

**SAP<sup>®</sup> Business One 中文版培训教材**  
**—SDK**

**2004 年 8 月**  
**SAP 中国公司**

## 目 录

|            |                                                             |           |
|------------|-------------------------------------------------------------|-----------|
| <b>第一章</b> | <b>SAP BUSINESS ONE SDK中文版概述.....</b>                       | <b>4</b>  |
| 一、         | 课程设置.....                                                   | 4         |
| 二、         | SAP Business One SDK简介.....                                 | 5         |
| 三、         | SAP Business One SDK的版本.....                                | 10        |
| 四、         | SAP Business One SDK的兼容性.....                               | 10        |
| 五、         | SAP Business One SDK应用示例: Data Transfer Workbench.....      | 11        |
| 六、         | 命名规则.....                                                   | 13        |
| <b>第二章</b> | <b>用户自定义表和字段.....</b>                                       | <b>14</b> |
| 第一节        | 用户自定义表、字段描述.....                                            | 14        |
| 一、         | 创建自定义表.....                                                 | 14        |
| 二、         | 添加自定义字段.....                                                | 16        |
| 三、         | 添加数据.....                                                   | 19        |
| 第二节        | 练习.....                                                     | 23        |
| <b>第三章</b> | <b>DI API.....</b>                                          | <b>24</b> |
| 第一节        | DI API 概述.....                                              | 24        |
| 一、         | 总体介绍.....                                                   | 24        |
| 二、         | 安装.....                                                     | 24        |
| 三、         | 体系结构.....                                                   | 24        |
| 四、         | 存取模式.....                                                   | 25        |
| 五、         | 版本控制.....                                                   | 26        |
| 六、         | 技术支持.....                                                   | 27        |
| 第二节        | 业务数据对象.....                                                 | 28        |
| 一、         | Business Object.....                                        | 28        |
| 二、         | Non-Business Objects .....                                  | 45        |
| 第三节        | WORKING WITH XML.....                                       | 49        |
| 一、         | 什么是XML.....                                                 | 49        |
| 二、         | XML文件的优点.....                                               | 49        |
| 三、         | DI API.中使用XML .....                                         | 49        |
| 四、         | 绑定XML文件.....                                                | 51        |
| 五、         | 其它工具中使用.....                                                | 51        |
| 六、         | 练习.....                                                     | 51        |
| 第四节        | 事务.....                                                     | 51        |
| 一、         | 概述.....                                                     | 51        |
| 二、         | 例子.....                                                     | 52        |
| 三、         | 练习.....                                                     | 53        |
| 第五节        | DI_API高级特性.....                                             | 53        |
| 一、         | Data Interface MetaData Objects .....                       | 53        |
| 二、         | Data Interface SBObob Object.....                           | 57        |
| 三、         | Data Interface Recordset Object vs. DataBrowser Object..... | 59        |
| <b>第四章</b> | <b>UI API.....</b>                                          | <b>62</b> |

|                              |            |
|------------------------------|------------|
| 第一节 UI API概述 .....           | 62         |
| 一、目的.....                    | 62         |
| 二、适用性.....                   | 62         |
| 三、开发说明.....                  | 62         |
| 四、运行需求.....                  | 62         |
| 第二节 UI API 体系介绍.....         | 62         |
| 一、体系.....                    | 62         |
| 二、兼容性.....                   | 63         |
| 第三节 UI API对象 .....           | 64         |
| 一、Application Object .....   | 65         |
| 二、Form Object .....          | 67         |
| 三、Item Object.....           | 70         |
| 四、ItemEvent Object .....     | 71         |
| 五、BubbleEvent.....           | 73         |
| 六、MenuItem Object.....       | 75         |
| 七、Menu Event Object .....    | 76         |
| 八、XML使用 .....                | 78         |
| 九、DataBind Object.....       | 80         |
| 第四节 应用同步处理.....              | 82         |
| 一、文档控制.....                  | 82         |
| 二、强制事件处理.....                | 83         |
| 第五节 UI API高级特性 .....         | 85         |
| 一、Working with Matrix.....   | 85         |
| 二、屏幕制作器.....                 | 86         |
| 三、Single Sign-On.....        | 89         |
| 四、Filtering Events .....     | 91         |
| 五、ToolBar Control .....      | 92         |
| 六、LinkButton .....           | 95         |
| <b>第五章 ADD ON的注册和安装.....</b> | <b>97</b>  |
| 第一节 总体介绍.....                | 97         |
| 一、概览.....                    | 97         |
| 第二节 ADD ON注册 .....           | 97         |
| 一、Add on的启动原理.....           | 97         |
| 二、Add on的注册 .....            | 98         |
| 第三节 安装程序.....                | 100        |
| 一、原理.....                    | 100        |
| 二、安装程序.....                  | 100        |
| <b>第六章 CASE STUDY.....</b>   | <b>101</b> |
| 第一节 需求分析.....                | 101        |
| 一、现有功能分析.....                | 101        |
| 二、差异分析.....                  | 101        |
| 三、具体需求分析:.....               | 101        |
| 第二节 设计.....                  | 101        |

|                                                          |     |
|----------------------------------------------------------|-----|
| 一、数据结构.....                                              | 101 |
| 二、用户界面.....                                              | 102 |
| 第三节 业务逻辑.....                                            | 103 |
| 一、主业务流程.....                                             | 103 |
| 二、个人所得税计算流程.....                                         | 104 |
| 三、操作流程.....                                              | 104 |
| 第四节 建立表和字段.....                                          | 105 |
| 一、自定义表.....                                              | 105 |
| 二、自定义字段.....                                             | 105 |
| 第五节 界面设计.....                                            | 106 |
| 一、界面.....                                                | 106 |
| 第六节 代码.....                                              | 111 |
| 一、建立与SAP Business One 的连接，取得Application, Company 对象..... | 111 |
| 二、添加计算薪水及个人所得税功能到菜单.....                                 | 114 |
| 三、通过菜单调起计算薪水及个人所得税表单.....                                | 115 |
| 四、表单初始化.....                                             | 115 |
| 五、界面操作.....                                              | 121 |
| 六、保存数据.....                                              | 123 |
| 七、选择公司及退出.....                                           | 127 |
| 第七节 注册及安装程序.....                                         | 128 |
| 第八节 练习.....                                              | 128 |
| 附录 CASE STUDY 源代码.....                                   | 129 |
| 一、SalaryForm.cls.....                                    | 129 |
| 二、SubMain.bas.....                                       | 144 |
| 教材说明.....                                                | 146 |

# 第一章 SAP Business One SDK 中文版概述

## 一、课程设置

### 1. 课程内容及时间分配

SAP Business One SDK 中文版培训课程主要为合作伙伴开设，根据产品的功能划分，共分为五门课。课程以 SDK 的开发功能和流程为主线，讲解 SAP Business One SDK 的主要功能。课程后附相应的习题以供练习，帮助用户了解开发原理和实现过程。

| 课程名称          | 内容                                                             | 课时   |
|---------------|----------------------------------------------------------------|------|
| 概述            | 课程总体介绍                                                         | 4 小时 |
| 用户自定义<br>表和字段 | 总体介绍<br>用户自定义表、字段描述                                            | 4 小时 |
| DI API        | DI_API 概览<br>业务数据对象<br>Working with XML<br>事务<br>DI_API 高级特性   | 8 小时 |
| UI API        | UI API 概述<br>UI API 体系介绍<br>UI API 对象<br>应用同步处理<br>UI API 高级特性 | 8 小时 |
| 注册和安装         | 总体介绍<br>Add on 注册<br>安装程序                                      | 4 小时 |
| Case Study    | 需求分析<br>设计<br>业务逻辑<br>建立表和字段<br>界面设计<br>代码<br>注册及安装程序          |      |

### 2. 课程要求

本课程参加人员的基本要求：基础的计算机知识和 SAP Business One 的业务知识，熟悉 Microsoft 操作系统和 MS Office 工具，具有基本的 MS Visual Studio 或 Java 开发知识和使用经验。

### 3. 课程目标

通过课程的学习，能够熟练使用 SAP Business One SDK 开发 SAP Business One 产品中欠缺的功能模块，增强产品模块功能，创建 SAP Business One 与第三方应用程序之间的接口，完成对 SAP Business One 的增值开发。

## 二、SAP Business One SDK 简介

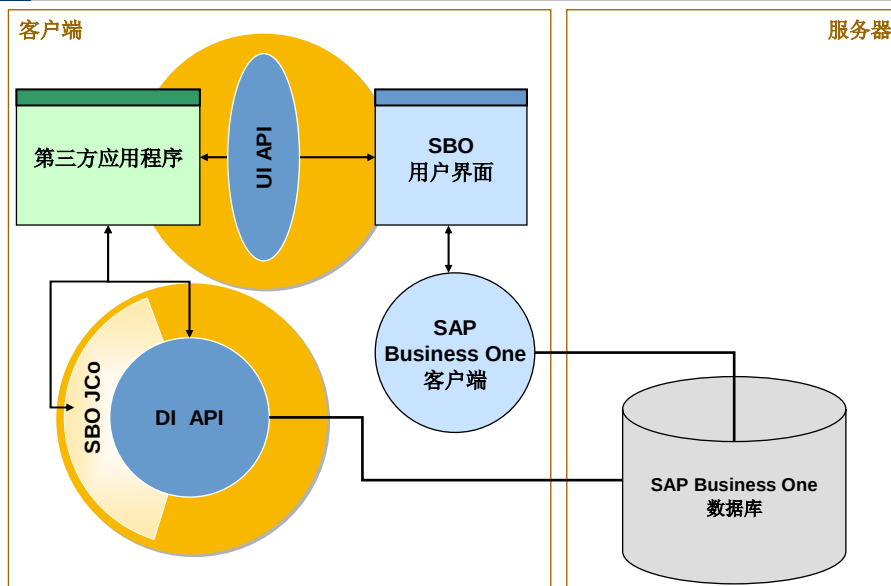
SAP Business One 是一个可靠、易用的软件解决方案，它包含了中小型企业业务管理和运作方面的所有功能模块，并内含了许多可供企业进行定制和配置的特性选项。SAP Business One SDK 是一种开放而且灵活的开发工具包，能让合作伙伴或客户在低成本条件下进一步扩展 SAP Business One 的产品功能，并可以与外部的行业解决方案集成。

SDK 的推出，使得您可以针对您的特殊需求来进一步开发定制企业解决方案。SAP Business One SDK 为软件开发者提供了一套全面的开发工具，允许您按照企业的特殊需求来定制并自动处理 SAP Business One 中的业务流程，并可以与外部的应用程序进行集成。在技术上它基于标准的与 Microsoft .NET, Microsoft COM, 和 Java 技术相兼容的平台，可以让您的开发团队在短时间内开发出增值的解决方案服务于客户，受益于公司。更为详尽的信息请访问：<http://service.sap.com/smb/development/sdk> 和 <http://sdn.sap.com>。

### 1. SDK 的定义

- SDK 的全称是软件开发工具包，它在开放的标准的基础上，内含了不同应用程序的编程接口，使得对 SAP Business One 所有业务对象的访问成为可能
- 您可以在业务数据级别通过 DI API 来访问 SAP Business One，几乎所有在 SAP Business One 客户端中的业务对象都被复制到了 DI API 中，这样就可以被外部的应用程序访问。
- 另一种则是基于用户界面级别的访问: UI API 提供了对应用程序的访问，它可以让您来增加或修改表单，而且它为您提供了修改现有业务逻辑的事件处理接口

#### 什么是 SAP Business One SDK?

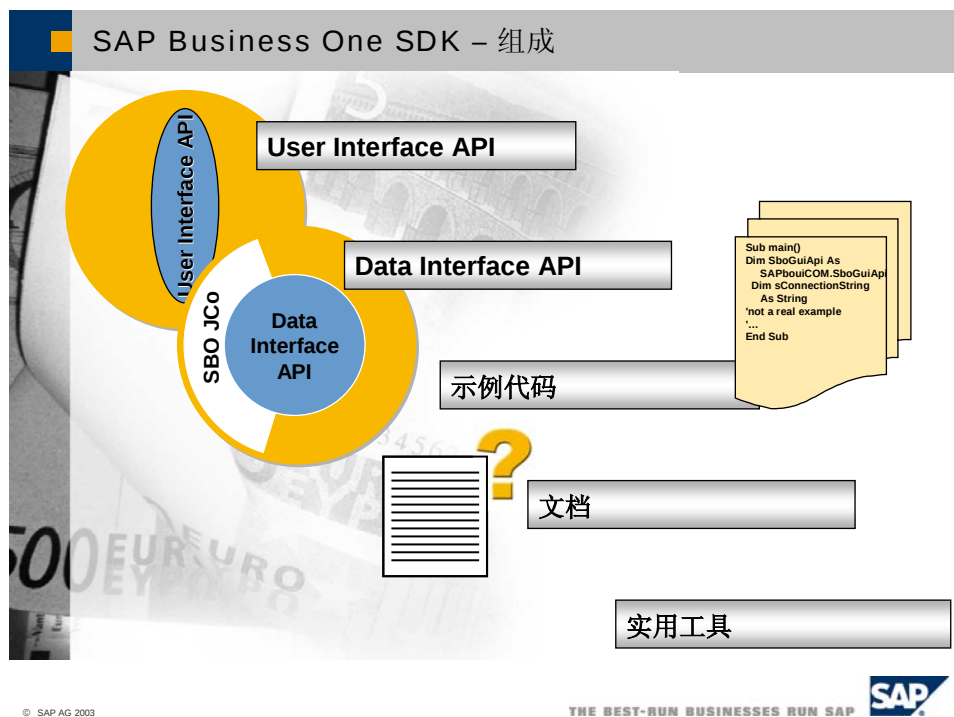


© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP



## 2. SDK 的组成



- SAP Business One SDK 包含以下工具：
  - 应用程序编程接口
  - 示例代码
  - 相关文档
  - 实用工具
- 这些工具为您提供了一个理想和开放的平台，用于扩展 SAP Business One 的业务功能，您可以使用以下的接口或工具来实现 SAP Business One 功能的增强：
  - 基于 Visual Basic, C or C++ 编程的 DI API
  - 基于 Visual Basic, C or C++ 编程的 UI API
  - 基于 Java 编程的 SAP Business One Java Connector (JCo)
  - Screen Painter
- **注意：** SDK 中不包含程序编辑和编译器，为了最大限度地获取灵活性和开放性，您可以自由选择开发环境，在本课程中，我们将用 Microsoft Visual Studio 和 Visual Basic 6 作为编程语言。

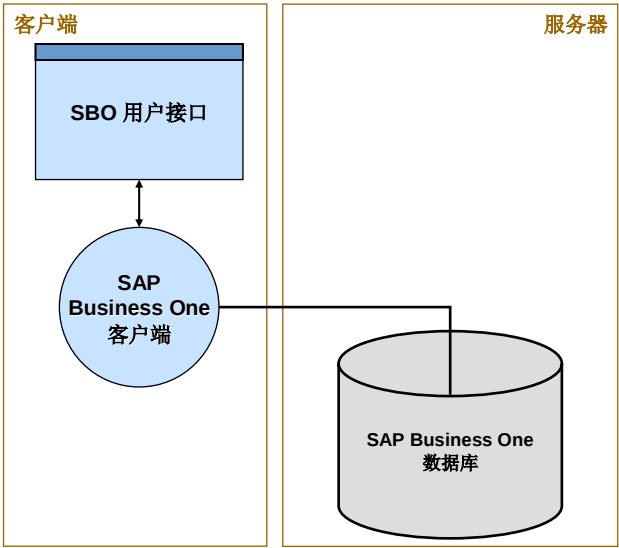
## 3. 推出 SDK 的目的

使用 SAP Business One SDK, 合作伙伴和客户可以扩充和修改 SAP Business One 的功能:

- 特定的行业功能
- 系统欠缺的功能
- 与第三方工具之间的接口

推出 SAP Business One SDK的目的

- 源代码不能访问
  - 您想扩展 SAP Business One的功能?
- ➔ 使用 SDK!



© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP 

4. 技术架构

4.1 SAP Business One

技术架构 – SAP Business One



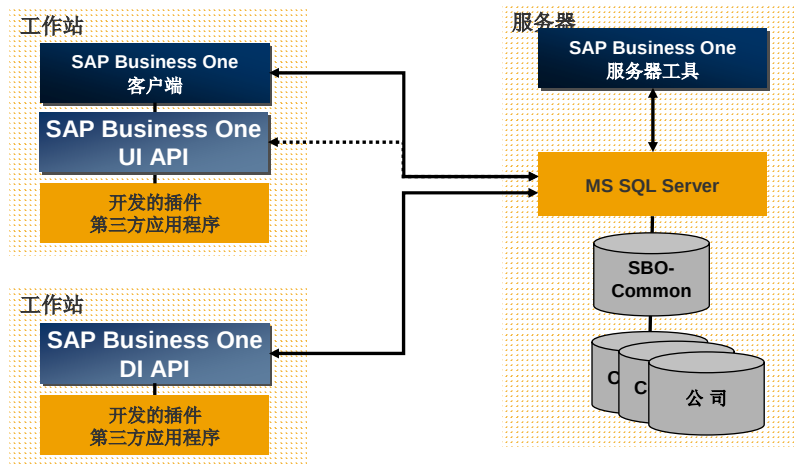
© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP 

4.2 API

## API 架构

UI API 开放了 SAP Business One 客户端的 UI 对象



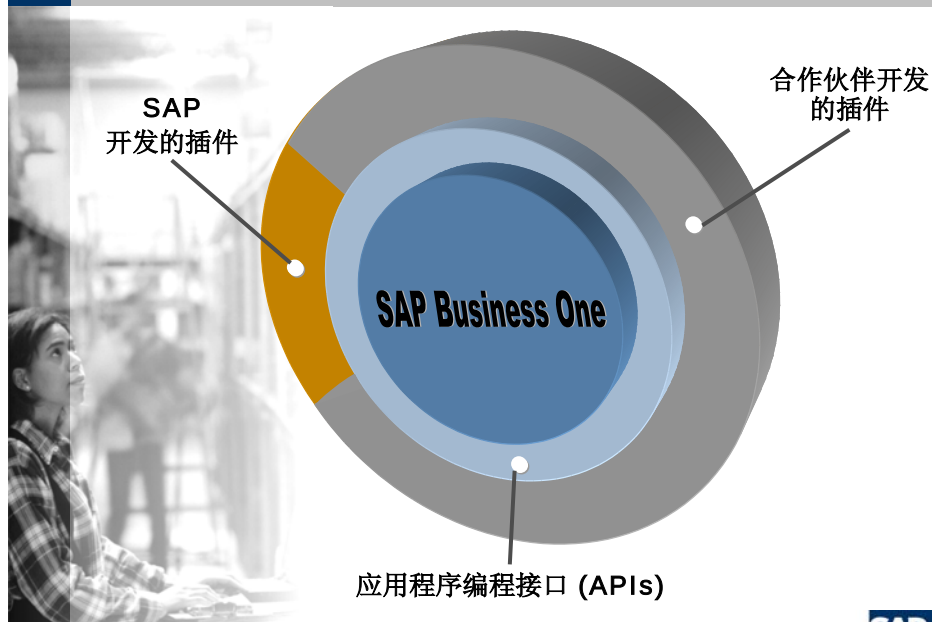
DI API 开放了 SAP Business One 的数据对象，可被用来在客户端独立地访问公司数据库。

© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP

## 4.3 SAP Business One SDK

## 产品架构



© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP

## 5. 适用的对象

SAP Business One 主要适用于工作在以下组织的程序开发人员：

- 合作伙伴  
主要用于创建 SAP Business One 与第三方应用程序之间的接口
- 服务和支持提供商（SSPs）

主要用于 SAP Business One 插件的实施或与第三方应用程序创建接口

- 客户

在 SAP Business One 项目实施过程中, 用于改变屏幕的布局设计等

## 6. 推荐运行环境

### 6.1 SAP Business One的推荐配置

|      | 服务器                                             | 工作站                                  |
|------|-------------------------------------------------|--------------------------------------|
| 软件环境 | NT 4 Server / Windows 2000 Server / MS SQL 2000 | Windows 98<br>Windows 2000 / NT / XP |
| CPU  | Pentium III 600 MHz 以上                          | Pentium III 300 MHz                  |
| 内存   | 512 MB 以上                                       | 128 MB RAM                           |
| 硬盘空间 | 2 GB 以上                                         | 400 MB                               |

### 6.2 支持的平台

- Microsoft Windows 2000
- Microsoft Windows XP

强烈推荐您安装操作平台的最新版补丁程序.

### 6.3 SAP Business One SDK 对系统的要求如下:

- SAP Business One 的系统要求
- 您使用的开发环境对系统的要求
- 用 UI\_API 来运行 5 个解决方案的插件程序要求至少256M 内存
- 运行多于 5 个解决方案的插件要求至少 512M 的内存.

SDK 对系统要求的详情请查看 Note: 628155

## 7. 支持的开发语言

SDK 所适用的开发语言必须是面向 MS COM 技术的编程语言, SAP 发布的开发环境如下:

- Microsoft Visual Studio 6.0 and .NET for Visual Basic
- Microsoft Visual Studio 6.0 and 7.0 (.NET) for C/C++

其它面向 COM 的编程语言也有可能可以工作, 但 SAP 目前还不提供针对这些开发环境的支持, 详情请查看 Note: 615987

## 8. 支持

对于合作伙伴和客户所需的支持, 可以通过访问 SAP 的 **Service Marketplace** 网站 (/sbosupport) 以 CSN Message 的形式提交 SAP 以获取支持; 对于将来的 API 中没有但由又要使用的特性, 合作伙伴可以以 DRQ Message 的形式提交 SAP, SDK 的 PM 收到这个 DRQ Message 后会在该版本内容基础上做进一步处理。

### 三、SAP Business One SDK 的版本

SDK 包括 Implementation、Runtime 和 Development 三个版本，其中：

1. **Runtime Version:** 只包含用户用 DI\_API 开发的应用程序所需的 DLL、Exe 和 Jar 文件和一些工具，主要用来让 SAP 的 Partner 来运行已经生成的 Add-on，其开发功能将在 SBO6.7 的版本中受限；
2. **Implementation Version:** 实施版被客户/合作伙伴用于：
  - 对于单个的 SAP Business One 实施，用于改变屏幕的布局和设计（为此主要使用 UI\_API 和 SCG）
  - 针对单个的客户设置，添加、隐藏、移动、改变按钮/字段和屏幕
  - 改变屏幕颜色
  - 示例：在用户角色和权限的基础上隐藏屏幕字段
  - 不能编写改变 SAP Business One 的业务逻辑的源代码
  - 在实施版上所做的开发不允许通过许可销售给第三方

注意：以上两个版本的 SDK 对已经购买了 SBO 产品 License 的用户是免费的，将内含在后续的 SBO 产品 CD 中。

3. **Development Version:** 该版本包含附加的开发工具、例子和相关文档，用于开发增强 SBO 产品功能的 Add-on，与以上两个版本最大的区别就在于：如果 Partner 或客户购买了该版本的 SDK，那么他们就有权力将开发完成的 Solution 转售给第三方。版权归 Partner 所有。

Development Version 被合作伙伴/客户用来：

- 创建与 SAP Business One 接口（主要应用 DI\_API）的自有的行业解决方案。  
SAP Business One 被用来与第三方的应用程序进行数据交换  
示例：POS 机系统集成，人力资源服务集成，与移动设备之间的数据同步
- 通过使用 SDK（主要使用 UI\_API）的工具和接口来改变 SAP Business One 的程序逻辑和业务处理流程(特性和模块)。  
示例：电子银行的解决方案，客户服务解决方案
- 通过许可协议销售已创建的行业解决方案或第三方的插件程序

### 四、SAP Business One SDK 的兼容性

对每一个版本的 SAP Business One 都有相应的 SDK 版本与之对应：

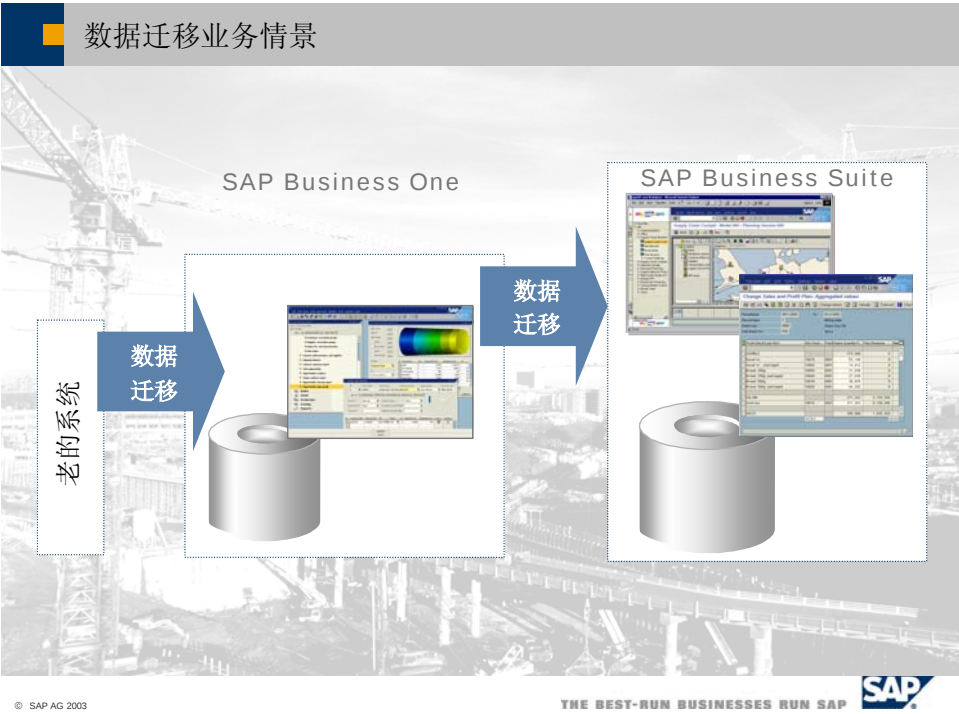
X 版本的 SDK 可以运行在 X 版或更高的（如 X+1）版本的 SAP Business One 下。  
新版本的 SDK 可以支持以前所有版本的接口和功能：

- 从技术角度上来讲，一个由 X 版本的 SDK 开发的应用程序无需修改也可以运行在 X+1 版的 SDK 的接口和组件下。
  - 面向业务对象的新特性和功能仅在匹配的 SAP Business One 版本下才是可用的。
  - 一个用 SDK 开发的应用程序无需修改即可要运行在高版本的 SDK 和 SAP Business One 下。
  - 有以下两个例外情况：
    - 安装了 SP0 的 6.01 版的 SDK 仅可以运行 SAP Business One 6.01 SP0 下
    - 对 SAP Business One 6.2 SP0，没有可用的 SDK 版本
- 详情请查看 Note: **626733**

|                              |           | SDK<br>版本     | 6.01 |     | 6.2 |     | 6.5 |     |
|------------------------------|-----------|---------------|------|-----|-----|-----|-----|-----|
| SAP<br>Business<br>One<br>版本 | 支持<br>软件包 | 支持<br>软<br>件包 | SP0  | SP1 | SP1 | SP2 | SP0 | SP1 |
| 6.01                         | SP0       |               | OK   | /   | /   | /   | /   | /   |
|                              | SP1       |               | /    | OK  | /   | /   | /   | /   |
| 6.2                          | SP0       |               | /    | /   | /   | /   | /   | /   |
|                              | SP1       |               | /    | OK  | OK  | /   | /   | /   |
|                              | SP2       |               |      | OK  | OK  | OK  | /   | /   |
| 6..5                         | SP0       |               | /    | OK  | OK  | OK  | OK  | /   |
|                              | SP1       |               | /    | OK  | OK  | OK  | OK  | OK  |

五、SAP Business One SDK 应用示例：Data Transfer Workbench

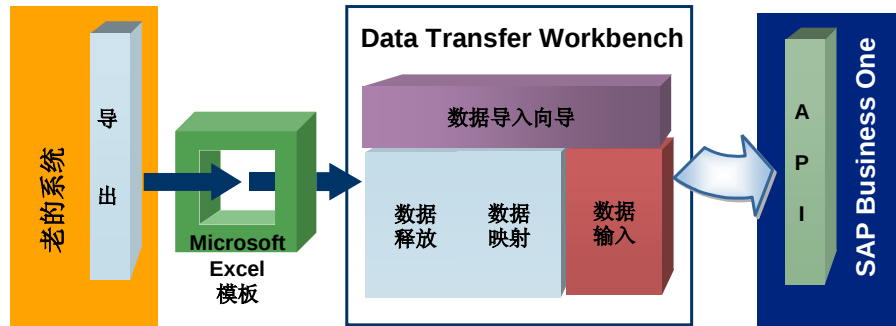
1. 数据迁移的业务情景



随着企业规模和业务的增长，客户的需求也会随着增长，为了满足企业数据迁移需求，SAP 提供了一个简单、实用的工具，使用它可以从老的系统将数据迁移到 SAP Business One，或从 SAP Business One 系统将数据迁移到 mySAP 的商务套件。

## 2. Data Transfer Workbench

### Data Transfer Workbench



© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP 

Data Transfer Workbench 可以将老的系统中的数据迁移到 SAP Business One 中，该工具不是 SAP Business One 的组件，所以必须单独安装；在 DTW 中，数据迁移主要分数据释放、数据映射和数据输入三个主要步骤，主要应用 DI API 以保证数据的一致性；6.5 版本的 DTW 还加入了 Excel 模板、数据输入向导和处理日志管理等新功能。

## 3. SAP 在 SDK6.5 上开发的其它插件

### SAP 在 SDK 6.5 上开发的其它插件

- HR Integration Toolkit
- Internet Sales for SAP Business One
- Support Tools, including Support Desk and Early Watch Alert
- File Format for Electronic Payment (Payment Engine)
- Fixed Assets (DE, AT, CH)
- Intrastat
- Stampit® (German only)
- MS Outlook Integration
- Business Configuration Sets (BC Sets)

=> see section „Solutions“ in the SMB area on SAP Service Marketplace

© SAP AG 2003

THE BEST-RUN BUSINESSES RUN SAP 

## 六、命名规则

### 1. 目的和描述

避免合作伙伴或软件供应商使用 SDK 开发的解决方案的命名冲突，namespace 定义了解决方案的所有命名规则，所以，命名前缀就是我们所说的 namespace，在 SAP 中，该命名必须是唯一的，通常情况下用 “\_” 来隔开命名前缀，如 “ROOT\_myname”，当然，在一个组织既定确保命名唯一性的命名规则内，一个命名可以被用于多个解决方案，namespace 是一套用于设置表单、菜单、用户自定义表格和字段命名唯一性的工具。

### 2. 前提条件

您所在的公司购买了 Development version 的 SDK，否则，您提交的申请将不会通过合同检查从而 SAP 也会拒绝您的命名申请。

### 3. 处理流程

新的namespace处理流程已经与 2004 年 6 月 17 日正式启用，允许用户通过 Service Marketplace (<http://service.sap.com/namespaces>) 进行namespace的预留，并可以快速和实时地处理命名申请。详情请查看Note: 647987

1)、进入 Service Marketplace，创建 namespace 申请。命名前缀必须遵循 /ROOT/ 的语法输入，ROOT 必须包含文字和数字字符且必须以一个字母打头，其最短不得少于 3 个字符，最多不超过 8 个字符。

2)、输入完成后点击“保存”，若该命名已经被预留，则系统会报错，需要改名完成后续操作；若该命名未被预留则可以被预留给您的公司。

3)、等待来自 SAP 的接受反馈

4)、在 SAP Service Marketplace 网站上预留已被 SAP 接受的 namespace。

5)、您可以正式使用为 SAP Business One 解决方案申请到的命名。

### 4. 注意事项

命名过程中，采用以下特殊对象的命名可能会导致一些问题，需要着重考虑：

- ☐ 数据库表和索引
- ☐ 软件包分发时所含的文件(dll.s, 可执行文件, ocx, 源文件等)
- ☐ 项目、模块、类和属性的名称
- ☐ 表单和屏幕的ID号
- ☐ 项目、模块和类的文件名
- ☐ SAP Business One 中的用户自定义字段

## 第二章 用户自定义表和字段

在 SBO SDK 开发中，免不了要用户自定义数据表、自定义字段，此时，我们可以通过 sqlserver 企业管理器创建数据库表，但是我们所创建的表，在数据 API 接口里的标准业务对象不能访问添加的数据库表、字段，这样，我们只能通过 RecordSet 对象来访问。

在 SAP UI\_API 中，我们还可以通过 **User Defined Fields & Tables** 方法来创建自定义表和自段。这样用户自定义的数据库表在 SAP 数据字典里是可见的。在 SDK 6.2 中，用户自定义表字符长度是 3 位，在 SDK 6.5 用户定义的表名字的长度被给予 20 个字符。这个的长度允许你实现名字前缀。用户自定义表的优势是他们完全统一到 SAP 中。因此，SAP 推荐你使用自定义表的办法解决你特殊的数据库表的问题。

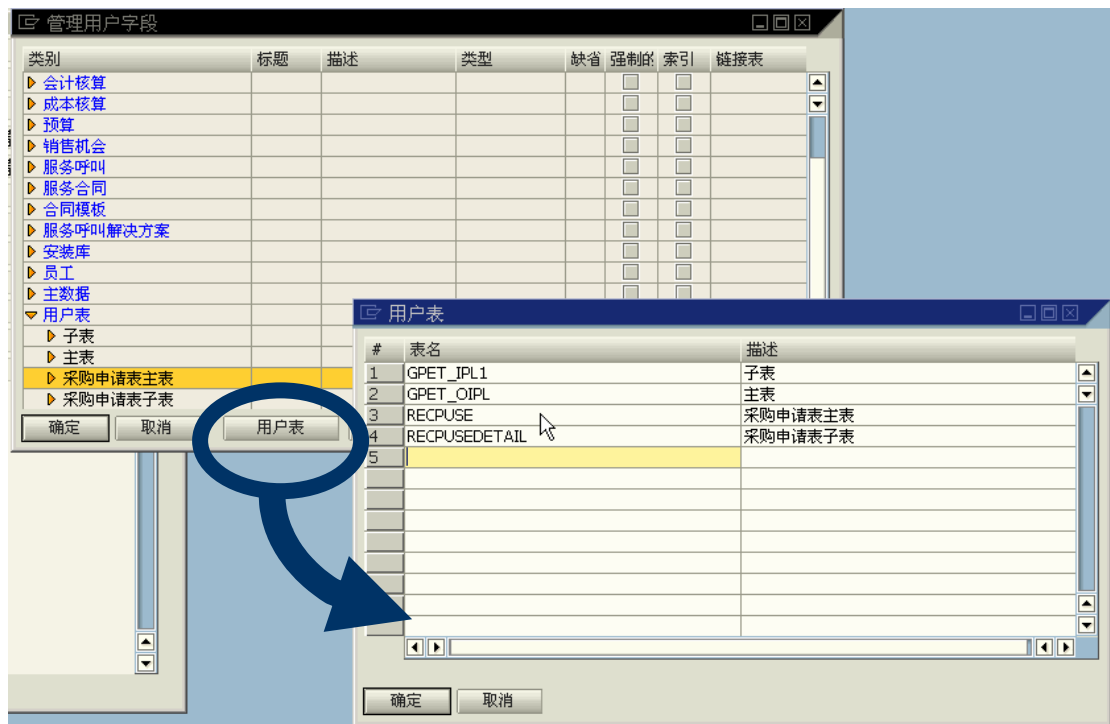
### 第一节 用户自定义表、字段描述

#### 一、创建自定义表

创建用户自定义表有 2 种方法，第一种方法我们可以通过 SBO 应用程序提供的自定义表功能来创建，第二种方法，我们可以通过 DI\_API 提供的 UserTablesMD 对象来创建。

##### 1. 方法一：

我们通过 SBO 产品的自定义表功能创建表。首先登陆 SBO 系统，然后选择：工具→管理用户字段，进入如下界面：，然后选择**用户表**，在弹出的界面中输入表名、表描述，然后**确定**即可创建表。注意：我们用户自定义的的表，在 SqlServer 企业管理器中察看的时候，都自动在表名前面添加@了字符。这也是用户自定义表和系统表的唯一区别。



## 2. 方法二:

我们可以通过 DI\_API 中提供的 UserTablesMD 对象来创建表。

### 1: UserTablesMD 对象介绍

|                  |                                                          |
|------------------|----------------------------------------------------------|
| GetByKey ( )     | 返回 True Or False , 检索是否存在给定的表, 如果存在返回 True, 否则, 返回 False |
| TableName        | 设置表名                                                     |
| TableDescription | 设置表的描述性文字                                                |
| Add              | 添加表到数据库中。                                                |
| Remove           | 删除表                                                      |

### 2: 代码:

‘功能: 添加自定义表

‘参数说明: fstrTable 设置自定义表名, fstrTableDesc 表的描述文字

‘其他说明: 首先 oCompany 定义为 SAPbobsCOM.Company, 且已经连接到数据库

```
private oCompany As SAPbobsCOM.Company

Private Function AddUserTable(fstrTable As String, fstrTableDesc As String) As Boolean
```

‘定义 UserTablesMD 对象。

```
Dim oUserTablesMD As SAPbobsCOM.UserTablesMD
```

```
Dim lRetCode As Long
```

```
Set oUserTablesMD = oCompany.GetBusinessObject(oUserTables)
```

```
If Not oUserTablesMD.GetByKey(Trim(fstrTable)) Then
```

```
oUserTablesMD.TableName = fstrUserTablesMDTable
```

```
oUserTablesMD.TableDescription = fstrTableDesc
```

```

RetCode = oUserTablesMD.Add
End If
If lRetCode <> 0 Then
    oCompany.GetLastError lErrCode, sErrMsg
    MsgBox sErrMsg
    AddUserTable = False
    Exit Function
End If
Set oUserTablesMD = Nothing
AddUserTable = True
End Function

```

## 二、添加自定义字段

添加自定义字段也有 2 种方法,第一种方法我们可以通过 SBO 应用程序提供的自定义字段功能来创建,第二种方法,我们可以通过 DI\_API 提供的 UserFieldsMD 对象来创建。自定义字段可以添加到系统原有的表中,也可以添加到用户自定义的表中。

### 1. 方法一:

我们通过 SBO 产品的自定义表功能创建表。首先登陆 SBO 系统,然后选择:工具→管理用户字段,进入如下界面:



我们在这里必须选择类别表,然后添加按钮为可选择状态,我们在这里既可以在系统表中添加自段,也可以在自定义表中添加,只要选择对应的表名即可,然后添加。弹出如下界面,在里面输入相应信息,然后**确定**创建自定义自段。

名词解释：

标题：字段的名称。

描述：字段的说明性文字。

类型：字段类型。

长度：如果字段类型为字母数字，则需要设置长度。

结构：一些特殊设置，比如要设置电话格式、多行文本录入等。

设置字段有效值：设置此项，那么此字段的内容必须是预设值中的内容。

设置字段默认值：给字段设置默认值。

必须字段：设置字段为必填项。

创建索引：设置此字段索引。

设置链接表：设置此字段链接的表

注意：我们用户自定义的字段，在 **SqlServer** 企业管理器中察看的时候，都自动在字段名前面添加 **U\_** 字符。这也是用户自定义字段和系统字段的区别。

## 2. 方法二：

我们可以通过 **DI\_API** 中提供的 **UserFieldsMD** 对象来创建表。

### 1: UserFieldsMD 主要对象介绍

|              |             |
|--------------|-------------|
| Name         | 设置字段名称。     |
| TableName    | 设置添加字段的表名。  |
| Description  | 设置字段的描述性文字。 |
| Add          | 添加表到数据库中。   |
| Type         | 设置字段类型      |
| DefaultValue | 设置字段的缺省值    |
| EditSize     | 可编辑长度       |

Remove          删除字段

2: 字段类型/结构描述:

| 字段类型/结构描述 |                            |
|-----------|----------------------------|
| 字母数字      | 地址、电话、文本                   |
| 数字        | 数字                         |
| 日期/时间     | 日期/时间                      |
| 单位与总计     | 价格, 金额, 度量, 数量, 百分比、<br>汇率 |
| 概览        | 连接, 图像                     |

3: 代码:

‘功能: 添加自定义字段

‘参数说明: TableName 设置表名, FieldName 字段名, FieldDesc 字段的描述文字

‘FieldType 字段类型 (参照类型说明), FieldSubType 字段结构设置, FieldSize 设置字段的长度

‘其他说明: 首先 oCompany 定义为 SAPbobsCOM.Company, 且已经连接到数据库  
private oCompany As SAPbobsCOM.Company

```
Private Function AddUserField(TableName As String, FieldName As String, FieldDesc
As String, FieldType As SAPbobsCOM.BoFieldTypes, Optional FieldSubType As
SAPbobsCOM.BoFldSubTypes, Optional FieldSize As Long) As Boolean
```

‘定义自定义自段对象变量

Dim oUserFieldsMD As SAPbobsCOM.UserFieldsMD

‘判断字段是否存在

If IsFieldsExist(TableName, FieldName) Then

AddUserField = True

Set oUserTablesMD = Nothing

Else

‘初始化自定义对象

Set oUserFieldsMD = oCompany.GetBusinessObject(oUserFields)

oUserFieldsMD.TableName = TableName

oUserFieldsMD.Name = FieldName

oUserFieldsMD.Description = FieldDesc

oUserFieldsMD.Type = FieldType

If FieldSubType Then oUserFieldsMD.SubType = FieldSubType

If FieldSize Then oUserFieldsMD.EditSize = FieldSize

```

' // 添加字段到表 中、
lRetCode = oUserFieldsMD.Add

' // 检查错误，是否正确的添加了字段，如果正确，则返回 0
If lRetCode <> 0 Then
    oCompany.GetLastError lErrCode, sErrMsg
    MsgBox sErrMsg
    AddUserField = False
    Set oUserTablesMD = Nothing
Else
    AddUserField = True
End If

End If
Set oUserTablesMD = Nothing
End Function

'IsFielsExist 函数判断表中是否存在自定义字段
'其中 Recordset、oCompany 对象在其他章节有介绍。
Private Function IsFielsExist(TableID As String, FieldName As String) As
Boolean
    Dim oRecordset As SAPbobsCOM.Recordset
    Set oRecordset = oCompany.GetBusinessObject(BoRecordset)
    oRecordset.DoQuery ("select * From CUFD " & _
        "where AliasID = '" & FieldName & "' and" & _
        "      TableID = '@' & TableID & "'")

    If oRecordset.EOF Then
        IsFielsExist = False
    Else
        IsFielsExist = True
    End If

    Set oRecordset = Nothing
End Function

```

### 三、添加数据

向表中添加数据也有 2 种方法，第一种方法我们可以通过 SBO 应用程序提供的填写用户表添加数据，第二种方法，我们可以通过 DI\_API 提供的 UserTable 对象来添加数据。



‘其他说明：首先 pCompany 定义为 SAPbobsCOM.Company，且已经连接到数据库

```

Private pCompany As SAPbobsCOM.Company
Private Sub cmdInsertRecord_Click()
    Dim pRecord As SAPbobsCOM.UserTable
    ‘找到自定义表，初始化 pRecord 变量
    Set pRecord = pCompany.UserTables.Item(CStr(txtTableName))
    Dim i As Integer
    ‘给变量赋值，注意：code、name 必须要赋值。
    pRecord.Code = txtUCode
    pRecord.Name = txtUName
    pRecord.UserFields("U_Make").Value = Trim(txtFieldValue(0))
    pRecord.UserFields("U_Model").Value = Trim(txtFieldValue(1))
    pRecord.UserFields("U_Color").Value = Trim(txtFieldValue(2))
    pRecord.UserFields("U_Year").Value = Trim(txtFieldValue(3))
    ‘向表中添加数据
    pRecord.Add
    pRecord.Remove
    ‘错误处理，如果正确的添加数据，则返回 0
    pCompany.GetLastError lErrCode, sErrMsg
    If lErrCode <> 0 Then
        MsgBox (sErrMsg)
    Else
        MsgBox ("The record has been added.")
    End If
End Sub

```

## 2. 修改数据

‘功能：修改自定义表中数据

‘参数说明：txtTableName 是文本框，设置自定义表名，txtFieldValue（）是一个文本框数组，传递字段新的变量值，txtUCode 是 Code 字段的变量值，

‘其他说明：首先 pCompany 定义为 SAPbobsCOM.Company，且已经连接到数据库

```

Private pCompany As SAPbobsCOM.Company
Private Sub cmdUpdateRecord_Click()
    ‘定义对象变量
    Dim pRecord As UserTable
    ‘找到自定义表，初始化 pRecord 变量
    Set pRecord = pCompany.UserTables.Item(CStr(txtTableName))
    ‘以 code 字段的值来检索所要修改的记录（因为 code 是主键）
    If pRecord.GetByKey(txtUCode) Then
        pRecord.UserFields("U_Make").Value = CStr(txtFieldValue(0))
        pRecord.UserFields("U_Model").Value = CStr(txtFieldValue(1))
        pRecord.UserFields("U_Color").Value = CStr(txtFieldValue(2))
        pRecord.UserFields("U_Year").Value = CStr(txtFieldValue(3))
    End If
End Sub

```

```

        pRecord.Update
        MsgBox ("修改成功！")
    Else
        MsgBox ("没有找到所要修改的记录！.")
    End If

```

```
End Sub
```

### 3. 删除数据

‘功能：删除自定义表中数据

‘参数说明：txtTableName 是文本框，设置自定表名，txtUCode Code 字段的变量值

‘其他说明：首先 pCompany 定义为 SAPbobsCOM.Company，且已经连接到数据库

```

Private pCompany As SAPbobsCOM.Company
Private Sub cmdDelRecord_Click()
    ‘定义对象变量
    Dim pRecord As UserTable
    ‘找到自定义表，初始化 pRecord 变量
    Set pRecord = pCompany.UserTables.Item(CStr(txtTableName))
    ‘以 code 字段的值来检索所要删除的记录（因为 code 是主键）
    If pRecord.GetByKey(txtUCode) Then
        pRecord.Remove
        MsgBox ("删除数据成功！")
    Else
        MsgBox ("没有找到所要删除的记录！.")
    End If

```

```
End Sub
```

### 4. 浏览数据

‘功能：浏览自定义表中数据

‘参数说明：txtTableName 是文本框，设置自定表名，txtFieldValue() 是一个文本框数组，显示字段的值，txtUCode 是 Code 字段的变量值，

‘其他说明：首先 pCompany 定义为 SAPbobsCOM.Company，且已经连接到数据库

```

Private pCompany As SAPbobsCOM.Company
Private Sub cmdGetRecord_Click()
    ‘定义对象变量
    Dim pRecord As UserTable
    ‘找到自定义表，初始化 pRecord 变量
    Set pRecord = pCompany.UserTables.Item(CStr(txtTableName))
    ‘以 code 字段的值来检索所要显示的记录（因为 code 是主键）
    If pRecord.GetByKey(txtUCode) Then
        Dim i As Integer
        ‘通过 UserFields 显示数据
        For i = 0 To 3
            txtFieldValue(i) = pRecord.UserFields(i).Value

```

```
Next i
txtUName = pRecord.Name
Else
txtFieldValue(0) = ""
txtFieldValue(1) = ""
txtFieldValue(2) = ""
txtFieldValue(3) = ""
MsgBox (" 没有找到此条记录! .")
End If
```

End Sub

到此，用户自定义表、字段的内容已全部介绍完毕。

## 第二节 练习

- 1: 在 vb 中新建一个工程，然后自定义表
- 2: 在刚才自定义的表中添加自定义字段
- 3: 在建好的表中添加数据。

## 第三章 DI API

### 第一节 DI API 概述

#### 一、总体介绍

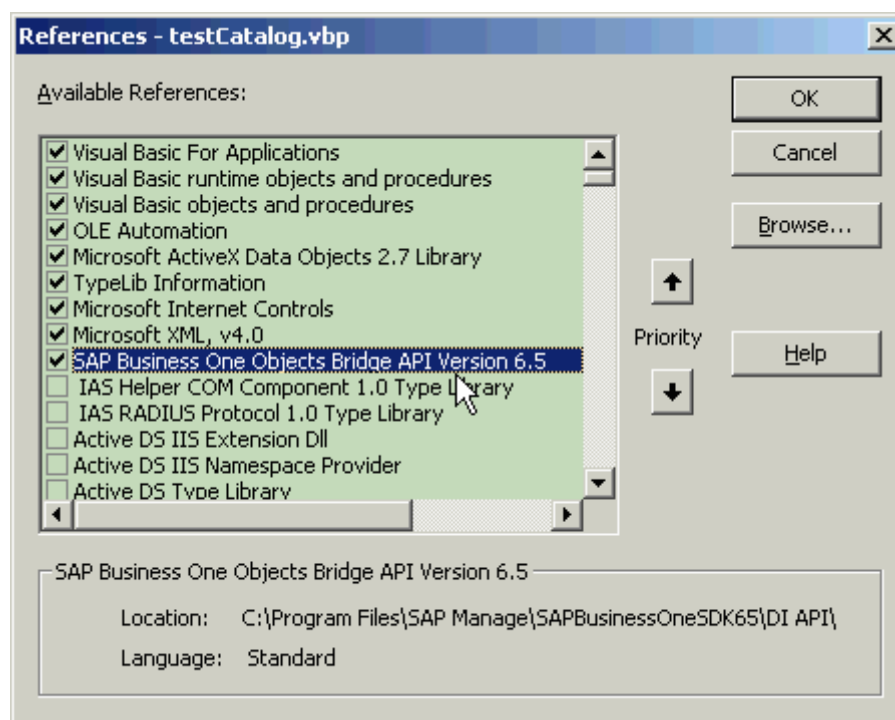
SBO Data Interface API (DI API) 是一组以DLL形式提供，三层结构工作模式的开发工具，目的是合作伙伴提高而且扩充 SBO 和用 SBO 整合外部的解决发案。

SBO DI API 能用来存取SBO 应用程序在数据库层次上的数据，扩充它的功能性，以便和第三方的解决方案连接，扩展 SBO 的功能满足客户的需要。

#### 二、安装

安装 SBO SDK- API, 运行DI API光盘中的 Setup.exe文件，启动安装程序的向导功能，按照向导逐步完成安装即可。安装完毕后，我们在VB工程中的引用对象中能够**SAP Business One Objects Bridge API Version 6.5** 对象，这说明我们已安装成功！

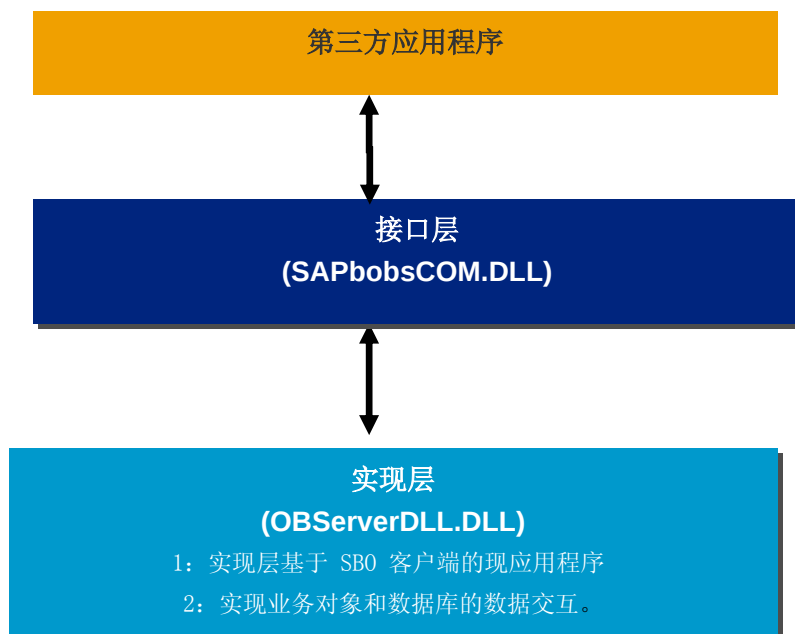
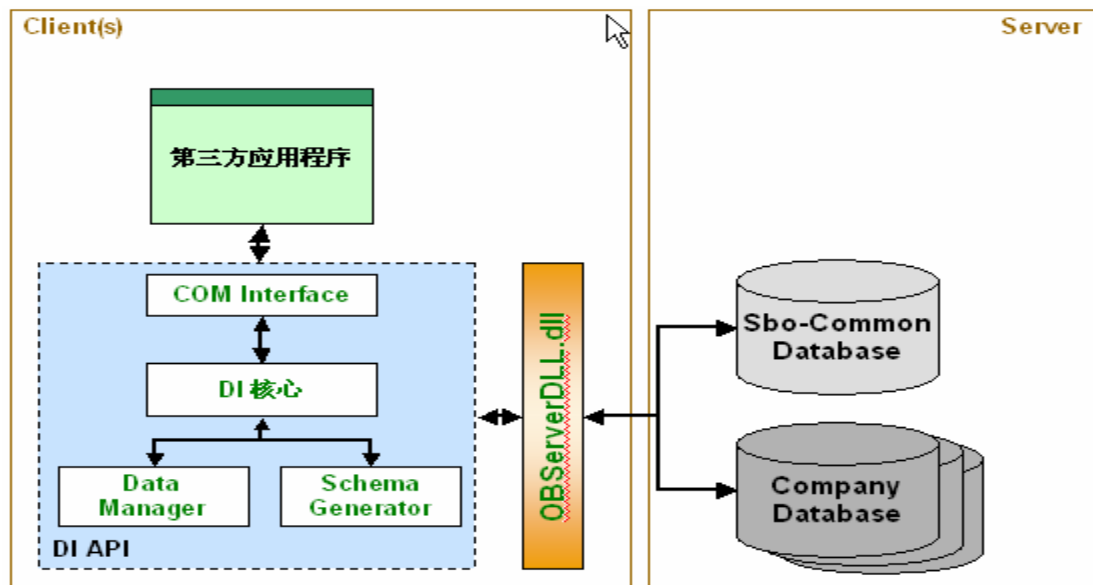
如下图所示：



#### 三、体系结构

在提供关于 DI API 的软件体系结构的明细中:所有的函数功能被包含在一个实现层 (OBSERVERDLL.DLL) 之中。DLL 以 SBO 客户端的现有源码为基础,也就是说, SBO 用户端的业务对象被复制到这个 DLL 中。通过 SAPbobsCOM.DLL 接口能够存取 SBO 客户端对象的方法、属性等。

如下图所示逻辑：

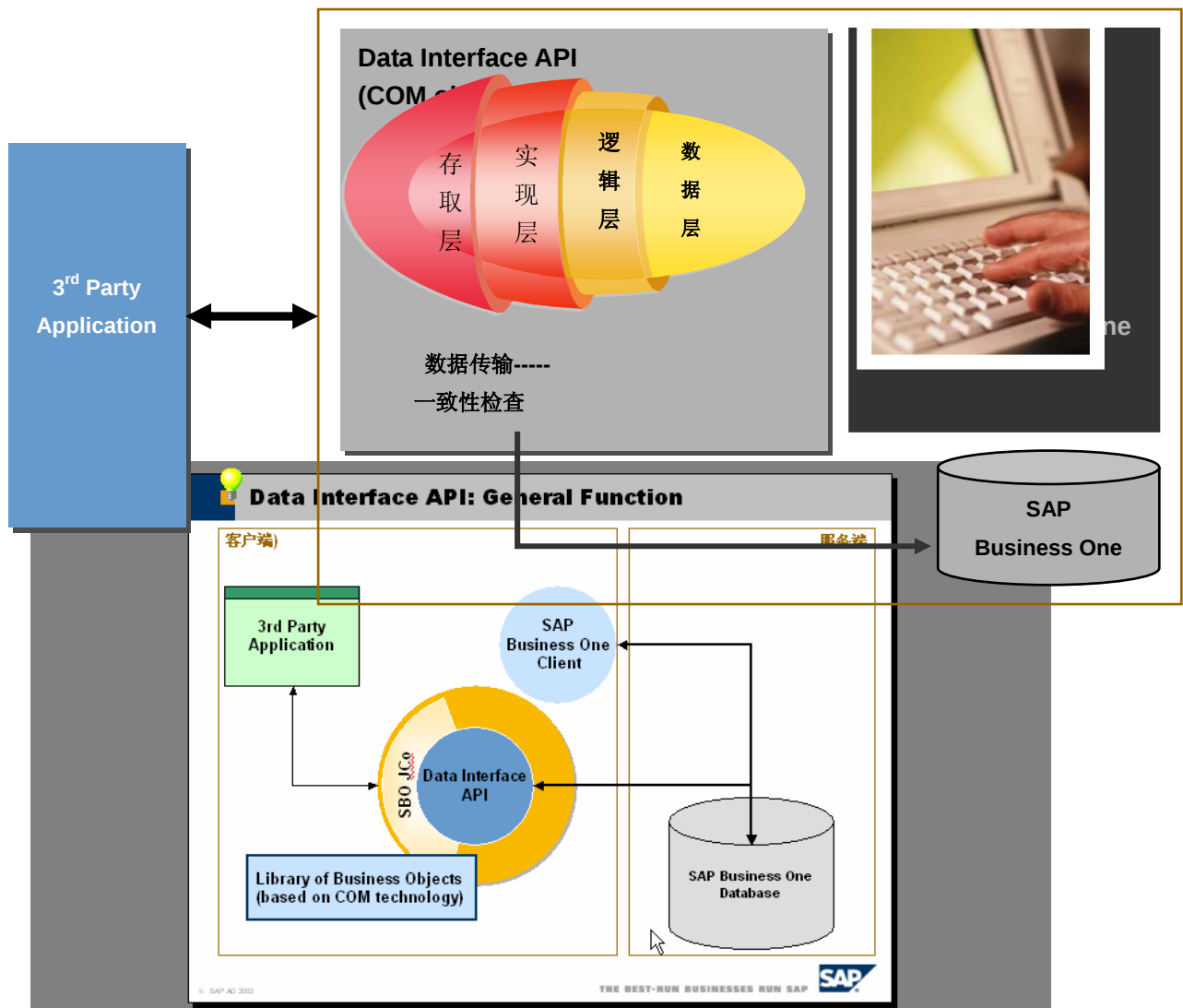


#### 四、存取模式

如下图所示：它主要展示业务对象在结构中的真正对象。 第三方应用程序如何去连接对象。我们所描述的数据是多层的。 这些层使用 COM 技术，是外部的应用程序存取数据

我们第三方应用程序如果要访问SBO 数据，那么必须通过DI\_API来访问，DI\_API 是一个以 COM 技术为基础的业务对象库。 这些业务对象能被一些开发语言（如 Visual Basic, C语言/C语言++, Delphi语言）任何一个支持 COM 技术的工具所利用

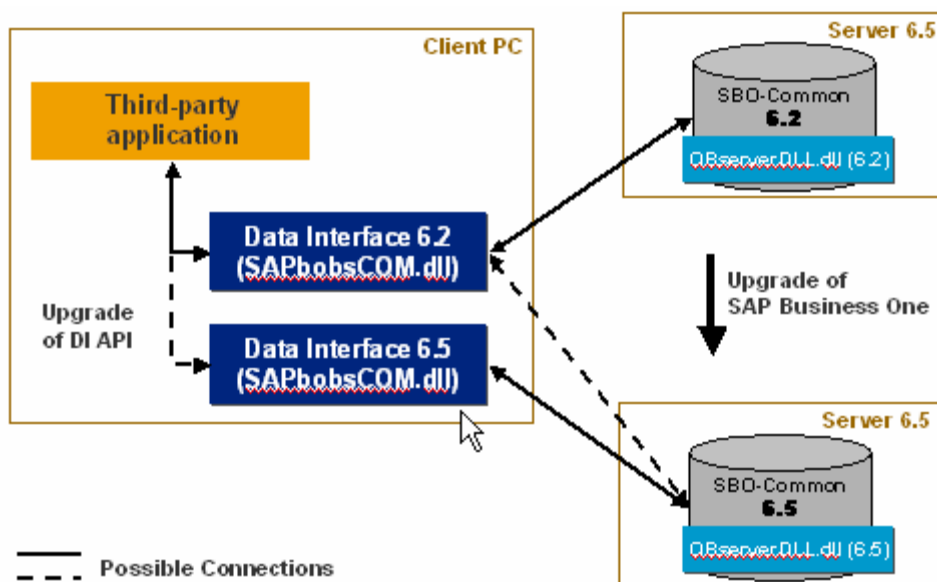
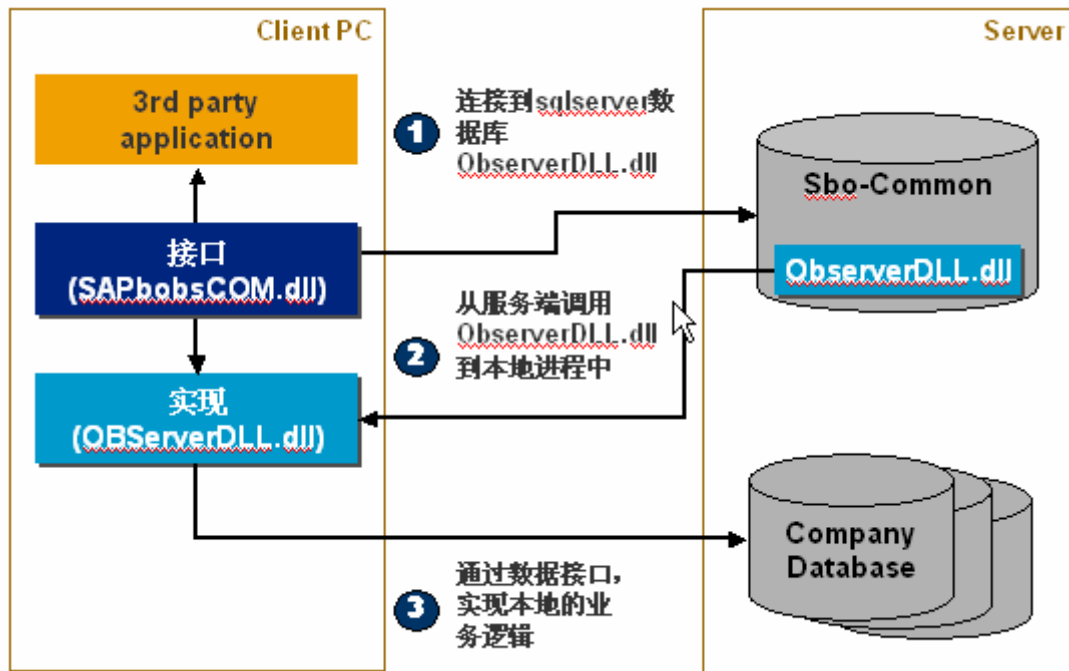
除此之外，DI\_API 也被用Java语言编写为包。 这意味这你也能使用 Java语言存取业务对象。它为我们提供了大多说的访问业务数据的函数、属性、方法。



## 五、版本控制

您不需要关心由于DLL版本的不同，是否您的应用程序不能运行，如果服务器端有一个较新的 OBCServerDLL.dll, 系统会自动地从服务器端下载最新 OBCServerDLL.dll到用户端。

在客户端程序连接到服务器端时，首先进行的是 OBCServerDLL.dll版本的验证，如果服务器端的OBCServerDLL.dll版本较本地的新，那么系统会自动下载到本地，更新本地的 OBCServerDLL.dll，并且将OBCServerDLL.dll文件加载到本地进程中。



## 六、技术支持

如果你有 CSN/OSS 中的账号, 那么你可以到 CSN/OSS 中的 SBO-SDK-DI 创建一个新的 Notes, 我们会有专人为您解答, 否则你可以按如下方式和我们联系:

- Tel +49-1805-343428,
- 网址 <http://service.sap.com/smb>, 或者参照 SAP Note No. 528296

## 第二节 业务数据对象

### 一、Business Object

#### 1、Master Data

##### 1.1、科目-----Chart Of Accounts

描述:

ChartOfAccounts 是 SBO 系统财务模块的一般分类帐科目的对象描述, 通过这个对象, 我们可以操作科目 (OACT) 里的数据, 包括: 增加、修改、删除、检索、格式化保存为 xml 文件等, 同时, 我们可以通过其提供的方法、属性访问科目对象的相关内容。

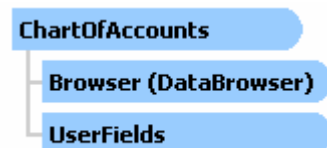
主要方法如下:

|            |                   |
|------------|-------------------|
| + Add      | 增加科目              |
| + GetByKey | 检索以主键为条件的科目记录     |
| + Remove   | 在条件许可的条件下, 删除科目信息 |
| + SaveXML  | 保存科目信息为 XML 文件格式  |
| + Update   | 更新科目信息 (主键字段除外)   |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 ChartOfAccounts 对象的描述。

图例

对象主要模型如下:



代码

下面代码演示了, 我们通过 GetByKey 找到 001 科目, 然后通过 Update 方法修改 ForeignName 得值, 同样我们可以用 add 方法增加科目, 用 Remove 删除科目。

```

Dim RetVal As Long
Dim vChartOfAccounts As SAPbobsCOM.ChartOfAccounts
Set vChartOfAccounts = pCompany.GetBusinessObject(oChartOfAccounts)
' 首先检索是否存在 001 科目, 如果存在, 则更改 ForeignName 的值
IF vChartOfAccounts.GetByKey("001") THEN
    vChartOfAccounts.ForeignName="Cash In Bank"
    vChartOfAccounts.Update
End if
  
```

'检查错误

```

If RetVal <> 0 Then
    vCmp.GetLastError ErrCode, ErrMsg
    MsgBox "Failed to Retrieve the record " & ErrCode & " " & ErrMsg
    Exit Sub
End If

```

### 说明

以上代码只是代码片段，代码认为我们已经定义了 pCompany 变量，并且在 vb 工程里添加了 ui\_api6.5 对象。在下面的例子中，也用同样的方法。

## 1.2、合作伙伴-----Business Partners

### 描述：

BusinessPartners 是 SBO 业务合作伙伴模块的业务伙伴主数据的主要对象，通过这个对象，我们可以操作合作伙伴（OCRD）里的数据，包括：增加、修改、删除、检索、格式化保存为 xml 文件等，同时，我们可以通过其提供的方法、属性访问合作伙伴对象的相关内容。

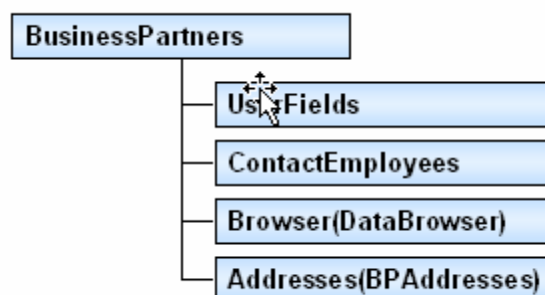
主要方法如下：

|            |                    |
|------------|--------------------|
| + Add      | 增加合作伙伴             |
| + GetByKey | 检索以主键为条件的合作伙伴记录    |
| + Remove   | 在条件许可的条件下，删除合作伙伴信息 |
| + SaveXML  | 保存合作伙伴信息为 XML 文件格式 |
| + Update   | 更新合作伙伴信息（主键字段除外）   |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 BusinessPartners 对象的描述。

### 图例

对象模型如下：



### 代码

下面代码演示了我们如何在合作伙伴业务主数据中添加一个合作伙伴信息。

'首先，假如我们已经连结到了数据库

...'定义变量

```

Dim vBP As SAPbobsCOM.BusinessPartners
'初始化合作伙伴对象
Set vBP = vCompany.GetBusinessObject(oBusinessPartners)
'给对象赋值, 合作伙伴代码、名称、类型 vBP.CardCode = "C08154711"
vBP.CardName = "James Tiberius Kirk"
vBP.CardType = cCustomer
'准备错误处理 Dim lRetCode As Integer
'添加合作伙伴数据到数据库 lRetCode = vBP.Add
If lRetCode <> 0 Then
    MsgBox ("Failed...; Returncode: " + CStr(lRetCode))
End If

```

#### 说明

同样我们还可以通过 `GetByKey()` 方法检索到指定的合作伙伴, 然后通过 `Remove` 删除合作伙伴信息, 同时还可以用 `update` 修改合作伙伴信息, 这里就不再详细叙述这些方法的使用

### 1.3、联系人---ContactEmployees

#### 描述:

`ContactEmployees` 是 SBO 联系人的主要对象, 通过这个对象, 我们可以操作联系人的主要数据, 包括:增加、修改、删除等操作, 同时, 我们可以通过其提供的方法、属性访问联系人对象的相关内容。

主要方法如下:

|                         |                                  |
|-------------------------|----------------------------------|
| <b>+ Add</b>            | <b>增加合作伙伴</b>                    |
| <b>+ SetCurrentline</b> | <b>设定对一个指定对象行设定。 计数器是从 0 开始。</b> |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 `ContactEmployees` 对象的描述。

#### 图例

对象模型如下:



代码下面代码演示了我们如何在添加合作伙伴业务主数据的同时添加联系人信息。

```

'首先, 假如我们已经连结到了数据库
...'定义变量
Dim vCompany As SAPbobsCOM.Company
Dim vBP As SAPbobsCOM.BusinessPartners
Private Sub cmdAddContactEmployee ()

    Dim nErr As Long
    Dim errMsg As String

    On Error GoTo ErrorHandler

```

```

' 初始化 vbp 对象（必须）
Set vBP = vCompany.GetBusinessObject(oBusinessPartners)
    '给合作伙伴对象赋值
vBP.CardCode = "CD1004"
vBP.CardName = "Bob Hope"
vBP.CardType = cCustomer
vBP.CommissionGroupCode = 0
vBP.CommissionPercent = 15
vBP.ContactPerson = "C1"
vBP.Currency = "USD"
vBP.DiscountPercent = 15
vBP.VatLiable = vLiable
vBP.ShippingType = 3
    '添加
Call vBP.ContactEmployees.Add
    '准备添加第一个联系人
Call vBP.ContactEmployees.SetCurrentLine(0)
vBP.ContactEmployees.Name = "John Cash"
vBP.ContactEmployees.Address = "LN"
vBP.ContactEmployees.E_Mail = "c1@abcd.com"
vBP.ContactEmployees.Fax = "8433777778"
vBP.ContactEmployees.MobilePhone = "8388888"
vBP.ContactEmployees.Phone1 = "88880000"
    '准备添加第二个联系人
Call vBP.ContactEmployees.SetCurrentLine(1)
vBP.ContactEmployees.Name = "C2"
vBP.ContactEmployees.Address = "BJ"
vBP.ContactEmployees.E_Mail = "c2@abcd.com"
vBP.ContactEmployees.Fax = "84338"
vBP.ContactEmployees.MobilePhone = "877388888"
vBP.ContactEmployees.Phone1 = "8888300"
    '添加地址
Call vBP.Addresses.Add

Call vBP.Addresses.SetCurrentLine(0)
vBP.Addresses.AddressName = "New Address"
vBP.Addresses.City = "New York"
    '添加如果正确，返回 0
If (0 <> vBP.Add()) Then
    MsgBox ("添加合作伙伴失败")
Else
    MsgBox ("添加合作伙伴成功")
End If

```

```
Exit Sub
ErrorHandler:
    MsgBox ("Exception: " + Err.Description)

End Sub
```

## 说明

### 1.4、物料-----Items

#### 描述:

Items 是 SBO 整个系统的物料数据的主要对象，通过这个对象，我们可以操作物料（OITM）里的数据，包括:增加、修改、删除、检索、格式化保存为 xml 文件等，同时，我们可以通过其提供的方法、属性访问物料对象的相关内容。

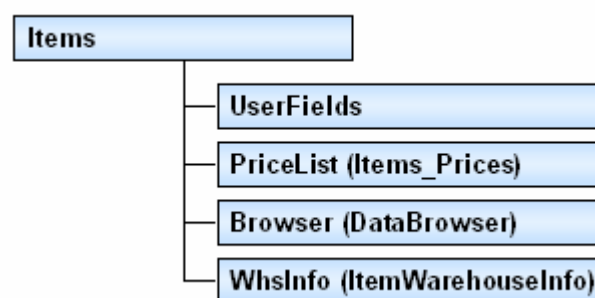
主要方法如下:

|            |                  |
|------------|------------------|
| + Add      | 增加物料             |
| + GetByKey | 检索以主键为条件的物料记录    |
| + Remove   | 在条件许可的条件下，删除物料信息 |
| + SaveXML  | 保存物料信息为 XML 文件格式 |
| + Update   | 更新物料信息（主键字段除外）   |
| + Cancel   | 取消对象的操作          |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 Items 对象的描述。

#### 图例

对象模型如下:



#### 代码

下面代码演示了我们如何在物料主数据中添加一个物料信息。

```
...' 定义变量
Dim vItem As SAPbobsCOM.Items' 初始化物料对象
' vCmp 是 company 对象，并且已和数据库建立连结
Set vItem = vCmp.GetBusinessObject(oItems)
' 查找物料编码=HU1001 的物料，如果找到则返回 TRUE
```

```

If vItem.GetByKey("HU1001") = True Then
    ' 更新物料名称为 Desktop Printer
    vItem.ItemName = "Desktop Printer"    vItem.Update
Else
    ' 否则没有找到匹配的数据
End If

```

### 说明

同样我们还可以通过 **GetByKey()** 方法检索到指定的物料，然后通过 **Remove** 删除物料信息，同样可以 **Add** 方法添加物料信息，这里就不再详细叙述这些方法的使用

## 1.5、物料:价格-----Items\_Prices

### 描述:

**Items\_Prices** 是 SBO 系统中物料的一个价格对象，通过这个对象，我们可以操作物料关于价格的信息，同时，我们可以通过其提供的方法、属性访问 **Items\_Prices** 对象的相关内容。

主要方法如下:

+ **SetCurrentline**      设定对一个指定对象行设定。计数器是从 0 开始。

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 **Items\_Prices** 对象的描述。

### 代码

下面代码演示了我们如何在物料主数据中添加一个价格信息。

```

... ' 定义变量
Dim vCompany As SAPbobsCOM.Company
Dim vItem      As SAPbobsCOM.Items
Private Sub cmdItemPrice_Click()
    On Error GoTo ErrorHandler
    ' 初始化物料对象
    Set vItem = vCompany.GetBusinessObject(oItems)
    vItem.ItemCode = "CD1004"
    vItem.ItemName = "Bob Hope"
    ' 添加价格
    Call vItem.PriceList.SetCurrentLine(0)
    vItem.PriceList.Price = 500
    vItem.PriceList.PriceListName = "Test"

    ' 添加是否成功，如果成功，返回 0
    If (0 <> vItem.Add()) Then
        MsgBox ("添加失败!")
    Else
        MsgBox ("添加成功!")
    End If

```

ErrorHandler:

MsgBox ("Exception: " + Err.Description)

End Sub

## 1.6、特殊价格-----Special Prices

描述:

SpecialPrices 是一个用来管理和处理我们客户、供应商的特殊价格的主要对象。主要方法如下:

|            |                     |
|------------|---------------------|
| + Add      | 增加特殊价格              |
| + GetByKey | 检索以主键为条件的特殊价格       |
| + Remove   | 在条件许可的条件下, 删除特殊价格信息 |
| + SaveXML  | 保存特殊价格信息为 XML 文件格式  |
| + Update   | 更新特殊价格信息 (主键字段除外)   |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 SpecialPrices 对象的描述。

代码

下面代码演示了我们如何通过 SpecialPrices 添加一个特殊价格信息。

'定义变量, 初始化 oSpecialPrices 对象

Dim vSP As SAPbobsCOM.SpecialPrices

Set vSP = vCompany.GetBusinessObject(oSpecialPrices)

'给对象变量赋值

vSP.CardCode = "D10002"

vSP.Currency = "Eur"

vSP.DiscountPercent = 30.3

vSP.ItemCode = "A00001"

vSP.Price = 355.3

vSP.PriceListNum = 0

'增加特殊价格

Call vSP.Add

## 1.7、产品结构-----Product Trees

描述:

**ProductTrees** 它就是我们常常所说的产品结构, 这个对象主要描述的是一个完工产品的组成物料、半成品的详细信息, 其中包括价格、数量等信息, 相当于一个产品的物料清单。我们通过**ProductTrees**对象可以维护所有资料

。主要方法如下:

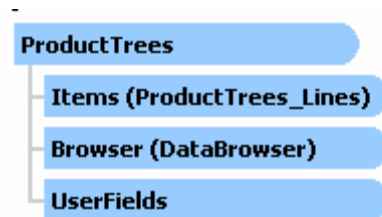
|            |                     |
|------------|---------------------|
| + Add      | 增加产品结构              |
| + GetByKey | 检索以主键为条件的产品结构信息     |
| + Remove   | 在条件许可的条件下, 删除产品结构信息 |

|           |                    |
|-----------|--------------------|
| + SaveXML | 保存产品结构信息为 XML 文件格式 |
| + Update  | 更新产品结构信息（主键字段除外）   |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 ProductTree 对象的描述。

#### 图例

对象模型如下



#### 代码

下面代码演示了我们如何通过 ProductTree 添加一个产品树。

```

Sub AddProdTree()
    Dim RetVal As Long
    Dim ErrCode As Long
    Dim ErrMsg As String
    ' 定义产品树对象，并且初始化
    Dim vItem As SAPbobsCOM.Items
    Set vItem = vCmp.GetBusinessObject(oItems)
    Dim vProdTree As SAPbobsCOM.ProductTrees
    Set vProdTree = vCmp.GetBusinessObject(oProductTrees)
    ' 得到我想创建物料树得物料
    RetVal = vItem.GetByKey("MyProdTree")
    ' 给变量赋值
    vProdTree.TreeCode = "MyProdTree"
    vProdTree.TreeType = iProductionTree
    ' 设置产品树的组合部分
    ' 第一个组合部分
    vProdTree.Items.ItemCode = "Assembly1"
    vProdTree.Items.Price = 20
    vProdTree.Items.Quantity = 1
    vProdTree.Items.Currency = "Eur"
    '' 第二个组合部分
    vProdTree.Items.Add
    vProdTree.Items.ItemCode = "Assembly2"
    vProdTree.Items.Price = 20
    vProdTree.Items.Quantity = 1
    vProdTree.Items.Currency = "Eur"

```

```

' 增加产品树
RetVal = vProdTree.Add
' 判断添加是否成功，如果成功返回 0
If (RetVal <> 0) Then
    vCmp.GetLastError ErrCode, ErrMsg
    MsgBox ErrCode & " " & ErrMsg
End If
End Sub

```

## 说明

上面只能添加一个产品结构的父项，一般的产品不可能没有子项的，如何在产品结构中添加产品子项，请参照 `ProductTrees_Lines` 对象的操作。

## 2、练习

- 1: 在 vb 中新建工程，然后得到合作伙伴的列表
- 2: 得到物料的列表(修改、查找、删除、增加)
- 3: 得到仓库列表
- 4: 得到用户列表
- 5: 得到业务合作伙伴的联系人

## 3、Transactional Data Objects

Transactional 对象包括 Documents, JournalEntries, Stock Transfer, Payments, 等等。

Transactional 是那种包含主、子表的对象，其中包括一个主表对象（例如：JournalEntries）和一个子表对象（举例来说 JournalEntries\_Lines），下面我们逐一介绍。

### 3.1、Journal Entries

#### 描述：

JournalEntries 是 SBO 财务模块的凭证（业务日志）业务对象（**OJDT**）。我们通过 JournalEntries 和 JournalEntries\_Lines 对象可以维护关于凭证的信息。

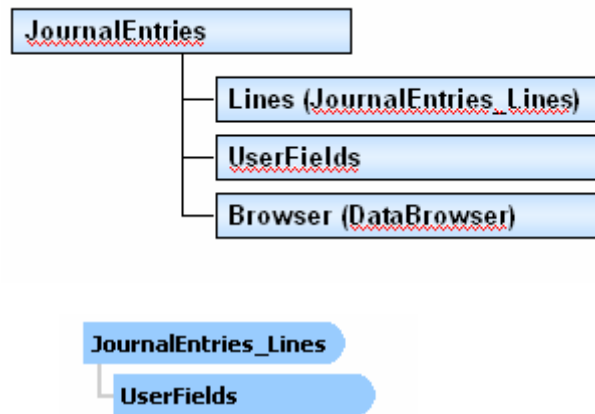
。主要方法如下：

|                  |                       |
|------------------|-----------------------|
| + Add            | 增加凭证                  |
| + GetByKey       | 检索以主键为条件的产凭证信息        |
| + Remove         | 在条件许可的条件下，删除凭证        |
| + SaveXML        | 保存凭证为 XML 文件格式        |
| + Update         | 更新凭证（主键字段除外）          |
| + SetCurrentline | 设定指定对象行开始。计数器是从 0 开始。 |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 JournalEntries 对象的描述。

#### 图例

对象模型如下



#### 代码

下面代码演示了我们如何通过 JournalEntries 添加一个凭证，并且将添加的信息保存为 XML 文件。

```

Private Sub AddJournalEntries_Click()
    On Error GoTo ErrorHandler
    Dim vCompany As SAPbobsCOM.Company
    ' 初始化 company 对象（关于 company 对象在下面章节有更详细的描述）
    Set vCompany = New SAPbobsCOM.Company
    ' 连结到数据库
    vCompany.CompanyDB = "SBODemo_China"
    vCompany.Password = "manager"
    vCompany.UserName = "manager"
    vCompany.Server = "(local)"

    If (0 <> vCompany.Connect()) Then
        MsgBox "Failed to connect"
        Exit Sub
    End If

    Dim nErr As Long
    Dim errMsg As String

    ' 增加凭证
    Dim vJE As SAPbobsCOM.JournalEntries
    Set vJE = vCompany.GetBusinessObject(oJournalEntries)
    vJE.TaxDate = Now
  
```

```

vJE.Lines.AccountCode = "110000"
vJE.Lines.ContraAccount = "10110"
vJE.Lines.Credit = 0
vJE.Lines.Debit = 150
vJE.Lines.DueDate = CDate("11/13/ 2002")
vJE.Lines.Line_ID = 0
vJE.Lines.ReferenceDate1 = Now
vJE.Lines.ShortName = "110000"
vJE.Lines.TaxDate = Now
Call vJE.Lines.Add
Call vJE.Lines.SetCurrentLine(1)
vJE.Lines.AccountCode = "10110"
vJE.Lines.ContraAccount = "110000"
vJE.Lines.Credit = 150
vJE.Lines.Debit = 0
vJE.Lines.DueDate = CDate("11/13/ 2002")
vJE.Lines.Line_ID = 1
vJE.Lines.ReferenceDate1 = Now
vJE.Lines.ShortName = "10110"
vJE.Lines.TaxDate = Now
If (0 <> vJE.Add()) Then
    MsgBox ("添加失败！")
Else
    MsgBox ("添加成功！")
    vJE.SaveXml ("c:\temp\JournalEntries" + Str(vJE.JdtNum) + ".xml")
End If

' 错误处理
Call vCompany.GetLastError(nErr, errMsg)
If (0 <> nErr) Then
    MsgBox ("Found error:" + Str(nErr) + ", " + errMsg)
End If

' 释放资源，断开数据库
Call vCompany.Disconnect
Set vCompany = Nothing
Exit Sub

ErrorHandler:
MsgBox ("Exception:" + Err.Description)
End Sub

```

### 3.2、Documents

#### 描述:

Documents主要描述的是销售、采购中的业务主数据（**ODRF**）。我们通过Documents和

Documents\_Lines对象可以维护相关信息

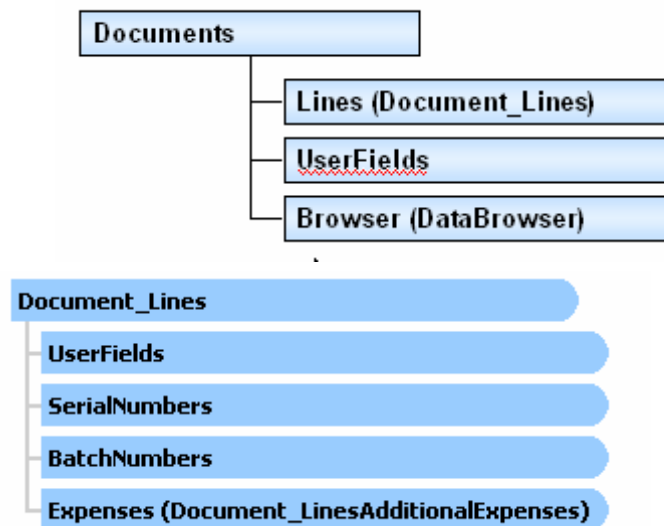
。主要方法如下：

|            |              |
|------------|--------------|
| + Add      | 增加数据         |
| + GetByKey | 检索数据         |
| + Remove   | 目前不支持数据删除    |
| + SaveXML  | 保存为 XML 文件格式 |
| + Update   | 更新数据（主键字段除外） |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 Documents 对象的描述。

## 图例

对象模型如下



## 代码

下面代码演示了我们如何通过 Documents 添加数据。

```

Sub AddInvoice_Click()
    Dim RetVal As Long
    Dim ErrCode As Long
    Dim ErrMsg As String
    ' 定义初始化 Documents 对象
    Dim vInvoice As SAPbobsCOM.Documents
    Set vInvoice = vCmp.GetBusinessObject(oInvoices)
    ' 给字段赋值
    vInvoice.Series = 0
    vInvoice.CardCode = "BP234"
    vInvoice.HandWritten = tNO
    vInvoice.PaymentGroupCode = "-1"
    vInvoice.DocDate = "21/8/2003"
    vInvoice.DocTotal = 264.6
  
```

```

' 添加 Invoice 表体第一行数据
vInvoice.Lines.ItemCode = "A00023"
vInvoice.Lines.ItemDescription = "Banana"
vInvoice.Lines.PriceAfterVAT = 2.36
vInvoice.Lines.Quantity = 50
vInvoice.Lines.Currency = "Eur"
vInvoice.Lines.DiscountPercent = 10

' 添加 Invoice 表体第二行数据
vInvoice.Lines.Add
vInvoice.Lines.ItemCode = " A00033"
vInvoice.Lines.ItemDescription = "Orange"
vInvoice.Lines.PriceAfterVAT = 118
vInvoice.Lines.Quantity = 1
vInvoice.Lines.Currency = "Eur"
vInvoice.Lines.DiscountPercent = 10
' 添加数据到数据库中
RetVal = vInvoice.Add
' 错误处理, 检查是否正确添加, 如果正确返回 0
If RetVal <> 0 Then
    vCmp.GetLastError ErrCode, ErrMsg
    MsgBox ErrCode & " " & ErrMsg
End If
End Sub

```

### 3.3、Payments

#### 描述:

**Payments** 是SBO系统银行模块的一个业务对象。 这个对象描述的是客户、供应商的收款和付款信息。付款方法包括现金、信用卡、等等。我们通过**Payments**对象可以维护相关信息。

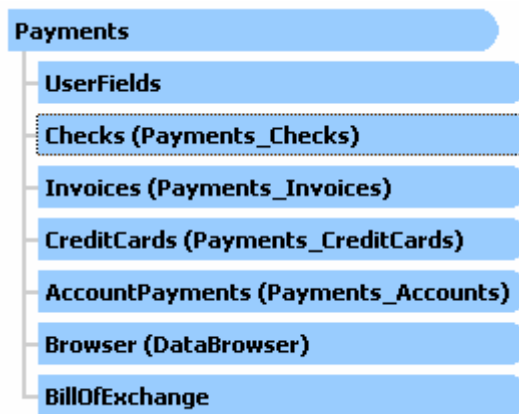
。主要方法如下:

|            |              |
|------------|--------------|
| + Add      | 增加数据         |
| + GetByKey | 检索数据         |
| + Remove   | 目前不支持数据删除    |
| + SaveXML  | 保存为 XML 文件格式 |
| + Update   | 更新数据（主键字段除外） |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 **Payments** 对象的描述。

#### 图例

对象模型如下



## 代码

下面代码演示了我们如何通过 Payments 添加数据。

```

Private Sub cmdTest_Click()
    On Error GoTo ErrorHandler
    Dim vCompany As SAPbobsCOM.Company

    ' 初始化 company 对象
    Set vCompany = New SAPbobsCOM.Company

    ' 设置连结数据库参数
    vCompany.CompanyDB = "SBODemo_China"
    vCompany.Password = "manager"
    vCompany.UserName = "manager"
    vCompany.Server = "(local)"
    ' 连结到数据库
    If (0 <> vCompany.Connect()) Then
        MsgBox "Failed to connect"
        Exit Sub
    End If

    Dim nErr As Long
    Dim errMsg As String
    ' 定义并且初始化 Payments 对象
    Dim vPay As SAPbobsCOM.Payments
    Set vPay = vCompany.GetBusinessObject(oIncomingPayments)
    ' 给字段变量赋值
    vPay.Address = "622-7"
    vPay.ApplyVAT = 1
    vPay.CardCode = "D10006"
    vPay.CardName = "Card D10004"
    vPay.CashAccount = "288000"
  
```

```

vPay.CashSum = 0
vPay.ContactPersonCode = 1
vPay.DocCurrency = "Eur"
vPay.DocDate = Now
vPay.DocRate = 0
vPay.DocTpyte = 0
vPay.HandWritten = 0
vPay.JournalRemarks = "Incoming - D10004"
vPay.LocalCurrency = tYES
vPay.Printed = 0
vPay.Referencel = 8
vPay.Series = 0
vPay.SplitTransaction = 0
vPay.TaxDate = Now
vPay.TransferAccount = "10100"
vPay.TransferDate = Now
vPay.TransferSum = 5031.2
'添加数据到数据库, 如果成功返回 0
If (vPay.Add() <> 0) Then
    MsgBox ("Failed to add a payment")
End If
' 错误处理
Call vCompany.GetLastError(nErr, errMsg)
If (0 <> nErr) Then
    MsgBox ("Found error:" + Str(nErr) + ", " + errMsg)
Else
    MsgBox ("Succeed in payment.add")
End If

' 断开数据库连结, 释放资源
Call vCompany.Disconnect
Set vCompany = Nothing
Exit Sub
ErrorHandler:
    MsgBox ("Exception:" + Err.Description)
End Sub

```

### 3.4、Stock Transfer

#### 描述:

**StockTransfer** 是SBO库存模块中的一个业务对象(OWTR)。这个对象描述的是产成品、物料从一个仓库调拨到另外一个仓库。我们通过**StockTransfer**和**StockTransfer\_Lines**对象可以维护相关信息

。主要方法如下:

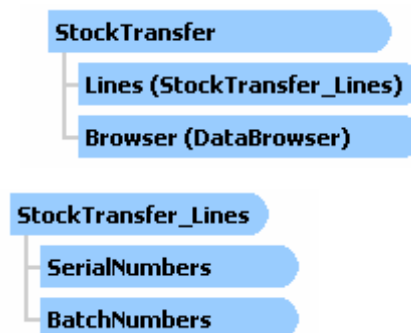
**+ Add**                      **增加数据**

|            |              |
|------------|--------------|
| + GetByKey | 检索数据         |
| + Remove   | 目前不支持数据删除    |
| + SaveXML  | 保存为 XML 文件格式 |
| + Update   | 更新数据（主键字段除外） |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于**StockTransfer**对象的描述。

### 图例

对象模型如下



### 代码

下面代码演示了我们如何通过**StockTransfer**添加数据。

```

Private Sub AddStockTransfer_Click()
    On Error GoTo ErrorHandler
    Dim vCompany As SAPbobsCOM.Company
    ' 初始化 company 对象
    Set vCompany = New SAPbobsCOM.Company
    ' 设置连结数据库参数
    vCompany.CompanyDB = "SBODemo_China"
    vCompany.Password = "manager"
    vCompany.UserName = "manager"
    vCompany.Server = "(local)"
    ' 连结到数据库
    If (0 <> vCompany.Connect()) Then
        MsgBox "Failed to connect"
        Exit Sub
    End If

    Dim nErr As Long
    Dim errMsg As String
    ' 定义并且初始化 StockTransfer 对象
    Dim vStockTransfer As SAPbobsCOM.StockTransfer
    Set vStockTransfer = vCompany.GetBusinessObject(oStockTransfer)
    ' 给字段变量赋值

```

```

With vStockTransfer
    .Address = "622-7"
    .CardCode = "D10006"
    .CardName = "Card D10004"
    .DocDate = Now
    .DocEntry = 0
    .DocNum = 1
    .CreationDate = Now
    .FromWarehouse = "01"
    .FinancialPeriod = 5
    .TransNum = 123
    .JournalMemo = "Incoming - D10004"
    .Series = 0
    .TaxDate = Now
    ' 添加 StockTransfer 表体第一行数据
    .Lines.ItemCode = "A00023"
    .Lines.ItemDescription = "Banana"
    .Lines.Price = 2.36
    .Lines.Quantity = 50
    .Lines.Currency = "Eur"
    .Lines.DiscountPercent = 10
    .Lines.WarehouseCode = "02"

    ' 添加 StockTransfer 表体第二行数据
    .Lines.Add
    .Lines.ItemCode = " A00033"
    .Lines.ItemDescription = "Orange"
    .Lines.Price = 118
    .Lines.Quantity = 1
    .Lines.Currency = "Eur"
    .Lines.DiscountPercent = 10
    .Lines.WarehouseCode = "02"
End
' 添加数据到数据库，如果成功返回 0
If (vStockTransfer.Add() <> 0) Then
    MsgBox ("添加失败！")
End If
' 错误处理
Call vCompany.GetLastError(nErr, errMsg)
If (0 <> nErr) Then
    MsgBox ("Found error:" + Str(nErr) + ", " + errMsg)
Else
    MsgBox ("添加成功！")
End If

```

```

' 断开数据库连结，释放资源
Call vCompany.Disconnect
Set vCompany = Nothing
Exit Sub
ErrorHandler:
    MsgBox ("Exception:" + Err.Description)
End Sub

```

## 4、练习

- 1: 在 vb 中新建工程，通过 Documents 在 SBO 中添加一张订单！
- 2: 通过 Messages 对象在系统中创建一条消息。

## 二、Non-Business Objects

### 1、Company

描述:

**Company** 这个对象主要作用是将 SBO 系统和 Microsoft SQL 服务器建立连结，并且存取数据。我们在前面也用到了此对象，在这里我们将做进一步的介绍

。主要方法如下:

|                       |                |
|-----------------------|----------------|
| + Connect             | 和数据库建立连结       |
| + Disconnect          | 断开和数据库的连结      |
| + GetCompanyList      | 得到公司全部数据库的信息   |
| + GetLastErrorMessage | 执行最后一步后，得到错误信息 |
| + StartTransaction    | 开始一个事务         |
| + EndTransaction      | 结束一个事务         |

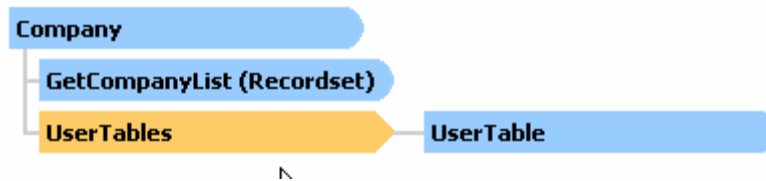
主要属性

|              |                 |
|--------------|-----------------|
| + CompanyDB  | 返回或设置数据库名       |
| + Connected  | 判断数据库状态（连接或、断开） |
| + DbUserName | 连结数据库的用户名       |
| + DbPassword | 连结数据库的密码        |
| + Server     | 服务器名称           |
| + UserName   | 登陆 SBO 系统的用户名   |
| + Password   | 登陆 SBO 系统的密码    |
| + UseTrusted | 和数据库连接          |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 **Company** 对象的描述。

图例

对象模型如下



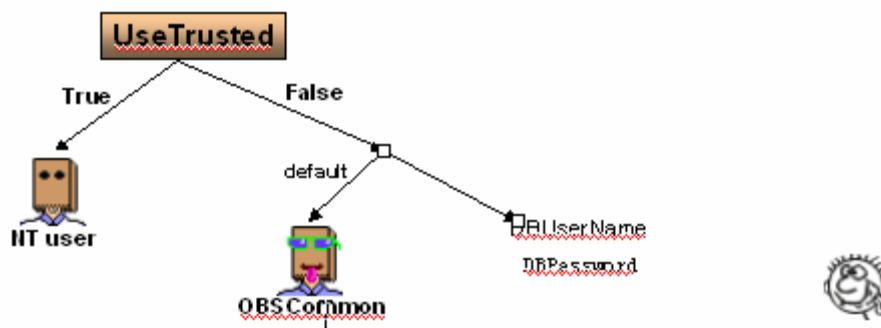
与 SQL Server 建立连接时的参数说明：

#### Connect to SQL Server

- Server
- UseTrusted
- DBUserName
- DBPassword

#### Connect to SBO

- CompanyDB
- UserName
- Password
- Language



#### 代码

下面代码演示了我们如何通过 **Company** 和 SBO 数据库建立连接。

```

' 定义 Company 对象
Dim vCompany As SAPBobsCOM.Company
Dim lRetCode, lErrCode As Long
Dim sErrMsg as String' 初始化 Company 对象
Set vCompany = New SAPBobsCOM.Company
' 给参数赋值
vCompany.Server = "(local)"
vCompany.CompanyDB = "SBODemo_China"
vCompany.UserName = "manager"
vCompany.Password = "manager"
' 设置语言类型
vCompany.Language = ln_English
' 设置和 SqlServer 连接时的信任关系, True 时使用 Nt 账户连接
vCompany.UseTrusted = True
' 和数据库建立连接 lRetCode = vCompany.connect' 检查是否连接成功,

```

如果成功返回 0

```

If lRetCode <> 0 Then
    vCompany.GetLastError lErrCode, sErrMsg

```

End If

## 说明

在 Company 对象中，还有一个方法是 GetLastError ()，在其他章节的代码中常常用到，它主要是返回执行数据操作后的错误信息，注意：它是返回最后一次执行操作数据库时的异常信息，如果没有错误发生，那么他返回空字符串。

## 2、 RecordSet

### 描述：

使用 RecordSet 运行 Sql 查询和存储过程

。主要方法如下：

|                |                |
|----------------|----------------|
| + DoQuery      | 执行 Sql 语句      |
| + MoveFirst    | 记录集中的第一条记录     |
| + MoveLast     | 记录集中的最后一条记录    |
| + MoveNaxt     | 下一条记录          |
| + MovePrevious | 上一条记录          |
| + SaveXML      | 将记录集保存为 XML 文件 |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 RecordSet 对象的描述。在其他章节也有更详细的解释。

### 图例

对象模型如下



### 代码：

```

‘定义变量
Dim Count As Long
Dim FldName As String
Dim FldVal As String
Dim i As Long
Dim RecSet As SAPbobsCOM.Recordset
‘初始化 RecordSet 对象。注意：必须这样初始化
Set RecSet = vCmp.GetBusinessObject(BoRecordset)
‘执行 Sql 语句
RecSet.DoQuery ("select * from OADM")
‘返回字段总数
Count = RecSet.Fields.Count
While RecSet.EOF = False
    ‘通过字段总数做循环，返回变量值、字段名称

```

```

For i = 0 To Count - 1
    FldName = RecSet.Fields.Item(i).Name
    FldVal = RecSet.Fields.Item(i).Value
Next i
'记录集指针下移，得到下一条记录信息。
RecSet.MoveNext
Wend

```

#### 说明

在 Recordset 中，其中 fields 是记录集中的字段集合，为了访问字段中的变量，我们也可以用 Fields(“字段名”).Value 属性访问。

### 3、DataBrowser

#### 描述：

使用 DataBrowser 可以浏览、导航数据，比如合作伙伴、物料等。

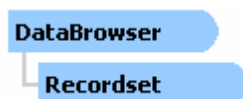
主要方法如下：

|                |                |
|----------------|----------------|
| + DoQuery      | 执行 Sql 语句      |
| + MoveFirst    | 记录集中的第一条记录     |
| + MoveLast     | 记录集中的最后一条记录    |
| + MoveNext     | 下一条记录          |
| + MovePrevious | 上一条记录          |
| + ReadXML      | 读取 XML 文件到记录集中 |

其它更多的属性、方法请参照 SAP Business One SDK - DI API Version 6.5 中关于 DataBrowser 对象的描述。在其他章节也有更详细的解释。

#### 图例

对象模型如下



#### 代码：

```

'下面的例子是使用 DataBrowser 如何去导航合作伙伴中的数据
Private Sub DataBrowserOperations()
    定义变量
    Dim BusinessPartners As SAPbobsCOM.BusinessPartners
    Dim oRecordSet As SAPbobsCOM.Recordset
    '// 得到合作伙伴业务主数据对象
    Set BusinessPartners = oCompany.GetBusinessObject(oBusinessPartners)
    '// 初始化 Recordset 对象
    Set oRecordSet = oCompany.GetBusinessObject(BoRecordset)

```

```

oRecordSet.DoQuery ("Select cardcode from ocrd where cardtype = 'C'")
BusinessPartners.Browser.Recordset = oRecordSet
'//存取数据

BusinessPartners.CardCode
BusinessPartners.CardName
If BusinessPartners.Browser.EOF = False Then
    BusinessPartners.Browser.MoveNext
End If
'//得到上一个 Business Partner 数据
If BusinessPartners.Browser.BoF = False Then
    BusinessPartners.Browser.MovePrevious
End If

End Sub

```

#### 4、 练习

- 1: 新建 vb 工程，选择数据库，并建立连接
- 2: 通过 Recordset 浏览合作伙伴信息
- 3: 通过 Browser 浏览业务合作伙伴数据

## 第三节 Working with XML

### 一、什么是 XML

XML文件是一种可扩展标示语言，它是一种储存数据技术，因为数据传输安全、快捷被广泛使用。

XML文件的结构如下：

- 层次
- 节点
- 标志

### 二、XML 文件的优点

- 使SBO数据库和客户的数据库之间能够进行大规模的数据交换（不管数据类型）
- 符合标准
- 高性、有效的
- 节省内存
- 存取速度快
- 简单明了、容易理解

### 三、DI API.中使用 XML

公司对象

- CmpObj.GetBusinessObjectFromXML (FileFullPath, Index) --- 读取窗体 XML 文件
- CmpObj.GetXMLLelementCount (FileFullPath) -----得到窗体对象总数
- CmpObj.GetXMLObjectType (FileFullPath, Index) -----读取对象类型

例如:

```
Private Sub XMLRead(vCmp As SAPbobsCOM.Company, _
                    vBP As SAPbobsCOM.BusinessPartners)
    Dim FileName As String
    Dim Ecount As Long
    Dim ii As Long

    FileName = "c:\temp\myXml.xml"
    ' 在 XML 文件中得到业务对象总数
    Ecount = vCmp.GetXMLLelementCount(FileName)
    ' 循环得到合作伙伴对象
    For ii = 0 To Ecount - 1
        if vCmp.GetXMLObjectType(FileName, ii) = oBusinessPartners Then
            ' 读取数据
            Set vBP = vCmp.GetBusinessObjectFromXML(FileName, ii)
            Exit For
        End If
    Next ii
End Sub
```

### 业务对象

- BusObj.GetBusinessObjectXmlSchema(Object) -----读取 XML 文件
  - BusObj.SaveXML (FileFullPath) -----保存为 XML 文件
- 例如:下面的代码是将业务合作伙伴的主数据取出, 然后使用 SaveXML () 对象保存为 XML 文件。

```
' 连接数据库
' 定义变量
Dim vBP As SAPbobsCOM.BusinessPartners
Set vBP = vCompany.GetBusinessObject(oBusinessPartners)
vBP.CardCode = "C08154711"
vBP.CardName = "Carl Sagan"
vBP.CardType = cCustomer
...
Call vBP.ContactEmployees.Add
Call vBP.ContactEmployees.SetCurentLine(0)
vBP.ContactEmployees.Name = "C1"
...
If (0 <> vBP.Add) Then
    MsgBox ("Failed to add an business partner")
Else
```

```

Dim objCode As String
MsgBox ("Succeeded add a business parnter, new objcode=" _
    + vBP.CardCode)
BP.SaveXml ("c:\temp\BP" + vBP.CardCode + ".xml")

End If

```

## 四、绑定 XML 文件

```

DocLevel                (XML 文档)
<BOM>                   (根节点)
<BO>
  <AdminInfo>            (TblLevel)
    <Object> 2 </Object>  (RowLevel)
  </AdminInfo>
    <OCRD>
      <row>
        <CardCode> Roni </CardCode>
        ... </row>
      </OCRD>
    </BO>
  </BOM>

```

## 五、其它工具中使用

例如，我们在 VB 中可以直接使用对象调用 XML 文件  
 在 VB 工程中添加 Microsoft XML, v3.0 对象使用对象 DOMDocument 和 IXMLDOMNode 操做 XML 文件.

## 六、练习

- 1: 在 vb 中新建工程，对去 xml 文件，将类型是合作伙伴的信息，添加到 sbo 系统中。
- 2: 将类型为联系人的信息，添加到联系人中。

# 第四节 事务

## 一、概述

DI API 支持二种不同类型的事务

### 1、Single Transaction

每个数据操作的执行对象都启动一个事务，操作结果依赖对象得执行结果（成功或失效），系统自动地提交或回滚当前数据操作。如果操作是成功，那么整个过程将被提交。

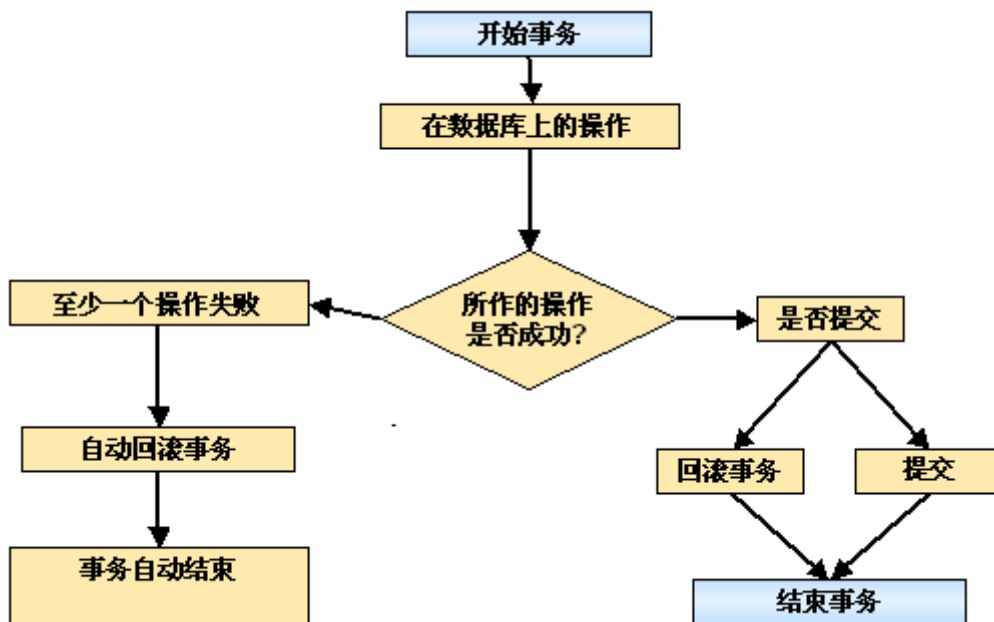
## 2、Global Transaction

在这个事务中，你能执行很多步数据操作，如果某一步的数据操作失败，那么整个事务将自动地被迫回滚。

开始和结束一个事务，我们用对象Company中的下面2个对象来处理：

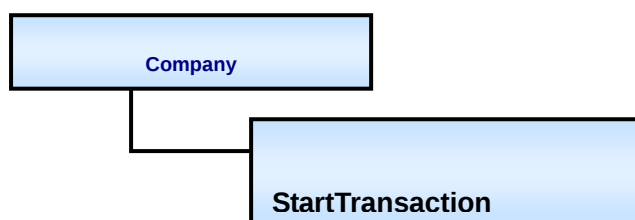
Company.StartTransaction

EndTransaction[wf\_RollBack, wf\_Commit]



## 二、例子

例如：开始一个事物：

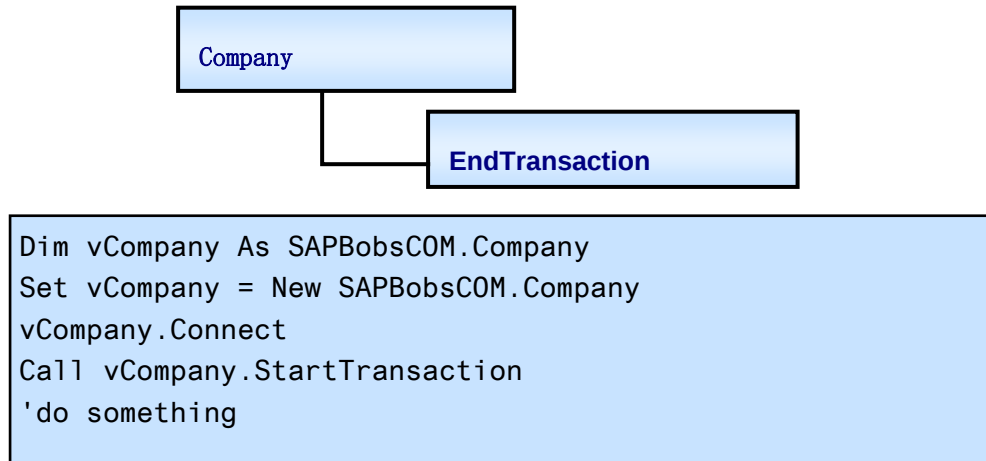


```

Dim vCompany As SAPBobsCOM.Company
Set vCompany = New SAPBobsCOM.Company
vCompany.Connect
Call vCompany.StartTransaction

public sub StartTransaction()
  
```

结束事务：



### 三、练习

- 1: 新建一个工程，在里面添加一段对 SBO 数据库操作的代码，如：添加合作伙伴信息  
然后启用事务，看看前后的变化。

## 第五节 DI\_API 高级特性

### 一、Data Interface MetaData Objects

#### 1、上下文

SAP Business One 通过用户自定义表，及用户自定义字段提供系统的可扩展性。SAP Business One 能够感知通过其提供的接口（界面功能或 DI\_API）创建的用户表。通过这种方式创建用户表相对于通过数据库管理工具创建用户表具有许多优势：能够保证数据的完整性、安全性及升级兼容，也可以利用 SBO 现在及将来可能提供的服务，如数据备份服务等。

DI\_API 提供了一组接口来扩展数据库即 MetaData objects。通过 API 来创建用户表相对于通过界面功能创建用户表更适于 AddOn 的开发。AddOn 通过 API 创建用户表更灵活，并且可维护性得到提高，方便升级、发布。

#### 2、类职责

MetaData objects 使扩展数据库成为可能。主要职责如下：

- ✧ 增加用户表
- ✧ 在 SAP Business One 允许的任何表中增加用户字段

MetaData objects 包括如下 DI API 接口：

UserTablesMD object: 用于管理用户自定义表，主要用于创建。

UserFieldsMD object: 管理用户自定义表或系统表的用户自定义字段。

UserKeysMD object : 用于管理用户自定义表的用户自定义关键字。用户自定义表中默认 Code 同 Name 构成主关键字。实际上 UserKeysMD 主要用于添加用户定义表的次关键字 (secondary keys)。最好在用户自定义表为空的时候创建关键字。

### 3、用法及示例

#### 3.1、使用 UserTableMD 添加用户自定义表



```

'//Object variable
Dim pUTables As SAPbobsCOM.UserTablesMD
'//Create Instance of UserTablesMD object
Set pUTables = pCompany.GetBusinessObject(oUserTables)
'//Check whether table already exists
If pUTables.GetByKey(CStr(txtTableName)) Then
    Set pUTables = Nothing
    Exit Sub
Else
    pUTables.TableName = txtTableName
    pUTables.TableDescription = txtTableDescription
    pUTables.Add
End If
'//IMPORTANT: Only one handle to a user table or field object
'//          should be alive at one time
Set pUTables = Nothing

```



3.2、使用 UserFieldsMD 对象创建用户自定义字段

```
//Object variable
Dim pUFields As SAPbobsCOM.UserFieldsMD

//Create Instance of UserTablesMD object
Set pUFields = pCompany.GetBusinessObject(oUserFields)

//Add field... "name"
pUFields.TableName = txtTableName
pUFields.Name = "name"
pUFields.Description = "渠道名称1"
pUFields.Type = db_Alpha
pUFields.EditSize = 20
pUFields.Add

//IMPORTANT: Only one handle to a user table or field object
//          should be alive at one time
Set pUFields = Nothing
```



### 3.3、使用 UserKeysMD 创建表的关键字

UserKeysMD 使用步骤:

1. 指定表名及关键字的名称
2. 添加属于关键字的字段
3. 添加关键字

```
//Object variable
Dim oUserKeysMD As SAPbobsCOM.UserKeysMD
//Create Instance of UserTablesMD object
Set oUserKeysMD = oCompany.GetBusinessObject(oUserKeys)
//set the table name and the key name
oUserKeysMD.TableName = Text1.Text
oUserKeysMD.KeyName = Text2.Text
//Determine whether the key is unique or not
oUserKeysMD.Unique = Combo1.ListIndex
//add column, don't use add for the first element
oUserKeysMD.Elements.ColumnAlias = "name"
' //add the key
lRetCode = oUserKeysMD.Add

//IMPORTANT: Only one handle to a user table or field object
//          should be alive at one time
Set oUserKeysMD = Nothing
```

## 4、注意

✧ 因为命名空间的原因 SBO 透明地在用户定义表的表名前添加@符号, 在用户定义字段

前添加 U\_

- ✧ 为了防止数据库阻塞，DI API 只允许一个 MetaData object 实例存在
- ✧ 如果你的使用的 DI API 版本为 6.2 则用户自定义表的表名限制为 3 个字符，用户自定义字段名限制为 8 个字符
- ✧ UserTable<>UserTablesMD,UserField object <> UserFieldsMD object

## 5、练习

1. 创建一张用户自定义表
2. 在自定义表上增加一个用户自定义字段
3. 创建用户定义关键字，将新增字段添加关键字中
4. 向用户自定义表中添加记录

## 二、Data Interface SBObob Object

### 1、上下文

SAP Business One 采用 Domain Object 或者称为 Business Object 的模式提供 DI API，而不是通过 Service Interface 的方式。目前没有提供 Business Object 的 Home Object 或 Factory Object。在集合操作时采用 Recordset 完成查询，导航浏览。如果希望以 Business Object 的方式导航浏览数据则使用相应 Business Object 的 DataBrowser 接口。对于一些常用的查询，例如枚举所有仓库，枚举所有用户等,DI API 提供了一个 Facade 接口即 SBObob Object。SBObob 能快速简便的获取许多重要信息，为了方便表格类型的数据操作，SBObob 接口成员基本都以 Recordset 的类型返回。除了担任 Façade 职责，SBObob 还担任 Utility 职责提供一些常用的附加功能。

### 2、类职责

提供系统常用信息获取的 Facade 接口。其主要接口成员包括：

1. GetBPList Method
2. GetDueDate Method
3. GetSystemCurrency Method
4. GetLocalCurrency Method
5. GetCurrencyRate Method
6. SetCurrencyRate Method
7. GetItemPrice Method
8. GetItemList Method
9. GetUserList Method
10. GetWareHouseList Method
11. GetContactEmployees Method

提供系统常用的公用程序 `utility` 接口，其主要成员包括：

1. `Format_DateToString Method`: 将系统日期转换成字符串
2. `Format_StringToDate Method`: 将日期形式的字符串转换成系统日期
3. `Format_MoneyToString Method`: 根据指定精度将货币转换成字符串
4. `GetIndexRate Method`
5. `GetObjectKeyBySingleValue Method`: 通过 Business Object 的键值获取单个对

象

### 3、用法及示例

#### 3.1、枚举前 10 个仓库列表

```
//Object variable
Dim oSBObob As SAPbobsCOM.SBObob
Dim oRecordset As SAPbobsCOM.Recordset
Dim i As Long

//Get an initialized SBObob object
Set oSBObob = oCompany.GetBusinessObject(BoBridge)

// Get an initialized Recordset object
Set oRecordset = oCompany.GetBusinessObject(BoRecordset)

// Execute the SBObob GetItemList method
Set oRecordset = oSBObob.GetWareHouseList()

// the RecorSet now contains a list of Keys of the Requested Type
// use the Recordset to set the Business object
// by using the GetByKey method
i = 0
List1.Clear
List1.AddItem ("Code" & vbTab & vbTab & "Name")
List1.AddItem ("-----")
Do Until oRecordset.EOF = True
    // make sure the record set didn't reach the EOF
    If oRecordset.EOF = False And i < 10 Then
        List1.AddItem oRecordset.Fields(0).Value & vbTab & vbTab &
oRecordset.Fields(1).Value
        i = i + 1
    End If
    oRecordset.MoveNext
Loop
```



## 4、练习

使用 `GetObjectKeyBySingleValue` 方法查找某个 Business Object

## 三、Data Interface

### Recordset Object vs. DataBrowser Object

#### 1、上下文

DI API 提供的大多数接口都是以 Business Object 方式提供，为了访问没有提供 API 的系统表及用户自定义的表，DI API 提供 Recordset 接口进行数据访问。虽然 Recordset 可以导航浏览数据，但是对于提供 API 的 Business Object 可以采用 DataBrowser 接口方便数据访问。DataBrowser 实现关系数据库的表同 Business Object 的映射。DataBrowser 相当于 Class Mapper。不同的 Business Object 提供 DataBrowser 接口的不同实现，对于 API 的使用人员不用关心其实现。DataBrowser 对象不能创建，只能通过 Business Object 的属性激活访问。一旦用 Recordset 初始化 DataBrowser 后，就能以某种 Business Object 类型进行导航，并通过对象属性访问相关信息。

#### 2、类职责

**Recordset:** 从数据库中查询数据，导航浏览查询结果集。对没有提供 DI API 的系统表及用户自定义表进行数据操作。通过其 `DoQuery` 方法在表上执行 DML SQL 查询。

**DataBrowser:** 提供 Recordset 同 xml 数据到 Business Object 间的映射。Business Object 利用其 DataBrowser 在 Recordset 同 xml 结果集中导航，并通过对象属性来访问数据。

### 3、用法及示例

#### 3.1、Recordset 用法及示例

Recordset 的使用步骤:

1. Definition
2. DoQuery
3. Browse the records

```
//Create a Recordset of all customers
Dim pRS As SAPbobsCOM.Recordset
Dim Count As Long
Dim FldName As String
Dim FldVal As String
Dim i As Long
Set pRS = pCompany.GetBusinessObject(BoRecordset)
//Executes SQL query
pRS.DoQuery "Select * From OCRD where CardType = 'C'"
Count = oRecordSet.Fields.Count
//Navigating to the first record in the result set
pRS.MoveFirst
//Reads the data from the current Record's fields
While pRS.EOF = False
//The inner loop run over all the fields in one record (line) of the table
For i = 0 To Count - 1
// Retrieve the field name
FldName = pRS.Fields.Item(i).Name
// Retrieve the field value
FldVal = pRS.Fields.Item(i).Value
//Here you can do what ever you want with the data
Next i
pRS.MoveNext
Wend
```

#### 3.2、DataBrowser 用法及示例

DataBrowser 使用步骤

1. 定义 Recordset
2. 执行 Recordset 的 DoQuery 进行查询
3. 将执行查询后的 Recordset 设置到 DataBrowser 属性中
4. 通过 DataBrowser 导航操纵数据

```
Dim BP As SAPbobsCOM.BusinessPartners
Set BP = oCompany.GetBusinessObject(oBusinessPartners)
Dim PropVal As String
//1) Define recordset for Data browser object
```

```

Dim RecSet As SAPbobsCOM.Recordset

'//2) Set
Set RecSet = oCompany.GetBusinessObject(BoRecordset)

'//3) Doing query on the RecSet
RecSet.DoQuery "select CardCode, CardName from OCRD where CardType = 'C'"

'4) Setting the data browser with the recordset
BP.Browser.Recordset = RecSet

BP.Browser.MoveFirst

While BP.Browser.EOF = False
    'Direct approach to the properties
    ProbVal = BP.CardCode
    PropVal = BP.CardName
    '//Here you can do what ever you want with the data
    BP.Browser.MoveNext
Wend

```

## 4、注意

- ✧ 推荐对于提供 API 的表不要采用 Recordset 进行 write 操作，尽量使用 DI API。有利于保证业务规则正确、数据完整及升级的兼容。
- ✧ 使用 Recordset 执行查询只支持 Data Manipulation Language (DML)，对于 Data Definition Language (DDL) 则是禁止的。例如：CREATE, DROP, ALTER, and TRUNCATE。如果需要创建删除表最好采用前面介绍的 MetaData Objects (UserTablesMD and UserFieldsMD)。
- ✧ 使用 DataBrowser 前一定要通过正确的 Recordset 对其进行初始化

## 5、练习

从 OCRD 表中提取 CardCode 及 CardName 信息：

1. 使用 Recordset 对象
2. 使用 DataBrowser 对象

## 第四章 UI API

### 第一节 UI API 概述

#### 一、目的

SAP Business One User Interface API(UI API)是一个开发工具包,它使得 SAP 合作伙伴的应用解决方法与 SAP Business One 实现集成,并且根据需要,可以对 SAP Business One 功能进行修改,使得基于 SAP Business One 解决方案的价值得以提升,通过使用 UI API 的扩展,能满足客户界面层和独特业务逻辑的需要。

#### 二、适用性

通过使用 UI API 的功能函数,实现对 SAP Business One 应用程序界面层功能进行扩展,为第三方基于 SAP Business One 实现行业解决方案提供了有效的工具。

#### 三、开发说明

SDK 的 UI API 是 DCOM 模式的控件包,开发用户需要熟悉 Microsoft Visual C/C++®, Microsoft Visual Basic®, and Microsoft .NET 开发平台。同时对 SAP Business One 标准应用具有一定程度了解。

#### 四、运行需求

基于 SDK UI API 开发的功能包,可以运行于 Microsoft Windows® 2000, Windows NT® 4.0, and Windows XP®操作平台上。

### 第二节 UI API 体系介绍

UI API 是 SAP Business One SoftWare Development Kit(SDK)一个重要的组成部分, SAP Business One SDK 主要由 DI API 和 UI API 构成,UI API 彻底展示了用户界面层控制的原理:

提供对象与方法轻松访问界面元素

捕获并处理 SAP Business One 系统内部事件

增加菜单、窗体、窗体元素等界面对象,并可实现对已有的窗体或窗体元素进行修改

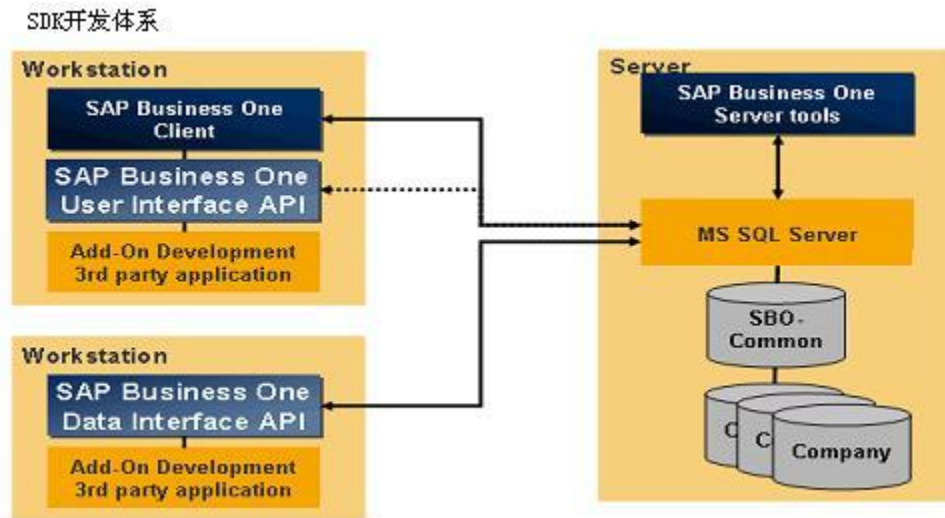
实现与第三方解决方案(基于 SDK UI API)一体化

正是由于 SDK 提供这些优点,独立软件商可以基于 SAP Business One 产品真正实现第三方解决方案

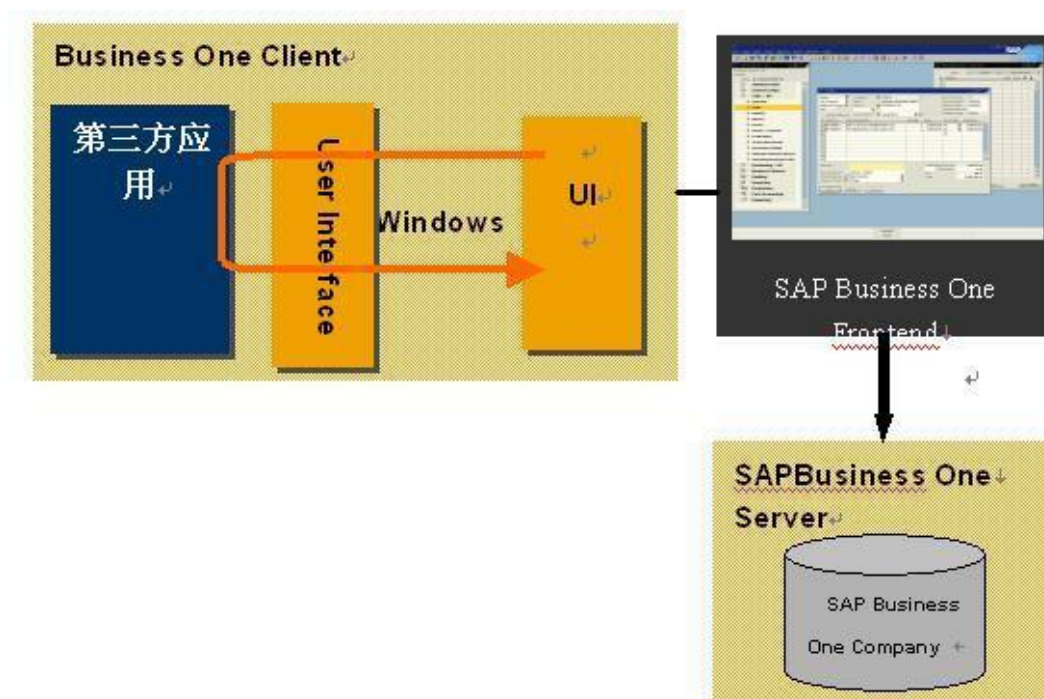
#### 一、体系

SAP Business One 是属于两层架构的软件系统,包括客户端和服务端。客户端由用户

界面和实现与服务端数据交换的业务逻辑对象组成；服务端有传真服务、邮件服务、拨号服务和数据库系统组成。SDK 中业务逻辑对象直接通过 ODBC 协议实现与数据库的连接。基于 SDK 扩展 SAP Business One 系统架构图如下

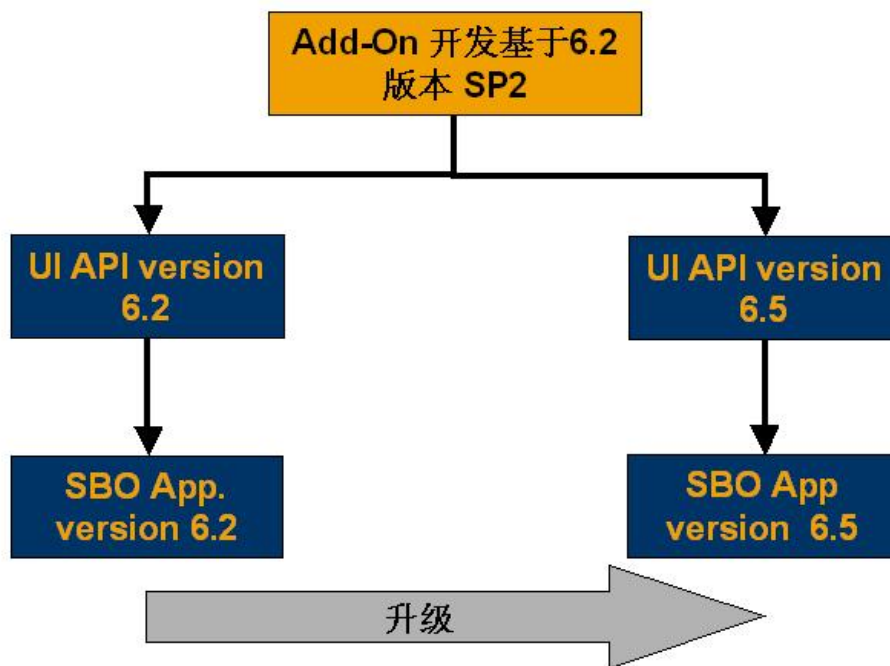


Add-On 解决方案通过 UI API 中 SboGuiApi 对象定义界面实例，通过 Application 对象取得 Business One 主线程实例，通过 SboGuiApi.Connect 实现与 SAP Business One 界面层的连接，过程如下图：



## 二、兼容性

SAP Business One 产品与 SDK 版本具有对应性关系，不同的 SAP Business One 应用有对应的 SDK 版本与之适应。SDK UI API 实现向下兼容原则，即使用低版本 UI API 开发的程序在高版本 UI API 下可以使用，如解决方案基于 SDK 第 X 版本开发，它也能运行于 SDK X+1 的版本环境下。



几点说明：

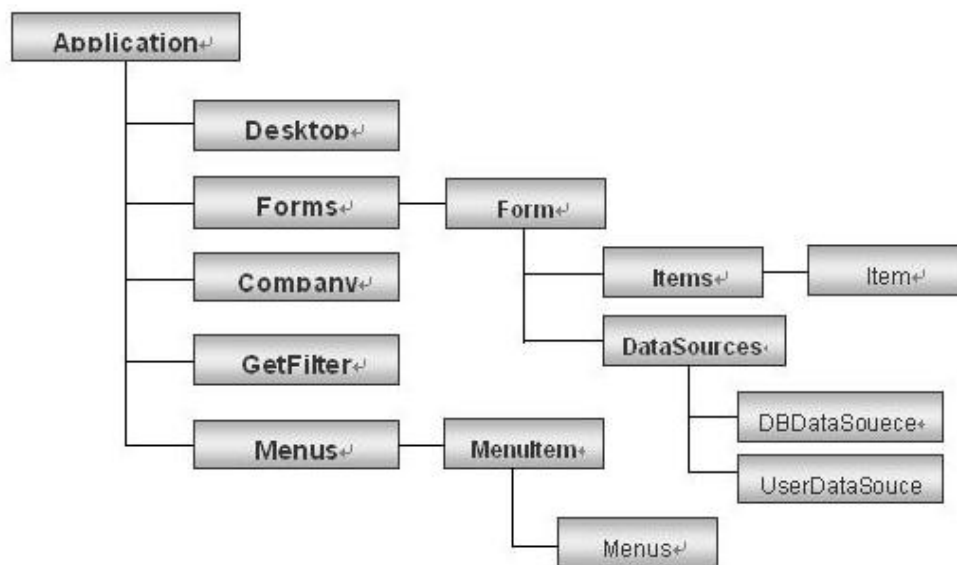
尽量保证 SDK 与 SAP Business One 系统版本的一致性，这将对基于 UI API 开发的解决方案具有最优的兼容性。

如果需要修改 SAP Business One 已有的功能，一定要创建一个新的用户模块，不宜直接对 SAP Business One 中提供的功能进行修改，以保证版本升级的兼容性。如 SAP Business One 应用系统中提供的窗体不适合用户的使用规则，可以通过 UI API 增加一个窗体的方式满足需求。

使用文档对 SAP Business One 界面修改的内容记录，当发布新版 SAP Business One 应用程序时，可以有效地帮助调整使用 UI API 开发的内容。

### 第三节 UI API 对象

UI API 中通过使用 Application Object 实现对窗体、菜单、公司、事件过滤等对象定制。



下面对 UI API 主要对象进行描述：

Application Object 是 UI API 中的根对象，该对象描绘当前 SAP Business One 应用，通过使用 Application Object 对应用的控制。

Form Object 描述 Business One 中的应用窗体，通过窗体对象实现对所属窗体中控件访问。

Item Object 通过该对象实现对窗体中控件的大小、位置、可见等属性的控制

Menu item Object 通过本对象实现对菜单项属性的控制。

| Application |                                                                                                               |
|-------------|---------------------------------------------------------------------------------------------------------------|
|             | <ul style="list-style-type: none"> <li>•描述一个 SAP Business One application应用</li> <li>•使用本对象对应用进行控制</li> </ul> |
| Form        |                                                                                                               |
|             | <ul style="list-style-type: none"> <li>•描述应用窗体</li> <li>•通过本对象实现对窗体中对象的访问与修改</li> </ul>                       |
| Item        |                                                                                                               |
|             | <ul style="list-style-type: none"> <li>•描述窗体中项目</li> <li>•通过本对象实现对位置、对象大小等属性修改</li> </ul>                     |
| ItemEvent   |                                                                                                               |
|             | <ul style="list-style-type: none"> <li>•事件句柄，由窗体中事件触发</li> </ul>                                              |
| MenuItem    |                                                                                                               |
|             | <ul style="list-style-type: none"> <li>•对菜单中对象描述</li> </ul>                                                   |
| MenuEvent   |                                                                                                               |
|             | <ul style="list-style-type: none"> <li>•事件句柄，由菜单click事件触发</li> </ul>                                          |

## 一、Application Object

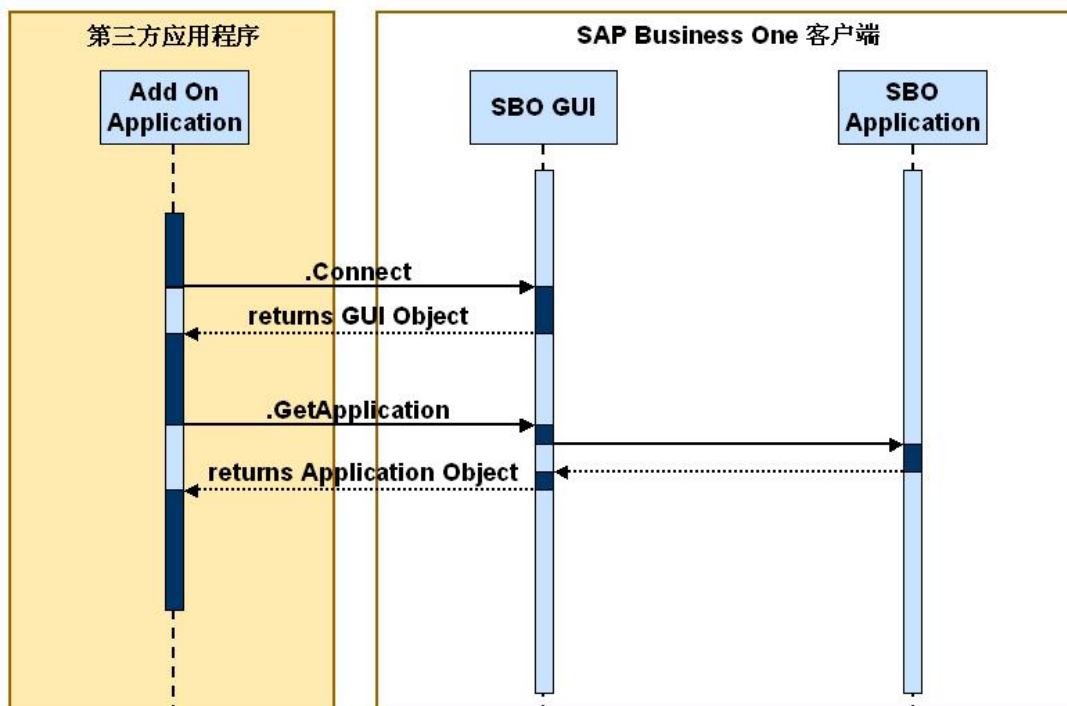
### 1、描述

Appcation Object 是 UI API 中的根对象，该对象描绘当前 SAP Business One 应用，通过 Appcation Object 对象实现对 Business One 应用的定制，Appcation Object 通过 SboGuiApp 对象取得。

通过 SAP Business One 应用中 AppEvent, ItemEvent 和 MenuEvent 句柄触发 Appcation Object 事件处理, 执行指定事件的逻辑功能。

第三方的应用实现与 SAP Buisness One 的客户端应用融合, 首先实例化 SboGuiApi, 再与 SAP Business One 应用建立连接; 其次, 通过 SboGuiApi Object 对象中 GetApplication 方法取得 Business One 应用实例; 最后通过 UI API 中提供的对象实现对界面层的操作。第三方 Add-On 与 Business One 集成应用实现的时序图如下:

## 2、图例



## 3、对象主要方法

|                                     |                                                  |
|-------------------------------------|--------------------------------------------------|
| <a href="#">ActivateMenuItem</a>    | 模拟菜单项 Click 事件                                   |
| <a href="#">GetFilter</a>           | 返回过滤事件                                           |
| <a href="#">GetLastBatchResults</a> | 返回最后一次调用 LoadBatchActions 方法结果字符串                |
| <a href="#">LoadBatchActions</a>    | 调用 XML 格式的文件, 生成 SAP Business One 中生成新对象(如窗体等对象) |
| <a href="#">MessageBox</a>          | 生成一个模态消息框                                        |
| <a href="#">SetFilter</a>           | 设置过滤事件, 对指定窗口事件过滤                                |
| <a href="#">SetStatusBarMessage</a> | 设置状态菜单中显示信息                                      |

## 4、代码

```
Private WithEvents SBO_Application As SAPbouiCOM.Application
```

```

Private oForm As SAPbouiCOM.Form
Private Sub SetApplication()
'*****
// 使用 SboGuiApi object to 建立连接
// 返回初始化 application object
'*****

Dim SboGuiApi As SAPbouiCOM.SboGuiApi
Dim sConnectionString As String
Set SboGuiApi = New SAPbouiCOM.SboGuiApi
// 通过下面的步骤实现连接
If Release =true Then
    sConnectionString = Split (Command)(0)
Else
    sConnectionString = “连接字符由 SAP 提供”
End If
// 连接到运行的 Business one 应用
SboGuiApi.Connect sConnectionString
//取得初始化 application object
Set SBO_Application = SboGuiApi.GetApplication()
End Sub

```

## 5、连接字符串说明

连接字符串(Connection string)是用于判别第三方应用程序(Add-On); 用于标识 SAP Business One 应用程序; 包含开发与运行 License 信息。UI API 提供两种连接模式实现第三方 Add-On 与 Business One 应用的连接。开发模式连接: 通过手动配置的方式在开发环境中维护连接字符串(connection string); 运行模式: SAP Business One 应用程序中直接提供连接字符串。

## 6、练习

- 6.1、使用 Visual Basic 创建一个新项目
- 6.2、定义连接 SAP Business One 需要的变量
- 6.3、连接 SAP Business One 应用程序主线程句柄
- 6.4、使用 MessageBox 在 SAP Business One 中显示消息

## 二、Form Object

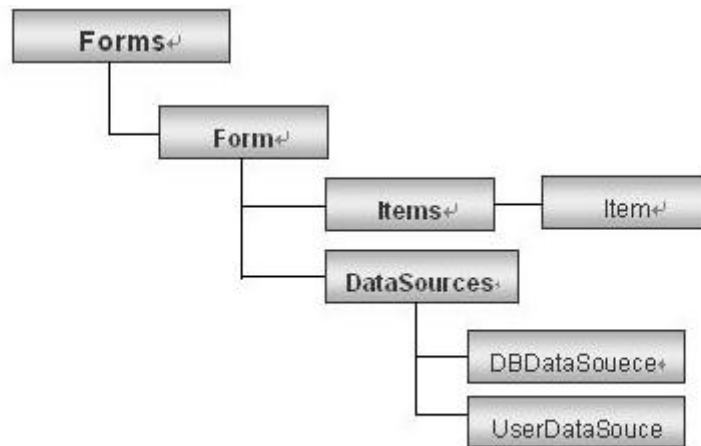
### 1、描述

窗体是 SAP Business One UI API 的重要组成部分, 可以通过 UI API 提供的 Form Object 实现对 SAP Business One 应用的窗体进行编辑, 进而实现对窗体中元素控制。根据实际项目的需要, 可以在系统中增加必要的窗体, 并实现与 Business One 系统的集成。

Business One 中每个窗体都被唯一标识, 通过 Form Object 中 Unique ID 辨别 Business One 中的窗体。Form

Object 中也保留有操作生成的数据，通过 Form Object 的 DataSources 属性进行访问。使用本对象可以实现对窗体的大小、位置、标题等属性进行编辑。通过 Form Object 的 Items 属性返回窗体中所含元素，再通过 Item Object 实现对窗体元素进行控制。

## 2、 图例



## 3、 对象主要方法

|                            |                |
|----------------------------|----------------|
| <a href="#">Close</a>      | 关闭窗体           |
| <a href="#">EnableMenu</a> | 设置窗体中工具栏菜单是否可用 |
| <a href="#">Freeze</a>     | 显示改变时设定窗体是否冻结  |
| <a href="#">GetAsXML</a>   | 将窗体生成 XML 资源文件 |
| <a href="#">Refresh</a>    | 刷新窗体           |
| <a href="#">Select</a>     | 激活窗体           |

## 4、 代码

### 4.1、 创建 Form

```

Public Sub CreateSimpleForm()
    Dim oForm As SAPbouiCOM.Form
    '// 增加一个新窗体
    Set oForm = SBO_Application.Forms.Add("SimpleForm", ft_Fixed)
    '//设定窗体的属性
    oForm.Title = "Simple Form"
    oForm.Left = 400
    oForm.Width = 300
    oForm.Top = 100
    oForm.Height = 100
    '//设定窗体的为可见状态
  
```

```
oForm.Visible = true
```

```
End Sub
```

#### 4.2、窗体中增加对象

```
Public Sub CreateSimpleFormItem()
```

```
Dim oForm As SAPbouiCOM.Form
```

```
Dim oItem As SAPbouiCOM.Item
```

```
Dim oButton As SAPbouiCOM.Button
```

```
Set oForm = SBO_Application.Forms.Add("SimpleForm", ft_Fixed)
```

```
oForm.Title = "Simple Form"
```

```
oForm.Left = 400
```

```
oForm.Width = 300
```

```
oForm.Top = 100
```

```
oForm.Height = 100
```

```
'''*****
```

```
''窗体中增加 Button 对象
```

```
''设定对象属性
```

```
'''*****
```

```
Set oItem = oForm.Items.Add("1", it_BUTTON)
```

```
oItem.Left = 5
```

```
oItem.Width = 65
```

```
oItem.Top = 50
```

```
oItem.Height = 19
```

```
Set oButton = oItem.Specific
```

```
oButton.Caption = "Ok"
```

```
Set oItem = oForm.Items.Add("2", it_BUTTON)
```

```
oItem.Left = 75
```

```
oItem.Width = 65
```

```
oItem.Top = 50
```

```
oItem.Height = 19
```

```
Set oButton = oItem.Specific
```

```
oButton.Caption = "Cancel"
```

```
End Sub
```

## 5、其它说明

SDK 中提供了专门进行“窗体”设计的开发工具 **Screen Painter**，它不依赖于其它开发环境，并集成到 SAP Business One 应用系统中。**Screen Painter** 可以独立完成窗体及窗体元素的开发；它可以将设计的窗体生成 XML 格式的文件，使得系统装载基于 XML 文件格式的窗体更加快捷，**Screen Painter** 进行“窗体”开发效率更高。

## 6、练习

6.1 在 SAP Business One 中创建一个新的窗体，窗体中包含以下信息：

供应商输入栏  
 供应商名称栏  
 报价单编号栏  
 报价数据栏  
 确认按钮  
 取消按钮

6.2 设置窗体中工具栏菜单上“发传真”和“导出到-MS\_EXCEL”功能不可用

## 三、Item Object

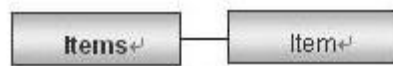
### 1、描述

在讲解 Item Object 前，首先看看 Items Collection。Items Collection 是 Form 中 Item Object 的集合，通过使用 Items Collection 中的 Object ID 或 Index 访问窗体中对应 Item Object 属性和方法。

Item Object 用于描述 SAP Business One 窗体中对象属性，使用 UniqueID 属性标识窗体中唯一 Item Object。使用 Item Object 维护 Item 属性，如大小、字符、位置等信息。

窗体中 Item Object 具有多种类型，使用 Visual Basic 进行窗体 Item Object 开发时，需要指定 Item 的类型(Item Types)，Item Object 主要有以下一些类型 Button、CheckBox、ComboBox、EditText、LinkButton、Matrix、OptionBtn、PictureBox、StaticText

### 2、图例



## 3、对象主要方法

### 3.1、Items Collection 方法

[Add](#)

窗体中增加一个新对象，如 Button、Static Text、Text Box 等

[Item](#)

通过 unique ID 或 index 返回对象

### 3.2、Item Object 方法

[Click](#)

激活 Item 对象 Click 事件

[Refresh](#)

使用数据库参数重新装载对象

[Update](#)

保存对象的数据，并重新装载该对象

## 4、 代码

```
Public Sub AddItemsToForm()
    Dim oForm As SAPbouiCOM.Form
    Dim oItem As SAPbouiCOM.Item
    Dim oButton As SAPbouiCOM.Button
    '// 增加新窗体
    Set oForm = SBO_Application.Forms.Add("SimpleForm", ft_Fixed)
    '*****
    '// 增加一个 Item 对象
    '*****
    Set oItem = oForm.Items.Add("1", it_BUTTON)
    oItem.Left = 5
    oItem.Width = 65
    oItem.Top = 50
    oItem.Height = 19
    Set oButton = oItem.Specific
    oButton.Caption = "测试 Button1"
    '*****
    '// 增加另一个 Button
    '*****
    Set oItem = oForm.Items.Add("2", it_BUTTON)
    oItem.Left = 75
    oItem.Width = 65
    oItem.Top = 50
    oItem.Height = 19
    Set oButton = oItem.Specific
    oButton.Caption = "测试 Button2"
End Sub
```

## 5 其它说明

此处无法设定 Item 项的数据绑定属性(Databind)，如需要设置要针对具体 Item 类型绑定

## 四、ItemEvent Object

使用 UI API 编写的第三程序，大多需要使用事件驱动，UI API 中支持三种类型事件触发 SAP Business One 中 Add-On 程序。

Application Event：应用程序关闭、公司之间切换、登陆语言改变、UI 服务停止事件

Menu Event：菜单事件

Item Event：窗体元素事件

在 UI API 中事件提供两种触发方法，响应前触发事件和响应后触发事件。

## 1、 描述

UI 事件, 如 Click、Form Load 等事件在窗体或窗体中 Item 对象发生。Item Events 如 LostFocus、GotFocus、FormActivate、FormLoad。

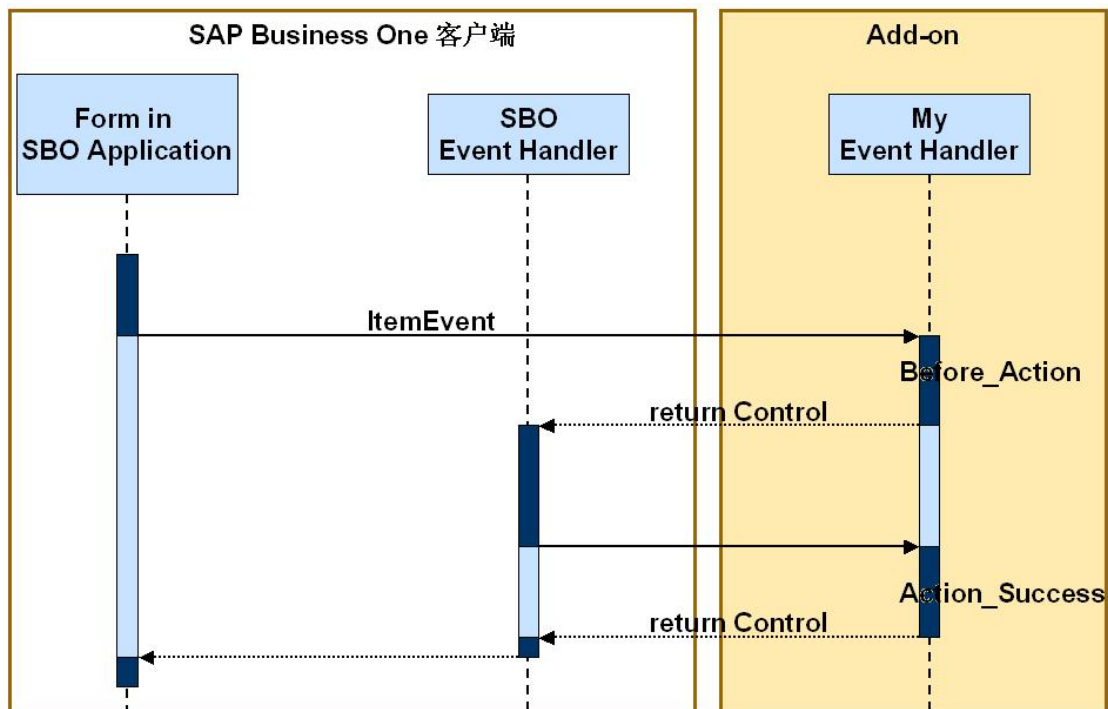
```
Public Event ItemEvent( _
    ByVal FormUID As String, _
    ByVal pVal As ItemEvent, _
    ByRef BubbleEvent As Boolean _
)
```

FormUID 指定事件发生窗体 Unique ID

PVal 事件发生对象控制参数

BubbleEvent 事件处理控制参数

## 2、 图例



## 3、 代码

'//通过事件在业务伙伴主数据增加一个 Button 弹出导入数据的窗口

```
Private Sub vAppl_ItemEvent(ByVal FormUID As String, pVal As SAPbouiCOM.IItemEvent, BubbleEvent
As Boolean)
    Dim strpath As String
    Dim editt As SAPbouiCOM.EditText
    Dim Form As SAPbouiCOM.Form
    If pVal.FormType = 134 Then
        If pVal.EventType = et_CLICK Then
            If pVal.ItemUID = "addPartner" Then
```

```

        Call CreateFormForPath
    End If
End If
If pVal.FormUID = "EXCEL14" Then
    If pVal.EventType = et_CLICK Then
        If pVal.ItemUID = "b1" Then
            Set editt = vAppl.Forms("EXCEL14").Items("filepath").Specific
            strpath = editt.String
            If strpath <> "" Then
                Call ReadEXCEL(strpath)
            Else
                MsgBox "请输入 EXCEL 文件路径！"
            End If
        End If
    End If
End If
End Sub

```

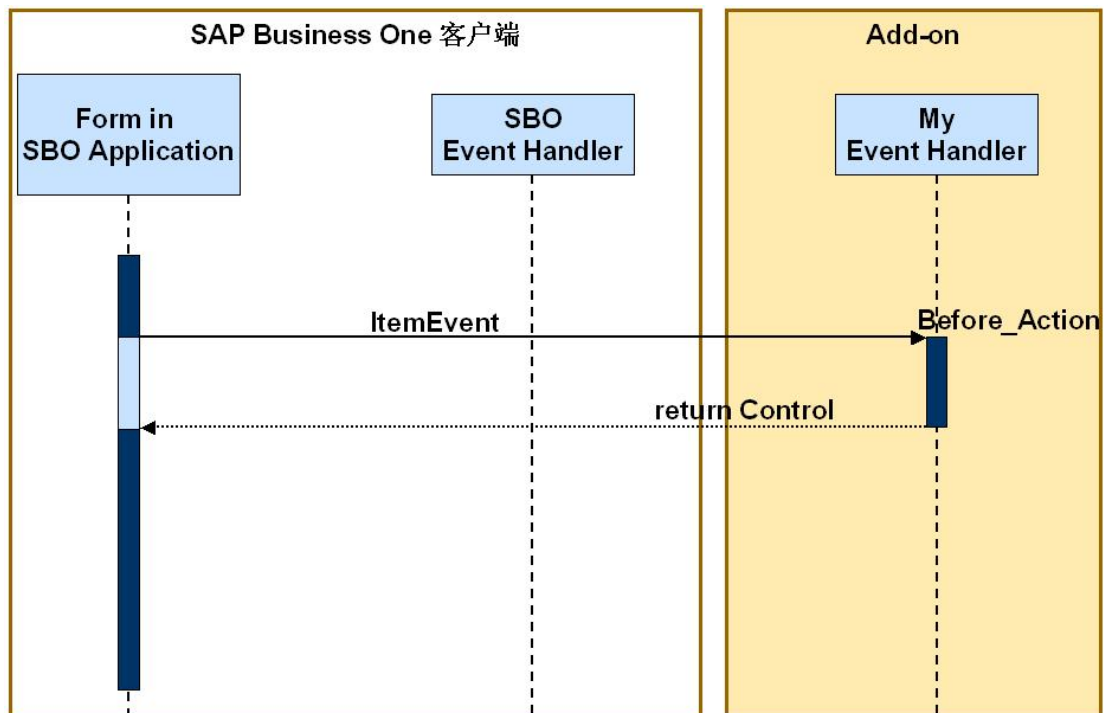
## 五、BubbleEvent

### 1、描述

BubbleEvent 用于控制 SAP Business One 应用事件的执行，如果参数值为“True” SAP Business One 应用程序将按照顺序执行应用系统事件和自定义的特殊事件。缺省模式下该参数值为 True；如果 BubbleEvent 参数值为“False”，停止执行系统事件直接执行自定义事件，当 SAP Business One 的“窗体”和 Items 中增加自己的事件处理功能替代系统中的事件时，BubbleEvent 的参数值设置为 False。

### 2、图例

BubbleEvent 为 False 参数情况下的时序图：



### 3、代码

```

'// BubbleEvent 在 False 和 True 参数下的执行情况
Private Sub con_ItemEvent(FormUID As String, pVal As SAPbouiCOM.ItemEvent, BubbleEvent As Boolean)
    Select Case pVal.FormUID
        '// 报价单事件处理
        Case C_FORMUID_SALES_QUOTATION
            Call eventHandlerSalesQuotation(formPtr, pVal, _ BubbleEvent)
        '// 订单事件处理
        Case C_FORMUID_SALES_ORDER
            Call eventHandlerSalesOrder(formPtr, pVal, _ BubbleEvent)
        '// 其它事件处理
        Case Else
            '//Business One 事件处理
            BubbleEvent = True
        End Select
    End Sub

```

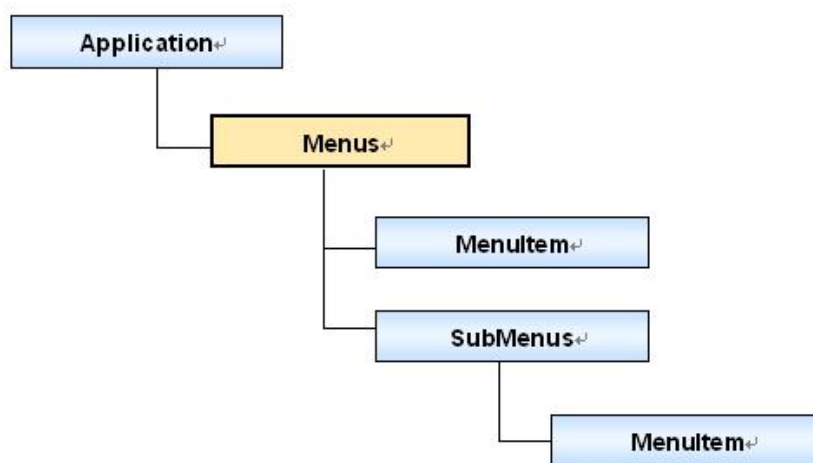
## 六、Menuitem Object

### 1、描述

在讲解 Menu item Object 之前，先看看 Menus Collection 对象。Menus Collection 是所有菜单项的集合，可以通过 Menus Collection 中 Add 和 Remove 方法增加新的菜单项或删除已存在的菜单项。

Menu item object 是对某个具体的菜单项进行描述，可以通过 Activate 方法激活增加的一个菜单，通过 Check、Enabled、string、Type 等属性实现对菜单属性的控制。通过 UID 属性唯一标识菜单项。

### 2、图例



### 3、对象主要方法与属性

#### 3.1 方法

[Activate](#)      激活菜单项 Click 事件

#### 3.2 属性

[Checked](#)      返回或设置菜单项是否被选择  
[Enabled](#)      返回或设置菜单项是否可用  
[String](#)      返回或设置菜单项字符串信息  
[SubMenus](#)      返回菜单项的子菜单  
[Type](#)      返回或设置菜单类型  
[UID](#)      菜单唯一编号 UID

## 4、代码

```
// 添加新菜单
Dim oMenus As SAPbouiCOM.Menus
Dim oMenuItem As SAPbouiCOM.MenuItem
Set oMenus = pApp.Menus
// 取得第四个菜单项
Set oMenuItem = oMenus.Item(3)
Set oMenus = MenuItem.Submenus
// 添加 popup 菜单
Set oMenuItem = oMenus.Add("MyMenu1", "User Forms", _
    mt_POPUP, oMenus.Count)
Set oMenus = oMenuItem.SubMenus
oMenus.Add("MyMenu2", "User Form", mt_STRING, 0)
```

## 5、其它说明

如果在 SAP Business One “模块”根菜单下添加菜单项，启动 SAP Business One 后，增加的菜单自动被加载到“模块”菜单中，目前第三方添加的菜单还不能给对应菜单一个标识图片，而且客户还不能对顶级菜单进行修改，如增加、删除菜单。

## 6、练习

6.1 在“模块”菜单中增加一个菜单项，并在该菜单下增加一个子菜单

## 七、Menu Event Object

### 1、描述

UI 事件，使用 Menu Event Object 响应菜单项的事件，如在菜单项进行 Click 操作。

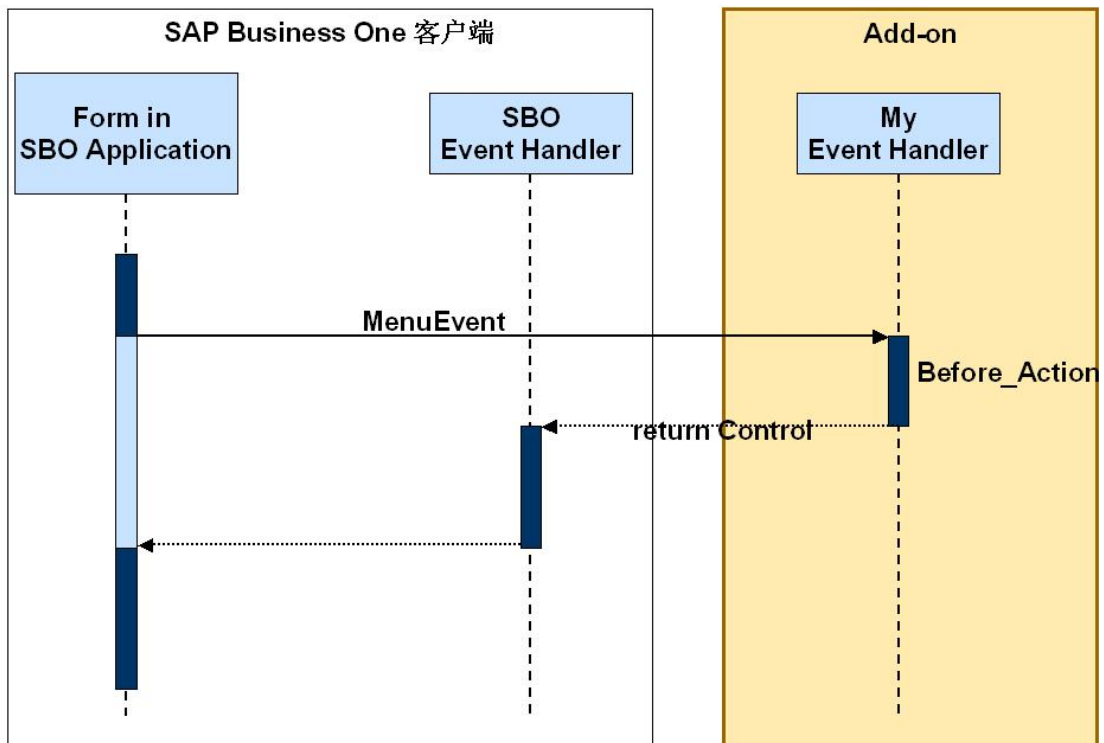
```
Public Event MenuEvent( _
    ByVal pVal As MenuEvent, _
    ByRef BubbleEvent As Boolean _
)
```

PVal 事件发生对象控制参数

BubbleEvent 布尔型返回参数，为 True 值系统响应自定义的菜单事件；为 False 值按照顺序响应系统事件和自定义事件。

## 2、图例

BubbleEvent 为 True 情况下菜单事件处理时序图：



## 3、对象主要属性

|                              |                                     |
|------------------------------|-------------------------------------|
| <a href="#">BeforeAction</a> | 返回值说明本事件是在 SAP Business One 之前或之后执行 |
| <a href="#">InnerEvent</a>   | 返回值判断是否为内部事件                        |
| <a href="#">MenuUID</a>      | 返回触发事件菜单编号                          |

## 4、代码

```
Private Sub SBO_Application_MenuEvent(pVal As SAPbouiCOM.IMenuEvent, BubbleEvent As Boolean)
'*****// 通过事件处理，使用自定义的窗口代替 Business One 窗体

' // 将 Bubble Event 设置为 True
'***** If
pVal.BeforeAction = True Then
    SBO_Application.SetStatusBarMessage _
        "Menu item: " + pVal.MenuUID + " sent an event BEFORE SAP Business One processes it.", bmt_Long, True
' //设置 BubbleEvent 停止 Business One 系统事件响应
```

```

BubbleEvent = True
Else
    SBO_Application.SetStatusBarMessage _
        "Menu item: " & pVal.MenuUID & " sent an event AFTER SAP Business One processes it.", bmt_Long, True
End If
End Sub

```

## 5、练习

5.1 上一节生成的子菜单中添加事件，点击子菜单打开用户窗体(第一节创建的窗体)

## 八、XML 使用

### 1、描述

SAP Business One 便捷地实现使用 XML 格式开发处理，可以将“窗体”导出生成 XML 格式文件，根据自己的需要对“窗体”进行修改，再通过 UI API 提供的方法导入系统，生成定制的用户“窗体”。通过 XML 方式极大降低了开发工作量，同时，通过批导入的方法提高系统响应效率。

### 2、图例

```

<?xml version="1.0" encoding="UTF-16" ?>
- <Application>
- <forms>
- <action type="add">
- <form appformnumber="2000000119" color="0" default_button="" height="325" left="336" pane="0" title="" top="44"
uid="XmlUserForm_0" visible="0" width="576">
+ <datasources>
<menus />
- <items>
    <action type="add">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="19" left="5" linkto="" right_just="0"
tab_order="0" to_pane="0" top="273" type="4" uid="1" visible="1" width="65">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="19" left="75" linkto="" right_just="0"
tab_order="0" to_pane="0" top="273" type="4" uid="2" visible="1" width="65">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="19" left="145" linkto="" right_just="0"
tab_order="0" to_pane="0" top="273" type="4" uid="MatUpBtn" visible="1" width="65">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="49" left="5" linkto="" right_just="0"
tab_order="0" to_pane="0" top="2" type="100" uid="Rect1" visible="1" width="321">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="14" left="157" linkto="" right_just="0"
tab_order="0" to_pane="0" top="8" type="16" uid="EditText1" visible="1" width="163">
+ <Item description="" disp_desc="1" enabled="1" from_pane="0" height="14" left="157" linkto="" right_just="0"
tab_order="0" to_pane="0" top="24" type="113" uid="ComboBox1" visible="1" width="163">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="14" left="7" linkto="EditText1" right_just="0"
tab_order="0" to_pane="0" top="8" type="8" uid="StaticTxt1" visible="1" width="148">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="14" left="7" linkto="ComboBox1"
right_just="0" tab_order="0" to_pane="0" top="24" type="8" uid="StaticTxt2" visible="1" width="148">
+ <Item description="" disp_desc="0" enabled="1" from_pane="0" height="210" left="5" linkto="" right_just="0"
tab_order="0" to_pane="0" top="55" type="127" uid="Matrix1" visible="1" width="557">
    </action>
    </items>
    </form>
    </action>
    </forms>
</Application>

```

### 3、常用方法

#### 3.1、生成 XML 文件

将 Form 生成 XML 文件，通过 Form Object 中 GetAsXML 方法实现

**Public Function GetAsXML() As String**

#### 3.2、装载 XML

将 XML 文件装载入系统生成“窗体”，通过 Application Object 中 LoadBatchActions 方法实现

**Public Sub LoadBatchActions(ByRef [xmlStr](#) As String )**

### 4、代码

```
'//生成报价单窗体 XML，修改后生成新的报价窗体
Dim oXMLDoc As New MSXML2.DOMDocument
Dim frm As SAPbouiCOM.Form
Dim xmlData As String
Dim m_TemplateFileForm_01 As String
m_TemplateFileForm_01 = "d:\xmlTemplates\template_file1.xml"
Set frm = m_app.Forms.GetFormByTypeAndCount(149, 1)
If frm Is nothing Then
    Exit Sub
End If
'//生成报价单 XML 文件
xmlData = frm.GetAsXML
oXMLDoc.loadXML sXMLString
oXMLDoc.save (App.Path & "\forms\" & sFileName & ".xml")
'//将修改 XML 文件装载入系统
oXMLDoc.Load (m_TemplateFileForm_01)
m_oApplication.LoadBatchActions oXMLDoc.xml
Set oXMLDoc = Nothing
```

### 5、练习

5.1 将 SAP Business One 系统中采购订单生成 XML 文件，修改窗体标题为“XX 公司采购订单”

5.2 使用生成的 XML 文件创建新采购订单窗体

## 九、DataBind Object

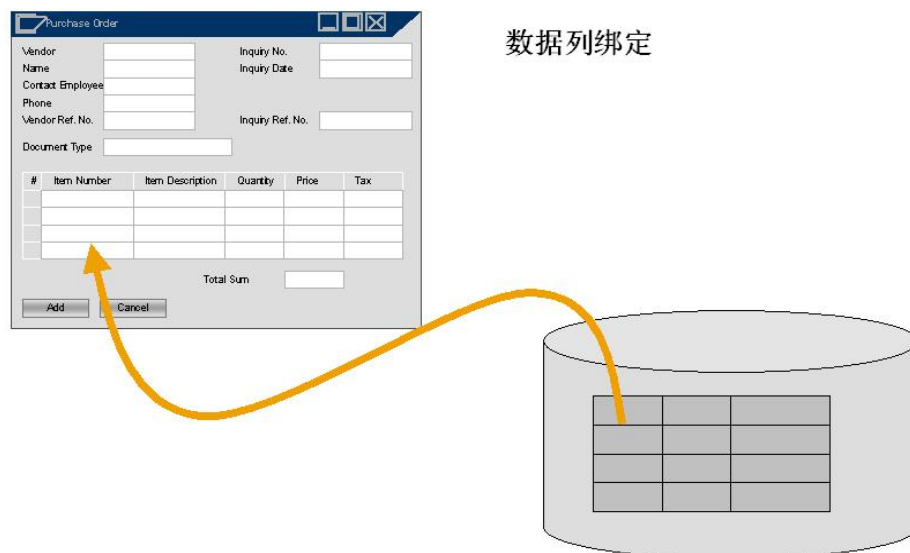
### 1、描述

通过 DataBind Object 实现数据源与“窗体”或窗体中元数的对应，数据源(DataSource)是窗体数据存放的容器，窗体或窗体中元数不必绑定到数据库层(Database)。通过与数据源绑定可以提高系统执行效率，无须经常更改用户接口(user interface)提供经常使用的数据。

DataSource Object 的有两种类型：DBDataSource 和 UserDataSource。其中 DBDataSource 实现与数据库中表的绑定，UserDataSource 是窗体元数数据存放的容器，这些数据没有物理存放于数据库中。

### 2、图例

下图直观地显示了数据库列与窗体中矩阵的绑定原理



### 3、对象主要方法

[SetBound](#)

设置绑定对象到用户定义的窗体元素上

### 4、操作步骤

实现数据与界面元数绑定一般遵循以下步骤

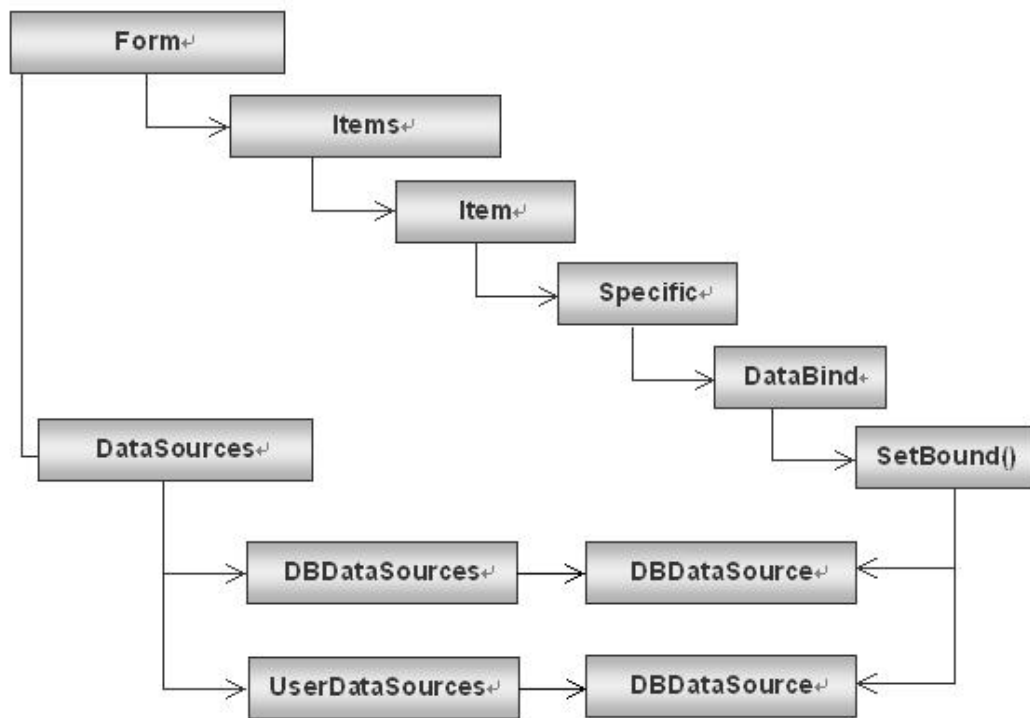
定义窗体(Define Form )

窗体添加数据源 (Add data source)

设定窗体元数与数据源的绑定

从数据源中取得数据

具体实现过程如下图：



## 5、代码

### 5.1、使用 UserDataSource 绑定

```

Public Sub BindItemToDataSource()
    Dim oForm As SAPbouiCOM.Form
    Dim oEditText As SAPbouiCOM.EditText
    '// 增加一个新窗体
    Set oForm = SBO_Application.Forms.Add("SimpleForm")
    '// 在窗体中添加 User Data Source
    oForm.DataSources.UserDataSources.Add "EditSource", dt_SHORT_TEXT, 20
    '// 增加 Text Edit 窗口元素
    Set oEditText = oForm.Items.Add("EditText1", it_EDIT).Specific
    '// 将 userDataSource 绑定到 text edit
    oEditText.DataBind.SetBound True, "", "EditSource"
End Sub
  
```

### 5.2、使用 DBDataSource 绑定

```

Public Sub BindMatrixToDataSource()
    Dim oDBDataSource As SAPbouiCOM.DBDataSource
    Dim oUserDataSource As SAPbouiCOM.UserDataSource
    Dim oMatrix As SAPbouiCOM.Matrix
    Dim i As Long
  
```

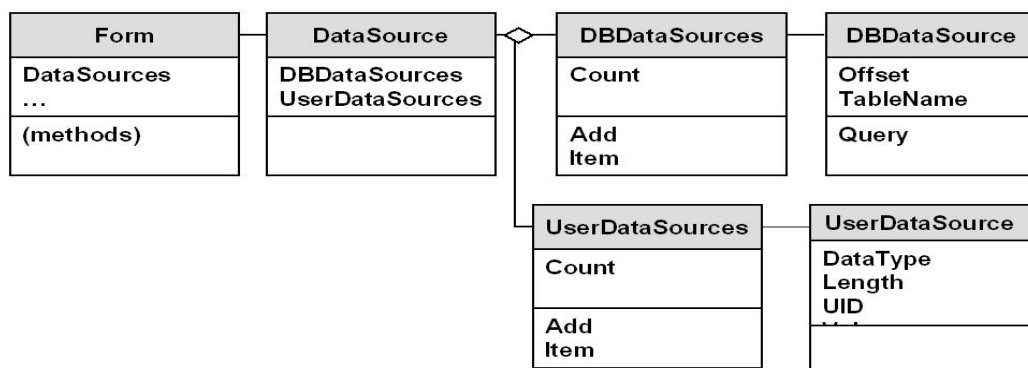
```

// 在窗体中添加 DBDataSource
Set oDBDataSource = Form.DataSources.DBDataSources.Item("OUSR")
Set oMatrix = Form.Items.Item("Matrix1").Specific
oMatrix.Clear
// 查询数据
oDBDataSource.Query
// 将数据绑定到窗体元素矩阵中
For i = 0 To oDBDataSource.Size - 1
    oDBDataSource.Offset = i
    oMatrix.AddRow
Next i
End Sub

```

## 6、其它说明

下类图显示了数据绑定 DataBind Object 与其它相关对象的关系



数据绑定属性的设置需要针对窗体中元素具体类型进行。

## 7、练习

- 6.1 对第一节创建的窗体中增加 **DataSource**，使用数据库中 **OQUT** 表以及其中数据
- 6.2 使用 **SetBound** 方法将窗体中元素绑定到相应的字段上
- 6.3 从 **DataSource** 中取得数据，并在定义的窗体元素中显示

## 第四节 应用同步处理

为了保证使用 UI API 开发的功能与 SAP Business One 系统能协同工作，提高 SAP Business One 系统与第三方 Add-On 系统内聚性。在使用 UI API 开发过程中，要从多个方面控制保证系统的同步，下文介绍两种系统同步的控制方法。

### 一、文档控制

文档控制对一个项目的成败起至关重要的重用，为项目后续阶段的分析提供可靠的数据依据。在使用 UI

API 开发过程中必须进行文档记录,避免 Add-On 解决方案之间的冲突,避免 Add-On 解决方案与 SAP Business One 应用之间的冲突。当出现 Add-On 之间或 Add-On 与 SAP Business One 应用冲突时,能通过文档迅速诊断原因。

在系统分析、系统设计和系统开发过程中,应详细记载事件、窗体、窗体元素的处理过程和处理方法。在案例实施过程中,从以下几个步骤考虑应用系统的同步问题:

预期情况分析,它是项目开展的先决条件。使用 UI API 开发前需要对项目背景、项目需求、开发风险、通过 UI API 需要实现的功能、其中使用到 UI API 的 Objects 的情况进行记载;实现过程处理。使用 SAP Business One 提供的 SDK 按照预期的设计和预计使用的 UI API 的 Objects 实现解决方案。在解决方案实现过程中,如果出现了与预期设计的偏离,需要通过文档进行记录,同时对预期设计的文档资料进行变更;在解决方案中涉及到事件的处理,应提前进行设计和定义,以保证解决方案事件与 SAP Business One 事件的协同处理。如通过设置 Bubble Event 参数的方法保证整个应用的事件同步;其它因数考虑,如解决方案打包等情况,这也是保证 Add-On 解决方案与 SAP Business One 同步的必要因数。

## 二、强制事件处理

为了保证应用同步,除了上文的介绍进行过程文档控制外,结合 SAP Business One 操作功能需要对一些特殊操作进行特别控制,如系统语言改变(Language Change)、公司切换(Company Change)、事件关闭(Shutdown Event)等。

### 1、语言集改变

当 SAP Business One 系统中语言发生变化(Interface Language was Changed),SAP Business One 中“模块”菜单、Add-On 菜单和具体菜单项将被重新装载,解决方案需要捕获语言改变事件,并改变 Add-On 中的新增加的菜单、界面和界面元素的语言模式。

### 2、公司改变

在解决方案处理过程,对操作公司发生改变(Company Database was Changed)的情况根据实际需要进行控制。进行公司切换时捕获事件,处理在新公司下的业务逻辑,如果 Add-On 功能无须在新公司中使用,在事件中终止 Add-On 功能。

### 3、关闭事件

在 Add-On 程序中可以对系统关闭事件(Shutdown Event)进行响应,添加关闭前必要的逻辑检查,并且根据需要重新启动 Add-On 程序。关闭事件处理,一方面纠正 SDK6.2 版本中的处理习惯,另一方面为 SDK6.7 奠定基础, SDK6.7 版本在 Add-On 程序中必须捕获关闭事件,卸载菜单并关闭 Add-On 功能。

### 4、AppEvent Event

上文介绍的强制事件处理通过 AppEvent 事件处理,AppEvent Event 主要包括以下内容控制: Application Shutdown 、 Company Database was Changed 、 Interface Language was changed、 UI Server Shutdown。AppEvent

Event 处理语法如下：

```
Public Event AppEvent( _
    ByVal EventType As BoAppEventTypes _
)
```

参数 EventType:

| 组成                    | 描述                    | 值  |
|-----------------------|-----------------------|----|
| aet_ShutDown          | 应用程序关闭                | 64 |
| aet_CompanyChanged    | 公司切换                  | 65 |
| aet_LanguageChanged   | 语言集改变                 | 66 |
| aet_ServerTermination | SAP Business One 服务终止 | 67 |

## 5、示例代码

```
Private Sub SBO_Application_AppEvent(ByVal EventType As SAPbouiCOM.BoAppEventTypes)
' //*****
' // 下面是 Application 发出的事件
' // 在对应的事件处理中增加需要处理的逻辑
' //*****
    Select Case EventType
        Case aet_ShutDown:
            SBO_Application.MessageBox "A Shut Down Event has been caught" _
                & vbNewLine & "Terminating Add On..."
            ' //*****
            ' // 此处处理其它逻辑
            ' //*****
        End
        Case aet_CompanyChanged:
            SBO_Application.MessageBox "A Company Change Event has been caught"
            ' //*****
            ' // 此处处理其它逻辑
            ' //*****
        Case aet_LanguageChanged:
            SBO_Application.MessageBox "A Language Change Event has been caught"
            ' //*****
            ' // 此处处理其它逻辑
            ' //*****
    End Select
End Sub
```

## 第五节 UI API 高级特性

### 一、Working with Matrix

#### 1、在 Form 上添加 Matrix

```
Set oItem = oForm.Items.Add("Matrix1", it_MATRIX)
Set oMatrix = oItem.SpecificSet oColumns = oMatrix.Columns
Set oColumn = oColumns.Add("MyCol1", it_EDIT)
oColumn.TitleObject.Caption = "My Column1"
oColumn.Width = 30oColumn.Editable = False
Set oColumn = oColumns.Add("MyCol2", it_EDIT)
oColumn.TitleObject.Caption = "My Column2"
oColumn.Width = 30oColumn.Editable = False
```

#### 2、用 UserDataSource or DBDataSource 绑定数据到 Matrix

1) 首先要在 Form 上添加 UserDataSource 和 DBDataSource

```
oForm.DataSources.DBDataSources.Add ("OCRD")
```

oForm.DataSources.UserDataSources.Add "EditSource", dt\_SHORT\_TEXT, 202) 然后通过 SetBound 来绑定刚才定义的数据源到 Matrix 中的列

**Set oMatrix = oForm.Items.Item("Matrix1").Specific**

**Set oColumns = oMatrix.Columns**

**'Bind to DBDataSource**

```
Set oColumn = oColumns.Item("MyCol1")
```

```
oColumn.DataBind.SetBound True, "OCRD", "CardCode "
```

**'Bind to UserDataSource**

```
Set oColumn = oColumns.Item("MyCol2")
```

```
oColumn.DataBind.SetBound True, "", "EditSource
```

#### 3、装载数据源中的数据到Matrix

```
Set oDBDataSource = oForm.DataSources.DBDataSources.Item("OCRD")
```

```
Set oUserDataSource = oForm.DataSources.UserDataSources.Item("EditSource")Set oMatrix =
```

```
oForm.Items.Item("Matrix1").SpecificoMatrix.Clear
```

```
oDBDataSource.Query
```

```
oUserDataSource.Value = "TestData "
```

```
For i = 0 To oDBDataSource.Size - 1
```

```
oDBDataSource.Offset = i
```

```
oMatrix.AddRow
```

```
Next i
```

#### 4、更新Matrix中的数据

首先，我们 GetLineData 得到 Matrix 列中的数据，在修改完数据后，通过 SetLineData ( ) 函数在将数据写入到 Matrix 中。

注意：这个操作是有限的，它只能更新数据源中的用户自定义数据源：即 `UserDataSource`。

例如：`Set oMatrix = oForm.Items.Item("Matrix1").Specific`  
`Set oColumn = oMatrix.Columns.Item("MyCol2")`  
`Set UDS = oForm.DataSources.UserDataSources.Item("EditSource")`  
`oMatrix.GetLineData (LineNum)`  
`UDS.Value = "My New Value"`  
`oMatrix.SetLineData (LineNum)` 在原有的 Matrix 中增加列在原有的 Matrix 中，一般都会有一个默认的空行，但是 `UI_API` 只允许我们在空的 Matrix 中添加列。这种情况下，我们不能在原有的 Matrix 中添加列。解决办法是，我们要捕捉 `et_FORM_LOAD` 事件，删除 Matrix 中的数据空行，然后添加列，并且恢复原来的空行，如下代码所示：

```
' 1) 在 form load 中捕捉 et_FORM_LOAD 事件
If (pVal.EventType = et_FORM_LOAD And pVal.FormType = 139 And _
    pVal.Before_Action = False) Then
    Set oForm = m_App.Forms.Item(pVal.FormUID)
    Set oItem = oForm.Items.Item("38")
    Set oMatrix = oItem.Specific
    ' 2) Remove the line from the matrix
    oMatrix.DeleteRow (1)
    ' 3) Add the new column
    Set oColumn = oMatrix.Columns.Add("MyCol", it_EDIT)
    oColumn.TitleObject.Caption = "TestCol "
    oColumn.Width = 90
    ' 4) Add the first line again
    oMatrix.AddRow
    oForm.Refresh
End If
```

## 6、练习

- 创建一个 Vb 工程，然后创建一个 Form，并在里面添加 Matrix 项目
- 在窗体中添加数据源
- 将数据源绑定到 Matrix 中的列中
- 显示数据到 Matrix 中

## 二、屏幕制作器

### 1、描述

屏幕制作器(Screen Painter)是用于界面设计与部署的开发工具包，是 SAP Business One Application 的一个 Add-On 功能，它能极大提高界面开发效率。使用屏幕制作器生成的界面与 SAP Business One 应用程序的界面风格高度统一。SAP Business One 7.1 版本程序中直接提供了屏幕制作器开发工具，通过“模块→开发工具→屏幕制作器”路径进入屏幕制作器功能，此时屏幕制作器的主菜单替换 SAP Business One 的应用主菜单，

可以直接进行系统界面开发。

屏幕制作器是一个相对独立的开发工具包，它不依赖其它开发环境；通过屏幕制作器生成的窗体符合XML标准，SAP Business One 装载窗体效率较高。由屏幕制作器生成的窗体文件，可以直接修改生成新的窗体文件，极大减低开发工作量。

## 2、屏幕制作器组成

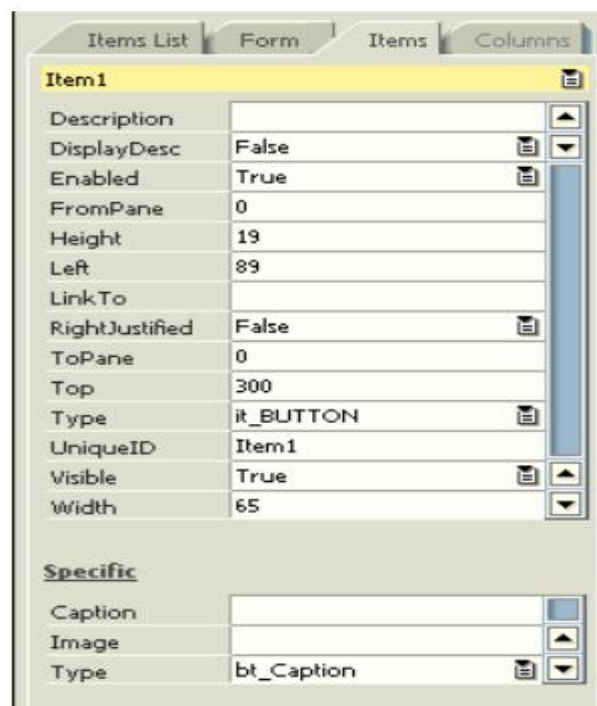
屏幕制作器(Screen Painter)主要有以下几个部分组成：工具箱、属性编辑器、新窗体。

工具箱：窗体元素，如矩阵(it\_MATRIX)、静态文本(it\_STATIC)、输入框(it\_EDIT)、选择框(it\_CHECK\_BOX)、按钮(it\_BUTTON)、窗体元素对齐等工具



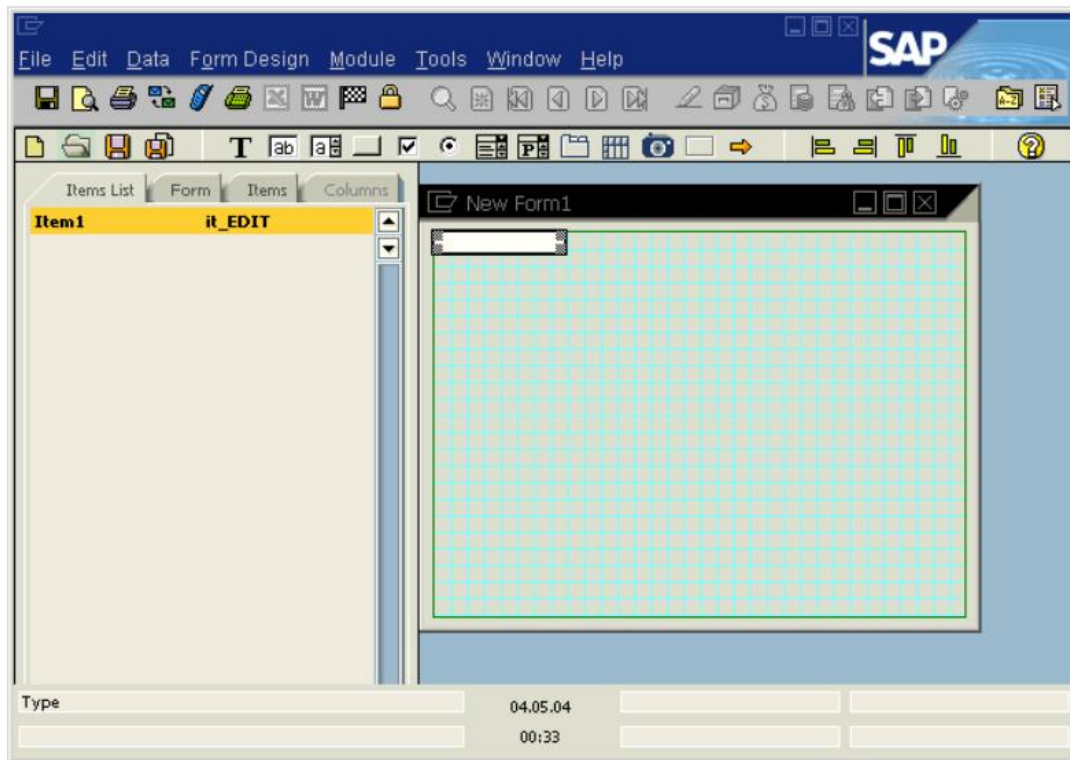
使用工具箱能迅速地创建“窗体”，增加并布置窗体元素，导入或导出设计的窗体。

属性编辑器：实现对窗体和窗体元素属性的定义或修改



上图可以看出属性设置有两种类型：一种是通用属性，它对窗体中所有元素都适用；另一种是特殊属性，它针对每个独特的元素设置相应的特征。窗体元素可见属性控制与一般设计工具存在一定的差异，窗体元素是否可见通过窗体中“Pane Level”实现，如果当前窗体“Pane Level”处于“From Pane”和“ToPane”之间处于窗体中的元素将可见，否则为不可见状态，如果窗体元素“From Pane”=0 并且“ToPane”=0，这样窗体元素将不受“Pane Level”控制，始终为可见状态。

新窗体：系统生成一个窗体模板，在此基础上进行窗体加工。



### 3、屏幕制作器使用步骤

#### 3.1、生成窗体文件

运行 SAP Business One 屏幕生成器，编辑窗体，部署窗体元素。设计完成后保存生成\*.srf 格式文件。

#### 3.2、运行窗体文件

开发工具中直接调用使用屏幕生成器产生的文件，在 SAP Business One 应用程序中生成窗体，参考代码如下：

##### 3.2.1、装载窗体文件

```
Private Sub m_App_MenuEvent(pVal As SAPbouiCOM.IMenuEvent, BubbleEvent As Boolean)
    Dim oForm As SAPbouiCOM.Form
    Dim oXMLDoc As New MSXML2.DOMDocument
    If (pVal.MenuUID = MY_MENU_ENTRY And pVal.BeforeAction = True) Then
        oXMLDoc.Load (App.Path & "\TrainingForm.srf")
        m_App.LoadBatchActions oXMLDoc.xml
        Set oForm = m_App.Forms.Item("MySrfForm")
    End If
End Sub
```

##### 3.2.2、运行窗体

```
Public oAddOn As CCompany
Const MY_MENU_ENTRY = "mn1"
Private Sub CmdAddToMenu_Click()
    Dim m_menus As SAPbouiCOM.Menus
```

```

Dim m_menuItem As SAPbouiCOM.MenuItem
Set m_menus = oAddOn.m_App.Menus
Set m_menus = m_menus.Item(4).SubMenus
Set m_menus = m_menus.Item(3).SubMenus
Set m_menuItem = m_menus.Add(MY_MENU_ENTRY, "MY Srf Form", mt_STRING, 1)
CmdRemoveFromMenu.Enabled = True
CmdAddToMenu.Enabled = False
End Sub

```

## 4、其它说明

目前使用 SAP Business One 7.1 版本 Screen Painter 导出的文件有时存在一些问题，需要使用记事本手动打开文件 srf，定位文件尾部，删除空格字符。同时，检查生成的 srf 文件中是否含有“<”、“>”、“&”等特殊字符，因为系统装载 srf 文件生成窗体时，对于以上的特殊字符系统无法处理，需要对这些特殊字符进行修正。

## 5、练习

- 5.1 使用 Screen Painter 绘制“客户询价单”
- 5.2 在 SAP Business One 模块菜单中增加询价单菜单，并定义 Click 事件
- 5.3 装载“客户询价单”XML 文件，在系统中生成询价窗体

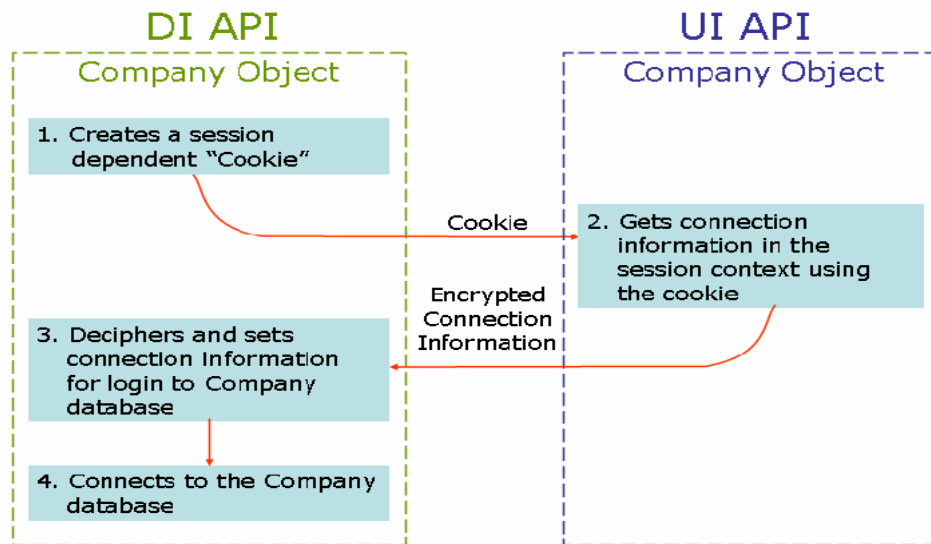
## 三、 Single Sign-On

### 1、上下文

SAP Business One 的 UI API 同 DI API 作为独立的两个 SDK。它们分别提供各自的连接方法，参见 Connecting to the UI API 和 Connecting to the DI API。但是许多 AddOn 需要同时使用 UI API 与 DI API，并希望两者使用相同的会话信息（例如公司、用户等）。在 SDK6.5 以前没有一种机制桥接 UI API 与 DI API。开发人员只能使用一些特殊的方法获取当前的会话信息然后连接 DI API。现在 SAP Business One SDK6.5 中可以通过 Single Sign-On 连接 UI API 与 DI API 了。

### 2、描述

通过 Single Sign On 机制，开发人员现在可以使用当前登录 SAP Business One 的用户认证信息创建 DI API 连接了。整个连接过程为通过 DI API 的 Company 接口创建一个 cookie，Cookie 代表当前 DI API 的会话信息，并提供给 UI API 作为密钥。UI API 将用户认证信息采用 cookie 加密后以连接字符串的形式返回。DI API 解密后使用用户认证信息创建连接。如图所示：



### 3、用法及示例

连接步骤

1. SetApplication: 连接到 SAP Business One 应用程序 (UI)
2. SetConnectionContext
  - a) 定义 DI 的 Company 对象
  - b) 通过 DI 的 Company 对象获取 Cookie
  - c) 通过 UI 的 Company 对象的 GetConnectionContext 获取连接上下文
  - d) 确保 DI 不处于连接状态
  - e) 通过 SetSboLoginContext 方法将连接上下文传给 DI API

#### 3. 连接到公司数据库 (DI)

代码示例:

```

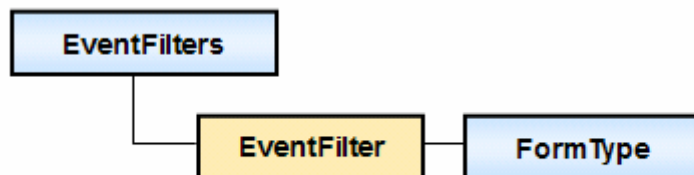
'//After connecting to UI, but before connecting to DI, do this:
'//Acquire the connection context cookie from the DI API
Dim sCookie As String
sCookie = oDICompany.GetContextCookie
'//Retrieve the connection context string from the
'//UI API using the acquired cookie.
Dim conStr As String
conStr = oUIApp.Company.GetConnectionContext(sCookie )
'//Set the connection context information to the DI API.
ret = oDICompany.SetSboLoginContext(conStr)
If Not ret = 0 then
    Exit Sub 'the operation has failed.
End If
'//Establish the connection to the company database.
Ret = oDICompany. Connect()
  
```

## 四、 Filtering Events

### 1、描述

SAP Business One 的 AddOn 采用独立的进程。虽然这样应用程序更为安全，但 SBO 应用程序同 AddOn 的通信都采用了 COM 的跨进程通信，如果频繁交互必然会影响性能。SDK 以前也采用了一些机制，例如通过 xml 创建 Form 减少交互来提高性能。在默认情况下，SBO 应用程序会将所有事件通知 AddOn 应用程序，而不管用户是否关心这些事件。即使用户不处理这些事件，也会给 SBO 同 AddOn 的性能造成很大的压力。SDK6.5 提供了事件过滤器，通过注册用户关心的事件，SBO 仅在这些事件发生时通知 AddOn，这样极大的改善了性能。AddOn 的开发人员应该有责任认真使用事件过滤器提高性能。

事件过滤器相关类结构：



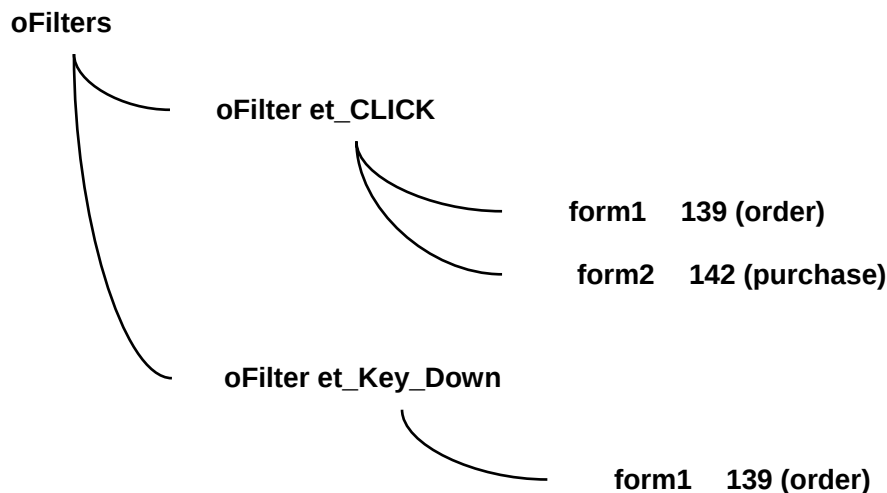
### 2、用法及示例

使用步骤：

1. 创建 EventFilters 集合
2. 在 EventFilters 集合中添加 EventFilter
  - a) 定义需要处理的事件类型
  - b) 定义需处理那个窗体的事件
3. 重复步骤 2 直到所有感兴趣的事件都加入 EventFilters
4. 激活 EventFilters 集合使其生效（通过 application 的 SetFilter 方法）

一旦定义了事件过滤器，只有声明的事件（指定窗体类型及事件类型）才会通知你的 add-on

示例描述：Add-on 仅接收采购收货窗体与销售订单窗体感兴趣的事件。Add-on 仅处理采购窗体中 et\_CLICK 事件，销售订单中的 et\_CLICK 及 et\_KEY\_DOWN 事件。对象示意图如下：



代码示例：

```
Private Sub SetFilters()
    '// First, create a new EventFilters object
    Set oFilters = New SAPbouiCOM.EventFilters

    '// Second, add an event type to the container
    '// (this method returns an EventFilter object)
    Set oFilter = oFilters.Add(et_CLICK)

    '// Third, assign the form types on which the event should be
    '// processed
    oFilter.Add 139 'Order Form
    oFilter.Add 142 'Purchase Order Form

    '// add a second event type to the container
    Set oFilter = oFilters.Add(et_KEY_DOWN)

    '// assign the form type on which this event should be processed
    oFilter.Add 139 'Orders Form
    '// Fourth, set the event filters object to the application
    SBO_Application.SetFilter oFilters
End Sub
```

## 五、 ToolBar Control

### 1、描述

SAP Business One 目前没有提供 ToolBar 接口，如果用户自己创建的窗体希望使用 ToolBar,可以利用系统现有的 ToolBar Button，例如：Next Record、Previous Record、First Data Record、Last Data Record。当系统 ToolBar Button 被点击时会触发其对应的 Menu Item 事件。用户捕获 Menu Item 的事件即可响应 ToolBar 事件。通过调用 Form 的 EnableMenu 方法可以激活或禁止某个 ToolBar Button。

### 2、用法及示例

示例说明：创建自己的窗体，查询业务伙伴并使用 ToolBar 上的导航按钮浏览信息。

#### 1. 激活希望使用的 ToolBar Button:

```
oForm.EnableMenu "1288", True 'Next Record
```

```
oForm.EnableMenu "1289", True 'Previous Record
oForm.EnableMenu "1290", True 'First Data Record
oForm.EnableMenu "1291", True 'Last Data Record
```

## 2. 查询数据库，获取希望得到的信息

```
Set oRecordSet = oCompany.GetBusinessObject(BoRecordset)
oRecordSet.DoQuery ("Select CardCode,CardName From OCRD")
```

## 3. 实现所选择的 Menu Item 事件

- a) 在每个 Menu 事件处理中相应地操作 RecordSet
- b) 更新创建的窗体上控件信息

```
Private Sub SBO_Application_MenuEvent(pVal As SAPbouiCOM.IMenuEvent, BubbleEvent As Boolean)
```

```
'//*****
'// use the menu item's IDs in order to respond to a specific menu item click
'// the menu items list with unique IDs should be included in the SDK help
'// another way to find the MenuUID is writing a simple Add-On which sends
'// a message box with the MenuID for Menu Events:
'// SBO_Application.MessageBox "The Menu Item's UID is: " & pVal.MenuUID
'//*****
```

```
'//*****
'// Respond only to the 'Next Record', 'Previous Record', 'First Data Record'
'// and 'Last Data Record' Menu Items
'//*****
```

```
If pVal.MenuUID = "1288" Or _
    pVal.MenuUID = "1289" Or _
    pVal.MenuUID = "1290" Or _
    pVal.MenuUID = "1291" Then

    '// Every menu item event is sent twice
    '// once before SAP SBO application handles it
    '// and once after SAP SBO application handles it
    '// to prevent double handling of the event
    '// use 1 of 2 options:
    '// 1. use the BeforeAction property (put your actions in an If clause)
    '// 2. set BubbleEvent to false (the event would be sent only once)
```

```
If pVal.BeforeAction = False Then
    Select Case pVal.MenuUID
        '// 'Next Record' has been clicked
        Case "1288"
            oRecordSet.MoveNext
            '// If reached EOF show last
            If oRecordSet.EOF = True Then
                oRecordSet.MoveLast
            End If
```

```

'// 'Previous Record' has been clicked
Case "1289"
    '// If reached BOF show first
    If oRecordSet.BoF = False Then
        oRecordSet.MovePrevious
    End If

'// 'First Data Record' has been clicked
Case "1290"
    oRecordSet.MoveFirst

'// 'Last Data Record' has been clicked
Case "1291"
    oRecordSet.MoveLast
End Select

'// set the new values to the form's fields
oCodeUserDataSource.Value = oRecordSet.Fields(0).Value
oNameUserDataSource.Value = oRecordSet.Fields(1).Value
oForm.Update
End If
End If

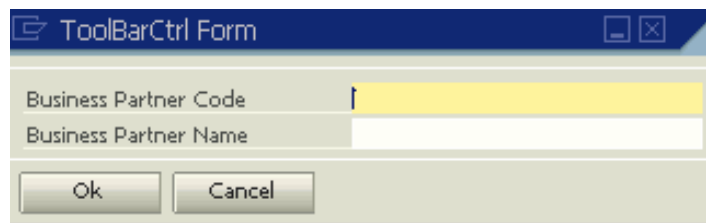
End Sub

```

### 3、练习

创建一个简单的 add-on，要求

1. 添加一个 Menu Item
2. 捕获菜单 click 事件
3. 创建如下所示窗体



4. 实现如下 Toolbar Item
  - a. Next Record
  - b. Previous Record
  - c. First Data Record

- d. Last Data Record
- 5. 捕获 Toolbar 事件导航浏览数据

## 六、 LinkButton

### 1、描述

在 SAP Business One，箭头连接大量用于相关对象间快速导航。例如发票窗体上客户字段前面的箭头能够打开业务伙伴主数据窗体并定位到相关客户记录位置。该箭头就是 LinkButton 对象。LinkButton 对象通过 LinkedObject 属性指定相关联的 Business Object 类型。

### 2、用法及示例

```
'//*****
'// Adding a Text Edit item
'//*****

Set oItem = oForm.Items.Add("EditText1", it_EDIT)
oItem.Left = 157
oItem.Width = 160
oItem.Top = 12
oItem.Height = 14

Set oEditText = oItem.Specific

'// bind the text edit item to the predefined BPCode user data source
oEditText.DataBind.SetBound True, "", "BPCode"
'//*****
'// Adding a Linked Button item
'//*****

Set oItem = oForm.Items.Add("LinkedBtn1", it_LINKED_BUTTON)
oItem.Left = 140
oItem.Width = 15
oItem.Top = 12
oItem.Height = 14

oItem.LinkTo = "EditText1"

Set oLinkedBtn = oItem.Specific

oLinkedBtn.LinkedObject = lf_BusinessPartner
```

### 3、注意

- ✧ 只有连接对象有效时 `LinkButton` 才会显现
- ✧ `Item` 不能 `LinkTo` 自身。连接的 `item` 的 `value` 应该是目标对象的 `key value`。例如 `CardCode` 属性

## 第五章 Add on 的注册和安装

### 第一节 总体介绍

#### 一、概览

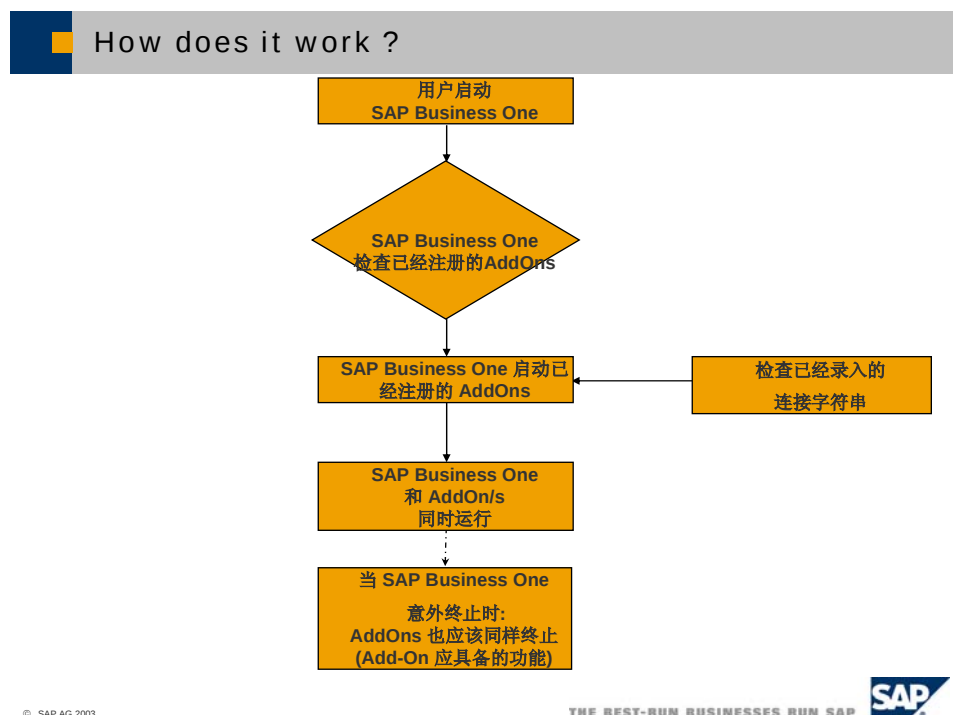
当合作伙伴完成针对 SAP Business One 的特定行业解决方案的开发后，您需要将该解决方案以单独的 CD 分发给客户，在分发之前，必须完成软件包的注册，使得该软件包在启动 SAP Business One 后自动运行，注册文件只需产生一次，注册后的 Add on 通过 UI API 跟 SAP Business One 建立连接，而且在将来分发给客户的软件包中必须包含此文件，在软件包的安装过程中，该文件包含的注册信息将被写进 Windows 的注册表中。另外，软件包的安装打包可以将解决方案的安装和注册一并完成，方便了客户的使用和维护。

### 第二节 Add on 注册

#### 一、Add on 的启动原理

当用户启动 SAP Business One 时，SAP Business One 会检查有哪些 Add on 已经注册过，并同时检查用于连接的 Add on 连接字符串的有效性，当这些验证通过之后，SAP Business One 和 Add on 将同时启动，考虑到程序的完整性，当 SAP Business One 非正常终止时，运行在 SAP Business One 上的 Add on 也应该终止执行，这一点合作伙伴在 Add on 开发时，应该注意到并使程序具备该功能。

注意：客户端要能正常运行这些 Add on，前提是客户端已经安装了这些 Add on 运行所需的 SDK 环境。



## 二、Add on 的注册

### 2.1、注册类型

#### 1)、开发模式 (Development mode)

- 程序代码中写入了 SAP 提供的连接字符串，例如：

0030002C0030002C00530041005000420044005F00440061007400650076002C0050004C006F00  
6D0056004900490056

#### 2)、客户应用模式 (Runtime mode)

- 在 SAP Business One 的客户端进行注册

#### 2：注册工具

**AddOnRegDataGen.exe** 是 SAP 提供的一个用于产生注册数据的文件，该工具的安装是 UI API 安装的一部分，通常被安装至路径 “C:\Program Files\SAP Manage\SAP Business One SDK\UI API\Tools” 下。

### 2.2、注册步骤

#### 1 产生注册文件

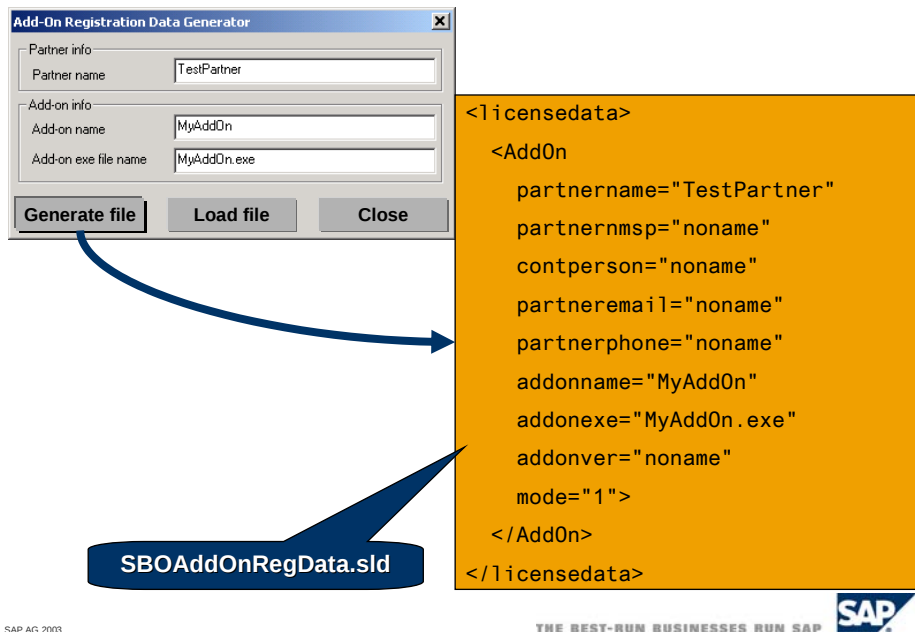
##### 1)、运行注册工具： AddOnRegDataGen.exe

##### 2)、输入合作伙伴的名称、Add on 和生成的对应的可执行文件的名称

在 SAP Business One 安装路径下的 Addons 路径下会看到以输入的合作伙伴名称相同的文件夹，标明了该 Add on 所属的开发商。

##### 3)、点击 “generate file” 按钮，则在注册工具所在的路径下会产生一个名为 “SBOAddOnRegData.sld” 的文件

## 产生注册文件



## 2 校验安装路径

SAP 推荐在客户端安装 Add on 的过程中完成注册，所以首先必须调用 SBOAddonReg.dll，在这里，主要调用了 SBOAddonReg.dll 中内含的三个功能，详情如下图：

### AddOn的注册工具 – SBOAddonReg.dll

我们主要从SBOAddonReg.dll 中调用以下三种功能

- **GetInstallPath** (ByVal installDataFile\$, \_  
ByVal outStr\$, \_  
lLen As Long) As Long

获取该Add on相应注册文件的完整路径并为SAP Business One提供相应的Add On的可执行文件的路径，以便SAP Business One的查找调用

- **RegisterAddon** ()

将Add on注册到 SAP Business One应用程序

- **UnRegisterAddon** (ByVal installDataFile As String) As Boolean

反注册一个 Add-On

新添加的功能:

- **RegisterAddonEx** (const char \*installDataFile )

用于解决以前有的 bug，您可以使用它来替代 RegisterAddon

## 第三节 安装程序

### 一、原理

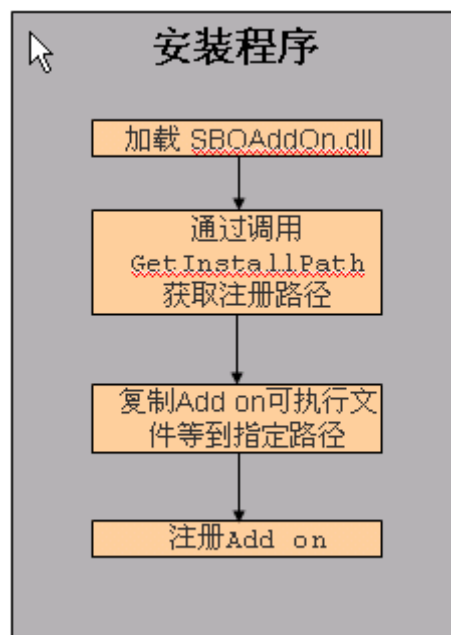
首先，要获取该 Add-on 相应的注册文件，及可执行文件的路径。方法及原理见第二节  
然后，将注册文件，Add-on 可执行文件及其它所需文件进行打包。

### 二、安装程序

#### 1、安装程序需要完成：

- 1.1 复制 Add on 可执行文件到正确路径
- 1.2 成功注册 Add on
- 1.3 卸载时，反注册 Add on

#### 2、安装时的流程入下图：



## 第六章 Case Study

本章节以一个具体需求为例，综合以上章节内容，指导开发人员按部就班使用 SAP Business One SDK 实现整个需求，包括需求分析、设计、业务逻辑、建立表和字段、界面设计、代码开发、注册及打包等方面内容。

客户的具体需求为：计算每个员工的薪水总额，并计算应缴纳的个人所得税。

### 第一节 需求分析

为客户开发 ADDON 的第一个步骤是进行需求分析并准备详细的需求分析描述文档。

#### 一、现有功能分析

1. 员工主数据 - 可用于读取员工信息
2. 日记帐分录 - 日记帐分录功能及 DI 接口

#### 二、差异分析

无录入、计算薪水和个人所得税功能

#### 三、具体需求分析：

1. 可对每个员工录入薪水并计算总额及个人所得税
2. 薪水类型分为工资、奖金、补贴
3. 发放薪水时，可同时发放一种或多种类型的薪水
4. 基于薪水总计自动计算个人所得税
5. 发放薪水同时生成日记帐分录，如下

借：营业费用 (55010101)

贷：应付工资 (21510101)

### 第二节 设计

设计阶段包括两部分的内容：数据结构设计和用户操作界面设计

#### 一、数据结构

根据前面所做的需求分析设计所需的表和字段，及相应的数据关系。  
根据第一节内容，设计表及表结构如下：

员工薪水及个人所得税 – 头表

|   | 字段名      | 描述    | 类型               | 结构      |
|---|----------|-------|------------------|---------|
| 1 | DocNum   | 单据号   | Alphanumeric     | Text    |
| 2 | DocDate  | 创建日期  | Date/Time        | Date    |
| 3 | EmpID    | 员工编码  | Numeric          | Numeric |
| 4 | DocTotal | 总计    | Units and Totals | Sum     |
| 5 | Tax      | 个人所得税 | Units and Totals | Sum     |

薪水明细表

|   | 字段名       | 描述  | 类型               | 结构      |
|---|-----------|-----|------------------|---------|
| 1 | DocNum    | 单据号 | Alphanumeric     | Text    |
| 2 | LineNum   | 行号  | Numeric          | Numeric |
| 3 | Type      | 类型  | Alphanumeric     | Text    |
| 4 | Amount    | 金额  | Units and Totals | Sum     |
| 5 | Reference | 备注  | Alphanumeric     | Text    |

## 二、用户界面

用户操作界面设计包括菜单项、表单、报表等。现设计的薪水及所得税操作界面如下：

薪水及所得税

员工编码

创建日期

员工姓名

| # | 类型 | 金额 | 备注 |  |  |  |
|---|----|----|----|--|--|--|
|   | 工资 |    |    |  |  |  |
|   | 奖金 |    |    |  |  |  |
|   | 补贴 |    |    |  |  |  |
|   |    |    |    |  |  |  |
|   |    |    |    |  |  |  |
|   |    |    |    |  |  |  |
|   |    |    |    |  |  |  |

总计

个人所得税

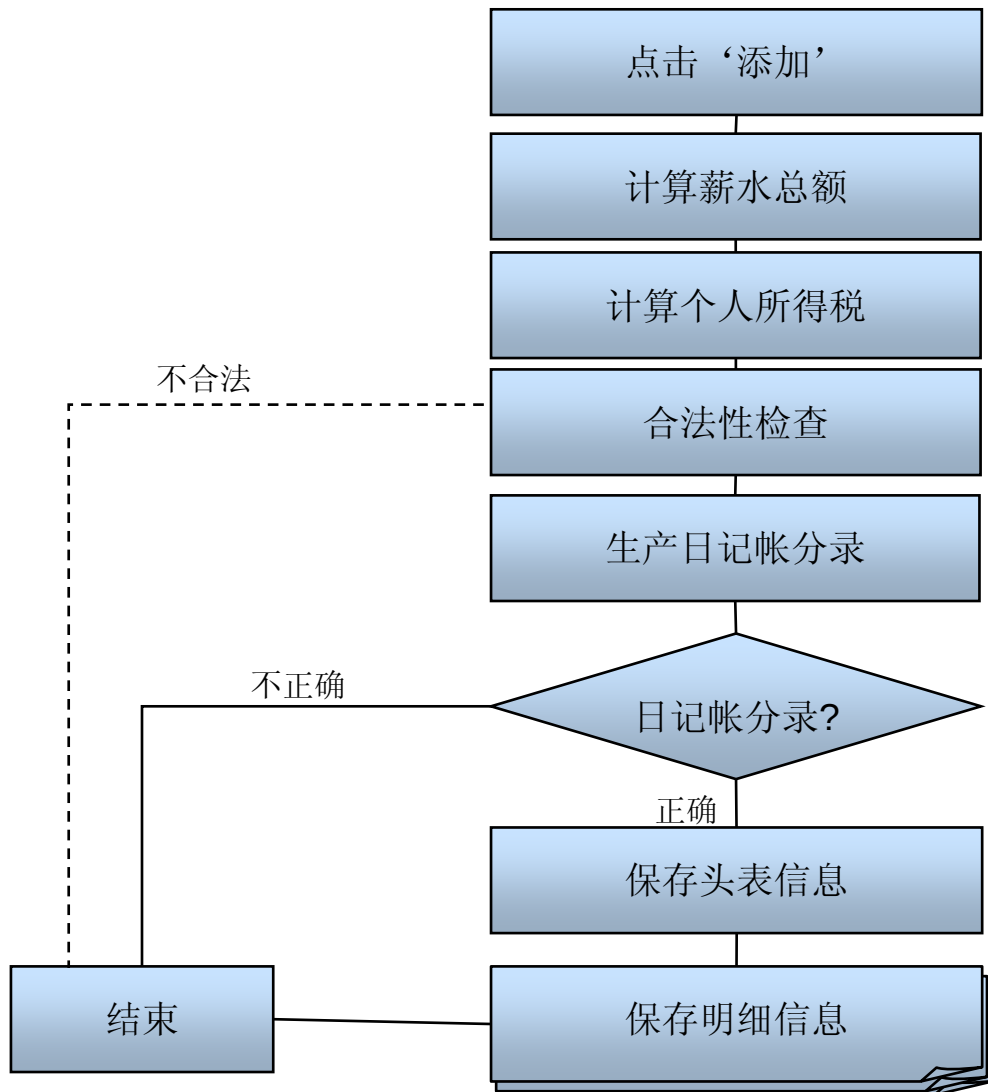
添加

取消

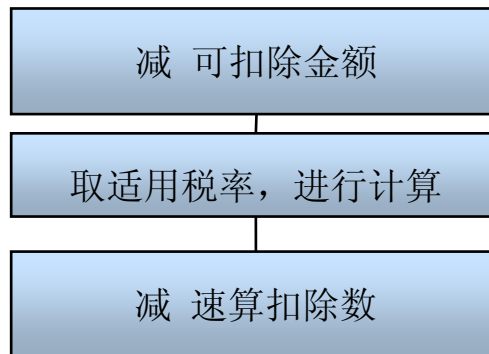
### 第三节 业务逻辑

#### 一、主业务流程

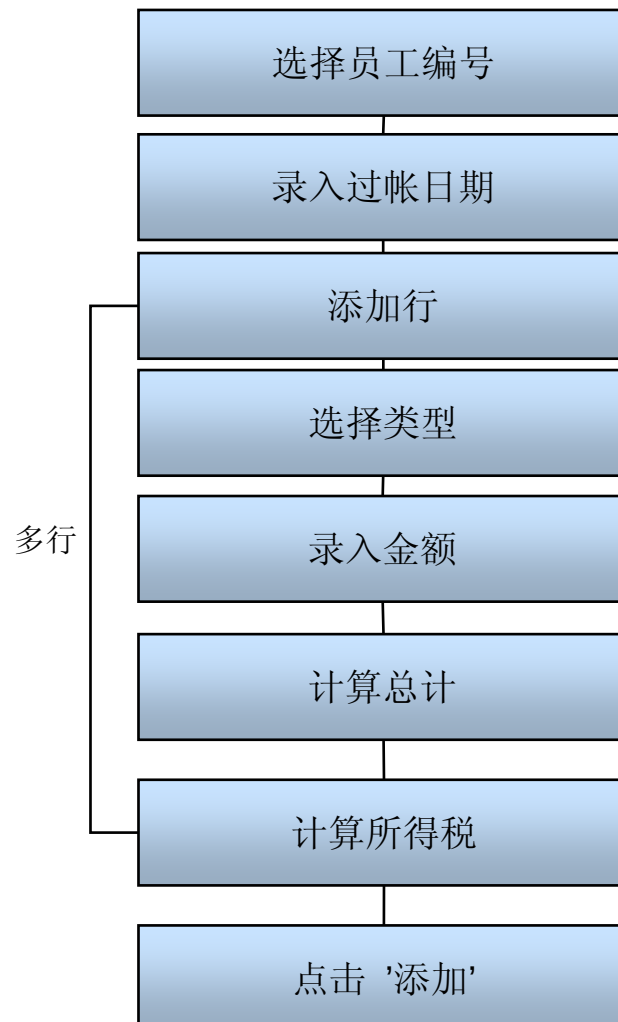
计算薪水及个人所得税



## 二、个人所得税计算流程



## 三、操作流程



## 第四节 建立表和字段

通过 SAP Business One 提供的管理用户字段工具(工具→管理用户字段)来创建在第二节里设计的表。  
(管理用户字段功能详细信息请参见第二章内容)

### 一、自定义表

创建自定义表  
ESS - 员工薪水及个人所得税 – 头表  
EST - 薪水明细表  
注：SQL Server 中的表名为@ESS 和@EST

### 二、自定义字段



注：通过管理用户字段添加的字段，在 SQL Server 中的字段名均加上前缀 U\_。

## 第五节 界面设计

### 一、界面

根据第二节的设计界面，用 SAP Business One SDK 实现界面代码如下  
Private Function CreateSalaryForm()

```

Dim oltem          As SAPbouiCOM.Item

Dim oButton        As SAPbouiCOM.Button
Dim oStaticText As SAPbouiCOM.StaticText
Dim oEditText      As SAPbouiCOM.EditText
Dim oComboBox      As SAPbouiCOM.ComboBox
Dim oCol           As SAPbouiCOM.Column
Dim oMatrix        As SAPbouiCOM.Matrix

'// 添加新表单
Set oForm = SBO_Application.Forms.Add("FormSalary", ft_Fixed)

'// 设表单属性
oForm.Title = "薪水及个人所得税"
oForm.Left = 400
oForm.Width = 400
oForm.Top = 100
oForm.Height = 300

'//*****
'// 添加控件并设置属性
'//*****

'//*****
'// 添加确定按钮
'//*****

Set oltem = oForm.Items.Add("BtnOK", it_BUTTON)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 250
oltem.Height = 19

Set oButton = oltem.Specific

oButton.Caption = "添加"

```

```
//*****  
// 添加取消按钮  
//*****  
  
Set oltem = oForm.Items.Add("2", it_BUTTON)  
oltem.Left = 75  
oltem.Width = 65  
oltem.Top = 250  
oltem.Height = 19  
  
Set oButton = oltem.Specific  
  
oButton.Caption = "取消"  
  
//*****  
// 添加员工 ID  
//*****  
  
Set oltem = oForm.Items.Add("CboEmpID", it_COMBO_BOX)  
oltem.Left = 75  
oltem.Width = 65  
oltem.Top = 5  
oltem.Height = 14  
Set oComboBox = oltem.Specific  
  
Set oltem = oForm.Items.Add("StxtEmpID", it_STATIC)  
oltem.Left = 5  
oltem.Width = 65  
oltem.Top = 5  
oltem.Height = 14  
  
oltem.LinkTo = "CboEmpID"  
  
Set oStaticText = oltem.Specific  
  
oStaticText.Caption = "员工编号"  
  
//*****  
// 添加员工姓名  
//*****  
  
Set oltem = oForm.Items.Add("TxtEmpNam", it_EDIT)  
oltem.Left = 75  
oltem.Width = 65
```

```
oltem.Top = 20
oltem.Height = 14
oltem.Enabled = False

Set oltem = oForm.Items.Add("StxtEmpNam", it_STATIC)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 20
oltem.Height = 14

oltem.LinkTo = "TxtEmpNam"

Set oStaticText = oltem.Specific

oStaticText.Caption = "员工姓名"

'//*****
'// 添加过帐日期
'//*****

Set oltem = oForm.Items.Add("TxtDocDate", it_EDIT)
oltem.Left = 315
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14
Set oEditText = oltem.Specific

Set oltem = oForm.Items.Add("StxtDocDat", it_STATIC)
oltem.Left = 245
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14

oltem.LinkTo = "TxtDocDate"

Set oStaticText = oltem.Specific

oStaticText.Caption = "过帐日期"

'//*****
'// 添加 Matrix
'//*****

Set oltem = oForm.Items.Add("MxLine", it_MATRIX)
oltem.Left = 5
```

```
oltem.Width = 395
oltem.Top = 40
oltem.Height = 180
Set oMatrix = oltem.Specific

Set oCol = oMatrix.Columns.Add("Col0", it_EDIT)
oCol.TitleObject.Caption = ""
oCol.Width = 0
oCol.Visible = False

Set oCol = oMatrix.Columns.Add("Col1", it_COMBO_BOX)
oCol.TitleObject.Caption = "类型"
oCol.Width = 80
oCol.DisplayDesc = True

Set oCol = oMatrix.Columns.Add("Col2", it_EDIT)
oCol.TitleObject.Caption = "金额"
oCol.Width = 120
oCol.RightJustified = True

Set oCol = oMatrix.Columns.Add("Col3", it_EDIT)
oCol.TitleObject.Caption = "备注"
oCol.Width = 175

'//*****
'// 添加总计
'//*****

Set oltem = oForm.Items.Add("TxtTotal", it_EDIT)
oltem.Left = 300
oltem.Width = 80
oltem.Top = 215
oltem.Height = 14
oltem.Enabled = False
oltem.RightJustified = True
Set oEditText = oltem.Specific

Set oltem = oForm.Items.Add("StxtTotal", it_STATIC)
oltem.Left = 230
oltem.Width = 65
oltem.Top = 215
oltem.Height = 14

oltem.LinkTo = "TxtTotal"
```

```
Set oStaticText = oltem.Specific
```

```
oStaticText.Caption = "总计"
```

```
'//*****
```

```
'// 添加所得税
```

```
'//*****
```

```
Set oltem = oForm.Items.Add("TxtTax", it_EDIT)
```

```
oltem.Left = 300
```

```
oltem.Width = 80
```

```
oltem.Top = 230
```

```
oltem.Height = 14
```

```
oltem.Enabled = False
```

```
oltem.RightJustified = True
```

```
Set oEditText = oltem.Specific
```

```
Set oltem = oForm.Items.Add("StxtTax", it_STATIC)
```

```
oltem.Left = 230
```

```
oltem.Width = 65
```

```
oltem.Top = 230
```

```
oltem.Height = 14
```

```
oltem.LinkTo = "TxtTax"
```

```
Set oStaticText = oltem.Specific
```

```
oStaticText.Caption = "个人所得税"
```

```
End Function
```

| 类型 | 金额 | 备注 |
|----|----|----|
|    |    |    |
|    |    |    |
|    |    |    |
|    |    |    |
|    |    |    |
|    |    |    |
|    |    |    |
|    |    |    |

## 第六节 代码

### 一、建立与 SAP Business One 的连接，取得 Application, Company 对象

#### 1. 新建 SalaryForm.cls

首先要以 WithEvents 的方式定义 Private 变量 SBO\_Application, 这样才能使用 SBO\_Application 提供的三个事件出发接口 AppEvent、ItemEvent 和 MenuEvent。代码如下：

```
Private WithEvents SBO_Application As SAPbouiCOM.Application
```

建立连接并取得 SBO\_Application 的代码如下

```
Private Sub Class_Initialize()
```

```
    SetApplication
```

```
End Sub
```

```
Private Sub SetApplication()
```

```
    Dim SboGuiApi          As SAPbouiCOM.SboGuiApi
```

```
    Dim sConnectionString  As String
```

```
    Set SboGuiApi = New SAPbouiCOM.SboGuiApi
```

```
    sConnectionString = Command
```

```
    '// 连接 SBO 应用程序
```

```
    SboGuiApi.Connect sConnectionString
```

```
    '// 取得 SBO_Application 对象
```

```
    Set SBO_Application = SboGuiApi.GetApplication()
```

```
End Sub
```

连接 DI API 及公司数据库

```
Private Sub Class_Initialize()
```

```
    SetApplication
```

```
    '//*****
```

```
    '// 连接 DI API
```

```
    '//*****
```

```

If Not SetConnectionContext = 0 Then
    SBO_Application.MessageBox "Failed setting a connection to DI API"
End
End If

'//*****
'// 连接公司数据库
'//*****

If Not SBO_Company.Connect = 0 Then
    SBO_Application.MessageBox "Failed connecting to the company's Data Base"
End
End If

```

End Sub

Private Function SetConnectionContext() As Long

```

Dim sCookie As String
Dim sConnectionContext As String
Dim lRetCode As Long

Set SBO_Company = New SAPbobsCOM.Company

sCookie = SBO_Company.GetContextCookie

sConnectionContext = SBO_Application.Company.GetConnectionContext(sCookie)

SetConnectionContext = SBO_Company.SetSboLoginContext(sConnectionContext)

```

End Function

Private Function ConnectToCompany() As Long

```

'// 建立公司数据库连接
ConnectToCompany = SBO_Company.Connect

```

End Function

2. 新建 SubMain.bas

'Option Explicit

Public Type POINTAPI

    X As Long

    Y As Long

End Type

'// The message structure

```

Public Type Msg
    hwnd As Long
    message As Long
    wParam As Long
    lParam As Long
    Time As Long
    pt As POINTAPI
End Type

'// Retrieves messages sent to the calling thread's message queue
Public Declare Function GetMessage Lib "user32" _
    Alias "GetMessageA" _
    (lpMsg As Msg, _
    ByVal hwnd As Long, _
    ByVal wParam As Long, _
    ByVal lParam As Long) As Long

'// Translates virtual-key messages into character messages
Public Declare Function TranslateMessage Lib "user32" _
    (lpMsg As Msg) As Long

'// Forwards the message on to the window represented by the
'// hWnd member of the Msg structure
Public Declare Function DispatchMessage Lib "user32" _
    Alias "DispatchMessageA" _
    (lpMsg As Msg) As Long

Public Msg As Msg

Public Sub Main()

    Dim SalaryForm As SalaryForm
    Set SalaryForm = New SalaryForm

    Do While GetMessage(Msg, 0&, 0&, 0&)
        TranslateMessage Msg
        DispatchMessage Msg
        DoEvents
    Loop

End Sub

```

注：若无特别说明，以下代码均指 SalaryForm.cls

## 二、添加计算薪水及个人所得税功能到菜单

Private Sub Class\_Initialize()

```
.....
'//*****
'// 添加菜单项
'//*****
AddSalaryFormToMenu
```

End Sub

Private Sub AddSalaryFormToMenu()

```
Dim oMenus      As SAPbouiCOM.Menus
Dim oMenuItem   As SAPbouiCOM.MenuItem
```

```
Set oMenus = SBO_Application.Menus
Set oMenuItem = oMenus.Item(4) '模块
```

```
Set oMenuItem.SubMenus.Item(10).SubMenus '人力资源
Set oMenuItem = oMenus.Add("MenuSalary", "计算薪水及所得税", mt_STRING, oMenus.Count)
```

End Sub



### 三、通过菜单调起计算薪水及个人所得税表单

在 SBO\_Application\_MenuEvent 里编写代码，调起薪水及所得税表单。

```
Private Sub SBO_Application_MenuEvent(pVal As SAPbouiCOM.IMenuEvent, BubbleEvent As Boolean)
```

```
    If pVal.MenuUID = "MenuSalary" And pVal.BeforeAction = True Then
```

```
        CreateSalaryForm
```

```
    End If
```

```
End Sub
```

### 四、表单初始化

1. 添加数据源，并与控件绑定
2. 控件初始化，如下拉菜单
3. 设置菜单项状态等。

```
Private oUSDDocDate As SAPbouiCOM.UserDataSource
Private oUSDEmpID As SAPbouiCOM.UserDataSource
Private oUSDDocTotal As SAPbouiCOM.UserDataSource
Private oUSDTax As SAPbouiCOM.UserDataSource
Private oUSDType As SAPbouiCOM.UserDataSource
Private oUSDAmount As SAPbouiCOM.UserDataSource
Private oUSDReference As SAPbouiCOM.UserDataSource
Private Function CreateSalaryForm()
```

```
    Dim oltem As SAPbouiCOM.Item
```

```
    Dim oButton As SAPbouiCOM.Button
```

```
    Dim oStaticText As SAPbouiCOM.StaticText
```

```

Dim oEditText As SAPbouiCOM.EditText
Dim oComboBox As SAPbouiCOM.ComboBox
Dim oCol As SAPbouiCOM.Column
Dim oMatrix As SAPbouiCOM.Matrix
Dim oRst As SAPbobsCOM.Recordset
Dim i As Long

'// 添加新表单
Set oForm = SBO_Application.Forms.Add("FormSalary", ft_Fixed)

oForm.DataSources.UserDataSources.Add "DocDate", dt_DATE, 10
oForm.DataSources.UserDataSources.Add "EmpID", dt_SHORT_NUMBER, 4
oForm.DataSources.UserDataSources.Add "DocTotal", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Tax", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Type", dt_SHORT_TEXT, 1
oForm.DataSources.UserDataSources.Add "Amount", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Ref", dt_SHORT_TEXT, 100
'// 设表单属性
oForm.Title = "薪水和所得税"
oForm.Left = 400
oForm.Width = 400
oForm.Top = 100
oForm.Height = 300

'//*****
'// 添加控件并设置属性
'//*****

'//*****
'// 添加确定按钮
'//*****

Set oltem = oForm.Items.Add("BtnOK", it_BUTTON)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 250
oltem.Height = 19

Set oButton = oltem.Specific

oButton.Caption = "添加"

'//*****
'// 添加取消按钮
'//*****

```

```

Set oltem = oForm.Items.Add("2", it_BUTTON)
oltem.Left = 75
oltem.Width = 65
oltem.Top = 250
oltem.Height = 19

```

```

Set oButton = oltem.Specific

```

```

oButton.Caption = "取消"

```

```

'//*****
'// 添加员工 ID
'//*****

```

```

Set oltem = oForm.Items.Add("CboEmpID", it_COMBO_BOX)
oltem.Left = 75
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14
Set oComboBox = oltem.Specific

```

```

'// 绑定数据源

```

```

oComboBox.DataBind.SetBound True, "", "EmpID"

```

```

'// 初始化下拉菜单

```

```

Set oRst = SBO_Company.GetBusinessObject(BoRecordset)
oRst.DoQuery ("SELECT EmpID, RTRIM(LastName) + ' ' + RTRIM(MiddleName) + ' ' +
RTRIM(FirstName) AS Name " + _
" FROM OHEM")
oRst.MoveFirst
For i = 1 To oRst.RecordCount
    oComboBox.ValidValues.Add oRst.Fields("EmpID").Value, oRst.Fields("Name").Value
    oRst.MoveNext
Next

```

```

Set oltem = oForm.Items.Add("StxtEmpID", it_STATIC)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14

```

```

oltem.LinkTo = "CboEmpID"

```

```

Set oStaticText = oltem.Specific

```

```
oStaticText.Caption = "员工编号"
```

```
'//*****
```

```
// 添加员工姓名
```

```
'//*****
```

```
Set oltem = oForm.Items.Add("TxtEmpNam", it_EDIT)
```

```
oltem.Left = 75
```

```
oltem.Width = 65
```

```
oltem.Top = 20
```

```
oltem.Height = 14
```

```
oltem.Enabled = False
```

```
Set oltem = oForm.Items.Add("StxtEmpNam", it_STATIC)
```

```
oltem.Left = 5
```

```
oltem.Width = 65
```

```
oltem.Top = 20
```

```
oltem.Height = 14
```

```
oltem.LinkTo = "TxtEmpNam"
```

```
Set oStaticText = oltem.Specific
```

```
oStaticText.Caption = "员工姓名"
```

```
'//*****
```

```
// 添加过帐日期
```

```
'//*****
```

```
Set oltem = oForm.Items.Add("TxtDocDate", it_EDIT)
```

```
oltem.Left = 315
```

```
oltem.Width = 65
```

```
oltem.Top = 5
```

```
oltem.Height = 14
```

```
Set oEditText = oltem.Specific
```

```
oEditText.DataBind.SetBound True, "", "DocDate"
```

```
Set oltem = oForm.Items.Add("StxtDocDat", it_STATIC)
```

```
oltem.Left = 245
```

```
oltem.Width = 65
```

```
oltem.Top = 5
```

```
oltem.Height = 14
```

```
oltem.LinkTo = "TxtDocDate"
```

```
Set oStaticText = oltem.Specific

oStaticText.Caption = "过帐日期"

'//*****
'// 添加 Matrix
'//*****

Set oltem = oForm.Items.Add("MxLine", it_MATRIX)
oltem.Left = 5
oltem.Width = 395
oltem.Top = 40
oltem.Height = 180
Set oMatrix = oltem.Specific

Set oCol = oMatrix.Columns.Add("Col0", it_EDIT)
oCol.TitleObject.Caption = ""
oCol.Width = 0
oCol.Visible = False

Set oCol = oMatrix.Columns.Add("Col1", it_COMBO_BOX)
oCol.TitleObject.Caption = "类型"
oCol.Width = 80
oCol.DisplayDesc = True
oCol.DataBind.SetBound True, "", "Type"

Set oCol = oMatrix.Columns.Add("Col2", it_EDIT)
oCol.TitleObject.Caption = "金额"
oCol.Width = 120
oCol.RightJustified = True
oCol.DataBind.SetBound True, "", "Amount"

Set oCol = oMatrix.Columns.Add("Col3", it_EDIT)
oCol.TitleObject.Caption = "备注"
oCol.Width = 175
oCol.DataBind.SetBound True, "", "Ref"

'//*****
'// 添加总计
'//*****

Set oltem = oForm.Items.Add("TxtTotal", it_EDIT)
oltem.Left = 300
oltem.Width = 80
oltem.Top = 215
```

```
oltem.Height = 14
oltem.Enabled = False
oltem.RightJustified = True
Set oEditText = oltem.Specific
oEditText.DataBind.SetBound True, "", "DocTotal"

Set oltem = oForm.Items.Add("StxtTotal", it_STATIC)
oltem.Left = 230
oltem.Width = 65
oltem.Top = 215
oltem.Height = 14

oltem.LinkTo = "TxtTotal"

Set oStaticText = oltem.Specific

oStaticText.Caption = "总计"

'//*****
'// 添加所得税
'//*****

Set oltem = oForm.Items.Add("TxtTax", it_EDIT)
oltem.Left = 300
oltem.Width = 80
oltem.Top = 230
oltem.Height = 14
oltem.Enabled = False
oltem.RightJustified = True
Set oEditText = oltem.Specific
oEditText.DataBind.SetBound True, "", "Tax"

Set oltem = oForm.Items.Add("StxtTax", it_STATIC)
oltem.Left = 230
oltem.Width = 65
oltem.Top = 230
oltem.Height = 14

oltem.LinkTo = "TxtTax"

Set oStaticText = oltem.Specific

oStaticText.Caption = "个人所得税"

'//设置菜单项状态
```

```
oForm.EnableMenu 1281, False '查找
oForm.EnableMenu 1282, True  '添加
```

```
oForm.EnableMenu 1292, True  '添加行
oForm.EnableMenu 1293, True  '删除行
oForm.EnableMenu 1294, True  '复制行
```

```
oForm.Visible = True
```

End Function



## 五、界面操作

任何对表单的操作都会触发 SBO\_Application\_ItemEvent 事件，通过编写相应代码处理

1. 选择员工编号后，显示其姓名，并自动 Matrix 里添加一行
2. Matrix 操作的处理
3. 点击添加

```
Private Sub SBO_Application_ItemEvent(ByVal FormUID As String, pVal As SAPbouiCOM.ItemEvent,
BubbleEvent As Boolean)
```

```
Dim oMatrix As SAPbouiCOM.Matrix
Dim nTotal As Double
```

```
If pVal.FormType = 60006 Then
```

```
Set oForm = SBO_Application.Forms(FormUID)
```

```
If pVal.Before_Action = False Then
```

```
    '\ 员工编号下拉菜单，选择时
```

```
    If pVal.ItemUID = "CboEmpID" And pVal.EventType = et_COMBO_SELECT Then
```

```
        CboEmpID_Select
```

```

        "\ 设焦点到过帐日期
        oForm.Items("TxtDocDate").Click
        "\ 重新设置控件的 Enabled 属性
        SetEnableToFalse
    End If
    "\ Matrix 控件事件
    If pVal.ItemUID = "MxLine" Then
        "\ 类型下拉菜单
        If pVal.EventType = et_COMBO_SELECT Then
            Set oMatrix = oForm.Items(pVal.ItemUID).Specific
            If oMatrix.RowCount = pVal.Row Then
                "\ 添加新行
                oMatrix.AddRow
            End If
        End If
    End If
    "\ 修改金额时，重新计算总计及个人所得税
    If pVal.ColUID = "Col2" And pVal.EventType = et_KEY_DOWN And pVal.CharPressed =
vbKeyTab Then
        nTotal = CalculateTotal
        oForm.Items("TxtTotal").Enabled = True
        oForm.Items("TxtTotal").Specific.Value = nTotal
        oForm.Items("TxtTotal").Enabled = False
        oForm.Items("TxtTax").Enabled = True
        oForm.Items("TxtTax").Specific.Value = CalculateTax(nTotal)
        oForm.Items("TxtTax").Enabled = False
        oForm.Update
    End If
End If
Else
    If pVal.ItemUID = "BtnOK" Then
        "\ 保存数据，并关闭表单
        If SaveSalary = True Then
            oForm.Close
        End If
    End If
End If
End If
End Sub

Private Sub CboEmpID_Select()

    Dim oComboBox As SAPbouiCOM.ComboBox
    Dim oEditText As SAPbouiCOM.EditText
    Dim oMatrix As SAPbouiCOM.Matrix

```

```

Set oComboBox = oForm.Items("CboEmpID").Specific
Set oEditText = oForm.Items("TxtEmpNam").Specific

oEditText.String = oComboBox.Selected.Description

'//*****
'// 类型下拉菜单初始化
'//*****
Set oMatrix = oForm.Items("MxLine").Specific
If oMatrix.RowCount = 0 Then
    oMatrix.AddRow
    Set oComboBox = oMatrix.Columns("Col1").Cells(1).Specific
    oComboBox.ValidValues.Add "S", "工资"
    oComboBox.ValidValues.Add "B", "奖金"
    oComboBox.ValidValues.Add "A", "补贴"
End If

End Sub

Private Sub SetEnableToFalse()

```

```

    If Not oForm Is Nothing Then
        oForm.Items("TxtEmpNam").Enabled = False
        oForm.Items("TxtTotal").Enabled = False
        oForm.Items("TxtTax").Enabled = False
        oForm.Update
    End If

```

```
End Sub
```

## 六、保存数据

1. 当点击添加时，首先要进行合法性检查。
2. 数据合法的情况下，保存数据，并生成日记帐。

```
Private Function ValidToSaveSalary() As Boolean
```

```

    ValidToSaveSalary = False

    '// 检查过帐日期
    If VBA.Trim(oForm.Items("TxtDocDate").Specific.String) = "" Then
        SBO_Application.SetStatusBarMessage "请输入过帐日期。", bmt_Long, True
        oForm.Items("TxtDocDate").Click ct_Regular
        Exit Function
    End If

```

```
ValidToSaveSalary = True
```

```
End Function
```

```
Private Function SaveSalary() As Boolean
```

```
    SaveSalary = False
```

```
    Dim oUserTables As SAPbobsCOM.UserTables
```

```
    Dim oTblESS      As SAPbobsCOM.UserTable
```

```
    Dim oTblEST      As SAPbobsCOM.UserTable
```

```
    Dim oMatrix      As SAPbouiCOM.Matrix
```

```
    Dim i            As Long
```

```
    Dim sErrMsg      As String
```

```
    Dim lErrCode     As Long
```

```
    Dim lRetCode     As Long
```

```
    Dim oRst         As SAPbobsCOM.Recordset
```

```
    Dim sDocNum      As String
```

```
    Dim oJE          As SAPbobsCOM.JournalEntries
```

```
    Dim dDocDate     As Date
```

```
    Dim oSBObob      As SAPbobsCOM.SBObob
```

```
    '// 合法性检查
```

```
    If ValidToSaveSalary = False Then
```

```
        Exit Function
```

```
    End If
```

```
    '// 过帐日期类型转换
```

```
    Set oSBObob = SBO_Company.GetBusinessObject(BoBridge)
```

```
    Set oRst = oSBObob.Format_StringToDate(oForm.DataSources.UserDataSources("DocDate").Value)
```

```
    dDocDate = oRst.Fields("Date").Value
```

```
    Set oSBObob = Nothing
```

```
    '// 开始一个 Transaciton 以保证数据完整行和一致性
```

```
    SBO_Company.StartTransaction
```

```
    '// 生成日记帐分录
```

```
    Set oJE = SBO_Company.GetBusinessObject(oJournalEntries)
```

```
    oJE.Memo = "计提应付工资"
```

```
    oJE.ReferenceDate = dDocDate
```

```
    '// 借方：营业费用
```

```
    oJE.Lines.Add
```

```
    oJE.Lines.AccountCode = "55010101"
```

```

oJE.Lines.Debit = oForm.DataSources.UserDataSources("DocTotal").Value
'// 贷方：应付工资
oJE.Lines.Add
oJE.Lines.AccountCode = "21510101"
oJE.Lines.Credit = oForm.DataSources.UserDataSources("DocTotal").Value

If oJE.Add <> 0 Then
    SBO_Company.GetLastError lErrCode, sErrMsg
    SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
    Exit Function
End If

'// 取得下一个 DocNum
Set oRst = SBO_Company.GetBusinessObject(BoRecordset)
oRst.DoQuery ("SELECT TOP 1 U_DocNum AS DocNum FROM [@ESS] ORDER BY DocNum DESC")
If oRst.RecordCount = 0 Then
    sDocNum = "1"
Else
    sDocNum = VBA.CStr(VBA.CInt(oRst.Fields("DocNum").Value + 1))
End If

'// 保存薪水及个人所得税—头表信息
Set oTblESS = SBO_Company.UserTables("ESS")
oTblESS.Code = AssignUID
oTblESS.Name = "ESS" + AssignUID
oTblESS.UserFields("U_DocNum").Value = sDocNum
oTblESS.UserFields("U_DocDate").Value = dDocDate
oTblESS.UserFields("U_EmplID").Value = oForm.DataSources.UserDataSources("EmplID").Value
oTblESS.UserFields("U_DocTotal").Value = VBA.CDBl(oForm.DataSources.UserDataSources("DocTotal").Value)
oTblESS.UserFields("U_Tax").Value = VBA.CDBl(oForm.DataSources.UserDataSources("Tax").Value)
If oTblESS.Add <> 0 Then
    SBO_Company.GetLastError lErrCode, sErrMsg
    SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
    Exit Function
End If

Set oMatrix = oForm.Items("MxLine").Specific

'// 保存薪水明细
Set oTblEST = SBO_Company.UserTables("EST")
For i = 1 To oMatrix.RowCount
    oMatrix.GetLineData (i)
    If oForm.DataSources.UserDataSources("Type").Value = "" Then
        '// 检查类型

```

```

If oForm.DataSources.UserDataSources("Amount").Value <> 0 Then
    SBO_Company.EndTransaction wf_RollBack
    SBO_Application.SetStatusBarMessage "请选择薪水类型", bmt_Long, True
    oMatrix.Columns("Col1").Cells(i).Click
    Exit Function
End If
Else
    oTblEST.Code = AssignUID
    oTblEST.Name = "ESS" + AssignUID
    oTblEST.UserFields("U_DocNum").Value = sDocNum
    oTblEST.UserFields("U_LineNum").Value = VBA.CStr(i)
    oTblEST.UserFields("U_Type").Value = oForm.DataSources.UserDataSources("Type").Value
    oTblEST.UserFields("U_Amount").Value = VBA.CDbl(oForm.DataSources.UserDataSources("Amount").Value)
    oTblEST.UserFields("U_Ref").Value = oForm.DataSources.UserDataSources("Ref").Value
    If oTblEST.Add <> 0 Then
        SBO_Company.GetLastError IErrCode, sErrMsg
        SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
        Exit Function
    End If
End If
Next

SBO_Company.EndTransaction wf_Commit

SaveSalary = True

End Function

```

薪水及个人所得税

员工编号 2 过帐日期 2004.07.12

员工姓名 李 惠

| 类型 | 金额       | 备注 |
|----|----------|----|
| 工资 | 1,800.00 |    |
| 奖金 | 300.00   |    |
| 补贴 | 200.00   |    |
|    | 0.00     |    |
|    |          |    |
|    |          |    |
|    |          |    |
|    |          |    |
|    |          |    |

总计 2,300.00

个人所得税 85.00

添加 取消

日记帐分录

| 编号  | 起息日        | 过帐日期       | 计税日期       | 参考 1 | 参考 2 | 事务代码 | 明细     |
|-----|------------|------------|------------|------|------|------|--------|
| 459 | 2004.07.12 | 2004.07.12 | 2004.07.12 |      |      |      | 计提应付工资 |

| 源  | 原始编号 | 模板类型 | 模板 | 标识 | 项目 |
|----|------|------|----|----|----|
| JE | 459  |      |    |    |    |

☐ 固定汇率
 ☐ 冲销
 ☐ 调整交易 (期间 13)

| 总帐科目/业务伙伴 | 总帐科目/业务伙伴名称 | 参考 1 | 参考 2 | 参考 3 | 冲销科目     |
|-----------|-------------|------|------|------|----------|
| 55010101  | 工资          |      |      |      | 21510101 |

| 借方 (外币) | 贷方 (外币) | 借方           | 贷方 | 借方 (系统货币)    | 贷方 (系统货币) |
|---------|---------|--------------|----|--------------|-----------|
|         |         | 2,300.00 RMB |    | 2,300.00 EUR |           |

| 起息日        | 过帐日期       | 计税日期       | 项目 | 计税组 | 利润中心 | 明细     |
|------------|------------|------------|----|-----|------|--------|
| 2004.07.12 | 2004.07.12 | 2004.07.12 |    |     |      | 计提应付工资 |

主表项目

| # | 总帐科目/业务名称     | 借方           | 贷方           | 起息日        | 主表项目 |
|---|---------------|--------------|--------------|------------|------|
| 1 | 55010101 工资   | 2,300.00 RMB |              | 2004.07.12 |      |
| 2 | 21510101 应付工资 |              | 2,300.00 RMB | 2004.07.12 |      |
|   |               |              |              |            |      |
|   |               |              |              |            |      |
|   |               | 2,300.00 RMB | 2,300.00 RMB |            |      |

更新 取消 取消模板
 ☐ 显示外币
 ☐ 显示系统货币

## 七、选择公司及退出

1. SAP Business One 在重新选择公司后，需要在 SBO\_Application\_AppEvent 更新 DI API 和公司数据库的连接
2. SAP Business One 退出时，Add on 不会自动退出，因此需要编写代码

Private Sub SBO\_Application\_AppEvent(ByVal EventType As SAPbouiCOM.BoAppEventTypes)

    "\ 重新选择公司

    If EventType = aet\_CompanyChanged Then

        '//\*\*\*\*\*

        '// 连接 DI API

        '//\*\*\*\*\*

        If Not SetConnectionContext = 0 Then

            SBO\_Application.MessageBox "Failed setting a connection to DI API"

        End

        End If

        '//\*\*\*\*\*

        '// 连接公司数据库

```

//*****

If Not SBO_Company.Connect = 0 Then
    SBO_Application.MessageBox "Failed connecting to the company's Data Base"
End
End If

End If

// SAP Business One 退出时，退出 ADD ON
If EventType = aet_ShutDown Then
    End
End If

End Sub

```

## 第七节 注册及安装程序

- 在为客户提供 Add on 时，需要一个单独的安装程序，它独立于 SAP Business One 的安装盘。它提供
    - ◇ 为客户开发的所有模块
    - ◇ 可进行安装、注册和卸载
    - ◇ 介绍文档或操作手册等
  - 需要确认 SDK 的兼容性与一致性
  - 确保 SDK 的更新与升级
- 详细信息请参见第五章内容

## 第八节 练习

假设客户增加需求如下：

1. 用户可选择和录入贷方应付工资科目，并需要录入日记帐分录的备注。
2. 借方科目根据不同的薪水类型，可选择或录入指定科目。
3. 日记帐分录保存成功后，需要通知用户分录编号。
4. 增加报销类型，且此类薪水不缴纳个人所得税。
5. 添加发放薪水功能。

请根据以上需求，并参考第一节到第六节内容，完成相应设计及代码。

## 附录 Case Study 源代码

### 一、SalaryForm.cls

```
// SAP MANAGE UI API 6.5 SDK Sample
//*****
//
// File:      SalaryForm.cls
//
// Copyright (c) SAP MANAGE
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF
// ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO
// THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A
// PARTICULAR PURPOSE.
//
//*****

//*****
// BEFORE STARTING:
// 1. Add reference to the "SAP Business One UI API"
// 2. Add reference to the "SAP Business One Objects Bridge API Version 6.5"
// -----
// 1.
//   a. Project->References
//   b. check the "SAP Business One UI API" check box
// 2.
//   a. Project->References
//   b. check the "SAP Business One Objects Bridge API Version 6.5" check box
//
//*****
```

Option Explicit

```
Private WithEvents SBO_Application As SAPbouiCOM.Application
Private SBO_Company      As SAPbobsCOM.Company
Private oForm            As SAPbouiCOM.Form
```

```
Private Sub Class_Initialize()
```

```
    SetApplication
```

```

'//*****
'// 连接 DI API
'//*****

```

```

If Not SetConnectionContext = 0 Then
    SBO_Application.MessageBox "Failed setting a connection to DI API"
End
End If

```

```

'//*****
'// 连接公司数据库
'//*****

```

```

If Not SBO_Company.Connect = 0 Then
    SBO_Application.MessageBox "Failed connecting to the company's Data Base"
End
End If

```

```

'//*****
'// 添加菜单项
'//*****
AddSalaryFormToMenu

```

End Sub

Private Sub SetApplication()

```

Dim SboGuiApi          As SAPbouiCOM.SboGuiApi
Dim sConnectionString  As String

```

```

Set SboGuiApi = New SAPbouiCOM.SboGuiApi

```

```

sConnectionString = Command

```

```

'// 连接 SBO 应用程序

```

```

SboGuiApi.Connect sConnectionString

```

```

'// 取得 SBO_Application 对象

```

```

Set SBO_Application = SboGuiApi.GetApplication()

```

End Sub

Private Function SetConnectionContext() As Long

```

Dim sCookie As String
Dim sConnectionContext As String
Dim lRetCode As Long

Set SBO_Company = New SAPbobsCOM.Company

sCookie = SBO_Company.GetContextCookie

sConnectionContext = SBO_Application.Company.GetConnectionContext(sCookie)

SetConnectionContext = SBO_Company.SetSboLoginContext(sConnectionContext)

```

End Function

Private Function ConnectToCompany() As Long

```

'// 建立公司数据库连接
ConnectToCompany = SBO_Company.Connect

```

End Function

Private Sub SBO\_Application\_AppEvent(ByVal EventType As SAPbouiCOM.BoAppEventTypes)

```

"\ 重新选择公司
If EventType = aet_CompanyChanged Then

```

```

'//*****
'// 连接 DI API
'//*****

```

```

If Not SetConnectionContext = 0 Then
    SBO_Application.MessageBox "Failed setting a connection to DI API"
End
End If

```

```

'//*****
'// 连接公司数据库
'//*****

```

```

If Not SBO_Company.Connect = 0 Then
    SBO_Application.MessageBox "Failed connecting to the company's Data Base"
End
End If

```

End If

'// SAP Business One 退出时，退出 ADD-ON

If EventType = aet\_ShutDown Then

End

End If

End Sub

Private Sub SBO\_Application\_ItemEvent(ByVal FormUID As String, pVal As SAPbouiCOM.IItemEvent, BubbleEvent As Boolean)

Dim oMatrix As SAPbouiCOM.Matrix

Dim nTotal As Double

If pVal.FormType = 60006 Then

Set oForm = SBO\_Application.Forms(FormUID)

If pVal.Before\_Action = False Then

'\ 员工编号下拉菜单，选择时

If pVal.ItemUID = "CboEmpID" And pVal.EventType = et\_COMBO\_SELECT Then

CboEmpID\_Select

'\ 设焦点到过帐日期

oForm.Items("TxtDocDate").Click

'\ 重新设置控件的 Enabled 属性

SetEnableToFalse

End If

'\ Matrix 控件事件

If pVal.ItemUID = "MxLine" Then

'\ 类型下拉菜单

If pVal.EventType = et\_COMBO\_SELECT Then

Set oMatrix = oForm.Items(pVal.ItemUID).Specific

If oMatrix.RowCount = pVal.Row Then

'\ 添加新行

oMatrix.AddRow

End If

End If

'\ 修改金额时，重新计算总计及个人所得税

If pVal.ColUID = "Col2" And pVal.EventType = et\_KEY\_DOWN And pVal.CharPressed =

vbKeyTab Then

nTotal = CalculateTotal

oForm.Items("TxtTotal").Enabled = True

oForm.Items("TxtTotal").Specific.Value = nTotal

```

        oForm.Items("TxtTotal").Enabled = False
        oForm.Items("TxtTax").Enabled = True
        oForm.Items("TxtTax").Specific.Value = CalculateTax(nTotal)
        oForm.Items("TxtTax").Enabled = False
        oForm.Update
    End If
End If
Else
    If pVal.ItemUID = "BtnOK" Then
        "\\ 保存数据，并关闭表单
        If SaveSalary = True Then
            oForm.Close
        End If
    End If
End If
End If
End Sub

Private Sub SBO_Application_MenuEvent(pVal As SAPbouiCOM.IMenuEvent, BubbleEvent As Boolean)

    If pVal.MenuUID = "MenuSalary" And pVal.BeforeAction = True Then
        CreateSalaryForm
    End If

End Sub

Private Sub AddSalaryFormToMenu()

    Dim oMenus      As SAPbouiCOM.Menus
    Dim oMenuItem   As SAPbouiCOM.MenuItem

    Set oMenus = SBO_Application.Menus
    Set oMenuItem = oMenus.Item(4) '模块

    Set oMenus = oMenuItem.SubMenus.Item(10).SubMenus '人力资源
    Set oMenuItem = oMenus.Add("MenuSalary", "计算薪水及所得税", mt_STRING, oMenus.Count)

End Sub

Private Function CreateSalaryForm()

    Dim oltem      As SAPbouiCOM.Item

    Dim oButton    As SAPbouiCOM.Button
    Dim oStaticText As SAPbouiCOM.StaticText

```

```

Dim oEditText As SAPbouiCOM.EditText
Dim oComboBox As SAPbouiCOM.ComboBox
Dim oCol As SAPbouiCOM.Column
Dim oMatrix As SAPbouiCOM.Matrix
Dim oRst As SAPbobsCOM.Recordset
Dim i As Long

'// 添加新表单
Set oForm = SBO_Application.Forms.Add("FormSalary", ft_Fixed)

oForm.DataSources.UserDataSources.Add "DocDate", dt_DATE, 10
oForm.DataSources.UserDataSources.Add "EmpID", dt_SHORT_NUMBER, 4
oForm.DataSources.UserDataSources.Add "DocTotal", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Tax", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Type", dt_SHORT_TEXT, 1
oForm.DataSources.UserDataSources.Add "Amount", dt_SUM, 16
oForm.DataSources.UserDataSources.Add "Ref", dt_SHORT_TEXT, 100

'// 设表单属性
oForm.Title = "薪水及个人所得税"
oForm.Left = 400
oForm.Width = 400
oForm.Top = 100
oForm.Height = 300

'//*****
'// 添加控件并设置属性
'//*****

'//*****
'// 添加确定按钮
'//*****

Set oltem = oForm.Items.Add("BtnOK", it_BUTTON)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 250
oltem.Height = 19

Set oButton = oltem.Specific

oButton.Caption = "添加"

'//*****
'// 添加取消按钮

```

```

'//*****

Set oltem = oForm.Items.Add("2", it_BUTTON)
oltem.Left = 75
oltem.Width = 65
oltem.Top = 250
oltem.Height = 19

Set oButton = oltem.Specific

oButton.Caption = "取消"

'//*****
'// 添加员工 ID
'//*****

Set oltem = oForm.Items.Add("CboEmpID", it_COMBO_BOX)
oltem.Left = 75
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14
Set oComboBox = oltem.Specific
oComboBox.DataBind.SetBound True, "", "EmpID"

'// 初始化下拉菜单
Set oRst = SBO_Company.GetBusinessObject(BoRecordset)
oRst.DoQuery ("SELECT EmpID, RTRIM(LastName) + ' ' + RTRIM(MiddleName) + ' ' + RTRIM(FirstName)
AS Name " + _
    " FROM OHM")
oRst.MoveFirst
For i = 1 To oRst.RecordCount
    oComboBox.ValidValues.Add oRst.Fields("EmpID").Value, oRst.Fields("Name").Value
    oRst.MoveNext
Next

Set oltem = oForm.Items.Add("StxtEmpID", it_STATIC)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14

oltem.LinkTo = "CboEmpID"

Set oStaticText = oltem.Specific

```

```
oStaticText.Caption = "员工编号"

'//*****
'// 添加员工姓名
'//*****

Set oltem = oForm.Items.Add("TxtEmpNam", it_EDIT)
oltem.Left = 75
oltem.Width = 65
oltem.Top = 20
oltem.Height = 14
oltem.Enabled = False

Set oltem = oForm.Items.Add("StxtEmpNam", it_STATIC)
oltem.Left = 5
oltem.Width = 65
oltem.Top = 20
oltem.Height = 14

oltem.LinkTo = "TxtEmpNam"

Set oStaticText = oltem.Specific

oStaticText.Caption = "员工姓名"

'//*****
'// 添加过帐日期
'//*****

Set oltem = oForm.Items.Add("TxtDocDate", it_EDIT)
oltem.Left = 315
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14
Set oEditText = oltem.Specific
oEditText.DataBind.SetBound True, "", "DocDate"

Set oltem = oForm.Items.Add("StxtDocDat", it_STATIC)
oltem.Left = 245
oltem.Width = 65
oltem.Top = 5
oltem.Height = 14

oltem.LinkTo = "TxtDocDate"
```

```
Set oStaticText = oltem.Specific

oStaticText.Caption = "过帐日期"

'//*****
'// 添加 Matrix
'//*****

Set oltem = oForm.Items.Add("MxLine", it_MATRIX)
oltem.Left = 5
oltem.Width = 395
oltem.Top = 40
oltem.Height = 180
Set oMatrix = oltem.Specific

Set oCol = oMatrix.Columns.Add("Col0", it_EDIT)
oCol.TitleObject.Caption = ""
oCol.Width = 0
oCol.Visible = False

Set oCol = oMatrix.Columns.Add("Col1", it_COMBO_BOX)
oCol.TitleObject.Caption = "类型"
oCol.Width = 80
oCol.DisplayDesc = True
oCol.DataBind.SetBound True, "", "Type"

Set oCol = oMatrix.Columns.Add("Col2", it_EDIT)
oCol.TitleObject.Caption = "金额"
oCol.Width = 120
oCol.RightJustified = True
oCol.DataBind.SetBound True, "", "Amount"

Set oCol = oMatrix.Columns.Add("Col3", it_EDIT)
oCol.TitleObject.Caption = "备注"
oCol.Width = 175
oCol.DataBind.SetBound True, "", "Ref"

'//*****
'// 添加总计
'//*****

Set oltem = oForm.Items.Add("TxtTotal", it_EDIT)
oltem.Left = 300
oltem.Width = 80
oltem.Top = 215
```

```
oltem.Height = 14
oltem.Enabled = False
oltem.RightJustified = True
Set oEditText = oltem.Specific
oEditText.DataBind.SetBound True, "", "DocTotal"

Set oltem = oForm.Items.Add("StxtTotal", it_STATIC)
oltem.Left = 230
oltem.Width = 65
oltem.Top = 215
oltem.Height = 14

oltem.LinkTo = "TxtTotal"

Set oStaticText = oltem.Specific

oStaticText.Caption = "总计"

'//*****
'// 添加所得税
'//*****

Set oltem = oForm.Items.Add("TxtTax", it_EDIT)
oltem.Left = 300
oltem.Width = 80
oltem.Top = 230
oltem.Height = 14
oltem.Enabled = False
oltem.RightJustified = True
Set oEditText = oltem.Specific
oEditText.DataBind.SetBound True, "", "Tax"

Set oltem = oForm.Items.Add("StxtTax", it_STATIC)
oltem.Left = 230
oltem.Width = 65
oltem.Top = 230
oltem.Height = 14

oltem.LinkTo = "TxtTax"

Set oStaticText = oltem.Specific

oStaticText.Caption = "个人所得税"

'//设置菜单项状态
```

```
oForm.EnableMenu 1281, False '查找
```

```
oForm.EnableMenu 1282, True '添加
```

```
oForm.EnableMenu 1292, True '添加行
```

```
oForm.EnableMenu 1293, True '删除行
```

```
oForm.EnableMenu 1294, True '复制行
```

```
oForm.Visible = True
```

```
Set oForm = Nothing
```

```
End Function
```

```
Private Sub CboEmpID_Select()
```

```
Dim oComboBox As SAPbouiCOM.ComboBox
```

```
Dim oEditText As SAPbouiCOM.EditText
```

```
Dim oMatrix As SAPbouiCOM.Matrix
```

```
Set oComboBox = oForm.Items("CboEmpID").Specific
```

```
Set oEditText = oForm.Items("TxtEmpNam").Specific
```

```
oEditText.String = oComboBox.Selected.Description
```

```
'//*****
```

```
'// 类型下拉菜单初始化
```

```
'//*****
```

```
Set oMatrix = oForm.Items("MxLine").Specific
```

```
If oMatrix.RowCount = 0 Then
```

```
oMatrix.AddRow
```

```
Set oComboBox = oMatrix.Columns("Col1").Cells(1).Specific
```

```
oComboBox.ValidValues.Add "S", "工资"
```

```
oComboBox.ValidValues.Add "B", "奖金"
```

```
oComboBox.ValidValues.Add "A", "补贴"
```

```
End If
```

```
End Sub
```

```
Private Function CalculateTotal() As Double
```

```
Dim oMatrix As SAPbouiCOM.Matrix
```

```
Dim i As Long
```

```
'// 计算薪水总额
```

```
Set oMatrix = oForm.Items("MxLine").Specific
```

```
For i = 1 To oMatrix.RowCount
```

```

oMatrix.GetLineData (i)
CalculateTotal = CalculateTotal + oForm.DataSources.UserDataSources("Amount").Value
Next

```

```
End Function
```

```
Private Function CalculateTax(ByVal pnSalary As Double) As Double
```

```
    Dim nTaxable As Double
```

```
    '// 计算个人所得税
```

```
    nTaxable = pnSalary - 1200
```

```
    Select Case nTaxable
```

```
        Case Is <= 0
```

```
            CalculateTax = 0
```

```
        Case Is <= 500
```

```
            CalculateTax = nTaxable * 5 / 100
```

```
        Case Is <= 2000
```

```
            CalculateTax = nTaxable * 10 / 100 - 25
```

```
        Case Is <= 5000
```

```
            CalculateTax = nTaxable * 15 / 100 - 125
```

```
        Case Is <= 20000
```

```
            CalculateTax = nTaxable * 20 / 100 - 375
```

```
        Case Is <= 40000
```

```
            CalculateTax = nTaxable * 25 / 100 - 1375
```

```
        Case Is <= 60000
```

```
            CalculateTax = nTaxable * 30 / 100 - 3375
```

```
        Case Is <= 80000
```

```
            CalculateTax = nTaxable * 35 / 100 - 6375
```

```
        Case Is <= 100000
```

```
            CalculateTax = nTaxable * 40 / 100 - 10375
```

```
        Case Else
```

```
            CalculateTax = nTaxable * 45 / 100 - 15375
```

```
    End Select
```

```
End Function
```

```
Private Function ValidToSaveSalary() As Boolean
```

```
    ValidToSaveSalary = False
```

```
    '// 检查过帐日期
```

```
    If VBA.Trim(oForm.Items("TxtDocDate").Specific.String) = "" Then
```

```
        SBO_Application.SetStatusBarMessage "请输入过帐日期。", bmt_Long, True
```

```

        oForm.Items("TxtDocDate").Click ct_Regular
    Exit Function
End If

```

```
ValidToSaveSalary = True
```

```
End Function
```

```
Private Function SaveSalary() As Boolean
```

```
    SaveSalary = False
```

```

    Dim oUserTables As SAPbobsCOM.UserTables
    Dim oTblESS      As SAPbobsCOM.UserTable
    Dim oTblEST      As SAPbobsCOM.UserTable
    Dim oMatrix      As SAPbouiCOM.Matrix
    Dim i             As Long
    Dim sErrMsg      As String
    Dim lErrCode      As Long
    Dim lRetCode      As Long
    Dim oRst          As SAPbobsCOM.Recordset
    Dim sDocNum       As String
    Dim oJE           As SAPbobsCOM.JournalEntries
    Dim dDocDate      As Date
    Dim oSBObob       As SAPbobsCOM.SBObob

```

```
'// 合法性检查
```

```
If ValidToSaveSalary = False Then
```

```
    Exit Function
```

```
End If
```

```
'// 过帐日期类型转换
```

```
Set oSBObob = SBO_Company.GetBusinessObject(BoBridge)
```

```
Set oRst = oSBObob.Format_StringToDate(oForm.DataSources.UserDataSources("DocDate").Value)
```

```
dDocDate = oRst.Fields("Date").Value
```

```
Set oSBObob = Nothing
```

```
'// 开始一个 Transaciton 以保证数据完整行和一致性
```

```
SBO_Company.StartTransaction
```

```
'// 生成日记帐分录
```

```
Set oJE = SBO_Company.GetBusinessObject(oJournalEntries)
```

```
oJE.Memo = "计提应付工资"
```

```
oJE.ReferenceDate = dDocDate
```

```

// 借方：营业费用
oJE.Lines.Add
oJE.Lines.AccountCode = "55010101"
oJE.Lines.Debit = oForm.DataSources.UserDataSources("DocTotal").Value
// 贷方：应付工资
oJE.Lines.Add
oJE.Lines.AccountCode = "21510101"
oJE.Lines.Credit = oForm.DataSources.UserDataSources("DocTotal").Value

If oJE.Add <> 0 Then
    SBO_Company.GetLastError IErrCode, sErrMsg
    SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
    Exit Function
End If

// 取得下一个 DocNum
Set oRst = SBO_Company.GetBusinessObject(BoRecordset)
oRst.DoQuery ("SELECT TOP 1 U_DocNum AS DocNum FROM [@ESS] ORDER BY DocNum DESC")
If oRst.RecordCount = 0 Then
    sDocNum = "1"
Else
    sDocNum = VBA.CStr(VBA.CInt(oRst.Fields("DocNum").Value + 1))
End If

// 保存薪水及个人所得税一头表信息
Set oTblESS = SBO_Company.UserTables("ESS")
oTblESS.Code = AssignUID
oTblESS.Name = "ESS" + AssignUID
oTblESS.UserFields("U_DocNum").Value = sDocNum
oTblESS.UserFields("U_DocDate").Value = dDocDate
oTblESS.UserFields("U_EmpID").Value = oForm.DataSources.UserDataSources("EmpID").Value
oTblESS.UserFields("U_DocTotal").Value =
VBA.CDbI(oForm.DataSources.UserDataSources("DocTotal").Value)
oTblESS.UserFields("U_Tax").Value = VBA.CDbI(oForm.DataSources.UserDataSources("Tax").Value)
If oTblESS.Add <> 0 Then
    SBO_Company.GetLastError IErrCode, sErrMsg
    SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
    Exit Function
End If

Set oMatrix = oForm.Items("MxLine").Specific

// 保存薪水明细
Set oTblEST = SBO_Company.UserTables("EST")
For i = 1 To oMatrix.RowCount

```

```

oMatrix.GetLineData (i)
If oForm.DataSources.UserDataSources("Type").Value = "" Then
    '// 检查类型
    If oForm.DataSources.UserDataSources("Amount").Value <> 0 Then
        SBO_Company.EndTransaction wf_RollBack
        SBO_Application.SetStatusBarMessage "请选择薪水类型", bmt_Long, True
        oMatrix.Columns("Col1").Cells(i).Click
        Exit Function
    End If
Else
    oTblEST.Code = AssignUID
    oTblEST.Name = "ESS" + AssignUID
    oTblEST.UserFields("U_DocNum").Value = sDocNum
    oTblEST.UserFields("U_LineNum").Value = VBA.CStr(i)
    oTblEST.UserFields("U_Type").Value = oForm.DataSources.UserDataSources("Type").Value
    oTblEST.UserFields("U_Amount").Value =
VBA.CDbI(oForm.DataSources.UserDataSources("Amount").Value)
    oTblEST.UserFields("U_Ref").Value = oForm.DataSources.UserDataSources("Ref").Value
    If oTblEST.Add <> 0 Then
        SBO_Company.GetLastError lErrCode, sErrMsg
        SBO_Application.SetStatusBarMessage sErrMsg, bmt_Long, True
        Exit Function
    End If
End If
Next

SBO_Company.EndTransaction wf_Commit

SaveSalary = True

End Function

Private Sub SetEnableToFalse()

    If Not oForm Is Nothing Then
        oForm.Items("TxtEmpNam").Enabled = False
        oForm.Items("TxtTotal").Enabled = False
        oForm.Items("TxtTax").Enabled = False
        oForm.Update
    End If

End Sub

```

## 二、SubMain.bas

```
// SAP MANAGE UI API 6.5 SDK Sample
//*****
//
// File:      SubMain.bas
//
// Copyright (c) SAP MANAGE
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF
// ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO
// THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A
// PARTICULAR PURPOSE.
//
//*****
```

Option Explicit

Public Type POINTAPI

    X As Long

    Y As Long

End Type

// The message structure

Public Type Msg

    hwnd As Long

    message As Long

    wParam As Long

    lParam As Long

    Time As Long

    pt As POINTAPI

End Type

// Retrieves messages sent to the calling thread's message queue

Public Declare Function GetMessage Lib "user32" \_

    Alias "GetMessageA" \_

    (lpMsg As Msg, \_

    ByVal hwnd As Long, \_

    ByVal wParamFilterMin As Long, \_

    ByVal wParamFilterMax As Long) As Long

// Translates virtual-key messages into character messages

```
Public Declare Function TranslateMessage Lib "user32" _
    (lpMsg As Msg) As Long

'// Forwards the message on to the window represented by the
'// hWnd member of the Msg structure
Public Declare Function DispatchMessage Lib "user32" _
    Alias "DispatchMessageA" _
    (lpMsg As Msg) As Long

Public Msg As Msg

Public Sub Main()

    Dim SalaryForm As SalaryForm
    Set SalaryForm = New SalaryForm

    Do While GetMessage(Msg, 0&, 0&, 0&)
        TranslateMessage Msg
        DispatchMessage Msg
        DoEvents
    Loop

End Sub

Public Function AssignUID() As String
    AssignUID = CStr(VBA.Timer)
End Function
```

## 教材说明

SAP Business One 中文版培训教材共分为两个子系列，一个系列为产品功能部分，主要介绍 SAP Business One 产品的核心功能。包括【敏捷财务】、【敏捷销售】、【敏捷采购】、【敏捷库存】、【敏捷服务】、【敏捷系统管理】共六本书；另外一个系列为开发工具部分，介绍了 SAP Business One 开发工具包-SDK 的使用。主要包括【SDK 培训教材】一书。该套教材提供给 SAP 公司中国的 SAP Business One 合作伙伴以及最终用户使用。

SAP Business One SDK 中文版培训课程主要为合作伙伴开设，课程主要以 SDK 的开发功能和流程为主线，讲解了 SAP Business One SDK 工具包中 DI API、UI API、Java Connector 和 Screen Painter 的主要功能。并将 SDK 6.5 版新增的功能专门列出，课程后附相应的习题以供练习，帮助用户了解开发原理和实现过程。

本版产品功能系列教材由 SAP 中国公司、SAP Business One 中国合作伙伴共同编写，由于时间有限，书中难免有遗误，请广大读者不吝指教。

参与本版产品功能系列教材编写的人员有：王文莉、卢军、侯天喜、朱勇伟、宋长青等。感谢所有参与人员。

SAP 中国及其合作伙伴联络方式：

SAP 中国  
上海市南京西路 388 号仙乐斯广场 31 层 200003  
电话：021-3302 4898  
传真：021-6334 5758

清华紫光股份有限公司  
北京市海淀区紫光大厦写字楼 10 层 100084  
电话：010-6279 1808  
传真：010-6279 1804

北京华软新元信息技术有限公司  
北京市朝阳区小营路 10 号阳明广场北楼 14 层 100101  
电话：010-8462 9126  
传真：010-8462 9137

方正科技集团股份有限公司  
北京市海淀区成府路 298 号中关村方正大厦 8 层 100871  
电话：010-8252 9999  
传真：010-8252 9566

汉普管理咨询（中国）有限公司  
北京市海淀区上地创业园联想北区研发大厦 G 座 6 层 100085  
电话：010-5886 3888  
传真：010-5886 3316