

BC - ABAP Workbench Tutorial



Release 4.6C



Copyright

© Copyright 2001 SAP AG. Alle Rechte vorbehalten.

Weitergabe und Vervielfältigung dieser Publikation oder von Teilen daraus sind, zu welchem Zweck und in welcher Form auch immer, ohne die ausdrückliche schriftliche Genehmigung durch SAP AG nicht gestattet. In dieser Publikation enthaltene Informationen können ohne vorherige Ankündigung geändert werden.

Die von SAP AG oder deren Vertriebsfirmen angebotenen Software-Produkte können Software-Komponenten auch anderer Software-Hersteller enthalten.

Microsoft[®], WINDOWS[®], NT[®], EXCEL[®], Word[®], PowerPoint[®] und SQL Server[®] sind eingetragene Marken der Microsoft Corporation.

IBM[®], DB2[®], OS/2[®], DB2/6000[®], Parallel Sysplex[®], MVS/ESA[®], RS/6000[®], AIX[®], S/390[®], AS/400[®], OS/390[®] und OS/400[®] sind eingetragene Marken der IBM Corporation.

ORACLE[®] ist eine eingetragene Marke der ORACLE Corporation.

INFORMIX[®]-OnLine for SAP und Informix[®] Dynamic Server[™] sind eingetragene Marken der Informix Software Incorporated.

UNIX[®], X/Open[®], OSF/1[®] und Motif[®] sind eingetragene Marken der Open Group.

HTML, DHTML, XML, XHTML sind Marken oder eingetragene Marken des W3C[®], World Wide Web Consortium, Massachusetts Institute of Technology.

JAVA[®] ist eine eingetragene Marke der Sun Microsystems, Inc.

JAVASCRIPT[®] ist eine eingetragene Marke der Sun Microsystems, Inc., verwendet unter der Lizenz der von Netscape entwickelten und implementierten Technologie.

SAP, SAP Logo, R/2, RIVA, R/3, ABAP, SAP ArchiveLink, SAP Business Workflow, WebFlow, SAP EarlyWatch, BAPI, SAPPHIRE, Management Cockpit, mySAP.com Logo und mySAP.com sind Marken oder eingetragene Marken der SAP AG in Deutschland und vielen anderen Ländern weltweit. Alle anderen Produkte sind Marken oder eingetragene Marken der jeweiligen Firmen.

Symbole

Symbol	Bedeutung
	Achtung
	Beispiel
	Empfehlung
	Hinweis
	Syntax
	Tip

Inhalt

BC - ABAP Workbench Tutorial	6
BC - ABAP Workbench Tutorial	7
Benutzerhinweise	8
Inhaltsübersicht	9
Voraussetzungen	11
Terminologie	12
Plattform	13
Namenskonvention für SAP-Objekte.....	14
Weiterführende Dokumentation	15
Lektion 1: Workbench-Konzepte und -Werkzeuge	16
Überblick.....	17
Übung 1: Die Workbench aufrufen	18
Übung 2: Die Werkzeuge der Workbench	19
Übung 3: In Objektlisten navigieren	21
Übung 4: Programme anlegen	22
Übung 5: Sprungmarken setzen	24
Übung 6: Transaktionen anlegen.....	25
Zusammenfassung	26
Lektion 2: Mit Tabellen arbeiten	27
Überblick.....	28
Übung 1: Tabellenobjekte anlegen	30
Übung 2: Tabellenfelder angeben.....	31
Übung 3: Datenelemente und Domänen definieren	32
Übung 4: Domänen wiederverwenden	34
Übung 5: Festwerte definieren	36
Übung 6: Technische Einstellungen vornehmen	37
Übung 7: Tabellen aktivieren.....	38
Zusammenfassung	39
Lektion 3: Bildschirmbilder gestalten	40
Überblick.....	41
Übung 1: Bilder anlegen	42
Übung 2: Elemente auf einem Bild anordnen	43
Übung 3: Bilder überarbeiten	45
Übung 4: Das Layout überprüfen.....	47
Übung 5: Das OK-Feld setzen	48
Zusammenfassung	49
Lektion 4: GUI-Status definieren	50
Überblick.....	51
Übung 1: GUI-Status definieren	52
Übung 2: Menüs in die Oberfläche einfügen	53
Übung 3: Funktionstasten definieren	54

Übung 4: Drucktasten definieren	55
Übung 5: Abschließende Arbeiten	56
Zusammenfassung	57
Lektion 5: Transaktionen programmieren.....	58
Überblick.....	59
Übung 1: Ablauflogik schreiben	60
Übung 2: Module anlegen	61
Übung 3: Globale Variablen deklarieren	62
Übung 4: Module programmieren	63
Übung 5: Nachrichtenklasse anlegen.....	65
Übung 6: Transaktion testen	66
Übung 7: Fehlersuche mit dem Debugger	67
Zusammenfassung	68
Lektion 6: Arbeiten im Team.....	69
Überblick.....	70
Übung 1: Entwicklungsklasse anlegen	72
Übung 2: Liste der Änderungsaufträge überprüfen.....	73
Übung 3: Entwickler hinzufügen.....	74
Übung 4: Programm anlegen.....	75
Übung 5: Änderungsaufträge freigeben.....	76
Zusammenfassung	78

BC - ABAP Workbench Tutorial

[Benutzerhinweise \[Seite 8\]](#)

[Lektion 1: Workbench-Konzepte und -Werkzeuge \[Seite 16\]](#)

[Lektion 2: Mit Tabellen arbeiten \[Seite 27\]](#)

[Lektion 3: Bildschirmbilder gestalten \[Seite 40\]](#)

[Lektion 4: GUI-Status definieren \[Seite 50\]](#)

[Lektion 5: Transaktionen programmieren \[Seite 58\]](#)

[Lektion 6: Arbeiten im Team \[Seite 69\]](#)

BC - ABAP Workbench Tutorial

[Benutzerhinweise \[Seite 8\]](#)

[Lektion 1: Workbench-Konzepte und -Werkzeuge \[Seite 16\]](#)

[Lektion 2: Mit Tabellen arbeiten \[Seite 27\]](#)

[Lektion 3: Bildschirmbilder gestalten \[Seite 40\]](#)

[Lektion 4: GUI-Status definieren \[Seite 50\]](#)

[Lektion 5: Transaktionen programmieren \[Seite 58\]](#)

[Lektion 6: Arbeiten im Team \[Seite 69\]](#)

Benutzerhinweise

Benutzerhinweise

Diese Benutzerhinweise enthalten Informationen, mit denen Sie vertraut sein sollten, ehe Sie mit dem *ABAP Workbench Tutorial* beginnen. Die Benutzerhinweise umfassen:

[Inhaltsübersicht \[Seite 9\]](#)

[Voraussetzungen \[Seite 11\]](#)

[Plattform \[Seite 13\]](#)

[Namenskonvention für SAP-Objekte \[Seite 14\]](#)

[Weiterführende Dokumentation \[Seite 15\]](#)

Inhaltsübersicht

Dieses *Tutorial* führt Sie in die Werkzeuge der ABAP Development Workbench ein. Die Workbench enthält die Werkzeuge zur Erstellung einer ABAP-Anwendung.

Die sechs Lektionen des Tutorials geben Ihnen einen Einblick in das Erstellen einer Anwendung mit Hilfe der Development Workbench. Wenn Sie die Lektionen abgeschlossen haben, sollten Sie mit der Rolle, die jedes Werkzeug im Entwicklungsprozeß spielt, vertraut sein.

ABAP Workbench Tutorial lehrt Sie nicht, mit *ABAP* zu programmieren! [Weiterführende Dokumentation \[Seite 15\]](#) zeigt Ihnen, welche Dokumentation Sie nach Absolvierung des Tutorials lesen sollten.

Die einzelnen Lektionen

In den einzelnen Lektionen des Tutorials sind so aufgebaut, daß Sie Ihnen den Prozeß der Erstellung einer Anwendung mit der Development Workbench schnell nahebringen. Am Beispiel einer ABAP-Transaktion lernen Sie die einzelnen Workbench-Werkzeuge kennen. Bei der Transaktion handelt es sich um eine vereinfachte Flugreservierung. Die einzelnen Lektionen bauen dabei aufeinander auf. Aus diesem Grund sollten Sie das Tutorial in der vorgegebenen Reihenfolge absolvieren. Das Tutorial besteht aus folgenden Lektionen:

- [Lektion 1: Workbench-Konzepte und -Werkzeuge \[Seite 16\]](#)

stellt die Werkzeuge der Workbench vor. Sie lernen, wie Sie die Workbench aufrufen und verlassen. Weiterhin lernen Sie den Umgang mit dem Object Browser. Sie lernen, wie Sie private und allgemeine Objektlisten anzeigen. Schließlich legen Sie ein Programmobjekt und eine Transaktion an.
- [Lektion 2: Mit Tabellen arbeiten \[Seite 27\]](#)

stellt das ABAP Dictionary vor. Sie lernen, wie Sie ein Tabellenobjekt anlegen. Weiterhin erfahren Sie, aus welchen Komponenten sich eine Tabelle zusammensetzt. Außerdem lernen Sie, wie Sie mit den Werkzeugen der Workbench Objekte im R/3-System freigeben.
- [Lektion 3: Bildschirmbilder gestalten \[Seite 40\]](#)

zeigt Ihnen den Umgang mit dem Screen Painter. Sie lernen, Bildelemente zu gestalten. Weiterhin lernen Sie, ein einfaches Bild so zu verfeinern, daß es einer kommerziellen Oberfläche gleicht.
- [Lektion 4: GUI-Status definieren \[Seite 50\]](#)

stellt den Menu Painter zum Definieren von GUI-Status vor. Mit dem Menu Painter definieren Sie die Symbol- und Drucktastenleisten in einem Anwendungsfenster sowie Menüs, Fenstertitel und Funktionstasten.
- [Lektion 5: Transaktionen programmieren \[Seite 58\]](#)

zeigt den letzten Schritt beim Erstellen einer Transaktion: wie Sie im ABAP Editor Ihre Programmtexte erfassen. Sie lernen, globale Variablen und Unterprogramme anzulegen. Im Editor erfassen Sie zudem Nachrichten für ein bestehendes Programm. Sie lernen, wie Sie Ihre Transaktion ausführen und Ihr Programm auf Fehler untersuchen.
- [Lektion 6: Arbeiten im Team \[Seite 69\]](#)

Inhaltsübersicht

zeigt den Umgang mit dem Workbench Organizer. Sie lernen, auf Objekte während der Entwicklung zuzugreifen und Programm-Coding freizugeben.

Nehmen Sie sich für die einzelnen Lektionen so viel Zeit wie nötig. Jede Lektion beginnt mit einem Überblick der Lernziele. Dann folgen mehrere Übungen. Nach jeder Übung haben Sie die Möglichkeit, Ihre Arbeit zu überprüfen. Am Ende einer Lektion gibt es eine kurze Zusammenfassung des behandelten Themas und eine Vorschau auf die daran anschließende Lektion.

Nachdem Sie die Lektionen erfolgreich abgeschlossen haben, sollten Sie in der Dokumentation [ABAP Development Workbench: Werkzeuge \[Extern\]](#) weiterlesen, um Ihr eben gewonnenes Wissen zu vertiefen.

Voraussetzungen

Das *ABAP Workbench Tutorial* wendet sich an Leser, die bereits mit dem R/3-System vertraut sind. Die Dokumentation *Einführung in das R/3-System* wird als bekannt vorausgesetzt. R/3-Kenntnisse aus betriebswirtschaftlichen Anwendungen sind ebenso nützlich. Wenn auch keine Detailkenntnisse der Programmiersprache ABAP erwartet werden, so sollten Sie dennoch Grundkenntnisse haben.

Um mit dem Tutorial arbeiten zu können, müssen Sie

- sich am R/3-System anmelden können.
Sofern Sie dieses Tutorial online lesen, sind Sie bereits angemeldet. Falls Sie sich am System nicht anmelden können, wenden Sie sich an ihren Systemverwalter, der Ihnen einen Benutzernamen, einen Mandanten und ein Kennwort nennen kann. Die Dokumentation *Einführung in das R/3-System* erklärt ausführlich, wie Sie sich am System anmelden.
- Release 3.0 des R/3-Systems haben.
Sollten Sie nicht wissen, welche Version Sie haben, wenden Sie sich an Ihren Systemverwalter.
- auf einer Unix-, Windows 95- oder Windows NT-Plattform arbeiten können.
Der graphische Fullscreen-Editor von SAP läuft nur auf Unix-, Windows 95- oder Windows NT-Plattformen. Auf anderen Plattformen müssen Sie den alphanumerischen Fullscreen-Editor verwenden.
- auf die Entwicklungsklasse SDW6 zugreifen können.
Wenden Sie sich an Ihren Systemverwalter, wenn Sie sich nicht sicher sind, ob diese Entwicklungsklasse in Ihrem System existiert.

Terminologie

Terminologie

Die vorliegende Dokumentation geht davon aus, daß Die´Sie mit dem Umgang mit der R/3-Oberfläche vertraut sind. Falls nicht, sollten Sie sich zunächst mit den Funktionen des R/3-Fensters in der Dokumentation *Einführung in das R/3-System* befassen. Dieses Tutorial verwendet die folgende Terminologie:

Begriff	Beschreibung
Dialogfenster (oder Dialog)	Ein Fenster, das das System auf dem Anwendungsfenster anzeigt. Ein Dialogfenster erscheint, wenn das R/3-System zusätzliche Informationen von Ihnen benötigt oder wenn es Ihnen eine Nachricht anzeigt.
eingeben	Eintippen von Informationen in ein Feld auf einem Bild oder in einem Dialogfenster.
Bild (oder Bildschirmbild)	Die Anordnung von Menüs, Tasten und Feldern, die in einem Fenster erscheinen. Eine ABAP-Anwendung kann über mehrere Bilder verfügen. Der Titel eines Bildes erscheint in der Titelleiste.
auswählen	Eine Aktion mit der Maus oder einer Funktionstaste zum Aktivieren einer Option. Wie Sie im R/3-System ein Element wählen, hängt von der Plattform und der Maus ab, mit der Sie arbeiten. Sollten Sie nicht wissen, wie Sie in Ihrer Umgebung Elemente wählen, schlagen Sie in der Einführungsdokumentation Ihres Systems nach.
markieren	Eine Aktion zum Aktivieren eines Auswahlknopfes, eines Listenelements oder eines Textfeldes.
Fenster	Ein grafisches Objekt in Ihrer Anzeige, das eine Anwendung enthält. Im Fall der Tutorial-Lektionen enthält das Fenster die ABAP-Anwendung.

Plattform

Die Bildschirmabgriffe für dieses Tutorial wurden unter Windows NT 3.5 erstellt. Sollten Sie z.B. Motif oder Windows 95 benutzen, sehen die Bilder in Ihrem System eventuell etwas anders aus. Dies hat jedoch keinen Einfluß auf die Vorgehensweise bei den einzelnen Übungen.

Namenskonvention für SAP-Objekte

Namenskonvention für SAP-Objekte

In diesem Tutorial müssen Sie viele neue ABAP-Objekte benennen. Dazu müssen Sie zwei R/3-Namenskonventionen kennen und einhalten.

1. Alle von Ihnen angelegten Objektnamen müssen mit **Y** oder **Z** beginnen. Damit garantieren Sie, daß sich Ihre Namen von den R/3-Systemversionen unterscheiden.
2. Die gewählten Namen müssen eindeutig sein. Das System meldet, wenn ein Name bereits existiert. Viele Entwicklungsobjekte sind systemweit verwendbar. Diese Konvention schützt also vor einem Verlust oder einer ungewollten Veränderung der Daten.

Darüber hinaus ist es eine gute Idee, während der Arbeit mit diesem Tutorial die Initialen Ihres Namens zur Unterscheidung zu verwenden. So vermeiden Sie Konflikte mit anderen Benutzern, die vielleicht gleichzeitig ebenfalls mit dem Tutorial arbeiten.

Wenn als beispielsweise Anna Jonas ein Objekt in der Form **Y<xx>ID** benennen soll, würde sie den Namen **YAJID** wählen. Ihr Objekt unterscheidet sich damit eindeutig von dem des Benutzers Robert Schmitt: **YRSID**.



Eine Nichtbeachtung der Namenskonvention kann zu Datenverlusten führen! Merken Sie sich daher diese beiden Konventionen gut, bevor Sie weiterlesen.

Weiterführende Dokumentation

In den folgenden Dokumentationen erfahren Sie mehr über die ABAP Development Workbench und das Programmieren mit ABAP:

- [ABAP-Benutzerhandbuch \[Extern\]](#). In dieser Dokumentation finden Sie detaillierte Schilderungen der einzelnen ABAP-Programmteile. Sie lernen mehr über
 - ABAP-Grundlagen
 - Reportprogrammierung,
 - Transaktionsentwicklung
 - fortgeschrittene Techniken
- [ABAP Development Workbench: Werkzeuge \[Extern\]](#). In dieser Dokumentation finden Sie detaillierte Ausführungen zu den einzelnen Werkzeugen der Workbench und zum Coding, das Sie mit ihnen produzieren können.
- [ABAP Dictionary \[Extern\]](#). In dieser Dokumentation finden Sie eine Darstellung der ABAP-Daten. Die unterschiedlichen Datenobjekte und ihre Handhabung werden ebenso beschrieben wie zahlreiche Besonderheiten.
- [Workbench Organizer \[Extern\]](#). In dieser Dokumentation lesen Sie, wie Sie große Projekte im R/3-System organisieren können. Einige der Themen sind: Organizer einrichten, das Transportsystem und Versionsverwaltung.
- [Erweiterte Funktionsbibliothek Anwendungen \[Extern\]](#). In dieser Dokumentation finden Sie Informationen zu:
 - standardisierten Dialogen
 - der zentralen Adreßverwaltung
 - Anwendungsprotokolle
 - Archivierung
- [Basis Programmierschnittstellen \[Extern\]](#). In dieser Dokumentation werden die Programmierschnittstellen für SAP-Komponenten beschrieben, u.a. die Hintergrundverarbeitung und das Batch-Input-System.

Lektion 1: Workbench-Konzepte und -Werkzeuge

Lektion 1: Workbench-Konzepte und -Werkzeuge

In dieser Lektion lernen Sie die grundlegenden Konzepte und die Handhabung der ABAP Development Workbench kennen. Die Lektion hat folgenden Inhalt:

[Überblick \[Seite 17\]](#)

[Übung 1: Die Workbench aufrufen \[Seite 18\]](#)

[Übung 2: Die Werkzeuge der Workbench \[Seite 19\]](#)

[Übung 3: In Objektlisten navigieren \[Seite 21\]](#)

[Übung 4: Programme anlegen \[Seite 22\]](#)

[Übung 5: Sprungmarken setzen \[Seite 24\]](#)

[Übung 6: Transaktionen anlegen \[Seite 25\]](#)

[Zusammenfassung \[Seite 26\]](#)

Überblick

Lektion 1 "Workbench-Konzepte und -Werkzeuge" gibt eine Einführung in die ABAP Development Workbench und ihre Werkzeuge. Wenn Sie diese Lektion durchgearbeitet haben,

- verstehen Sie die Workbench-Konzepte
- können Sie die Workbench in einem R/3-Fenster aufrufen
- können Sie die Workbench verlassen
- kennen Sie die Werkzeuge der Workbench und ihre Funktionen
- können Sie in einem Programm navigieren
- können Sie ein Programm anlegen
- können Sie eine Transaktion anlegen

Bevor Sie mit dieser Lektion beginnen, sollten Sie jedoch den Abschnitt [Benutzerhinweise \[Seite 8\]](#) gelesen haben.

Die Workbench-Konzepte


Mit der ABAP Development Workbench schreiben Sie Anwendungsprogramme. Die Workbench ist eine grafische [Programmierungsumgebung \[Extern\]](#). Sie rufen die Programmierwerkzeuge über Drucktasten, Dialogfenster und Fenster des R/3-Systems auf. Im R/3-System heißt ein von einem Entwickler angelegter Programmteil [Entwicklungsobjekt \[Extern\]](#).

Eine ABAP-Anwendung ist entweder eine [Transaktion \[Extern\]](#) oder ein [Report \[Extern\]](#). Die Beispielanwendung, die Sie im Laufe dieser Lektion anlegen sollen, ist eine Transaktion. Eine Transaktion ist eine [Anwendung \[Extern\]](#) für den Endbenutzer. Transaktionen verarbeiten Benutzereingaben und führen dann eine oder mehrere Aktionen durch. Eine Anwendung zum Erfassen von Bestellungen ist beispielsweise eine Transaktion. Reports dagegen sind Anwendungen, für die keine oder nur wenige Benutzerinteraktionen erforderlich sind.

Hinter jeder Transaktion steht ein [Modulpool \[Extern\]](#)-Programm. Dieser Begriff steht für eine Sammlung von ABAP-Sprachkomponenten, die eine Transaktion steuern.

Übung 1: Die Workbench aufrufen

Übung 1: Die Workbench aufrufen

Melden Sie sich am R/3-System an, und wählen Sie anschließend aus der Menüleiste *Werkzeuge* → *ABAP Workbench*. Sie gelangen auf das Bild *ABAP Development Workbench*  [\[Extern\]](#).

Mit der Funktion *Zurück* können Sie jederzeit auf das Einstiegsbild der Workbench zurückkehren. Über *System* → *Abmelden* können Sie das R/3-System jederzeit verlassen. Wenn Sie sich abmelden, fordert Sie das System zur Bestätigung der Aktion auf.

Überprüfen Sie Ihre Arbeit

Probieren Sie die Drucktasten und Menüs der Workbench aus. Sie sollten mit den Drucktasten in der Symbolleiste vertraut sein. Näheres zum Aufbau der Systemoberfläche erfahren Sie in der Dokumentation *Einführung in das R/3-System*.

Während Sie mit dem Tutorial arbeiten, können Sie jederzeit aufhören, Ihre Arbeit sichern und das System verlassen. Sie sollten diese Funktionen sicher beherrschen. Üben Sie, die Workbench von verschiedenen Stellen aus zu verlassen.

Übung 2: Die Werkzeuge der Workbench

In der vorigen Übung haben Sie gelernt, wie Sie die Workbench aufrufen und verlassen. Nun werden Sie mit den einzelnen Werkzeugen der Workbench vertraut gemacht. Rufen Sie die Workbench auf. Auf dem Einstiegsbild der ABAP Development Workbench werden Ihnen in der Drucktastenleiste die sechs Werkzeuge der Workbench angeboten. Jedes dieser Entwicklungswerkzeuge übernimmt bestimmte Aufgaben im Rahmen der ABAP-Programmierung.

Das ABAP Dictionary speichert systemweite Datendefinitionen. Wenn Sie eine neue Datendefinition anlegen, übernimmt das Dictionary alle erforderlichen Arbeiten. Im Dictionary können Sie die "Definition" von Objekten in Ihrem R/3-System nachschlagen.

Im ABAP Editor legen Sie neues Coding an oder ändern bereits vorhandenes Coding. Der ABAP Editor kann überprüfen, ob Sie in Ihrem Programm die richtige Syntax verwendet haben. Ist die Syntax einmal korrekt, können Sie im Editor Ihr Programm generieren und ausführen sowie Fehler suchen und beseitigen.

Die Funktionsbibliothek enthält betriebswirtschaftliche Funktionsbausteine und Programmierhilfen. Wenn Sie einen neuen Funktionsbaustein anlegen, übernimmt die Funktionsbibliothek die hierzu erforderliche Verarbeitung.

Der Screen Painter und der Menu Painter sind Werkzeuge für die einfache und schnelle Gestaltung der grafischen Benutzungsoberfläche (GUI) Ihres Programms. Mit dem Screen Painter erzeugen Sie Bedienelemente, Texteingabefelder und andere Bildelemente. Der Menu Painter hilft Ihnen, die erforderlichen Menüs sowie die Funktions- und Drucktastenbelegung zu erstellen.

Der Object Browser

Der Object Browser ist ein spezielles Werkzeug der ABAP Development Workbench. Er liefert den entsprechenden Kontext für Ihr Programm. Wenn ein Schreiner einen Schrank zusammenbaut, hat er alle Bauteile wie Bretter und Nägel physisch vor sich liegen. Beim Zusammenbauen liefert ihm das Auge den Kontext, in dem die einzelnen Teile zueinander stehen.

Da ein Programm aus einer Vielzahl von Beziehungen zwischen Daten besteht, ist es für einen Programmierer schwer, die Beziehungen zwischen den einzelnen Datenkomponenten auszumachen. Der Object Browser liefert den notwendigen Zusammenhang, um die Beziehungen innerhalb eines Programms zu erfassen.

Mit dem Object Browser können Sie durch eine Liste der Entwicklungsobjekte navigieren. Entwicklungsobjekte sind die einzelnen Bestandteile, aus denen Sie Ihre Anwendung zusammensetzen. Sie können auch einzelne Entwicklungsobjekte anzeigen. In der folgenden Übung lernen Sie, wie Sie mit dem Object Browser durch eine Objektliste navigieren.

Der Object Browser ruft für die von Ihnen eingeleitete Aktion automatisch das erforderliche Werkzeug auf. Wenn Sie zum Beispiel eine neue Datendefinition vom Bild des Object Browsers anlegen, ruft der Browser das ABAP Dictionary auf und kehrt zu Ihrem Ausgangsbild zurück, nachdem Sie die Definition angelegt haben.

Sie können eine ganze Anwendung im Object Browser anlegen, ohne direkt die anderen Werkzeuge der Workbench aufrufen zu müssen. Wir empfehlen ausdrücklich, Anwendungen im Browser anzulegen, denn dort können Sie tatsächlich "sehen", was Sie entwickeln. Für die Lektionen in diesem Lernprogramm arbeiten Sie im Browser.

Übung 2: Die Werkzeuge der Workbench


Wiederholen Sie das Gelernte


In dieser Übung haben Sie ein wenig über das Leistungsvermögen der einzelnen Werkzeuge der Development Workbench erfahren. Nehmen Sie sich ein wenig Zeit, das Gelernte zu rekapitulieren. Ordnen Sie den folgenden Aufgaben das jeweilige Werkzeug zu, mit dem Sie die Aufgabe erledigen können:

- Fehler in einem Reportprogramm suchen und beseitigen
- Eine Datendefinition nachschlagen
- Eine Drucktaste in einem Dialogfenster anordnen
- Eine Liste der Bilder eines Programms ansehen

In dieser Übung wurde auch der Object Browser vorgestellt. Sie haben erfahren, daß SAP empfiehlt, ABAP-Programme im Object Browser zu schreiben.

Übung 3: In Objektlisten navigieren

Eine Objektliste ist eine grafische Anordnung von zusammenhängenden Entwicklungsobjekten. Eine Programmobjektliste enthält z.B. alle Objekte eines Programms. Eine Objektliste ist ähnlich aufgebaut wie das Dateiverzeichnis eines grafischen Datei-Managers  [Extern]. In dieser Übung lernen Sie, wie Sie eine Objektliste aufrufen und darin navigieren. Rufen Sie die Development Workbench auf, und gehen Sie dann so vor:

1. Wählen Sie *Object Browser*, um das Werkzeug zu starten.
Sie gelangen auf das Bild *Object Browser: Einstieg*. Sie müssen nun angeben, ob Sie eine Objektliste oder ein Einzelobjekt ansehen möchten. In dieser Übung sollen Sie sich eine Objektliste ansehen.
2. Markieren Sie im Gruppenrahmen *Objektliste* die Option *Programm*, und stellen Sie den Cursor auf das zugehörige Feld.
Das System zeigt rechts neben dem Feld die Werthilfe-Drucktaste an. Das Erscheinen dieser Taste gibt an, daß hier mehrere Eingabemöglichkeiten bestehen.
3. Tragen Sie im Feld *Programm* den Namen `tutprog` ein.
4. Wählen Sie *Anzeigen*.
Der Browser zeigt die Entwicklungsobjekte des Programms an. Das Programm TUTPROG enthält verschiedene Objektarten. Neben jeder Objektart sehen Sie ein Mappensymbol. Ein Pluszeichen (+) im Symbol bedeutet, daß die Mappe geschlossen ist.
5. Wählen Sie *Globale Daten* durch Doppelklick aus.
Der Browser zeigt eine Liste der globalen Daten an, die im Programm TUTPROG enthalten sind. Das Minuszeichen (-) neben *Globale Daten* zeigt an, daß diese Mappe jetzt geöffnet ist.
6. Wählen Sie *ANSWER* mit Doppelklick aus.
Der Object Browser startet den ABAP Editor. Falls Sie den Editor bislang noch nicht benutzt haben, werden Sie aufgefordert, den PC-Modus zu bestätigen. Wählen Sie dazu *Weiter*. Der Editor zeigt das Programm TUTPROG an der Stelle an, an der das globale Feld *ANSWER* deklariert ist  [Extern].
7. Kehren Sie mit *Zurück* zum Bild *Object Browser: Einstieg* zurück.

Wiederholen Sie das Gelernte

In dieser Übung haben Sie gelernt, wie Sie eine Objektliste für ein bestimmtes Programm aufrufen. Mit dem Object Browser haben Sie sich in der Objektliste für das Programm TUTPROG bewegt. Dieses Programm enthält die Beispieltransaktion, die Sie in Laufe dieses Tutorials erstellen. Nehmen Sie sich einen Moment Zeit, um einige Objekte in der Objektliste des Programms zu öffnen.

Wenn Sie möchten, können Sie das Programm ausführen. Markieren Sie dazu den Knoten TUTPROG und wählen Sie dann *Entwicklungsobjekt* → *Testen/Ausführen*. Wenn das System Sie auffordert, eine Ausführungsart anzugeben, markieren Sie *Direkt*.

Übung 4: Programme anlegen

Übung 4: Programme anlegen

In dieser Übung lernen Sie, wie Sie ein Programm anlegen. Das Programm, das Sie anlegen, wird schließlich eine vollständige Anwendung im R/3-System bilden. Melden Sie sich am R/3-System an, und rufen Sie das Bild *Object Browser: Einstieg* auf. So legen Sie ein Programm an:

1. Markieren Sie *Lokale priv. Objekte*.

Für jeden Benutzer gibt es eine Liste mit lokalen privaten Objekten. Standardmäßig geht der Browser davon aus, daß Sie Ihre eigenen Objektlisten ansehen möchten.

2. Wählen Sie *Anzeigen*.

Das System zeigt nun Ihre Objektliste an. Zeigen Sie die Liste zum ersten Mal an, sehen Sie nur den Knoten *Objektarten Entwicklungsklasse*. Dieser Knoten ist interaktiv.

3. Wählen Sie *Objektarten Entwicklungsklasse* mit Doppelklick aus.

Das System fordert Sie auf, die gewünschte Objektart anzugeben.

4. Markieren Sie *Programmobjekte*, und wählen Sie dann *Weiter*.

Das System zeigt eine Liste mit möglichen Programmobjektarten an.

5. Wählen Sie einen Namen für Ihr Programm.

Beachten Sie dabei die gültige Namenskonvention. Das empfohlene Format für Ihren Programmnamen ist **SAPMZ<bbb>**, wobei **<bbb>** für Ihre Initialen steht.

6. Tragen Sie den Programmnamen im entsprechenden Feld ein [\[Extern\]](#).

7. Vergewissern Sie sich, daß *Programm* markiert ist, und wählen Sie dann *Anlegen*.

Das System fordert Sie auf, Ihre Auswahl zu bestätigen.

8. Vergewissern Sie sich, daß der Programmname korrekt ist und die Option *Mit TOP-Include* markiert ist.

9. Wählen Sie *Weiter*.

Das System fordert Sie auf, den Namen des TOP-Include-Programms anzugeben. Die Namenskonvention für Includes ist **MZ<bbb>TOP**, wobei **<bbb>** für Ihre Initialen steht.

10. Tragen Sie den Namen für das TOP-Include-Programm ein, und wählen Sie dann zur Bestätigung *Weiter*.

Das System zeigt das Bild der Programmattribute an.

11. Tragen Sie Attributwerte für Ihr Programm folgendermaßen ein:

<i>Titel</i>	Modulpool Flug buchen
<i>Typ</i>	M
<i>Anwendung</i>	*


12. Sichern Sie Ihre Programmattribute.

Das System legt Ihr Programm und das Include an.

Übung 4: Programme anlegen

Überprüfen Sie Ihre Arbeit

In dieser Übung haben Sie ein neues Programm und ein Include angelegt. Sie werden mit diesen Objekten später noch weiterarbeiten. Nehmen Sie sich nun Zeit, Ihre Arbeit zu prüfen. Zeigen Sie im Object Browser Ihre lokalen privaten Objekte an. Falls Sie sich noch auf dem Bild der Programmattribute befinden, können Sie auch mit der Funktion *Zurück* in den Object Browser verzweigen.

Wählen Sie im Object Browser *Programme* und *Includes*. Ihre Objektliste sollte nun das von Ihnen angelegte Programm und Include enthalten  [\[Extern\]](#).

Übung 5: Sprungmarken setzen

Übung 5: Sprungmarken setzen

In dieser Übung lernen Sie, wie Sie Sprungmarken setzen. Eine Sprungmarke kann man mit einem Lesezeichen vergleichen. Über Sprungmarken können Sie gezielt auf eine bestimmte Liste im Browser zugreifen. So setzen Sie Sprungmarken:

1. Wählen Sie *Springen* → *Sprungmarken*.

Das System zeigt eine List mit Sprungmarken an. Die Einstiegsmarke definiert das erste Bild, das das System anzeigt, wenn Sie den Object Browser starten. Standardmäßig ist dies das Einstiegsbild der Workbench. Sie sollen nun Ihre private Objektliste als Ihren Einstieg setzen.

2. Markieren Sie *Klasse \$TMP*.
3. Wählen Sie *Neue Einstiegsmarke*.

Klasse \$TMP ist nun der erste Eintrag in der Liste.

4. Wählen Sie `ENTER`.

Überprüfen Sie Ihre Arbeit

Mit der Funktion *Zurück* gelangen Sie auf das Einstiegsbild der ABAP Development Workbench. Rufen Sie nun den Object Browser auf. Das System springt direkt zu Ihrer privaten Objektliste, ohne zuerst das Einstiegsbild anzuzeigen.

Sie können aus der Menüleiste des Object Browser *Springen* → *Einstiegsbild* wählen, um zum Einstiegsbild zurückzukehren.

Übung 6: Transaktionen anlegen

In dieser Übung lernen Sie, wie Sie eine Transaktion anlegen. Eine Transaktion ist ein [Entwicklungsobjekt \[Extern\]](#). Gehen Sie auf das Bild mit Ihrer privaten Objektliste. So legen Sie eine Transaktion an:

1. Öffnen Sie die Mappe *Programm* und wählen Sie Ihr neues Programm mit Doppelklick aus.

Das System zeigt die Objektliste des Programms an. Die Liste enthält die zu dem Programm gehörenden Objekte. Die Liste ist zu diesem Zeitpunkt leer.

2. Wählen Sie *Objektarten Programm* mit Doppelklick aus.

Das System fordert Sie auf, eine Objektart anzugeben.

3. Markieren Sie *Transaktion*.

Sie müssen nun einen Transaktionsnamen eingeben. Das entsprechende Format ist **ZF<xx>**, wobei <xx> für Ihre Initialen steht. Halten Sie die [SAP-Namenskonvention \[Seite 14\]](#) ein.

4. Tragen Sie den Transaktionsnamen im entsprechenden Feld ein, und wählen Sie anschließend *Anlegen*.

Das System fordert Sie auf, einen Transaktionstyp anzugeben.

5. Markieren Sie *Dialogtransaktion*, und wählen Sie anschließend *Weiter*.

Das System fordert Sie auf, weitere Angaben zu machen.

6. Geben Sie folgende Daten ein:

<i>Transaktionstext</i>	Flugdaten erfassen
<i>Programm</i>	Der Name des Programms, das Sie in Übung 4 angelegt haben.
<i>Dynpronummer</i>	100

7. Wählen Sie *Sichern*.

Das System legt die neue Transaktion an

Überprüfen Sie Ihre Arbeit

In Ihrer Programmobjektliste sollte nun der Eintrag *Transaktionen* erscheinen. Rufen Sie die neue Transaktion auf, und lesen Sie die für die Transaktion angezeigten Informationen.

Zusammenfassung

Zusammenfassung

In der Lektion 1 wurden Sie mit der ABAP Development Workbench und ihren Werkzeugen bekannt gemacht. Sie haben gelernt, daß die Workbench eine Programmierumgebung für die ABAP-Programmiersprache ist. Sie haben weiterhin gelernt, die Workbench im System aufzurufen und zu verlassen.

Sie sollten die Namen der sechs wichtigsten Werkzeuge der Workbench kennen:

- Object Browser
- ABAP Dictionary
- ABAP Editor
- Funktionsbibliothek
- Screen Painter
- enu Painter

In dieser Lektion nahm der Object Browser großen Raum ein. Sie haben gelernt, daß dieses Werkzeug hilft, Ihre Programmumgebung zu überblicken. Mit dem Browser haben Sie die verschiedenen Teile eines bereits vorhandenen Programms angezeigt und ein neues Programm angelegt. Sie haben zudem gelernt, wie Sie Einstiegsmarken setzen.

In der nächsten Lektion...

In Lektion 2 lernen Sie, wie Sie eine Tabelle anlegen und sie lernen die Konzepte und Einzelheiten zu einigen grundlegenden Bestandteilen des ABAP Dictionary.

Lektion 2: Mit Tabellen arbeiten

In dieser Lektion lernen Sie, wie Sie eine Tabelle anlegen. Die Lektion hat folgenden Inhalt:

[Überblick \[Seite 28\]](#)

[Übung 1: Tabellenobjekte anlegen \[Seite 30\]](#)

[Übung 2: Tabellenfelder angeben \[Seite 31\]](#)

[Übung 3: Datenelemente und Domänen definieren \[Seite 32\]](#)

[Übung 4: Domänen wiederverwenden \[Seite 34\]](#)

[Übung 5: Festwerte definieren \[Seite 36\]](#)

[Übung 6: Technische Einstellungen vornehmen \[Seite 37\]](#)

[Übung 7: Tabellen aktivieren \[Seite 38\]](#)

[Zusammenfassung \[Seite 39\]](#)

Überblick

Überblick

Tabellen spielen eine Schlüsselrolle im R/3-System. Tabellen definieren die Struktur einer Datenbank. Bevor Sie selbst Tabellen gestalten und anlegen können, müssen Sie die Funktionsweise von Tabellen kennen. Wenn Sie diese Lektion durchgearbeitet haben,

- können Sie eine Tabelle anlegen
- kennen Sie die wichtigsten Bestandteile einer Tabelle
- kennen Sie das Werkzeug, mit dem Sie Tabellen anlegen
- können Sie eine Tabelle im Object Browser anlegen
- können Sie eine Tabelle aktivieren
- können Sie ein Feld einer Tabelle hinzufügen
- können Sie Festwerte definieren
- können Sie angeben, wie das System eine Tabelle verarbeiten soll

Am Ende dieser Lektion definiert Ihre Tabelle die für Ihre Transaktion wichtigen Daten, z.B. Abflugzeit, Flugnummer und Ankunftszeit.

Bestandteile einer Tabelle

Jede Tabelle im R/3-System besteht aus verschiedenen Komponenten:

Tabellenobjekt	repräsentiert eine Tabelle im ABAP Dictionary.
Felder	definieren die in einer Tabelle gespeicherten Informationen.
Datenelementobjekte	beschreiben Feldinhalte und bestimmen, wie ein Feld dem Endbenutzer angezeigt werden soll. Datenelemente erscheinen als eigenständige Objekte im ABAP Dictionary. Sie können sie daher innerhalb der gleichen Tabelle mehrmals oder in Feldern in verschiedenen Tabellen gleichzeitig einsetzen.
Domänenobjekte	beschreiben mögliche Feldwerte. Eine Domäne gibt Informationen wie z.B. den Datentyp oder die Anzahl der Stellen in einem Feld an. Domänen erscheinen wie Datenelemente als Objekte im ABAP Dictionary, und Sie können Domänen genauso wiederverwenden.
Technische Einstellungen	geben an, wie das R/3-System eine Tabelle bearbeitet.

Zusammenfassung der Tabellengestaltung

Vor dem Anlegen einer Tabelle machen Sie sich in der Regel einen Plan und überlegen, welche Daten Sie benötigen. Dazu müssen Sie wissen, wie die Daten in Beziehung stehen. In dieser Lektion wurde die vorbereitende Planung für Sie bereits erledigt. Wenn Ihr Plan einmal steht, legen Sie eine Tabelle so an:

- Ein Tabellenobjekt anlegen
- Felder angeben

- Datenelemente und Domänen definieren
- Technische Einstellungen vornehmen
- Die Tabelle aktivieren

Beim Anlegen der Tabelle sollten Sie Ihre Arbeit sichern. Beim Sichern wird ein Objekt in der SAP-Datenbank abgelegt. Das Statusattribut der Tabelle wird auf gesichert gesetzt. Andere Benutzer können gesicherte Objekte ansehen, jedoch nicht in ABAP-Programmen ansprechen.

Nach der vollständigen Erstellung des Tabellenobjekts aktivieren Sie es in der Datenbank mit der Funktion *Aktivieren*. Wenn Sie eine Tabelle aktivieren, führt das System folgende Aufgaben durch:


- Es prüft die Syntax.
- Es aktualisiert den Tabellenstatus und setzt ihn auf *Aktiv*.
- Es kompiliert ein Laufzeitobjekt der Tabelle.

Die aktive Tabelle kann nun in anderen Programmen und von anderen Benutzern verwendet werden.

Übung 1: Tabellenobjekte anlegen

Übung 1: Tabellenobjekte anlegen

Zeigen Sie mit dem Object Browser Ihr Tabellenobjekt an, und rufen Sie Ihre private Objektliste auf. So legen Sie dann eine Tabelle an:

1. Wählen Sie *Objektarten Entwicklungsklasse* mit Doppelklick aus.
Das System fordert Sie in einem Dialogfenster auf, eine Objektart anzugeben.
2. Wählen Sie *Dictionary-Objekte* mit Doppelklick aus, und wählen Sie dann *Weiter*.
Das System fordert Sie auf, die gewünschte Objektart anzugeben.
Das System verarbeitet Ihre Eingaben jetzt über das ABAP Dictionary. Sie mußten dazu den Object Browser nicht verlassen. Das System hat das Dictionary automatisch gestartet.
3. Markieren Sie *Tabelle*, und tragen Sie einen Tabellennamen in das entsprechende Feld ein.
Da Dictionary-Objekte im R/3-System globale Objekte sind, müssen sie eindeutige Namen haben. Folgen Sie bei der Namensgebung der Konvention **Z<xx>FL**, wobei **<xx>** für Ihre Initialen steht  [\[Extern\]](#).
4. Wählen Sie *Anlegen*.
Das System fordert Sie auf, zusätzliche Informationen anzugeben.
5. Tragen Sie folgende Werte ein:

Kurzbeschreibung	Flugtabelle
Auslieferklasse	A
6. Markieren Sie *Tab.pflege erlaubt*.
Sichern Sie Ihre Tabelle.
Das System legt ein Tabellenobjekt an und fügt es zu Ihrer lokalen Objektliste hinzu.

Überprüfen Sie Ihre Arbeit

Nehmen Sie sich einen Moment Zeit, Ihre Arbeit zu überprüfen. Mit *Zurück* kommen Sie wieder zu Ihrer lokalen Objektliste zurück, wo nun zusätzlich die Mappe *Dictionary-Objekte* angezeigt werden sollte. Zeigen Sie die ABAP-Dictionary-Objekte an, um Ihr neues Tabellenobjekt anzusehen.

Übung 2: Tabellenfelder angeben

In Übung 1 haben Sie ein leeres Tabellenobjekt angelegt. In dieser Übung sollen Sie nun Tabellenfelder hinzufügen. Die Felder einer Tabelle definieren die Informationen, die in einer Tabelle gespeichert sind. Normalerweise hat eine Tabelle ein oder mehrere Felder. Sie können eine Tabelle ohne Felder anlegen, jedoch würde das im R/3-System wenig Sinn ergeben.

Um ein Feld hinzuzufügen, zeigen Sie Ihre private Objektliste an. Gehen Sie dann so vor:


1. Markieren Sie Ihr neues Tabellenobjekt.
2. Wählen Sie *Ändern*.

Das System zeigt das Bild *Tabelle/Struktur: Felder ändern* an. Sie legen Felder an, indem Sie Informationen im Gruppenrahmen *Feldattribute* angeben.

3. Tragen Sie **FLID** als Namen für das erste Feld ein.

Das Feld FLID ist das Schlüsselfeld für die Tabelle.

4. Markieren Sie das Ankreuzfeld *Key*.

Das System stellt das Ankreuzfeld markiert dar  [\[Extern\]](#).

5. Sichern Sie Ihre Änderungen.


Das System entfernt leere Felder und sichert Ihre Tabelle.

Restliche Felder definieren

Sie haben nun ein einzelnes Feld in Ihrem Tabellenobjekt. Das Feld FLID ist der **Tabellenschlüssel**. Nehmen Sie sich einen Moment Zeit, um die restlichen Felder in Ihrer Tabelle anzulegen. Für diese müssen Sie nicht das Ankreuzfeld *Key* markieren. Mit *Bearbeiten* → *Neue Felder* zeigen Sie einige leere Felder an. Legen Sie nun die folgenden Felder an:

LVCITY
LVDATE
LVTIME
REGLR
CHRTR
MOVIE
SNACK
FMEAL
ARCITY
ARDATE
ARTIME

Überprüfen Sie Ihre Arbeit

Sie haben nun ein Tabellenobjekt und die Felder der Tabelle angelegt  [\[Extern\]](#). Sie können Ihre Tabelle vom Dictionary aus ansehen. Um auf das Dictionary von einem Bild des Object Browser zuzugreifen, wählen Sie *Umfeld* → *ABAP Dictionary*.

Übung 3: Datenelemente und Domänen definieren

Übung 3: Datenelemente und Domänen definieren

In der vorigen Übung haben Sie die in Ihrer Tabelle benötigten Felder angelegt. In dieser Übung lernen Sie, wie Sie Datenelemente und Domänen definieren. Datenelemente beschreiben den Inhalt eines Feldes und bestimmen, wie ein Feld auf einem Bild erscheint. Domänen beschreiben mögliche Eingabewerte für ein Feld.

Bevor Sie mit dieser Übung beginnen, zeigen Sie Ihre Tabelle in der Workbench an. Falls Sie die Tabelle noch nicht angezeigt haben, navigieren Sie mit dem Object Browser zu Ihrem Tabellenobjekt, und zeigen Sie es an. Definieren Sie dann ein Datenelement und eine Domäne für ein Feld:

1. Stellen Sie für Ihre Tabelle den Änderungsmodus ein.

Falls der Änderungsmodus noch nicht eingeschaltet ist, wählen Sie *Anzeigen <-> Ändern*.

2. Wählen Sie einen Datenelementnamen für das Feld **FLID**.

Da Datenelemente Objekte sind, müssen sie eindeutige Namen haben. Wählen Sie einen Namen der Konvention **Z<xx>_FLID**, wobei **<xx>** für Ihre Initialen steht.

3. Tragen Sie den gewählten Namen für das Feld **FLID** in der Spalte *Datenelem.* ein.

4. Wählen Sie dann *Datenelem.* mit Doppelklick aus.


Bestätigen Sie im folgenden Dialogfenster, daß Sie das Datenelement anlegen möchten.

5. Wählen Sie *Weiter*.

Das System zeigt das Bild *ABAP-Dictionary: Datenelement ändern an*.

6. Tragen Sie folgende Daten ein:

<i>Kurzbeschreibung</i>	Flug-Nr.
<i>Domänenname</i>	Z<xx>_FLID

Sie können den Datenelementnamen als Domänennamen angeben  [\[Extern\]](#).

7. Geben Sie im Gruppenrahmen *Texte*  [\[Extern\]](#) folgendes ein:

<i>Feldbezeich. kurz</i>	9	Flug-Nr.
<i>mittel</i>	13	Flug-Kenn-Nr.
<i>lang</i>	16	Flug-Kennnummer

Das System verwendet die im Gruppenrahmen *Texte* gemachten Angaben später als Feldbezeichner auf Ihrer Oberfläche.

8. Sichern Sie das Datenelement.

9. Wählen Sie das Feld *Domänenname* mit Doppelklick aus.

Das System bestätigt Ihre Auswahl.

10. Wählen Sie *Weiter*.

Das System zeigt das Bild *ABAP-Dictionary: Domäne ändern an*.

11. Tragen Sie folgende Daten ein:

Übung 3: Datenelemente und Domänen definieren

<i>Kurzbeschreibung</i>	Flug-Nr .
<i>Datentyp</i>	Char
<i>Feldlänge</i>	5

12. Wählen Sie *Aktivieren*.

Das System sichert und aktiviert das Domänenobjekt. Dadurch wird das neue Domänenobjekt in der SAP-Datenbank bekannt gemacht. Der Status der Domäne sollte *aktiv* sein.

13. Gehen Sie zum Bild *ABAP-Dictionary: Datenelement ändern* zurück.

14. Wählen Sie *Aktivieren*, um das neue Datenelement der SAP-Datenbank hinzuzufügen.

15. Gehen Sie zum Bild *ABAP-Dictionary: Tabelle/Struktur: Felder ändern* zurück, und sichern Sie Ihre Arbeit.

Überprüfen Sie Ihre Arbeit

Sie haben zwei neue Objekte definiert: ein Datenelement- und ein Domänenobjekt. Diese Objekte erscheinen nun in Ihrer lokalen privaten Objektliste. Zeigen Sie Ihre lokale private Objektliste an. Der Eintrag *Dictionary-Objekte* enthält nun zwei neue Mappen: *Datenelemente* und *Domänen*. Zeigen Sie Ihr neues Datenelement und die Domäne an.

Übung 4: Domänen wiederverwenden

Übung 4: Domänen wiederverwenden

In der vorigen Übung haben Sie ein Datenelement und eine Domäne für ein einzelnes Feld in einer Tabelle definiert. Auf dem Bild *ABAP-Dictionary: Tabelle/Struktur: Felder ändern* können Sie die vollständige Information für das Feld `Z<xx>_FLID` sehen. In dieser Übung sollen Sie nun die Datenelemente und Domänen für die restlichen Felder in Ihrer Tabelle definieren.

In dieser Übung lernen Sie weiterhin, wie Sie vorhandene Domänen Definitionen wiederverwenden können. Ein Feld bezieht sich auf ein einzelnes Datenelement und eine einzelne Domäne. Sie können Datenelemente und Domänen für Felder derselben Tabelle oder für Felder mehrerer Tabellen wiederverwenden. In dieser Übung lernen Sie, wie Sie Domänen und Datenelemente für Felder derselben Tabelle einsetzen.

Zusätzliche Datenelemente und Domänen angeben

Zeigen Sie das Bild *ABAP-Dictionary: Tabelle/Struktur: Felder ändern* an. Legen Sie dann, wie in der Übung 3 beschrieben, Datenelemente und Domänen für die folgenden Felder an:

Feld	Datenelem.	Kurzbeschreibung	Domäne	Datentyp	Feldlänge
LVCITY	Z<xx>_LVCITY	Abf. Ort	Z<xx>_CITY	CHAR	5
LVDATE	Z<xx>_LVDATE	Abf. Datum	Z<xx>_DATE	DATS	8
LVTIME	Z<xx>_LVTIME	Abf. Zeit	Z<xx>_TIME	TIMS	6



Zu den Datenelementen: In dieser Übung werden die Werte für den Gruppenrahmen *Texte* nicht vorgegeben. Tragen Sie Werte ein, die Ihnen sinnvoll erscheinen.

Domänen für restliche Datenelemente wiederverwenden

Spezifizieren Sie nun mit den drei neuen von Ihnen angelegten Domänen (`Z<xx>_CITY`, `Z<xx>_DATE`, `Z<xx>_TIME`) und der vom System definierten Domäne `CHAR1` die restlichen Datenelemente und Domänen. So definieren Sie die restlichen Datenelemente:

1. Stellen Sie für Ihre Tabelle den Änderungsmodus ein.
2. Tragen Sie einen Datenelementnamen im Feld *Datenelem.* ein.

Da Datenelemente Objekte sind, müssen sie eindeutige Namen haben. Wählen Sie einen Namen der Konvention `Z<xx>_REGL`, wobei `<xx>` für Ihre Initialen steht.

3. Wählen Sie *Datenelem.* mit Doppelklick aus.

Das System bestätigt, daß Sie ein Datenelement erstellen wollen.

4. Wählen Sie *Weiter.*

Das System zeigt das Bild *ABAP-Dictionary: Datenelement ändern* an.


5. Geben Sie folgende Daten ein:

Kurzbeschreibung **Linienflug**

Übung 4: Domänen wiederverwenden

Domänenname CHAR1

CHAR1 ist eine vom System definierte Domäne.

6. Füllen Sie die Felder im Gruppenrahmen *Texte*  [Extern] aus.
7. Sichern Sie das Datenelement.
8. Wählen Sie *Aktivieren*, um das neue Datenelement im Dictionary hinzuzufügen.
9. Gehen Sie zum Bild *ABAP-Dictionary: Tabelle/Struktur: Felder ändern* zurück, und sichern Sie Ihre Arbeit.
10. Wiederholen Sie für die restlichen Felder die Schritte 2 bis 9, und verwenden Sie dabei folgende Informationen:

Feld	Datenelement	Kurzbeschreibung	Domainenname
CHRTR	Z<xx>_CHRTR	Charterflug	CHAR1
MOVIE	Z<xx>_MOVIE	Film	CHAR1
SNACK	Z<xx>_SNACK	Imbiß	CHAR1
FMEAL	Z<xx>_FMEAL	Essen	CHAR1
ARCITY	Z<xx>_ARCITY	Ankunftsort	Z<xx>_CITY
ARDATE	Z<xx>_ARDATE	Ankunftsdatum	Z<xx>_DATE
ARTIME	Z<xx>_ARTIME	Ankunftszeit	Z<xx>_TIME

Vergessen Sie nicht, für jedes Datenelement den Gruppenrahmen *Texte* auszufüllen.

Überprüfen Sie Ihre Arbeit

In dieser Übung haben Sie gelernt, eine vorhandene Domäne wiederzuverwenden. Sie wissen nun weiterhin, daß es auch vom System definierte Domänen gibt und daß Sie diese verwenden können. Gehen Sie mit dem Object Browser zu Ihrer lokalen privaten Objektliste, und zeigen Sie die von Ihnen angelegten Objekte an.

Vergleichen Sie in Ihrer Liste die Anzahl der neuen Datenelemente mit der Anzahl der Domänen. Es gibt mehr Datenelemente als Domänen. Beachten Sie, daß jede Domäne nur einmal in der Liste erscheint, obwohl sie von mehreren Feldern verwendet wird. Sie können weiterhin sehen, daß die vom System definierten Domänen (beispielsweise CHAR1) nicht in Ihrer Domänenliste aufgeführt sind, da diese eben systemweit definiert sind und nicht nur lokal.

Übung 5: Festwerte definieren

Übung 5: Festwerte definieren

In der vorigen Übung haben Sie Domänen für die Felder Ihrer Tabelle definiert. Eine Domäne beschreibt die möglichen Eingabewerte für ein Feld. Oft können Sie die zulässigen Werte einschränken und auf einen bestimmten Bestand von [Festwerten \[Extern\]](#) reduzieren.

In dieser Übung lernen Sie, wie Sie Festwerte für die Domäne `Z<xx>_CITY` festlegen. Gehen Sie zu Ihrer privaten Objektliste. So legen Sie dann Festwerte für die Domäne `Z<xx>_CITY` an:

1. Wählen Sie *Domäne* mit Doppelklick aus.
2. Markieren Sie die Domäne für die Stadt, und wählen Sie *Ändern*.

Das Bild *Dictionary: Domäne pflegen* erscheint.

3. Wählen Sie *Festwerte*.
4. Tragen Sie als Untergrenze `MUC` ein.
5. Tragen Sie `Munich` als *Kurzbeschreibung* ein.
6. Machen Sie dann noch folgende Angaben:

Wert	Kurzbeschreibung
TXL	Berlin, Tegel
THF	Berlin, Tempelhof
DEN	Denver
EWR	Newark, New Jersey
JFK	New York
CDG	Paris
YYZ	Toronto

7. Sichern Sie die Liste.
8. Kehren Sie zum Bild *ABAP-Dictionary: Domäne: Festwerte ändern* zurück.
9. Wählen Sie *Aktivieren*, um Ihre Eingaben zu sichern und die SAP-Datenbank zu aktualisieren.

Überprüfen Sie Ihre Arbeit

Verwenden Sie den Object Browser, um Ihre Domänen ansehen. Zeigen Sie die Domäne `Z<xx>_CITY` an, und prüfen Sie die Festwerte, die Sie angelegt haben.

Übung 6: Technische Einstellungen vornehmen

Übung 6: Technische Einstellungen vornehmen

In dieser Übung lernen Sie, wie Sie technische Einstellungen für Ihre Tabelle pflegen. Technische Einstellungen haben Einfluß darauf, wie das System die Tabelle handhabt. Gehen Sie zum Bild *Tabelle/Struktur*. So pflegen Sie im Änderungsmodus technische Einstellungen:

1. Wählen Sie *Technische Einstellungen*.
2. Schalten Sie in den Änderungsmodus.
3. Tragen Sie **APPL1** im Feld *Datenart* ein.

Der Wert **APPL1** zeigt an, daß die Tabelle häufig aktualisiert wird.

4. Tragen Sie **1** im Feld *Größenkategorie* ein.

Der Wert **1** zeigt an, daß es sich um eine kleine Tabelle handelt.

5. Sichern Sie Ihre Einstellungen.
6. Gehen Sie zum Bild *Tabelle/Struktur: Felder ändern* zurück.
7. Sichern Sie Ihre Tabelle.

Überprüfen Sie Ihre Arbeit

Sie haben technische Einstellungen für Ihre Tabelle gepflegt. Auf dem Bild *Technische Einstellungen* können Sie Ihre Arbeit überprüfen.

Übung 7: Tabellen aktivieren


Übung 7: Tabellen aktivieren

In den vorangegangenen Übungen haben Sie ein vollständiges Tabellenobjekt angelegt. In dieser Übung lernen Sie nun, wie Sie eine Tabelle aktivieren. Nachdem eine Tabelle aktiviert ist, können andere Programme auf diese Tabelle verweisen. So aktivieren Sie eine Tabelle:

1. Rufen Sie Ihr Tabellenobjekt im Änderungsmodus auf.
2. Wählen Sie *Aktivieren*.

Das System kompiliert den darunterliegenden Tabellencode, erstellt ein Laufzeitobjekt der Tabelle und setzt den Tabellenstatus auf *Aktiv*.

Überprüfen Sie Ihre Arbeit

Nach der Kompilierung zeigt das System den Status *Aktiv* an  [\[Extern\]](#).

Zusammenfassung

In Lektion 2 haben Sie eine neue Tabelle angelegt. Diese Tabelle definiert die Daten, die in Ihrer Anwendung verwendet werden, z.B. Abflugszeit, Flugnummer und Ankunftsdatum. Sie haben gelernt, daß eine Tabelle aus fünf Teilen besteht:

- einem Tabellenobjekt
- Feldern
- Datenelementobjekten
- Domänenobjekten
- technischen Einstellungen

Ohne diese fünf Bestandteile können Sie keine Tabelle zur SAP-Datenbank hinzufügen. Sie haben zudem gelernt, daß Dictionary-Objekte wie Tabellen, Datenelemente und Domänen systemweite Objekte sind.

Im Laufe dieser Lektion haben Sie Schritt für Schritt gelernt, wie Sie eine Tabelle anlegen. Die einzelnen Schritte sind:

- Ein Tabellenobjekt anlegen
- Felder anlegen
- Datenelemente und Domänen definieren
- Technische Einstellungen vornehmen
- Die Tabelle aktivieren

Dabei haben Sie auch gelernt, daß Sie vorhandene Datenelemente und Domänen wiederverwenden können. Sie können Datenelemente und Domänen, die Sie für eine Tabelle angelegt haben, für mehrere Tabellen verwenden.

Weitere Hintergrundinformationen finden Sie in.

In der nächsten Lektion...

In Lektion 3 lernen Sie, wie Sie mit dem Screen Painter arbeiten. Mit dem Screen Painter gestalten Sie die Bildschirmbilder Ihrer Anwendung. Zusätzlich lernen Sie in Lektion 3, wie Sie Elemente und Drucktasten positionieren. Schließlich lernen Sie, wie Sie Feldattribute für Bilder angeben.

Lektion 3: Bildschirmbilder gestalten**Lektion 3: Bildschirmbilder gestalten**

In dieser Lektion üben Sie die Arbeit mit dem Screen Painter. Die Lektion hat folgenden Inhalt:

[Überblick \[Seite 41\]](#)

[Übung 1: Bilder anlegen \[Seite 42\]](#)

[Übung 2: Elemente auf einem Bild anordnen \[Seite 43\]](#)

[Übung 3: Bilder überarbeiten \[Seite 45\]](#)

[Übung 4: Das Layout überprüfen \[Seite 47\]](#)

[Übung 5: Das OK-Feld setzen \[Seite 48\]](#)

[Zusammenfassung \[Seite 49\]](#)

Überblick

In den vorangegangenen Lektionen haben Sie eine Tabelle angelegt, die die Daten Ihrer Transaktion beschreibt. In dieser Lektion sollen Sie ein Bildschirmbild gestalten, in das der Endbenutzer Daten eingeben kann. Ein Bildschirmbild ist eine Anordnung von grafischen Elementen, die in einem [Fenster \[Extern\]](#) angezeigt werden. Wenn Sie diese Lektion durchgearbeitet haben,

- kennen Sie die für ein ABAP-Bild relevanten Konzepte.
- können Sie ein Einstiegsbild anlegen.
- können Sie die entsprechenden Bildelemente anordnen.
- können Sie Bildelemente charakterisieren.
- einen Prototyp eines Bildes starten.

Die grafische Benutzungsoberfläche (GUI) einer ABAP-Anwendung besteht aus Bildern und Menüs. In der Lektion 4 lernen Sie, wie Sie Menüs anlegen. Um mit dieser Lektion beginnen zu können, müssen Sie Lektion 2 erfolgreich abgeschlossen haben.

Konzepte

Zur Erstellung eines Bildschirmbildes müssen Sie die Komponenten kennen, aus denen sich die grafischen Bildelemente und das Bild selbst zusammensetzen. Zu Bildelementen gehören beispielsweise Drucktasten, Auswahlknöpfe, Beschriftungen und Rahmen, die durch folgende Angaben definiert werden:

Attribute	Die Attribute definieren das Bildelement und enthalten u.a. eine Beschreibung, den Typ sowie die Position.
Layout	Das Layout bezieht sich auf die Anordnung der Elemente auf dem Bild.
Feldattribute	Feldattribute beschreiben das Element. Sie legen beispielsweise fest, daß ein Feld keine Eingaben in Form von Zahlen akzeptiert.
Ablauflogik	Bei der Ablauflogik handelt es sich um eine Reihe von Anweisungen, die die Beziehung zwischen Bildelementen und der darunterliegenden Anwendung regeln.

Sie erstellen und pflegen alle ABAP-Bildelemente mit dem Screen Painter.

Der Screen Painter verfügt über einen Fullscreen Editor, mit dem Sie die Bilder gestalten. Für die Arbeit mit dem Fullscreen Editor stehen Ihnen zwei Darstellungsarten zur Verfügung: die grafische und die alphanumerische Darstellung. Im grafischen Modus verwenden Sie die Maus und Dialogfenster, um die Elemente zu identifizieren und zu erstellen. Dieser Modus ist jedoch nur auf UNIX/Motif-, Windows 95- und Windows NT-Plattformen verfügbar.

Der alphanumerische Modus steht dagegen auf allen Plattformen zur Verfügung und ist als Standardeinstellung vorgegeben. In dieser Lektion arbeiten wir mit dem grafischen Modus.

Übung 1: Bilder anlegen

Übung 1: Bilder anlegen

In dieser Übung legen Sie ein Bildschirmbild an und bestimmen die zugehörigen Attribute. Zeigen Sie dazu die [Objektliste \[Extern\]](#) für Ihr Programm an, und gehen Sie dann so vor:

1. Wählen Sie *Objektarten Programm* mit Doppelklick aus.

Der Browser zeigt die Liste der vorhandenen Programmobjekte an.

2. Markieren Sie *Dynpro*.

Sie müssen nun eine Nummer für Ihr Bild angeben.

3. Tragen Sie für Ihr Bild die Nummer 100 ein.

4. Wählen Sie *Anlegen*.

Das Bild *Ändern Dynproattribute* erscheint.

5. Tragen Sie dann die folgenden Daten ein.

Kurzbeschreibung	Flugdaten erfassen
Dynprototyp	Normal
Folgedynpro	100

ABAP-Programme können die Bildfolge dynamisch bestimmen. Sie müssen daher kein anderes Bild für das Attribut *Folgedynpro* angeben.

6. Sichern Sie das Bild, und gehen Sie zu Ihrer Objektliste zurück.

Überprüfen Sie Ihre Arbeit

In dieser Übung haben Sie ein Bildschirmbild angelegt. In der Objektliste Ihres Programms sollte nun der Eintrag *Dynpros* erscheinen. In den folgenden Übungen werden Sie Ihr Bild mit grafischen Elementen versorgen.

Übung 2: Elemente auf einem Bild anordnen

Übung 2: Elemente auf einem Bild anordnen

In dieser Übung lernen Sie, wie Sie mit dem Fullscreen Editor des Screen Painters Elemente in Ihr Bildschirmbild einfügen. Öffnen Sie die Objektliste für Ihr Programm, und gehen Sie folgendermaßen vor:

1. Markieren Sie Dynpronummer *100*, und wählen Sie *Ändern*.
Der Browser zeigt das Bild *Ablauflogik Editieren* an.
2. Vergewissern Sie sich, daß der grafische Modus des Fullscreen Editors aktiviert ist.
Wählen Sie dazu *Einstellungen* → *Graf. Fullscreen*.
3. Wählen Sie *Fullscreen*.
Das Fenster *Screen Painter: Ändern* wird geöffnet  [\[Extern\]](#).
4. Wählen Sie *Dict/ProgFelder*.
Der Screen Painter zeigt das Dialogfenster *Dict/Programmfelder* an, auf dem Sie existierende Felder aus einer Tabelle oder Programm auf Ihr Bild kopieren können.
5. Tragen Sie Ihren Tabellennamen im Feld *Tabellen-/Feldname* ein.
6. Wählen Sie *Holen aus Dict*.
Das System zeigt eine Liste der Felder in Ihrer Tabelle an.
7. Markieren Sie das Feld *Flug-Nr*.
8. Setzen Sie *Schablone* und setzen Sie *Schlüsselwort* auf *Mittel*.
9. Wählen Sie *Übernehmen*.
Das Fenster des Screen Painters wird im Vordergrund angezeigt.
10. Plazieren Sie das Tabellenfeld, indem Sie den Cursor im Arbeitsbereich positionieren und die linke Maustaste drücken.
Im Arbeitsbereich wird für das Feld *Flug-Nr* ein Feldbezeichner und ein Eingabefeld angezeigt. Sie können ein bereits plaziertes Element verschieben, indem Sie es auswählen und an die neue Position ziehen  [\[Extern\]](#).
11. Sichern Sie Ihre Arbeit.



Restliche Felder hinzufügen

Probieren Sie nun einige neue Verfahren aus, um die restlichen Elemente in Ihr Bild einzufügen. Folgende Elemente sollten noch vorhanden sein:


- Abflugsort/Ankunftsort
- Abflugdatum/Ankunftsdatum
- Abflugzeit/Ankunftszeit
- Linienflug
- Charterflug
- Imbiß

Übung 2: Elemente auf einem Bild anordnen

- Essen
- Film

Sie können mehrere Elemente gleichzeitig hinzufügen, indem Sie sie im Dialogfenster *Dict/Programmfelder* markieren. Verwenden Sie dieses Verfahren für Abflugort, Abflugsdatum und Abflugszeit. Markieren Sie die Felder als eine Gruppe  [Extern] und fügen Sie die Felder dann mit *Übernehmen* als Gruppe ein  [Extern].

Fügen Sie nun die restlichen Felder in Ihr Bild ein. Verwenden Sie dabei Gruppen, wie Sie es bereits für die Abflugsdaten gelernt haben. Die erste Gruppe umfaßt Ankunftsort, Ankunftsdatum und Ankunftszeit, die zweite Gruppe Linienflug und Charterflug und die dritte Gruppe Essen, Imbiß und Film.

Sie können mehrere Elemente positionieren, indem Sie die gewünschten Elemente mit einem "Gummiband" markieren. Halten Sie dann die linke Maustaste gedrückt, und ziehen Sie die Gruppe an die neue Position  [Extern].

Überprüfen Sie Ihre Arbeit

Sie haben mit dem Screen Painter eine einfache Oberfläche geschaffen. Betrachten Sie die Oberfläche nun aus dem Blickwinkel des Benutzers. Wählen Sie im Fullscreen Editor *Dynpro* → *Testen*. Sie werden aufgefordert, die Fensterkoordinaten anzugeben.


Das neue Bildschirmbild wird im Fenster Ihrer SAP-Anwendung angezeigt. Sie sollten nun mit der neu erstellten Oberfläche experimentieren und beispielsweise Werte in die Felder eingeben. Dabei werden Sie merken, daß die Domäne die Arbeit mit einigen Feldern beeinflusst. Versuchen Sie beispielsweise, die Abflugszeit mit der Wertehilfe-Drucktaste einzugeben.

Übung 3: Bilder überarbeiten

In dieser Übung lernen Sie, wie Sie Ihre Bildschirmbilder überarbeiten, um sie übersichtlicher zu gestalten. Sie sollen zwei bereits vorhandene Textfelder in Auswahlknöpfe umwandeln und bestimmte Bildbereiche definieren, indem Sie sie beschriften oder in Gruppenrahmen stellen. Öffnen Sie im Fullscreen Editor des Screen Painters zunächst das Bild 100.


Felder in Auswahlknöpfe umwandeln

Die Felder *Linienflug* und *Charterflug* Ihres Bildschirmbilds beziehen sich auf zwei unterschiedliche Flugarten; beim Flug kann es sich entweder um einen Linienflug oder um einen Charterflug handeln. In der Benutzungsoberfläche dienen Auswahlknöpfe dazu, genau eine Auswahl unter mehreren Optionen zu treffen, die sich gegenseitig ausschließen. Um die Textfelder in Auswahlknöpfe umzuwandeln, gehen Sie folgendermaßen vor:

1. Markieren Sie die Felder *Linienflug* und *Charterflug* mit einem "Gummiband".
Beide Felder werden im Screen Painter markiert.
2. Wählen Sie *Bearbeiten* → *Umwandeln* → *Auswahlknopf* → *Knopf links*.
Der Screen Painter wandelt die beiden Textfelder in Auswahlknöpfe um. Sie müssen die beiden Auswahlknöpfe nun in einer Gruppe zusammenfassen, damit sie sich gegenseitig ausschließen.
3. Wählen Sie *Bearbeiten* → *Auswahlknopfgruppe* → *Definieren*.
Der Screen Painter faßt die Auswahlknöpfe in einer Gruppe zusammen. An den Seiten der Gruppe erscheinen kleine rautenförmige Griffe  [Extern], mit denen Sie die Form der Gruppe verändern können. Mit dem an der Oberseite der Gruppe angezeigten rechteckigen Griff verschieben Sie die Gruppe.
4. Sichern Sie die Änderungen.

Gruppenrahmen erstellen

Ein Gruppenrahmen bedeutet, daß die umrahmten Elemente inhaltlich zusammengehören. Um die Elemente Ihres Bildes in einen Gruppenrahmen mit der Bezeichnung *Flugdaten* einzuschließen, gehen Sie folgendermaßen vor:

1. Wählen Sie in der Werkzeugpalette das Werkzeug *Box* aus.
2. Stellen Sie den Cursor in den Arbeitsbereich.
An der Form des Cursors erkennen Sie, daß ein Gruppenrahmen gezeichnet wird.
3. Zeichnen Sie einen Gruppenrahmen um die Flugdaten auf Ihrem Bild.
Stellen Sie den Cursor links neben die Gruppe. Halten Sie die linke Maustaste gedrückt, und ziehen Sie den Cursor nach rechts unten  [Extern].
Während Sie den Cursor ziehen, wird ein Rahmen um die Gruppe gezeichnet. Dieser Rahmen bleibt markiert, wenn Sie die linke Maustaste loslassen.
4. Vergewissern Sie sich, daß der Gruppenrahmen markiert ist, und geben Sie dann über die Tastatur **F**l**u**g**d**a**t**e**n** ein.
Das System erstellt einen Titel für den neuen Gruppenrahmen.

Übung 3: Bilder überarbeiten


5. Wiederholen Sie die Schritte 1 bis 3, um einen Gruppenrahmen um die beiden Auswahlknöpfe zu erstellen.
6. Legen Sie als Titel für den neuen Gruppenrahmen **Klassifizierung** fest.
7. Sichern Sie die Änderungen.

Felder in Ankreuzfelder umwandeln

Wandeln Sie nun einige der vorhandenen Felder in Ankreuzfelder um. Mit Ankreuzfeldern können Sie bestimmte Optionen aktivieren oder deaktivieren. Wenn Sie eine Option aktivieren, wird das entsprechende Kästchen mit einer Markierung versehen. So erstellen Sie Ankreuzfelder:

1. Markieren Sie die Felder *Imbiß*, *Essen* und *Film*.
Alle drei Felder werden im Screen Painter markiert.
2. Wählen Sie *Bearbeiten* → *Umwandeln* → *Ankreuzfeld* → *Knopf links*.
Der Screen Painter wandelt die Felder in Ankreuzfelder um. Da Ankreuzfelder unabhängig von den anderen Optionen markiert werden können, brauchen Sie sie nicht in Gruppen zusammenzufassen.
3. Sichern Sie die Änderungen.

Überprüfen Sie Ihre Arbeit

Nehmen Sie sich einen Moment Zeit, um Ihre Arbeit zu überprüfen und das neu gestaltete Dialogfenster zu testen. Im Testmodus wird Ihr Bild so angezeigt wie ihn die Benutzer später sehen  [\[Extern\]](#). Setzen Sie *Umfang der Simulation* diesmal auf *Vollständige Ablauflogik*.

Beachten Sie, daß bei der Verwendung von Auswahlknöpfen eine bestimmte Anwendungslogik zum Tragen kommt. Es kann stets nur einer der gruppierten Auswahlknöpfe markiert sein. Beim Testen Ihres neuen Bildes werden Sie feststellen, daß der Screen Painter diese Anwendungslogik automatisch integriert hat.

Übung 4: Das Layout überprüfen

In dieser Übung lernen Sie, wie Sie das Layout eines Bildschirmbildes überprüfen und mit dem Dialogfenster *Feldattribute* die einzelnen Elemente anordnen. Bei der Erstellung Ihrer Benutzungsoberfläche sollten Sie die im *BC - SAP Style Guide* aufgeführten Normen beachten. Öffnen Sie im Fullscreen Editor des Screen Painters das Bild 100, und überprüfen Sie das Layout. Wählen Sie dazu *Dynpro* → *Prüfen* → *Layout*. Das System zeigt daraufhin eine Liste aller Layout-Fehler an.

Mit dem Dialogfenster *Feldattribute* können Sie die Position eines Elements schnell und einfach berichtigen. Gehen Sie dazu folgendermaßen vor:

1. Markieren Sie das Element.
2. Wählen Sie *Feldattr.*

Das System zeigt die aktuellen Attribute des Elements an.

3. Geben Sie einen neuen Wert in das Feld *Zeile* ein.

Dieser Wert definiert die vertikale Position des Elements.

4. Geben Sie einen neuen Wert in das Feld *Spalte* ein.

Dieser Wert definiert die horizontale Position des Elements.

5. Wählen Sie *Schließen*.

Das Element erscheint an der neuen Position.

Überprüfen Sie Ihre Arbeit

Testen Sie nun das überarbeitete Bild 100. Die Position der Elemente sollte sich geändert haben  [\[Extern\]](#).

Übung 5: Das OK-Feld setzen


Übung 5: Das OK-Feld setzen

In dieser Übung verlassen Sie den Fullscreen Editor und verwenden das Bild *Feldliste*, um das OK-Feld zu setzen. Jedes SAP-Bild verfügt über ein OK-Code-Feld, das die Daten vom Bild an die zugrundeliegende Anwendung weiterleitet. Um den OK-Code zu setzen, öffnen Sie Ihr Bild im Bildschirm *Ablauflogik Anzeigen*, und gehen Sie dann folgendermaßen vor:

1. Wählen Sie *Feldliste*. Wenn das Bild bereits im Änderungsmodus geöffnet ist, wählen Sie *Springen* → *Sichten d. Feldliste* → *Feldtypen*.

Das System zeigt die Feldliste für Ihr Bild an. Hier werden dieselben Daten, die auch auf dem Dialogfenster *Attribute* verfügbar sind, in tabellarischer Form dargestellt.

2. Blättern Sie nach unten, bis Sie einen Eintrag finden, der für *FTyp* den Wert *OK* enthält.

Für diesen Eintrag enthält das Feld *Feldname* keine Angabe  [\[Extern\]](#).

3. Geben Sie als *Feldname* **OK-CODE** ein.

Das Feld **OK-CODE** steht für jedes Bild zur Verfügung. Es handelt sich hierbei um ein nicht sichtbares Element, das daher nicht im Fullscreen Editor des Screen Painters angezeigt wird. Mit **OK-CODE** übertragen Sie Informationen vom Bildschirmbild zurück zur Anwendung.

4. Sichern Sie die Änderungen.

Überprüfen Sie Ihre Arbeit

Überprüfen Sie Ihre Arbeit, indem Sie die Feldliste für Ihr Bild anzeigen. Das Feld **OK-CODE** muß unten in der Liste erscheinen.

Zusammenfassung

In Lektion 3 haben Sie sich mit dem Screen Painter vertraut gemacht. Sie haben gelernt, daß sich ein Bildschirmbild aus mehreren grafischen Elementen zusammensetzt, die in einem Fenster angezeigt werden. Grafische Elemente sind beispielsweise Auswahlknöpfe, Drucktasten, Textfelder und Ankreuzfelder.

Sie haben erfahren, daß zu den grafischen Elementen Attribute gehören, die beispielsweise den Namen und die Position des Elements definieren.

Sie haben weiterhin gelernt, wie Sie Ihre Bilder übersichtlicher gestalten können, indem Sie den einzelnen Bildbereichen Gruppenrahmen und Feldbezeichner hinzufügen. Schließlich wurde noch gezeigt, wie Sie mit dem Screen Painter einen Prototyp der neuen Benutzungsoberfläche anzeigen können.

In der nächsten Lektion...

Die grafische Benutzungsoberfläche (GUI) einer ABAP-Anwendung besteht aus Bildschirmbildern und Menüs. In Lektion 4 lernen Sie, wie Sie Menüs erstellen.

Lektion 4: GUI-Status definieren

Lektion 4: GUI-Status definieren

In dieser Lektion lernen Sie, wie Sie einen GUI-Status definieren. Die Lektion hat folgenden Inhalt:

[Überblick \[Seite 51\]](#)

[Übung 1: GUI-Status definieren \[Seite 52\]](#)

[Übung 2: Menüs in die Oberfläche einfügen \[Seite 53\]](#)

[Übung 3: Funktionstasten definieren \[Seite 54\]](#)

[Übung 4: Drucktasten definieren \[Seite 55\]](#)

[Übung 5: Abschließende Arbeiten \[Seite 56\]](#)

[Zusammenfassung \[Seite 57\]](#)


Überblick

In dieser Lektion lernen Sie, wie Sie mit dem Menu Painter einen GUI-Status sowie die zugehörigen Menüleisten erstellen. In der vorherigen Lektion haben Sie ein [Bildschirmbild \[Extern\]](#) erstellt; in dieser Lektion fügen Sie Menüleisten in das Bildschirmbild ein. Wenn Sie diese Lektion durchgearbeitet haben:

- kennen Sie die Konzepte, die den ABAP-Menüs zugrundeliegen.
- können Sie eine Menüleiste für ein Bild erstellen.
- können Sie Funktionstasten definieren.
- können Sie eine Drucktastenleiste für ein Bild erstellen.
- können Sie Fenstertitel angeben.

Konzepte

Sie erstellen eine GUI-Oberfläche in ABAP mit zwei verschiedenen Werkzeugen. Mit dem Screen Painter erstellen Sie Bilder mit den zugehörigen Bedienelementen wie Auswahlknöpfe, Ankreuzfelder, Eingabefelder und Drucktasten. Mit dem Menu Painter erstellen Sie die restlichen Oberflächenelemente:

Status	definiert die Kombination der verfügbaren Menüleisten, Menülisten, Funktionstastenbelegungen und Funktionen einer Oberfläche. Eine Editor-Anwendung kann z.B. zwei verschiedene Status haben: Anzeige- und Änderungsstatus. Beim Ändern ist die Funktion <i>Sichern</i> aktiv, in der Anzeige jedoch nicht.
Menüleisten	definieren die Funktionen, die der Benutzer auswählen kann. Funktionen können an unterschiedlichen Stellen angeboten werden. In einem modalen Dialogfenster erscheinen die Funktionen am unteren Rand als Drucktasten  [Extern] . In Primärfenstern können Funktionen sowohl in Menüs als auch in der Symbol- oder Drucktastenleiste angeboten werden.
Menüliste	enthält die Einträge eines bestimmten Menüs. Das Menü <i>Bearbeiten</i> eines Editors enthält z.B. die Funktionen <i>Kopieren</i> , <i>Ausschneiden</i> und <i>Einfügen</i> .
Funktionstastenbelegung	definiert die Funktionstasten, die bestimmten Oberflächenfunktionen zugeordnet sind.
Funktionen	definieren individuelle Funktionen wie <i>Ersetzen</i> , <i>Suchen</i> oder <i>Ausschneiden</i> .
Titel	definieren die Fenstertitel einer Oberfläche.

Die einzelnen Komponenten können gleichzeitig in verschiedenen Status Anwendung finden. Die von Ihnen definierte Funktion *Löschen* können Sie z.B. in den Status eines Editors, eines Datei-Managers und einer Buchhaltungsanwendung verwenden.

Nachdem Sie den GUI-Status erstellt haben, folgt das [Generieren \[Extern\]](#). Wenn Sie ein Menü generieren, erstellt das System das dazugehörige Laufzeitobjekt. Dieses Laufzeitobjekt wird verwendet, wenn der Benutzer die Anwendung ausführt.

Übung 1: GUI-Status definieren

Übung 1: GUI-Status definieren

Ein [GUI-Status \[Extern\]](#) gehört zu einem bestimmten Bildschirmbild oder zu einer Gruppe von Bildschirmbildern. In dieser Übung definieren Sie den GUI-Status für Ihre Bilder. Gehen Sie dazu folgendermaßen vor:

1. Öffnen Sie die Objektliste für Ihr Programm.
2. Wählen Sie *Objektarten Programm* durch Doppelklick aus.
Sie müssen eine Objektart angeben.
3. Markieren Sie *GUI-Status*.
4. Geben Sie 100 in das Statusfeld ein.
Der Wert, den Sie hier eingeben, identifiziert den Status. Sie können auch einen Namen eingeben, der die Funktion eines Status beschreibt, etwa **Flgderfas**.
5. Wählen Sie *Anlegen*.
Das Dialogfenster *Status anlegen* wird angezeigt.
6. Geben Sie **Flugdaten erfassen** in das Feld *Kurztext* ein.
7. Markieren Sie *Dynpro*.
8. Wählen Sie *Weiter*.
Das Dialogfenster *Status pflegen* wird angezeigt.
9. Sichern Sie den neuen Status.

Überprüfen Sie Ihre Arbeit

Nehmen Sie sich einen Moment Zeit, um zu überprüfen, ob der GUI-Status korrekt angelegt wurde. Da es sich bei einem Status um ein Entwicklungsobjekt handelt, hat das System in der Objektliste Ihres Programms den Eintrag *GUI-Status* erstellt. Überprüfen Sie, ob der neu erstellte Status unter diesem Eintrag aufgeführt wird.

Übung 2: Menüs in die Oberfläche einfügen

Übung 2: Menüs in die Oberfläche einfügen

In dieser Übung lernen Sie, wie Sie Menüleisten in die Bilder integrieren, die Sie in Lektion 3 erstellt haben. Öffnen Sie die Objektliste Ihres Programms, und gehen Sie folgendermaßen vor:


1. Öffnen Sie den Status, den Sie in Übung 1 angelegt haben.

Das Dialogfenster *Status pflegen* wird angezeigt.

2. Vergewissern Sie sich, daß der Status sich im Änderungsmodus befindet.

In diesem Beispiel verwenden Sie die Standardmenüs.

3. Klicken Sie *Normvorschläge* an.

Das System zeigt die Standardmenüs an  [\[Extern\]](#).

4. Ersetzen Sie das Menü *<Objekt>* durch **F1ug**.

Plazieren Sie dazu den Cursor in das Feld, und geben Sie **F1ug** ein.

5. Wählen Sie **F1ug** durch Doppelklick aus.

Eine Liste der Standardmenüeinträge wird angezeigt. Zur Definition von Menüeinträgen müssen Sie gültige Werte in die Spalte *Funk* eingeben.

6. Geben Sie die folgenden Menüeinträge an:

CREA Anlegen

UPDA Ändern

DISP Anzeigen

DELE Löschen

7. Sichern Sie die Änderungen.

Überprüfen Sie Ihre Arbeit

Testen Sie die neue Menüleiste mit dem Bildschirmbild *Flugdaten erfassen*. Vergewissern Sie sich, daß der GUI-Status 100 ausgewählt ist, und wählen Sie *Oberfläche* → *Status testen*. Im Dialogfenster *Statussimulation* geben Sie 100 für die Bildnummer ein.

Wenn Sie die Simulation starten, sehen Sie, daß das System automatisch die Menüs *System* und *Hilfe* eingefügt hat.

Übung 3: Funktionstasten definieren

Übung 3: Funktionstasten definieren

In dieser Übung sollen Sie Funktionstasten für Ihre Menüs definieren. Öffnen Sie Status 100 in Ihrem Programm, und vergewissern Sie sich, daß der Status sich im Änderungsmodus befindet. Gehen Sie dann folgendermaßen vor:

1. Geben Sie **FLUG F-Tasten** in das Feld *Tastenzuordnung* ein.
2. Blättern Sie nach unten bis zum Bereich *Empfohlene Funktionstastenbelegung*.
3. Geben Sie **DELE** als Funktion und für die Funktionstaste *Umsch-F2* ein.
4. Blättern Sie nach unten bis zum Bereich *Frei belegbare Funktionstasten*.
5. Belegen Sie die ersten drei Tasten folgendermaßen:

F5	UPDA	Ändern
F7	DISP	Anzeigen
F8	CREA	Anlegen

6. Sichern Sie Ihre Änderungen.

Überprüfen Sie Ihre Arbeit

Nehmen Sie sich einen Moment Zeit, um die soeben vorgenommenen Änderungen zu testen. Im Moment können Sie die Funktionstasten noch keiner umfassenden Prüfung unterziehen; Sie können sich lediglich vergewissern, daß ihre Auswahl keine Fehlermeldung verursacht.

Übung 4: Drucktasten definieren

In dieser Übung lernen Sie, wie Sie Drucktasten für die Drucktastenleiste und die Symbolleiste definieren. Für alle Funktionen, die Sie Funktionstasten zugewiesen haben, können Sie auch Drucktasten erstellen. In der vorigen Übung haben Sie die Funktionstaste *Anlegen* definiert; in dieser Übung erstellen Sie die entsprechende Drucktaste. Öffnen Sie Status 100, und vergewissern Sie sich, daß er sich im Änderungsmodus befindet. Gehen Sie dann folgendermaßen vor:

1. Wählen Sie das erste Feld unter *Drucktastenleiste*.
2. Geben Sie die Funktion **CREA** ein, und drücken Sie ENTER.

Daraufhin wird automatisch der Text *Anlegen* angezeigt, den Sie bei der Definition der Funktionstaste eingegeben haben, sowie das entsprechende Symbol.

3. Geben Sie die restlichen Funktionen ein:

DISP
UPDA
DELE

4. Fügen Sie die SAP-Standardfunktionen (*Zurück*, *Beenden* und *Abbrechen*) im Bereich *Symbolleiste* ein:



5. Sichern Sie die Änderungen.

Überprüfen Sie Ihre Arbeit

Überprüfen Sie die neuen Drucktasten, indem Sie *Testen* wählen.

Übung 5: Abschließende Arbeiten

Übung 5: Abschließende Arbeiten

In Übung 1 haben Sie den Status für ein Dialogfenster definiert. In dieser Übung lernen Sie, wie Sie den Titel für ein Bildschirmbild oder ein Fenster anlegen und das [Generieren \[Extern\]](#) des GUI-Status durchführen. Öffnen Sie zunächst die Objektliste Ihres Programms.

Schnittstelle benennen

So definieren Sie einen Titel für Ihre Schnittstelle:

1. Wählen Sie *Objektarten Programm* durch Doppelklick aus.
Das System zeigt eine Liste der Objektarten an, die Sie erstellen können.
2. Markieren Sie *GUI-Titel*.
3. Geben Sie 100 für den Titelcode an.
4. Wählen Sie *Anlegen*.
Das Dialogfenster *Titel anlegen* erscheint.
5. Geben Sie als Titel **Flugdaten erfassen** an.
6. Wählen Sie *Sichern*.

Den Status generieren

Wie Sie bereits wissen, erstellt das System bei der Generierung eines Status ein Laufzeitobjekt der Menüs für den Benutzer. Um den neuen Status zu generieren, öffnen Sie die Objektliste des Programms und gehen folgendermaßen vor:

1. Wählen Sie den GUI-Status 100.
2. Wählen Sie *Generieren/Aktivieren*.

In der Statuszeile wird gemeldet, daß der Status 100 generiert wurde  [\[Extern\]](#).

Überprüfen Sie Ihre Arbeit

In der Objektliste des Programms erscheint ein neuer Eintrag, der sämtliche GUI-Titel enthält. Mit der Testfunktion können Sie die Titel, Bildschirme und Menüs der fertiggestellten Oberfläche testen. Dabei können Sie sowohl die Bildschirmnummer als auch eine Titelnummer angeben.

Zusammenfassung

Lektion 4 hatte den GUI-Status zum Inhalt. Sie haben gelernt, daß jede ABAP-Oberfläche über zumindest einen [GUI-Status \[Extern\]](#) verfügt. Ein GUI-Status gehört zu einer Menüleiste und beschreibt die Elemente der Menüleiste und deren Verfügbarkeit.

Sie haben gelernt, wie Sie mit dem Menu Painter Menüleisten erstellen können. Mit dem Menu Painter haben Sie Menüs, Funktionstasten und Drucktasten erstellt. Anschließend haben Sie mit der Testfunktion des Menu Painter einen Prototyp Ihrer neuen Oberfläche getestet.

Weiterhin haben Sie einen GUI-Titel erstellt und gelernt, daß eine vollständige Oberfläche aus einem Bildschirmbild, einem Status und einem Titel besteht.

In der nächsten Lektion...

Die nächste Lektion bietet eine Einführung in das Schreiben von Programm-Coding. Sie lernen, wie Sie Ihre Oberfläche an die zugrundeliegenden Transaktionen binden und Systemmeldungen an den Benutzer erstellen. Am Ende von Lektion 5 finden Sie ein kurzes Beispiel zur Arbeit mit dem Werkzeug zur Fehlersuche und -beseitigung (Debugger).

Lektion 5: Transaktionen programmieren

Lektion 5: Transaktionen programmieren

Lektion 5 erläutert, wie Sie den ABAP-Editor einsetzen. Folgende Themen werden behandelt:

[Überblick \[Seite 59\]](#)

[Übung 1: Ablauflogik schreiben \[Seite 60\]](#)

[Übung 2: Module anlegen \[Seite 61\]](#)

[Übung 3: Globale Variablen deklarieren \[Seite 62\]](#)

[Übung 4: Module programmieren \[Seite 63\]](#)

[Übung 5: Nachrichtenklasse anlegen \[Seite 65\]](#)

[Übung 6: Transaktion testen \[Seite 66\]](#)

[Übung 7: Fehlersuche mit dem Debugger \[Seite 67\]](#)

[Zusammenfassung \[Seite 68\]](#)

Überblick

In dieser Lektion benutzen Sie den Editor, um ABAP-Coding einzugeben. Wenn Sie diese Lektion abgeschlossen haben, können Sie

- zugrundeliegende Programmierkonzepte erkennen.
- die Ablauflogik für ein Dynpro anlegen.
- die Schnittstelle zu einem Bildschirmbild generieren.
- Programmbausteine anlegen.
- Programmcode aus vorhandenen Modulen kopieren.
- die Syntax prüfen.
- Nachrichten anlegen.

Programmierkonzepte

Um Ihre Anwendung fertigzustellen, müssen Sie Anweisungen erstellen, die dem R/3-System mitteilen, wie die vom Benutzer eingegebenen Informationen verarbeitet werden sollen. Diese Anweisungen bestehen aus der [Dynpro \[Extern\]](#) -Ablauflogik und den ABAP-Modulen.

Jedes Bildschirmbild hat eine Ablauflogik. Die Ablauflogik besteht aus Verarbeitungsschritten vor der Ausgabe des Bildes (PBO) und Verarbeitungsschritten, die stattfinden, nachdem der Benutzer Daten eingegeben hat (PAI). Die Ablauflogik wird mit 20 Schlüsselwörtern geschrieben. Diese Schlüsselwörter können ihrerseits auf ABAP-Module verweisen.

Die PBO-Module (Process Before Output) und die PAI-Module (Process After Input) sind spezielle ABAP-Module, die die Verarbeitung steuern. Sie sind im Modulpool einer Transaktion abgelegt.

In dieser Lektion legen Sie auch eine Nachrichtenklasse an. Eine Nachrichtenklasse faßt die Nachrichten zusammen, die von einer bestimmten Anwendung oder Gruppe von Anwendungen benutzt werden.


Übung 1: Ablauflogik schreiben

Übung 1: Ablauflogik schreiben

Die erste Übung zeigt, wie Sie die Ablauflogik für Ihre Bildschirmbilder schreiben. Diese Ablauflogik besteht aus weniger als 20 Schlüsselwörtern, die auf ABAP-Programmmodule zeigen. Ausgangspunkt ist die Liste der Programmobjekte.

1. Holen Sie Dynpro 100.

Der Editor für die Screen-Painter-Ablauflogik erscheint.

2. Vergewissern Sie sich, daß Sie sich im Änderungsmodus befinden.
3. Nach dem Eintrag *process before output*  [\[Extern\]](#) geben Sie folgendes ein:

```
module initialize_100.
```

Das Schlüsselwort MODULE identifiziert das ABAP-Modul, das die Verarbeitung definiert.

4. Geben Sie nach dem Eintrag *process after input* mehrere Leerzeilen ein.
5. Geben Sie folgende Schlüsselwörter und Werte ein:

```
field <tabellenname>-<feldname>.  
module fcode_100.
```

<tabellenname> ist der Name Ihrer Tabelle. <feldname> ist der Name des Feldes "Flug-Nr." Vergessen Sie nicht den Punkt hinter jeder Anweisung.

6. Wählen Sie *Dynpro* → Generieren.

Damit sichern Sie Ihre Änderungen und erzeugen ein [Laufzeitobjekt \[Extern\]](#) Ihres Dynpros.

Überprüfen Sie Ihre Arbeit

Sie haben die Ablauflogik für Ihr Dynpro erstellt. Wählen Sie im Screen Painter *Dynpro* → *Prüfen* → *Syntax*, um sicherzustellen, daß das Coding keine Syntaxfehler enthält.

Übung 2: Module anlegen

In der letzten Übung haben Sie Ablauflogik erstellt, die bestimmte ABAP-Module aufruft. Übung 2 zeigt, wie Sie diese Module anlegen.

1. Öffnen Sie Dynpro 100 im Änderungsmodus.

2. Holen Sie das Modul `initialize_100` mit Doppelklick.

Das System fragt Sie, ob Sie ein neues Modul anlegen wollen. Falls ein Modul dieses Namens bereits vorhanden wäre, würde das System dieses Modul automatisch anzeigen.

3. Wählen Sie *Ja*.

Ein Dialogfenster erscheint, in dem Sie nach dem Namen des Include-Programms gefragt werden, in die Sie das Modul stellen wollen.

4. Wählen Sie *Neues Include*.

Sie müssen hier einen Namen für das Include angeben. Der Name sollte die Form `MZ<bb>001` haben. Beachten Sie die [Namenskonvention für SAP-Objekte \[Seite 14\]](#).

5. Geben Sie den Namen des neuen Include-Programms an, und wählen Sie *Weiter*.

Das System zeigt das Include mit dem neuen Modul an.

6. Sichern Sie das neue Include-Programm, und kehren Sie in die Ablauflogik zurück.

7. Wiederholen Sie die Schritte 1 bis 6 für das Modul `rcode_100`, verwenden Sie jedoch die Form `MZ<bb>I01` für den Namen des Include-Programms.

Überprüfen Sie Ihre Arbeit

Testen Sie Ihre Änderungen. Wenn Sie in einer Dynpro-Ablauflogik eine Moduldefinition durch Doppelklick auswählen, zeigt das System automatisch das Coding an, auf das die Definition sich bezieht. Testen Sie das mit Ihren neuen Modulen. Sie können auch Ihre Tabelle automatisch anzeigen, indem Sie die Feldreferenz in der Ablauflogik mit Doppelklick auswählen.

Übung 3: Globale Variablen deklarieren

Übung 3: Globale Variablen deklarieren

In dieser Übung deklarieren Sie globale Variablen. Sie legen diese Deklarationen im Top-Include-Programm Ihres Programms an. Zur Erinnerung: Sie haben dieses Include in [Übung 4 der Lektion 1 \[Seite 22\]](#) angelegt. Zeigen Sie die Liste Ihrer Programmobjekte an, und gehen Sie dann folgendermaßen vor:

1. Markieren Sie Ihr Top-Include-Programm, und wählen Sie *Ändern*.

Das System öffnet Ihr Programm zur Bearbeitung. *program* muß die erste Deklaration im Programm sein. Das System hat diese Deklaration automatisch hinzugefügt, als Sie das Include angelegt haben. Geben Sie Ihre Änderungen unterhalb der Programmdeklaration ein.

2. Deklarieren Sie Ihre Tabelle mit folgender Anweisung:

```
tables: <tabellenname>.
```

Ersetzen Sie die Variable `<tabellenname>` mit dem Namen Ihrer Tabelle. Für jedes Element in einem Bildschirmfeld, in dem der Benutzer Daten eingeben kann und aus dem Ihr Programm Daten entnimmt, müssen Sie eine Variable deklarieren.

3. Fügen Sie die folgenden Deklarationen hinzu:

```
data answer.  
data: ok-code(4), fcode (4).
```

Hier müssen Sie eine Nachrichtenklasse angeben, die die Form `z<x>` oder `y<x>` haben sollte. Beachten Sie die [Namenskonvention für SAP-Objekte \[Seite 14\]](#).

4. Erweitern Sie die Anweisung PROGRAM durch eine Nachrichtennummer.

```
program sapmzaaa message-id <xx>.
```

5. Sichern Sie Ihre Änderungen.

Überprüfen Sie Ihre Arbeit


Testen Sie Ihre Änderungen, und vergewissern Sie sich, daß Sie die globalen Daten richtig eingegeben haben. Wählen Sie *Programm* → *Prüfen* → *Aktuelles Programm*, um Ihre Programmsyntax zu prüfen. Das System zeigt eventuelle Fehler an.

Sie können auch durch Doppelklick auf den Tabellennamen die Tabelle anzeigen.

Übung 4: Module programmieren

In der vorhergehenden Übung haben Sie mit dem ABAP-Editor Coding in Ihr Top-Include-Programm eingefügt. In dieser Übung werden Sie über die Tastatur und mit besonderen Hilfsmitteln der Workbench-Umgebung sowohl Coding kopieren als auch vordefinierte Funktionen automatisch einbinden.


Coding im Editor eingeben

1. Öffnen Sie die Liste *PBO-Module* in der Liste Programmobjekte.
2. Markieren Sie das Modul *INITIALIZE_100*, und wählen Sie *Ändern*.
Das System zeigt das Modul im Änderungsmodus an.
3. Geben Sie in der Zeile nach *module initialize_100 output*  [\[Extern\]](#) folgendes ein:


```
set pf-status '100'.  
set titlebar '100'.
```
4. Sichern Sie Ihre Änderungen, und kehren Sie in die Programmliste zurück.

Coding aus einem anderen Programm kopieren

Häufig erstellen Programmierer neue Programme, indem sie bestehende ähnliche Programme kopieren und entsprechend abändern. Dabei helfen Ihnen die Werkzeuge der Workbench:

1. Kehren Sie zum Einstiegsbild des Object Browser zurück.
2. Markieren Sie *Programm*, und geben Sie als Programmnamen **TUTPROG** ein.
Das Musterprogramm wird angezeigt.
3. Wählen Sie *Anzeigen*.
4. Öffnen Sie das PAI-Modul **FRAGMENT**.
Das System zeigt das Modul im Anzeigemodus an.
5. Stellen Sie den Cursor auf die Zeile mit dem Eintrag *fcode = ok-code*.
6. Wählen Sie *Bearbeiten* → *Markieren*.
Das System markiert die Zeile  [\[Extern\]](#).
7. Markieren Sie den Eintrag *endcase* in Zeile 97, und wählen Sie *Markieren*.
8. Wählen Sie *Block/Ablage* → *Kopieren in Clipboard*.
9. Verlassen Sie den Baustein, und kehren Sie in die Programmobjektliste zurück.
Wenn Sie möchten, können Sie auch die Funktion *Sprungmarken* verwenden, um zu Ihrer Objektliste zurückzukehren.
10. Markieren Sie das Modul **FCODE_100**, und wählen Sie *Ändern*.
11. Stellen Sie den Cursor auf die leere Zeile unter `module fcode_100 input`.
12. Wählen Sie *Block/Ablage* → *Einsetzen aus Clipboard*.
Das System fügt den Abschnitt aus **FRAGMENT** ein.

Übung 4: Module programmieren

13. Wählen Sie *Bearbeiten* → *Ersetzen*, um *wbtable* mit Ihrem Tabellennamen  [Extern] zu ersetzen.
14. Sichern Sie Ihre Änderungen.

Systemfunktionen einfügen

Sie können im ABAP-Editor Funktionsschablonen automatisch in Ihr Coding einfügen und diese Schablonen dann für Ihre Anwendung anpassen. So fügen Sie Schablonen ein:

1. Gehen Sie im Änderungsmodus in Ihr Modul `fcod_e_100`.
2. Suchen Sie den Eintrag `endif`, der sich unmittelbar vor dem Eintrag `when space` befindet.

Dieser `endif`-Eintrag muß ungefähr um Zeile 97 herum erscheinen.

3. Fügen Sie vor `endif` eine Leerzeile ein.
4. Wählen Sie *Bearbeiten* → *Anweisungsmuster*.

Das Dialogfenster zum Einfügen einer Anweisung erscheint.

5. Wählen Sie CALL FUNCTION.
6. Geben Sie `POPUP_TO_CONFIRM_LOSS_OF_DATA` ein.
7. Wählen Sie *Weiter*.

Das System fügt eine Schablone für diese Funktion ein.

8. Fügen Sie Ihrem Coding die Bedingung IF hinzu und vervollständigen Sie die Schablone.

Das Ergebnis sieht folgendermaßen aus (den roten Text müssen Sie hinzufügen):

```
message e004 with <tabellenname>-flid.
else.
    call function 'POPUP_TO_CONFIRM_LOSS_OF_DATA'
        exporting
            textline1 = 'Flug löschen?'
            TEXTLINE2 = ' '
            titel = 'Achtung'
            START_COLUMN = 25
            START_ROW = 6
    importing
        answer = answer.
    check answer ne 'N'.
    delete <tabellenname>.
    clear <tabellenname>.
    message s003 with <tabellenname>-flid.
endif.
```


9. Sichern Sie Ihre Änderungen.

Überprüfen Sie Ihre Arbeit

Prüfen Sie die Syntax in Ihrem Modul mit *Programm* → *Prüfen*.

Übung 5: Nachrichtenklasse anlegen

Diese Übung zeigt, wie Sie eine Nachrichtenklasse anlegen. Nachrichtenklassen dienen dazu, die passenden Nachrichten an den Endbenutzer auszugeben. Öffnen Sie Ihre Programmobjektliste, und gehen Sie folgendermaßen vor:

1. Gehen Sie in Ihr Top-Include-Programm.
2. Wählen Sie die Nachrichtennummer mit Doppelklick aus.
Das System fordert Sie auf zu bestätigen, daß Sie eine neue Nachrichtenklasse anlegen wollen.
3. Wählen Sie *Ja*.
Das System zeigt das Bild *Nachrichtenklasse pflegen* an.
4. Geben Sie im Feld *Kurztext* folgendes ein:
Nachrichten zur Fluganwendung
5. Sichern Sie Ihre Änderungen.
Das Dialogfenster *Pflegen Objektkatalogeintrag* erscheint.
6. Wählen Sie *Lokales Objekt*.
Sie kehren zum Bild *Nachrichtenklasse pflegen* zurück.
7. Wählen Sie *Nachrichten*.
Das System zeigt das Bild *Nachrichten pflegen* an.
8. Wählen Sie *Alle pflegen*, um die Nachrichten zu bearbeiten.
9. Geben Sie folgendes  **[Extern]** ein:

```
000  Flug bereits erfasst
001  Flug & erfolgreich ANGELEGT
002  Flug & erfolgreich AKTUALISIERT
003  Flug & erfolgreich GELÖSCHT
004  Flug & existiert nicht
```


Zur Laufzeit ersetzt das System das Zeichen & (Ampersand) mit der entsprechenden Flugnummer.
10. Sichern Sie Ihre Änderungen.

Überprüfen Sie Ihre Arbeit

Holen Sie das Modul `fcode_100`. Wählen Sie eine Nachrichtennummer durch Doppelklick aus, z.B. `e000`. Wenn Sie Ihre Nachrichten erfolgreich angelegt haben, zeigt das System die Nachrichtendefinition an.


Übung 6: Transaktion testen

Übung 6: Transaktion testen

In diesem Beispiel testen Sie Ihre Transaktion aus einem anderen SAP-Modus heraus. Starten Sie Ihre Transaktion aus der Programmobjektliste:

1. Markieren Sie Ihren Programmnamen, und wählen Sie *Entwicklungsobjekt* → *Generieren/Aktivieren*.

Sollte das Programm noch Syntaxfehler enthalten, gibt das System eine entsprechende Fehlermeldung aus.

2. Öffnen Sie einen zweiten SAP-Modus, indem Sie *System* → *Erzeugen Modus* wählen.
3. Geben Sie Ihren Transaktionsnamen in das Befehlsfeld  [\[Extern\]](#) ein.

Das System startet Ihre Transaktion in diesem Fenster.

4. Legen Sie einen Flug an.
5. Kehren Sie in den ersten Modus zurück, und gehen Sie in Ihre private Objektliste.
6. Wählen Sie *DDIC-Objekte*.
7. Markieren Sie Ihre Tabelle, und wählen Sie *Ausführen*.

Das Auswahlbild des Data Browser erscheint.

8. Wählen Sie *Programm* → *Ausführen*, um Ihre Tabelle anzuzeigen.

Das System zeigt alle Flüge in Ihrer Anwendungsdatenbank an.

Überprüfen Sie Ihre Arbeit

Testen Sie alle Funktionen Ihrer Oberfläche. Sie können z.B.:

- einen Flug mit Ziel Peru anlegen
- einen bestehenden Flug löschen
- einen nicht bestehenden Flug anzeigen
- einen bestehenden Flug ändern

Vergewissern Sie sich, daß jede Ihrer Fehlermeldungen an der richtigen Stelle im Dialog erscheint. Wenn Sie Ihren Test abgeschlossen haben, überarbeiten Sie Ihre Flugdatenbank noch einmal.

Übung 7: Fehlersuche mit dem Debugger

Sie können den Debugger (Online-Testhilfe) nutzen, um Fehler in Ihrem Coding zu finden. Wenn Sie sich in Ihrer Transaktion befinden, können Sie den Debugger durch Eingeben von `/h` im Befehlsfeld starten. Für diese Übung starten Sie den Debugger aus Ihrer Programmobjektliste:

1. Markieren Sie Ihre Transaktion, und wählen Sie *Entwicklungsobjekt* → *Testen/Ausführen*.
Das Dialogfenster *Ausführungsarten* erscheint.
2. Wählen Sie *Debugging* und dann *Weiter*.
Ihre Anwendung wird im Debugging-Modus angezeigt.
3. Wählen Sie die Felddeklaration der Flugnummer durch Doppelklick aus.
Das System fügt diese Variable in den Gruppenrahmen *Variablen* ein. Zunächst ist keine Flugnummer vorhanden.
4. Wählen Sie *Einzelanschritt*.
Das System startet die Ausführung Ihres Programms. Nach jedem Schritt kehren Sie zum Debugger zurück.
5. Gehen Sie Schritt für Schritt durch das gesamte Programm durch.

Überprüfen Sie Ihre Arbeit

Um Ihre Arbeit zu testen, können Sie Ihre Flugtabelle intern analysieren. Wählen Sie aus Ihrer Programmobjektliste das Tabellenobjekt, und wählen Sie *Umfeld* → *Data Browser* → *Tabelleninhalt*. Wenn Sie sich im Data Browser befinden, können Sie mit *Ausführen* eine Tabelle mit den von Ihnen gemachten Einträgen anzeigen.

Zusammenfassung

Zusammenfassung

Lektion 5 hat Sie mit der Verwendung des ABAP-Editors zum Erfassen von ABAP-Coding vertraut gemacht. Sie haben einige der Möglichkeiten kennengelernt, mit denen Sie Ihre eigenen Anwendungen entwickeln können. In dieser Lektion haben Sie den Editor verwendet, um:

- eine Dynpro-Ablauflogik anzulegen.
- eine Schnittstelle zum Bildschirmbild zu generieren.
- PAI- und PBO-Module anzulegen.
- Programmbausteine aus vorhandenen Modulen zu kopieren.
- Schablonen einzufügen und zu modifizieren.

In der nächsten Lektion...

In dieser Lektion haben Sie Ihre Transaktion zum Anlegen eines Fluges abgeschlossen. Die nächste Lektion führt Sie in die Werkzeuge und Konzepte ein, die Sie für die Programmierarbeit in einem Team benötigen.

Lektion 6: Arbeiten im Team

Diese Lektion führt Sie in die Werkzeuge und Konzepte ein, mit denen Sie vertraut sein müssen, um ABAP-Anwendungen im Team zu entwickeln. Die folgenden Themen werden behandelt:

[Überblick \[Seite 70\]](#)

[Übung 1: Entwicklungsklasse anlegen \[Seite 72\]](#)

[Übung 2: Liste der Änderungsaufträge überprüfen \[Seite 73\]](#)

[Übung 3: Entwickler hinzufügen \[Seite 74\]](#)

[Übung 4: Programm anlegen \[Seite 75\]](#)

[Übung 5: Änderungsaufträge freigeben \[Seite 76\]](#)

[Zusammenfassung \[Seite 78\]](#)

Überblick

Überblick

Nachdem Sie die Rolle der einzelnen Workbench-Werkzeuge in der Anwendungsentwicklung kennengelernt haben, führt Sie diese Lektion in die Werkzeuge und Konzepte ein, die Sie benötigen, um eine Anwendung in einem Team von Programmierern zu entwickeln. Nach Bearbeitung dieser Lektion können Sie

- die zugrundeliegenden Konzepte für die Entwicklung von ABAP-Anwendungen im Team verstehen.
- eine Entwicklungsklasse im SAP-System anlegen.
- Änderungen Ihrer Entwicklungsklasse verfolgen.
- neue Änderungsaufträge freigeben.

ABAP-Entwicklungskonzepte für die Arbeit im Team

Mit ABAP können Sie die Arbeit an großen Projekten unter mehreren Entwicklern aufteilen. Nehmen wir z.B. ein Projekt, in dem eine Finanzbuchhaltungsanwendung mit einem Debitoren- und einem Kreditorenmodul entwickelt werden soll. Die ABAP-Umgebung unterstützt Sie beim Anlegen eines Arbeitsbereichs für das Projekt in Ihrem System. Sie können dann jedem Entwickler bestimmte Aufgaben zuweisen und den Fortschritt der Arbeiten verfolgen.

Das Werkzeug, das Sie für die Verfolgung von Entwicklungsprojekten einsetzen, ist der Workbench Organizer. Die Entwicklung eines großen ABAP-Projekts in einer Entwicklungsgruppe besteht aus folgenden Schritten:

- **Entwicklungsklasse anlegen**

Eine [Entwicklungsklasse \[Extern\]](#) faßt logisch zusammenhängende Objekte zusammen, z.B. alle Objekte, die zu einer Finanzbuchhaltungsanwendung gehören. Eine Entwicklungsklasse ist eine Art [Entwicklungsobjekt \[Extern\]](#).
- **Änderungsauftrag anlegen**

Ein [Änderungsauftrag \[Extern\]](#) protokolliert die Änderungen an einem Entwicklungsobjekt. Eine solche Änderung ist z.B. das Anlegen eines Programms in einer Entwicklungsklasse. Ein Änderungsauftrag bezieht sich immer auf genau einen ABAP-Benutzer.
- **Projekt programmieren**

ABAP-Programme bestehen aus Transaktionen, Reports, Bildschirmbildern und anderen Entwicklungsobjekten. Wenn die Komponenten eines Projekts von verschiedenen Entwicklern bearbeitet werden, müssen Sie jedem Programmierer eine [Aufgabe \[Extern\]](#) innerhalb des Änderungsauftrags zuweisen. So können Sie verfolgen, wer ein Programm geändert hat.
- **Änderungsauftrag freigeben**

Solange ein Änderungsauftrag oder eine Aufgabe mit einem Entwicklungsobjekt zusammenhängt, ist das Objekt gesperrt. Die Sperre verhindert auch, daß andere Benutzer dieses Objekt bearbeiten. Nach Abschluß und Test der Änderungen wird der Auftrag vom Entwickler freigegeben.

In einer Standard-SAP-Installation laufen das Entwicklungssystem und das Produktivsystem getrennt auf zwei verschiedenen Maschinen. Neue Anwendungen werden im Entwicklungssystem

erarbeitet und anschließend in das Produktivsystem transportiert. Die tägliche Arbeit findet im Produktivsystem statt.

Die Trennung von Produktiv- und Entwicklungssystem ist wegen des Zeitpunkts der Wirksamkeit von Änderungen in den Anwendungen erforderlich. Denn wenn in einer bestehenden ABAP-Anwendung Änderungen vorgenommen werden, werden sie sofort wirksam. Durch die Trennung wird verhindert, daß neue Entwicklungen den täglichen Arbeitsfluß stören.

Übung 1: Entwicklungsklasse anlegen

Übung 1: Entwicklungsklasse anlegen

In dieser Übung legen Sie eine Entwicklungsklasse für eine fiktive Finanzbuchhaltungsanwendung an. Zur Erinnerung: Eine Entwicklungsklasse faßt logisch zusammenhängende Objekte zusammen. Sie werden dabei auch einen Änderungsauftrag anlegen, der die Entwicklung der Anwendung protokolliert und die Objekte in dieser Entwicklungsklasse gegen Änderungen durch nicht autorisierte Benutzer sperrt.

Ausgangspunkt ist das Einstiegsbild des Object Browser. Gehen Sie folgendermaßen vor:

1. Markieren Sie Entwicklungsklasse.

Jetzt müssen Sie einen Namen für Ihre Entwicklungsklasse wählen. Verwenden Sie hier einen Namen der Form **z<xxx>**, wobei **<xxx>** für Ihre Initialen steht.


2. Geben Sie den Namen für Ihre Entwicklungsklasse ein.

3. Wählen Sie *Anzeigen*.

Das System bestätigt, daß die Klasse noch nicht existiert, und fragt, ob Sie sie anlegen wollen.

4. Wählen Sie *Ja*.


Das Bild *Sicht: "Entwicklungsklassen" ändern: Detail* erscheint.

5. Geben Sie eine Kurzbeschreibung für die neue Entwicklungsklasse ein  [\[Extern\]](#).

6. Wählen Sie *Sichern*.

Das System fordert Sie auf, einen Änderungsauftrag anzulegen.

7. Wählen Sie *Auftrag anlegen*.

Das Bild *Auftrag anlegen* erscheint. Sie müssen hier einen möglichst beschreibenden Namen für den Änderungsauftrag angeben. Dieser Name sollte den Entwicklungsstand wiedergeben (z.B. Release 1.0 für eine Neuentwicklung)  [\[Extern\]](#).

8. Geben Sie eine Kurzbeschreibung ein, und wählen Sie *Sichern*.

Das Bild *Abfrage Änderungsauftrag* erscheint.

9. Wählen Sie *Weiter*.

Der Anfang Ihrer neuen Entwicklungsklassenliste wird angezeigt.

Überprüfen Sie Ihre Arbeit

Ihre Entwicklungsliste enthält zunächst lediglich einen Eintrag für Entwicklungsklassenobjekte. Sie können die zugehörigen Informationen durch Doppelklick auf den Namen der Entwicklungsklasse überprüfen.

Übung 2: Liste der Änderungsaufträge überprüfen

Übung 2: Liste der Änderungsaufträge überprüfen

Diese Übung zeigt, wie Sie die Änderungsaufträge anzeigen können, die Ihnen zugeordnet sind. Zur Erinnerung: Ein Änderungsauftrag protokolliert die Änderungen an einem Entwicklungsobjekt. Jeder Änderungsauftrag ist genau einem ABAP-Benutzer zugeordnet. So können Sie Ihre Änderungsaufträge anzeigen. Gehen Sie in Ihre neue Entwicklungsklasse und führen Sie folgendes aus:

1. Wählen Sie *Umfeld* → *Workbench Organizer*.

Das Einstiegsbild des Workbench Organizer erscheint.

2. Überprüfen Sie, ob die folgenden Werte gesetzt sind:

Aufträge für Benutzer


Änderbar

Lokal

3. Wählen Sie *Anzeigen*.

Die Änderungsaufträge, an denen Sie beteiligt sind, werden angezeigt.

Überprüfen Sie Ihre Arbeit

Zeigen Sie alle Einträge der Hierarchie an  [\[Extern\]](#). Ihre Liste enthält einen Änderungsauftrag und eine Aufgabe. Der Änderungsaufgabe ist ein Benutzername zugeordnet. Ihre Liste sollte zeigen, daß Sie eine einzige Aktion in Verbindung mit der neuen Entwicklungsklasse durchgeführt haben, nämlich das Anlegen der Klasse selbst.

Übung 3: Entwickler hinzufügen

Übung 3: Entwickler hinzufügen

Die letzte Übung hat gezeigt, wie Sie den (die) Ihnen zugeordneten [Änderungsauftrag \[Extern\]](#) anzeigen können. Nur Ihr eigener Name erscheint in Verbindung mit dieser Liste. Es werden jedoch drei Entwickler am Projekt beteiligt sein. Wenn Sie die Aktivitäten eines anderen Entwicklers in derselben Änderungsauftragsliste verfolgen wollen, müssen Sie zuerst eine [Aufgabe \[Extern\]](#) für diesen Benutzer anlegen, indem Sie den betreffenden Benutzer der Liste hinzufügen.

1. Gehen Sie in die Liste der Änderungsaufträge.
2. Markieren Sie den Änderungsauftrag.

Wenn Sie den Auftrag nicht markieren, können Sie keinen Benutzer hinzufügen.


3. Wählen Sie *Mitarbeiter hinzufügen*.

Das Bild *Mitarbeiter hinzufügen* erscheint.

4. Geben Sie einen Benutzernamen ein.
5. Wählen Sie *Weiter*.

Der Benutzer erscheint in der Liste.

Überprüfen Sie Ihre Arbeit

Ihre Liste sollte jetzt die Namen von 2 Programmierern enthalten  [\[Extern\]](#). Da der neue Benutzer (KESSLERK) noch keine Aktivitäten in Verbindung mit der Aufgabe durchgeführt hat, erscheint kein Mappensymbol neben dem Namen.

Übung 4: Programm anlegen

Sie sind mit dem Anlegen eines Programms bereits aus der [Lektion 1 \[Seite 22\]](#) vertraut. Das Vorgehen beim Anlegen eines Programms in einer Teamumgebung ist jedoch etwas anders. Wenn Sie ein Programm oder sonstige Entwicklungsobjekte in einer Teamumgebung anlegen, müssen Sie sowohl eine [Entwicklungs-klasse \[Extern\]](#) als auch einen [Änderungsauftrag \[Extern\]](#) angeben. So legen Sie ein Programm in einer Teamumgebung an:

1. Gehen Sie in die Liste der Entwicklungsobjekte, die Sie in Übung 1 angelegt haben.
2. Wählen Sie *Objektarten Entwicklungs-klasse* mit Doppelklick aus.
Das System fordert Sie auf, die Objektart anzugeben, die Sie anlegen möchten.
3. Vergewissern Sie sich, daß *Programmobjekt* markiert ist, und wählen Sie *Weiter*.
Eine Liste der möglichen Programmobjektarten wird angezeigt.
4. Wählen Sie einen Programmnamen.
Zur Erinnerung: Es gibt eine spezielle [Namenskonvention für SAP-Objekte \[Seite 14\]](#), an die Sie sich halten sollten. Wählen Sie einen Programmnamen der Form **SAPMZ<bb>**, wobei **<bb>** für Ihre Initialen steht.
5. Geben Sie den Programmnamen ein.
6. Vergewissern Sie sich, daß die Option *Programm* markiert ist, und wählen Sie *Anlegen*.
Das System fordert Sie auf, Ihre Angaben zu bestätigen.
7. Überprüfen Sie den Programmnamen.
8. Deaktivieren Sie *MIT TOP INCLUDE*.
9. Wählen Sie *Weiter*.

Das Bild für die Eingabe der Programmattribute erscheint.

10. Geben Sie folgende Attribute für Ihr Programm an:

<i>Titel</i>	Tutorial: Debitorenbuchhaltung
<i>Typ</i>	M
<i>Anwendung</i>	*

11. Wählen Sie *Sichern*.
Sie werden aufgefordert, einen Änderungsauftrag anzugeben.
12. Geben Sie den Änderungsauftrag an, den Sie in [Übung 1 \[Seite 72\]](#) angelegt haben.
13. Wählen Sie *Weiter*.

Das System kehrt zu den Programmattributen zurück.

Überprüfen Sie Ihre Arbeit

Zeigen Sie die Liste Ihrer Änderungsaufträge an. Ihre Aufgabe enthält einen Eintrag für ABAP-Programme. Unter diesem Eintrag finden Sie Ihr neues Programm. Sie können das Programm durch Doppelklick auf den Namen anzeigen.

Übung 5: Änderungsaufträge freigeben

Übung 5: Änderungsaufträge freigeben

In dieser Übung geben Sie den zuvor angelegten Änderungsauftrag frei. Wenn Sie einen Änderungsauftrag freigeben, machen Sie neue Programme oder Funktionen für andere Benutzer im System verfügbar. Nach der Freigabe können andere Benutzer die enthaltenen Objekte ändern. Sie können einen Änderungsauftrag erst dann freigeben, wenn Sie zuvor alle zugehörigen Aufgaben freigegeben haben. Um Ihren Änderungsauftrag freizugeben, zeigen Sie die Liste der Änderungsaufträge an, und gehen Sie folgendermaßen vor:

1. Markieren Sie Ihre Aufgabe.
2. Wählen Sie *Freigeben*.

Ein Bild erscheint, in dem Sie eine Beschreibung zum Auftrag bzw. zur Aufgabe erfassen können.
3. Geben Sie eine kurze Beschreibung für die freizugebende Aufgabe ein.

Beispiel:
Release 1 des Finanzbuchhaltungsprogramms
4. Wählen Sie *Endfassung sichern*.

Die Aufgabe wird zur Freigabe vorgemerkt.
5. Wählen Sie *Zurück*.

Die Liste der Änderungsaufträge wird wieder angezeigt. Da Sie eine Aufgabe, die Ihnen nicht gehört, nicht freigeben können, müssen Sie zunächst alle Aufgaben des Änderungsauftrags Ihrem Benutzernamen zuordnen. Das ist möglich, weil Sie Eigentümer des Änderungsauftrags sind.
6. Markieren Sie die Aufgabe, die dem anderen Benutzer gehört.
7. Wählen Sie *Eigentümer ändern*.

Ein Dialogfenster erscheint, in dem Sie den Benutzernamen ändern können.
8. Geben Sie Ihren Namen im Feld *Neuer Benutzer* ein.
9. Wählen Sie *Bestätigen*.


Die Aufgabe wird Ihrem Benutzernamen zugeordnet. Da sie keine Aktivitäten enthält, sollten Sie sie löschen anstatt freigeben.
10. Markieren Sie die Aufgabe, und wählen Sie *Auftrag/Aufgabe* → *Löschen*.

Das System fordert Sie auf, die Aktion zu bestätigen.
11. Wählen Sie *Ja*.

Das System löscht die leere Aufgabe.
12. Markieren Sie den Änderungsauftrag.
13. Wählen Sie *Freigeben*.

Das System gibt alle Änderungen im System frei.

Übung 5: Änderungsaufträge freigeben**Überprüfen Sie Ihre Arbeit**

Nehmen Sie sich Zeit, Ihre Änderungen zu testen. Gehen Sie in das Einstiegsbild des *Workbench Organizer*, und markieren Sie *Änderbar* und *Freigegeben*  [Extern]. Wenn Sie danach die Änderungsaufträge anzeigen, an denen Sie beteiligt sind, erscheinen sowohl die freigegebenen als auch die laufenden (änderbaren) Änderungsaufträge. Sie können die Anzeige auf die freigegebenen Aufträge einschränken, indem Sie *Änderbar* deaktivieren.

Zusammenfassung

Zusammenfassung

Lektion 6 hat Sie in die Grundkonzepte eingeführt, die Sie kennen müssen, wenn Sie ABAP-Anwendungen im Team entwickeln wollen. Sie sollten jetzt mit folgenden Begriffen vertraut sein:

- Entwicklungsklasse
- Änderungsauftrag
- Aufgabe

Sie haben gelernt, daß eine Entwicklungsklasse logisch zusammenhängende Objekte zusammenfaßt. So hängen z.B. alle Entwicklungsobjekte einer Finanzbuchhaltungsanwendung logisch zusammen. Den Entwicklungsprozeß können Sie mit Hilfe von Änderungsaufträgen und Aufgaben verfolgen. Sowohl Änderungsaufträge als auch Aufgaben gehören zu einem bestimmten Benutzer.

Sie haben den Workbench Organizer genutzt, um alle Änderungsaufträge anzuzeigen, an denen ein Benutzer beteiligt ist. Sie haben einen Benutzer in einem bestehenden Änderungsauftrag eingefügt und später den Eigentümer einer Aufgabe geändert, um diese Aufgabe löschen zu können. Schließlich haben Sie mit dem Workbench Organizer Ihr Programm für die allgemeine Nutzung freigegeben.

Weitere Informationen

Sie haben die Lektionen des *ABAP Workbench Tutorial* abgeschlossen, das Ihnen eine Vorstellung von der Funktion der einzelnen Werkzeuge bei der Entwicklung einer ABAP-Anwendung vermittelt hat. Wenn Sie mit der ABAP-Programmierung vertraut sind, finden Sie weitere Informationen in der Dokumentation [ABAP Development Workbench: Werkzeuge \[Extern\]](#). Hier werden alle Workbench-Werkzeuge im Detail behandelt.

Näheres zum Programmieren mit ABAP finden Sie in der Dokumentation [ABAP-Benutzerhandbuch \[Extern\]](#).