

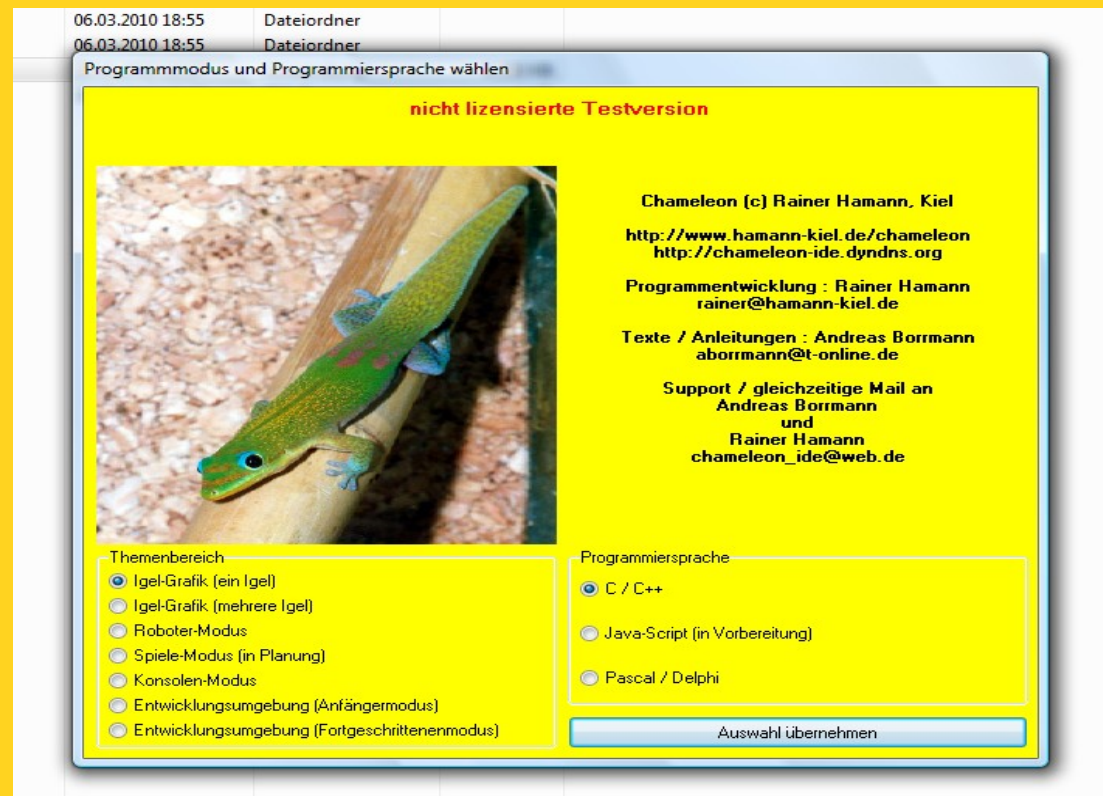
# Chameleon

ist eine Programmierumgebung,

die von Schülern auch ohne Vorkenntnisse sofort genutzt werden kann und

die über mehrere Zwischenstufen auf den Einsatz großer Entwicklungsumgebungen vorbereitet.

Stets stehen die Inhalte der Informatik im Vordergrund.



# Gründe für die Entwicklung

- zuerst einfache Entwicklungsumgebungen wie
  - Logo
  - Kara, Hamster oder ähnliche
- dann riesiger Sprung auf professionelle IDEs
  - Delphi
  - Eclipse
  - Netbeans
  - Visual C / C#

# Was muss besser werden?

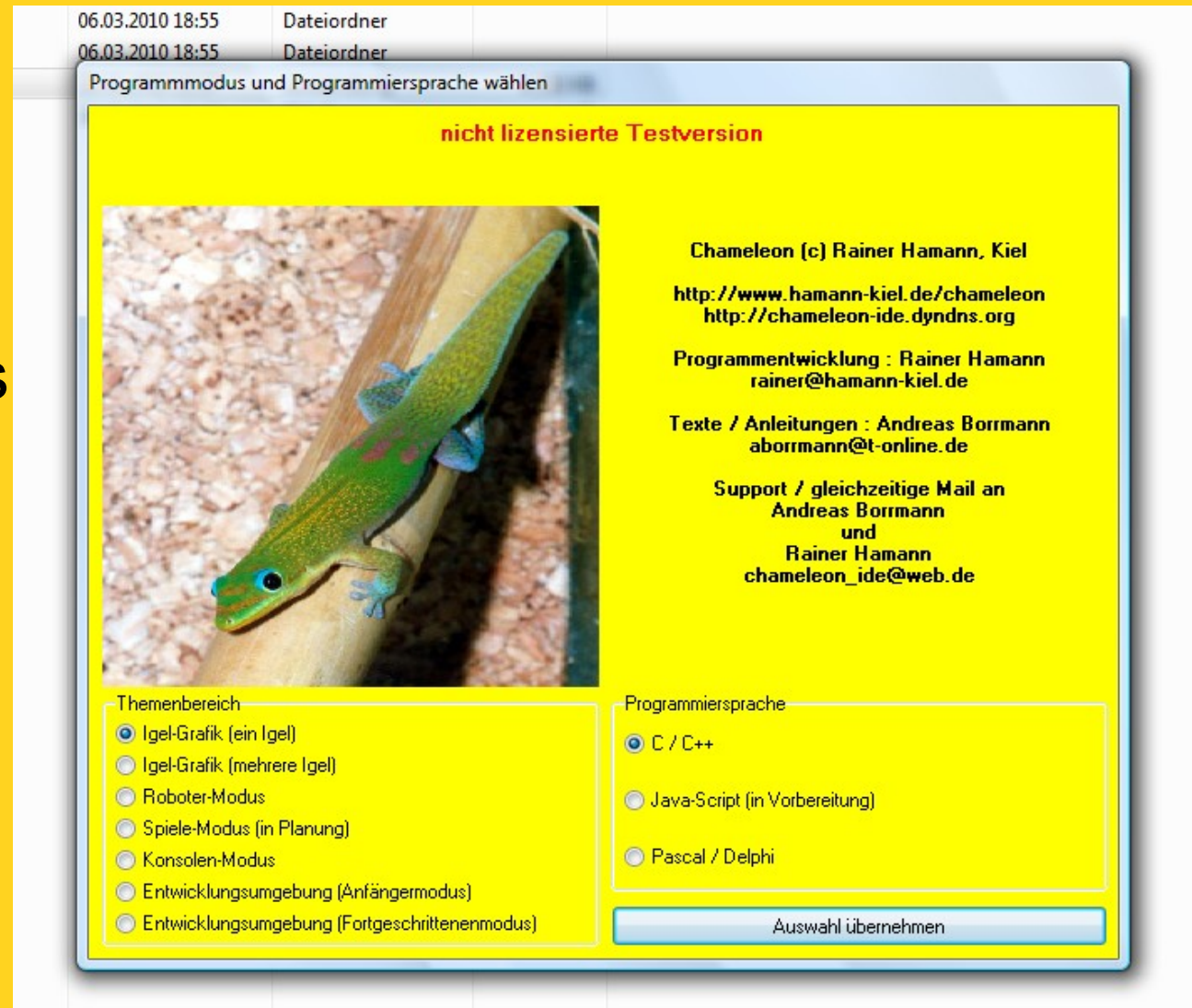
- sanfterer Anstieg der Lernkurve
  - Unterstützung bei der Quelltexterstellung
    - Nassi-Schneidermann-Diagramm
  - zunehmend eigene Quelltexteingaben
    - Unterstützung durch Wizards
  - anfangs ohne Objektorientierung
  - Anwendung von / Umgang mit Objekten
  - Bearbeitung von Objekten
  - vollständig objektorientiert

# Problemstellungen

- Schwerpunkt Algorithmen
  - an das Alter angepasste Probleme
  - interessante Aufgabenstellungen
  - zunehmende Komplexität
  - Entwicklung eigener Lösungsverfahren
    - Förderung der Methodenkompetenz
    - und der Selbstkompetenz

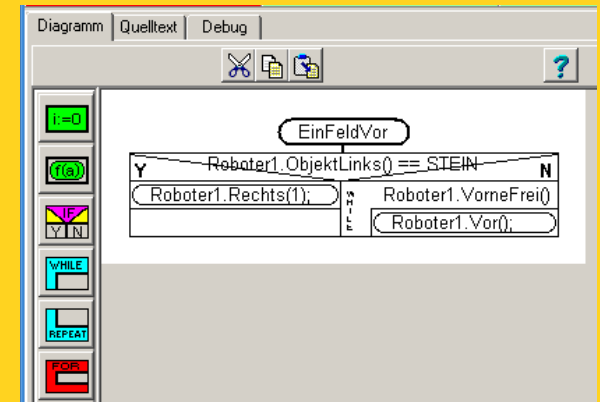
# Was bietet Chameleon

- Problemwahl
  - Igel / Turtle
  - Roboter
  - Spiele-Modus
  - IDE einfach
  - IDE komplex
- Sprachwahl
  - C++
  - Delphi
  - Java



# Was hilft den Schülern?

- einheitliche Programmierumgebung für alle Anforderungsbereiche (Wiedererkennung)
- programmieren in Diagrammen
- viele Wizards
- schülergerechte Hilfen und Aufgabenstellungen



The screenshot shows a dialog box for configuring parameters. It has three rows, each with a 'Name' field, a 'Typ' dropdown menu, and a 'Referenz-Param' checkbox. The first row has 'Name' and 'Typ' set to 'int'. The second row has 'Name' and 'Typ' set to 'int' and 'Referenz-Param' checked. The third row has 'Name' and 'Typ' set to 'int' and 'Referenz-Param' checked. A dropdown menu on the right shows 'char', 'string', and 'bool' as options.

# Igel-Grafik

Default - Wine desktop

C++-Turtle

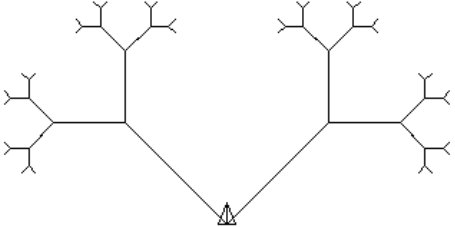
neues Programm Programm laden Programm speichern Quelltext einrücken Wischen Programm starten Stop Einzelschrittmodus Einzelschritt

Funktions-Deklaration einfügen Variablen-Deklaration einfügen Bedingung einfügen Schleifen Funktionsaufrufe

Diagramm Quelltext Debug

igel.pc

```
1 using igelmodus;
2 void v(float a)
3 {
4     if (a > 5) {
5         Links(45);
6         Vor(a);
7         v(a/2);
8         Zurueck(a);
9         Rechts(90);
10        Vor(a);
11        v(a/2);
12        Zurueck(a);
13        Links(45);
14    }
15 }
16 {
17     v(100);
18 }
```



1 : 1

# Multi-Igel-Modus

The screenshot displays the Chameleon-Turtle IDE interface. The main window shows a C++ source code file with the following content:

```
1 using igelmodus. igel;
2
3 TIgel i[4];
4 void InitIgel() {
5     i[0] = Igel1;
6     i[1] = Igel2;
7     i[2] = Igel3;
8     i[3] = Igel4;
9 }
10 void ZumStart() {
11     for (int n=0; n<4; n++) {
12         i[n].OhneStift();
13         i[n].GeheNach(0, -300);
14         i[n].MitStift();
15     }
16 }
17 void Rennen() {
18     do {
19         int nr = random(0, 4);
20         int schritte = random(0, 3);
21         i[nr].Vor(schritte);
22     }
23     while (i[nr].WoIgelY() < 300);
24 }
25 { // hier beginnt die Programmausführung
26     InitIgel();
27     ZumStart();
28     Rennen();
29 }
30
```

The IDE interface includes a menu bar with options like 'neues Programm', 'Programm laden', 'Programm speichern', 'Drucken', 'Bild speichern', 'Quelltext einrücken', 'Wischen', and 'Syntaxhilfe'. Below the menu bar are buttons for 'Programm starten', 'Stop', 'Weiter', and 'Einzelschritt'. A toolbar contains buttons for 'Funktionsdeklaration einfügen', 'Variablendeklaration einfügen', 'Verzweigung einfügen', 'Schleifen', and 'Funktionsaufrufe'. The main workspace is divided into a code editor on the left and a graphical area on the right. The graphical area contains four vertical lines, each with a small upward-pointing triangle at the bottom, representing the turtles. At the bottom of the IDE, there are two panels: 'Variablen' and 'Funktionsaufrufe und Parameter'. The 'Funktionsaufrufe und Parameter' panel shows the function 'Rennen()' being called. The Windows taskbar at the bottom shows the system tray with the time 17:24 and the language set to DE.

# Robotermodus

The screenshot displays a C++ IDE window titled "Default - Wine desktop" with a sub-window "C++-Roboter". The interface includes a menu bar with options like "neues Programm", "Programm laden", "Programm speichern", "Quelltext einrücken", "Programm starten", "Stop", and "Einzelschrittmodus". Below the menu bar are buttons for "Funktions-Deklaration einfügen", "Variablen-Deklaration einfügen", "Bedingung einfügen", "Schleifen", and "Funktionsaufrufe". The main workspace is divided into a code editor and a simulation area.

The code editor shows the following C++ code for "roboter.pc":`1 {  
2 Roboter1.SetzeGeschwindigkeit(24);  
3 do  
4 {  
5 if ( Roboter1.LinksFrei())  
6 {  
7 Roboter1.Links(1);  
8 }  
9 else  
10 {  
11 if (! Roboter1.VorneFrei())  
12 {  
13 Roboter1.Rechts(1);  
14 }  
15 }  
16 if ( Roboter1.Dagewesen())  
17 {  
18 Roboter1.LoescheGemerkteRichtung();  
19 }  
20 Roboter1.MerkeRichtung();  
21 Roboter1.Vor();  
22 }  
23 while ((Roboter1.ObjektLinks != ZIEL) && (Rob  
24 Roboter1.WieSchnell();`

The simulation area shows a maze with a red brick wall. A blue robot is positioned at the bottom left. The maze contains several yellow arrows indicating the robot's path. The robot's current position is marked with a blue arrow pointing left. The maze is 20x20 cells. The robot starts at (18, 1) and moves towards the goal at (1, 18).

The debug console at the bottom left shows the following error messages:

```
Vor : Fehler -> Feld (16|15) ist nicht frei  
Vor : Fehler -> Feld (18|13) ist nicht frei  
Vor : Fehler -> Feld (18|18) ist nicht frei  
Vor : Fehler -> Feld (12|18) ist nicht frei  
Vor : Fehler -> Feld (11|17) ist nicht frei  
Vor : Fehler -> Feld (9|18) ist nicht frei  
Vor : Fehler -> Feld (6|18) ist nicht frei  
Vor : Fehler -> Feld (4|18) ist nicht frei
```

# IDE Anfängermodus

The screenshot displays the Chameleon IDE interface. At the top, the title bar reads "Chameleon (c) R.Hamann, Kiel, Texte von A.Bormann, Rendsburg". The menu bar includes "Datei", "Projekt", "Editor", "Designer", "Debug", and "Hilfe". Below the menu bar is a toolbar with various icons. A secondary toolbar contains buttons for "Funktionsdeklaration einfügen", "Variablen.deklaration einfügen", "Verzweigung einfügen", "Schleifen", "for-Schleife", and "Funktionsaufrufe".

The main workspace is divided into three panes. On the left is the "Eigenschaften" (Properties) pane, showing a tree view of properties for a selected component, such as "Action", "Anchors", "Caption", "Font", and "Width". Below it is the "Formulare" (Forms) pane, showing a tree view of the form structure with components like "frmform1(TForm)", "Memo1(TMemo)", and "Button1(TButton)".

The central pane is the code editor, displaying Pascal code for a unit named "form1". The code includes an interface, a type definition for "Tfrmform1", and implementation details for the "Create" constructor and the "Button1Click" event handler. The code is as follows:

```
1 unit form1;
2
3 interface
4
5 uses Forms, Controls, StdCtrls, SysUtils, Classes, Graphics,
6     Dialogs, ExtCtrls, Buttons, ImgList, Igel, Koordinatensystem;
7
8 type
9     Tfrmform1 = class(TForm)
10         Button1: TButton;
11         Memo1: TMemo;
12     private
13         procedure Button1Click(Sender: TObject);
14     protected
15     public
16         MainForm: boolean;
17         constructor Create(AOwner: TComponent); override;
18         procedure FormClose(Sender: TObject; var Action: TCloseAction);
19     end;
20
21 implementation
22
23 constructor Tfrmform1.Create(AOwner: TComponent);
24 begin
25     inherited Create(AOwner);
26     MainForm := false;
27     // Initcode wird hier automatisch eingefügt
28     // eigenen Initialisierungscode nach dieser Zeile einfügen
29 end;
30
31 procedure Tfrmform1.FormClose(Sender: TObject; var Action: TCloseAction);
32 begin
33     Action := caFree;
34     if MainForm then
35         TerminateScript;
36 end;
37
38
39 ////////////////////////////////////////////////////////////////////
40 // {Enter comment here}
41 ////////////////////////////////////////////////////////////////////
42 procedure Tfrmform1.Button1Click(Sender: TObject);
43 begin
44 end;
45
46
47 end;
```

On the right side is the "Komponenten" (Components) palette, which lists various standard Delphi components such as "Label", "Edit", "Memo", "Button", "CheckBox", "Radiobutton", "ListBox", "ComboBox", "GroupBox", and "Radiogroup".

In the foreground, a "Schleifen" (Loops) dialog box is open. It has tabs for "für Anfänger", "for-Schleife", "while-do-Schleife", and "do-while-Schleife". The "für Anfänger" tab is selected. It contains input fields for "Name der Zählvariablen", "Startwert (Zahlenwert oder Variablenname)", and "Endwert (Zahlenwert oder Variablenname)". Below these are radio buttons for "Zählrichtung" (Zählrichtung) with options "aufwärts" (selected) and "abwärts". At the bottom, there is a "Quelltextzeile" (Source code line) input field.

At the bottom of the IDE, there is a status bar showing "43:6 Modified" and a "Compiler" pane with tabs for "Suchergebnisse", "Hilfe-Stichworte", "Fehler", and "Überw".

# IDE Fortgeschrittenenmodus

The screenshot displays an IDE window titled "endsburg" in advanced mode. The main editor shows Pascal code for a form class `Tfrmform1`. The code includes a constructor `Create`, a `FormClose` procedure, and a `Button1Click` procedure. The `Button1Click` procedure is currently selected. A small dialog box is open in the top-left corner, showing a list of numbers (6, 7, 8, 9, 10) and a button labeled "Button1". On the right side, a "Komponenten" (Components) palette is visible, listing various UI controls such as Label, Edit, Memo, Button, Checkbox, Radiobutton, ListBox, ComboBox, Groupbox, Radiogroup, Panel, and New. The bottom status bar shows the time "62:19" and the word "Runing". The taskbar at the bottom includes icons for "chameleon" and the system clock "17:30".

```
16 form1 = class(TForm)
17   Button1: TButton;
18   Memo1: TMemo;
19   procedure Button1Click(Sender: TObject);
20   protected
21   end;
22 implementation
23   constructor Tfrmform1.Create(AOwner: TComponent);
24   begin
25     inherited Create(AOwner);
26     MainForm := false;
27     // FormCreateStart
28     Memo1 := TMemo.Create(self);
29     Memo1.Parent := self;
30     Memo1.Left := 88;
31     Memo1.Top := 24;
32     Memo1.Width := 185;
33     Memo1.Height := 89;
34     Memo1.TabOrder := 0;
35     Button1 := TButton.Create(self);
36     Button1.Parent := self;
37     Button1.Left := 144;
38     Button1.Top := 136;
39     Button1.Width := 75;
40     Button1.Height := 25;
41     Button1.Caption := 'Button1';
42     Button1.TabOrder := 1;
43     Button1.OnClick := Button1Click;
44     // FormCreateEnde
45     // eigenen Initialisierungscode nach dieser Zeile einfügen
46   end;
47   procedure Tfrmform1.FormClose(Sender: TObject; var Action: TCloseAction);
48   begin
49     Action := caFree;
50     if MainForm then
51       TerminateScript;
52   end;
53   // (Enter comment here)
54   // (Enter comment here)
55   // (Enter comment here)
56   // (Enter comment here)
57   // (Enter comment here)
58   // (Enter comment here)
59   // (Enter comment here)
60   procedure Tfrmform1.Button1Click(Sender: TObject);
61   var
62     n: integer;
63   begin
```

# Diagramm-Editor

The screenshot displays the Chameleon-Roboter software interface, which is used for programming a robot's path through a maze. The interface is divided into several sections:

- Top Bar:** Contains menu options such as "neues Programm", "Programm laden", "Programm speichern", "Drucken", "Bild speichern", "Quelltext einrücken", and "Syntaxhilfe". It also includes a "Programm starten" button and a "Stop" button.
- Toolbar:** Features buttons for "Funktionsdeklaration einfügen", "Variablendeklaration einfügen", "Verzweigung einfügen", "Schleifen", and "Funktionsaufrufe".
- Diagramm Editor:** The central area for creating a flowchart. It includes a "Diagramm" tab and a "Debug" tab. The flowchart for the "zur\_wand" function is visible, showing a sequence of steps: "Roboter1 Rechts(1)", a "while" loop containing "Roboter1 Vor()", and another "Roboter1 Rechts(1)".
- Maze Simulation:** On the right side, there is a grid-based maze with red brick walls. A blue triangle indicates the robot's starting position at the top left. A checkered flag indicates the goal. The maze has four vertical walls with gaps, creating a path through them.
- Bottom Bar:** Shows the Windows taskbar with the "chameleon" application icon and the system clock displaying "16:47".

# Quelltextergänzung

```
39 /// {Enter comment here}
40 /// //////////////////////////////////////
41 procedure TForm1.Button1Click(Sender: TObject);
42 begin
43     for n := 1 to 10 do begin
44         Mem1.
45     end;
46
47
48 end.
```

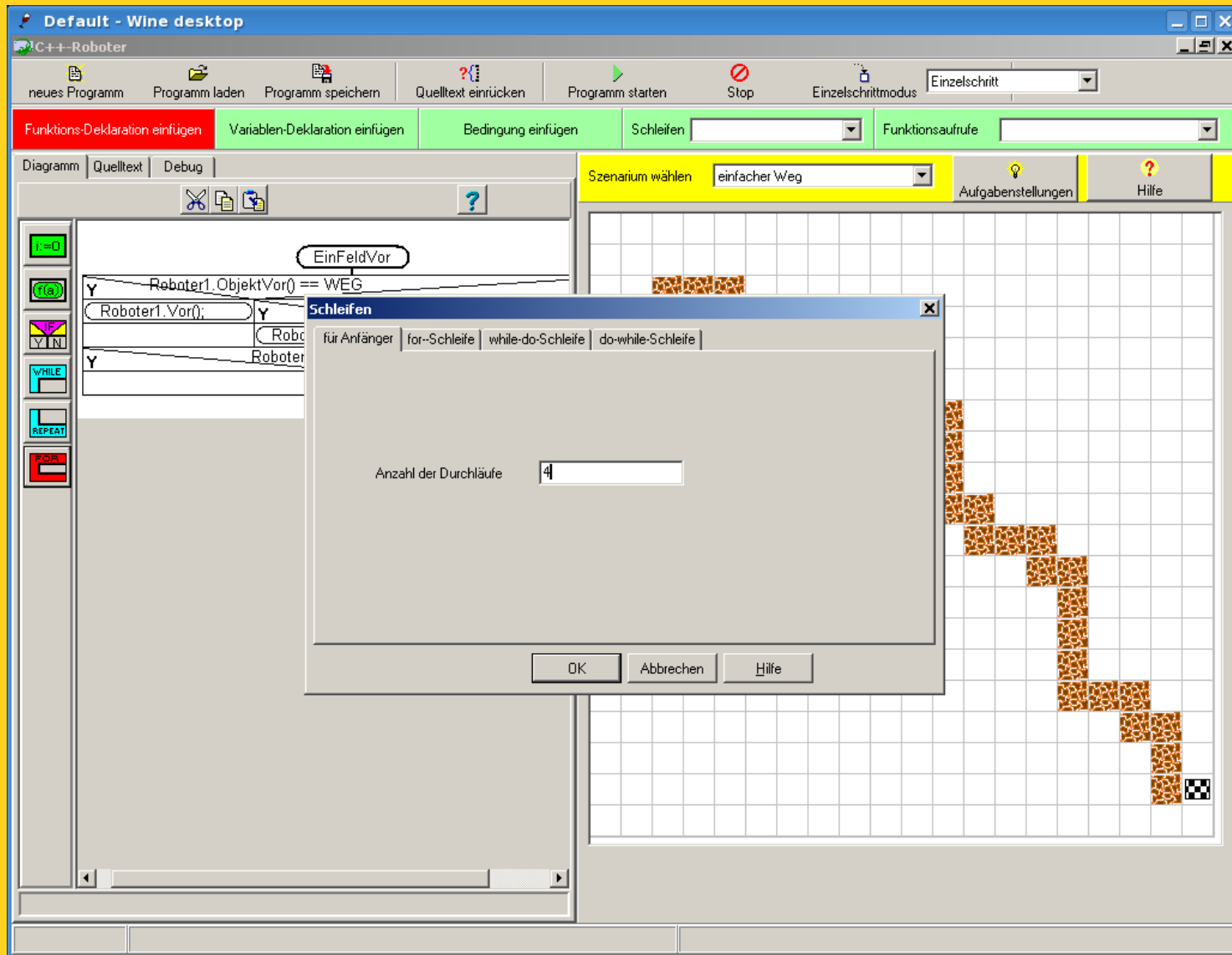
```
property Align: TAlign [alNone, alTop, alBottom, alLeft, alRight, alClient, alCustom]
property Alignment: TAlignment [taLeftJustify, taRightJustify, taCenter]
property Anchors: TAnchors [akLeft, akTop, akRight, akBottom]
property BevelEdges: TBevelEdges [beLeft, beTop, beRight, beBottom]
property BevelInner: TBevelCut [bvNone, bvLowered, bvRaised, bvSpace]
property BevelKind: TBevelKind [bkNone, bkTile, bkSoft, bkFlat]
property BevelOuter: TBevelCut [bvNone, bvLowered, bvRaised, bvSpace]
property BiDiMode: TBiDiMode [bdLeftToRight, bdRightToLeft, bdRightToLeftNoAlign, bdRightToLeftReadingOnly]
property BorderStyle: TBorderStyle [bsNone, bsSingle]
property Color: TColor
property Constraints: TSizeConstraints
property Ctl3D: Boolean [False, True]
property Cursor: TCursor
property DragCursor: TCursor
property DragKind: TDragKind [dkDrag, dkDock]
property DragMode: TDragMode [dmManual, dmAutomatic]
property Enabled: Boolean [False, True]
property Font: TFont
property Height: Integer
property HelpContext: THelpContext
property HelpKeyword: String
property HelpType: THelpType [htKeyword, htContext]
property HideSelection: Boolean [False, True]
```

# Wizard für Verzweigungen

The screenshot shows the 'C++-Roboter' IDE interface. The top menu bar includes options like 'neues Programm', 'Programm laden', 'Programm speichern', 'Quelltext einrücken', 'Programm starten', 'Stop', and 'Einzelschrittmodus'. Below the menu bar, there are buttons for 'Funktions-Deklaration einfügen', 'Variablen-Deklaration einfügen', 'Bedingung einfügen', 'Schleifen', and 'Funktionsaufrufe'. The main workspace is divided into several sections:

- Diagramm:** A flowchart showing a loop structure. The loop body contains a conditional branch: `Roboter1_ObjektVor() == WEG`. If true, it executes `Roboter1.Vor();`. Another branch is visible: `Roboter1_ObjektRechts() == WEG`.
- Bedingungen (Conditions):** A dialog box listing various robot state variables for selection. The list includes:
  - Roboter#.ZahlVor() : integer
  - Roboter#.AktuelleRichtung() : integer
  - Roboter#.AktuellerText() : string
  - Roboter#.AktuelleZahl() : integer
  - Roboter#.Dagewesen() : bool
  - Roboter#.Geschwindigkeit() : integer
  - Roboter#.LinksFrei() : bool
  - Roboter#.ObjekteGeladen() : integer
  - Roboter#.ObjektLinks() : integer
  - Roboter#.ObjektRechts() : integer
  - Roboter#.ObjektVor() : integer
  - Roboter#.RechtsFrei() : bool
  - Roboter#.TextLinks() : string
  - Roboter#.TextRechts() : string
  - Roboter#.TextVor() : string
  - Roboter#.VorDemRand() : bool
  - Roboter#.VorneFrei() : bool
  - Roboter#.xPosition() : integer
  - Roboter#.yPosition() : integer
  - Roboter#.ZahlLinks() : integer
  - Roboter#.ZahlRechts() : integer
  - Roboter#.ZahlVor() : integer
- Szenarium wählen:** A dropdown menu set to 'einfacher Weg'.
- Aufgabenstellungen:** A grid-based maze environment with a path of brown blocks leading to a goal (black and white checkered square).

# Wizard für Schleifen



# Wizards

- für Funktionsdeklarationen
  - Abfrage vieler Parameter
- für Variablendeklarationen
- für Laufzeitbedingungen
- für Schleifen
  - auch: einfache Schleife für Anfänger
- für Standardfunktionen
  - Abfrage der Parameter

# Fehlersuche

- Einzelschrittmodus
  - automatisch
  - manuell
  - Breakpoints in Diagrammen mit Anzeige aller
    - globalen Variablen
    - lokalen Variablen
- in Vorbereitung
  - Breakpoints im Quelltext

# Debugmodus im Diagramm

The screenshot displays the Chameleon-Roboter software interface. At the top, the title bar reads "Chameleon-Roboter". Below it, a menu bar includes options like "neues Programm", "Programm laden", "Programm speichern", "Drucken", "Bild speichern", "Quelltext einrücken", and "Syntaxhilfe". A toolbar contains buttons for "Programm starten", "Stop", "Weiter", and "bis zum Haltepunkt".

The main workspace is divided into several sections. On the left, there are tabs for "Diagramm", "Quelltext", and "Debug". Below these are icons for file operations and a list of actions: "NaechsterZug", "runter", "herum", "rauf", "zur\_wand", and "nichts".

The central area shows a flowchart for the "NaechsterZug" function. It starts with a loop condition: `while( Roboter1.ObjektVor() != ZIEL)`. Inside the loop, there are four sequential steps: `runter();`, `herum();`, `rauf();`, and `zur_wand();`.

On the right side, a red banner reads "gesperrt, solange das Programm läuft!". Below this is a grid-based environment with four vertical brick walls. A blue triangle representing the robot is positioned at the top left of the first wall. A small black and white checkered square is located on the right side of the grid.

At the bottom of the window, there are two panels: "Variablen" and "Funktionsaufrufe und Parameter". The "Funktionsaufrufe und Parameter" panel shows the function call "NaechsterZug()".

The Windows taskbar at the bottom indicates the time as 16:45 and shows the application name "chameleon".