1 Winform 界面风格

Phenix 在 Phenix.Windows 程序集里为 WinForm 桌面软件研发提供了一整套界面框架（比如 BarManager 等组件），规范了关键的界面交互逻辑。

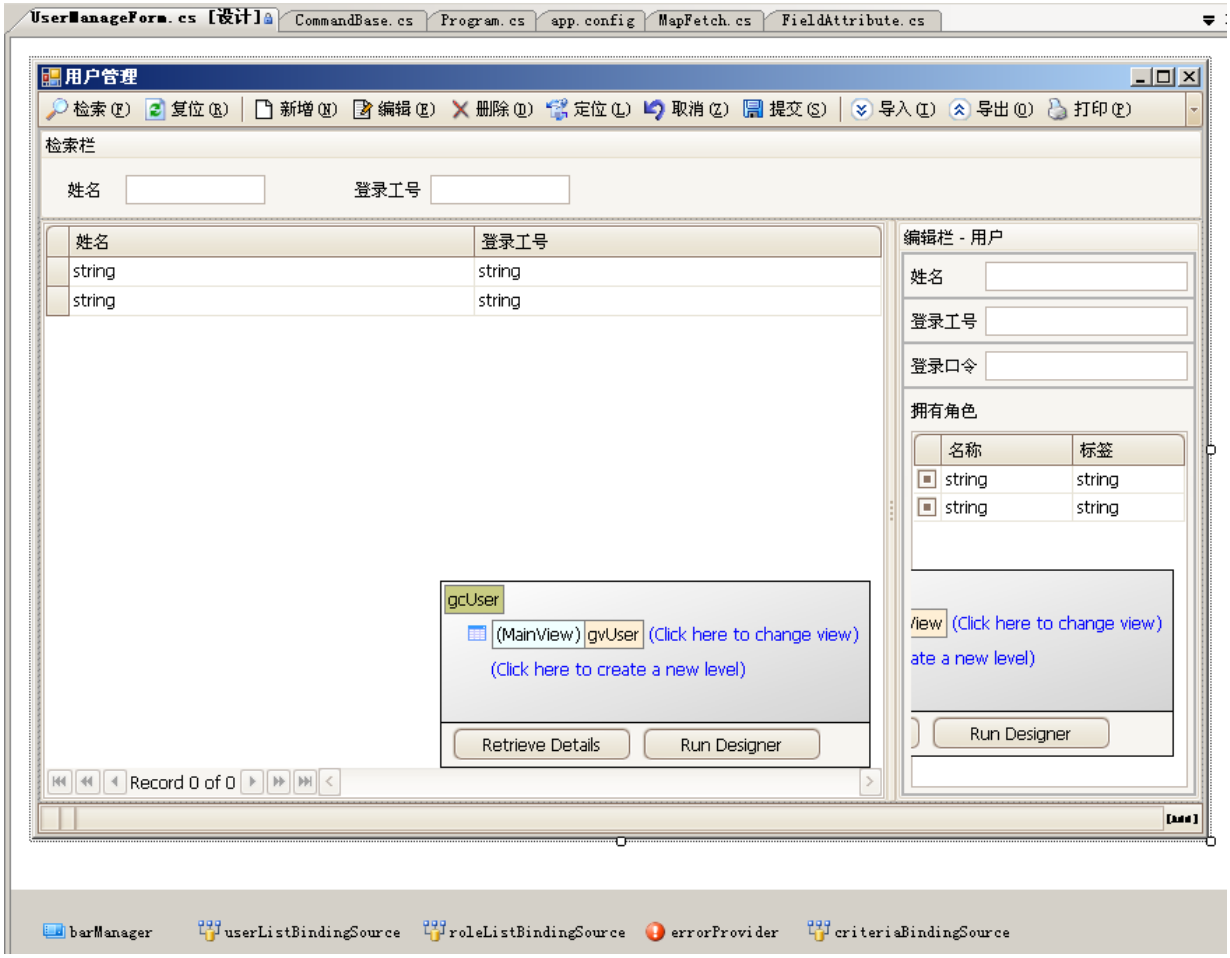
本文对 WinForm 界面的整体风格做一个规范性的描述，为此类研发项目的设计规范提供一个参考。

1.1 界面布局

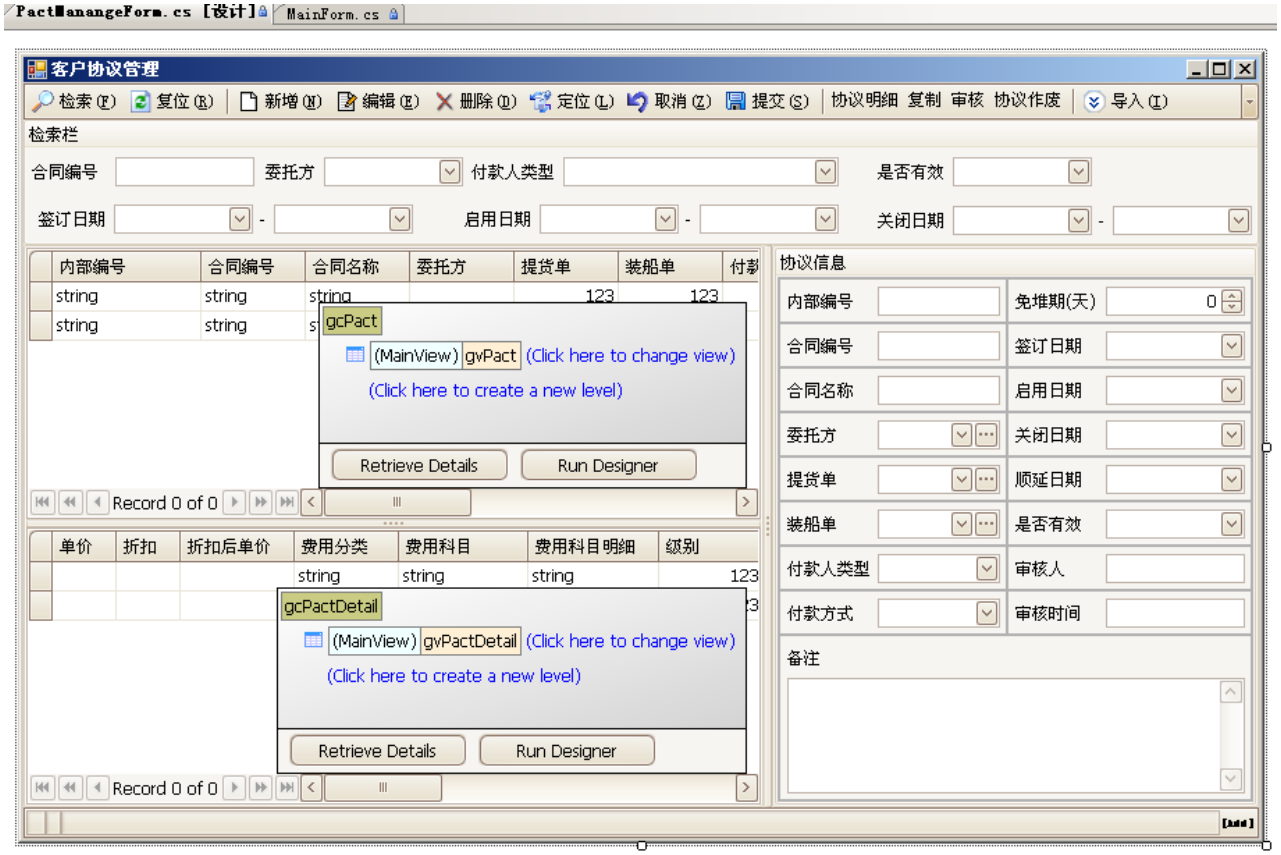
原则上，按照下述方式布局：

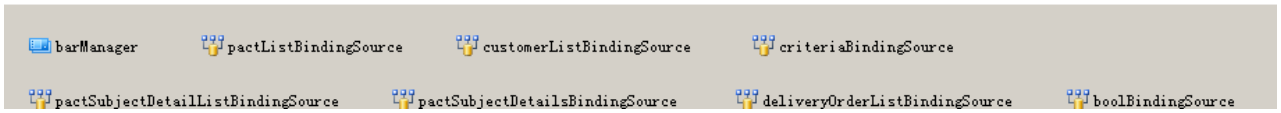
- 顶部、底部：为工具栏、状态栏，可拖放 Phenix.Windows.BarManager 组件搭建；
- 上部：为检索栏，用 Phenix.Windows.CriteriaGroupControl 占位，按照设计要求拖放编辑控件，与查询类绑定，并建议归集到 DevExpress.XtraLayout.LayoutControl 里；
- 中部左侧：一般用于数据清单的导航，可用 DevExpress.XtraGrid.GridControl 组件，按照设计要求与业务类绑定；
- 中部右侧：一般用于数据清单中当前对象的编辑，用 Phenix.Windows.BusinessGroupControl 占位，按照设计要求拖放编辑控件，与业务类绑定，并建议归集到 DevExpress.XtraLayout.LayoutControl 里；

下图是单数据集界面示例：

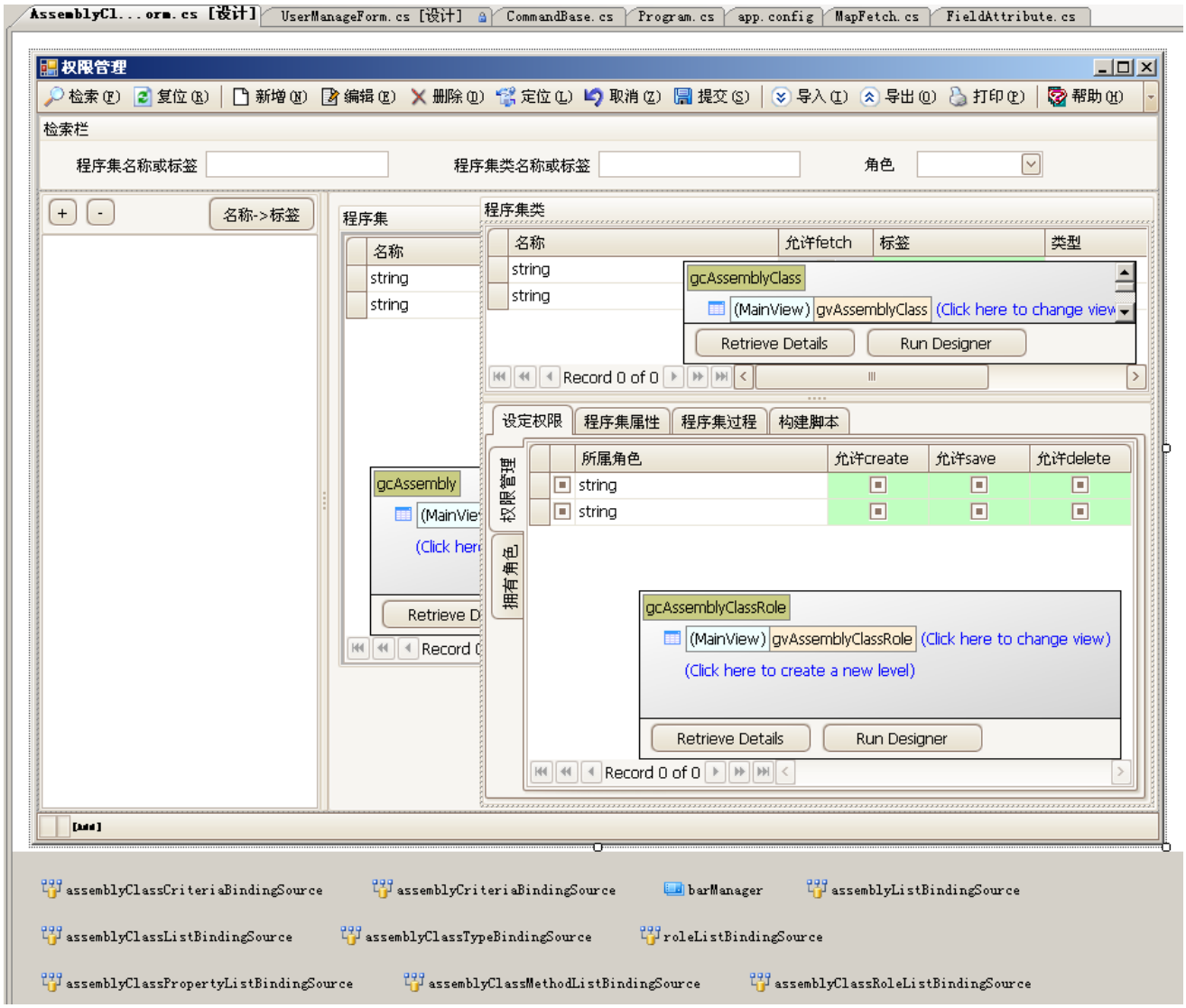


下图是主从数据集界面示例：





下图是树状结构界面示例：

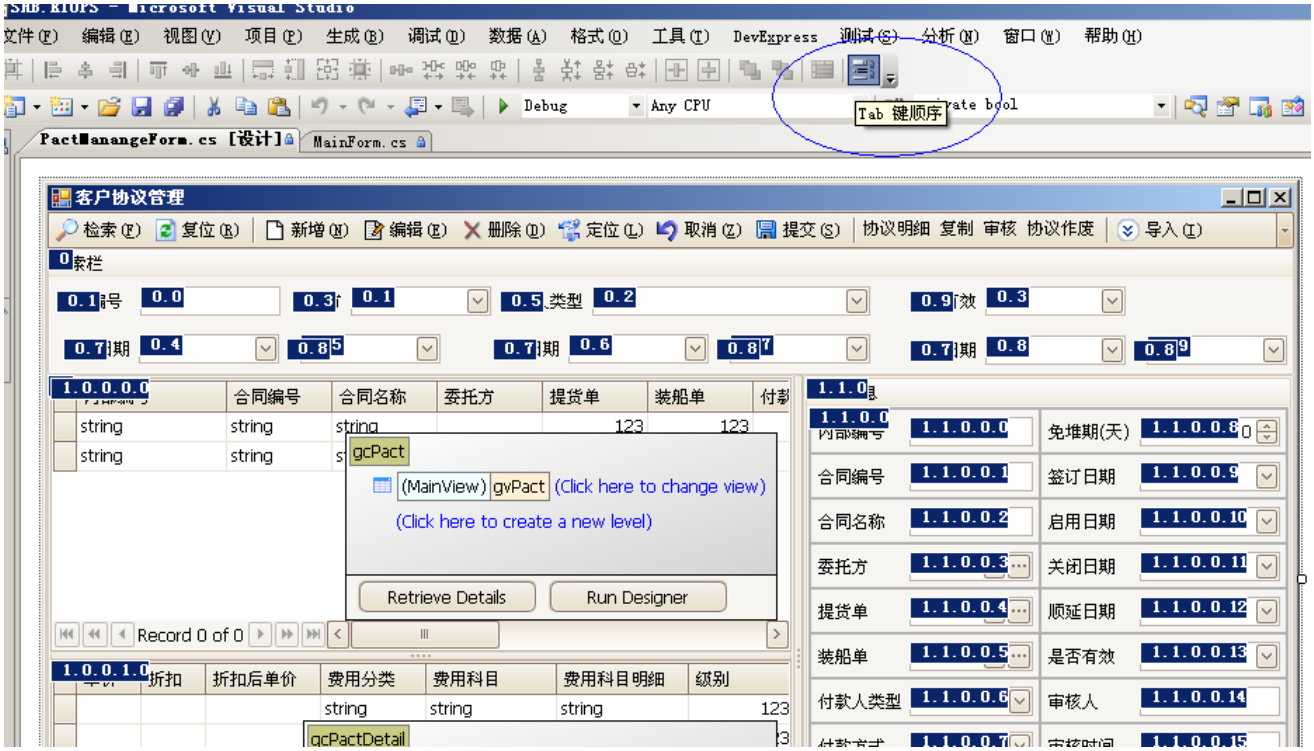


上述示例仅做参考。

除了工具栏、状态栏、检索栏（原则上需要固定它们的布局）外，中部区块主要用于业务数据的操作，需要根据具体的设计要求进行布局，并不需要拘泥于上述这几种形式。

1.2 Tab 键顺序

在 Windows 工程编码完毕、提交测试之前，开发者必须自行检查界面 Tab 键顺序。



1.3 界面美观

界面是否美观、整洁，直接影响到用户是否乐意使用你的软件。

- 除非特殊场景，字体用黑色、细线、5 号、宋体。
- 除非特殊场景，文字界面的窗体内不允许出现 3 种颜色的字体（/背景），图形界面也应该控制颜色数量，以保证清爽耐看，不会造成视疲劳。

提交用户的界面，必须经过 IDE 的排版功能校正位置，以保证：

- 纵向的标题栏，右对齐；
- 纵向的输入框，左右两端对齐；
- 标题栏和输入框的间隔为 1—2 个字符之间；
- 左右的纵列（标题栏的最左端+输入框的最右端为一纵列），间隔相同，间距为 4—6 个字符之间；
- 左右的控件，它们的横轴在一条直线上；
- 上下的控件，间隔相同，间距为 1/3—1/2 个字符之间；
- 业务对话框上的按钮，纵向排放在界面右侧上方，主按钮（一般为确认按钮）放在最上方，取消按钮放在最下方。

1.4 友好提示

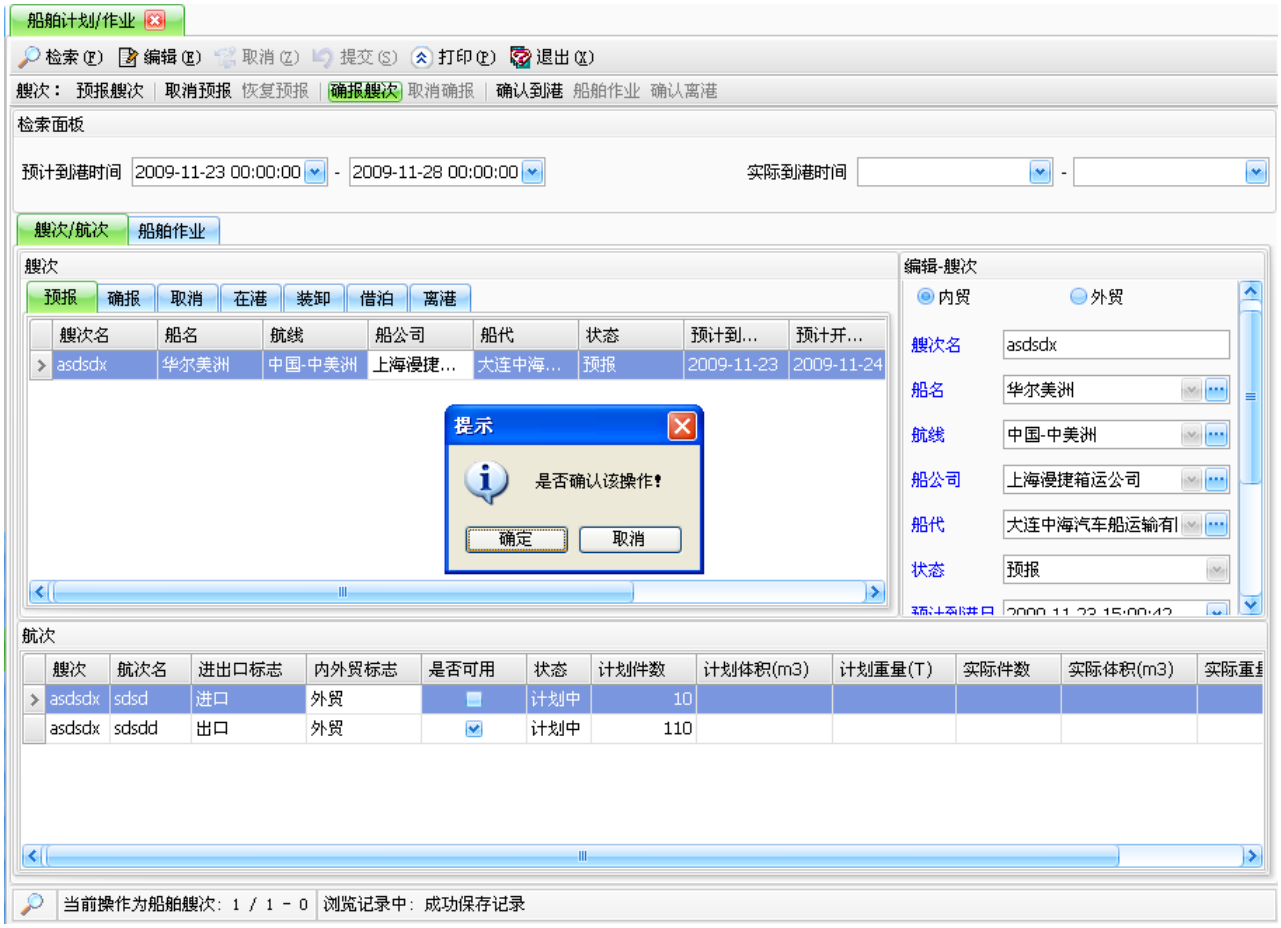
明确、及时、不唐突的提示也很重要，可大幅提升软件的界面交互友好性。

如果有必要，可提供常规提示（比如保存数据后是否需要弹出确认对话框等）的配置功能，让用户

根据自身需要自行取舍。

- 通过状态条等方式，实时显示当前的运行状态；
- 对于关键（重要、危险）操作，给出指示性信息；
- 对于没有明显返回结果的操作，给出操作结束的提示，例如：查询未检索到数据应该给出提示；
- 对于非法操作，给出明确的提示信息，并通过光标聚焦等方式引导用户纠正错误；
- 对于耗时较长（超过 3 秒）的操作，应该给出计算进度状态条，并尽可能允许用户中止操作；
- 提示信息应该明确当前操作的数据，以及本功能操作所带来的影响或结果等；

下图是反例（提示信息含糊不清）：



1.5 下拉选择框

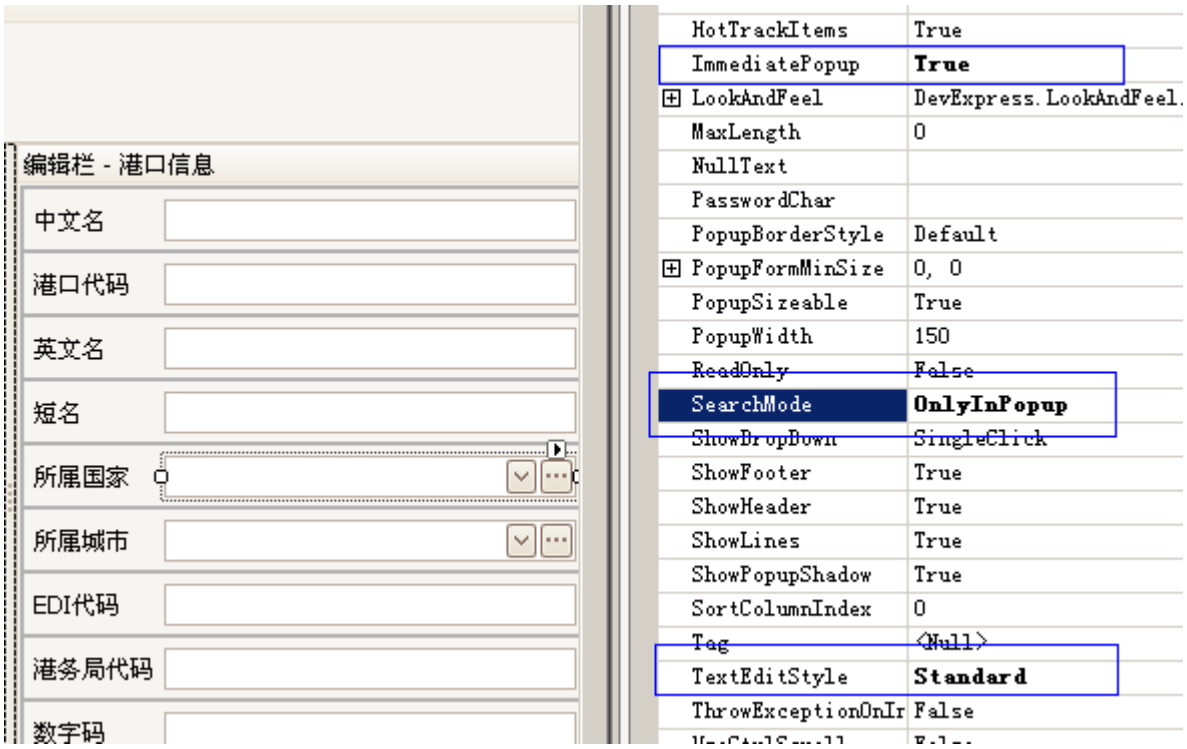
LookUp 控件的 Properties.DisplayMember，不应该是 ID 之类无意义的内容，而应该是关联对象的名称属性、代码属性等能让用户看得懂的内容。

如果输入内容的范围仅局限在于关联对象集合内，那么应该如下设置：

```
LookUp 控件的 Properties.ImmediatePopup = true;
LookUp 控件的 Properties.SearchMode = DevExpress.XtraEditors.Controls.SearchMode.OnlyInPopup;
```

LookUp 控件的 Properties.TextEditStyle = DevExpress.XtraEditors.Controls.TextEditStyles.Standard;

如下图：



下拉选择框中字段排列顺序，一般为：第一列为输入字段，第二列为输入框的显示字段，再后面的列起到补充提示作用。

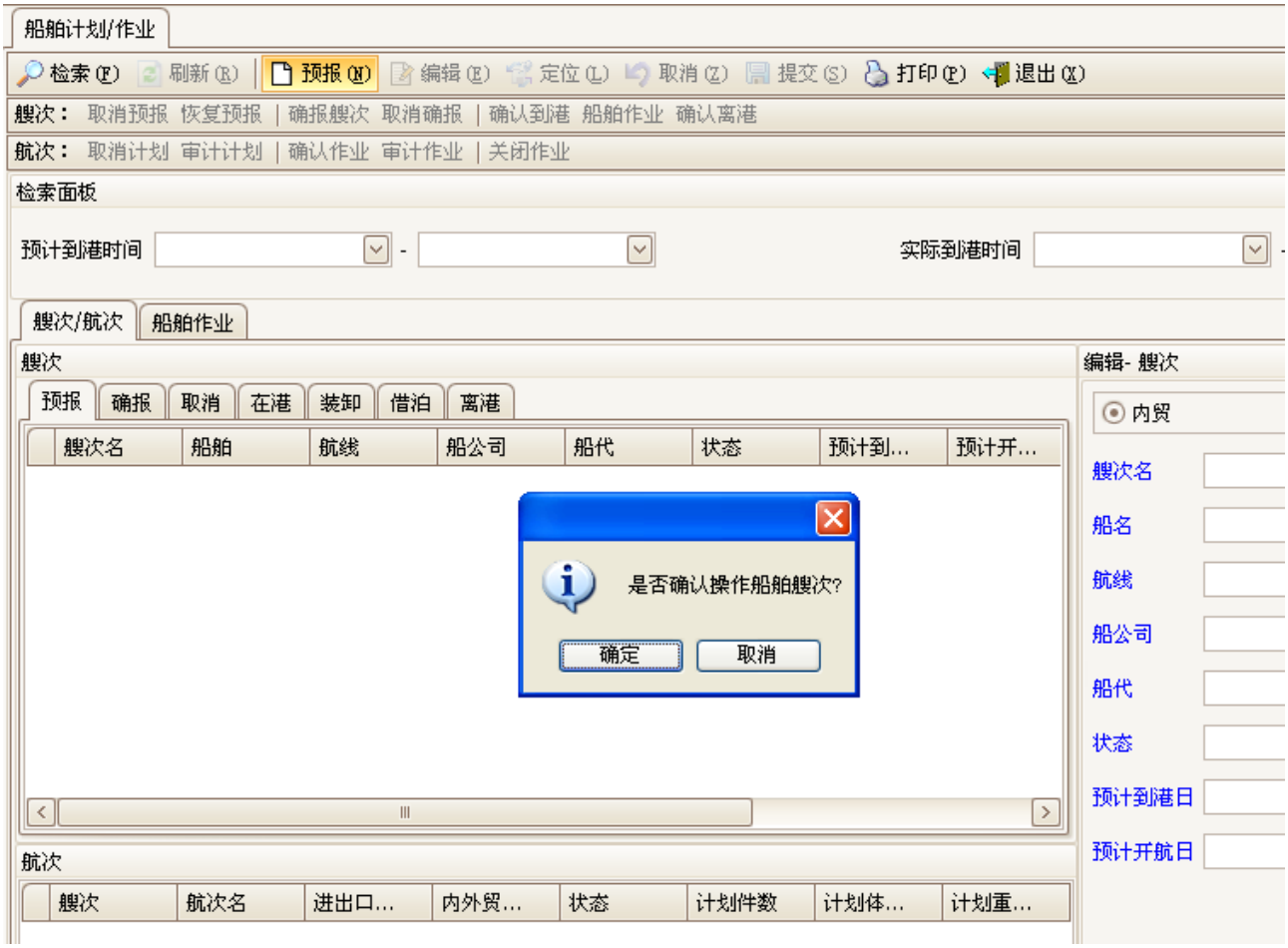
如果存在多列的情况，需要带上字段标题。

下图为一经典案例：

1.6 流畅的操作感受

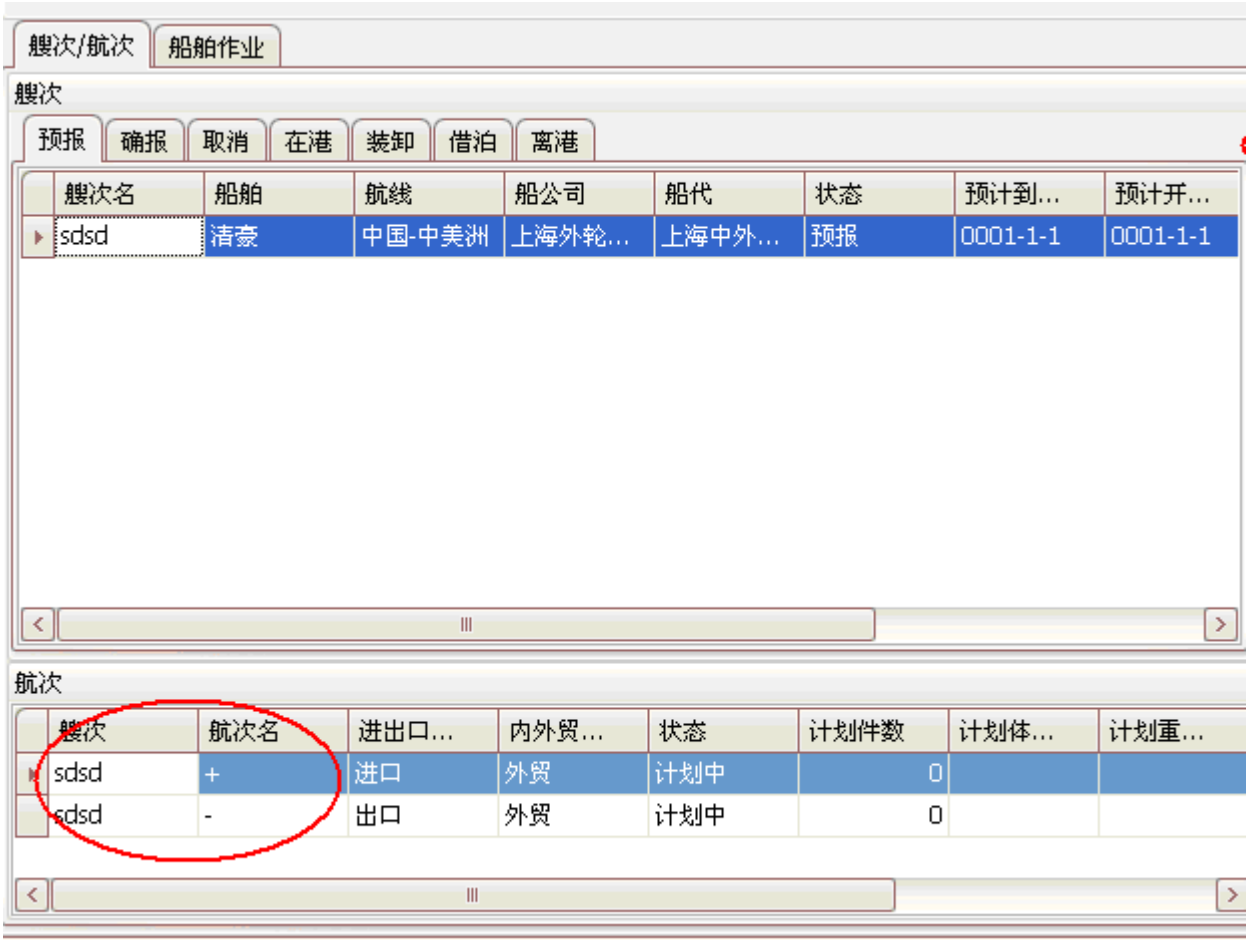
首先，要避免无必要、高频率的提示。

下图是个反例：“预报”功能按钮实质上是新增数据，只有点击“提交”功能按钮才会被持久化到数据库，之前是没有任何风险的，况且还能通过“取消”功能按钮来清除掉这些数据，所以不必做任何提示。



其次，要在用户日常频繁操作的界面上下功夫。比如，在编辑界面上提供批量编辑多条记录的功能，或者新增数据提交成功后可以直接进入下一轮的新增界面，而不是每次都让用户去点击“新增”功能按钮。

见下图：



1.7 按钮灰亮控制

按钮灰亮的控制起到两种作用：

- 操作权限控制：优先级最高，基本由框架实现。
- 业务逻辑控制：除特殊业务场景外，不应该在点击功能按钮时弹出对话框，提示说本功能因为什么什么原因不允许使用；而应该根据当前操作的数据状态、业务流程状态等判断条件，将功能按钮置为不允许操作的灰色显示状态。

1.8 多记录的选择

在多记录的选择操作场景下，应该提供全选、反选等辅助功能按钮，尤其当记录数大于 10 条以上时，这样的设计非常必要。

1.9 界面导航

界面导航功能，一般用于通过当前记录导航到其关联记录的浏览或编辑界面上。

原则上，代码表都要能导航，并（在权限允许的情况下）允许编辑它。编辑成功后，应该自动刷新本界面的代码表内容（比如下拉列表等）（本功能可配置为框架自动完成）。

导航功能必须采取插件的方法实现，以避免工程之间产生复杂的依赖关系（指工程引用等）。

下图为实现效果示例：