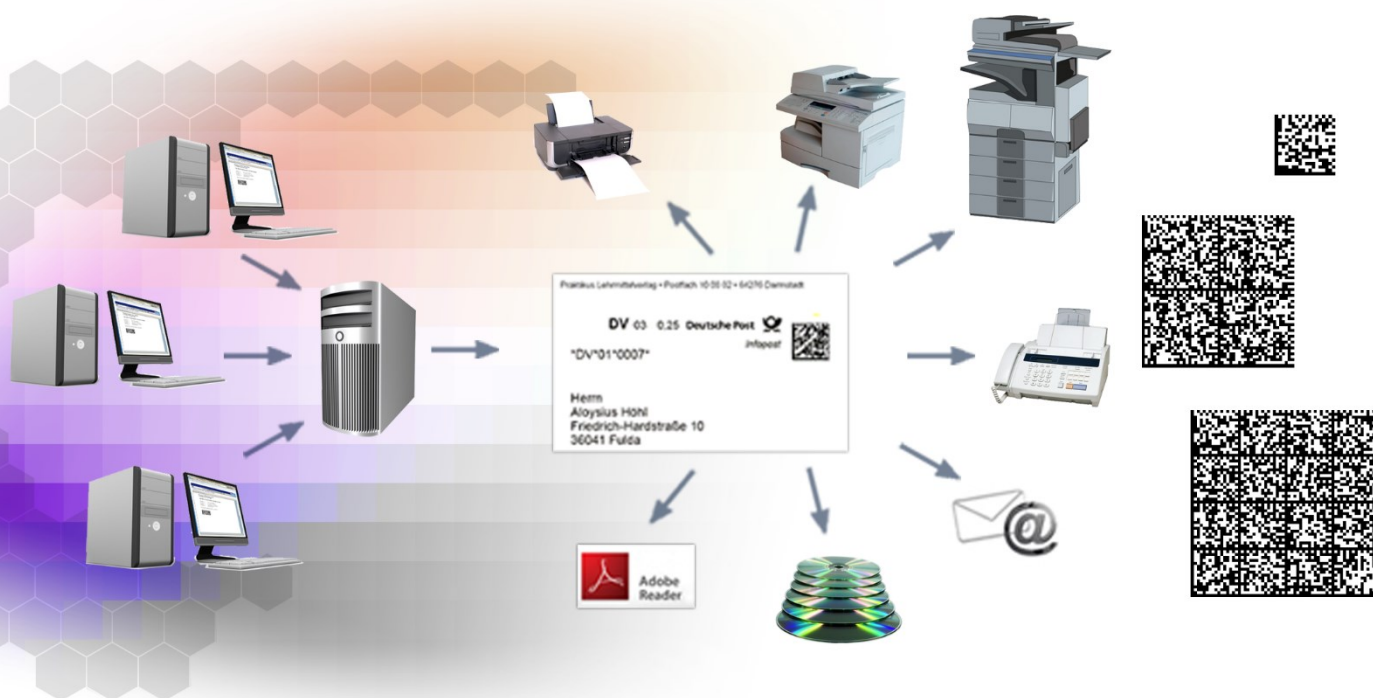




SUCHY MIPS



RBarc/Datamatrix Für SAP® Systeme Version 13 Handbuch

Stand Dezember 2021

© Copyright 2007-2021 Suchy MIPS, München. Alle Rechte vorbehalten.

Diese Dokumentation dient ausschließlich Informationszwecken und kann jederzeit ohne vorherige Ankündigung verändert werden. Suchy MIPS übernimmt keine Gewährleistung oder Garantie hinsichtlich der Richtigkeit und Genauigkeit der Angaben in dieser Dokumentation.

Diese Dokumentation oder Teile daraus dürfen nicht vervielfältigt, in Datenbanken gespeichert oder in irgendeiner Form elektronisch, fotomechanisch, auf Datenträgern oder auf irgendeine andere Weise übertragen werden, ohne dass vorher die schriftliche Zustimmung von Suchy MIPS eingeholt wurde. Verstöße gegen das Copyright ziehen zivil- und strafrechtliche Konsequenzen nach den internationalen Copyright Gesetzen nach sich.

Die Dokumentation beschreibt Installation und Anwendung einer Copyright geschützten Software. Die Benutzer sind verantwortlich für die Einhaltung aller einschlägigen Urheberrechtsgesetze und der jeweiligen Lizenzvereinbarung.

RBarc/Datamatrix ist geistiges Eigentum der Fa. Suchy MIPS.

MS-Windows ist eine eingetragene Marke der Microsoft Corporation, Inc.

R/3, SAP, mySAP and Netweaver sind eingetragene Marken der SAP AG.

Alle verwendete Firmen-, Markennamen und Warenzeichen sind Eigentum der jeweiligen Inhaber und dienen lediglich zur Identifikation und Beschreibung der Produkte und Dienstleistungen. Eine Haftung für Richtigkeit, Vollständigkeit und Aktualität wird nicht übernommen.

Suchy MIPS GmbH
Prinzregentenstraße 128

81677 München
Deutschland

Tel.: +49 (0) 89 - 944 19 77 - 0

Fax: +49 (0) 89 - 944 19 77 - 13

E-Mail: info@suchymips.de

Internet: www.suchymips.de

Inhaltsverzeichnis

Nr.	Titel	Seite
1	Einführung	3
2	Systemvoraussetzungen	4
3	Installation	5
4	Unterschiede zwischen der Demo- und Vollversion von RBarc/Datamatrix	6
5	Testen der Installation von RBarc/Datamatrix	7
5.1	Testen mit SAPscript.....	7
5.2	Testen mit Smart Forms.....	8
5.3	Testen mit Interactive Forms.....	9
6	Datamatrix in Formulare einbinden	10
6.1	Das Funktionsprinzip von RBarc/Datamatrix.	10
6.2	Typisierung der Variablen	11
6.3	Optionale Parameter zur Definition der Barcode Eigenschaften.	12
6.4	Datamatrix Druck mit SAPscript.....	14
6.4.1	Barcode Definition im SAPscript Formular.....	14
6.5	Datamatrix Druck mit Smart Forms.....	18
6.5.1	Barcode Definition im Smart Forms Formular.....	18
6.6	Datamatrix Druck mit Interactive Forms.....	24
6.6.1	Erweiterung der Interactive Forms Schnittstelle	24
6.6.2	Erweiterung des Interactive Forms Formulars	28
6.7	Definition der Barcode Eigenschaften.....	31
7	Beispiel für eine Aufgabenstellung in einem SAPscript Formular	40
7.1	Aufgabe	40
8	Beispiel für Aufgabenstellung in einem Smart Forms Formular.	42
8.1	Aufgabe	42
9	Anhang 1: Neues Paket (Entwickungsklasse) erstellen.....	44
10	Anhang 2: Ein Include in der ABAP Workbench anlegen.	46
11	Anhang 3: Ein Programm in der ABAP Workbench anlegen.	49
12	Anhang 4: Inhalt der Installationsdateien in ABAP Programme einfügen.....	52
13	Anhang 5: Aktivieren von Includes und Programmen.....	53
14	Anhang 6: Ein ABAP Programm ausführen.	55
15	Anhang 7: Ein SAPscript Formular importieren (hochladen).	56
16	Anhang 8: Ein Smart Forms Formular hochladen.....	58
17	Anhang 9: Smart Forms Formular testen.....	60
18	Anhang 10: Eine Interactive Forms Schnittstelle hochladen.	62
19	Anhang 11: Ein Interactive Forms Formular hochladen.....	63
20	Anhang 12: Ein Interactive Forms Formular testen	64

1 Einführung

RBarc/Datamatrix ist ein ABAP Programm zur on-the-fly Erzeugung des 2D Barcodes Datamatrix auf SAP Systemen (R/3, MySAP, ERP usw.). Die entsprechenden Funktionen stehen in **SAPscript**-, **Smart Forms**- und **Interactive Forms** Formularen zur Verfügung.

RBarc/Datamatrix ist voll kompatibel zur GS1 DATAMATRIX ECC 200 Spezifikation

Die mit **RBarc/Datamatrix** erzeugten 2D Barcodes werden innerhalb des SAP Systems so erzeugt, dass sie integraler Bestandteil des auszugebenden Dokumentes werden. Die daraus resultierenden Vorteile sind, dass Dokumente mit Barcodes auf beliebigen Druckern ausgegeben können - unabhängig vom Hersteller und der verwendeten Druckersprache. Hardwareerweiterungen werden nicht benötigt. Ein weiterer Vorteil ist, dass Barcodes auf den Dokumenten immer sichtbar sind, auch wenn das Dokument gefaxt, gemailt, in ein PDF-Dokument umgewandelt oder archiviert wurde.

Wichtig:

RBarc/Datamatrix ändert den SAP Standard nicht. Die zu **RBarc/Datamatrix** gehörenden Programme beginnen mit dem Buchstaben "Z" und werden im sog. Kundenraum installiert. Somit werden keine wichtigen Programmteile bei eventuellen Updates überschrieben.

2 Systemvoraussetzungen

RBarc/Datamatrix setzt ein installiertes SAP® System R/3 vers. 3.x oder höher voraus (auch MySAP ERP und NetWeaver).

Wichtig:

RBarc/Datamatrix ist für Administratoren (Installation) und Entwickler (Barcode Gestaltung und Einbindung ins Formular) bestimmt. Konsequenterweise sind gute Kenntnisse in Basis bei der Installation Voraussetzung. Für das Einbinden von Barcodes in **SAPscript**-, **Smart Forms**- oder **Interactive Forms**-Formulare sollten mindestens Grundkenntnisse in dem jeweiligen Formularverfahren als auch in ABAP Programmierung vorhanden sein.

Alle Anwender, die Dokumente mit von **RBarc/Datamatrix** erzeugten Barcodes drucken, müssen folgende Berechtigung für das Objekt **S_BDS_DS** besitzen:

ACTVT = 01, 02, 03 und 06
CLASSNAME = DEVC_STXD_BITMAP
CLASSTYPE = OT

Die entsprechenden Einstellungen können ggf. mit der Transaktion "**PFCG**" durchgeführt werden. Die User Rolle muss die Transaktion SE78 und die Autorisierung für **BC-SRV-KPR-BDS** (Technischer Name **S_BDS_DS**) enthalten. **S_BDS_DS** gehört zu den **Basis-Central-Funktionen**.

Bei fehlenden Rechten des Benutzers erscheint kein Barcode auf dem Ausdruck oder Barcodes werden aus dem System nicht gelöscht.

3 Installation

Die Installationsschritte werden hier in Kurzform beschrieben. Falls Sie mit manchen Vorgängen nicht vertraut sind (z.B. Anlegen eines Programms), dann finden Sie bei jedem Schritt einen Verweis auf die Seite, auf der der Vorgang genauer beschrieben wird. Um zur detaillierten Beschreibung automatisch zu springen, klicken Sie mit der linken Maustaste auf den Querverweis.

Das Programm besteht aus einem ABAP-Programm (Report) und einem ABAP Include. Beide Dateien liegen im Verzeichnis „**Install**“.

Die Testformulare für **Smart Forms**, **SAPscript** und **Interactive Forms** befinden sich in entsprechenden Unterverzeichnissen: „**Test-Smartforms**“, „**Test-SAPscript**“, „**Test-Interactiveforms**“..

1. Starten Sie das **SAP GUI** und loggen sich auf dem SAP System als Administrator an.
2. Starten Sie den **Objekt Navigator** (Transaktion **SE80**) und erstellen ein neues Paket (Entwickungsklasse) mit dem Namen **ZDATAMATRIX** (empfohlen).
(Gehe zu Seite 44: [Anhang 1: Neues Paket \(Entwickungsklasse\) erstellen](#))
3. Legen Sie in dem angelegten Paket (Entwickungsklasse) **ZDATAMATRIX** ein Include mit dem Namen **ZDATAMATRIX** an und löschen den automatisch erzeugten Inhalt.
(Gehe zu Seite 46: [Anhang 2: Ein Include in der ABAP Workbench anlegen](#))
4. Kopieren Sie in das neu erstellte Include **ZDATAMATRIX** den Inhalt der Datei **ZDATAMATRIX.INC** aus dem Verzeichnis **Install**, sichern und aktivieren es.
(Gehe zu Seite 52: [Anhang 4: Inhalt der Installationsdateien in ABAP Programme einfügen](#))
5. Legen Sie in dem Paket **ZDATAMATRIX** ein Programm (Report) mit dem Namen **Z_SET_DATAMATRIX** an und löschen den ev. automatisch erzeugten Inhalt.
(Gehe zu Seite 44: [Anhang 3: Ein Programm in der ABAP Workbench anlegen](#))
6. Kopieren Sie in das neu erstellte Programm **Z_SET_DATAMATRIX** den Inhalt der Datei **Z_SET_DATAMATRIX.PRG** aus dem Verzeichnis **Install**, sichern das Programm und aktivieren es.

Damit ist die Installation von RBarc/Datamatrix abgeschlossen.

Jetzt können noch Testobjekte installiert werden, mit denen je ein Smartforms, Interactiveforms und Sapscript Formular mit einem Datamatrix Barcode gedruckt werden können.

7. Legen Sie in dem Paket **ZDATAMATRIX** ein Programm mit dem Namen **ZSF_DM_PRINT** an und löschen Sie den ev. automatisch erzeugten Inhalt.
8. Kopieren Sie in das neu erstellte Programm **ZSF_DM_PRINT** den Inhalt der Datei **ZSF_DM_PRINT.PRG** aus dem Verzeichnis **Test-Smartforms**, sichern und aktivieren es.
9. Legen Sie in dem Paket **ZDATAMATRIX** ein Programm mit dem Namen **ZIF_DM_PRINT** an und löschen Sie den ev. automatisch erzeugten Inhalt.
10. Kopieren Sie in das neu erstellte Programm **ZIF_DM_PRINT** den Inhalt der Datei **ZIF_DM_PRINT.PRG** aus dem Verzeichnis **Test-Interactiveforms**, sichern und aktivieren es.
11. Legen Sie in dem Paket **ZDATAMATRIX** ein Programm mit dem Namen **ZSS_DM_PRINT** an und löschen Sie den ev. automatisch erzeugten Inhalt.
12. Kopieren Sie in das neu erstellte Programm **ZSS_DM_PRINT** den Inhalt der Datei **ZSS_DM_PRINT.PRG** aus dem Verzeichnis **Test-Sapscript**, sichern und aktivieren es.

(Gehe zu Seite 53: [Anhang 5: Aktivieren von Includes und Programmen](#))

4 Unterschiede zwischen der Demo- und Vollversion von RBarc/Datamatrix

- Die maximale Matrixgröße in der Demoversion beträgt 14 x 14 (Parameter msize = 3), das sind 8 Codewords, die kodiert werden können. Mit 8 Codewords lassen sich z.B. max. 16 Ziffern oder 10 Buchstaben kodieren. Im gemischten Modus oder bei Verwendung von Sonderzeichen, wie z.B. GS (Group Separator) ist die verschlüsselbare Datenmenge nicht einfach vorhersehbar, da beim Wechsel von Alphabet zu Alphabet ein gewisser Datenoverhead entsteht, deren Menge nicht nur von den verwendeten Alphabeten sondern auch von der Verteilung der Zeichen abhängt.
- Die Ziffer „1“ wird durch eine „0“ ersetzt und umgekehrt.

Wichtig:

Falls Sie Tests mit echten Daten durchführen möchten, kontaktieren Sie uns um eine Teststellung zu vereinbaren. Sie haben die Möglichkeit die Software 4 Wochen lang zu testen und genießen in dieser Zeit den vollen technischen Support.

5 Testen der Installation von RBarc/Datamatrix

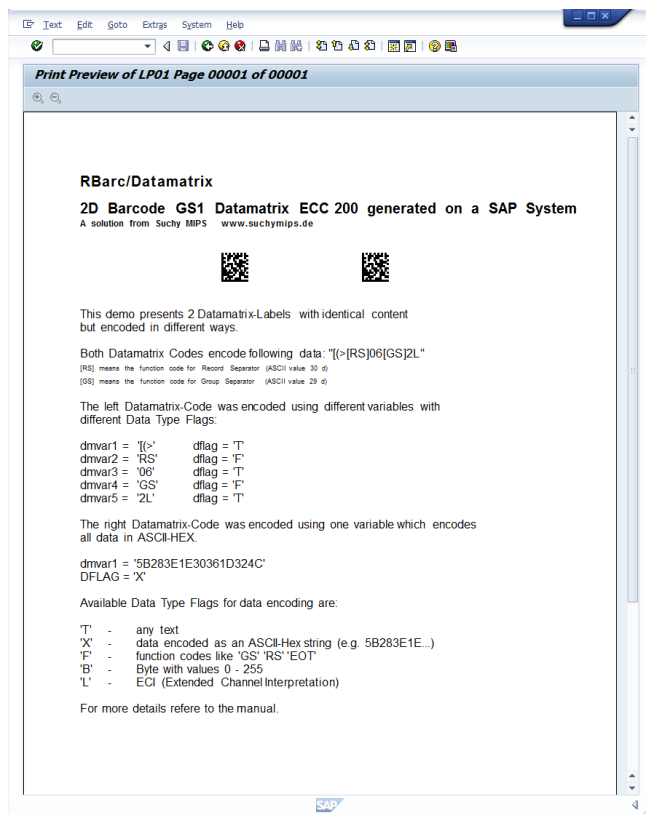
In den Verzeichnissen **Test-Smartforms**, **Test-Interactiveforms** und **Test-Sapscript** befinden sich Test-Formularobjekte und dazugehörige Druckprogramme (Reprints).

5.1 Testen mit SAPscript

1. Starten Sie das SAP Standardprogramm **RSTXSCR**P und importieren das **SAPscript** Formular **ZSS_DM_FORM** aus der Datei **ZSS_DM_FORM.FOR**.
(Gehe zu Seite 56: [Anhang 7: Ein SAPscript Formular importieren \(hochladen\)](#))
2. Um das **SAPscript** Testformular **ZSS_DM_FORM** zu drucken, führen Sie das Programm **ZSS_DM_PRINT** aus.
(Gehe zu Seite 55: [Anhang 6: Ein ABAP Programm ausführen](#)).

Das Programm **ZSS_PRINT** druckt das **SAPscript** Formular **ZSS_DM_FORM**.

Sie können das Formular auf dem Drucker ausgeben, oder in der Druckvorschau ansehen. Das Ergebnis sollte in etwa wie folgt aussehen:



Um einen Barcode in ein eigenes SAPscript Formular einzubinden, lesen Sie weiter auf Seite 14 unter [Barcode Definition im SAPscript Formular](#)

5.2 Testen mit Smart Forms

1. Starten Sie die Transaktion „**SmartForms**“ und laden Sie das **Smart Forms** Formular **ZSF_DM_FORM** aus der Datei **zsf_dm_form.xml** hoch. Sichern und aktivieren Sie das Formular.
(Gehe zu Seite 58: [Anhang 8: Ein Smart Forms Formular hochladen](#)).

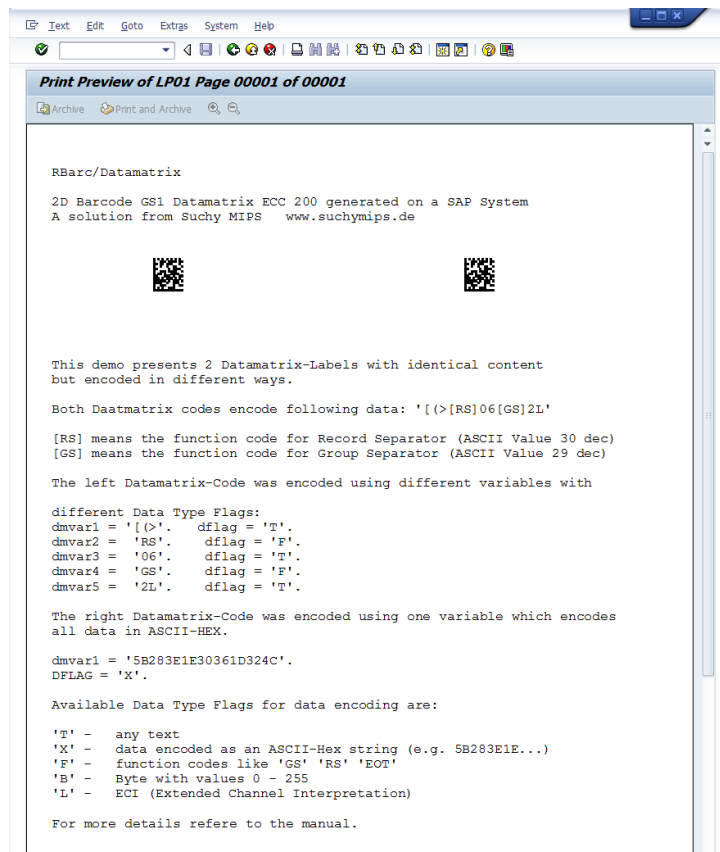
Achtung:

Diese Funktion ist ab R/3 Vers. 4.7 verfügbar. Bei R/3 Vers. 4.6x müssen Sie ein **Smart Forms** Formular von Hand erstellen, siehe dazu: Gehe zu Seite 18: [Datamatrix Druck mit Smart Forms](#).

3. Um das **Smart Forms** Testformular **ZSF_DM_FORM** zu drucken, führen Sie das Programm **ZSF_DM_PRINT** aus.
(Gehe zu Seite 55: [Anhang 6: Ein ABAP Programm ausführen](#)).

Das Programm **ZSF_PRINT** druckt das **Smart Forms** Formular **ZSF_DM_FORM**.

Sie können das Formular auf dem Drucker ausgeben, oder in der Druckvorschau ansehen. Das Ergebnis sollte in etwa wie folgt aussehen:



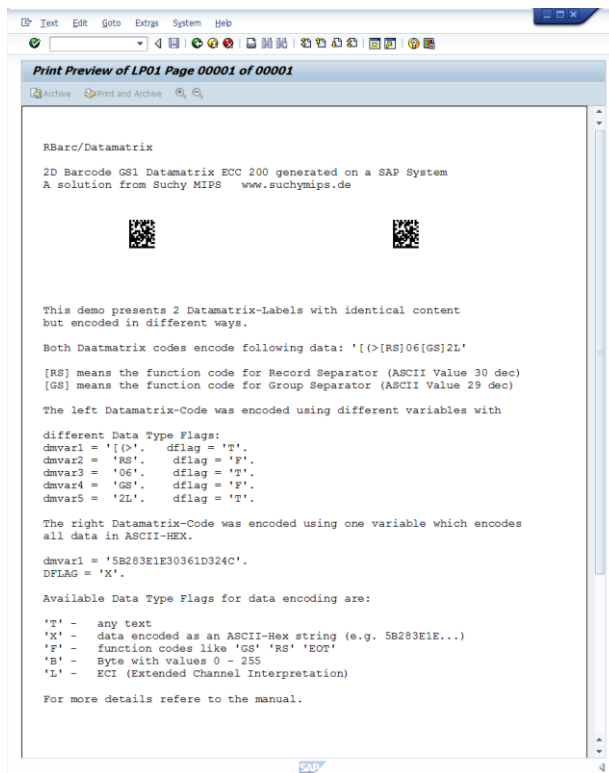
Um einen Barcode in ein eigenes **Smart Forms** Formular einzubinden, lesen Sie weiter unten, unter: „**Drucken von Barcodes aus Smart Forms**“ (Gehe zu Seite 18: [Datamatrix Druck mit Smart Forms](#))

5.3 Testen mit Interactive Forms

1. Starten Sie die Transaktion „SFP“ und laden die **Interactive Forms** Schnittstelle **ZIF_DM_INTERFACE** aus der Datei **zif_dm_interface.xml** hoch. Sichern und aktivieren Sie die Schnittstelle.
(Gehe zu Seite 62: [Anhang 10: Eine Interactive Forms Schnittstelle hochladen](#))
2. Starten Sie die Transaktion „SFP“ und laden das **Interactive Forms** Formular **ZIF_DM_FORM** aus der Datei **zif_dm_form.xml**. Sichern und aktivieren Sie das Formular.
(Gehe zu Seite 63: [Anhang 11: Ein Interactive Forms Formular hochladen](#))
4. Um das **Interactive Forms** Testformular **ZIF_DM_FORM** zu drucken, führen Sie das Programm **ZIF_DM_PRINT** aus.
(Gehe zu Seite 55: [Anhang 6: Ein ABAP Programm ausführen](#)).

Das Programm **ZIF_PRINT** druckt das **Interactive Forms** Formular **ZIF_DM_FORM**.

Sie können das Formular auf dem Drucker ausgeben, oder in der Druckvorschau ansehen. Das Ergebnis sollte in etwa wie folgt aussehen:



Um einen Barcode in ein eigenes **Interactive Forms**-Formular einzubinden, lesen Sie weiter unten, unter: „Drucken von Barcodes aus **Interactive Forms**“ (Gehe zu Seite:24 [Datamatrix Druck mit Interactive Forms](#))

6 Datamatrix in Formulare einbinden

Datamatrix ist ein sog. 2D Barcode, mit dem eine große Anzahl von Zeichen kodiert werden kann (bis zu 3116 numerische oder 2335 alphanumerische Zeichen oder 1556 Bytes). In ERP Anwendungen handelt es sich dabei meistens nicht um fließenden Text, sondern um Inhalte bestimmter Datenfelder aus der Datenbank, die in einem Barcode Label kodiert werden sollen, womöglich mit Feldseparatoren voneinander getrennt und mit einem Daten Vor- und Nachspann versehen. Im Folgenden wird das Funktionsprinzip von **RBarc/Datamatrix** als auch die Regeln für die Implementierung von Datamatrix Barcodes in den verschiedenen SAP Formulartechnologien: **SAPscript**, **Smart Forms** und **Interactive Forms** beschrieben.

Hilfreiche Beispiele mit Aufgabenstellung aus der Praxis und Implementierungslösungen finden Sie in den folgenden Kapiteln:

Beispiel für eine Aufgabenstellung in einem SAPscript Formular

Beispiel für eine Aufgabenstellung in einem SAPscript Formular

6.1 Das Funktionsprinzip von RBarc/Datamatrix.

Das Funktionsprinzip von **RBarc/Datamatrix** sieht wie folgt aus:

- Alle Variablen, die im Datamatrix Barcode kodiert werden sollen, werden typisiert und in eine interne Schnittstellentabelle paarweise (Variable / Typ) geschrieben. Die Typisierung der Variablen wird im nächsten Abschnitt beschrieben.
- Optional werden Parameter für Barcode Eigenschaften, wie z.B. **Modulgröße**, **Matrixgröße** oder **Grafikauflösung** in die Schnittstellentabelle geschrieben. Die optionalen Parameter werden im übernächsten Abschnitt beschrieben. Wenn optionale Parameter nicht definiert werden, dann werden für diese Parameter Standardwerte automatisch von **RBarc/Datamatrix** gesetzt.
- Die Schnittstellentabelle mit den zu kodierenden Variablen, den dazugehörigen Typen als auch eventuell definierten optionalen Parametern wird an eine Formroutine des Programms **Z_SET_DATAMATRIX** übergeben. Der Namen der Formroutine ist anders für jede SAP-Formulartechnologie und lautet:

für SAPscript : **GEN_DATAMATRIX_SS**
für Smart Forms: **GEN_DATAMATRIX_SF**
für Interactive Forms: **GEN_DATAMATRIX_IF**

- Während der Laufzeit erzeugt das Programm **Z_SET_DATAMATRIX** den Barcode dynamisch und übergibt dessen einmaligen, dynamisch erzeugten Namen bzw. den Barcode als binäres Objekt an das aufrufende Formular.
- Im Formular wird der Barcode dynamisch eingebunden.
- Anschließend wird der Barcode aus dem System gelöscht.

Bemerkung:

Die Übergabe aller Parameter über eine Schnittstellentabelle erlaubt dass Kodieren einer beliebigen Anzahl von Variablen, ohne dass jedes mal die Schnittstelle angepasst werden müsste.

Die Schnittstelle zwischen dem Formular und dem Programm **Z_SET_DATAMATRIX** ist vom Prinzip her für alle SAP Formulartechnologien gleich. Technologiebedingt gibt es jedoch Implementierungsunterschiede zwischen **SAPscript**, **Smart Forms** und **Interactive Forms**. Deshalb werden Einzelheiten der Implementierung in den entsprechenden Kapiteln für die jeweilige Formulartechnologien beschrieben.

6.2 Typisierung der Variablen

Der Barcode Datamatrix kann alle ASCII Werte im Bereich von 0 – 255 (dezimal) kodieren. Da nicht all diese Werte in einem SAP System als Zeichen darstellbar sind, entwickelten wir ein spezielles Verfahren, das Ihnen erlaubt alle Zeichenwerte zu kodieren. Dazu müssen die Variablen typisiert werden, bevor sie in die Schnittstellentabelle geschrieben werden. Dies geschieht mit dem Parameter "**dflag**". Folgende Indikatoren (Werte für "**dflag**") sind gültig:

'T' – jede alfanumerische Variable, wie z.B. 'ABCDabcd12345AbCd'.

'X' – Variablen im ASCII HEX Format 0x00 – 0xFF. Beispiel: die Variable '**123ABC**' mit **dflag = 'T'** könnte in Form von '**303132414243**' mit **dflag = 'X'** übergeben werden. Auf diese Weise lassen sich auch Zeichen kodieren, die sonst nicht darstellbar sind, wie z.B. eine binäre 0 als ASCII-HEX '00'.

'Z' – Variablen im ASCII HEX Format 0x00 – 0xFF. Alle Werte werden im sog. Base 256 Modus (1 Codeword pro Byte) kodiert.

'Y' – Variablen im ASCII Format Alle Zeichen werden im sog. Base 256 Modus (1 Codeword pro Byte) kodiert.

'F' – Variable ist ein Funktionszeichen. Es kann nur eine Variable dieses Typs pro Tabellenzeile übergeben werden. Wenn mehrere Zeichen dieses Typs nebeneinander kodiert werden müssen, muss jedes Zeichen einzeln mit dem dazugehörigen Typ (dflag) in die Schnittstellentabelle eingetragen werden. Folgende Funktionszeichen werden unterstützt:

- ASCII Zeichen 0 bis 31:
'NUL','SOH','STX','ETX','EOT','ENQ','ACK','BEL','BS','HT',
'LF','VT','FF','CR','SO','SI','DLE','DC1','DC2','DC3','DC4',
'NAK','SYN','ETB','CAN','EM','SUB','ESC','FS','GS','RS','US'.
- Das EAN Funktionszeichen: '**FNC1**'
- Das "Extended Channel Interpretation" Zeichen: '**ECI**'
- Macro 05 für den Industrie Header/Trailer: [(>RS05GS RSEOT: '**MAC05**'.
- Macro 06 für den Industrie Header/Trailer [(>RS06GS RSEOT: '**MAC06**'.

'B' – Variable stellt einen Binärwert in Dezimalformat dar. Es können Werte von 0 bis 255 übergeben werden. Es darf nur ein Wert pro Schritt übergeben werden. Wenn mehrere Zeichen dieses Typs nebeneinander kodiert werden müssen, muss jedes Zeichen einzeln, zusammen mit dem dazugehörigen Typ (dflag) übergeben werden. Auch mit dieser Methode lassen sich - ähnlich wie beim Typ 'X' alle ASCII Werte von 0 bis 255 verschlüsseln.

'L' – Die Variable stellt einen sog. "Extended Channel Interpretation" Wert da. Jede Variable dieser Art muss mit 'ECI' beginnen, gefolgt von einer 6-stelligen Nummer. Der ECI bleibt gültig, bis nicht ein anderer ECI gesendet wird oder bis zum Ende der Daten. Gültige ECI Nummern sind 0 bis 999999. Der Standard ECI Wert beträgt 000003 (ASCII für Zeichen 0 bis 127 und ISO 8859-1 für Zeichen 128 bis 255. Mit dem ECI wird eine bestimmte Zeicheninterpretation erzwungen (Standard = Westeuropäisch). Bitte beachten Sie, dass ECI nur von speziellen Scannern verarbeitet werden kann. Eine genaue Beschreibung des ECI Protokolls finden Sie in dem Dokument "*Extended Channel Interpretation (ECI) Assignments*". Beispiel: ECI000978.

6.3 Optionale Parameter zur Definition der Barcode Eigenschaften.

Optionale Parameter dienen der Steuerung von äußeren Barcode Eigenschaften. Dazu gehören die **Größe des Barcodes**, die **Größe der Matrix**, und die **Auflösung der Grafik**. Wird ein Parameter nicht definiert, dann wird er automatisch auf den Standardwert gesetzt.

MSIZE Größe der Matrix gemäß ISO-Spezifikation. Die ISO-Norm spezifiziert 30 Ausprägungen des Datamatrix Barcodes, die hier mit **msize** gewählt werden können. Normalerweise versucht man die zu kodierenden Daten in einer möglichst kleinen Matrix unter zu bringen. In manchen Anwendungen ist es jedoch erforderlich, eine konstante Barcode Größe zu erhalten. In diesem Fall muss ein Wert für **msize** vorgegeben werden. Der Wert sollte so gewählt werden, dass möglichst alle in Frage kommenden Daten darin Platz haben. Erweist sich die gewählte Matrixgröße während der Laufzeit als zu klein wird eine Fehlermeldung ausgegeben und der Barcode wird nicht erzeugt.

Zulässige Werte: 0 bis 30.

Standardwert: **0** (automatische Berechnung der kleinstmöglichen Matrix).



msize = 4 (16x16)



msize = 11 (36x36)

Beide Barcodes verschlüsseln den gleichen Text "Suchy MIPS"

Eine vollständige Auflistung aller Matrixgrößen finden Sie im Kapitel "Definition der Barcode Eigenschaften"

RES Angabe der Auflösung, mit der die Barcode Grafik generiert werden soll. Je kleiner die Auflösung, desto schneller wird der Barcode generiert und desto weniger Daten werden erzeugt.
Eine Verdoppelung der Auflösung bedeutet eine Vervierfachung der Datenmenge bei gleich bleibender Barcodegröße oder eine Verkleinerung der Barcodegröße um das Vierfache bei gleich bleibendem **X_DIM**. Da die Module eines Datamatrix Barcodes quadratisch sind, hat die Auflösung keinen entscheidenden Einfluss auf seine Qualität. Höhere Auflösungen sollten nur dann gewählt werden, wenn eine gewünschte Barcode Größe mit der Standardauflösung nicht erreicht werden kann.

Falls **RES** erhöht wird, ohne dass **X_DIM** geändert wird, dann verkleinert sich der Barcode und seine Größe kann genauer mit **X_DIM** gesteuert werden.

Falls eine höhere Auflösung bei gleichbleibender Barcodegröße gewünscht wird, dann muss der Parameter **X_DIM** um den gleichen Faktor wie die Auflösung erhöht werden.

Zulässige Werte: > 0 (Pixel pro Zoll)

Standardwert: **150**



RES = 300 dpi, X_DIM = n



RES = 600 dpi, X_DIM = n

Beide Barcodes verschlüsseln den gleichen Text "Suchy MIPS"

X_DIM Horizontale Größe des kleinsten Moduls. Je größer **X_DIM** desto größer der Barcode. Die Einheit ist **1/RES** Zoll. Je höher die Auflösung desto kleiner ist die **X_DIM** Einheit und desto feiner kann die Barcodegröße bestimmt werden.

Zulässige Werte: 1 bis 100.
Standardwert: **4**



X_DIM = 10



X_DIM = 20

Beide Barcodes verschlüsseln den gleichen Text "Suchy MIPS"

Eine detaillierte Beschreibung des Parameters X_DIM finden Sie im Kapitel "Definition der Barcode Eigenschaften"

Y_DIM Sollte nur in Ausnahmefällen definiert werden.

Zulässige Werte: 1 bis 100
Standardwert: **X_DIM**

XPOS **XPOS** gilt nur für SAPscript. Der Parameter bestimmt die horizontale Position der Barcode Grafik relativ zum linken Fensterrand des Fensters, in dem der Barcode eingefügt wird. Der Wert darf nicht negativ sein.

Zulässige Werte: 0 bis Seitenbreite
Standardwert: 0

YPOS **YPOS** gilt nur für SAPscript. Der Parameter bestimmt die vertikale Position der Barcode Grafik relativ zur Zeile, in der der Barcode eingefügt wird. Der Wert darf nicht negativ sein.

Zulässige Werte: 0 bis Seitelänge
Standardwert: 0

Autoheight Art der Platzreservierung für die Barcodegrafik in SAPscript Formularen. Wenn Autoheight auf **'Y'** gesetzt wird, dann wird der ganze Raum links und rechts daneben für die Grafik beansprucht und es können keine Texte daneben platziert werden. Wird der Wert dagegen auf **'N'** gesetzt, dann beansprucht die Grafik keinen Platz.

Zulässige Werte: 'Y' 'N'
Standardwert: 'N'

6.4 Datamatrix Druck mit SAPscript

6.4.1 Barcode Definition im SAPscript Formular

Das folgende Beispiel zeigt die Implementierung eines Datamatrix Labes mit 5 verschiedenen Variablen in einem **SAPscript** Formular.

F..	L	Row Text	R
	+...1....+...2....+...3....+...4....+...5....+...6....+...7..	
/E		ITEM_LINE	
*		<HL>RBarc/Datamatrix	
*			
*		<HL>2D Barcode GS1 Datamatrix ECC 200 generated on a SAP System	
*		<HN>A solution from Suchy MIPS www.suchymips.de	
*			
*			
/*		***** RBarc/Datamatrix *****	
/*		***** definition of encoding data and data types *****	
/:		DEFINE &DMVAR1& = '(>'	
/:		DEFINE &DFLAG1& = 'I'	
/:		DEFINE &DMVAR2& = 'RS'	
/:		DEFINE &DFLAG2& = 'F'	
/:		DEFINE &DMVAR3& = '06'	
/:		DEFINE &DFLAG3& = 'I'	
/:		DEFINE &DMVAR4& = 'GS'	
/:		DEFINE &DFLAG4& = 'F'	
/:		DEFINE &DMVAR5& = '2L'	
/:		DEFINE &DFLAG5& = 'I'	
/:			
/*		***** RBarc/Datamatrix *****	
/*		***** definition of optional barcode parameters *****	
/:		DEFINE &RES& = 150	
/:		DEFINE &MSIZE& = 0	
/:		DEFINE &X_DIM& = 4	
/:		DEFINE &XPOS& = '50.00'	
/:		DEFINE &YPOS& = '05.00'	
/*		***** RBarc/Datamatrix *****	
/*		***** performing barcode generation *****	
/:		PERFORM GEN_DATAMATRIX_SS IN PROGRAM Z_SET_DATAMATRIX	
/:		USING &DMVAR1&	
/:		USING &DFLAG1&	
/:		USING &DMVAR2&	
/:		USING &DFLAG2&	
/:		USING &DMVAR3&	
/:		USING &DFLAG3&	
/:		USING &DMVAR4&	
/:		USING &DFLAG4&	
/:		USING &DMVAR5&	
/:		USING &DFLAG5&	
/:		USING &QRVAR5&	
/:		USING &DFLAG5&	
/:		USING &RES&	
/:		USING &MSIZE&	
/:		USING &X_DIM&	
/:		USING &XPOS&	
/:		USING &YPOS&	
/:		CHANGING &DMNAME&	
/:		CHANGING &LINES&	
/:		CHANGING &RESULT&	
/:		ENDPERFORM	
/*			
/*		***** RBarc/Datamatrix *****	
/*		***** including barcode bitmap *****	
/:		BITMAP &DMNAME& OBJECT GRAPHICS ID BMAP TYPE BMON XPOS &XPOS& MM	
/*			
/*		***** RBarc/Datamatrix *****	
/*		***** deleting barcode bitmap *****	
/:		PERFORM DEL_DM_SS IN PROGRAM Z_SET_DATAMATRIX	
/:		USING &DMNAME&	
/:		CHANGING &RESULT&	
/:		ENDPERFORM	
/*		***** END of BARCODE *****	

Erläuterungen:

```
DEFINE &DMVAR1& ='[(>Ä'  
DEFINE &DFLAG1& = 'T'
```

...

Definition der zu kodierenden Variablen. Es gibt keine Einschränkungen in der Anzahl der Variablen. Es können jederzeit weitere Variablen hinzugefügt werden. Die Namen der Variablen sind vorgeben und dürfen nicht geändert werden. Die zu kodierenden Daten müssen in **DMVARx** gespeichert werden, wobei **x** die Nummer der Variable in aufsteigender Reihenfolge, beginnend mit **1** ist.

```
DEFINE &RES& = 150
```

Definition der Grafikauflösung, wie unter 6.3 beschrieben. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden

```
DEFINE &MSIZE& = 0
```

Definition der Matrixgröße, wie unter 6.3 beschrieben. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden

```
DEFINE &X_DIM& = 4
```

Definition der Horizontalen Größe des kleinsten Moduls, wie unter 6.3 beschrieben. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden.

```
DEFINE &XPOS& = '50.00'
```

Definition der horizontalen Position der Barcode Grafik relativ zum linken Rand des Fensters. Die Einheit ist Millimeter. Der Wert muss in Hochkommata angegeben werden und muss größer oder gleich Null sein. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden.

```
DEFINE &YPOS& = '0.00'
```

Definition der vertikalen Position der Barcode Grafik zur aktuellen Zeile, in der der Barcode ausgegeben wird. Die Einheit ist Millimeter. Der Wert muss in Hochkommata angegeben werden und muss größer oder gleich Null sein. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden.

```
DEFINE &AUTOHEIGHT& = 0
```

Definition der Raumbehandlung, den der Barcode einnimmt, wie unter 6.3 beschrieben. Wird der Parameter weggelassen, dann muss auch die entsprechende **USING** Variable aus dem **PERFORM GEN_DATAMATRIX_SS** Befehl entfernt werden.

PERFORM GEN_DATAMATRIX_SS IN PROGRAM Z_SET_DATAMATRIX

```
USING &DMVAR1&  
USING &DFLAG1&  
USING &DMVAR2&  
USING &DFLAG2&  
USING &DMVAR3&  
USING &DFLAG3&  
USING &DMVAR4&  
USING &DFLAG4&  
USING &DMVAR5&  
USING &DFLAG5&  
USING &RES&  
USING &MSIZE&  
USING &X_DIM&  
CHANGING &DMNAME&  
CHANGING &DMWIDTH&  
CHANGING &RESULT&
```

ENDPERFORM

Übergabe der zu kodierenden Variablen "**DMVAR...**", der dazugehörigen Variablentypen "**DMFLAG...**" und der optionalen Barcodeigenschaften "**RES**" (Resolution) "**MSIZE**" (Matrixgröße) und "**X_DIM**" (Breite des kleinsten Moduls) an die **Formroutine GEN_DATAMATRIX_SS** im Programm **Z_SET_DATAMATRIX**. Die Anzahl der Variablen ist nicht limitiert und kann vom Formularersteller beliebig erweitert werden - unter Berücksichtigung der oben beschriebenen Nummerierungsregel. Als Rückgabeparameter wird der Name des Barcodes geliefert (Parameter "**DMNAME**"), die Barcodebreite in Pixeln der aktuellen Auflösung ("**LINES**") und das Resultat ("**RESULT**"). Wenn keine Fehler auftauchen, dann gilt **RESULT = 'No errors'**.

IBITMAP &DMNAME& OBJECT GRAPHICS ID BMAP TYPE BMON XPOS &XPOS& MM

Der Barcode wird dynamisch in das Formular eingebunden. Mit dem Parameter "**XPOS**" kann der Barcode innerhalb des Fensters, in dem er eingefügt wird, horizontal positioniert werden. Gültige Einheiten sind **CM** und **MM**.

PERFORM DEL_DM_SS IN PROGRAM Z_SET_DATAMATRIX

```
USING &DMNAME&  
CHANGING &RESULT&
```

ENDPERFORM

Nach dem Einbinden in das Formular wird der Barcode aus dem System gelöscht. Der Parameter **&RESULT&** übergibt das Ergebnis. Im positiven Fall enthält die Variable **RESULT** den Wert "**No errors**". Es ist sehr wichtig den Barcode zu löschen. Im anderen Fall bleibt er im System erhalten und muss nachträglich gelöscht werden. Prüfen Sie nach erfolgten Tests mit der Transaktion "**SE78**" ob Grafiken, die mit "**ZDM**" beginnen vorhanden sind. Sie können die Grafiken nachträglich mit dem Programm "**ZDELBMP**", das im Verzeichnis "**UTILITY**" zu finden ist, löschen.

Merke:

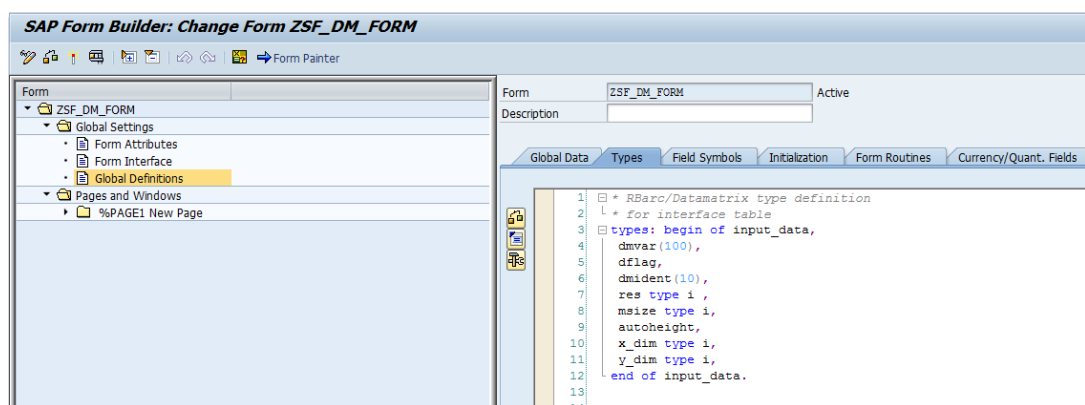
- Die Anzahl der zu kodierenden Variablen (**DMVAR...**) ist nicht limitiert.
- Jeder Variable muss der dazugehörige Typ (**DMFLAG...**) zugeordnet werden.
- Die zu kodierenden Variablen müssen lückenlos, beginnend mit **1**, durchnummeriert werden.
- Der zu Variable **DMVARx** gehörende Typ **DFLAGx** muss die gleiche Nummer tragen.
- Variable **DMVARx** und Ihr Typ **DFLAGx** müssen hintereinander als **USING** Parameter übergeben werden.
- Es dürfen nur tatsächlich vorhanden oder explizit mit **DEFINE** deklarierte Variablen als **USING** Parameter übergeben werden.
- Die Parameter "**RES**", "**MSIZE**", "**XPOS**", "**YPOS**" und "**X_DIM**" sind optional. Werden Sie nicht verwendet, dann kommen Standardwerte automatisch zum Einsatz.

6.5 Datamatrix Druck mit Smart Forms

6.5.1 Barcode Definition im Smart Forms Formular

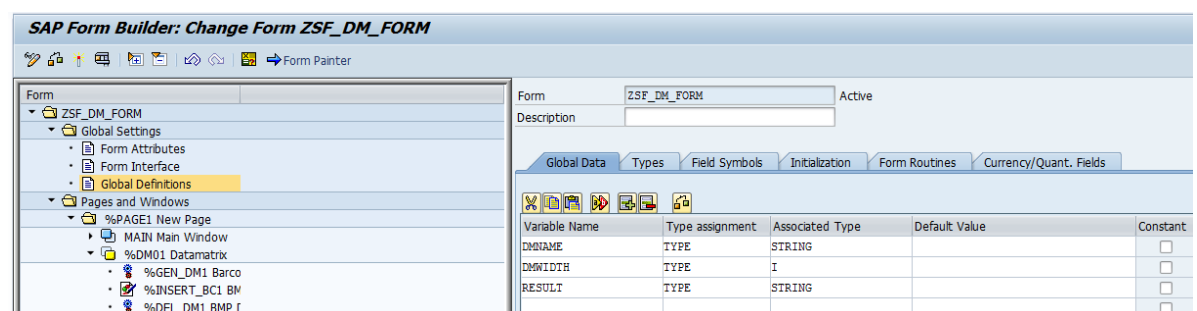
1. Definieren Sie im Reiter **"Typen"** aus dem Baum **"Globale Definitionen"** den type **input_data** wie folgt:

```
types: begin of input_data,  
  dmvar(100),  
  dflag,  
  dmident(10),  
  res type i ,  
  msize type i,  
  autoheight,  
  x_dim type i,  
  y_dim type i,  
end of input_data.
```

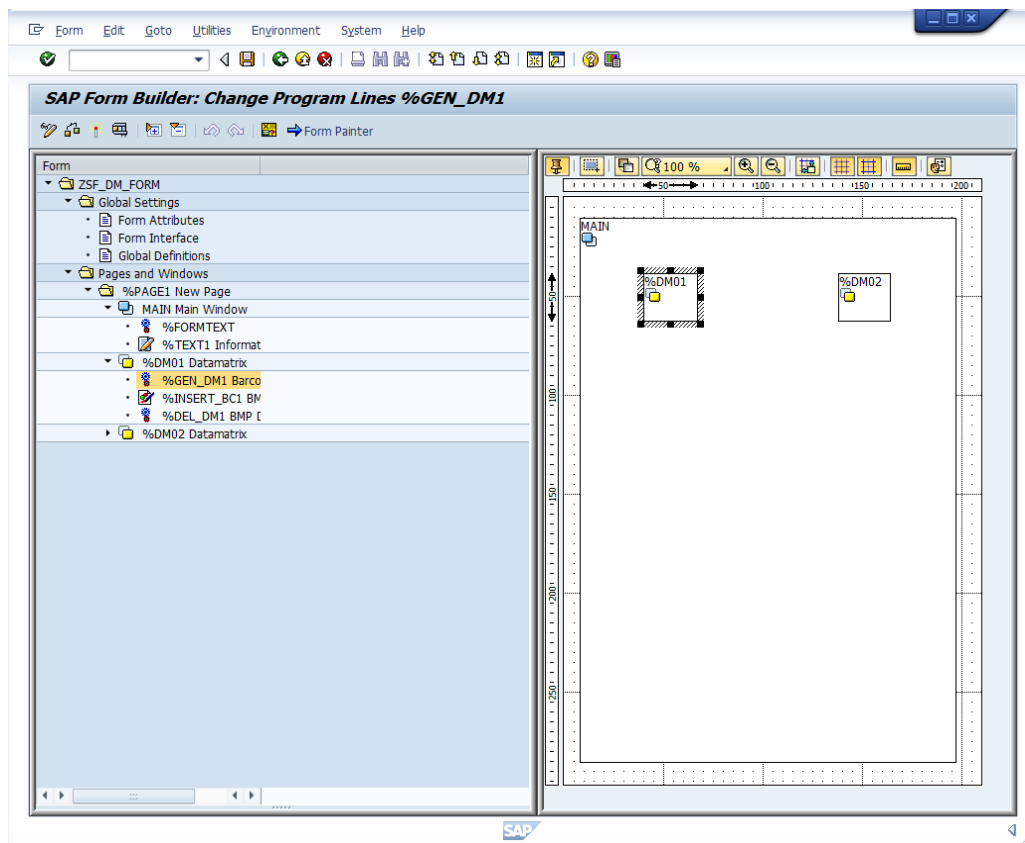


2. Definieren Sie im Reiter **"Globale Daten"** aus dem Baum **"Globale Definitionen"** die folgenden Variablen:

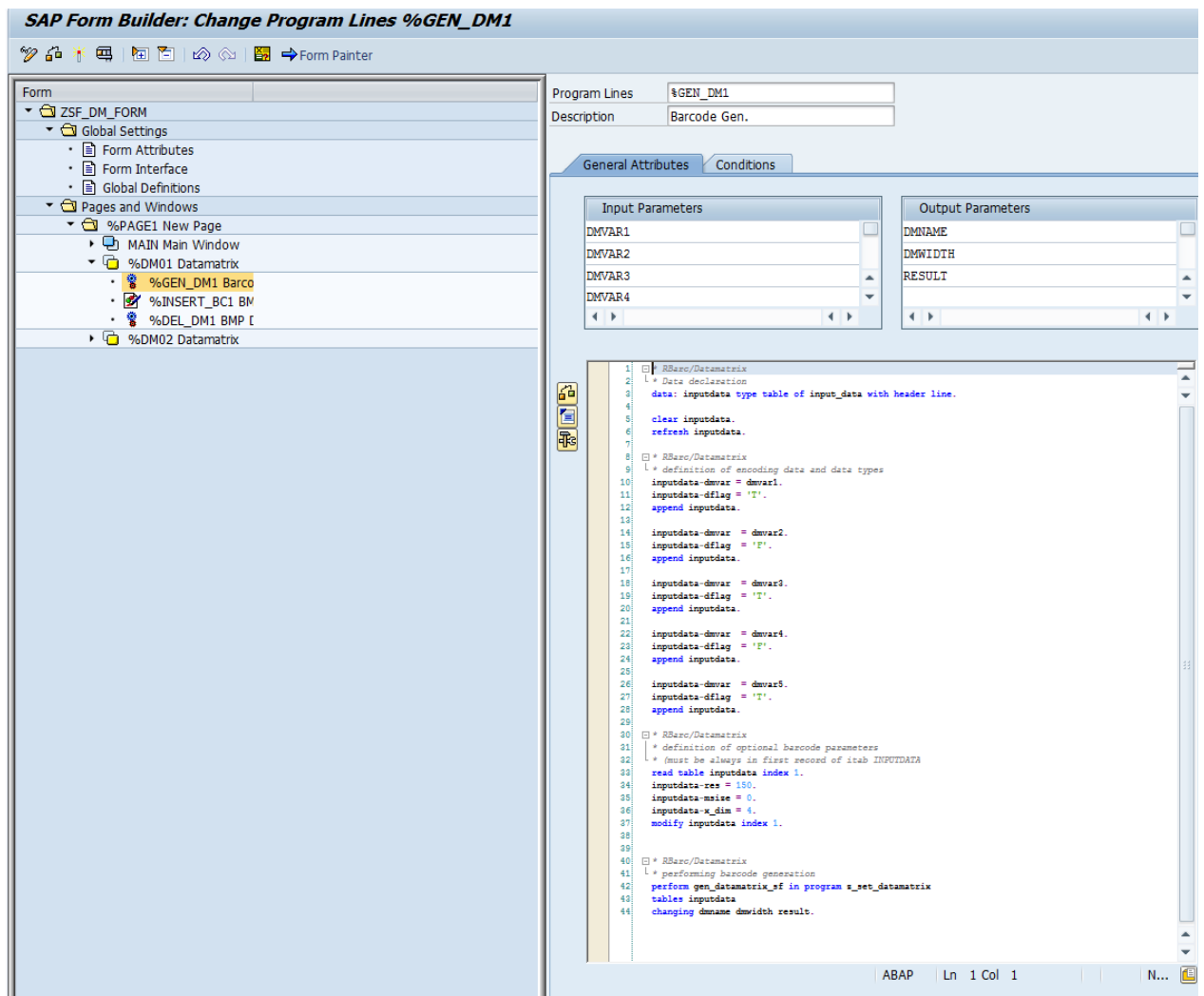
DMNAME	Type String
DMWIDTH	Type I
RESULT	Type String



3. Erstellen Sie ein Fenster, das groß genug ist, um den Barcode aufzunehmen.
Im folgenden Beispiel ist das das Fenster "DM01 Datamatrix".



- Erstellen Sie in dem gerade erzeugten Fenster einen Knoten vom Type "**Programmzeilen**".



- Geben Sie als **Input Parameter** alle zu kodierenden Variablen ein.
- Geben Sie als **Output Parameter** die Variablen **DMNAME**, **DMWIDTH** und **RESULT** an (**DMWIDTH** und **RESULT** dienen der eigenen Verwendung).
- Deklariieren Sie die **interne Tabelle INPUTDATA**
`data: inputdata type table of input_data with header line.`
- Löschen Sie den Inhalt der Tabelle **INPUTDATA** (wichtig, wenn z.B. das Formular mehrere Seiten hat auf denen die gleichen Programmknöten verwendet wird).
`clear inputdata.`
`refresh inputdata.`

- Schreiben Sie die jeweilige Variable in das Tabellenfeld **DMVAR** und den dazugehörigen Datentypen in das Tabellenfeld **DFLAG** und hängen den jeweiligen Datensatz an die Tabelle **INPUTDATA** an

```
inputdata-dmvar = dmvar1.  
inputdata-dflag = 'T'.  
append inputdata.
```

... usw. für jede zu kodierende Variable.

Die Anzahl der kodierbaren Variablen ist nicht limitiert. Die Variablen mit ihrem dazugehörigen Typen müssen lückenlos hintereinander in die Tabelle geschrieben werden. Jede Variable bildet also einen Datensatz (=Record) in der Tabelle.

- Nachdem alle zu kodierenden Variablen in die Tabelle eingetragen wurden, lesen Sie den ersten Datensatz dieser Tabelle ein und tragen dort die gewünschten Werte für **RES** (Resolution), **MSIZE** (Matrixgröße) und **X_DIM** (Größe des kleinsten Moduls) ein

```
read table inputdata index 1.  
inputdata-res = 150.  
inputdata-msize = 0.  
inputdata-x_dim = 4.  
modify inputdata index 1.
```

Dieser Teil ist optional. Falls er nicht verwendet wird, dann werden für die Parameter **RES**, **MSIZE** und **X_DIM** Standardwerte verwendet.

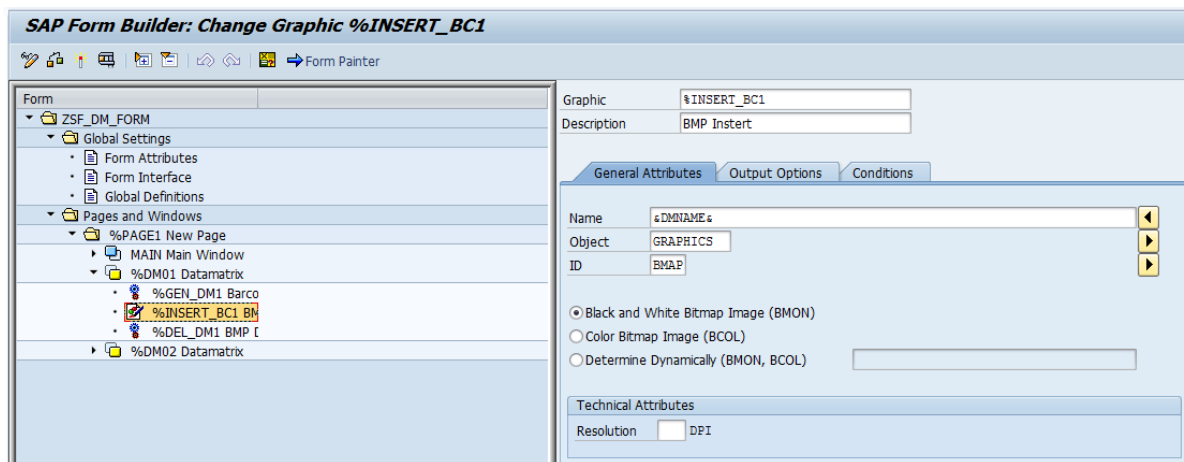
- Übergeben Sie die Tabelle **INPUTDATA** an die Formroutine **GEN_DATAMATRIX_SF** im Programm **Z_SET_DATAMATRIX**

```
perform gen_datamatrix_sf in program z_set_datamatrix  
tables inputdata  
changing dmname dmwidth result
```

- Als Rückgabeparameter wird der Name des Barcodes geliefert (Parameter "**DMNAME**"), die Barcodebreite in Pixeln der aktuellen Auflösung ("**DMWIDTH**") und das Resultat ("**RESULT**"). Wenn keine Fehler auftreten, dann gilt **RESULT = 'No errors'**.

.

5. Erstellen Sie im Fenster für den Datamatrix Barcode einen zweiten Knoten vom Typ "**Grafik**" und füllen Sie die Felder **Name**, **Objekt** und **ID**, wie im folgenden Bild dargestellt:



Name = &DMNAME&

Objekt = GRAPHICS

ID = BMAP

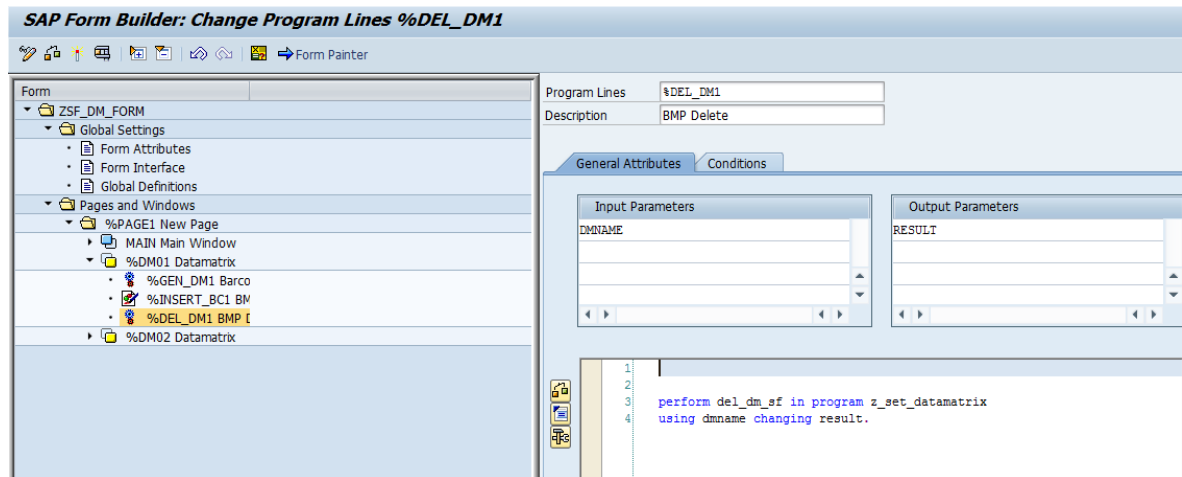
Auflösung = Leer lassen

6. Erstellen Sie im Fenster für den Datamatrix Barcode einen dritten Knoten vom Typ "**Programmzeilen**" und fügen Sie dort folgende Statements ein:

```
perform del_dm_sf in program z_set_datamatrix  
using dmname  
changing result.
```

Geben Sie als **Input Parameter** alle **DMNAME** ein.

Geben Sie als **Output Parameter** **RESULT** ein (optional, falls Sie es verwenden möchten).



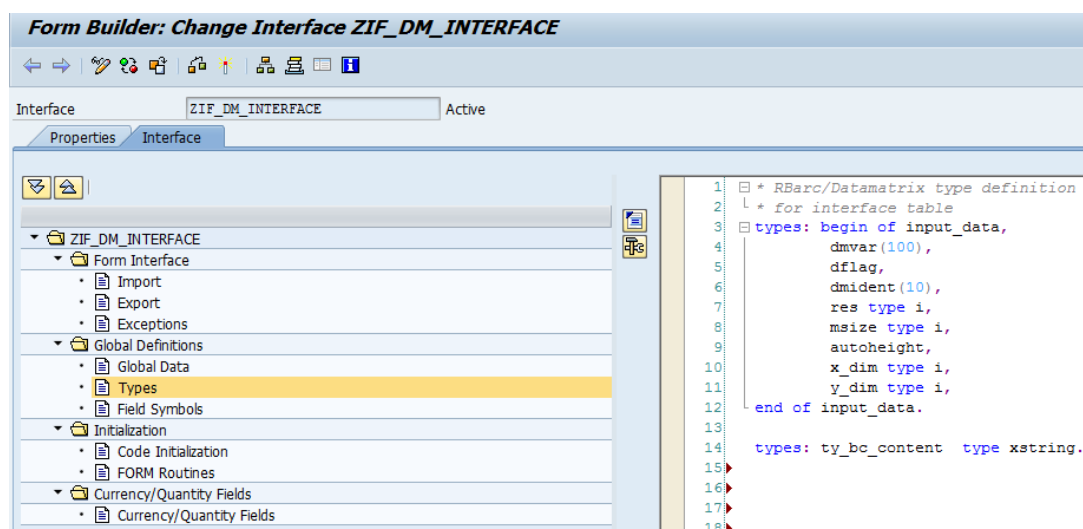
6.6 Datamatrix Druck mit Interactive Forms

6.6.1 Erweiterung der Interactive Forms Schnittstelle

1. Öffnen Sie mit der Transaktion "SFP" die von Ihrem Interactive Forms Formular verwendete Schnittstelle. Legen Sie im Knoten „**Globale Definition→ Typen**“ die Typen **input_data** und **ty_bc_content** wie folgt an:

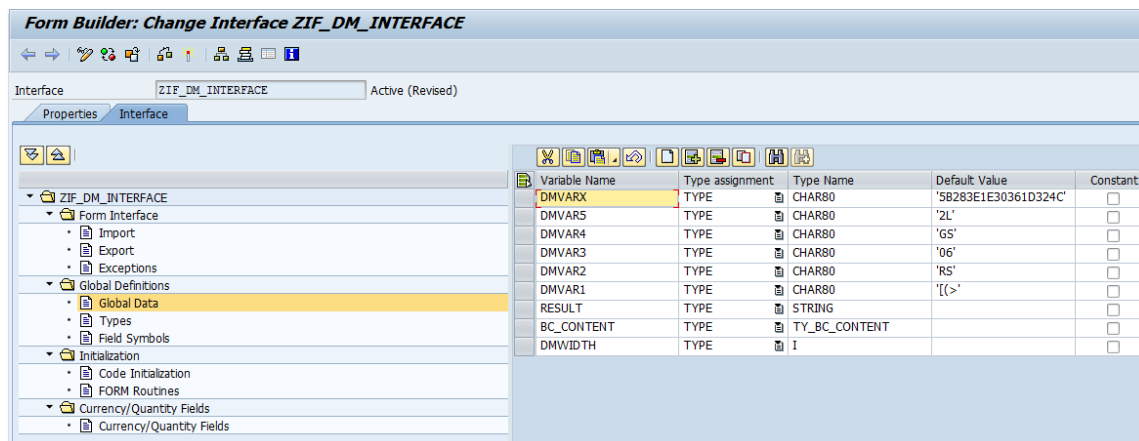
```
types: begin of input_data,  
    dmvar(100),  
    dflag,  
    dmident(10),  
    res type i,  
    msize type i,  
    autoheight,  
    x_dim type i,  
    y_dim type i,  
end of input_data.
```

```
types: ty_bc_content type xstring.
```



2. Definieren Sie im Knoten „**Globale Definitionen→ Globale Daten**“ folgende globale Variablen.

DMWIDTH	TYPE	I
RESULT	TYPE	STRING
BC_CONTENT	TYPE	TY_BC_CONTENT



Bemerkung:

im obigen Beispiel (Screenshot) wurden auch zu kodierende Variablen **DMVAR1** bis **DMVAR5** als globale Variablen in der Schnittstelle angelegt. In der Regel werden das jedoch Variablen aus der Workflow der Datenaufbereitung sein. Sie müssen dann mit geeigneten Mitteln dafür sorgen, dass die benötigten Variablen in der Schnittstelle auch sichtbar sind. In der Regel ist es dann nicht notwendig, solche Variablen als globale Variablen zu definieren, wie das z.B. der Fall bei dem Feld **MATNR** aus der Tabelle **MARD** wäre.

Für mehrere Barcodes in einem Formular können mehrere Globale variablen vom Type **TY_BC_CONTENT** angelegt werden.

3. Fügen Sie folgende Kodierung im Knoten „Initialisierung→Code Initialisierung“ ein (hier ein Beispiel mit 5 verschiedenen Variablen):

data: inputdata type table of input_data with header line.

clear inputdata.
refresh inputdata.

inputdata-dmvar = dmvar1.
inputdata-dflag = 'T'.
append inputdata.

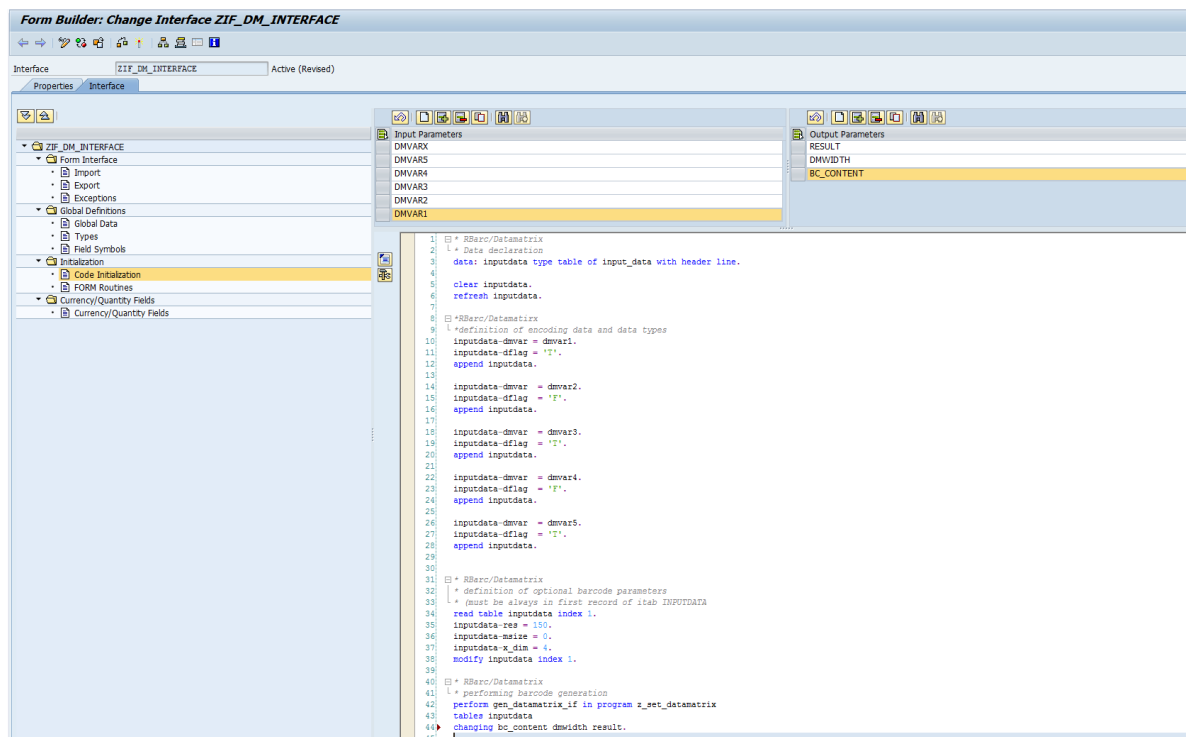
inputdata-dmvar = dmvar2.
inputdata-dflag = 'F'.
append inputdata.

inputdata-dmvar = dmvar3.
inputdata-dflag = 'T'.
append inputdata.

inputdata-dmvar = dmvar4.
inputdata-dflag = 'F'.
append inputdata.

inputdata-dmvar = dmvar5.
inputdata-dflag = 'T'.
append inputdata.

read table inputdata index 1.
inputdata-res = 150.
inputdata-msize = 0.
inputdata-x_dim = 4.
modify inputdata index 1.



Erläuterungen:

- Geben Sie als **Input Parameter** alle zu kodierenden Variablen ein.
- Geben Sie als **Output Parameter** die Variable **BC_CONTENT** und **RESULT** an
- Deklarieren Sie **die interne Tabelle INPUTDATA**
`data: inputdata type table of input_data with header line.`
- Löschen Sie den Inhalt der Tabelle **INPUTDATA** (wichtig, wenn z.B. das Formular mehrere Seiten hat auf denen die gleichen Programmknotten verwendet wird).
`clear inputdata.`
`refresh inputdata.`
- Schreiben Sie die jeweilige Variable in das Tabellenfeld **DMVAR** und den dazugehörigen Datentypen in das Tabellenfeld **DFLAG** und hängen den jeweiligen Datensatz an die Tabelle **INPUTDATA** an
`inputdata-dmvar = dmvar1.`
`inputdata-dflag = 'T'.`
`append inputdata.`

... usw. für jede zu kodierende Variable.

Die Anzahl der kodierbaren Variablen ist nicht limitiert. Die Variablen mit ihrem dazugehörigen Typen müssen lückenlos hintereinander in die Tabelle geschrieben werden. Jede Variable bildet also einen Datensatz (=Record) in der Tabelle.
- Nachdem alle zu kodierenden Variablen in die Tabelle eingetragen wurden, lesen Sie den ersten Datensatz der Tabelle und tragen dort die gewünschten Werte für **RES** (Resolution), **MSIZE** und **X_DIM** (Größe des kleinsten Moduls) ein.
`read table inputdata index 1.`
`inputdata-res = 150.`
`inputdata-msize = 0.`
`inputdata-x_dim = 4.`
`modify inputdata index 1.`

Dieser Teil ist optional. Falls er nicht verwendet wird, dann werden für die Parameter **RES**, **MSIZE**, und **X_DIM** Standardwerte verwendet.
- Übergeben Sie die Tabelle **INPUTDATA** an die Formroutine **GEN_DATAMATRIX_IF** im Programm **Z_SET_DATAMATRIX**
`perform gen_datamatrix_if in program z_set_datamatrix`
`tables inputdata`
`changing bc_content dmwidth result.`

Als Rückgabe Parameter werden folgende Variablen übergeben: **bc_content** (das ist die Barcode Grafik), **dmwidth** (die Barcodebreite in Pixel der aktuellen Auflösung) und **result** (das Ergebnis). Ist die Barcodegenerierung fehlerfrei verlaufen, dann ist **result = 'No errors'**.

Wichtig:

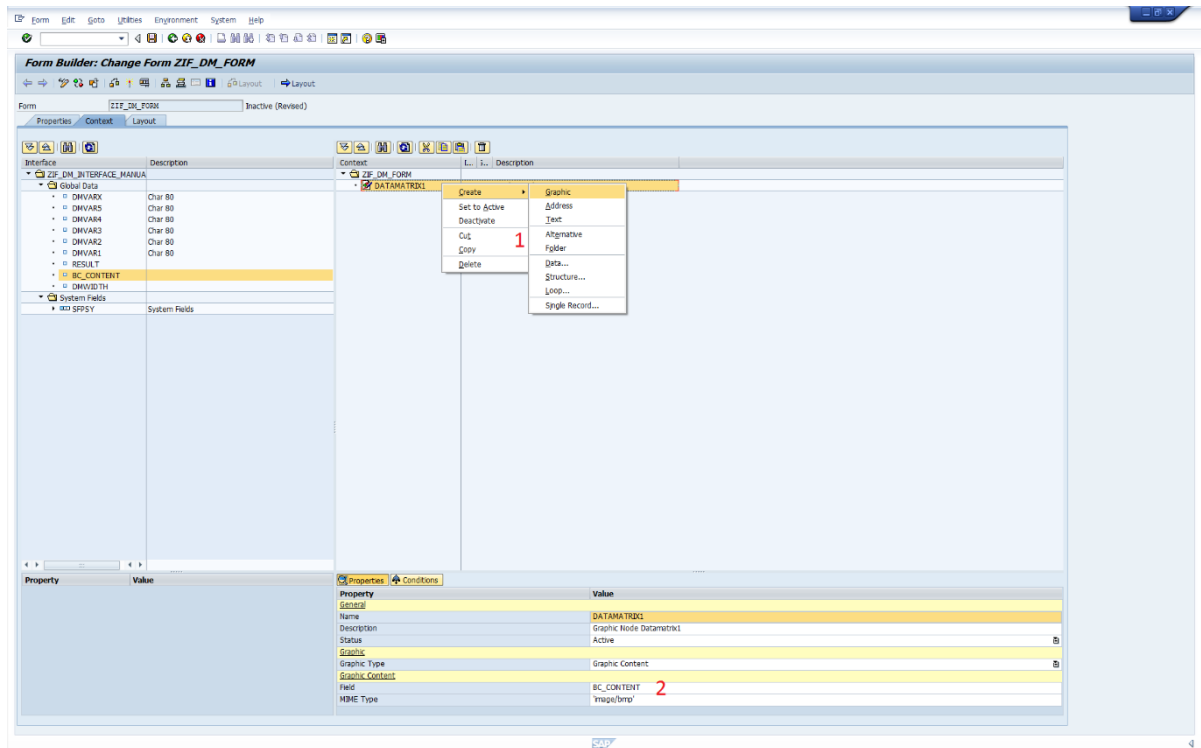
Bitte beachten Sie, dass die verwendeten Variablen im Knoten „Initialisierung“ bekannt gemacht werden müssen, d.h. Sie müssen immer als Eingabeparameter neben den anderen Parametern eingetragen werden.

6.6.2 Erweiterung des Interactive Forms Formulars

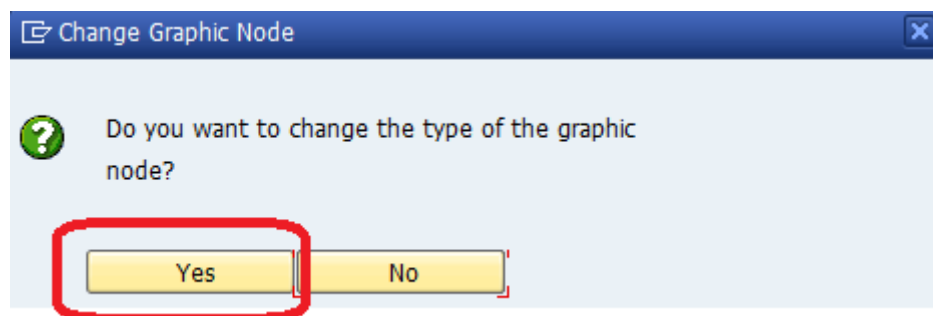
Im **Interactive Forms** Formular wird der mit **RBarc/Datamatrix** erzeugte Barcode als Bildobjekt dargestellt. Zunächst muss daher der Kontext um die notwendigen Elemente für einen Barcode erweitert werden.

1. Markieren Sie im Formular (Transaktion **SFP**) im Kontextfenster (rechts) den Formularnamen, klicken auf die rechte Maustaste und wählen aus dem Kontextmenü **Anlegen→Grafik** um eine Grafik anzulegen.

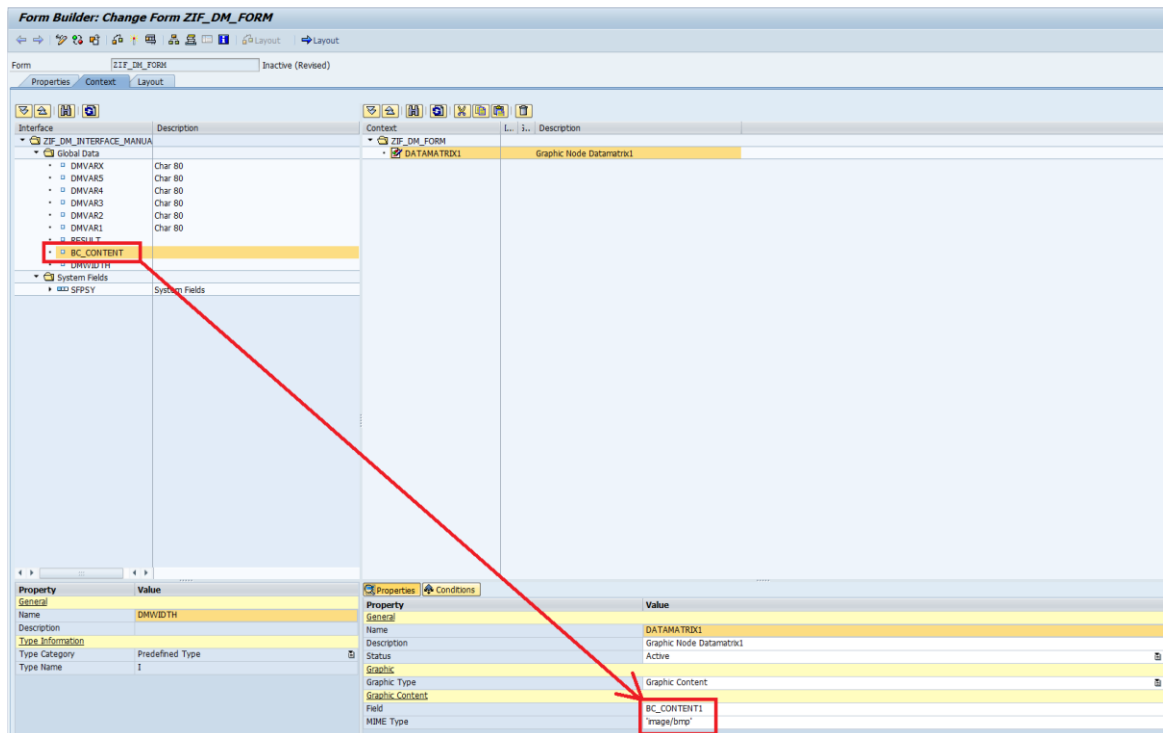
Geben Sie der Grafik einen Namen und eine Beschreibung. Im Beispiel lautet der Name der Grafik **DATAMATRIX1**.



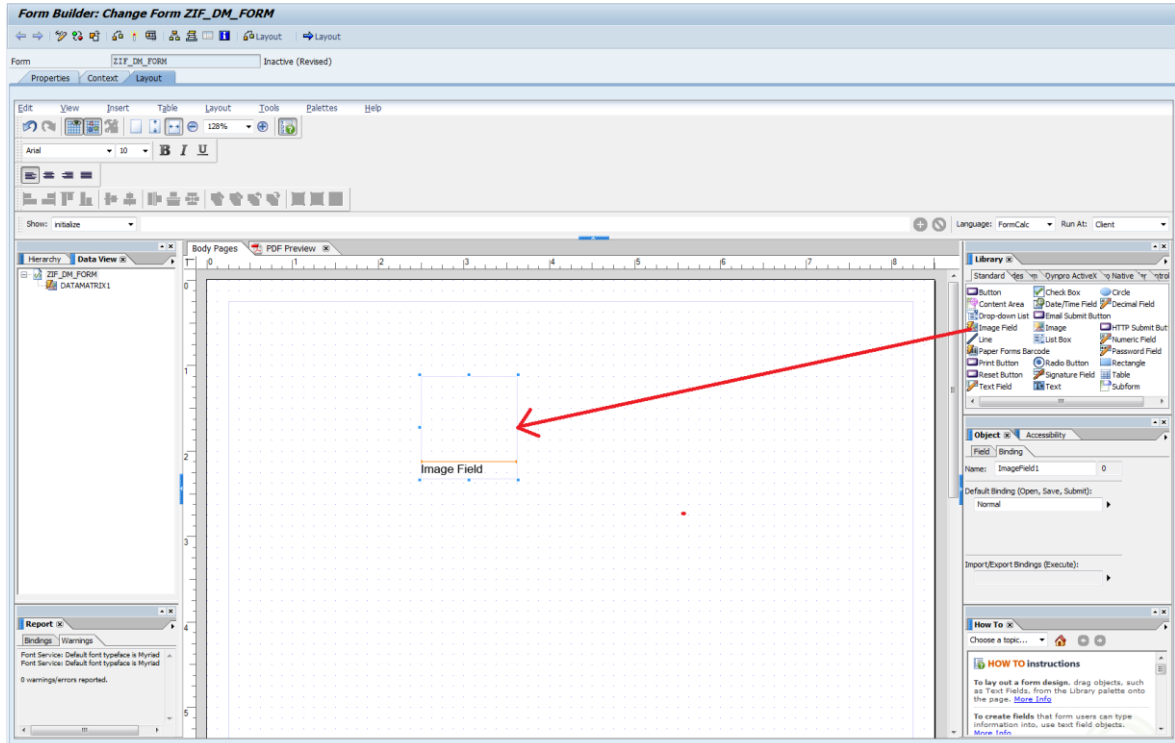
2. Wechseln Sie jetzt den Grafiktyp auf „**Grafikinhalt**“ (siehe Pt. 2 in der Grafik) und bestätigen Sie das Popup mit „**Ja**“.



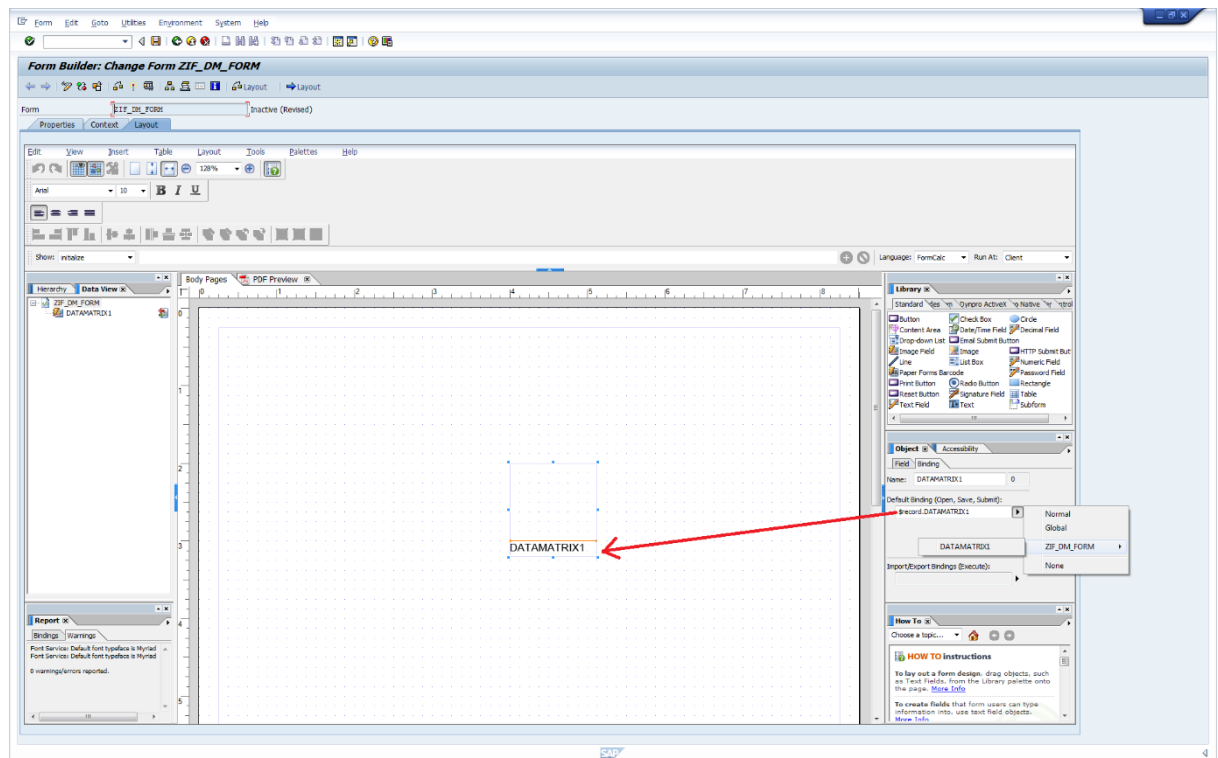
- Verknüpfen Sie jetzt die Grafik **DATAMATRIX1** mit dem Datenfeld, der den Inhalt des Barcodes enthält, indem Sie mit gedrückter linken Maustaste aus der Schnittstelle das Feld **BC_CONTENT** in das passende Feld auf der rechten Seite ziehen. Geben Sie als **MIME-Typ** 'image/bmp' an.



- Wechseln Sie in die Layoutansicht des Form Builders und legen Sie ein Element vom Typ „Bildfeld“ an.



5. Geben Sie dem Bildfeld als Datenbindung den im Kontext definierten Grafikknoten mit (im Beispiel **DATAMATRIX1**).

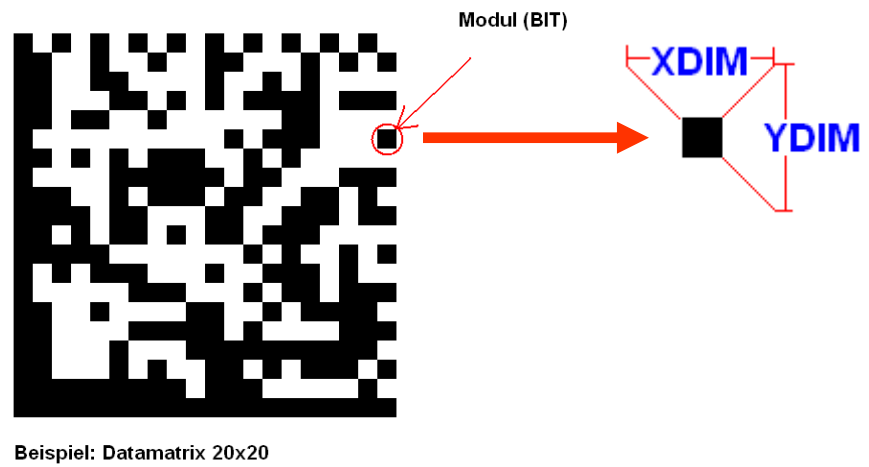


Fertig!

Durch die soeben erzeugte Datenbindung wird bei der Verarbeitung des Interactive Forms an der ausgewählten Stelle der von RBarc/Datamatrix ON-THE-FLY erzeugte Datamatrix Barcode eingefügt.

6.7 Definition der Barcode Eigenschaften.

:



Ein DATAMATRIX Barcode besteht aus schwarzen und weißen Modulen, deren Größe variabel ist.

Folgende Größen für einen Datamatrix Barcode können definiert werden:



Msize	Anzahl Module	Datencodewords
0	auto	Je nach Datenmenge
1	10x10	3
2	12x12	5
3	14x14	8
4	16x16	12
5	18x18	18
6	20x20	22
7	22x22	30
8	24x24	36
9	26x26	44
10	32x32	62
11	36x36	86
12	40x40	114
13	44x44	144
14	48x48	174
15	52x52	204
16	64x64	280
17	72x72	368
18	80x80	456
19	88x88	576
20	96x96	696
21	104x104	816
22	120x120	1050
23	132x132	1304
24	144x144	1558
25	8x18	5
26	8x32	10
27	12x26	16
28	12x36	22
29	16x36	32
30	16x48	49

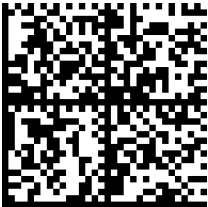
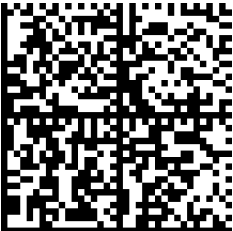
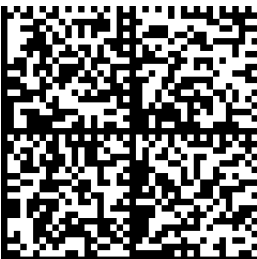
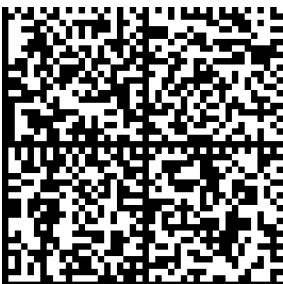

In der obigen Tabelle wurde die Anzahl der Datencodewords angegeben, die je nach gewählter Größe (Parameter **msize**) kodiert werden kann. Bitte bedenken Sie, dass dieser Wert nur ein Anhaltspunkt für die Menge der Daten ist, die wirklich kodiert werden können. In manchen Fällen werden die zu kodierenden Daten so komprimiert, dass mehr Daten kodiert werden können, als die jeweils angegebene Anzahl von Codewords. In manchen, ungünstigen Fällen kann es jedoch passieren, dass die Anzahl der Daten, die kodiert werden können, geringer ist, als die angegebene Anzahl von Datencodewords für eine bestimmte Barcode Größe.



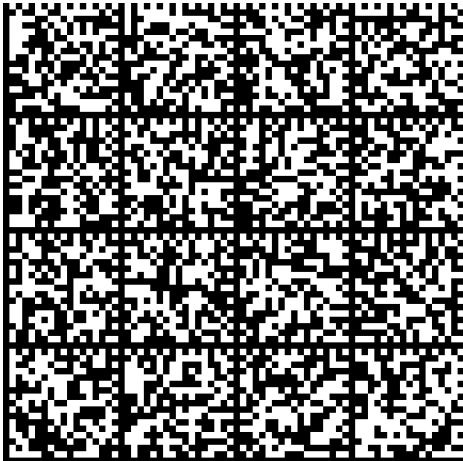
Bemerkung:

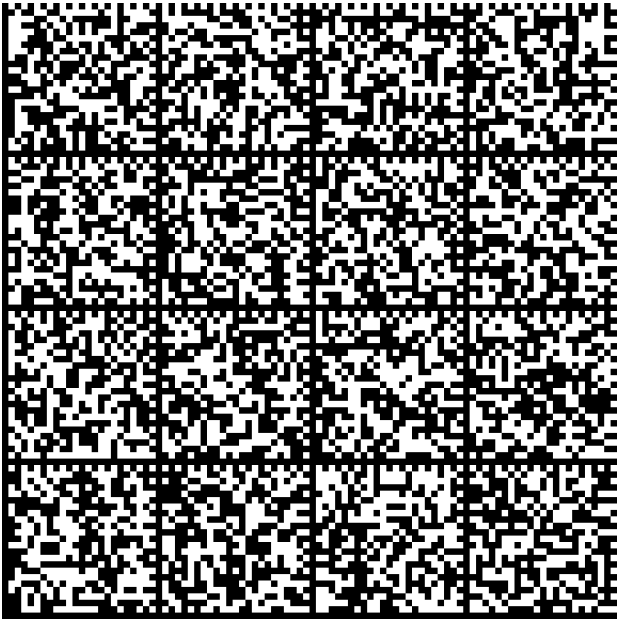
Bei Angabe des Wertes **0** für **msize**, wird die Größe automatisch berechnet und zwar so, dass die Daten in einem kleinstmöglichen Raster kodiert werden. Ein wichtiger Aspekt dieses Verfahrens ist, dass bei Seriendruckern die Barcodes verschiedene Größen haben können, je nach Anzahl der Daten, die zum Kodieren anfallen.

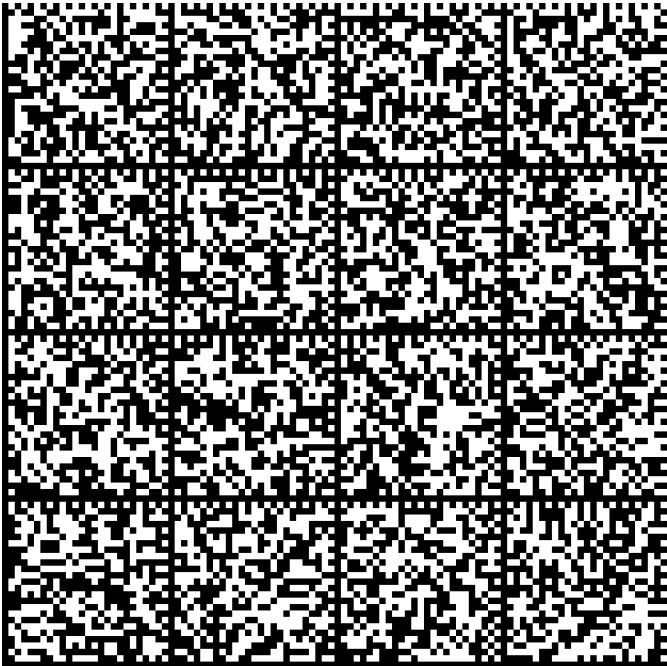
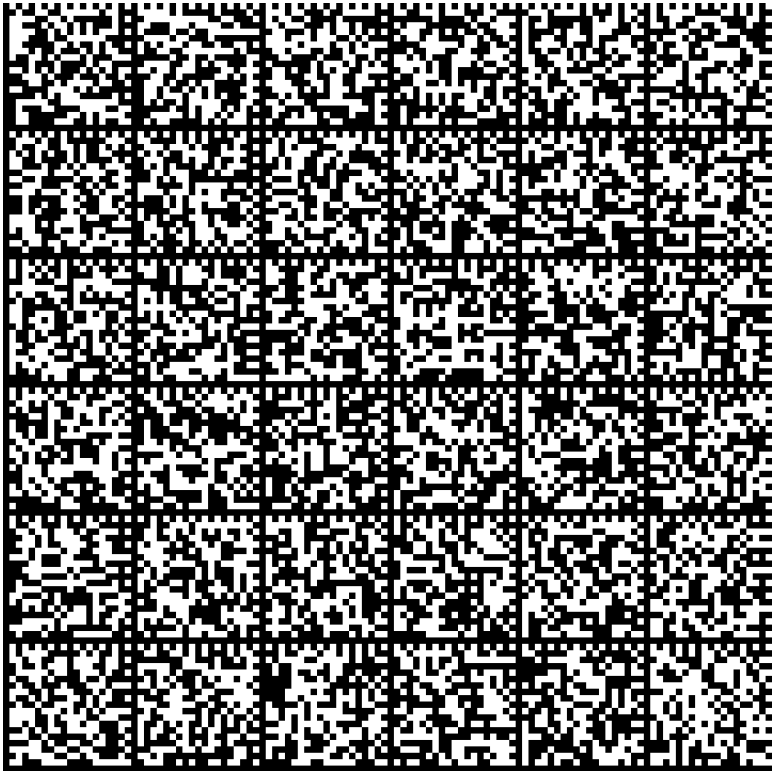
Demgegenüber kann die Größe mit dem Parameter **msize** fest definiert werden, sodass alle Barcodes in gleicher Größe erscheinen. In diesem Fall droht jedoch die Gefahr, dass bei zu großer Datenmenge der Barcode mit vorgegebener Größe nicht erzeugt werden kann und eine Fehlermeldung erscheint. Sie sollten also in einem solchen Fall den Barcode entsprechend überdimensionieren und in Ihren Tests nicht nur Ziffern (beste Datenkompression), sondern wechselnd Klein-, Großbuchstaben, Sonderzeichen und eine kleine Anzahl von Ziffern verwenden.

Matrix Größe	Barcode Beispiel
10x10	
12x12	
14x14	
16x16	
18x18	
20x20	
22x22	
24x24	
26x26	

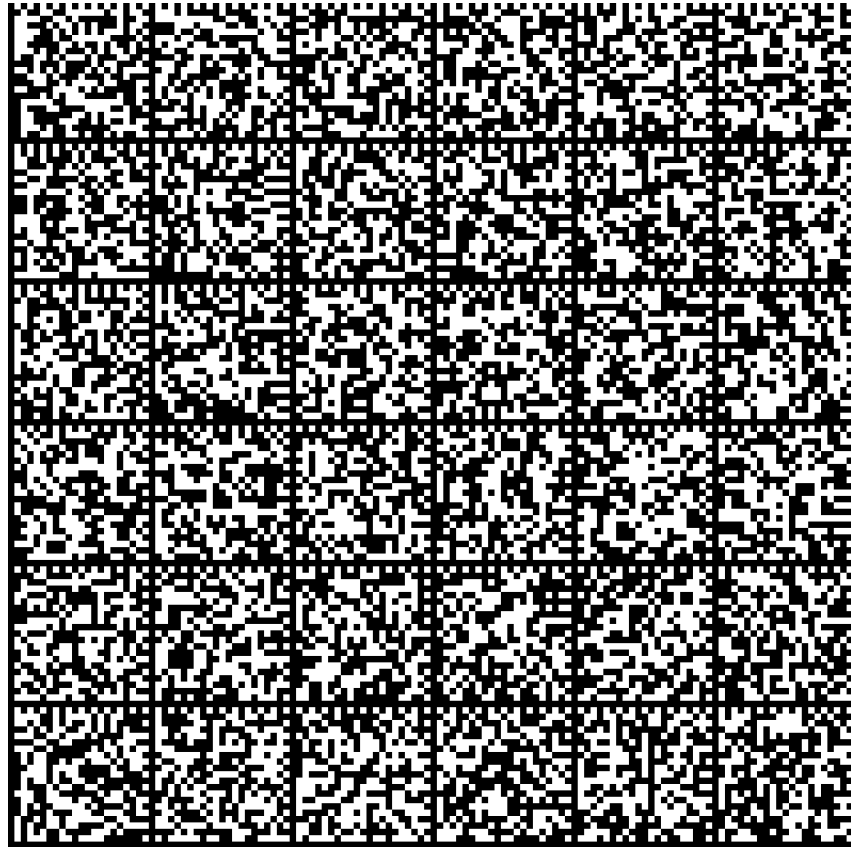
32x32	
36x36	
40x40	
44x44	
48x48	

52x52	
64x64	
72x72	

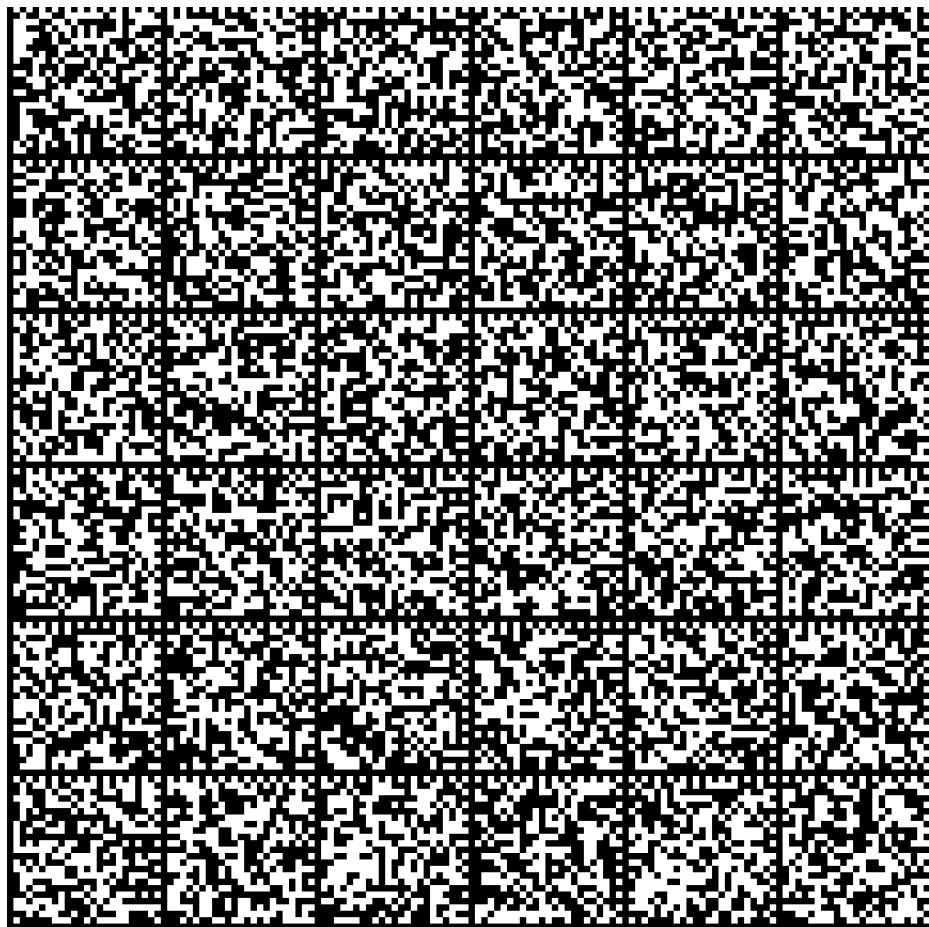
80x80	
88x88	
96x96	

104x104	
120x120	

132x132



144x144



8x18	
8x32	
12x26	
12x36	
16x36	
16x48	

7 Beispiel für eine Aufgabenstellung in einem SAPscript Formular.

7.1 Aufgabe

Es sollen folgende Felder kodiert werden:

VBELN aus der Tabelle **LIKP**,
VBELN aus der Tabelle **VBRK** und
ADDRNUMBER aus der Tabelle **ADRT**.

Die Variablen sollen voneinander mit einem **GS** (Group Separator) als Feldseparator getrennt werden. Der Barcode soll **50 Mm** vom linken Rand des Fensters gedruckt werden. Das Matrixmodul soll **1 Mm** breit und **1 Mm** hoch sein. Die Matrixgröße soll automatisch berechnet werden

Lösung:

Definieren Sie im **SAPscript** Formular die 3 Variablen, deren Typen (dflag) und die Variablen mit den Barcode Eigenschaften (**RES**, **MSIZE** und **X_DIM**). Übergeben Sie alle definierten Variablen an die Formroutine **GEN_DATAMATRIX_SS** im Report **Z_SET_DATAMATRIX**. Anschließend lassen Sie die von **RBarc/Datamatrix** erzeugte Grafik dynamisch in das Formular einbinden und vom System löschen.

```
/: DEFINE &DMVAR1& = &LIKP-VBELN&  
/: DEFINE &DFLAG1& = 'T'  
/: DEFINE &DMVAR2& = 'GS'  
/: DEFINE &DFLAG2& = 'F'  
/: DEFINE &DMVAR3& = &VBRK-VBELN&  
/: DEFINE &DFLAG3& = 'T'  
/: DEFINE &DMVAR4& = 'GS'  
/: DEFINE &DFLAG4& = 'F'  
/: DEFINE &DMVAR5& = &ADRT-ADDRNUMBER&  
/: DEFINE &DFLAG5& = 'T'  
/: DEFINE &RES& = 150  
/: DEFINE &MSIZE& = 0  
/: DEFINE &X_DIM& = 6
```

(bitte Seite wechseln)


```

/: PERFORM GEN_DATAMATRIX_SS IN PROGRAM Z_SET_DATAMATRIX

/: USING &DMVAR1&
/: USING &DFLAG1&
/: USING &DMVAR2&
/: USING &DFLAG2&
/: USING &DMVAR3&
/: USING &DFLAG3&
/: USING &DMVAR4&
/: USING &DFLAG4&
/: USING &DMVAR5&
/: USING &DFLAG5&
/: USING &RES&
/: USING &MSIZE&
/: USING &X_DIM&
/: CHANGING &DMNAME&
/: CHANGING &DMWIDTH&
/: CHANGING &RESULT&

/: ENDPERFORM

/: BITMAP &DMNAME& OBJECT GRAPHICS ID BMAP TYPE BMON XPOS &XPOS& MM

/: PERFORM DEL_DM_SS IN PROGRAM Z_SET_DATAMATRIX
/: USING &DMNAME&
/: CHANGING &RESULT&

/: ENDPERFORM

```

In der oben dargestellten Implementierung werden zunächst alle Variablen Typen zugeordnet (z.B. wird **&LIKPVBELN&** als Text definiert). Dabei wird der Separator **"GS"** als Funktionscode mit dem Typenbezeichnung **dflag = 'F'** kodiert.

Im nächsten Schritt werden Barcode Eigenschaften mit **RES**, **MSIZE** und **X_DIM** definiert.

Alle definierten Variablen und Parameter werden in einem Schritt an die Formroutine **GEN_DATAMATRIX_SS** im Programm **Z_SET_DATAMATRIX** übergeben.

Die von **RBarc/Datamatrix** erzeugte Grafik mit dem Namen **&DMNAME&** wird in das Formular mit der Anweisung **BITMAP...** eingebunden.

Zum Schluss wird die Grafik durch den Aufruf der Formroutine **DEL_DM_SS** im Programm **Z_SET_DATAMATRIX** aus dem System gelöscht.

8 Beispiel für Aufgabenstellung in einem Smart Forms Formular.

8.1 Aufgabe.

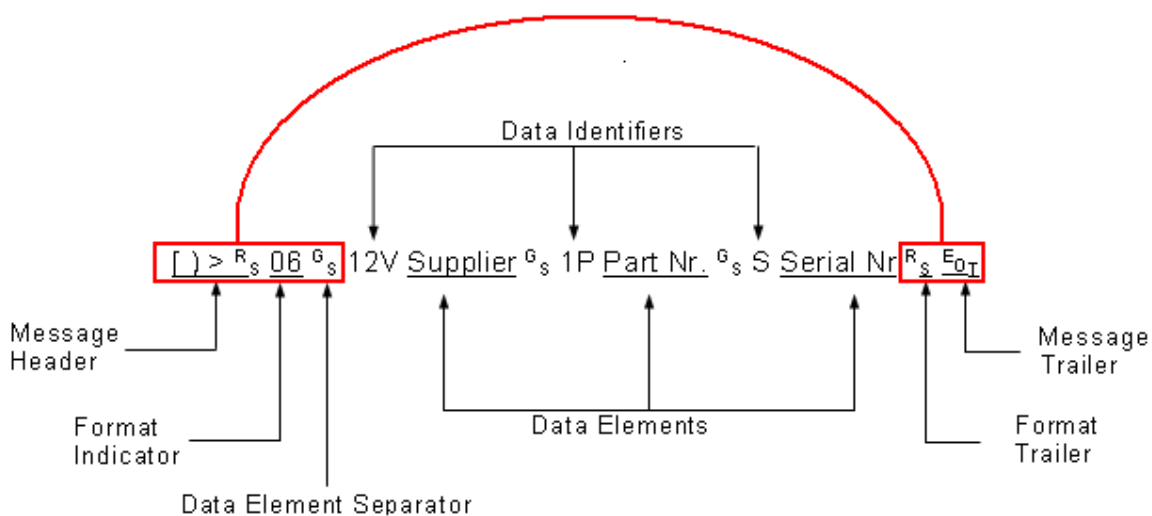
Es sollen 3 Variablen kodiert werden:

die Hersteller-Nummer "**SuplierNo**",

die Artikel-Nummer "**PartNo**"

die Seriennummer eines Artikels "**SerNo**".

Die Variablen sollen in ein sog. **Message Envelop** "verpackt" werden (siehe Abbildung). Das Matrixmodul soll **1 Mm** breit und **1 Mm** hoch sein. Die Matrixgröße soll automatisch berechnet werden.



Der "**Message Envelop**" ist eine normierte Art von Datenübergabe nach ANSI Norm MH10.8.7. Dabei wird vor den Daten ein Message Header – je nach Norm " $[F]>R_s06G_s$ " oder " $[F]>R_s05G_s$ " und nach den Daten ein Trailer R_sE_{OT} kodiert. Vor jede Variable wird ein normierter Identifikator gestellt (z.B. 12V für Supplier) und die Daten werden voneinander durch den Feldseparator G_s getrennt.

Lösung:

1. Als erstes erzeugen Sie das Barcode Fenster mit den 3 dazugehörigen Knoten, wie im Kapitel **Drucken von Barcodes aus Smart Forms** beschrieben. Weitere Definitionen müssen nur im ersten Programmzeilen Knoten durchgeführt werden.
2. Der Inhalt der Knoten 2 und 3 bleibt immer unverändert (wie unter **Drucken von Barcodes aus Smart Forms** beschrieben).

(bitte Seite wechseln)

3. Erstellen Sie im ersten Barcode-Programmknoten folgenden Code:

Datendeklaration

```
data: inputdata type table of input_data with header line.  
clear inputdata.  
refresh inputdata.
```

Kodierung von 'MAC06' als Funktionscode

```
inputdata-dmvar = 'MAC06'.  
inputddata-dflag = 'F'.  
append inputdata.
```

Kodierung von 12V + suplierno als Text

```
concatenate '12V' suplierno into inputdata-dmvar.  
inputdata-dflag = 'T'.  
append inputdata.
```

Kodierung von 'GS' als Funktionscode

```
inputdata-dmvar= 'GS'.  
inputddata-dflag = 'F'.  
append inputdata.
```

Kodierung von 1P + partno als Text

```
concatenate '1P' partno into dmvar.  
inputdata-dflag = 'T'.  
append inputdata.
```

Kodierung von 'GS' als Funktionscode

```
inputdata-dmvar5= 'GS'.  
inputddata-dflag = 'F'.  
append inputdata.
```

Kodierung von S + serialo als Text

```
concatenate 'S' serialno into dmvar.  
inputdata-dflag = 'T'.  
append inputdata.
```

Definition der Barcode Eigenschaften

```
read table inputdata index 1.  
inputdata-res = 150.  
inputdata-dmsize = 0.  
inputdata-x_dim = 4.  
modify inputdata index 1.
```

Übergabe der Schnittstellentabelle an z_set_datamatrix

```
perform gen_datamatrix_sf in program z_set_datamatrix  
tables inputdata  
changing dmname.
```

In der oben dargestellten Implementierung werden zunächst alle Variablen Typen zugeordnet (z.B. wird **12+SUPLIerno** als Text definiert). Dabei wird der Separator "**GS**" als Funktionscode mit dem Typenbezeichnung dflag = '**F**' kodiert. Auch **MAC06** wird als Funktionscode mit der Typenbezeichnung "**F**" übergeben.

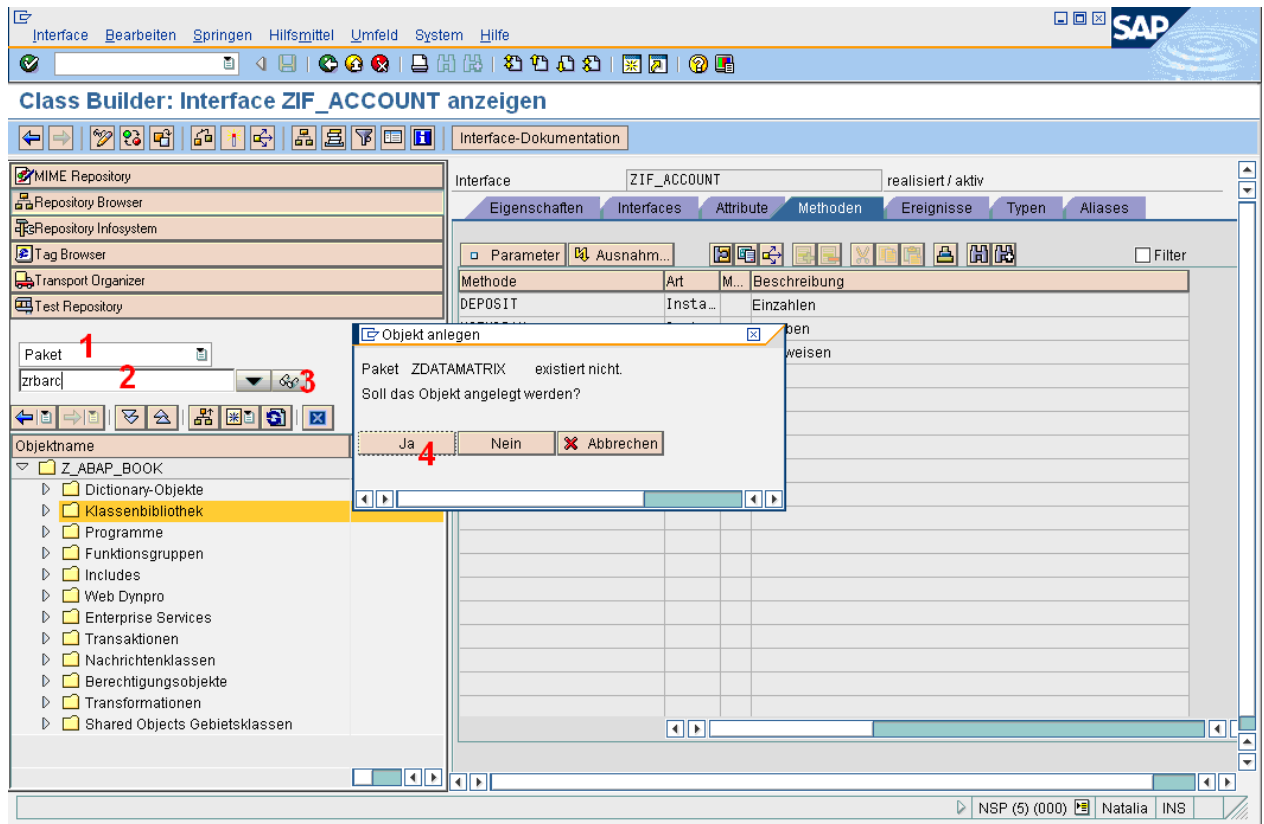
Im nächsten Schritt werden Barcode Eigenschaften mit **RES**, **MSIZE** und **X_DIM** definiert und in den ersten Satz der Schnittstellentabelle **INPUTDATA** geschrieben.

Die Schnittstellentabelle **INPUTDATA** wird an die Formroutine **gen_datamatrix_sf** im Programm **z_set_datamatrix** übergeben.

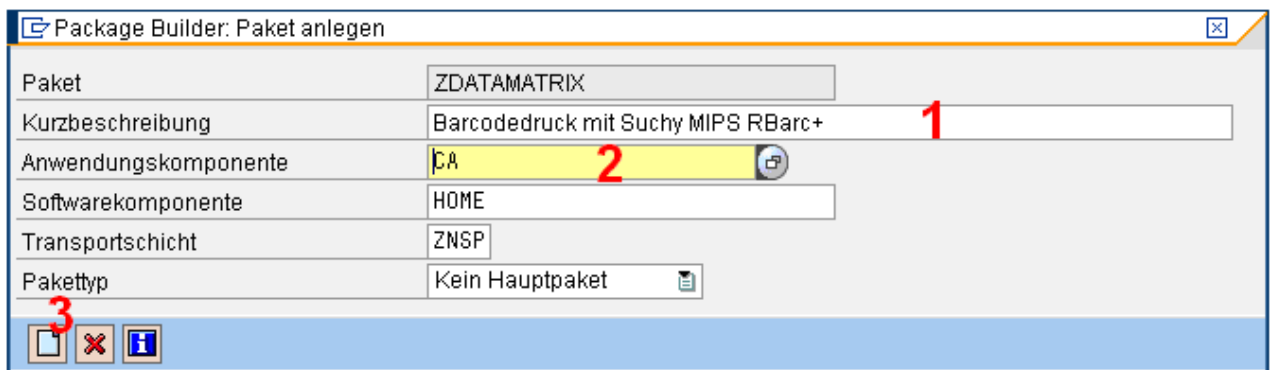
Als Rückgabeparameter wird der Name des Barcodes **DMNAME** geliefert. Der Barcode wird in der weiteren Formularverarbeitung dynamisch eingebunden.

9 Anhang 1: Neues Paket (Entwickungsklasse) erstellen.

1. Starten Sie die Transaktion **SE80 (Objekt Navigator)**.
2. Es erscheint folgender Dialog:



3. Wählen Sie als darzustellendes Objekt „Paket“ (1) (auf älteren Systemen „Entwickungsklasse“).
4. Geben Sie unter dem Objekttypfeld den Namen **ZDATAMATRIX** ein (2).
5. Klicken Sie auf das Brillensymbol für Anzeige (3).
6. Es erscheint die Meldung, dass das Paket nicht vorhanden ist und Sie werden gefragt, ob Sie es anlegen möchten. Klicken Sie auf **Ja** (4).
7. Es erscheint folgender Dialog:



8. Fügen Sie eine Kurzbeschreibung ein (1).
9. Wählen Sie „CA“ als Anwendungskomponente (2).
10. Klicken Sie auf das Bestätigungssymbol (3) um das Paket anzulegen.

11. Es erscheint die Abfrage für transportierbaren Workbench-Auftrag.

Abfrage transportierbarer Workbench-Auftrag

Paket: ZDATAMATRIX

Auftrag: NSPK900046 Workbench-Auftrag

Kurzbeschreibung: barcodes

✓ | | | ✗

Eigene Aufträge

12. Falls Sie für diese Installation noch keinen neuen Auftrag angelegt haben, legen Sie jetzt einen an. Sonst springen Sie weiter zum Pt. 16.
13. Klicken Sie auf das Symbol um einen neuen Transportauftrag zu erstellen.

Auftrag anlegen

Auftrag: Workbench-Auftrag

Kurzbeschreibung: Barcodedruck mit Suchy MIPS RBarc+ 1

Projekt:

Inhaber: BCUSER

Status: Neu

Letzte Änderung: 12.01.2007 12:42:54

Quellmandant: 000

Ziel: CL5

Aufgaben: Mitarbeiter, BCUSER

2 | | | ✗

14. Tragen Sie eine Kurzbeschreibung für den Auftrag ein (1).
15. Klicken Sie auf das Diskettensymbol (2) um den Eintrag zu speichern.
16. Die Auftragsnummer erscheint in dem Feld „Auftrag“

Abfrage transportierbarer Workbench-Auftrag

Paket: ZDATAMATRIX

Auftrag: NSPK900048 Workbench-Auftrag

Kurzbeschreibung: Barcodedruck mit Suchy MIPS RBarc+

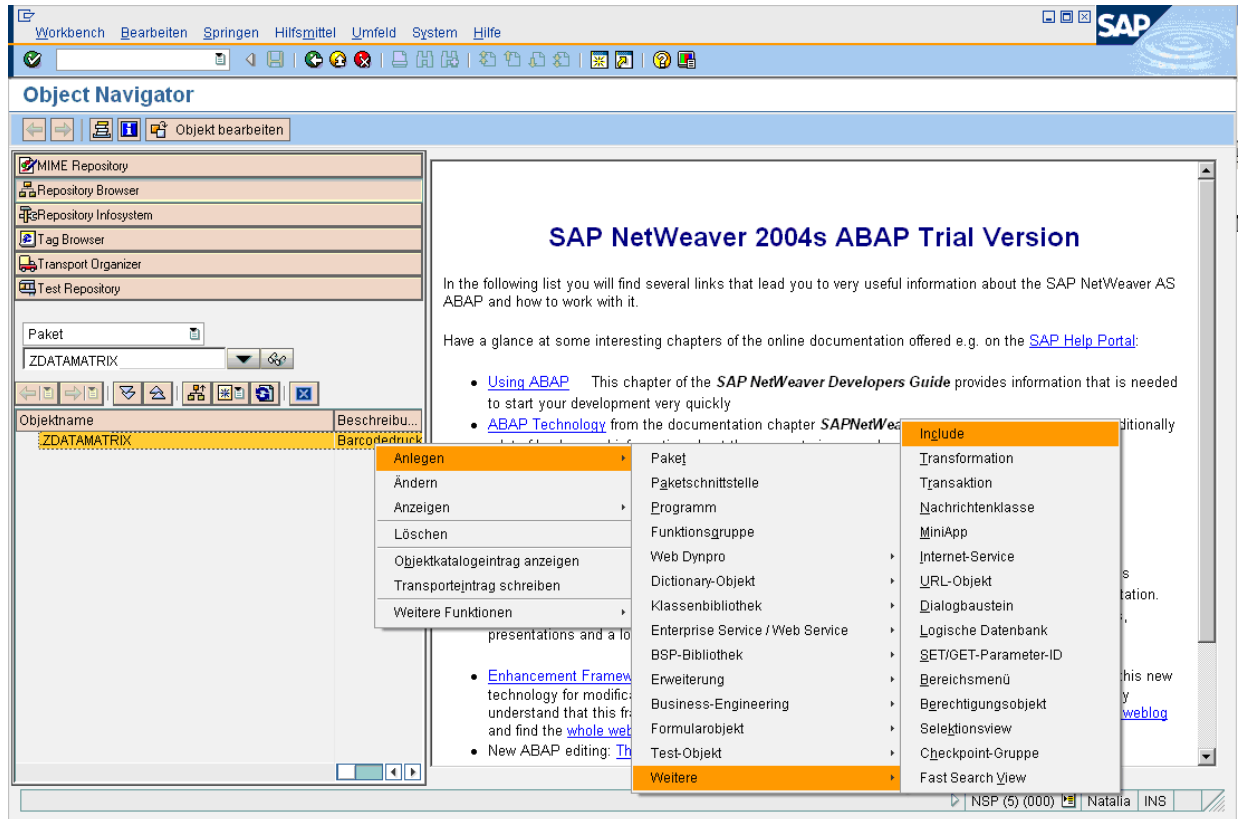
✓ 1 | | | ✗

Eigene Aufträge

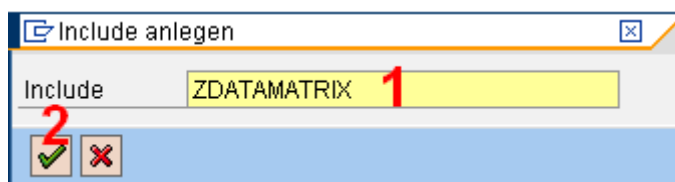
17. Klicken Sie auf das grüne Häkchen (1) um den Auftrag zu speichern.

10 Anhang 2: Ein Include in der ABAP Workbench anlegen.

1. Falls noch nicht geschehen, starten Sie den **Objekt Navigator** (Transaktion **SE80**) und wählen Sie das angelegte Paket (Entwickungsklasse) **ZDATAMATRIX**.



2. Markieren Sie den Objektnamen **ZDATAMATRIX** und klicken Sie auf die rechte Maustaste.
3. Wählen Sie aus dem Kontextmenü **Anlegen/Weitere/Include** (Bei älteren Systemen erscheint **Include** gleich an zweiter Stelle).
4. Geben Sie in dem folgenden Dialog einen Namen für das zu erstellende Include (**1**) und klicken auf das grüne Häkchen (**2**).



5. Klicken Sie in dem folgenden Dialog auf „Sichern“ (1) um das Include zu erstellen.

ABAP: Programmeigenschaften

Titel **ZDATAMATRIX**

Originalsprache DE Deutsch

Erstellt 12.01.2007 BCUSER

Letzte Änderung

Status neu(überarbeitet)

Attribute

Typ Include-Programm

Status

Anwendung

Berechtigungsgruppe

☐ Editorsperre

1 Sichern

6. Achten Sie darauf, dass das Include dem neu angelegten Paket (Entwickungsklasse) **ZDATAMATRIX** hinzugefügt wird (1) und klicken Sie auf das Diskettensymbol (2) um das Include zu sichern.

Objektkatalogeintrag anlegen

Objekt R3TR PROG ZRBARC_2006X

Attribute

Paket **ZDATAMATRIX** 1

Verantwortlicher CUST

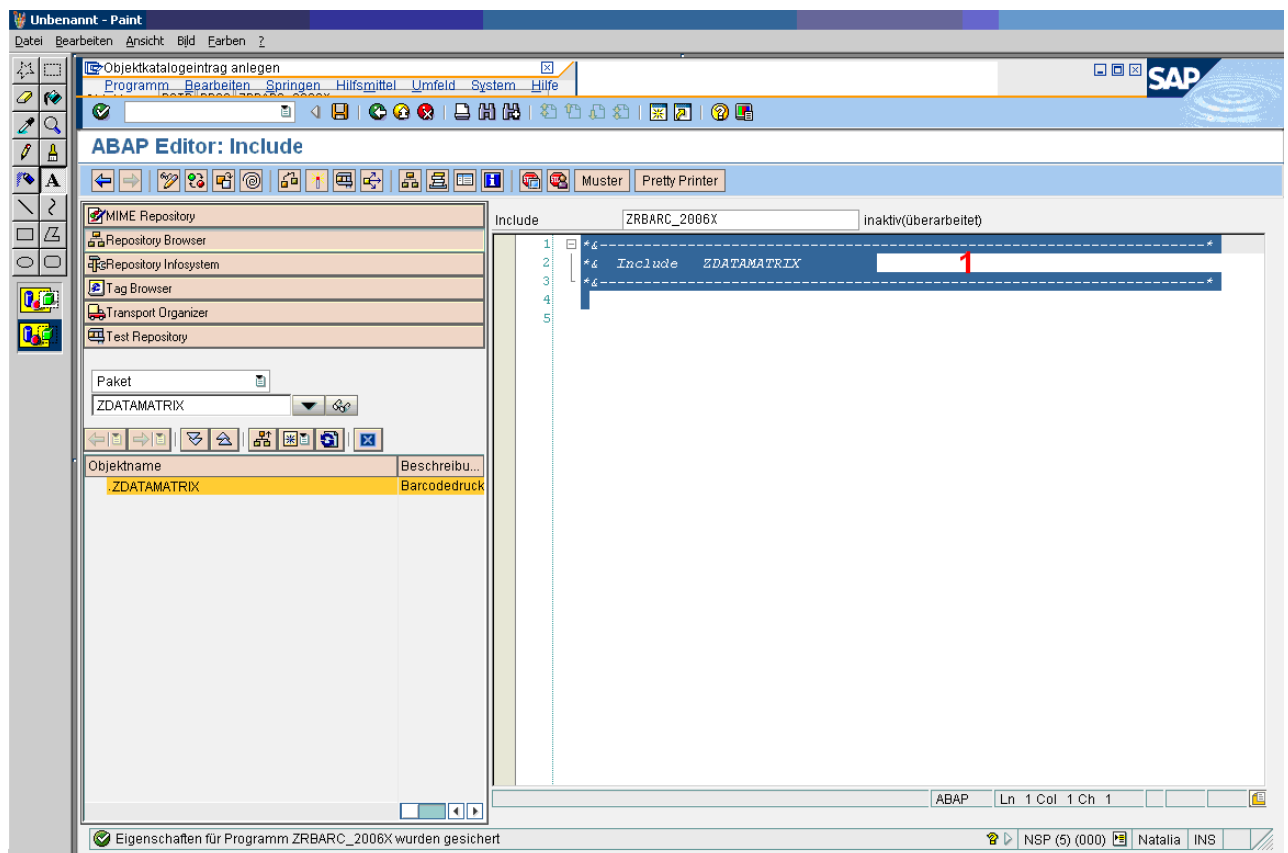
Originalsystem NSP

Originalsprache DE Deutsch

2 Lokales Objekt Sperrübersicht

7. Es erscheint der Dialog für das Anlegen eines Transportauftrages. Dieser sollte zu diesem Zeitpunkt schon vorhanden sein, da er beim Anlegen des Pakets (Entwicklungsklasse) **ZDATAMATRIX** erstellt wurde. Die Auftragsnummer sollte also diesmal die gleiche sein, wie beim Anlegen des Paketes (**1**).

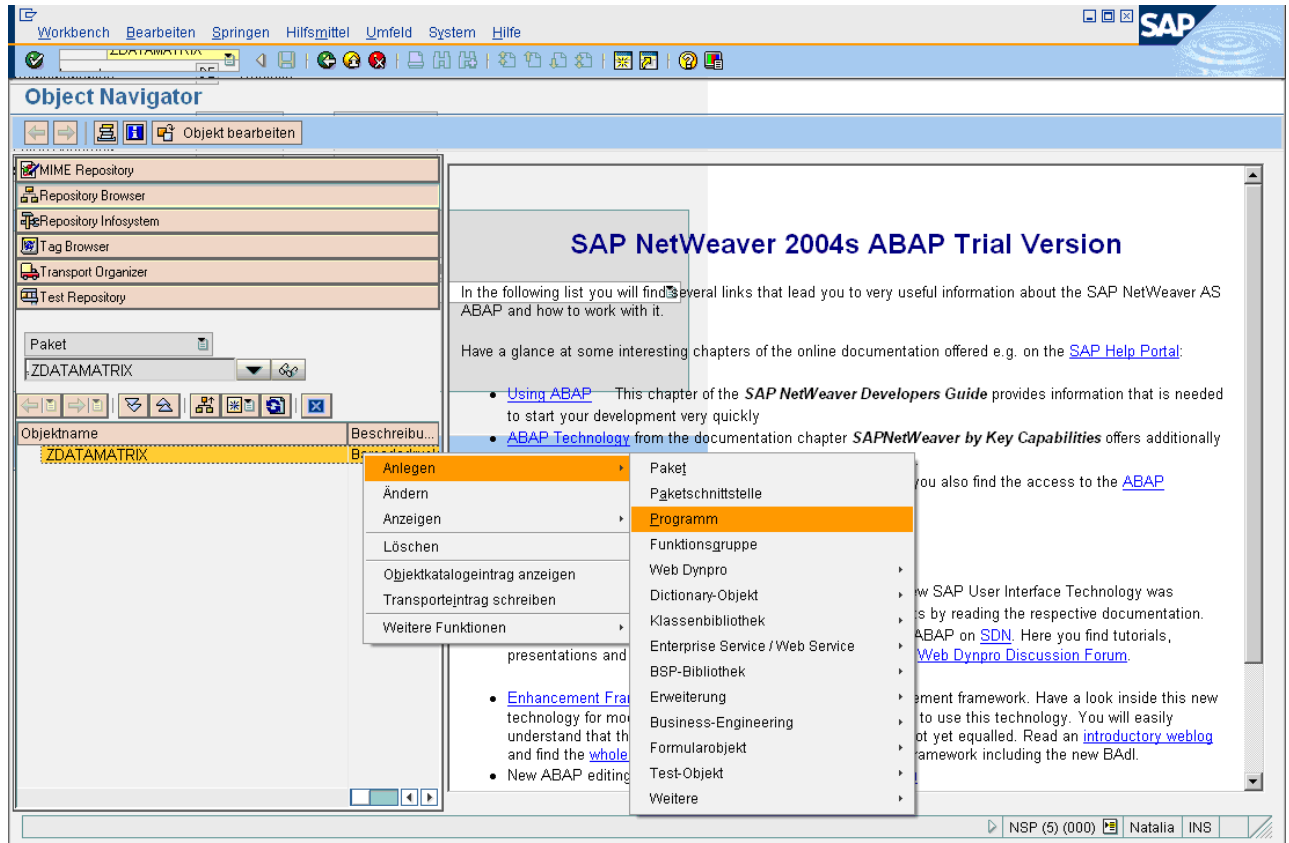
8. Klicken Sie auf das grüne Häkchen (**2**), um den Transportauftrag zu speichern.
9. Das Include wird gespeichert und der Quellcode (noch fast leer) erscheint auf der rechten Bildschirmseite.



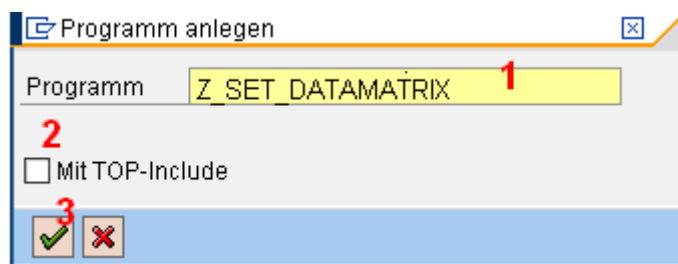
10. Löschen Sie den Inhalt des Quellcodes, der beim Anlegen des Includes automatisch erzeugt wurde.
11. Fügen Sie den Quellcode aus der entsprechenden Installationsdatei über die Zwischenablage ein und speichern Sie das Include. Damit ist die Installation des Includes abgeschlossen.
(Gehe zu Seite 52: [Anhang 4: Inhalt der Installationsdateien in ABAP Programme einfügen.](#))

11 Anhang 3: Ein Programm in der ABAP Workbench anlegen.

1. Falls noch nicht geschehen, starten Sie den **Objekt Navigator** (Transaktion **SE80**) und wählen Sie das angelegte Paket (Entwickungsklasse) **ZDATAMATRIX**.



2. Markieren Sie den Objektnamen **ZDATAMATRIX** und klicken Sie auf die rechte Maustaste.
3. Wählen Sie aus dem Kontextmenü **Anlegen/Programm**.
4. Geben Sie in dem folgenden Dialog einen Namen für das anzulegende Programm (**1**) an, wählen Sie das Häkchen „Mit TOP-Include“ ab (**2**) und klicken Sie auf das grüne Häkchen (**3**).



5. Klicken Sie in dem folgenden Dialog auf „Sichern“ (1), um das Programm zu erstellen.

ABAP: Programmeigenschaften

Titel: Report Z_SET_DATAMATRIX

Originalsprache: DE Deutsch

Erstellt: 12.01.2007 BCUSER

Letzte Änderung:

Status: neu(überarbeitet)

Attribute

Typ: Ausführbares Programm

Status:

Anwendung:

Berechtigungsgruppe:

Logische Datenbank:

Selektionsbildversion:

☐ Editorsperre ☒ Festpunktarithmetik

☒ Unicodeprüfungen aktiv ☐ Start über Variante

1 Sichern

6. Achten Sie darauf, dass das Programm dem neu angelegten Paket (Entwicklungsklasse) **ZDATAMATRIX** hinzugefügt wird (1) und klicken Sie auf das Diskettensymbol (2) um das Programm zu sichern.

Objektkatalogeintrag anlegen

Objekt: R3TR PROG ZDATAMATRIX

Attribute

Paket: ZDATAMATRIX 1

Verantwortlicher: CUST

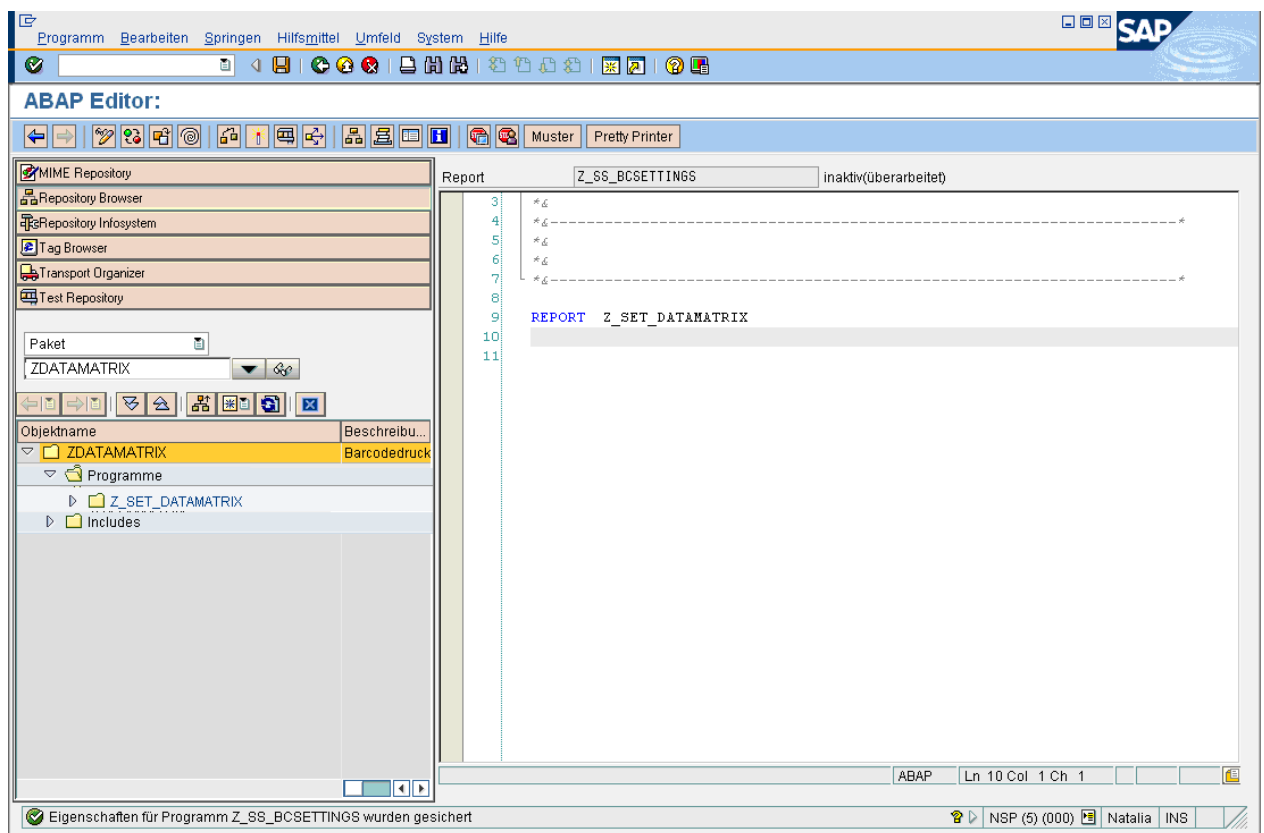
Originalsystem: NSP

Originalsprache: DE Deutsch

2 Lokales Objekt Sperrübersicht

7. Es erscheint der Dialog für das Anlegen eines Transportauftrages. Dieser sollte zu diesem Zeitpunkt schon vorhanden sein, da er beim Anlegen des Pakets (Entwicklungsklasse) **Z_SET_DATAMATRIX** erstellt wurde. Die Auftragsnummer sollte also diesmal die gleiche sein, wie beim Anlegen des Paketes (1).

8. Klicken Sie auf das grüne Häkchen (2), um den Transportauftrag zu speichern.
9. Das Programm wird gespeichert und der Quellcode (noch fast leer) erscheint auf der rechten Bildschirmseite.

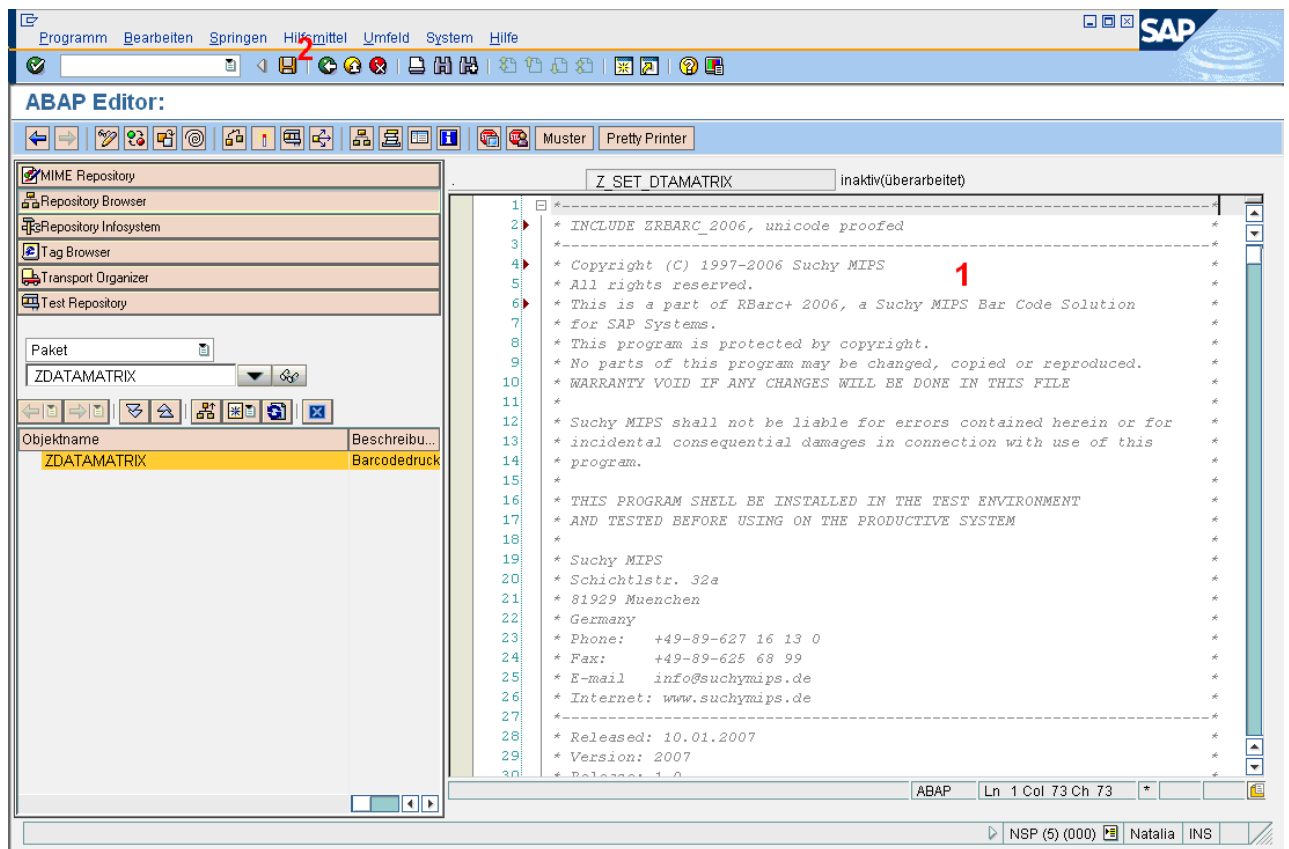


10. Löschen Sie den Inhalt des Quellcodes, der beim Anlegen des Includes automatisch erzeugt wurde.
11. Fügen Sie den Quellcode aus der entsprechenden Installationsdatei über die Zwischenablage ein und speichern Sie das Programm. Damit ist die Installation des Programms abgeschlossen.
(Gehe zu Seite 52: [Anhang 4: Inhalt der Installationsdateien in ABAP Programme einfügen.](#))

12 Anhang 4: Inhalt der Installationsdateien in ABAP Programme einfügen.

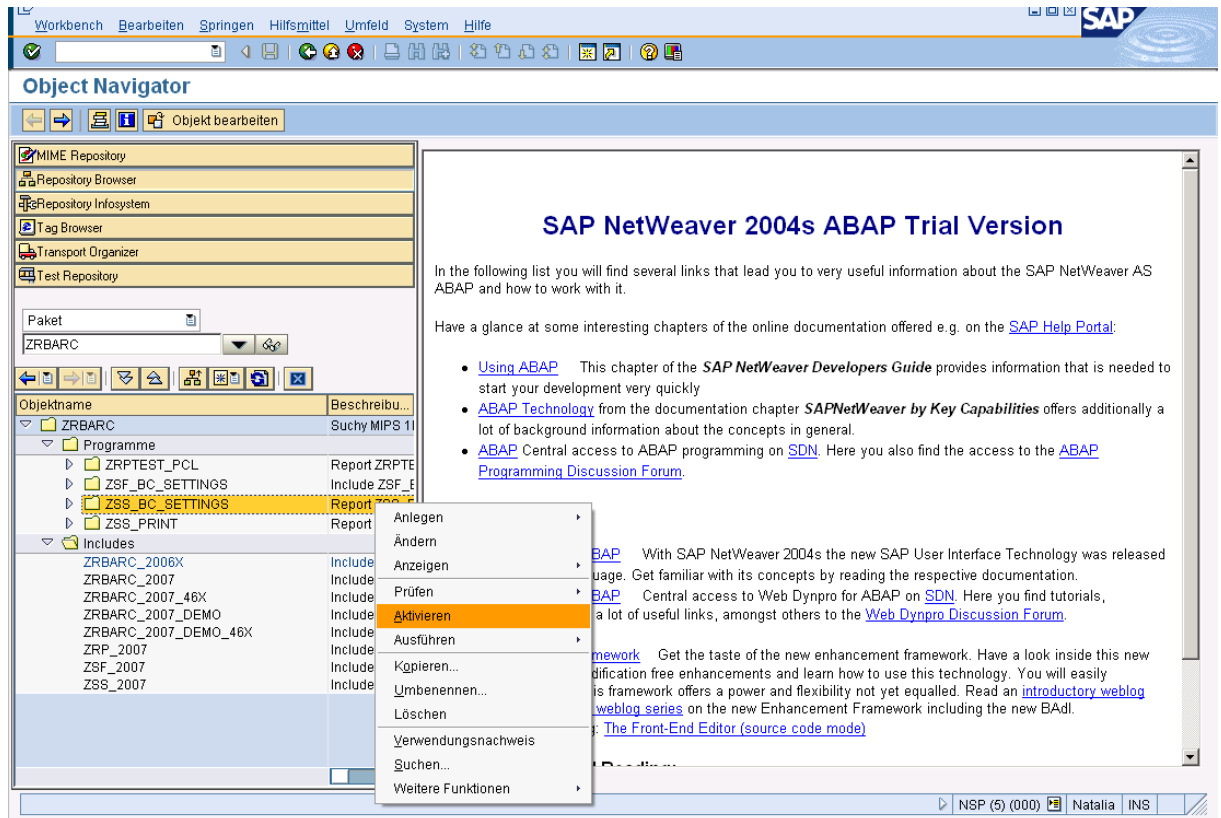
Dieser Vorgang betrifft sowohl Includes als auch Programme. Er wird anhand der Datei **Z_SET_DATAMATRIX** erläutert, gilt aber grundsätzlich für alle Installationsdateien mit der Erweiterung **.PRG** und **.INC**.

1. Öffnen Sie die Datei **Z_SET_DATAMATRIX.PRG** aus dem Verzeichnis **Install** mit dem Windows Editor (oder einem anderen Texteditor, falls Sie nicht unter Windows arbeiten).
2. Markieren Sie den gesamten Inhalt der Datei mit den Tasten **Strg+A**.
3. Kopieren Sie den Inhalt der Datei in die Zwischenablage mit den Tasten **Strg+C**.
4. Wechseln Sie im SAPGui zum geöffneten Quellcode des Programms **Z_SET_DATAMATRIX**.
5. Klicken Sie mit der Maus in das Fenster mit dem ABAP Quellcode (1) und drücken Sie die Tasten **Strg+V**, um den Quellcode aus der Zwischenablage einzufügen.
6. Klicken Sie anschließend auf das Diskettensymbol (2), um den Quellcode zu speichern.

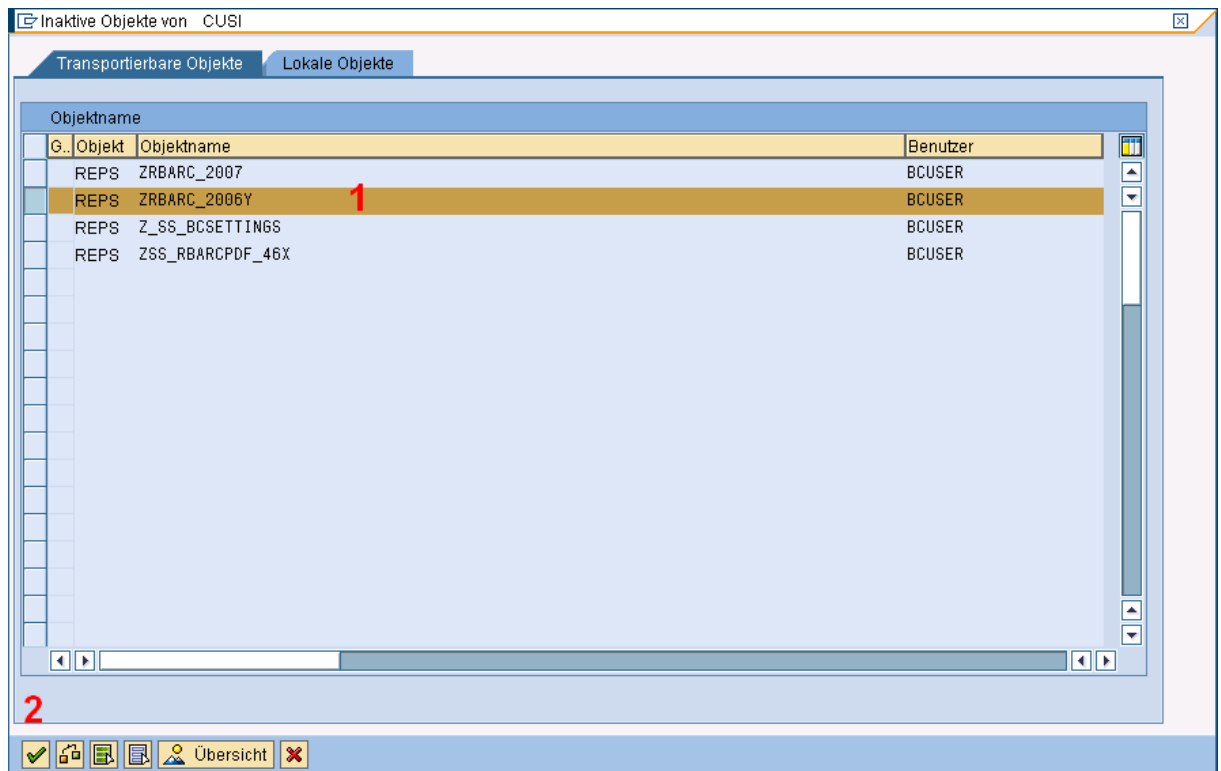


13 Anhang 5: Aktivieren von Includes und Programmen

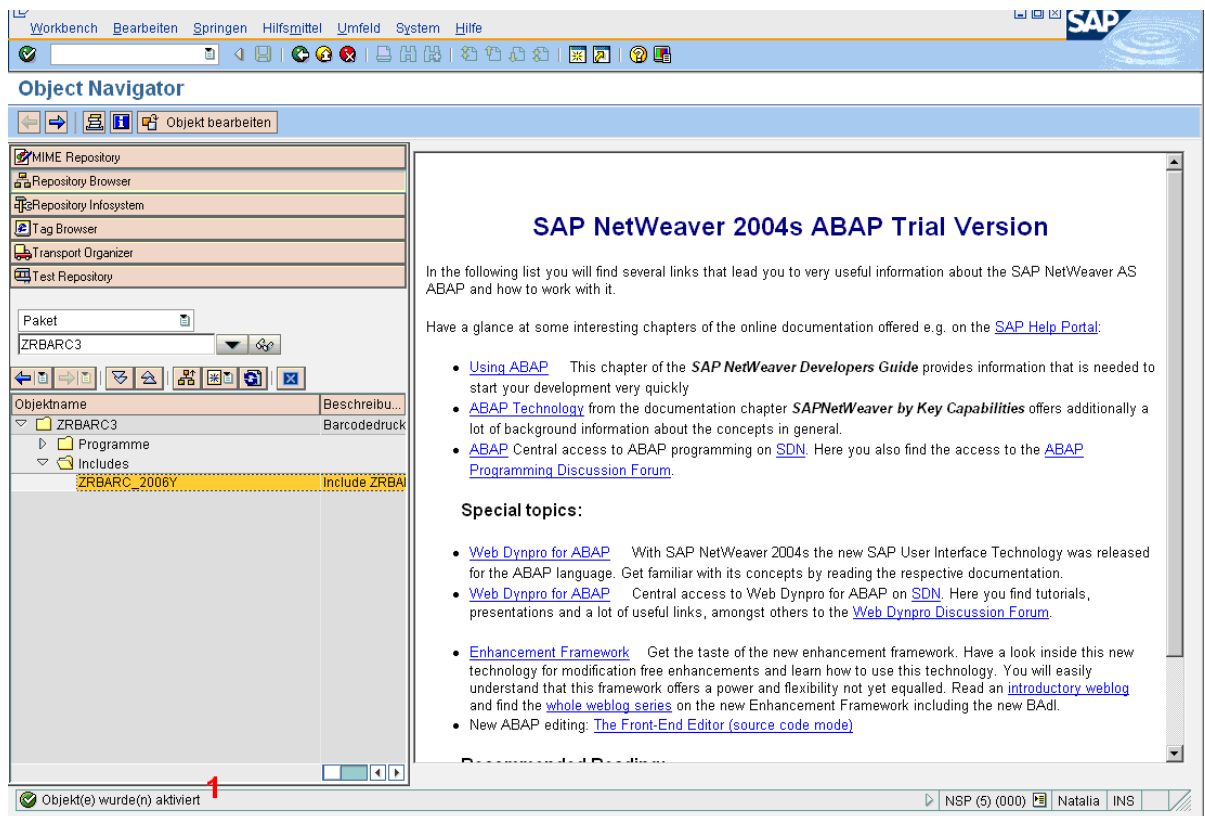
1. Öffnen Sie im SAPGui den **Objekt Navigator** (Transaktion **SE80**).
2. Wählen Sie als Objekttyp „**Paket**“
3. Geben Sie den Paketnamen **ZDATAMATRIX** ein.
4. Im Fenster Objektname erscheint der Name des Pakets mit den Baumknoten „**Programme**“ und „**Includes**“.
5. Expandieren Sie den Baumknoten **Programme** (1) und den Baumknoten **Includes** (2).
6. Wählen Sie das zu aktivierende Objekt, halten Sie die rechte Maustaste gedrückt und wählen Sie aus dem Kontextmenü „**Aktivieren**“



7. Es erscheint die Liste aller inaktiven Objekte. Das gewählte Objekt ist markiert (1). Klicken Sie auf das grüne Häkchen (2) um mit dem Vorgang fortzusetzen.

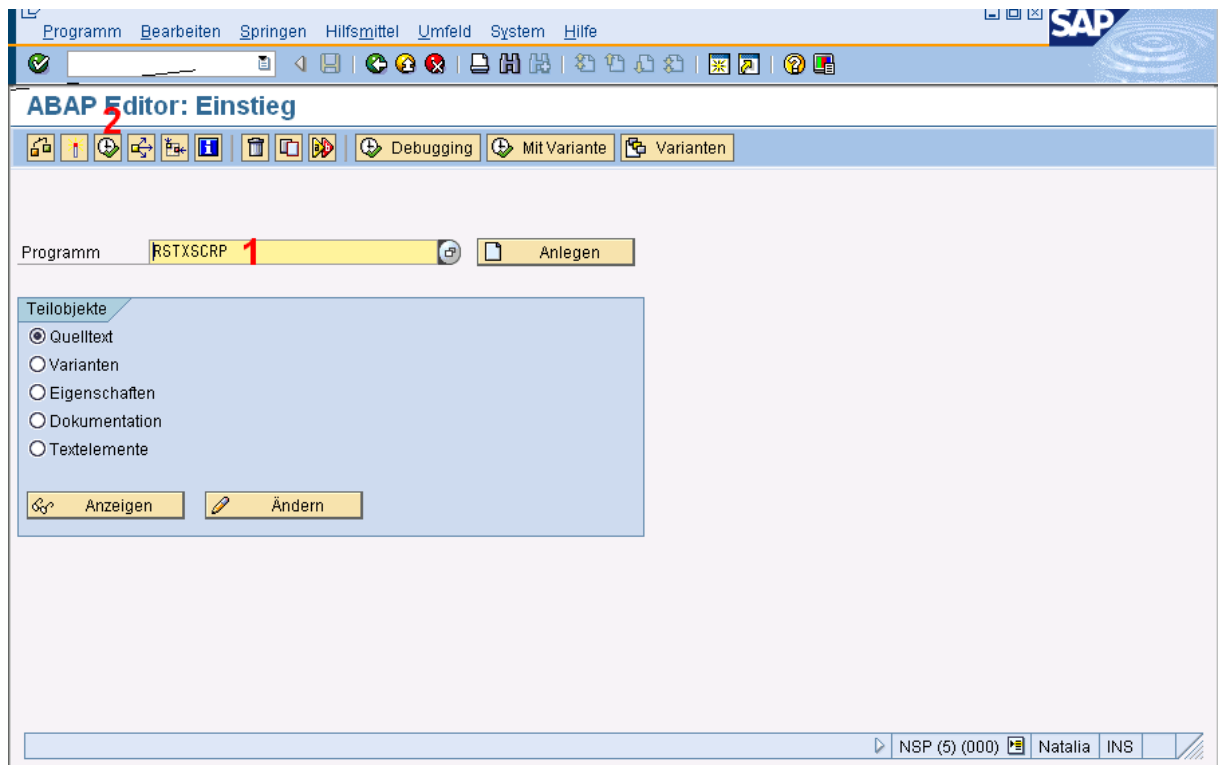


8. Nach der Aktivierung erfolgt eine Meldung in der Statusleiste (1). Nur bei erfolgreicher Aktivierung können die Programme einwandfrei arbeiten.



14 Anhang 6: Ein ABAP Programm ausführen.

1. Starten Sie die Transaktion **SE38**.
2. Geben Sie im Feld „**Programm**“ den Namen des auszuführenden Programms ein (**1**).
3. Klicken Sie im Menü auf das Startsymbol (**2**).



15 Anhang 7: Ein SAPscript Formular importieren (hochladen).

1. Starten Sie das SAP Standardprogramm **RSTXSCR**.
(Gehe zu: [Anhang 6: Ein ABAP Programm ausführen](#))
2. Wählen Sie den Objekttyp „**Formular**“ (1).
3. Tragen Sie im Feld „Objektname“ **ZSS_DM_FORM** ein (2).
4. Tragen Sie den Modus „**IMPORT**“ ein (3).
5. Klicken Sie auf das Startsymbol (4)

Programme Bearbeiten Springen System Hilfe

Upload/Download von SAPscript-Objekten

Objektselektion und Modussteuerung

☒ Formular 1
☐ Stil
☐ Standardtext
Text-ID ST
Sprache DE
☐ Gerätetyp
Objektname ZSS_DM_FORM 2
Modus (EXPORT,IMPORT) IMPORT 3

Steuerparameter für Datei-Operation

☒ Von/auf Frontend
☐ Von/auf Applikationsserver
Dateiname C:\temp*****&&&
☐ Binäres Dateiformat
☐ Dateinhalt anzeigen

Kontrolle über Sprachversionen

Sprachenvektor
☐ Nur Originalsprache export.

6. Es erscheint der Dialog für den Objektkatalogeintrag. Fügen Sie das Formular dem Paket (Entwicklungsgruppe) **ZDATAMATRIX** hinzu (1). Klicken Sie auf das Diskettensymbol (2) um den Eintrag zu speichern.

Objektkatalogeintrag anlegen

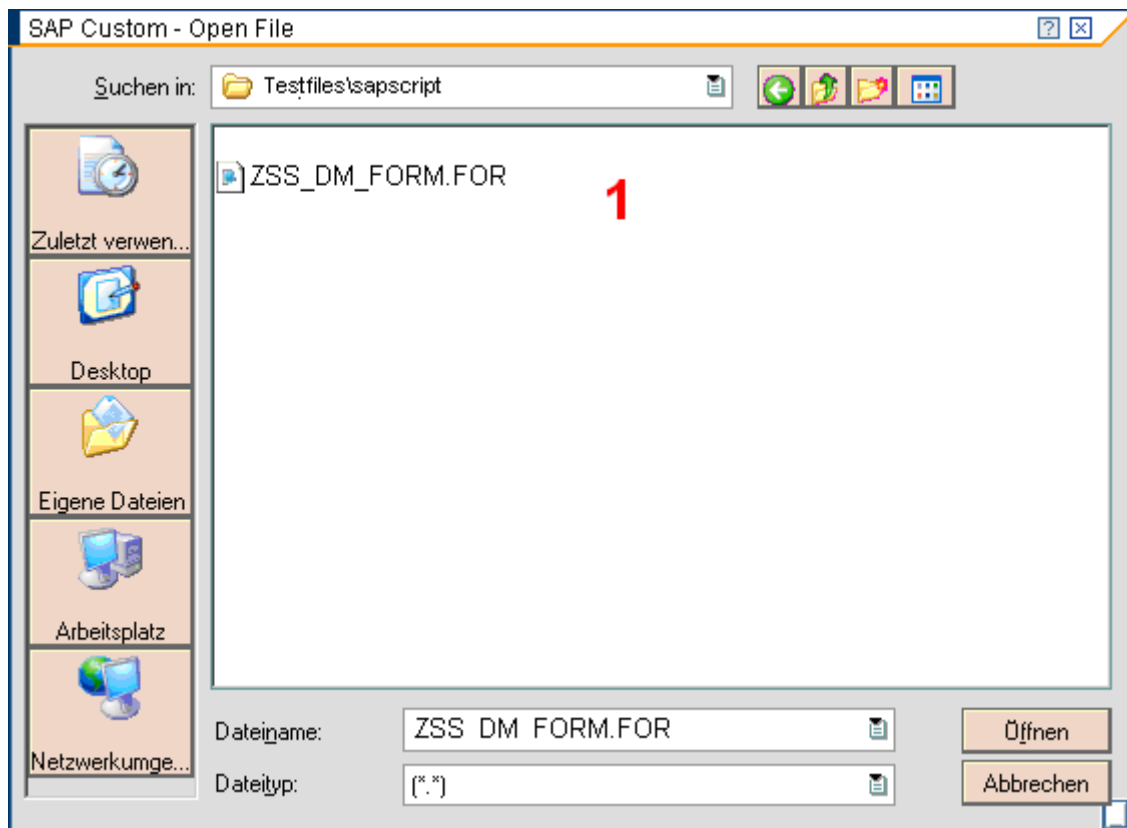
Objekt R3TR FORM ZSS_DM_FORM

Attribute

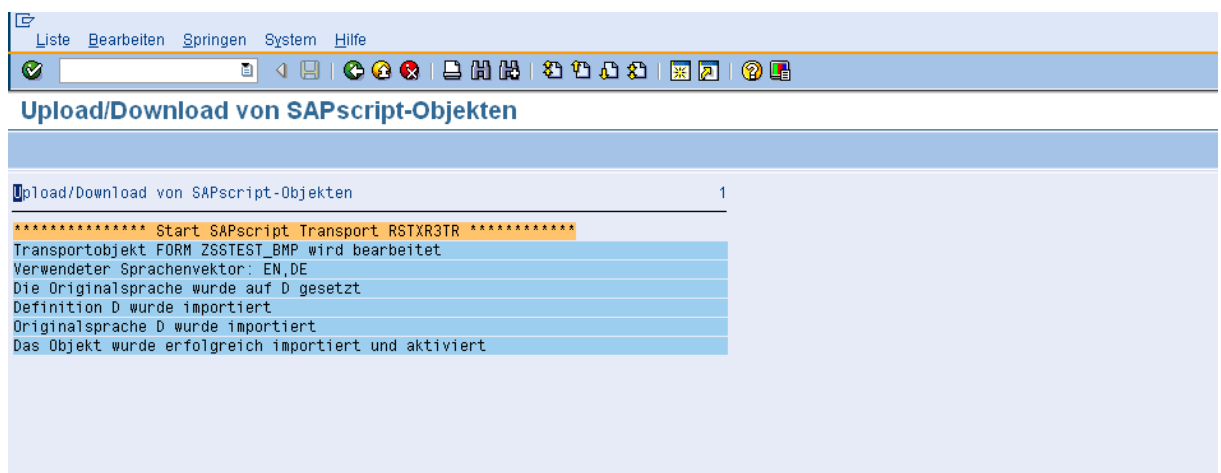
Paket ZDATAMATRIX 1
Verantwortlicher CUST
Originalsystem NSP
Originalsprache

2 Lokales Objekt Sperrübersicht

7. Es erscheint der Dialog für den Transportauftrag. Lassen Sie den bisherigen Transportauftrag bestehen (1) und klicken Sie auf das grüne Häkchen, um den Eintrag zu speichern (2).
8. Jetzt erscheint das Dateiauswahl Menü. Wählen Sie die Datei **ZSS_DM_FORM.FOR** aus dem Verzeichnis **TestForms** (1) und klicken Sie auf **Öffnen**.

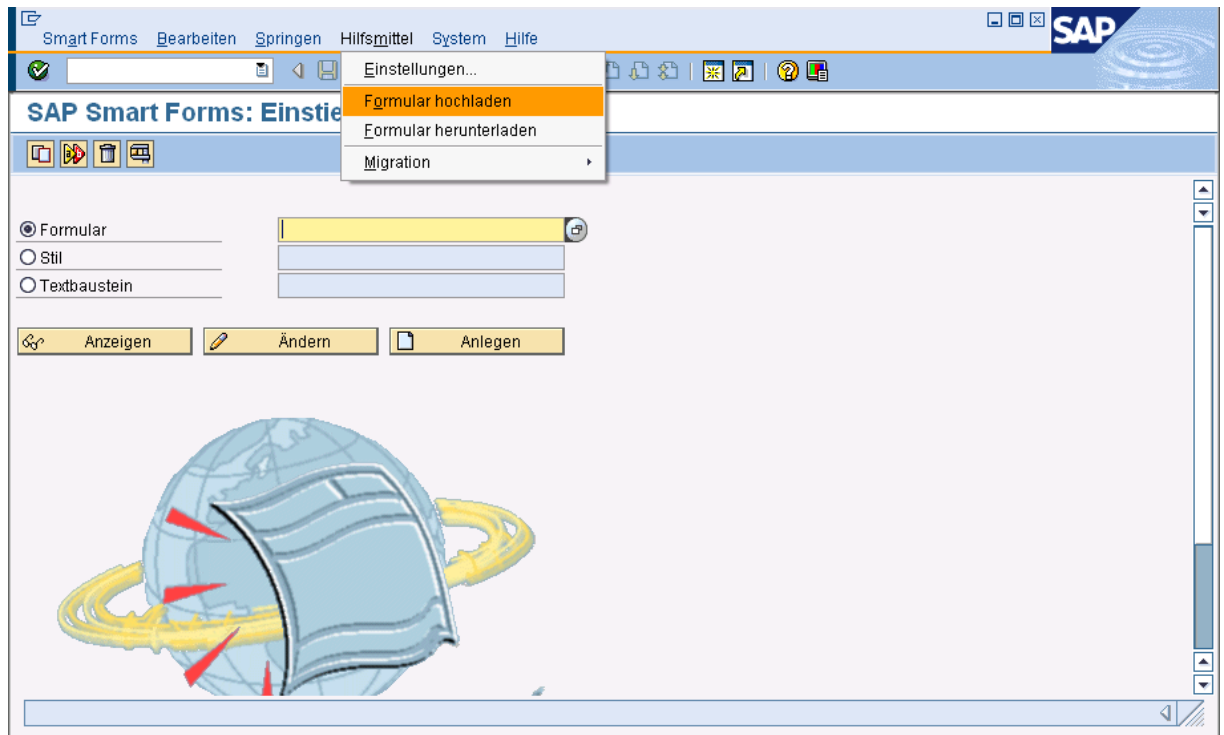


9. Das Formular wird importiert und zum Schluss erscheint eine Meldung, ähnlich, wie diese:

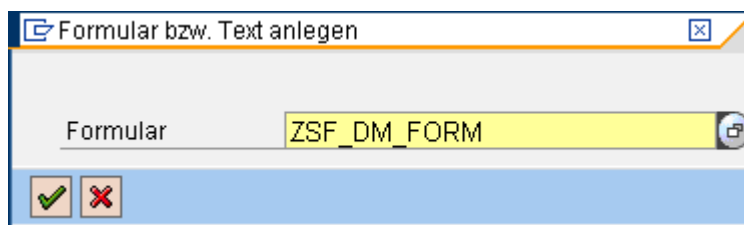


16 Anhang 8: Ein Smart Forms Formular hochladen.

1. Starten Sie die Transaktion **Smartforms**.
2. Wählen Sie aus dem Menü **Hilfsmittel / Formular hochladen**

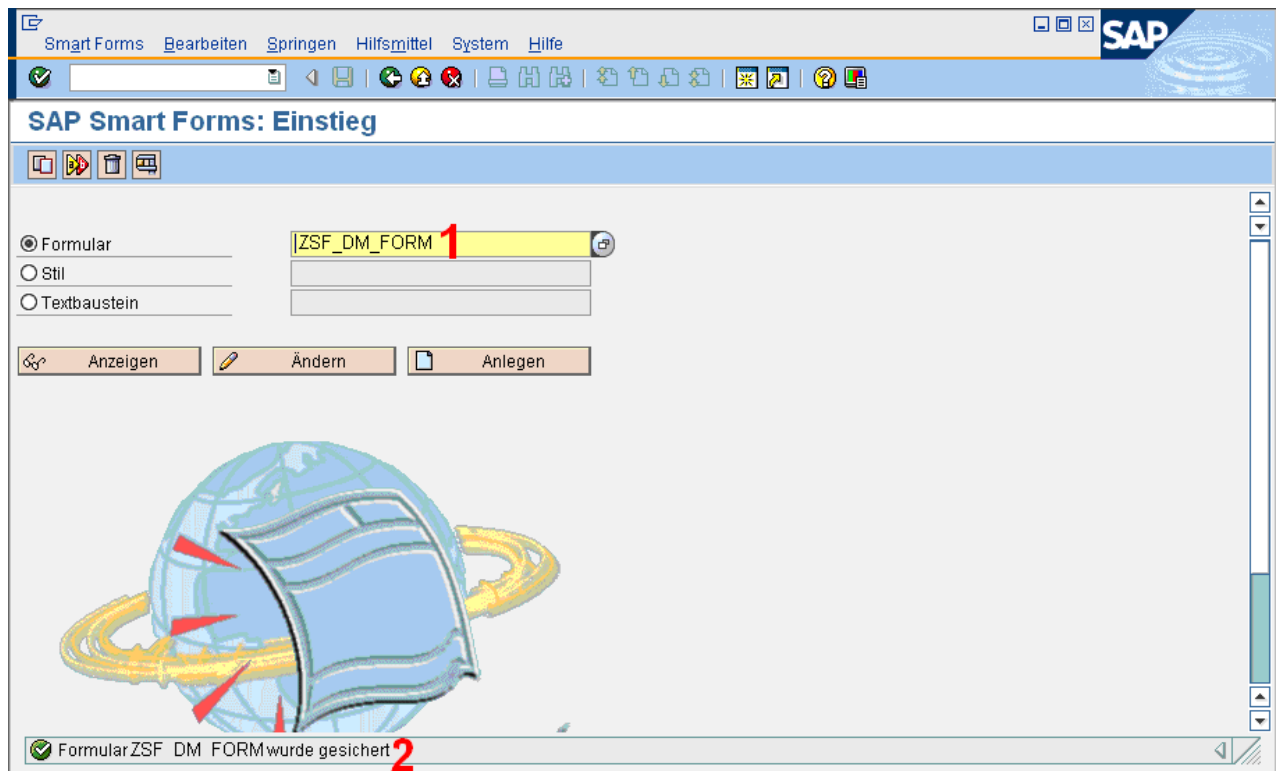


3. Geben Sie im nächsten Dialog den Formularnamen **ZSF_DM_FORM** ein und klicken Sie auf das grüne Häkchen.



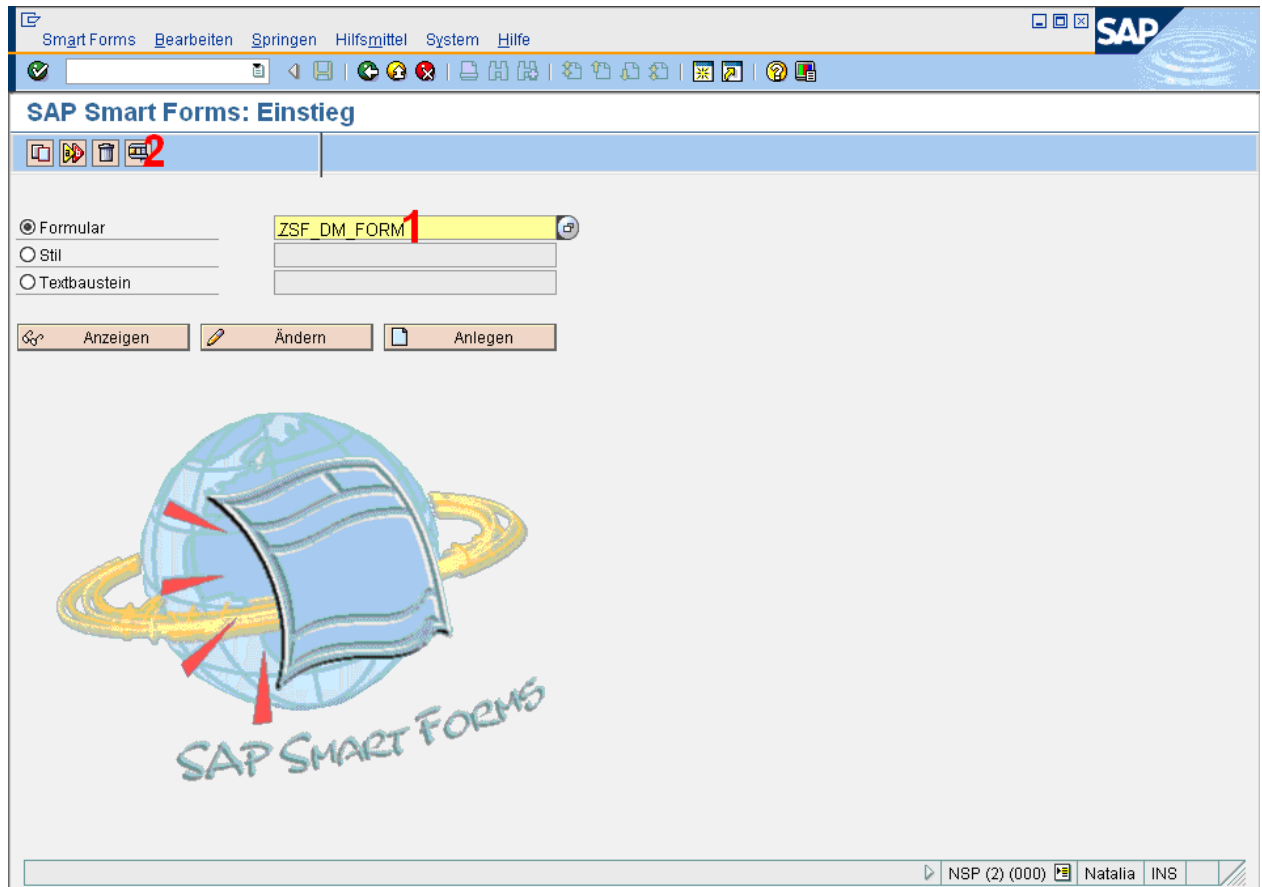
4. Es erscheint der Dateiauswahl Dialog. Wählen Sie die Datei **ZSF_DM_FORM.XML** aus dem Verzeichnis **TestForms** und klicken Sie auf **Öffnen**.

5. Das Formular wird hochgeladen. Danach erscheint der Name des hochgeladenen Formulars im Feld **Formular** (1) und eine Meldung in der Statusleiste (2)

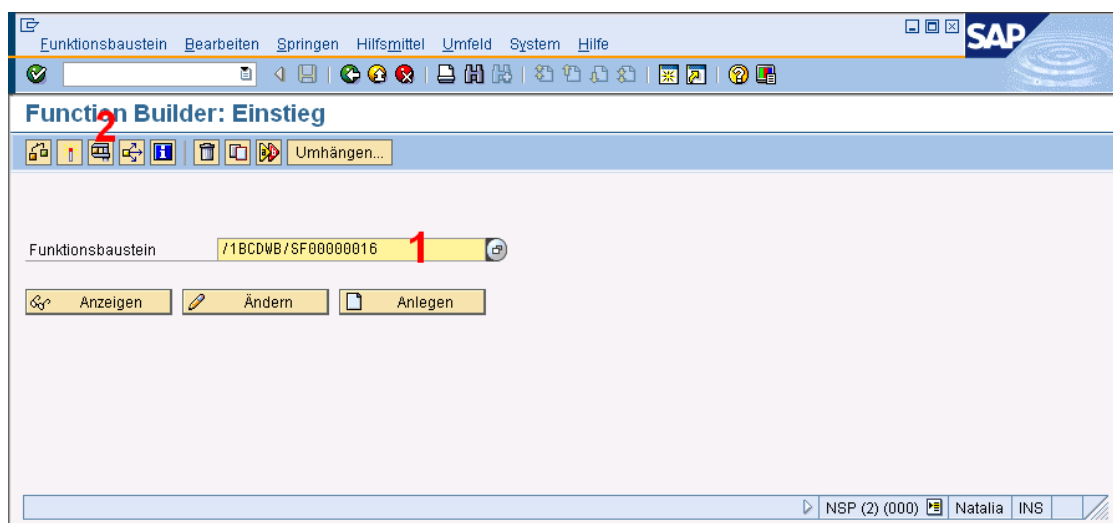


17 Anhang 9: Smart Forms Formular testen.

1. Starten Sie die Transaktion **SmartForms**.
2. Geben Sie im Feld **Formular** den Namen **ZSF_DM_FORM** ein (1) und klicken Sie anschließend auf das Symbol **Testen** (2).



3. Das System erzeugt einen Funktionsbaustein, der im nächsten Dialog erscheint (1). Klicken Sie erneut auf das Symbol **Testen** (2).



4. Klicken Sie im nächsten Dialog auf das Symbol **Ausführen** (1).

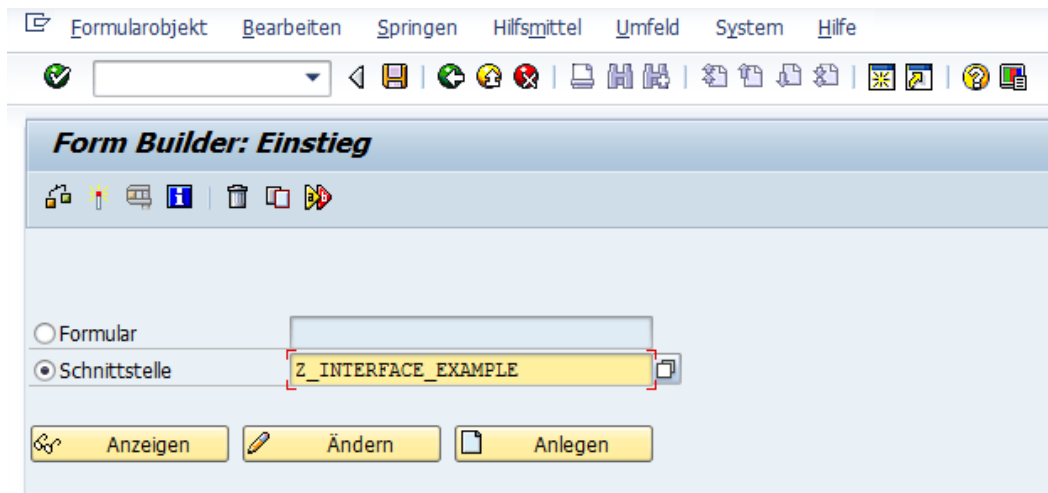
The screenshot shows the SAP 'Funktionsbaustein testen: Eingabebild' (Function Block Test: Input Screen) dialog. The title bar includes 'FBAusteine', 'Bearbeiten', 'Springen', 'Hilfsmittel', 'System', and 'Hilfe'. The main area contains a 'Debugging' button and a table for 'Import-Parameter'. The table has two columns: 'Import-Parameter' and 'Wert'. The parameters listed are ARCHIVE_INDEX, ARCHIVE_INDEX_TAB, ARCHIVE_PARAMETERS, CONTROL_PARAMETERS, MAIL_APPL_OBJ, MAIL_RECIPIENT, MAIL_SENDER, and OUTPUT_OPTIONS. The values are 00.00.0000, 0 Einträge, and <Initial> respectively. A red arrow points to the 'Ausführen' (Execute) button in the top toolbar.

5. Danach startet das Druckprogramm. Geben Sie den Namen des Druckers ein, auf dem der Barcode gedruckt werden soll (1).
6. Setzen Sie ein Kästchen im Feld **Sofort Ausgeben** (2)
7. Klicken Sie auf Drucken (3), falls Sie das Formular mit den Barcodes auf dem Drucker ausgeben möchten, oder auf Druckansicht (4), falls Sie das Formular mit den Barcodes auf dem Bildschirm betrachten möchten.

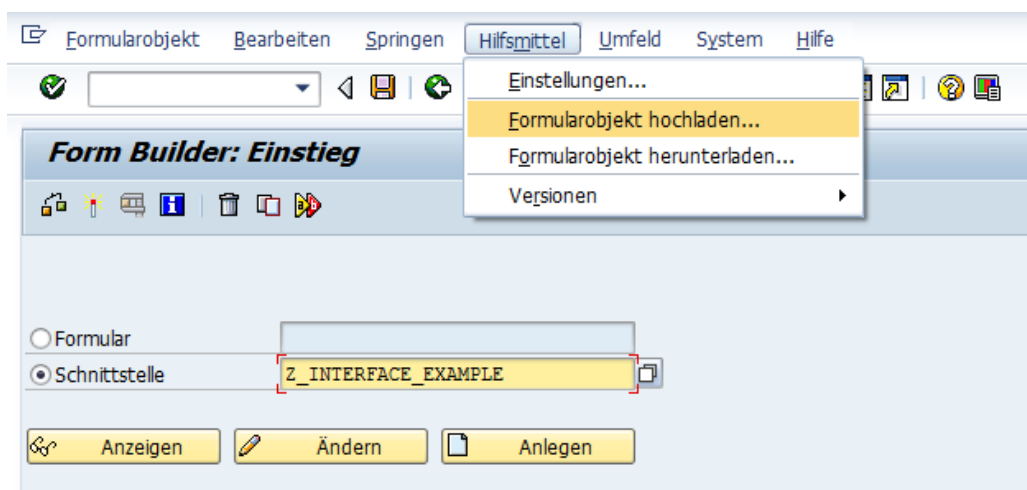
The screenshot shows the SAP 'Drucken:' (Print) dialog. It contains several sections: 'Ausgabegerät' (Output Device) with a text field containing 'LP01' (marked with a red 1), 'Seitenausw.' (Page Selection), 'Spool-Auftrag' (Spool Job) with fields for 'Name' (SMART), 'Titel' (empty), and 'Berechtigung' (empty), 'Spool-Steuerung' (Spool Control) with a checked 'Sofort ausgeben' (Print Immediately) checkbox (marked with a red 2), 'Exemplare' (Copies) with a text field containing '1', and 'Deckblatteinstellungen' (Cover Sheet Settings) with a dropdown menu set to 'nicht ausgeben'. At the bottom, there are two buttons: 'Drucken' (Print) (marked with a red 3) and 'Druckansicht' (Print Preview) (marked with a red 4).

18 Anhang 10: Eine Interactive Forms Schnittstelle hochladen.

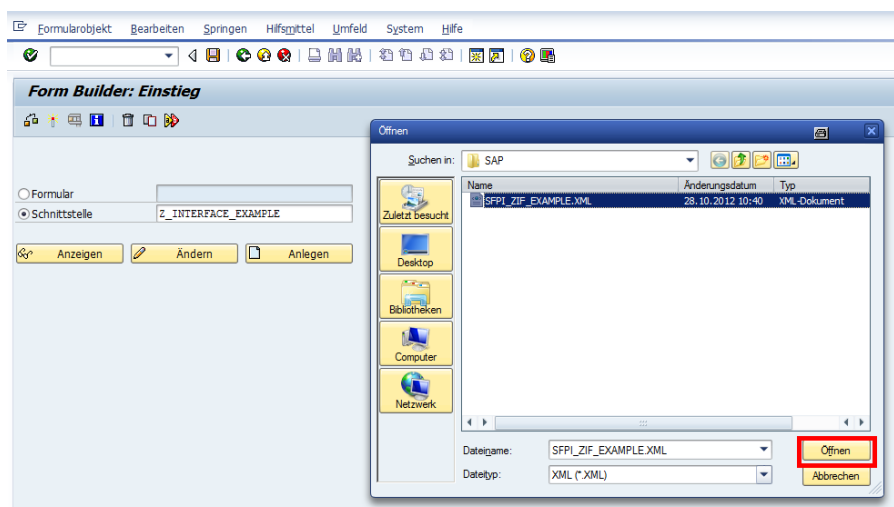
1. Starten Sie den FormBuilder durch Aufruf der Transaktion **SFP**.
2. Markieren Sie den Radiobutton **Schnittstelle** und geben Sie der Schnittstelle einen Namen.



3. Wählen Sie aus dem Menü **Hilfsmittel** → **Formularobjekt hochladen**.

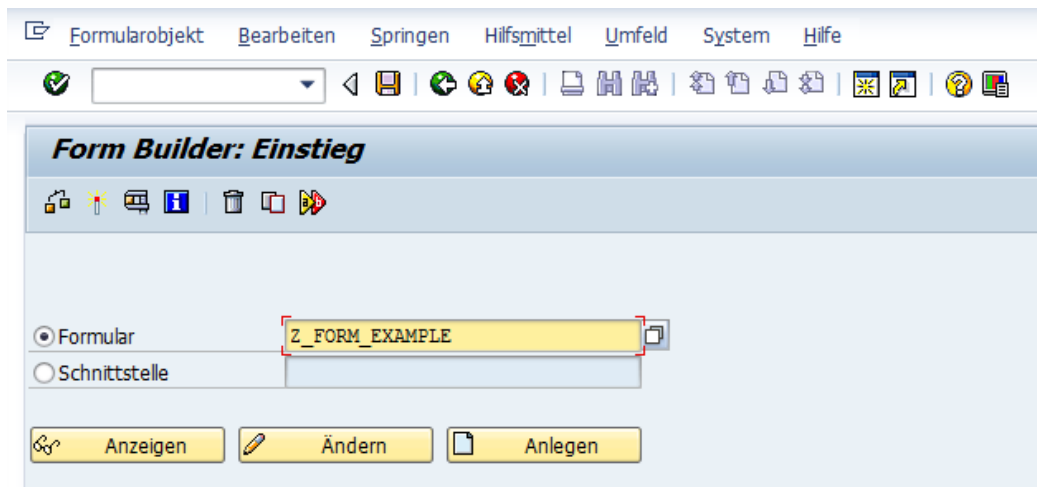


4. Wählen Sie aus dem Dateiselektor die XML-Datei aus und betätigen Sie den Button **Öffnen**.

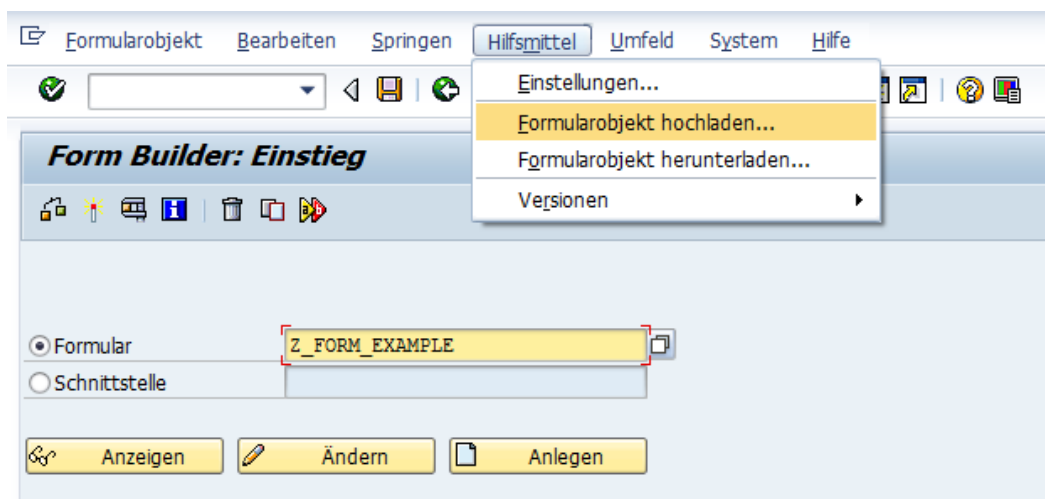


19 Anhang 11: Ein Interactive Forms Formular hochladen.

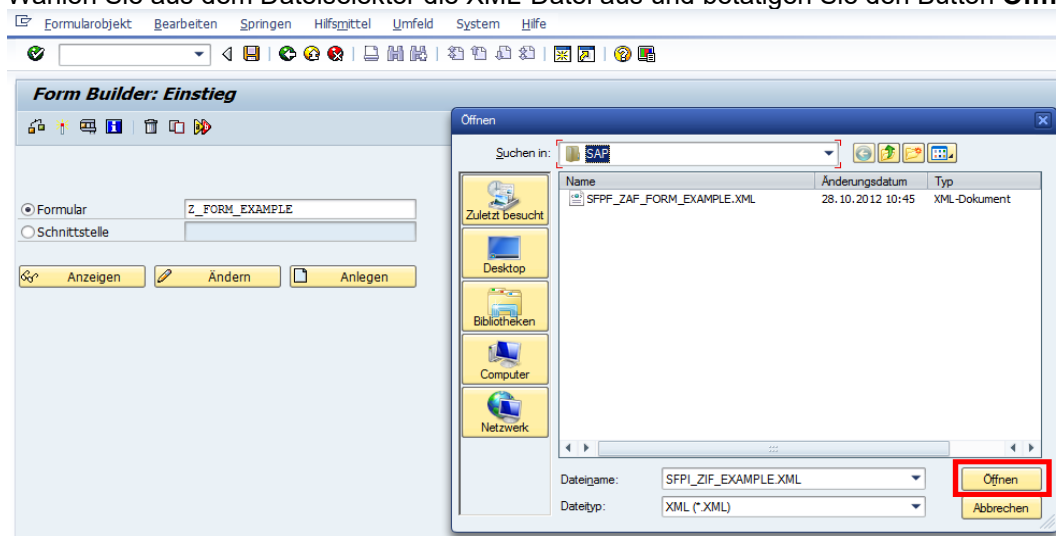
1. Starten Sie den FormBuilder durch Aufruf der Transaktion **SFP**.
2. Markieren Sie den Radiobutton **Formular** und geben Sie dem Formular einen Namen.



3. Wählen Sie aus dem Menü **Hilfsmittel** → **Formularobjekt hochladen**.

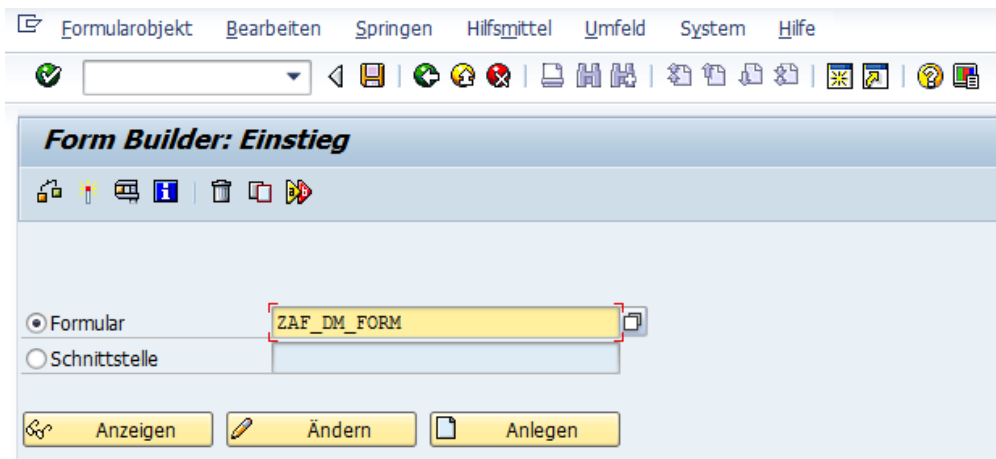


4. Wählen Sie aus dem Dateiselektor die XML-Datei aus und betätigen Sie den Button **Öffnen**.

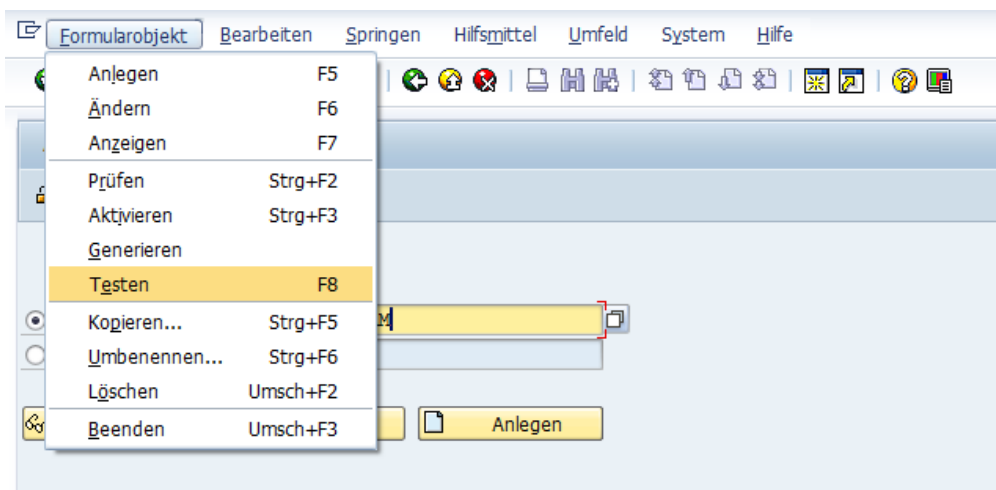


20 Anhang 12: Ein Interactive Forms Formular testen

1. Starten Sie den FormBuilder durch Aufruf der Transaktion **SFP**.
2. Markieren Sie den Radiobutton **Formular** und geben Sie den Namen des Formulars ein (hier im Beispiel **ZAF_DM_FORM**).

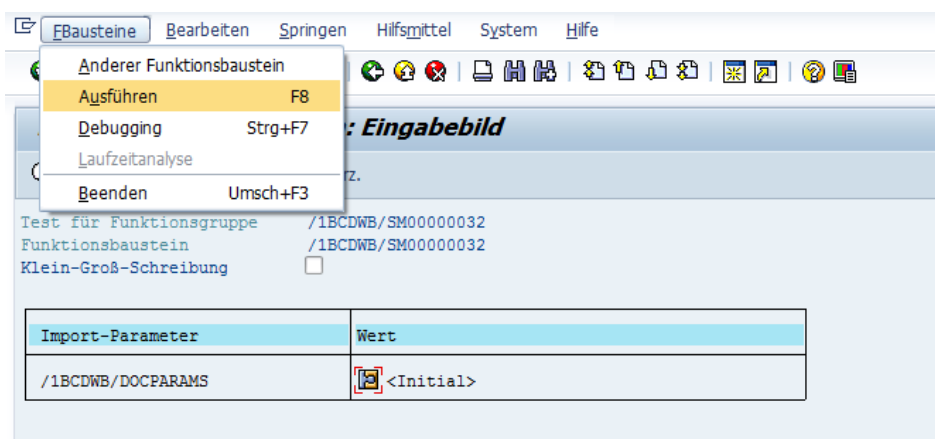


3. Wählen Sie im Menü **Formularobjekt** → **Testen**.



Der generierte Funktionsbaustein wird nun angezeigt.

4. Wählen Sie nun im Menü **FBausteine** → **Ausführen**.



5. Geben Sie einen Drucker ein, dem kein SAPWIN oder SWIN-Gerätetyp zugeordnet ist und betätigen Sie den Button **Druckbildansicht**.

System Hilfe

SAP

Drucken:

Ausgabegerät: LOCA

Spoolauftrag

Name: PBFORM LOCA XXXX

Text für Deckblatt:

Berechtigung:

Spool-Steuerung

☒ Sofort ausgeben

☒ Löschen nach Ausgabe

☐ Neuer Spool-Auftrag

☐ Spool-Auftrag abschließen

Spool-Verweildauer: 8 Tag(e)

Ablagemodus: 1 Nur Drucken

Exemplare

Anzahl Exemplare: 1

Deckblatteinstellungen

SAP-Deckblatt: nicht ausgeben

Empfänger:

Abteilung:

Drucken Druckansicht

6. Das Formular sollte nun in der Druckbildansicht zu sehen sein.

Print Preview of LP01 Page 00001 of 00001

RBarc/Datamatrix

2D Barcode GS1 Datamatrix ECC 200 generated on a SAP System

A solution from Suchy MIPS www.suchymips.de

This demo presents 2 Datamatrix-Labels with identical content but encoded in different ways.

Both Datamatrix codes encode following data: '[(>[RS]06[GS]2L'

[RS] means the function code for Record Separator (ASCII Value 30 dec)

[GS] means the function code for Group Separator (ASCII Value 29 dec)

The left Datamatrix-Code was encoded using different variables with different Data Type Flags:

dmvar1 = '[(>'. dflag = 'T'.

dmvar2 = 'RS'. dflag = 'F'.

dmvar3 = '06'. dflag = 'F'.

dmvar4 = 'GS'. dflag = 'F'.

dmvar5 = '2L'. dflag = 'T'.

The right Datamatrix-Code was encoded using one variable which encodes all data in ASCII-HEX.

dmvar1 = '5B283E1E30361D324C'.

DFLAG = 'X'.

Available Data Type Flags for data encoding are:

T - any text

X - data encoded as an ASCII-Hex string (e.g. 5B283E1E...)

F - function codes like 'GS' 'RS' 'EOT'

B - Byte with values 0 - 255

L - ECI (Extended Channel Interpretation)

For more details refer to the manual.