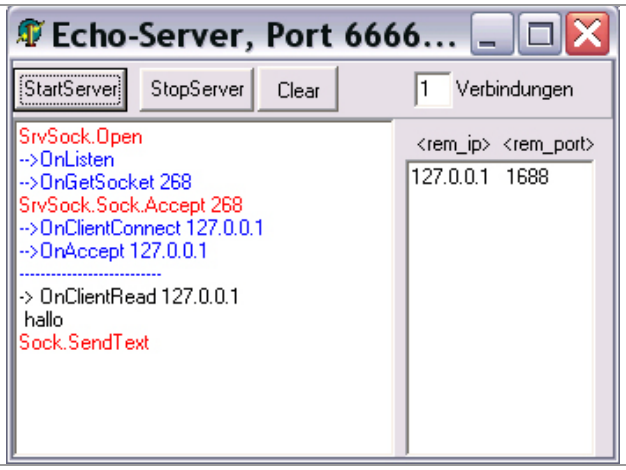
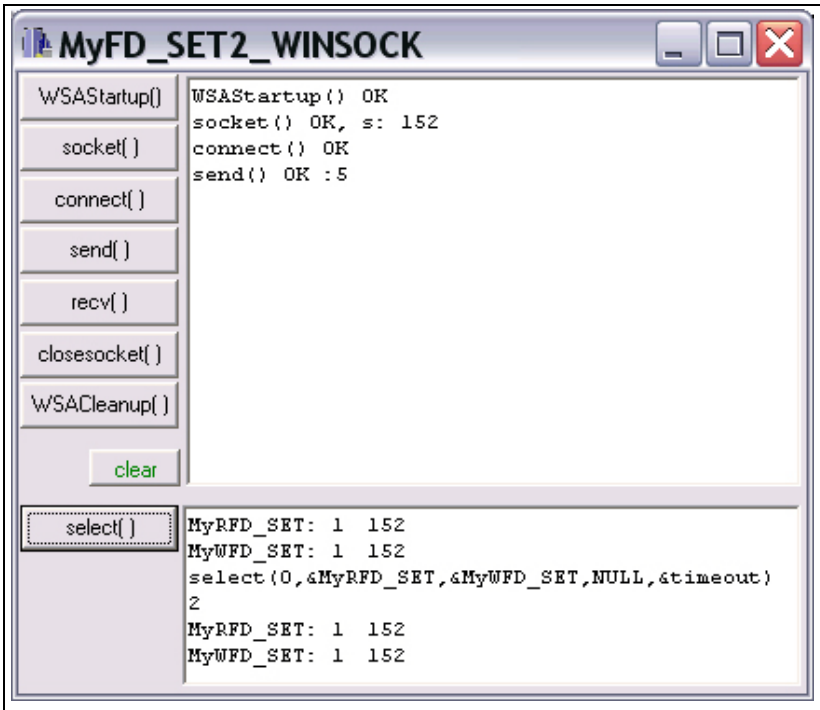
1.1 Ziel des Projektes

- Nutzung der Erkenntnisse aus MyFD_SET1 im Umgang mit Strukturen vom Typ fd_set
- Grundlegender Umgang mit der Funktion **select()**, bei der Abfrage ob Sockets lesbar und/oder schreibbar sind.
- Umsetzung:
  - Wir nutzen erst mal einen Clientsocket.
  - Hier sind die Verhältnisse übersichtlich, da nur ein Socket existiert.
- Im Beispiel MyEchoSrv1 werden dann die Erkenntnisse auf ein Serversocket angewendet.



- **Voraussetzung**: Downloaden Sie EchoSrv.EXE und starten Sie den Server.

1.2 Realisierung des Projektes MyFD_SET1



- Kopieren Sie das Beispiel MyEchoClt1_WINSOCK in den Ordner MyFD_SET2.
- Benennen Sie das Projekt um.
- Fügen Sie Button select() und das RichEdit2 hinzu.
- Legen Sie Strukturen vom Typ fd_set an:
  - MyRFD_SET, für die lesbaren Sockets
  - MyWFD_SET, für die schreibbaren Sockets.
- Im unteren RichEdit werden beide Strukturen vor und nach dem Aufruf von select() angezeigt.
- Man sieht im Beispiel:
- Der Socket mit dem Descriptor 152 ist sowohl schreibbar als auch lesbar.

```
//---------------------------------------------------------------------------
#include <vcl.h>
#include <winsock.h> //erforderlich
#include <stdio.h>   //erforderlich z.B. für putchar()
#include <string>
#pragma hdrstop
#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
int s;          //socket descriptor
fd_set MyRFD_SET, MyWFD_SET; //strukturen zur descriptorverwaltung anlegen
int len;        //für send() und recv()
char buf[4095]; //für send() und recv()
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
            : TForm(Owner)
{
}
//---------------------------------------------------------------------------
//---------------------------------------------------------------------------
void __fastcall TForm1::WSAStartup_Click(TObject *Sender)
{//===(1)Am API anmelden
 WSADATA wsaD;
  if  (WSAStartup (MAKEWORD(2,0),&wsaD) ==0) //MAKEWORD(maj,min)
      RichEdit1->Lines->Add("WSAStartup() OK");
  else
      RichEdit1->Lines->Add("WSAStartup() ERR");
}
//---------------------------------------------------------------------------
void __fastcall TForm1::socket_Click(TObject *Sender)
{//===(2)Socket errichten
  s = socket(AF_INET,SOCK_STREAM,0);
  if (s!=0)
      RichEdit1->Lines->Add("socket() OK, s: "+IntToStr(s));
  else
      RichEdit1->Lines->Add("socket() ERR");
}

//---------------------------------------------------------------------------
void __fastcall TForm1::connect_Click(TObject *Sender)
{//===(3)Adresse in Struktur sockaddr_in eintragen
   sockaddr_in  sAddr;
   sAddr.sin_family=AF_INET;
   sAddr.sin_port=htons(6666);
   sAddr.sin_addr.s_addr=inet_addr("127.0.0.1");
   sAddr.sin_addr.S_un.S_addr;
 //===(4)Verbindung herstellen, das Ziel geht aus sAddr hervor
   if (connect (s,(sockaddr*)&sAddr,sizeof(sockaddr))==0)
       RichEdit1->Lines->Add("connect() OK");
      else
       RichEdit1->Lines->Add("connect() ERR: "+IntToStr(WSAGetLastError()));
}
//---------------------------------------------------------------------------
void __fastcall TForm1::recv_Click(TObject *Sender)
{//===(5)Daten aus dem Socket lesen
   len = recv(s, buf, sizeof(buf),0);
   if ((len!=0)&& (len!=SOCKET_ERROR))
   {
       RichEdit1->Lines->Add("recv() OK :"+IntToStr(len));
       buf[len]=0;
       RichEdit1->Lines->Add(buf);
      }
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::send_Click(TObject *Sender)
{//===(5)Daten in den Socket schreiben
      char *buf="hallo"+0;
      len =send (s,buf,strlen(buf),0);
      if (len!=SOCKET_ERROR)
        RichEdit1->Lines->Add("send() OK :"+IntToStr(strlen(buf)));
      else
        RichEdit1->Lines->Add("connect() ERR: "+IntToStr(WSAGetLastError()));
}
//---------------------------------------------------------------------------
void __fastcall TForm1::closesocket_Click(TObject *Sender)
{//===(6)Verbindung abbauen
   if (closesocket(s)!=SOCKET_ERROR)
       RichEdit1->Lines->Add("closesocket() OK");
   else
       RichEdit1->Lines->Add("closesocket() ERR: "+IntToStr(WSAGetLastError()));
}
//---------------------------------------------------------------------------
void __fastcall TForm1::WSACleanup_Click(TObject *Sender)
{//===(7)Am API abmelden
     WSACleanup();
       RichEdit1->Lines->Add("WSACleanup() OK");
}
//---------------------------------------------------------------------------
void __fastcall TForm1::clear_Click(TObject *Sender)
{
  RichEdit1->Clear();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::anzeigeFD_SETs(TObject *Sender)
{
  RichEdit2->Lines->Add("MyRFD_SET: "+IntToStr(MyRFD_SET.fd_count)
                                     +"\t"+IntToStr(MyRFD_SET.fd_array[0]));
  RichEdit2->Lines->Add("MyWFD_SET: "+IntToStr(MyWFD_SET.fd_count)
                                     +"\t"+IntToStr(MyWFD_SET.fd_array[0]));
}
//---------------------------------------------------------------------------
void __fastcall TForm1::select_Click(TObject *Sender)
{
   RichEdit2->Clear();
   FD_ZERO(&MyRFD_SET); FD_ZERO(&MyWFD_SET);
   FD_SET(s,&MyRFD_SET); //eintragen des socketdescriptors in MyReadFD_SET
   FD_SET(s,&MyWFD_SET); //eintragen des socketdescriptors in MyWriteFD_SET
   struct timeval timeout;
   timeout.tv_sec=1; timeout.tv_usec=0;
   anzeigeFD_SETs(Form1);
   RichEdit2->Lines->Add("select(0,&MyRFD_SET,&MyWFD_SET,NULL,&timeout)");
   int retWert=select(0,&MyRFD_SET,&MyWFD_SET,NULL,&timeout);
   RichEdit2->Lines->Add(IntToStr(retWert));
   anzeigeFD_SETs(Form1);
}
//---------------------------------------------------------------------------
```