```
238:          Form := GetParentForm(Self);
239:          if Form <> nil then
          Form.ModalResult := ModalResult;
240:        end;
241:   inherited Click;   { Aufruf der Methode des Vorfahrs }
242: end;
243:
244: procedure TSuperButton.MouseDown(Button:TMouseButton;
                              Shift:TShiftState;
                              X,Y:integer);
245: begin
246:   if not FToggle then Status := stEin;
247:   inherited MouseDown(Button, Shift, X, Y );
248: end;
249:
250: procedure TSuperButton.MouseUp(Button:TMouseButton;
                            Shift:TShiftState;
                            X,Y:integer);
251: begin
252:   if not FToggle then Status := stAus;
253:   inherited MouseUp(Button, Shift, X, Y );
254: end;
```

Durch die Erweiterung von TSuperButton mit den Eigenschaften ColorEin und ColorAus ist Color unnötig geworden und wird daher nicht mehr in published angeführt und wird auch nicht mehr im Objektinspektor angezeigt. Wenn diese neue Eigenschaft im Objektinspektor geändert wird, wird dies aber nicht im Formular sichtbar. Damit aber die gesetzte Eigenschaft dargestellt wird, muß die Komponente nach dem Laden ihrer Eigenschaftswerte noch initialisiert werden. Dies erfolgt mit der Methode Loaded, die die nunmehr nicht mehr sichtbare Eigenschaft Color setzt und anschließend das Objekt neu zeichnet. **(Z. 256..271)**

```
256: procedure TSuperButton.Loaded;
257: begin
258:   inherited Loaded;
259:   if FStatus=stEin
260:   then begin
262:          Color := FColorEin;
264:        end
265:   else begin
267:          Color := FColorAus;
269:        end;
270:   Invalidate;
271: end;
```

Z.273..279: Zum Abschluß muß die neue Komponente noch bei Delphi registriert werden, wofür die Prozedur Register zuständig ist. In Z.279 wird die Unit beendet.

```
273: procedure Register;
274: begin
275:   RegisterComponents('Beispiele', [TSuperButton]);
276: end;
279: end.
```

Auf der Diskette sind noch weitere Eigenschaften ergänzt: FontColorEin, FontColorAus und statt Caption sind CaptionAus und CaptionEin vorhanden.

Unerläßlich für das Verständnis des Verhaltens waren aber die Quelldateien aus der VCL, die im Delphi V1.0 Paket nicht enthalten waren und zusätzlich gekauft wurden.

Trotz einiger kleiner Problem war das Erstellen einer neuen Komponente aber recht einfach und das Ergebnis zeigt auch einen gut verständlichen Programmcode. Ein großer Vorteil ist aber darin zu sehen, daß alles in Delphi selbst (Pascal) zu realisieren ist und nicht eine weitere Programmsprache gelernt werden muß. □

```
Master Programmer
=================
[
uuid(2573F8F4-CFEE-101A-9A9F-00AA00342820)
]
library LHello
{
// bring in the master library
importlib("actimp.tlb");
importlib("actexp.tlb");

// bring in my interfaces
#include "pshlo.idl"

[
uuid(2573F8F5-CFEE-101A-9A9F-00AA00342820)
]
cotype THello
{
interface IHello;
interface IPersistFile;
};
};

[
exe,
uuid(2573F890-CFEE-101A-9A9F-00AA00342820)
]
module CHelloLib
{

// some code related header files
importheader(<windows.h>);
importheader(<ole2.h>);
importheader(<except.hxx>);
importheader("pshlo.h");
importheader("shlo.hxx");
importheader("mycls.hxx");

// needed typelibs
importlib("actimp.tlb");
importlib("actexp.tlb");
importlib("thlo.tlb");

[
uuid(2573F891-CFEE-101A-9A9F-
00AA00342820), aggregatable
]
coclass CHello
{
cotype THello;
};
};


#include "ipfix.hxx"

extern HANDLE hEvent;

class CHello : public CHelloBase
{
public:
IPFIX(CLSID_CHello);

CHello(IUnknown *pUnk);
~CHello();

HRESULT  __stdcall PrintSz(LPWSTR
pwszString);

private:
static int cObjRef;
};


#include <windows.h>
#include <ole2.h>
#include <stdio.h>
#include <stdlib.h>
#include "thlo.h"
#include "pshlo.h"
#include "shlo.hxx"
#include "mycls.hxx"

int CHello::cObjRef = 0;

CHello::CHello(IUnknown *pUnk) :
CHelloBase(pUnk) {
cObjRef++;
return;
}

HRESULT  __stdcall  CHello::PrintSz(LPWSTR
pwszString) {
printf("%ws\/n", pwszString);
return(ResultFromScode(S_OK));
}


CHello::~CHello(void)
{

// when the object count goes to zero,
stop the server cObjRef--;
if( cObjRef == 0 ) PulseEvent(hEvent);

return;
}

#include <windows.h>
#include <ole2.h>
#include "pshlo.h"
#include "shlo.hxx"
#include "mycls.hxx"

HANDLE hEvent;
int _cdecl main(
int argc,
char * argv[]
) {
ULONG ulRef;
DWORD dwRegistration;
CHelloCF *pCF = new CHelloCF();

hEvent = CreateEvent(NULL, FALSE, FALSE,
NULL);

// Initialize the OLE libraries
CoInitializeEx(NULL,
COINIT_MULTITHREADED);

CoRegisterClassObject(CLSID_CHello, pCF,
AL_SERVER,
REGCLS_MULTIPLEUSE, &dwRegistration);

// wait on an event to stop
WaitForSingleObject(hEvent, INFINITE);

// revoke and release the class object
CoRevokeClassObject(dwRegistration); ulRef
= pCF->Release();

// Tell OLE we are going away.
CoUninitialize();

return(0); }

extern CLSID CLSID_CHello;
extern UUID LIBID_CHelloLib;

CLSID CLSID_CHello = { /*
2573F891-CFEE-101A-9A9F-00AA00342820 */
0x2573F891,
0xCFEE,
0x101A,
{ 0x9A, 0x9F, 0x00, 0xAA, 0x00, 0x34,
0x28, 0x20 }
};

UUID LIBID_CHelloLib = { /*
2573F890-CFEE-101A-9A9F-00AA00342820 */
0x2573F890,
0xCFEE,
0x101A,
{ 0x9A, 0x9F, 0x00, 0xAA, 0x00, 0x34,
0x28, 0x20 }
};

#include <windows.h>
#include <ole2.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "pshlo.h"
#include "shlo.hxx"
#include "clsid.h"

int _cdecl main(
int argc,
char * argv[]
) {
HRESULT  hRslt;
IHello        *pHello;
ULONG  ulCnt;
IMoniker * pmk;
WCHAR  wcsT[_MAX_PATH];
WCHAR  wcsPath[2 * _MAX_PATH];

// get object path
wcsPath[0] = '\/0';
wcsT[0] = '\/0';
if( argc > 1) {
mbstowcs(wcsPath, argv[1], strlen(argv[1])
+ 1); wcsupr(wcsPath);
} else { fprintf(stderr, "Object path must
be specified\/n");
return(1); }

// get print string
if(argc > 2)
mbstowcs(wcsT, argv[2], strlen(argv[2]) +
1);
else
wcscpy(wcsT, L"Hello World");

printf("Linking to object %ws\/n",
wcsPath); printf("Text String
%ws\/n", wcsT);

// Initialize the OLE libraries
hRslt = CoInitializeEx(NULL,
COINIT_MULTITHREADED);

if(SUCCEEDED(hRslt)) {


hRslt = CreateFileMoniker(wcsPath, &pmk);
if(SUCCEEDED(hRslt))
hRslt = BindMoniker(pmk, 0, IID_IHello,
(void **)&pHello);

if(SUCCEEDED(hRslt)) {

// print a string out
pHello->PrintSz(wcsT);

Sleep(2000);
ulCnt = pHello->Release();
}
else
printf("Failure to connect, status: %lx",
hRslt);

// Tell OLE we are going      CoUninitial-
ize(); }

return(0);
}
□
```