If you want to set the root directory for generated CAN drivers, enter the desired root directory in the line `RootDir=` of the section `[CAN_Driver]`.

When generating CAN drivers, sub-directories are created automatically in the root directory. The generated CAN drivers are stored in these sub-directories.

3. Store the file CANdb.INI and close the editor.

If **CANdb++** was opened during editing, you have to exit **CANdb++** and open it again to work with the new settings.

## 7 Appendix C: Address table

Vector Informatik GmbH	Vector Informatik GmbH Ingersheimer Str. 24 D-70499 Stuttgart Phone: +49 (711) 80670-0 Fax: +49 (711) 80670-111 <a href="mailto:info@de.vector.com">mailto:info@de.vector.com</a> <a href="http://www.vector.com">http://www.vector.com</a>
Vector CANtech, Inc.	Vector CANtech, Inc. Suite 550 39500 Orchard Hill Place USA-Novi, Mi 48375 Phone: +1 (248) 449 9290 Fax: +1 (248) 449 9704 <a href="mailto:info@us.vector.com">mailto:info@us.vector.com</a> <a href="http://www.vector.com">http://www.vector.com</a>
Vector France SAS	Vector France SAS 168, Boulevard Camélinat F-92240 Malakoff Phone: +33 (1) 4231 4000 Fax: +33 (1) 4231 4009 <a href="mailto:information@fr.vector.com">mailto:information@fr.vector.com</a> <a href="http://www.vector.com">http://www.vector.com</a>
Vector GB Ltd.	Vector GB Ltd. Rhodium, Central Boulevard Blythe Valley Park Solihull, Birmingham West Midlands B90 8AS United Kingdom Phone: +44 (121) 50681 50 Fax: +44 (121) 50681 66 <a href="mailto:info@uk.vector.com">mailto:info@uk.vector.com</a> <a href="http://www.vector.com">http://www.vector.com</a>

**Vector Japan Co., Ltd.**  
Vector Japan Co., Ltd.  
Seafort Square Center Bld. 18F  
2-3-12, Higashi-shinagawa, Shinagawa-ku  
J-140-0002 Tokyo  
Phone: +81 3 (5769) 7800  
Fax: +81 3 (5769) 6975  
<mailto:info@jp.vector.com>  
<http://www.vector.com>

**Vector Korea IT Inc.**  
Vector Korea IT Inc.  
# 1406 Mario Tower  
222-12 Guro-dong, Guro-gu  
Seoul 152-848  
Republic of Korea  
Phone: +82(0)2 2028 0600  
Fax: +82(0)2 8070601  
<mailto:info@kr.vector.com>  
<http://www.vector.com>

**VecScan AB**  
VecScan AB  
Theres Svenssons Gata 9  
SE-417 55 Göteborg  
Phone: +46 (31) 76476-00  
Fax: +46 (31) 76476-19  
<mailto:info@se.vector.com>  
<http://www.vector.com>

**Vector Informatik India Prv. Ltd.**  
Vector Informatik India Private Limited  
4/1/1/1 Sutar Icon  
Sus Road  
Pashan  
Pune 411021  
India  
Phone: +91.9673.336575  
Fax: —  
<mailto:info@in.vector.com>  
<http://www.vector.com>



## 8 Index

### A

Address table .....	34
Attributes .....	25

### B

Basics .....	7
--------------	---

### C

CANdb.ini .....	33
Communications matrix .....	22
Consistency check .....	28
Conventions .....	4

### E

Extensions .....	31
------------------	----

### F

File name extensions .....	31
Functional features .....	9

### I

INI files .....	32
Installing CANdb++ .....	10

### L

Linking objects .....	20
Links .....	10

### O

Object types .....	10
--------------------	----

### P

Program versions .....	8
Program window .....	11

### R

Registered trademarks .....	5
Relations .....	10

### S

Support .....	5
---------------	---

### T

Tutorial .....	13
----------------	----

### U

User-defined attributes .....	25
-------------------------------	----

### V

Value tables .....	23, 25
Vector.ini .....	32
Version administration .....	29

### W

Warranty .....	5
----------------	---

## **Get more Information!**

### **Visit our Website for:**

- > News
- > Products
- > Demo Software
- > Support
- > Training Classes
- > Addresses

**[www.vector.com](http://www.vector.com)**