

Voor wij begin

Kopie van het netwerk Visual Age, toolkit en fixpacks naar eigen computer naar een tijdelijk locatie, en uitpakken.

Gebruik FTPPM

HOST = 192.168.2.26

User = Test

Password = Test

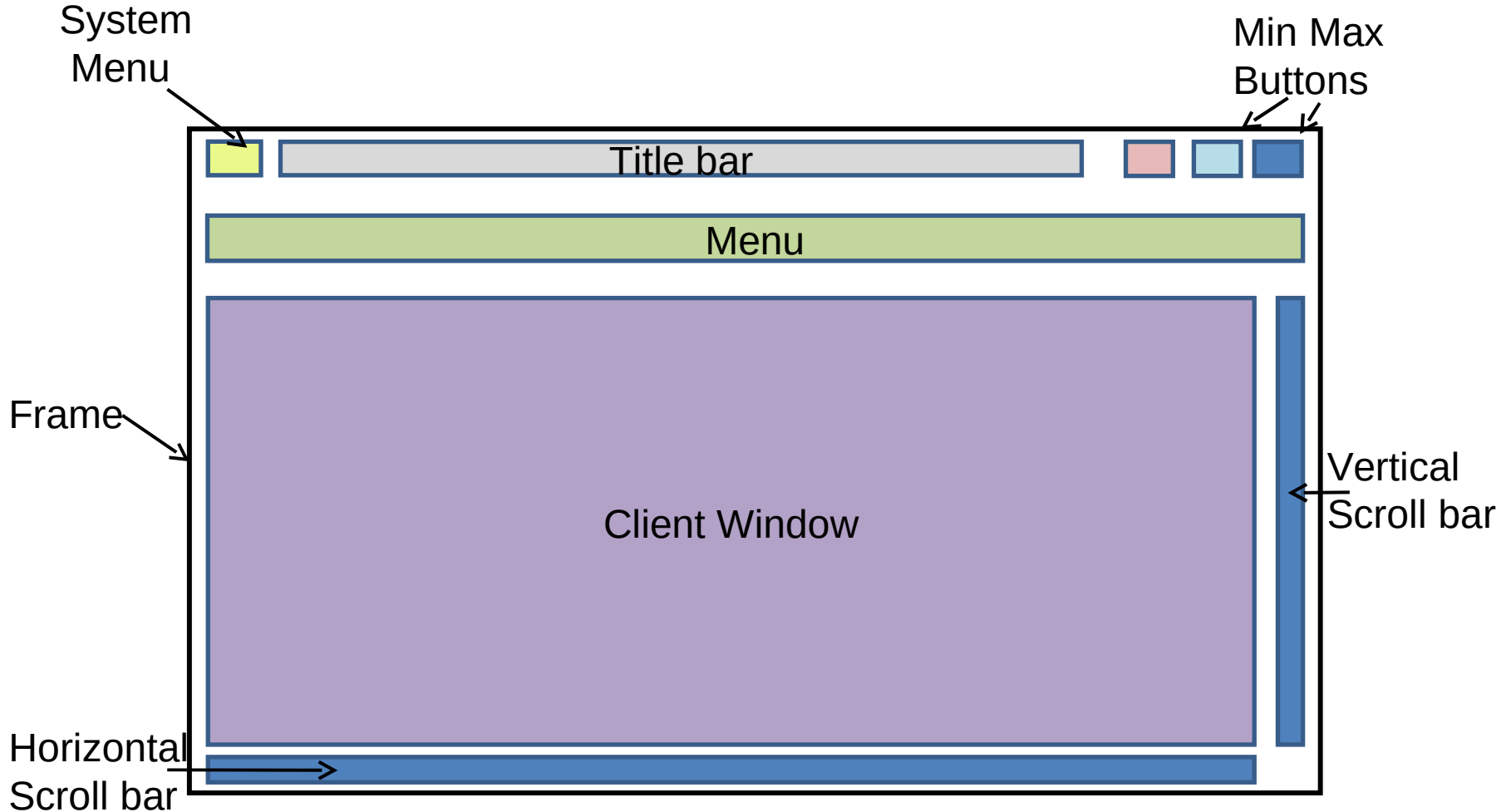
Account = Keith

Selekt /pools/A/A0/PM_Programming

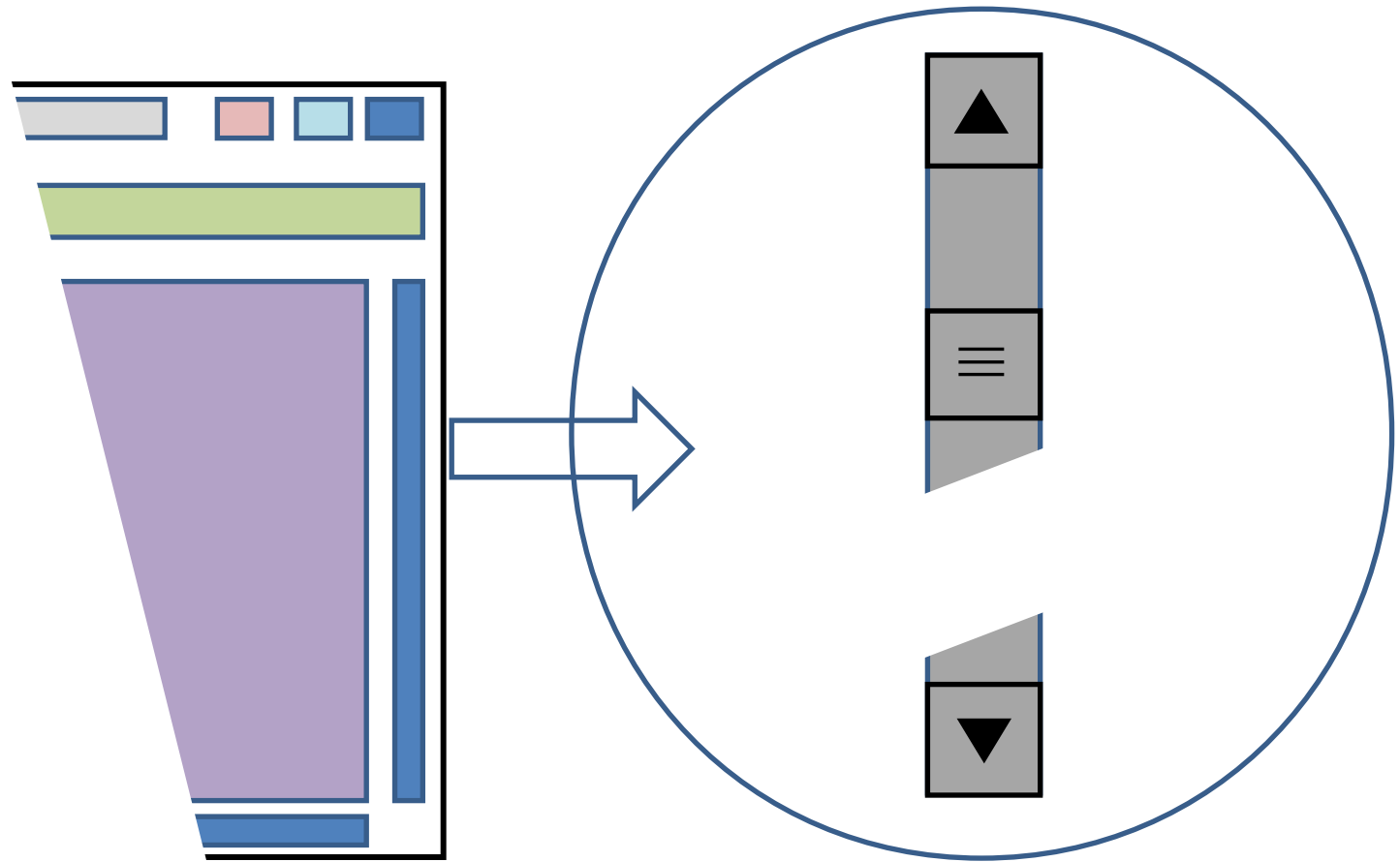
PM Programming

- Deze Workshop
 - Wat is een window
 - Wat een PM programma is
 - Structuur van een PM programma
 - Benodigdheden
 - Concepten
 - Compileren
 - Debuggen
 -

Het Standaard Window



Het Standaard Window



Window

- Een window heeft zowel een ouder (parent) als een (owner) eigenaar:
- Het top level window is de desktop (HWND_DESKTOP). Dus de eerste window heeft als parent de handle HWND_DESKTOP.

Window

Parent Child relationship

- Het kind window is beperkt qua positie in dat het kan nooit buiten de rand van de ouder window. Als het wordt verplaatst binnen dat window wordt het desnoods bijgesneden!
- De window eigenaar is degene dat de boodschappen ontvangt.

Demo child/parent

Wat is een PM programma

- Het is een programma dat gebruik maakt van OS/2's grafische gebruikers Interface (GUI).
- Iedere programma heeft bij definitie een of meerdere windows.
- In een PM programma (bijna) alle communicatie is via boodschappen.

PM Programming

- Benodigdheden
 - A compiler
 - VCC 3.0
 - VCC Fixpacks 3.08
 - IBM Toolkit Ver 4.52
 - Kennis
 - Van het C programmeer taal

Waarom VCC

- Is inclusief diverse tools, linker, C en C++ compiler en een zeer handig contextgevoelig editor* EVFXLXPM
- Via editor geselecteerde API, direct toegang tot informatie in hulp bestanden.
- Goed debugger
- Project georiënteerd
- Relatief eenvoudige in gebruik

* Editor kan niet gebruikt worden met sommige schermresoluties!

De Bestanden

- Programma code xxx.c*
- Resource code xxx.rc
- Uitvoerbaar code xxx.exe
- Object code xxx.obj
- Definitie code xxx.def*
- Headerbestanden xxx.h
- Bibliotheken xxx.lib
- Maak bestanden xxx.mak*

* Onze bestanden

Minimaal PM programma code

```
#define INCL_WIN           // nodig voor Window commando
                           // (b.v. WinCreateWindow)
#include <os2.h>           // bronbestand (header file)
int main (void)
{
    HAB          hab ;    // Anker block handvat
    HMQ          hmq ;    // Boodschappen wachtrij handvat
    QMSG         qmsg ;// Boodschappen wachtrij structuur

    hab    = WinInitialize (0) ;
    hmq    = WinCreateMsgQueue (hab, 0) ;
```

---- DOE IETS ----

Minimaal PM programma code

// wachtrij voor een boodschap en doorsturen naar de
desbetreffende procedure

```
while (WinGetMsg (hab, &qmsg, NULLHANDLE, 0, 0))  
    WinDispatchMsg (hab, &qmsg) ;
```

```
WinDestroyWindow (hwndFrame) ; // sluit window  
WinDestroyMsgQueue (hmq) ;      // sluit queue  
WinTerminate (hab) ;             // sluit programma  
return 0 ;                       // geef retour code  
}
```

Minimaal PM programma code

- Een ankerblok (anchor block) of houvast creëren
- Een boodschap (message) wachtrij creëren

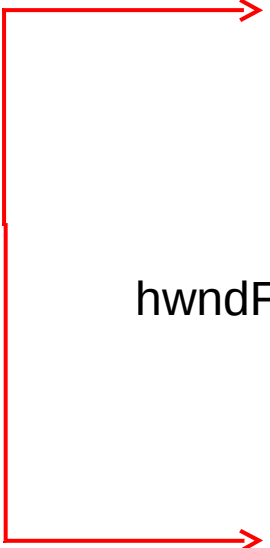
---- DOE IETS ----

- Wacht op een “message”
- Verstuur “message” naar het desbetreffende procedure
- Als het “message” is een einde “message” dan
 - het window vernietigen
 - Het wachtrij vernietigen
 - Programma beëindigen

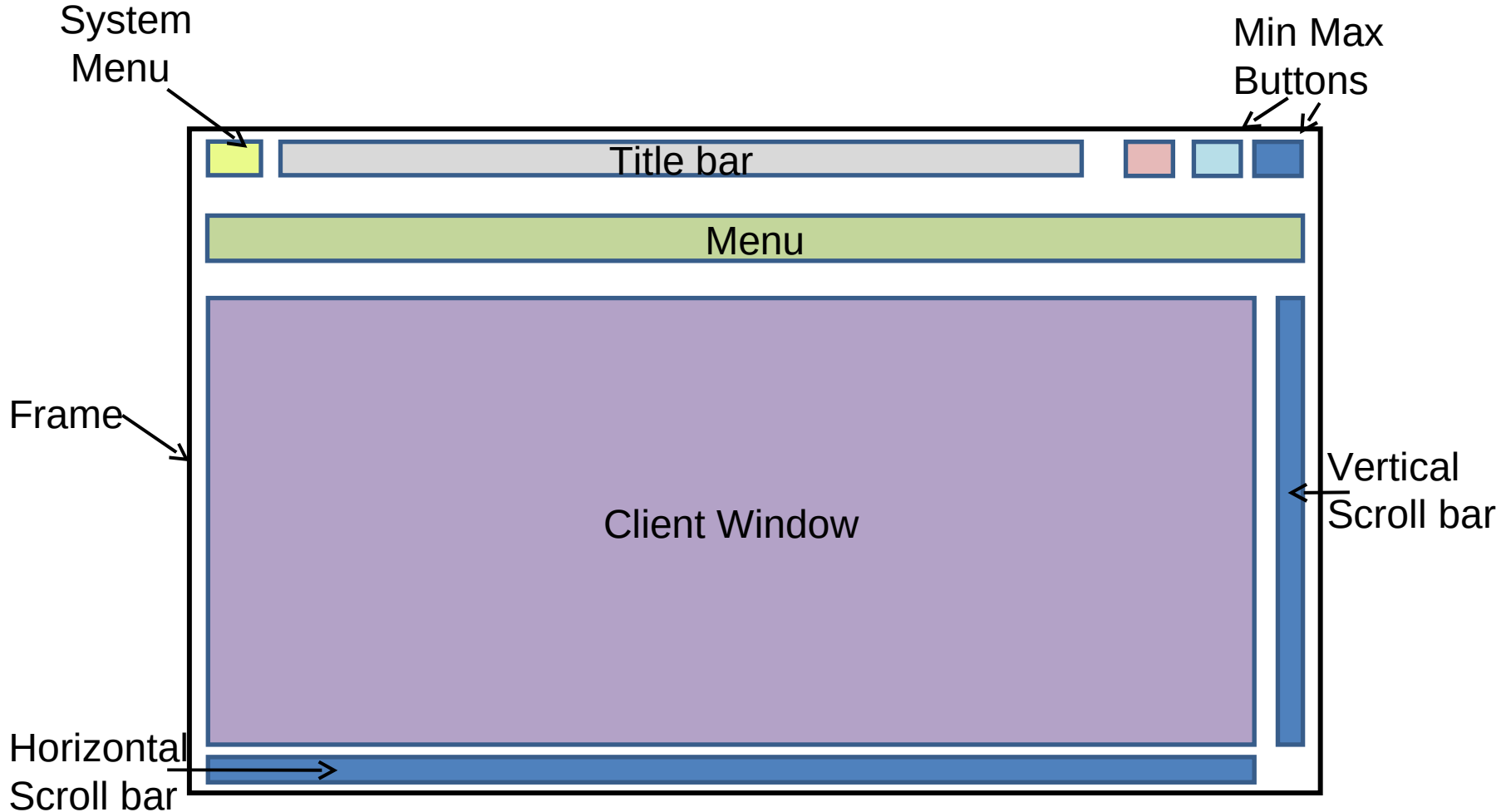
Doe iets code

Creëer een standaard 'window'

```
WinRegisterClass (  
    hab,                                // Anchor block handle  
    szClientClass,                      // Name of class being registered  
    ClientWndProc,                      // Window procedure for class  
    CS_SIZEREDRAW,                     // Class style  
    0) ;                               // Extra bytes to reserve  
  
hwndFrame = WinCreateStdWindow (  
    HWND_DESKTOP,                      // Parent window handle  
    WS_VISIBLE,                        // Style of frame window  
    &flFrameFlags,                     // Pointer to control data  
    szClientClass,                     // Client window class name  
    NULL,                              // Title bar text  
    0L,                                // Style of client window  
    0,                                 // Module handle for resources  
    0,                                 // ID of resources  
    &hwndClient) ;                     // Pointer to client window handle
```



Het Standaard Window



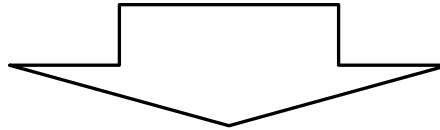
Het Standaard Window Identifiers

FID_SYSMENU	-	Systeem Menu
FID_TITLEBAR	-	Titel bar
FID_MENU	-	Programma menu
FID_MINMAX	-	Minimaliseren/maximaliseren knoppen
FID_CLIENT	-	Cliënt window
FID_VERTSCROLL	-	Verticaal schuifbalk
FID_HORIZ	-	Horizontaal schuifbalk

Doe iets code

definities voor WinCreateStdWindow

```
HWND          hwndFrame ;
static CHAR    szClientClass[ ] = "Welkom" ;
static ULONG   flFrameFlags  = FCF_TITLEBAR | FCF_SIZEBORDER
                               | FCF_MINMAX | FCF_SYSMENU
                               | FCF_SHELLPOSITION | FCF_TASKLIST;
```



FCF_TITLEBAR	-	window heeft een titel
FCF_SIZEBORDER	-	window kan verkleind/vergroot worden
FCF_MINMAX	-	window heeft min max knoppen
FCF_SYSMENU	-	window heeft een systeem menu
FCF_SHELLPOSITION	-	window heeft een gedefinieerde positie
FCF_TASKLIST	-	Programma wordt opgenomen in de taaklijst

Doe iets code

WinCreateStdWindow – Frame Flags

FCF_MIN , FCF_MAX, FCF_VERTSCROLL, FCF_HORIZSCROLL,
FCF_DLGBORDER, FCF_BORDER, FCF_NOBYTEALIGN,
FCF_NOMOVEWITHOWNER, FCF_ICON*, FCF_ACCELTABLE*,
FCF_SCREENALIGN, FCF_MOUSEALIGN, FCF_HIDEBUTTON,
FCF_HIDEMAX, FCF_AUTOICON*,

FCF_STANDARD = (FCF_SYSMENU, FCF_SYSMENU,
FCF_SIZEBORDERMENU FCF_MINMAX, FCF_ICON,
FCF_ACCELTABLE*, FCF_SHELLPOSITION en FCF_TASKLIST)

*Deze opties maak gebruik van informatie in een resource. Als er geen resource is dan zal de WinCreateStdWindow of WinCreateWindow een fout genereren en geen window wordt gemaakt.

Window procedures

- Iedere window heeft een geassocieerd procedure, maar meerdere windows kan een en hetzelfde procedure hebben!
- In het window of dialoog procedure worden boodschappen afgehandeld.
- Het procedure neemt als informatie:
Window handle, boodschap, parameter 1 en 2
(hwnd, msg, mp1, mp2)

Window procedures

- In onze voorbeeld worden 3 boodschappen afgehandeld:
 - WM_CREATE: Het window wordt gecreëerd
 - Wij laat een geluid horen
 - WM_PAINT: Het window moet worden vernieuwd
 - Wij vullen de window in en schrijf wat tekst
 - WM_CLOSE: Het programma wordt beëindig
 - Wij laat een geluid horen

Doe iets code

Het window procedure

```
MRESULT EXPENTRY ClientWndProc (HWND hwnd, ULONG msg,  
                                MPARAM mp1, MPARAM mp2)  
{  
    HPS          hps;  
    RECTL        rcl ;  
  
    switch (msg)  
    {  
        case WM_CREATE:  
            DosBeep (261, 100) ; DosBeep (330, 100) ;  
            DosBeep (392, 100) ; DosBeep (523, 500) ;  
            return 0 ;
```

Doe iets code

Het window procedure

```
case WM_PAINT:  
    hps = WinBeginPaint (hwnd, NULLHANDLE, NULL) ;  
    WinQueryWindowRect (hwnd, &rcl) ;  
    WinDrawText (hps, -1, "Welkom!", &rcl, CLR_NEUTRAL,  
                CLR_BACKGROUND, DT_CENTER |  
                DT_VCENTER | DT_ERASERECT) ;  
    WinEndPaint (hps) ;  
    return 0 ;
```

Doe iets code

Het window procedure

```
case WM_DESTROY:
    DosBeep (523, 100) ;
    DosBeep (392, 100) ;
    DosBeep (330, 100) ;
    DosBeep (261, 500) ;
    return 0 ;
}
return WinDefWindowProc (hwnd, msg, mp1, mp2) ;
}
```

WinBeginPaint

```
#define INCL_WINWINDOWMGR  
#include <os2.h>
```

```
HWND    hwnd;        // Handle of window where drawing is  
                        going to occur.  
HPS      hps;          // Presentation-space handle.  
PRECTL   prclPaint;    // Bounding rectangle.  
HPS      hpsPaintPS;   // Presentation-space handle.  
  
hps = WinBeginPaint(hwnd, hps, prclPaint);
```

WinQueryWindowRect

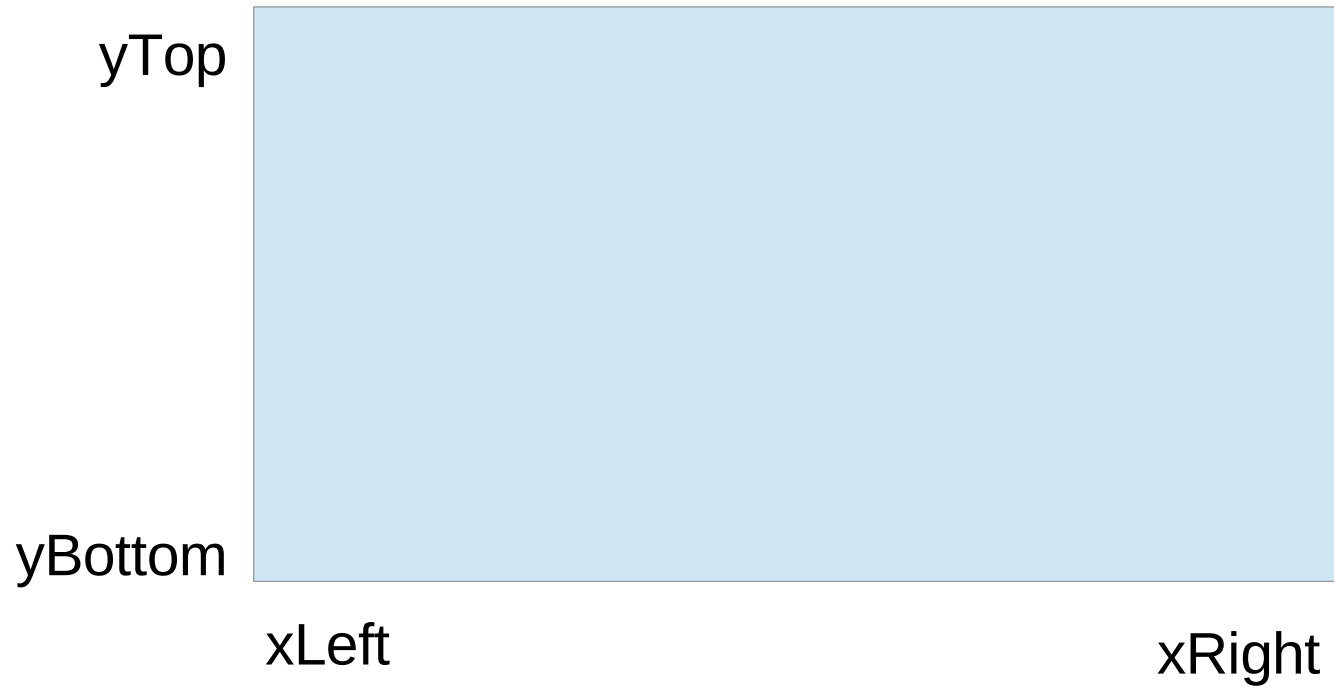
```
#define INCL_WINWINDOWMGR // Or use INCL_WIN, INCL_PM
```

```
Also in COMMON section  
#include <os2.h>
```

```
HWND    hwnd;        // Window handle whose rectangle is retrieved.  
PRECTL   prclDest;    // Window rectangle.  
BOOL     rc;          // Rectangle-returned indicator
```

```
rc = WinQueryWindowRect(hwnd, prclDest);
```

Structure RECTL



Structure RECTL

```
typedef struct _RECTL {  
    LONG    xLeft;        // X-coordinate of left-hand edge of rectangle.  
    LONG    yBottom;     // Y-coordinate of bottom edge of rectangle.  
    LONG    xRight;       // X-coordinate of right-hand edge of rectangle.  
    LONG    yTop;         // Y-coordinate of top edge of rectangle.  
} RECTL;
```

```
typedef RECTL *PRECTL;
```

WinDrawText

```
#define INCL_WINWINDOWMGR // Or use INCL_WIN, INCL_PM,  
#include <os2.h>
```

```
HPS      hps;           // Presentation-space handle.  
LONG     cchText;       // Number of characters in the string.  
PCH      lpchText;      // Character string to be drawn.  
PRECTL   prcl;          // Text rectangle.  
LONG     clrFore;       // Foreground color.  
LONG     clrBack;       // Background color.  
ULONG    flCmd;         // Flags to determines how text is drawn  
LONG     lChars;        // Number of chars drawn
```

```
lChars = WinDrawText(hps, cchText, lpchText,  
                    prcl, clrFore, clrBack, flCmd);
```

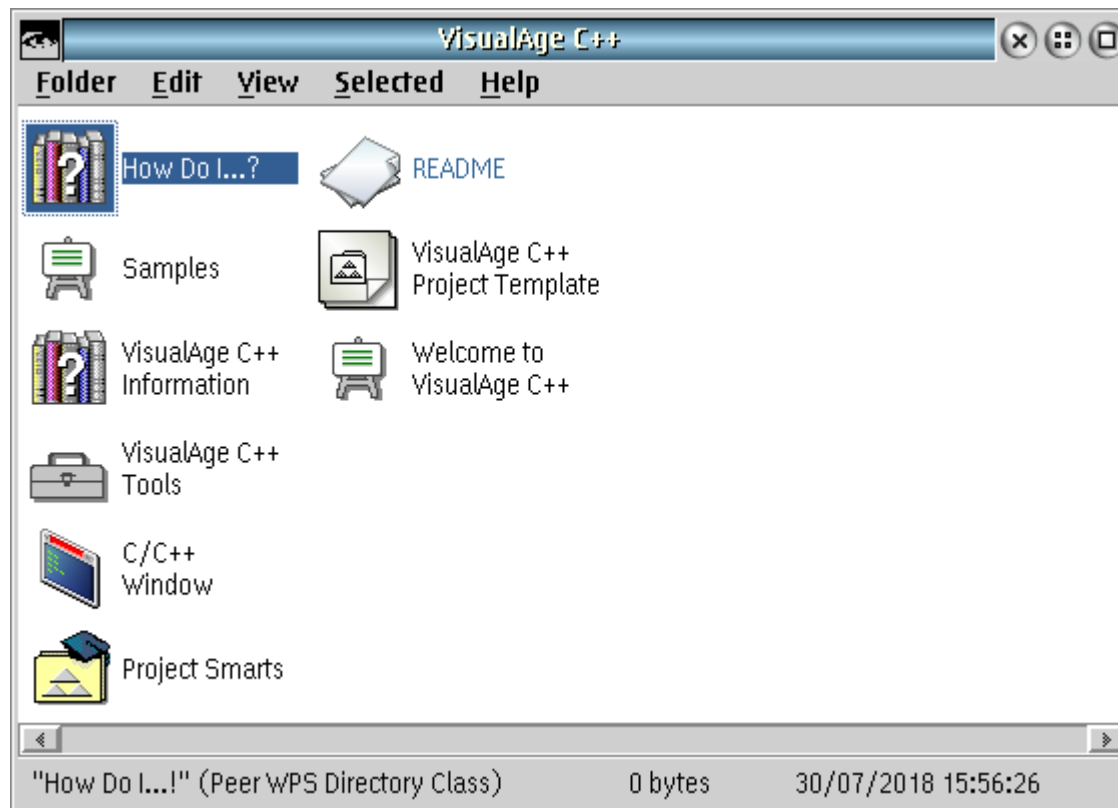
Message structure

```
typedef struct _QMSG
{
    HWND      hwnd;      // Window handle.
    ULONG     msg;       // Message identity.
    MPARAM    mp1;       // Parameter 1.
    MPARAM    mp2;       // Parameter 2.
    ULONG     time;      // Message time.
    POINTL    ptl;       // Pointer position when message
                        // was generated.
    ULONG     reserved;  // Reserved.
} QMSG;
```

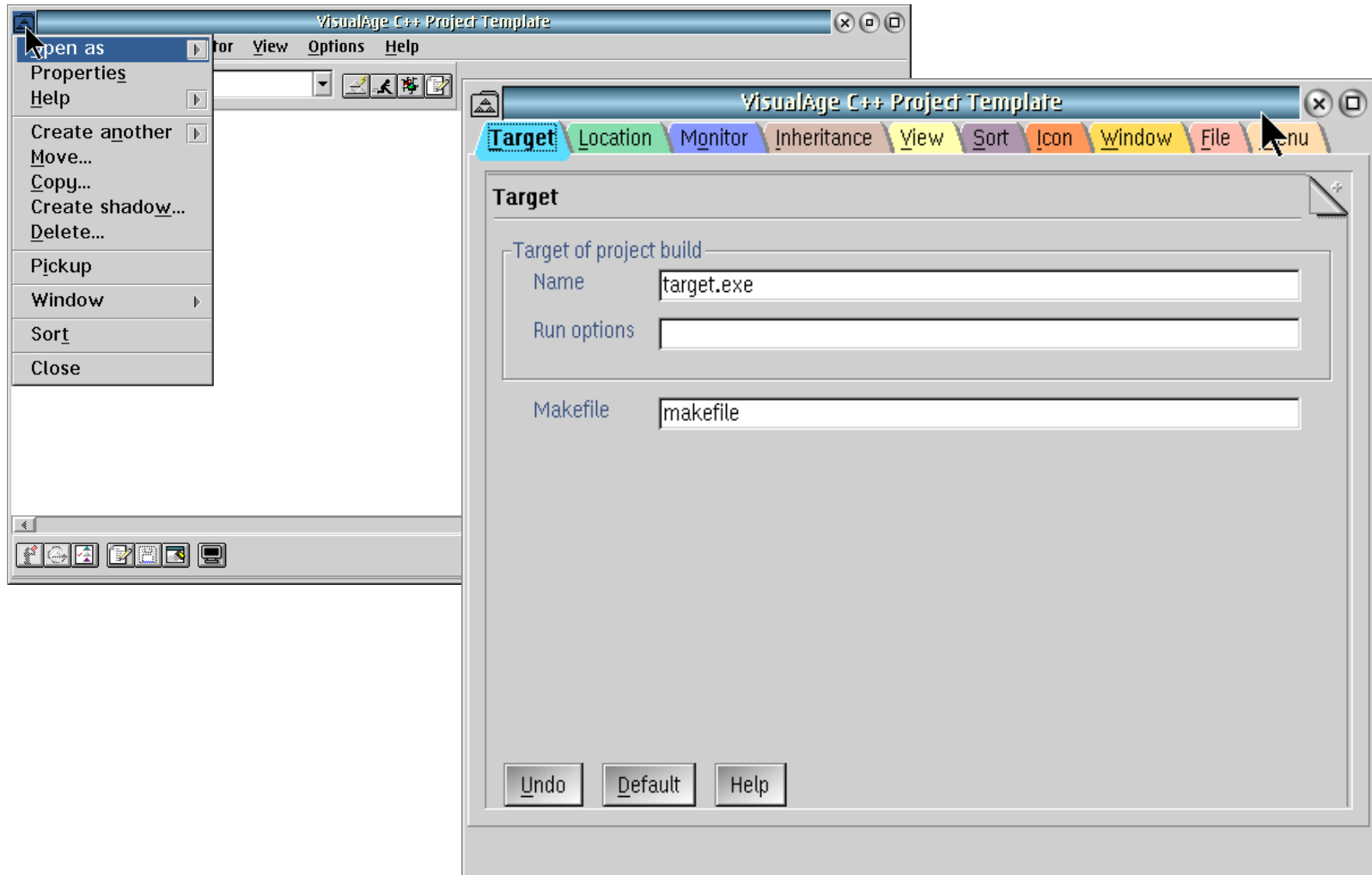
Uitleg diverse onderdelen VCC

- Template
 - Editor
 - Compiler/Linker
 - Make bestand
 - Debugger
 - Run

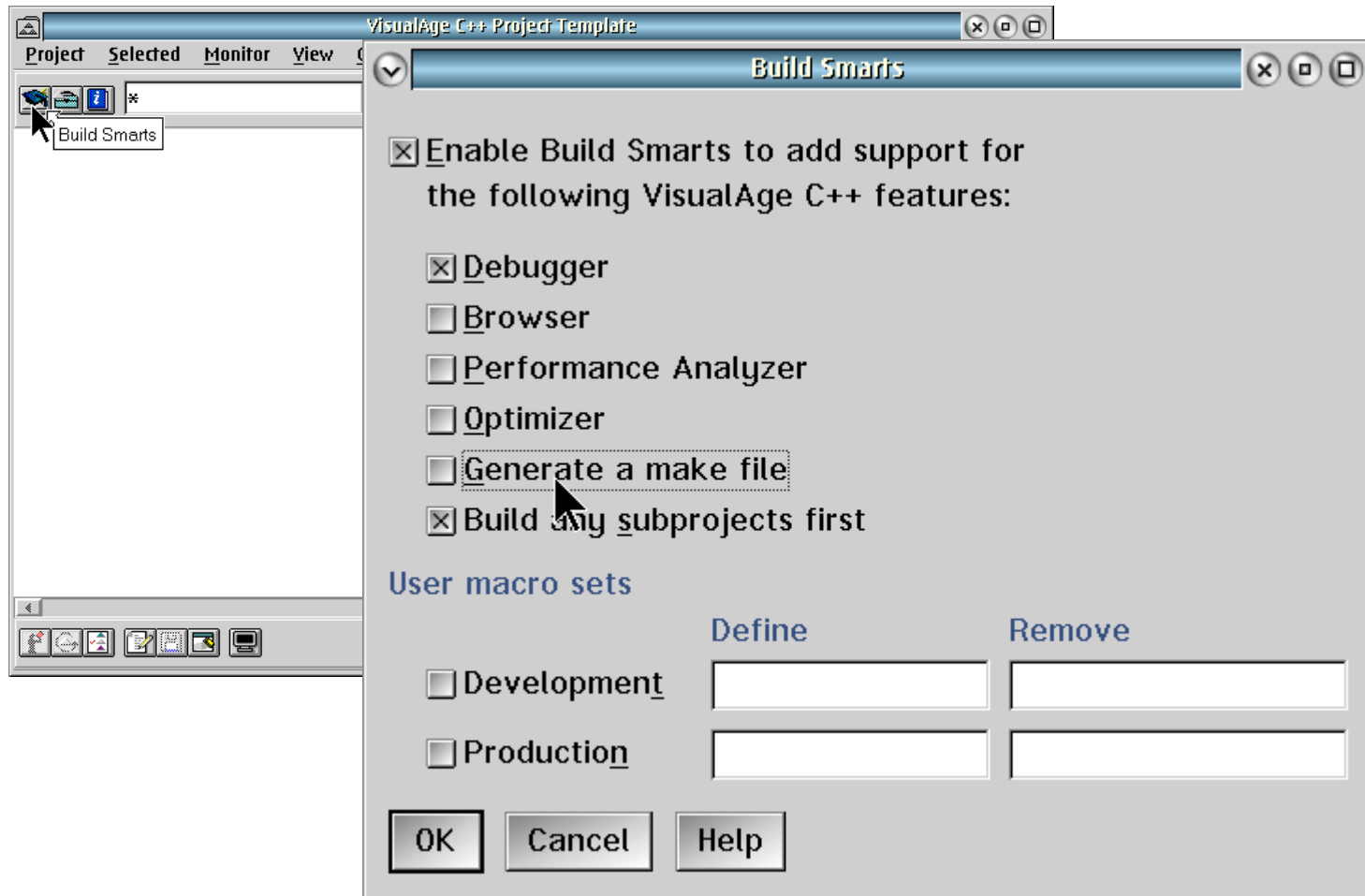
VCC Inhoud



VCC Project template



VCC Project template



VCC Editor

- Demo

VCC Compiler

- Demo

PM Code

Het definitie bestand

Welkom.def

```
;-----  
;  info betreffende de definitie bestand  
;-----
```

```
NAME      Welkom      WINDOWAPI  
DESCRIPTION 'Welkom.c  -- programma HCC Ecs-gg 2018'  
PROTMODE  
STACKSIZE      4096
```

PM Code

Het Make bestand

Welkom.mak

```
.C.obj:  
    @echo " Compile::C++ Compiler "  
    icc.exe /Ti /C /Ss+ %s
```

```
{h:\welkom}.C.obj:  
    @echo " Compile::C++ Compiler "  
    icc.exe /Ss+ /Ti /C %s
```

PM Code

Het Make bestand

```
all: \  
    .\welkom.exe  
  
.\welkom.exe: \  
    .\welkom.obj  
    @echo " Link::Linker "  
    icc.exe @<<  
    /B" /de /pmttype:pm"  
    /Fewelkom.exe  
    .\welkom.obj  
<<  
.\welkom.obj: \  
    h:\welkom\welkom.C
```

Voorbereiding

- Installeer VCC 3.0
- Run \vcc\install.exe
- Toolkit deselecteren
- Installeer

NIET OP NIEUW STARTEN!!!

Vorbereiding

- Modificeer LIBPATH in config.sys
 - zodat x:\IBMCPP\DLL wordt na het einde van het lijn verplaatsen (anders zal de desktop bevrozen tijdens het opstarten)
 - REBOOT
- Installeer de Toolkit 4.52
 - REBOOT

Voorbereiding

- Open de fixpack 308
 - Vervolgens fixpack installeren met deze volgorde:
 - CTC308
CTC308 zal tijdens het updaten een fout genereren als in config.sys het commando
SET LOCPATH=x:\IBMCPPI\LOCALE bestaat.
Oplossing tijdelijk uit remmen
 - CTO308
 - CTW308
 - Reboot
 - CTV308
 - CTD308
 - CTU308
 - reboot

Voorbereiding

In het make program welkom.mak moet alle toewijzingen waarin `h:\welkom\` voorkomt worden veranderd naar het lokaal welkom directory!

VOORBEELD:

De welkom bestanden zijn op jouw computer bewaard in `d:\pmprogram\welkom1\` dan

`h:\welkom\`

wordt

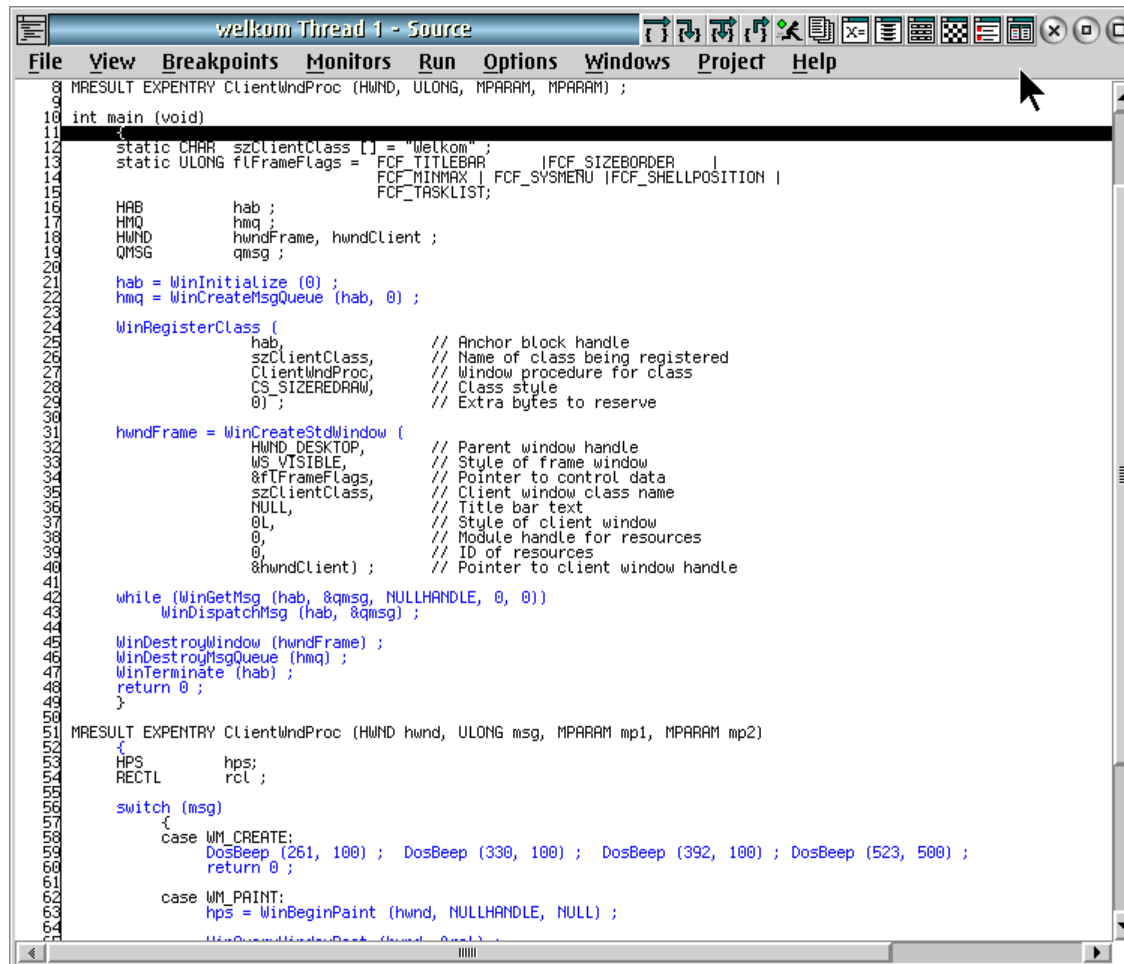
`d:\pmprogram\welkom1\`

in welkom.mak

Oefening 1

- In het template set
 - Bij “Target -> Name”
 - welkom.exe
 - Bij “Location” -> Source directories for project files
 - Het folder dat jij gaat gebruiken for jouw programma
 - Bij Project Smarts deselect “MakeMake”
 - Kopie het inhoud van Oefening 2 (van het netwerk locatie)naar jouw folder

Debuggen



```
1  MRESULT EXPENTRY ClientWndProc (HWND, ULONG, MPARAM, MPARAM) ;
2
3  int main (void)
4  {
5      static CHAR szClientClass [] = "welkom";
6      static ULONG flFrameFlags = FCF_TITLEBAR | FCF_SIZEBORDER |
7                                  FCF_MINMAX | FCF_SYSMENU | FCF_SHELLPOSITION |
8                                  FCF_TASKLIST;
9
10     HAB      hab ;
11     HMQ      hmq ;
12     HWND     hwndFrame, hwndClient ;
13     QMSG     qmsg ;
14
15     hab = WinInitialize (0) ;
16     hmq = WinCreateMsgQueue (hab, 0) ;
17
18     WinRegisterClass (
19         hab,                // Anchor block handle
20         szClientClass,      // Name of class being registered
21         ClientWndProc,      // Window procedure for class
22         CS_SIZEEDRAW,       // Class style
23         0);                 // Extra bytes to reserve
24
25     hwndFrame = WinCreateStdWindow (
26         HWND_DESKTOP,      // Parent window handle
27         WS_VISIBLE,        // Style of frame window
28         &flFrameFlags,     // Pointer to control data
29         szClientClass,     // Client window class name
30         NULL,              // Title bar text
31         0,                 // Style of client window
32         0,                 // Module handle for resources
33         0,                 // ID of resources
34         &hwndClient);      // Pointer to client window handle
35
36     while (WinGetMsg (hab, &qmsg, NULLHANDLE, 0, 0))
37         WinDispatchMsg (hab, &qmsg);
38
39     WinDestroyWindow (hwndFrame);
40     WinDestroyMsgQueue (hmq);
41     WinTerminate (hab);
42     return 0;
43 }
44
45 MRESULT EXPENTRY ClientWndProc (HWND hwnd, ULONG msg, MPARAM mp1, MPARAM mp2)
46 {
47     HPS      hps;
48     RECTL    rcl;
49
50     switch (msg)
51     {
52     case WM_CREATE:
53         DosBeep (261, 100); DosBeep (330, 100); DosBeep (392, 100); DosBeep (523, 500);
54         return 0;
55
56     case WM_PAINT:
57         hps = WinBeginPaint (hwnd, NULLHANDLE, NULL);
58         WinEndPaint (hps);
59     }
```

VCC Debugger

- Demo

Oefening 2

- Compiler en test Welkom programma
- Rem out regel 59 “DosBeep (261, 100)
 - Compileren
 - Testen
 - Wat is de verschil
- Rem out regel 66 WinDrawText(.....
 - Compileren
 - Testen
 - Wat is de verschil