

Tutorial 3

CICS

Ziel dieser Aufgabe ist es, ein "Hello World"-Programm zu schreiben und mittels CICS auf dem Bildschirm auszugeben.

Aufgabe: Arbeiten Sie nachfolgendes Tutorial durch.

Wir erinnern uns: TSO ist ein OS/390-Subsystem. CICS ist ein weiteres OS/390-Subsystem. Jedes der beiden Subsysteme hat eine eigene Benutzerschnittstelle (eine eigene Shell). Um eine CICS-Anwendung zu erstellen, müssen wir mit beiden Subsystemen arbeiten: Mit TSO, um die Anwendung zu erzeugen, und mit CICS, um die Anwendung (unter dem CICS-Subsystem) auszuführen. Da OS/390 ein Multi-User-Betriebssystem ist (multisession-fähig), können wir gleichzeitig eine TSO-Session und eine CICS-Session auf unserem Arbeitsplatzrechner laufen lassen. Jede Session läuft in einem eigenen Fenster.

Wir starten unseren 3270-Emulator zunächst für eine TSO-Session und loggen uns ein.

Wir arbeiten uns zum "Data Set Utility"-Screen vor und erzeugen (Allocate) einen neuen Partitioned Dataset: "PRAKT20.CICS.TEST04".

Außerdem brauchen wir noch einen Partitioned Dataset mit dem vorgegebenen Namen "PRAKT20.LIB", dessen Members von der Entwicklungsumgebung während der CICS-Programmentwicklung mit Daten gefüllt werden.

Dieser Name besteht aus einem "Projekt"-Begriff und einem "Group"-Begriff. Es fehlt der "Type"-Begriff. Wenn wir das Typ-Feld leer lassen, kann es sein, dass TSO dies nicht akzeptiert. Zur Abhilfe tragen wir den Namen 'PRAKT20.LIB' (mit Hochkommas!) in die Zeile "Data Set Name" ein. Damit wird auch dieser Dataset allokiert.

*Aufgabe: Legen Sie die Datasets "PRAKT20.CICS.TEST04" , "PRAKT20.LIB" an (Record format: FB).
Verwenden Sie dazu die gleichen Parameter wie in der Aufgabe zu Tutorial 1.*

Unsere Anwendung besteht aus zwei Programmteilen und einem JCL-Script für die Übersetzung. Wir erstellen diese als Members in dem neuen Partitioned Dataset "PRAKT20.CICS.TEST04".

Ein sauber strukturiertes CICS-Programm besteht aus zwei Teilen: Business Logic und Presentation Logic. Business Logic ist der Teil, in dem Berechnungen erfolgen und Daten in einer Datenbank gelesen/geschrieben werden. Presentation Logic ist der Teil, in dem die Ergebnisse der Berechnungen so aufgearbeitet werden, dass sie dem Benutzer in einer ansprechenden Art auf dem Bildschirm dargestellt werden.

Business Logic wird in Sprachen wie C++, COBOL, PL/1 usw. geschrieben. Für die Presentation Logic gibt es viele Alternativen. Die modernste Alternative benutzt Java Server Pages und einen Web Application Server, um den Bildschirminhalt innerhalb eines Web Browsers darzustellen. Die älteste (und einfachste) Alternative verwendet das CICS BMS (Basic Mapping Support)-Subsystem. BMS-Programme werden in der BMS-Sprache geschrieben. In unserem Beispiel wird die Business Logic in C und die Presentation Logic in BMS geschrieben.

Wir fangen mit dem letzteren an, rufen den "Edit Entry Panel Screen" auf und editieren ein Member "MAP01" für den neu angelegten Partitioned Dataset "PRAKT20.CICS.TEST04" (s. Abbildung 1).

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
EDIT          PRAKT20.CICS.TEST04(MAP01) - 01.02          Columns 00001 00072
***** Top of Data *****
==MSG> -CAUTION- Profile changed to CAPS ON (from CAPS OFF) because the
==MSG>           data does not contain any lower case characters.
==MSG> -Warning-  The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 //PREPARE  JOB ( ),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID
000002 //ASSEM    EXEC DFHMAPS,MAPNAME='S04SET',RMODE=24
000003 //SYSUT1   DD *
000004 S04SET    DFHMSD TYPE=MAP,MODE=INOUT,LANG=C,STORAGE=AUTO,TIOAPFX=YES
000005 *          MENU MAP.
000006 LABEL04   DFHMDI SIZE=(24,80),CTRL=(PRINT,FREEKB)
000007           DFHMDF POS=(9,23),ATTRB=(ASKIP,NORM),LENGTH=34,           X
000008           INITIAL='WELCOME TO THE MAGIC WORLD OF CICS'
000009           DFHMDF POS=(12,27),ATTRB=(ASKIP,NORM),LENGTH=26,         X
000010           INITIAL='MAY THE FORCE BE WITH YOU!'
000011           DFHMSD TYPE=FINAL
000012           END
000013 /*
000014 //
Command ==>
F1=Help      F3=Exit      F5=Rfind     F6=Rchange   F12=Cancel
Scroll ==> PAGE

```

Abbildung 1: Das BMS-Programm

Unser BMS-Programm verwendet 3 Befehlstypen: DFHMSD, DFHMDI und DFHMDF.

Ein BMS-Bildschirm verwendet das 24 Zeilen x 80 Zeichen / Zeile 3270-Bildschirmformat. Eingabe- und Ausgabe-Daten werden als Felder innerhalb der 24x80-Matrix dargestellt, jeweils mit der Angabe: Zeilenadresse, Spaltenadresse und Feldlänge. Dies geschieht mit Hilfe des DFHMDF-Befehls. Der DFHMDF-Befehl in Zeile 000007 definiert ein Feld, welches in Zeile 9, Spalte 23 beginnt, 34 Zeichen lang ist, und mit dem Wert "WELCOME TO THE MAGIC WORLD OF CICS" initialisiert wird. Unser Beispiel-BMS-Programm enthält 2 derartige DFHMDF-Befehle.

Der DFHMSD-Befehl (Zeile 000004) definiert einen "Mapset" mit dem Namen "S04SET". Eine Transaktion involviert in der Regel mehrere unterschiedliche Screens, z.B einen Screen, in dem der Benutzer zu einer Eingabe aufgefordert wird und einen weiteren Screen, welcher die Ergebnisse der Anfrage wiedergibt. Alle Maps (Screens) eines Transaktionstyps werden zu einem Mapset zusammengefaßt.

Die einzelnen Maps (Screens) eines Mapsets werden durch den Befehl DFHMDI definiert und zur Kennzeichnung mit einer Label versehen. In unserem einfachen "Hello World"-Beispiel besteht der Mapset aus einer einzigen Map, die in Zeile 000006 mit der Bezeichnung "LABEL04" definiert wird.

Das Member "MAP01" stellt in Wirklichkeit ein JCL-Script dar. Im Gegensatz zu Tutorial 2 wird das zu verarbeitende File nicht mit INFILE='xxx.yyy.zzz' angegeben. Der JCL-Befehl in Zeile 000003 "///SYSUT1 DD *" besagt, dass das zu verarbeitende File unmittelbar danach folgt (Zeile 000004 bis 000012).

Wir geben auf der Kommandozeile den ISPF-Befehl "SUB" ein. Es wird die Prozedur "DFHMAPS" ausgeführt.

JCL findet das Member "DFHMAPS" (Zeile 000002) in der Library "SYS1.PROCLIB(DFHMAPS)". Durch die Ausführung von "DFHMAPS" werden zwei Ausgabe-Files erzeugt. Einmal wird der übersetzte BMS-Quellcode in einen Member in einer MAPLIB mit dem Namen "CICSTS13.CICS.SDFHLOAD" gestellt. Hier kann ihn die BMS-Komponente des CICS-Subsystems später finden.

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
VIEW          PRAKT20.LIB(S04SET) - 01.00                      Columns 00001 00072
***** ***** Top of Data *****
==MSG> -Warning- The UNDO command is not available until you change
==MSG>          your edit profile using the command RECOVERY ON.
000001 union
000002 {
000003 struct {
000004     char          dfhms1Y12";
000005     } label04i;
000006
000007 struct {
000008     char          dfhms2Y12";
000009     } label04o;
000010
000011 } label04;
000012
***** ***** Bottom of Data *****

Command ==>>
F1=Help      F3=Exit      F5=Rfind     F6=Rchange   F12=Cancel
Scroll ==>> PAGE

```

Abbildung 2: Das Member "S04Set"

Das zweite Ausgabe-File wird als Member "S04SET" in den Partitioned Dataset "PRAKT20.LIB" gestellt (s. Abbildung 2).

Dies ist als Hilfe für die Erstellung des (in C zu schreibenden) Business Logic-Programmes gedacht. Alle Ein- und Ausgabe-Daten, die auf dem Bildschirm wiedergegeben werden sollen, werden ja bereits durch das BMS-Programm definiert. Das Business Logic-Programm bearbeitet diese Daten als C-Strukturen, und diese C-Strukturen werden von DFHMAPS während der Übersetzung von "MAP01" gleich miterzeugt und in "PRAKT20.LIB(S04SET)" abgespeichert.

Beim Erzeugen des Business Logic-Programmes bietet es sich an, "PRAKT20.LIB(S04SET)" als Basis zu benutzen und zu vervollständigen. Hiermit ist sichergestellt (und Fehler ausgeschlossen), dass Presentation Logic und Business Logic identische Datendarstellungen benutzen.

In "PRAKT20.LIB(S04SET)" könnte es sein, dass die beiden Zahlen "12" nicht in eckige Klammern eingeschlossen sind. Anstatt von

"char	dfhms1[12];"	könnte möglicherweise
"char	dfhms1 12 ;"	oder
"char	dfhms1Ý12";"	oder sonstiges auf dem Bildschirm angezeigt werden.

Der Grund ist, dass der 3270-Emulator die hexadecimalen Zahlen "AD" und "BD", die die eckigen Klammern repräsentieren, nicht immer als eckige Klammern darstellt. Je nach eingestellter Host-Zeichenumsetzungstabelle (Host Code Page) werden eckige Klammern auch anders auf dem Bildschirm angezeigt.

Aufgabe: Schreiben Sie das BMS-Programm und übersetzen Sie dieses. Modifizieren Sie die beiden Strings "WELCOME TO THE MAGIC ..." und "MAY THE FORCE ..." so, dass diese Sie als Autor eindeutig identifizieren, z.B. so: "ICH BIN DIE CICS-ANWENDUNG ... VON GRUPPE 4...". (Hinweis: Eine fehlerlose Übersetzung des BMS-Programmes wird durch die Statusmeldung "MAXCC=0" beendet).

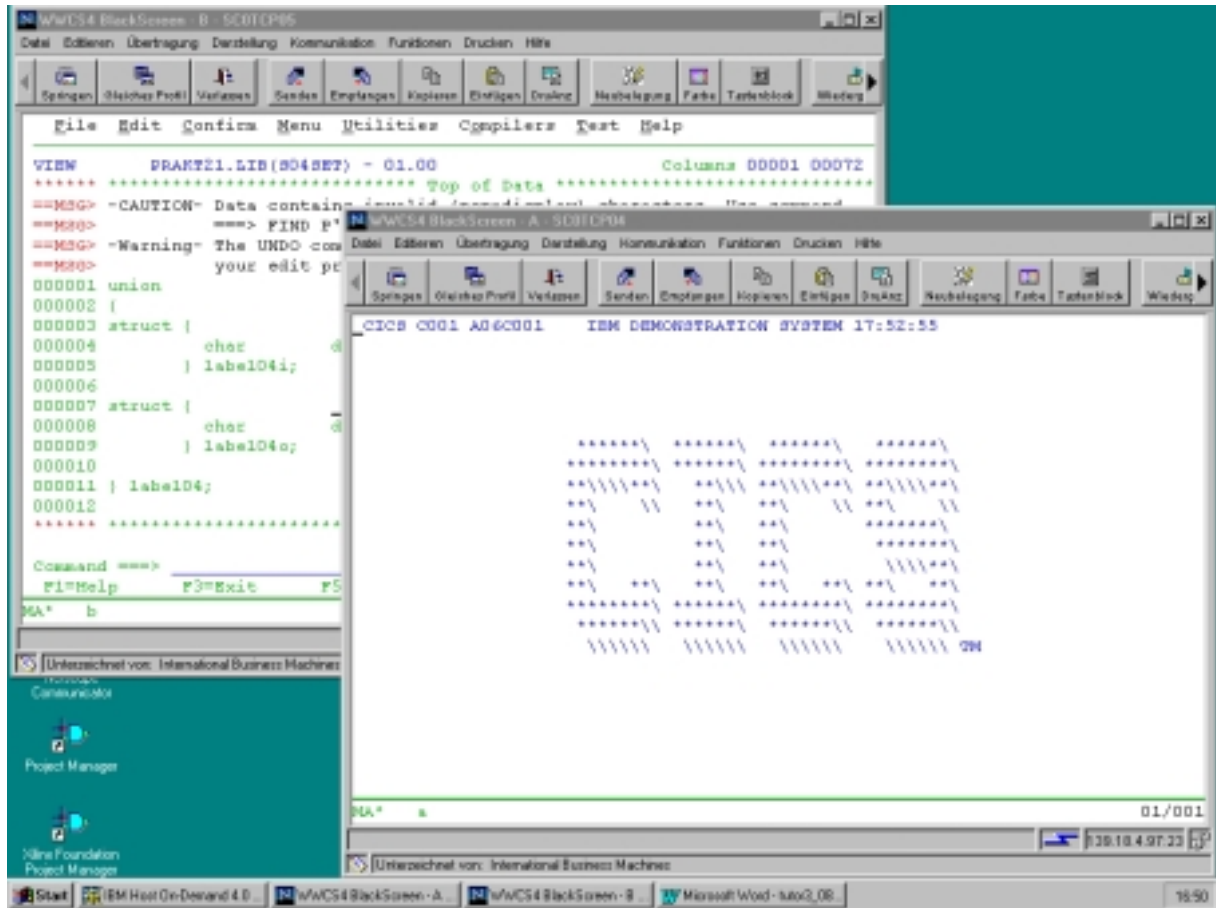


Abbildung 3: 2 TSO-Screens auf dem Desktop

Wir entwickeln unsere Anwendung unter TSO. Wir wollen sie unter CICS laufen lassen. Dazu muß sie als Teil des CICS-Subsystems installiert werden. Es ist komfortabel, mit 2 OS/390-Sessions gleichzeitig zu arbeiten. Das Fenster (s. Abbildung 3) links oben gibt die TSO-Session wieder. Wir starten unter Host On-Demand eine zweite OS/390-Session ("BlackScreen" → "Sitzung starten"). Die zweite Session ist in dem Fenster rechts unten wiedergegeben.


```

CICS C001 A06C001      IBM DEMONSTRATION SYSTEM 24:00:00

          *****\  *****\  *****\  *****\
          *****\  *****\  *****\  *****\
          **\\/\**\  **\\/\  **\\/\**\  **\\/\**\
          **\      \  **\      \  **\      \  **\      \
          **\      \  **\      \  **\      \  *****\
          **\      \  **\      \  **\      \  *****\
          **\      \  **\      \  **\      \  //\\/\**\
          **\      \  **\      \  **\      \  **\      \
          *****\  *****\  *****\  *****\
          *****\  *****\  *****\  *****\
          //\\/\  //\\/\  //\\/\  //\\/\  TM

DFHAC2001 02/04/01 11:01:01 A06C001 Transaction '' is not recognized. Check
that the transaction name is correct.
    
```

Abbildung 6: Fehlermeldung

Eine (belanglose) Fehlermeldung erscheint (s. Abbildung 6). Mit der "Tab"-Taste bewegen wir den Cursor auf die unterste Zeile.

```

CICS C001 A06C001      IBM DEMONSTRATION SYSTEM 24:00:00

          *****\  *****\  *****\  *****\
          *****\  *****\  *****\  *****\
          **\\/\**\  **\\/\  **\\/\**\  **\\/\**\
          **\      \  **\      \  **\      \  **\      \
          **\      \  **\      \  **\      \  *****\
          **\      \  **\      \  **\      \  *****\
          **\      \  **\      \  **\      \  //\\/\**\
          **\      \  **\      \  **\      \  **\      \
          *****\  *****\  *****\  *****\
          *****\  *****\  *****\  *****\
          //\\/\  //\\/\  //\\/\  //\\/\  TM

DFHAC2001 02/04/01 11:01:01 A06C001 Transaction '' is not recognized. Check
that the transaction name is correct. ceda display group(*)
    
```

Abbildung 7: Beispielkommando

CICS erwartet, dass man eine (von vielen) Transaktionen aufruft. Die unterschiedlichen Transaktionen werden normalerweise durch die Eingabe einer aus vier Zeichen bestehenden Transaktions-ID aufgerufen.

Der CICS-Kommandointerpreter ist ebenfalls als Transaktion implementiert. Er wird mit der Transaktions-ID "CEDA" aufgerufen, gefolgt von einer Parameterliste, welche CICS-Kommandos sowie Eingabedaten enthält.

Als Beispiel geben wir das Kommando "ceda display group(*)" gefolgt von der Eingabetaste ein (s. Abbildung 7).

```

display group(*)
ENTER COMMANDS
  GROUP
  AOR2TOR
  ARTT
  ATC
  CBPS
  CEE
  CICREXX
  CSQ
  CSQCKB
  CSQSAMP
  CTA1TCP
  C001EZA
  C001TCP
  DBA1
  DFH$ACCT
  DFH$AFFY
  DFH$AFLA
+ DFH$BABR

                                SYSID=C001 APPLID=A06C001
RESULTS: 1 TO 17                TIME: 00.00.00 DATE: 01.035
PF 1 HELP          3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 8: Auflistung der Gruppen

Wenn unter CICS Anwendungen (Transaktionen) installiert werden, dann wird für jede Transaktion eine "Group" angelegt. In der "Group" befinden sich typischerweise Members wie das Anwendungsprogramm selbst, der dazugehörige Mapset sowie ein Eintrag, der die Transaktion mit einer (normalerweise 4-stelligen) TRID (**TR**ansaktions-**ID**) verknüpft.

Die Liste der bereits installierten Gruppen ist 17 Screens lang (s. Abbildung 8).

```

CEDA DEFINE MAPSET(S04SET) GROUP(PRAKT20)
ENTER COMMANDS
GROUP
AOR2TOR
ARTT
ATC
CBPS
CEE
CICREXX
CSQ
CSQCKB
CSQSAMP
CTAITCP
C001EZA
C001TCP
DBAI
DFH$ACCT
DFH$AFFY
DFH$AFLA
+ DFH$BABR

RESULTS: 1 TO 17
PF 1 HELP          3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
SYSID=C001 APPLID=A06C001
TIME: 00.00.00 DATE: 01.037

```

Abbildung 9: Definition des Mapsets "S04SET" und der Gruppe "PRAKT20"

Wir definieren für unsere Transaktion eine eigene Gruppe "PRAKT20" und den dazugehörigen Mapset als "S04SET". Hierzu überschreiben wir die oberste Zeile, die als Kommandozeile dient, mit dem CEDA-Befehl "CEDA DEFINE MAPSET(S04SET) GROUP(PRAKT20)" (s. Abbildung 9, bitte Großbuchstaben benutzen!)

Um zu bestätigen drücken wir die Eingabetaste.

```

CEDA DEFINE MAPSET(S04SET) GROUP(PRAKT20)
OVERTYPE TO MODIFY
CEDA DEFINE Mapset( S04SET )
Mapset      : S04SET
Group       : PRAKT20
Description ==>
Resident    ==> No                No | Yes
USAge       ==> Normal            Normal | Transient
USElpacopy  ==> No                No | Yes
Status      ==> Enabled           Enabled | Disabled
RSI         : 00                  0-24 | Public

CICS RELEASE = 0530

I New group PRAKT20 created.

SYSID=C001 APPLID=A06C001
TIME: 00.00.00 DATE: 01.037
DEFINE SUCCESSFUL
PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 10: Bestätigung der Erstellung der Gruppe "PRAKT 20"

CICS teilt uns mit, dass die neue Gruppe "PRAKT20" erstellt wurde (s. Abbildung 10). Führen wir nochmals den Befehl "CEDA DISPLAY GROUP(*)" aus, so finden wir in der dargestellten Liste den Eintrag "PRAKT20".

```

CEDA INSTALL GROUP(PRAKT20)
OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFINE Mapset( S04SET )
Mapset      : S04SET
Group       : PRAKT20
Description ==>
Resident    ==> No                No | Yes
USAge       ==> Normal           Normal | Transient
USElpacopy  ==> No                No | Yes
Status      ==> Enabled          Enabled | Disabled
RSI         : 00                  0-24 | Public

I New group PRAKT20 created.

                                SYSID=C001 APPLID=A06C001
DEFINE SUCCESSFUL                TIME: 00.00.00 DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 11: Installation der Gruppe "PRAKT20"

Bis jetzt haben wir dem CICS-Subsystem mitgeteilt, dass eine neue Gruppe "PRAKT20" und in ihr ein Mapset "S04SET" existiert. Als nächster Schritt muß "PRAKT20" in der Anwendungs-Programmbibliothek von CICS installiert werden. Dies geschieht mit dem "INSTALL"-Kommando des CEDA Command Line-Interpreters. Er wird in die oberste Zeile eingegeben und anschließend mit der Eingabetaste bestätigt (s. Abbildung 11).

```

CEDA INSTALL GROUP(PRAKT20)
OVERTYPE TO MODIFY
CEDA Install
All
Connection ==>
DB2Conn ==>
DB2Entry ==>
DB2Tran ==>
DOctemplate ==>
Enqmodel ==>
File ==>
Journalmodel ==>
LSrpool ==>
Mapset ==>
PARTitionset ==>
PARTner ==>
PROcesstype ==>
PROfile ==>
PROgram ==>
+ Requestmodel ==>

                                SYSID=C001 APPLID=A06C001
                                TIME: 00.00.00 DATE: 01.037
INSTALL SUCCESSFUL
PF 1 HELP          3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 12: Erfolgreiche Installation

CEDA teilt mit, dass die Installation erfolgreich war (s. Abbildung 12).

Wir wechseln in das TSO-Fenster zurück und rufen das Member "PRAKT20.LIB(S04SET)" auf (s. Abbildung 13, die Zeichen links und rechts von "12" könnten auch anders auf dem Bildschirm dargestellt sein).

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
VIEW          PRAKT20.LIB(S04SET) - 01.00          Columns 00001 00072
*****      ***** Top of Data *****
==MSG> -Warning- The UNDO command is not available until you change
==MSG>          your edit profile using the command RECOVERY ON.
000001 union
000002 {
000003 struct {
000004     char          dfhms1Y12";
000005     } label04i;
000006
000007 struct {
000008     char          dfhms2Y12";
000009     } label04o;
000010
000011 } label04;
000012
*****      ***** Bottom of Data *****

Command ==>          Scroll ==> PAGE

```

Abbildung 13: Member "S04SET"

Bei der Übersetzung des MAP BMS-Quellcodes wurde für uns als Nebenprodukt im Member "PRAKT20.LIB(S04SET)" ein Template für unser Anwendungsprogramm erzeugt. Wir kopieren den Member in einen neuen Member "PROG04" des Partitioned Datasets "PRAKT20.CICS.TEST04".

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
VIEW          PRAKT20.CICS.TEST04(PROG04) - 01.02          Columns 00001 00072
***** ***** Top of Data *****
==MSG> -CAUTION- Profile changed to NUMBER OFF (from NUMBER ON STD).
==MSG>          Data does not have valid standard numbers.
==MSG> -CAUTION- Profile changed to CAPS OFF (from CAPS ON) because data
==MSG>          contains lower case characters.
==MSG> -Warning- The UNDO command is not available until you change
==MSG>          your edit profile using the command RECOVERY ON.
000001 union
000002 {
000003 struct {
000004     char          dfhms1Y12";
000005     } label04i;
000006
000007 struct {
000008     char          dfhms2Y12";
000009     } label04o;
000010
000011 } label04;
000012
Command ===>
F1=Help      F3=Exit      F5=Rfind     F6=Rchange   F12=Cancel   Scroll ===> PAGE

```

Abbildung 14: Das kopierte Member "S04SET"

Das Template nimmt dort die Zeilen 000001 -000012 ein.

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
VIEW          PRAKT20.CICS.TEST04(PROG04) - 01.02          Columns 00001 00072
000013 main()
000014 {
000015     EXEC CICS SEND MAP("label04") MAPSET("s04set") ERASE;
000016 }
***** ***** Bottom of Data *****

Command ===>
F1=Help      F3=Exit      F5=Rfind     F6=Rchange   F12=Cancel   Scroll ===> PAGE

```

Abbildung 15: Die eingefügte main-Routine

Dann fügen wir in Zeile 000013 - 00016 die main-Routine unseres Anwendungsprogrammes hinzu (s. Abbildung 15). Sie enthält einen einzigen Befehl, einen CICS-Befehl "SEND MAP".

Dieser bewirkt, dass die Map mit der Adresse "LABEL04" aus dem Mapset "S04SET" mit Hilfe des 3270-Protokolls an den Bildschirm des Endbenutzers übertragen wird.

```

File Edit Confirm Menu Utilities Compilers Test Help
-----
EDIT          PRAKT20.CICS.TEST04(START04) - 01.03          Columns 00001 00072
*****      ***** Top of Data *****
==MSG> -CAUTION- Profile changed to NUMBER ON STD (from NUMBER OFF).
==MSG>          Data has valid standard numbers.
==MSG> -CAUTION- Profile changed to CAPS ON (from CAPS OFF) because the
==MSG>          data does not contain any lower case characters.
==MSG> -Warning- The UNDO command is not available until you change
==MSG>          your edit profile using the command RECOVERY ON.
000010 //CICSPRE JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000020 //          TIME=1440
001100 //          EXEC PROC=CTOCICS,REG=0M
001300 //TRN.SYSIN DD DISP=SHR,DSN=PRAKT20.CICS.TEST04(PROG04)
001400 //LKED.SYSIN DD *
001500          NAME PROG04(R)
*****      ***** Bottom of Data *****

Command ==> SUB          Scroll ==> PAGE
F1=Help      F3=Exit      F5=Rfind      F6=Rchange      F12=Cancel

```

Abbildung 16: JCL-File START04

Dieses Programm soll nun übersetzt werden. Dazu wird für den Partitioned Dataset "PRAKT20.CICS.TEST04" ein neues Member "START04" erstellt (s. Abbildung 16). Dies ist ein JCL-File, das die Prozedur CTOCICS (Compile to CICS) enthält. CTOCICS ruft zunächst den CICS-Precompiler auf, der alle CICS-Befehle in C-Befehle übersetzt. Anschließend wird der C-Compiler aufgerufen, der ein Maschinenprogramm erstellt und in eine für das CICS-Subsystem zugängliche Library stellt. "START04" wird mit dem Kommando "SUB" ausgeführt.

Wir geben "SUB" ein und warten, bis der Job ausgeführt wurde (s. Abbildung 16).

Als nächsten Schritt wechseln wir wieder von der TSO- zur CICS-Session.

```

CEDA DEFINE PROGRAM(PROG04) GROUP(PRAKT20)
OVERTYPE TO MODIFY
CEDA Install
All
Connection ==>
DB2Conn ==>
DB2Entry ==>
DB2Tran ==>
DOctemplate ==>
Enqmodel ==>
File ==>
Journalmodel ==>
LSrpool ==>
Mapset ==>
PARTitionset ==>
PARTner ==>
PROcesstype ==>
PROfile ==>
PROgram ==>
+ Requestmodel ==>

                                SYSID=C001 APPLID=A06C001
INSTALL SUCCESSFUL                TIME: 00.00.00 DATE: 01.037
PF 1 HELP          3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 17: Definition des Programmes "PROG04"

Wir definieren für die Gruppe "PRAKT20" unser Anwendungsprogramm als "PROG04", indem wir den entsprechenden CEDA-Befehl

"CEDA DEFINE PROGRAM(PROG04) GROUP(PRAKT20)"

in die oberste Zeile schreiben und anschließend die Eingabetaste betätigen (s. Abbildung 17).

```

CEDA DEFINE PROGRAM(PROG04) GROUP(PRAKT20)
OVERTYPE TO MODIFY
CEDA DEFine Program( PROG04 )
PROGRAM      : PROG04
Group       : PRAKT20
Description  ==>
Language     ==> CObol | Assembler | Le370 | C | Pli
RELoad      ==> No      No | Yes
RESident    ==> No      No | Yes
USAge       ==> Normal  Normal | Transient
USElpacopy  ==> No      No | Yes
Status      ==> Enabled Enabled | Disabled
RSl         : 00      0-24 | Public
CEDf        ==> Yes     Yes | No
Datalocation ==> Below  Below | Any
EXECKey     ==> User   User | Cics
CONcurrency ==> Quasirent Quasirent | Threadsafe
REMOTE ATTRIBUTES
DYNAMIC     ==> No      No | Yes
+ REMOTESystem ==>

                                SYSID=C001 APPLID=A06C001
DEFINE SUCCESSFUL                TIME: 00.00.00 DATE: 01.037
PF 1 HELP 2 COM 3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 18: Definition von "PROG04"

CEDA will einiges von uns wissen (s. Abbildung 18). Wir übernehmen alle Default-Werte und geben als Sprache "Le370" an (s. Abbildung 19).

```

CEDA DEFINE PROGRAM(PROG04) GROUP(PRAKT20)
OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFine PROGram( PROG04  )
  PROGram      : PROG04
  Group        : PRAKT20
  Description   ==>
  Language     ==> Le370                          CObol | Assembler | Le370 | C | Pli
  RELoad       ==> No                             No | Yes
  RESident     ==> No                             No | Yes
  USAge        ==> Normal                         Normal | Transient
  USElpacopy   ==> No                             No | Yes
  Status       ==> Enabled                         Enabled | Disabled
  RSl          : 00                               0-24 | Public
  CEdf         ==> Yes                             Yes | No
  DAtalocation ==> Below                           Below | Any
  EXECKey      ==> User                            User | Cics
  COncurrency  ==> Quasirent                       Quasirent | Threadsafe
REMOTE ATTRIBUTES
  DYnamic      ==> No                             No | Yes
+ REMOTESystem ==>

                                SYSID=C001 APPLID=A06C001
DEFINE SUCCESSFUL                TIME: 00.00.00 DATE: 01.037
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 19: Auswahl der Sprache

Was ist denn "Le370" für eine Sprache? "Le370" ist überhaupt keine Sprache, sondern eine Laufzeitumgebung. CICS braucht an dieser Stelle in Wirklichkeit nicht die Angabe der Quellsprache unseres Anwendungsprogramms (wir haben es ja bereits übersetzt), sondern die Angabe der Laufzeitumgebung des von uns verwendeten Compilers. Alle modernen OS/390-Compiler verwenden eine gemeinsame Laufzeitumgebung, die den Namen "Le370" trägt. Mit Betätigung der Eingabetaste erscheint der nächste Screen.

```

OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFINE PROGRAM( PROG04 )
  PROGRAM      : PROG04
  Group       : PRAKT20
  Description  ==>
  Language    ==> Le370          CObol | Assembler | Le370 | C | Pli
  REload     ==> No             No | Yes
  RESident   ==> No             No | Yes
  USAge      ==> Normal        Normal | Transient
  USElpacopy ==> No             No | Yes
  Status     ==> Enabled       Enabled | Disabled
  RSl        : 00              0-24 | Public
  CEdf       ==> Yes           Yes | No
  DAlocation ==> Below         Below | Any
  EXECKey    ==> User         User | Cics
  COncurrency ==> Quasirent   Quasirent | Threadsafe
REMOTE ATTRIBUTES
  DYNAMIC    ==> No           No | Yes
+ REMOTESystem ==>

                                           SYSID=C001 APPLID=A06C001
DEFINE SUCCESSFUL                          TIME: 00.00.00 DATE: 01.037
PF 1 HELP 2 COM 3 END                      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 20: Erfolgreiche Definition

Nach dem Betätigen der Eingabetaste erscheint der Bildschirm in Abbildung 20. Wir verlassen die Definition mit der Eingabe von F3, als Ergebnis davon erscheint "SESSION ENDED" (s. Abbildung 21).

```

CEDA CEDA DEFINE PROGRAM(PROG04) GROUP(PRAKT20)
STATUS: SESSION ENDED

```

Abbildung 21: Definition wurde beendet

In diesen Bildschirm geben wir in die oberste Zeile den nächsten CEDA-Befehl, gefolgt von der Eingabetaste, ein (s. Abbildung 22).

```
CEDA DEFINE TRANS(TR04) GROUP(PRAKT20)
STATUS:  SESSION ENDED
```

Abbildung 22: Definition der Transaktion TR04

Unsere Transaktion soll wie alle anderen Transaktionen vom Bildschirm über eine 4-stellige Transaktions-ID aufgerufen werden. Wir wählen hierfür die ID "TR04" und teilen diese Wahl mit Hilfe des CEDA DEFINE-Befehls mit (s. Abbildung 22). Genauso wie "PROG04" wird dies Bestandteil der Gruppe "PRAKT20". Abschließend betätigen wir die Eingabetaste.

```
DEFINE TRANS(TR04) GROUP(PRAKT20)
OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFine TRANSAction( TR04 )
  TRANSAction ==> TR04
  Group        ==> PRAKT20
  Description  ==>
  PROgram     ==>
  TWasize     ==> 00000                          0-32767
  PROfile     ==> DFHCICST
  PArtitionset ==>
  SStatus     ==> Enabled                        Enabled | Disabled
  PRIMedsize  : 00000                            0-65520
  TASKDATAloc ==> Below                          Below | Any
  TASKDATAKey ==> User                            User | Cics
  STorageclear ==> No                            No | Yes
  RUnaway     ==> System                          System | 0 | 500-2700000
  SHutdown    ==> Disabled                        Disabled | Enabled
  ISolate     ==> Yes                             Yes | No
  Brexit      ==>
+ REMOTE ATTRIBUTES
  S PROGRAM OR REMOTESYSTEM MUST BE SPECIFIED.
                                                    SYSID=C001 APPLID=A06C001
PF 1 HELP 2 COM 3 END                            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 23: Der Definitions-Screen

CEDA will mehrere Angaben von uns und schlägt eine Reihe von Default-Werten vor (s. Abbildung 23).

```

DEFINE TRANS(TR04) GROUP(PRAKT20)
OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFINE TRANSAction( TR04 )
  TRANSAction  ==> TR04
  Group        ==> PRAKT20
  Description   ==>
  PROGram      ==> PROG04
  TWasize      ==> 00000          0-32767
  PROFile      ==> DFHCICST
  PARTitionset ==>
  STATus       ==> Enabled          Enabled | Disabled
  PRIMedsize   : 00000          0-65520
  TASKDATALoc ==> Below          Below | Any
  TASKDATAKey  ==> User          User | Cics
  STOrageclear ==> No            No | Yes
  RUNaway      ==> System        System | 0 | 500-2700000
  SHutdown     ==> Disabled      Disabled | Enabled
  ISolate      ==> Yes          Yes | No
  BRexit       ==>
+ REMOTE ATTRIBUTES
  S PROGRAM OR REMOTESYSTEM MUST BE SPECIFIED.
                                           SYSID=C001 APPLID=A06C001

PF 1 HELP 2 COM 3 END                      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 24: Eingabe der Parameter

Wir übernehmen alle Default-Werte und geben in die Zeile "PROGram" den Namen unseres Anwendungsprogrammes, nämlich "PROG04" ein und bestätigen mit der Eingabetaste (s. Abbildung 24).

```

OVERTYPE TO MODIFY                                CICS RELEASE = 0530
CEDA DEFINE TRANSAction( TR04 )
  TRANSAction  : TR04
  Group        : PRAKT20
  Description   ==>
  PROGram      ==> PROG04
  TWasize      ==> 00000          0-32767
  PROFile      ==> DFHCICST
  PARTitionset ==>
  STATus       ==> Enabled          Enabled | Disabled
  PRIMedsize   : 00000          0-65520
  TASKDATALoc ==> Below          Below | Any
  TASKDATAKey  ==> User          User | Cics
  STOrageclear ==> No            No | Yes
  RUNaway      ==> System        System | 0 | 500-2700000
  SHutdown     ==> Disabled      Disabled | Enabled
  ISolate      ==> Yes          Yes | No
  BRexit       ==>
+ REMOTE ATTRIBUTES
                                           SYSID=C001 APPLID=A06C001
  DEFINE SUCCESSFUL                          TIME: 22.00.13 DATE: 01.037
PF 1 HELP 2 COM 3 END                      6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 25: Erfolgreiche Definition

Die Meldung "DEFINE SUCESSFUL" erscheint (s. Abbildung 25).

Wir verlassen dieses Menü mit F3.

```
CEDA DEFINE TRANS (TR04) GROUP (PRAKT20)
STATUS:  SESSION ENDED
```

Abbildung 26: Nach dem Verlassen des Menüs

Der Bildschirm in der Abbildung 26 erscheint. Wir haben CICS den Namen unseres Anwendungsprogrammes und eine dazugehörige Transaktions-ID bekanntgegeben. Jetzt müssen diese in die CICS-Programmbibliothek übernommen (installiert) werden.

```
CEDA INSTALL GROUP (PRAKT20)
STATUS:  SESSION ENDED
```

Abbildung 27: Aufruf der Installation

Wir geben in die oberste Zeile das CEDA-INSTALL-Kommando ein und drücken die Eingabetaste (s. Abbildung 27).

```

INSTALL GROUP(PRAKT20)
OVERTYPE TO MODIFY
CEDA Install
All
Connection ==>
DB2Conn ==>
DB2Entry ==>
DB2Tran ==>
DOctemplate ==>
Engmodel ==>
File ==>
Journalmodel ==>
LSrpool ==>
Mapset ==>
PARTitionset ==>
PARTner ==>
PROcesstype ==>
PROFile ==>
PROGram ==>
+ Requestmodel ==>

                                     SYSID=C001 APPLID=A06C001
INSTALL SUCCESSFUL                   TIME: 22.02.15 DATE: 01.037
PF 1 HELP                            3 END                   6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

```

Abbildung 28: Installation war erfolgreich

CEDA bestätigt, dass die Installation erfolgreich war (s. Abbildung 28). Wir verlassen mit F3 dieses Menü.

```

CEDA INSTALL GROUP(PRAKT20)
STATUS: SESSION ENDED

```

Abbildung 29: Beendete Installation

Der obige Bildschirm erscheint (s. Abbildung 29). Unsere Transaktion ist als Teil der CICS-Anwendungsbibliothek installiert worden und kann nun aufgerufen und damit ausgeführt werden. Hierzu löschen wir die oberste Zeile (die CEDA-Kommandozeile) ganz, und rufen unsere Anwendung auf, indem wir dort unsere Transaktions-ID, nämlich "TR04", eingeben.

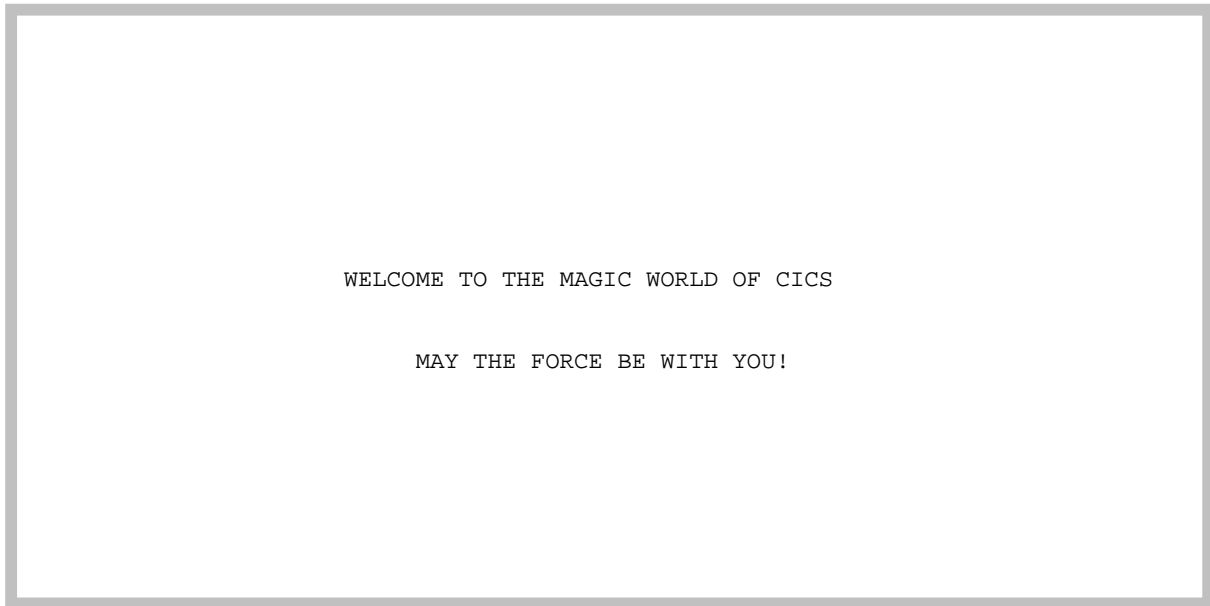


Abbildung 30: Ausgabe des Programmes auf dem Bildschirm

Nach dem Betätigen der Eingabetaste erscheint unsere CICS-Transaktion auf dem Monitor (s. Abbildung 30). Damit ist die Aufgabe gelöst.

Wir betätigen die Eingabetaste, um zum nächsten Screen zu gelangen.

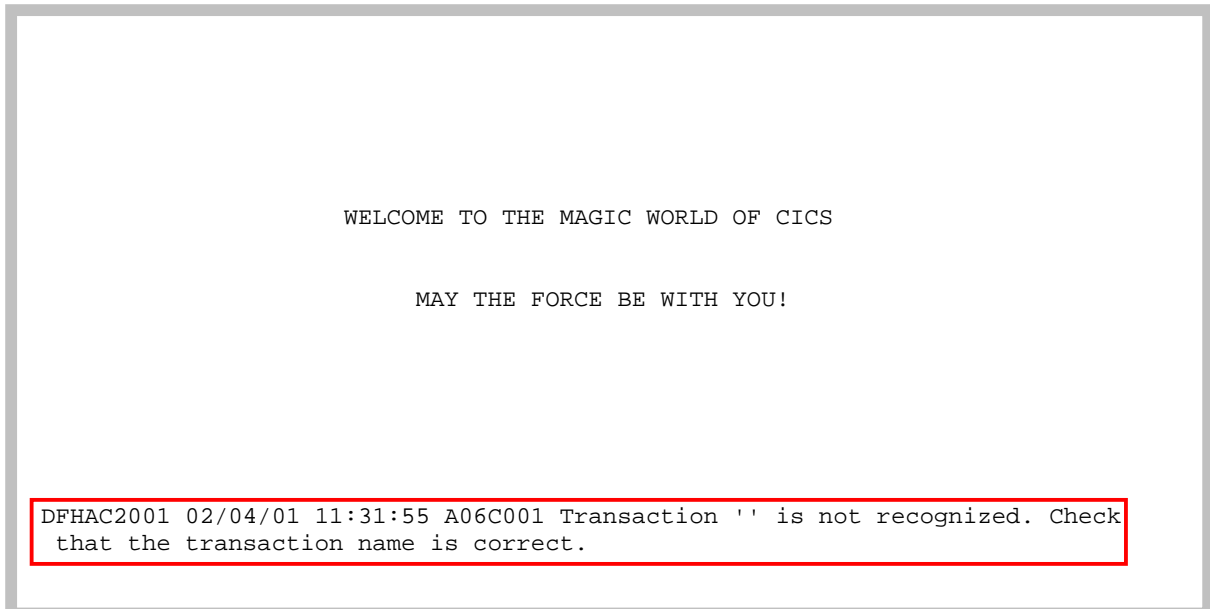


Abbildung 31: Fehlermeldung

Dies terminiert die Bildschirmausgabe unserer Transaktion und erzeugt wieder eine (belanglose) Fehlermeldung (s. Abbildung 31).

```

WELCOME TO THE MAGIC WORLD OF CICS

MAY THE FORCE BE WITH YOU!

DFHAC2001 02/04/01 11:31:55 A06C001 Transaction '' is not recognized. Check
that the transaction name is correct. CEDA DISPLAY GROUP(PRAKT20)

```

Abbildung 32: Befehlseingabe zur Ansicht aller gespeicherter Daten in Group "PRAKT20"

Alle Bestandteile unserer Transaktion sind in der Gruppe PRAKT20 gespeichert. Wir schauen sie uns an, indem wir den Befehl "CEDA DISPLAY GROUP(PRAKT20)" eingeben und anschließend die Eingabetaste drücken (s. Abbildung 32).

```

DISPLAY GROUP(PRAKT20)
ENTER COMMANDS
NAME      TYPE      GROUP      DATE      TIME
S04SET    MAPSET    PRAKT20    01.034    24.00.00
PROG04    PROGRAM   PRAKT20    01.034    24.00.00
TR04      TRANSACTION PRAKT20    01.034    24.00.00

RESULTS: 1 TO 3 OF 3
PF 1 HELP      3 END 4 TOP 5 BOT 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL

SYSID=C001 APPLID=A06C001
TIME: 00.00.00 DATE: 01.035

```

Abbildung 33: Alle gespeicherten Daten in Group PRAKT20

Die Gruppe "PRAKT20" besteht aus den drei Komponenten "S04SET", "PROG04" und "TR04", die wir unter CICS definiert und anschließend installiert haben.

***Aufgabe:** Bereiten Sie die CICS-Transaktion vor und führen Sie diese dann aus. (Hinweis: Die Übersetzung des C-Programmes mittels des JCL-Scriptes "start04" erfolgt fehlerfrei, wenn diese mit der Statusmeldung "MAXCC=4" beendet wird). Benutzen Sie bitte als CICS-Gruppennamen ihren Login-Namen (z.B. "PRAKT20" oder "PRAKT5". Erzeugen Sie per Print-Screen ein Bild ihres Fensters, welches die Bildschirmausgabe von CICS enthält (Ihre Version von "WELCOME TO THE MAGIC...", die Sie eindeutig identifiziert). Achten Sie darauf, dass das Bild nicht mehr als 250 KByte Speicherplatz belegt. Sehr gut ist das JPEG-Format, das mit weniger als 100 KByte auskommt. Schicken Sie mir dieses Bild als Bitmap- oder JPEG-Bild zu. Die Daten ihrer Arbeit löschen Sie bitte nicht.*

Organisatorisches

- Bearbeiten Sie die Aufgaben bis zum 21.12.2001.
- Die Lösung schicken Sie an michaels@informatik.uni-leipzig.de.
- Bei Bedarf sind auch Konsultationen nach Vereinbarung möglich.