

第3章 使用视图进行工作

3.1 关于视图

就像表单是Notes输入数据的主要组成部分一样，视图是Notes查看信息的主要组成部分。在一个Notes视图中，每行代表一个文档，一列代表文档中的一个域或由文档提供的信息计算的值。一列也可以就是与文档无关而只是为视图的某种目的服务的一个静态文本或一个固定数字。

1. 视图与文件夹

开发者根据文档的内容、创建日期或者其他特性，使用公式决定哪些文档包含在视图中。如果一个文档符合视图的要求，它就被包含进来，否则，不包含它。

文件夹可以给予用户更多的灵活性，因为文件夹中包含的文档不像视图那样规定严格。另一方面，一个文件夹与视图行为表现基本相同。用户可以从文件夹中拖出或拖进文档而不作任何改变，他们可以从文件夹中移出文档，而不真正地从数据库中删除文档。

注意文件夹的灵活性同时也是它的缺点，用户可以把一个并非为设计中应该显示的特殊文档拖进文件夹中。

2. 视图的功能

这儿的大部分功能同样适用于文件夹，如果有所不同将具体指出它。

以用户的观点，视图的主要功能是为每个文档显示足够多的信息以便用户识别它，并且允许用户在视图中打开文档，有时你还可以看到文档组的摘要信息，比如总计或平均值。但是对于开发者来说，你应该了解关于视图的更多的信息：

- 访问限制：你可以定义一个访问列表规定谁可以打开一个特定的视图。注意这不是一个真正的安全功能，因为它不能阻止用户自己打开某个文档。
- 以层次结构显示主文档和答复文档：你可以创建一个包含主文档，答复文档，答复的答复文档的视图，并以层次结构显示它，在讨论数据库中，就是根据这样的思路显示的。
- 自动展开和折叠类：你可以决定当用户第一次打开视图时，视图中的类应该是展开的还是折叠的。
- 创建一个多行的文档：你可以允许每个文档包含9行，对于长标题行或多值域来说非常有用。
- 创建视图操作：你可以创建一个视图操作执行普通操作任务（如显示一个帮助文档）或在视图的一个或多个文档的操作。
- 以日历形式显示一个视图：对于一个对日期或时间很敏感的视图来说，将其显示为日历形式是很有帮助的。

视图对于开发者来说，其另外一个重要性在于它提供了一条使用脚本或者公式获得数据的方便有效的途径。改变一个视图的设计，对使用脚本，程序或者公式从视图中获得数据的应用程序将造成致命的错误。作为开发者来说，除非只是做一些外观的变化，必须保证了解视图被使用在何种地方（改变行的次序或添加一个行可不仅仅是外观的变化！）。

视图还被用来进行全文搜索，当用户对数据库进行全文搜索的时候，是在搜索一个特别的视图而非真正对数据库进行全文搜索。

总之，视图和文件夹是在一系列文档中查询或显示信息的非常方便的方法。显示以列表方式进行，从而使我们方便的查询一个文档，并且在人或程序查询信息方面提供帮助。

大量的文档资料教我们一步一步地创建 Notes视图，添加列，使用上面提到的功能。但是，它并没有火箭上天那样复杂，它主要由对话框中的选择属性组成。本章的其余部分假设你已经精通怎样创建视图，我们只是介绍怎样添加一些可以在 Web中使用的功能，以便使你的视图界面更友好，功能更强大。

3.2 视图和Web

作为Domino可以运行的前提你不必对视图做任何特殊的事情。默认情况下，Domino把视图转换为HTML表并提供一个标准的操作，比如搜索或展开。如果视图被分类，Domino显示一个三角标志，单击三角标志把该分类展开。图 3-1是以Domino形式显示的视图。



图3-1 默认的Domino形式视图

Domino可以把大量 Notes视图形式的功能转换为 HTML形式。图 3-2显示了一个在浏览器上显示的日历形式视图（这是在我的 Notes信件数据库中的日历形式的视图）。日历式视图可

以保留项目日程或其他重要数据的记录。这个视图还包含一个列图标，当任务完成的时候显示82号图标（一个核对标志）。



图3-2 通过HTML表格Domino可以显示日历视图

被包含在Domino表单中的格式属性同样可以包含在视图中。例如，如果你在 Notes中把列的属性设置为加粗，斜体，居中，则 Domino将以这些特性显示这些列。如果 Notes视图有列总计，背景色，Domino视图将反应出来（尽管Domino不支持列颜色的变化）。与Notes视图一样，Domino视图可以包含视图的操作按钮，帮助用户导航，组成新的文档甚至运行 LotusScript代理。另外，Domino的HTML通用文本功能允许你通过在列的公式里包含 HTML标签改变一个视图的外观，这是显示视图的灵活性的体现。

Domino还提供一个重要的格式化功能：在表单中嵌入一个视图，你可以使用这种功能修饰显示视图的页面的外观，增加一些环绕的图形或其他元素，或者一个用户可以输入搜索请求的域。

3.3 使用选择公式

视图的选择公式决定在视图将包含那些文档，选择公式使用如下格式：

SELECT condition

视图的选择公式可能就像默认的选择数据库中所有文档 SELECT @ALL一样简单，或者也可能很复杂，如：

```
SELECT (Form *= "Income" : "Expense" AND Amount > 5000) OR
(Priority="High")
```

这个视图选择公式选择了所有优先权为高的文档以及收入或支出额大于 5000 的文档。

3.4 使用视图列公式

和计算域中的返回值的公式一样，一个列公式返回一个值（而不是执行某个操作）。一个列公式的最终结果必须是一个字符串，换句话说，列公式的值可以是除了富态文本以外的任何类型数据。RTF域不能在列公式中使用。

列公式只在数据库内部使用以便保持视图的索引。那些需要与用户或其他数据库进行交互的函数，例如 @Prompt和@DbLookup不能被使用在列公式中。表 3-1 给出了几个例子。

表3-1 列公式例子

公 式	描 述
UnitPrice	返回UnitPrice域的值
UnitPrice*Quantity	返回UnitPrice域的值与Quantity域的值的乘积（如果其中有一个不是数字类型则返回一个数据类型不匹配错）
Itemtotal/Quantity	返回Itemtotal域的值除以Quantity域的值的商，如果Quantity域的值为0则返回@ERROR（如果其中有一个不是数字类型则返回一个数据类型不匹配错）
FirstName+ " " +LastName	把FirstName，空格，LastName连接在一起返回一个字符串，例如FirstName的值是Gray，LastName的值是Cole,则返回Gray Cole
Price+Tax	返回Price和Tax的和
@If(Status = " Complete " ; 85;Status 5 " Overdue " ;87;0)	如果状态是完成返回 85，延误，返回 87，否则返回 0（对于显示图标 的列来说，文档完成返回笑脸，文档延误显示哭脸，否则不显示图标）
" Last modified: " + @Text(@Modified)	显示文档最后更改的时间和日期，例如： Last modified:08/01/98 06:16:19 PM

3.5 使用表单公式

一个表单公式决定使用特定视图打开文档以后使用哪个表单。Notes使用下面的逻辑决定显示时使用哪个表单。

- 如果表单被保存在文档中，显示文档时总是使用该表单。表单公式不能改变它。
- 如果表单没有被保存在文档中而且视图有一个表单公式，则公式返回的表单就是将要显示中被使用的表单。
- 如果没有表单公式或者表单公式返回一个无效值，则由表单域决定使用哪个表单。
- 如果表单域中的值无效，则使用默认的数据库表单。

在决定使用哪个表单显示内容上表单公式非常得心应手。例如，我曾经开发过的一个工作流软件把一般的新闻故事文档转换为一个在 Web上发行的数据库。从不同门类收集的新闻故事（体育的，商业的等）被以不同的表单显示。为了完成这种功能，我只是为每个门类创建一个表单，然后使用一个表单公式来根据门类决定使用哪个表单。

关于这个主题的一点变化是可以为每个门类创建一个分离的视图，在每个视图中使用一

个表单公式，这样一来，如果一个新闻故事属于两个门类（如世界杯属于体育类又属于国际类），则可以根据用户的切入点的不同以不同的表单显示出来。

3.6 使用视图图标

你可以使用 Notes 提供的小图标传递文档中的信息，在此你只需要在视图的列属性中选择 “display values as icons” 列公式可以返回一个对应于你想使用的图标的数字。图 3-3 显示了包含当前图标组的视图（这个列表将不断扩充）。

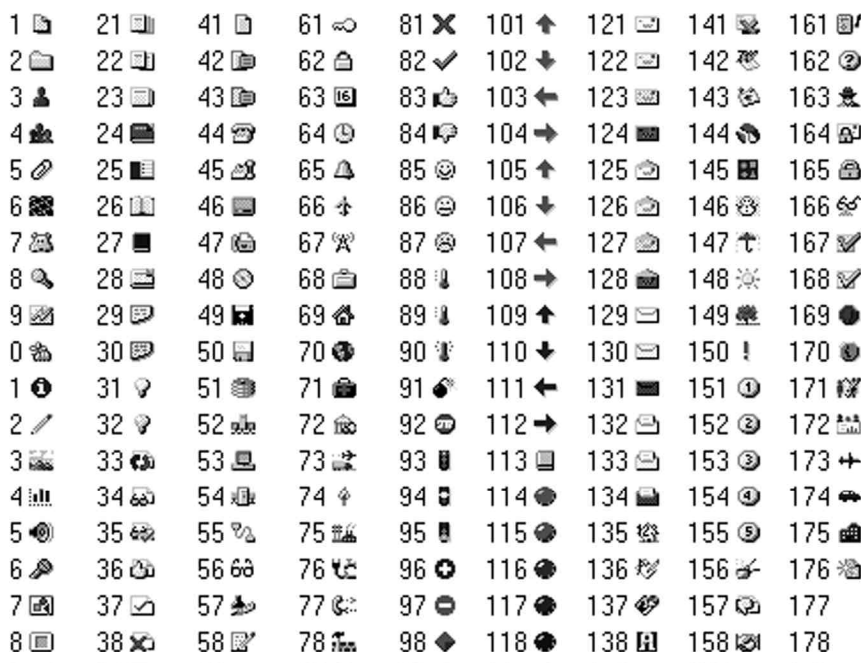


图3-3 在Domino中可以使用的小图标

上面这个视图可以在 CD 中的 ViewIcons.nsf 中找到，并详细说明了如何在视图添加图标。为了创建这个视图，我首先创建一个 Num 表单，在其中包含一个同样命名为 Num 的数字域，然后创建 20 个文档，数字为从 1 到 20，最后创建一个 ViewIcons 视图，该视图中的各列如表 3-2。

表3-2 在ViewIcons视图中的各列

列	公 式	是否显示为图标
1	Num	
2	Num	是
3	Num+20	
4	Num+20	是
5	Num+40	
6	Num+40	是

(续)

列	公 式	是否显示为图标
7	Num+60	
8	Num+60	是
9	Num+80	
10	Num+80	是
11	Num+100	
12	Num+100	是
13	Num+120	
14	Num+120	是
15	Num+140	
16	Num+140	是
17	Num+160	
18	Num+160	是

一个最简单和最有用的例子是显示文档的状态。例如，如果文档是只读的下面的公式返回图标62（锁定标志）：

```
@If(@IsAvailable($Readers);62;0)
```

如果没有预先定义好的图标符合你的需要，你还可以创建你自己的图标并使用 HTML 通用文本创建一个到该图标的链接。在本章的后面将讲述在视图中使用 HTML 通用文本的例子。

如果你使用了视图的图标列，记住 Domino 把每个图标转换为一个 链接，用户将必须下载所有图标。如果你想对文档提供某些可视化信息的话，你还可以使用不同的颜色进行分类（见在视图中使用 HTML 通用文本）。使用不同的文本颜色表示不同的值，比如年龄或状态。这种技术不增加用户的下载时间，加以适当的应用也可以通过视图迅速传递许多重要的信息。

提示 创建一个小的 GIF 图片（1311 像素）并将其命名为 vwicnXXX.gif，XXX 代表三个十进制数。最好从 999 开始，依此递减，以免与 Domino 的某个图标的代号相冲突。把这个 GIF 文件放在 Domino 图标目录中即可。

一旦你完成了这个任务以后可以把这个图标的号码放进视图的图标列中，Domino 将可靠地告诉浏览器下载你的图片。（这将造成 Notes 和 Domino 的不一致，在 Notes 中，如果超出了限制，它显示了一个空白图标，在浏览器中，如果不存在 vwicnXXX.gif，你将得到“无法连接”图形）

此技巧由 DOVID GROSS 提供

3.7 创建视图操作按钮

你可以创建能在 Web 中运行的视图操作按钮（在 Notes 客户端你可以创建显示在菜单上的操作，但很显然，这不适用于 Web 应用程序）。

默认情况下，在 Domino 的 Web 视图上没有所谓的“选定”或“当前”文档。因此你不能

创建某个在视图的选定文档上运行的操作（对这种问题，在本章随后描写的视图 applet 中提供了一个解决办法，“behave like forms”视图提供了另外一种解决办法）。

尽管 Domino 不支持直接使用 LotusScript 的按钮，但是它可以在按钮中使用 @Command([ToolsRunMacro]) 运行一个 LotusScript 代理。当然，这个代理只可以使用 Web 支持的 LotusScript 功能。例如，不能在 Domino 视图中创建一个按钮使用 NotesUIWorkspace.DialogBox 方法显示一个对话框。

默认情况下，Domino 的操作按钮必须被转换为 Domino URL。如果它不能被转换为 URL，按钮不能被显示。表 3-3 显示了代表性的转换。

表 3-3 公式的转换

公 式	URL
@Command([Compose];"OrderForm")	http://server/db/OrderForm?OpenForm
@Prompt([OK];"Hello";"Hello there!")	不转换
@Command([OpenView];"Orders")	http://server/db/Orders?OpenView

如果在视图中显示了一个表单，标准的链接（Previous，Next，Expand，Collapse，Search）不再显示。如果你仍然想显示这些按钮，必须自己在用来显示视图的表单中重新创建这些按钮或热点。你所需要的公式在表 3-4 中列出。

表 3-4 按钮的公式

按 钮	公 式
Previous	@DbCommand("Domino";"ViewPreviousPage");
Next	@DbCommand("Domino";"ViewNextPage")
Expand	@Command([ViewExpandAll]);
Collapse	@Command([ViewCollapseAll]);
Search	@Command([ViewShowSearchBar]);

把这些按钮群集在子表单中会更加方便以后一起加入到表单中。

3.8 使用单个类视图

在 R5 中新增加了可以嵌入视图的单个类。使用这个功能，你可以开发单个类的视图，使其根据用户或某些其他准则动态处理数据。

它的运行原理是这样的。在一个表单或一个页面中嵌入一个视图，并为它制定一个单个类的公式，当视图被打开的时候，Domino 执行公式并只显示属于这个类的文档。例如，如果你有一个视图由用户名分类，制定如下公式可只显示当前用户的文档：

```
@UserName
```

如果你想得到更灵活的结果，可以根据打开视图的 URL 的参数决定类别。这里允许你使用 \$\$Return 公式或 WebQuerySave 代理对浏览器进行重定向。为此，只须简单地在显示视图的表单中包含 QUERY_STRING 域，然后使用如下公式：

```
@ReplaceSubstring(@Right(QUERY_STRING;"Category=");"+";" " ")
```

下面的URL使公式返回字符串“Java Applet”

```
http://server/db/form?OpenForm&Category=Java +Applet
```

如果没有指定一个类，则显示消息“No Documents Found”。如果指定一个类，但是视图中没有一个文档属于这个类，产生一个“Entry not Found in index”错。在SmartView.nsf数据库中的SingleCategory表单是一个包含一个嵌入的单个类视图的表单的例子。

例子：动态的单个类视图

Navigator3.04™ Navigator4.05™ Domino4.6.1™

Explorer3.0™ Explorer4.01™ Domino5.0™

单个类的视图现在非常时髦，但是动态内容的概念可以追述到使用URL参数决定显示视图。图3-4显示了一个SmartView.nsf数据库中的SelectView表单，这个表单包含三个可编辑的关键字域：FileType，Username和Title，它们代表了你可能想要使用的不同准则。



图3-4 允许用户选择视图和类

试着在三个域中的一个选择一个值并单击 Submit按钮。如果你让三个域都为空或者在超过一个域中选取了值，则下面的输入有效性公式产生一个错误信息：

```
selections := @Trim(FileType : Username : Title);  
num := @Elements(selections);  
@If(num != 1; @Failure("Please go back and select one file type,  
username, or title."); @Success)
```

否则，\$\$Return执行，浏览器重定向：

```
view := @If(FileType != ""; "Downloads+by+Category"; Username != "";  
"Downloads+by+Username"; "Downloads+by+Title");  
cat := @ReplaceSubstring(@Trim(FileType : Username : Title); " "; "+");  
url := "/RMKelleher/SmartView.nsf/Dynamic?OpenForm&View= " + view +  
"&Category= " + cat;  
 "[" + url + "]"
```


这个公式首先根据在哪个域中选择值决定打开哪个视图，域中的值决定在视图中选择那一类的文档。结果类似于如下的 URL：

```
/db/Dynamic?OpenForm&View=Downloads+by+Category&Category=Image
```

动态的表单包含三个域，如表 3-5 所示。

表3-5 动态表单中的域

域	公 式
QUERY_STRING	QUERY_STRING
View	@ReplaceSubstring(@Left(@Right(QUERY_STRING;"&View="); "&"); "+"; " ")
Category	@ReplaceSubstring(@Right(QUERY_STRING;"&Category="); "+"; " ")

完成这个例子只需要两个非常简单的公式。嵌入的视图使用下面的公式决定显示哪个视图：

View

嵌入的视图使用下面的公式决定在视图中显示哪个类：

Category

图3-5显示了该 Web 页的结果。

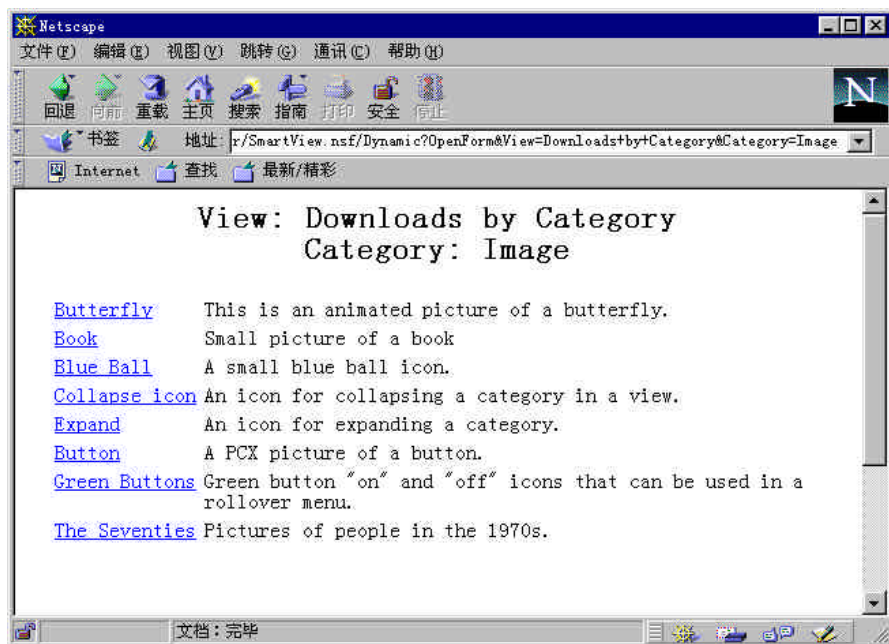


图3-5 这个表单使用URL参数决定显示那个视图以及在视图中显示那个类

3.9 在视图使用HTML通用文本

你可以在视图使用 Domino 的 HTML 通用文本功能，下面是在视图使用 HTML 通用文

本的几个例子：

- 改变行的格式。
- 显示Java小应用程序和图像。
- 改变一个视图连接的动作。
- 在行的基础上使用JavaScript。

在视图中使用HTML通用文本有两种不同的方式。以前，我们仅仅是选择把 HTML 标签用方括号括住，例如：

```
"[<h1>]" + Subject + "[</h1>]"
```

Domino的最新版本还允许你选择一个视图属性“把视图内容看做 HTML”。当启动这个属性的时候，整个视图的内容被看做 HTML 通用文本。

3.9.1 改变行的格式

你可以使用标准的 Notes 列属性定义特定列的所有行的文本颜色。但是当你只想改变某些行的颜色的时候该怎么办呢？你可以使用 HTML 标签的 COLOR 属性根据域值或其他准则改变文本的颜色。例如假设你想把危险的文档设置为红色，如果某个文档被分类为 Critical，下面列中的公式把 Subject 域的值显示为红色：

```
critical := @Contains(Category; "Critical");
color := @If(critical; "[<FONT COLOR= \"ff0000\">]"; "");
endcolor := @If(critical; "[</FONT>]"; "");
color + Subject + endcolor;
```

你可以在 HTML 中通过名称指定某种颜色，例如，标签 导致直到标签 的文本都以绿色显示，（其他颜色包括 red, blue, cyan, magenta, yellow, black, white）。绿色的代码是 00ff00，第一对十六进制数表示红色的数量（在这个例子表示没有红色），第二对十六进制数表示绿色的数量（在这个例子中表示绿色最高），第三对十六进制数表示蓝色的数量（在这个例子中表示没有蓝色），为了保证在 256 色显示器中仍然图像逼真，只使用由 00, 33, 66, 99, cc, ff 等组成的颜色。

发现某种颜色的 RGB 代码的方法是使用该颜色格式的某些文本，在 Domino 中显示这个文档，然后看 HTML 源代码。访问 <http://www.lynda.com/hex.html> 对我们也有很大的帮助。

你还可以在列公式中使用 HTML 标签改变特定的列的风格、对齐方式、大小等其他属性。例如，下面的列公式只对 Main Topic 文档的 Subject 域使用 <H2> 风格标签：

```
@If(Form = "Main Topic"; "[<H2>]" + Subject + "[</H2>]"; Subject);
```

3.9.2 在视图中显示 Java 小应用程序和图像

如果你使用表单显示一个视图，可以在表单中包含图形，小应用程序和其他项目。但是在视图特定的行中显示它们应该怎么办呢？为了显示这些项目你只需要写一段将返回 HTML 标签的列公式。例如，下面的列公式根据文档的状态返回一个 标签或一个空字符串：

```
@If(Status = "Complete"; "[<img src = \" /Graphics/ Complete.gif\">]"; "");
```

如果文档的状态是 Complete, Domino 显示 Complete.gif (存储在服务器文件系统中的图形), 否则就什么也不显示。

你可以使用一个类似的方法显示一个 Java 小应用程序。如果小应用程序有参数, 作为公式的一部分你甚至可以计算这些参数的值。例如, 假设你有一个名为 ScrollingText.class 的小应用程序, 这个小应用程序包含一个字符串作为参数:

```
<applet codebase="/applets" code="ScrollingText.class" width=500
height=25>
<param name="Text" value="Important Announcement">
Important Announcement
</applet>
```

现在假设你想使用滚动文本显示重要的信息, 代替文档中的单词 Important Announcement。你可以编写一个列公式, 只为某个文档显示小应用程序, 例如:

```
showjava := @If(Type="Important Announcement");
applet := "[<applet codebase=\" /applets\" \" +
"code=\"ScrollingText.class\" width=500 height=25>\" +
"<param name=\"Text\" value=\"\" + Subject + \"\>]";
endapplet := "[</applet>]";
@If(showjava; applet + Subject + endapplet; Subject);
```

我不推荐因为这个目的而使用 Java 小应用程序, 因为用户必须等到小应用程序下载以后才能从视图中得到信息。如果用户的浏览器不支持 Java 或用户未启动 Java, 则他们根本不能得到重要信息。Java 更常常使用在引用程序中, 以便通过用户接口提供某些值, 而不是仅仅为了得到一个更漂亮但需要更多的时间去下载的 Web 页。

3.9.3 HTML 通用文本

Navigator3.04™	Navigator4.05™	Domino4.6.1™
Explorer3.0™	Explorer4.01™	Domino5.0™

图3-6显示了一个 ViewIcons.nsf 例子中的 Announcements 视图。这个绚丽的视图通过下面的方式使用 HTML 通用文本为紧急声明设置颜色:

- 黄色背景。
- 红色字体。
- 字体加粗。
- 闪烁画面。

另外, 在文档链接中不是使用文本 (比如一个标题), 而是使用一个打开文件夹图像。

当然, 你几乎不会在一个视图中使用如此多的技术, 我把这些技术使用于一个例子中只是为了更为方便的讲解。

Announcements 视图使用表单显示, 并将其命名为 \$\$ViewTemplate for Announcements, 这个表单包含下列元素:

- 标准的视图操作按钮: Previous、Next、Expand、Collapse 和 Search。

- 打开和关闭HTML<TABLE>的标签。
- 在<TABLE>和</TABLE>标签之间，包含一个名为\$\$ViewBody的域。

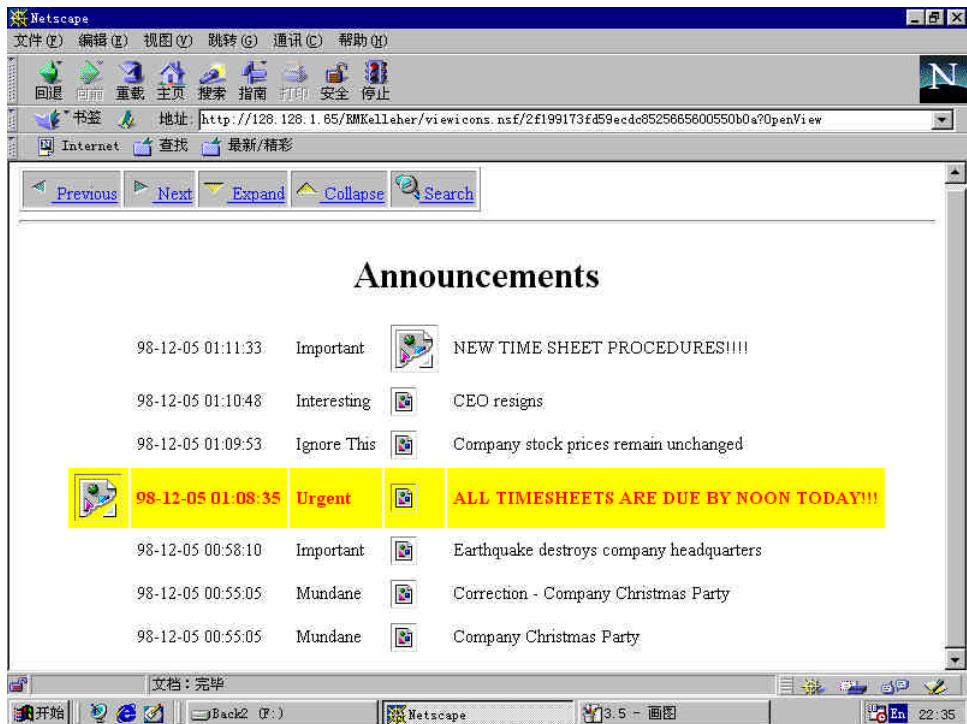


图3-6 视图使用HTML通用文本对紧急申明设置颜色

表单的名称告诉 Domino，当用户在 Web 浏览器中打开 Announcements 视图的时候，使用这个表单去显示它（见本章后面的在表单中嵌入视图）。

Announcement 视图被设置为“把视图中内容看做 HTML”，这意味着 Domino 不做任何转换，它只是简单地把你放在视图中的文本传递出去。作为一个开发者，这意味着你的控制能力增强了，相应地可以更多地决定视图。当启动这项属性的时候，Domino 不再产生表格或线条去分割开各个行，也不产生指向文档的 <A HREF> 链接，你必须自己完成这些工作，通常这都要依赖于列公式，Announcements 视图包含的列公式如表 3-6

表3-6 Announcements 视图中的列

列 号	公 式	描 述
1	@Modified	这个隐藏列被用来根据最上面的输入进行排序。注意尽管这个视图被标志为“看做行 HTML”，隐藏一个列仍然会使 Domino 不在浏览器显示该列
2	bgcolor := @If(Priority = "Urgent"; "yellow"; "white"); "<tr bgcolor=\""+ bgcolor + "\">" + @NewLine	这个列对 HTML 表格进行初始化。如果消息紧急，行的背景色显示为黄色

(续)

列 号	公 式	描 述
3	<pre>"<td>" + @If(Priority="Urgent"; ""; "")+"</td>"</pre>	如果消息紧急, 这个列使用HTML 标签显示一个闪烁的画面。这个图像文件被作为数据库中文档的附件进行保存。在这个列中<TD>和</TD>标签被用来表明表格单元的开始和结束
4	<pre>fontcolor := @If(Priority = "Urgent"; "Red"; "black"); bold := @If(Priority + "Urgent"; ""; ""); endbold := @If(Priority="Urgent"; ""; ""); "<td>" + bold + @Text(@Modified) + "" + endbold + "</td>" + @NewLine</pre>	这个列显示最后一次改变的时间和日期。如果消息紧急, 文本为红色加粗, 否则为黑色正常体
5	<pre>fontcolor := @If(Priority = "Urgent"; "red"; "black"); bold := @If(Priority + "Urgent"; " "; ""); endbold := @If(Priority="Urgent"; ""; ""); "<td>" + bold + Priority + "" + endbold + "</td>" + @NewLine</pre>	这个列显示了消息的优先权, 消息紧急, 文本为红色加粗否则为黑色正常体
6	<pre>unid := @Text(@DocumentUniqueID); db := @ReplaceSubstring(@Subset(@DbName;-1;"\";"/"); url := "/" + db + "/Announcements/" + unid + "?OpenDocument"; "<td>" + "" + "</td>" + @NewLine</pre>	这个列显示了一个经过<A HREF>标签的“打开文件夹”图像链接
7	<pre>fontcolor := @If(Priority = "Urgent"; "red"; "black"); bold := @If(Priority = "Urgent"; ""; ""); endbold := @If(Priority="Urgent"; ""; ""); "<td>" + bold + Subject + "" + endbold + "</td>" + @NewLine</pre>	这个列显示了标题, 如果消息紧急, 文本为红色加粗, 否则为黑色正常体
8	<pre>"</tr>" + @NewLine</pre>	这个列结束HTML表格的一行。@Newline 函数使Domino在HTML中添加换行符而不是作为单个列输出整个视图

在视图中使用HTML通用文本, 你可以对Domino的视图的外观进行大量的控制, 如果你是一个HTML或Notes公式的新手, 则这个过程相对于创建一个Notes视图, 并由Domino确定它的显示方式而非你自己确定其显示方式要复杂的多。但是, 从另外一方面看, 非Domino的Web开发者需要编写一个脚本或Java servlet连接到数据库中, 进行一次查询, 以HTML形式输出一个结果, 不同的记录很有可能指向另外一个脚本或小服务程序以便得到或输出细节。与此相比, 在此创建一个视图可能还要容易一些。

3.9.4 改变视图连接的动作

Navigator3.04™

Navigator4.05™

Domino4.6.1™

Explorer3.0™

Explorer4.01™

Domino5.0™

图3-7显示了在MyDatabase.nsf中的数据库视图。这个数据库包含三个文档，对于不同的数据库来说每个文档包含不同的信息：

Hostname：如果你愿意的话你也可以输入 IP 地址。

DBFilePath：相对与服务器数据目录的数据库的地址和文件名。例如对于我的邮件数据库我输入了mail\rekellehe.nsf。

Title：数据库名称。

Description：关于数据库内容的简要描述。

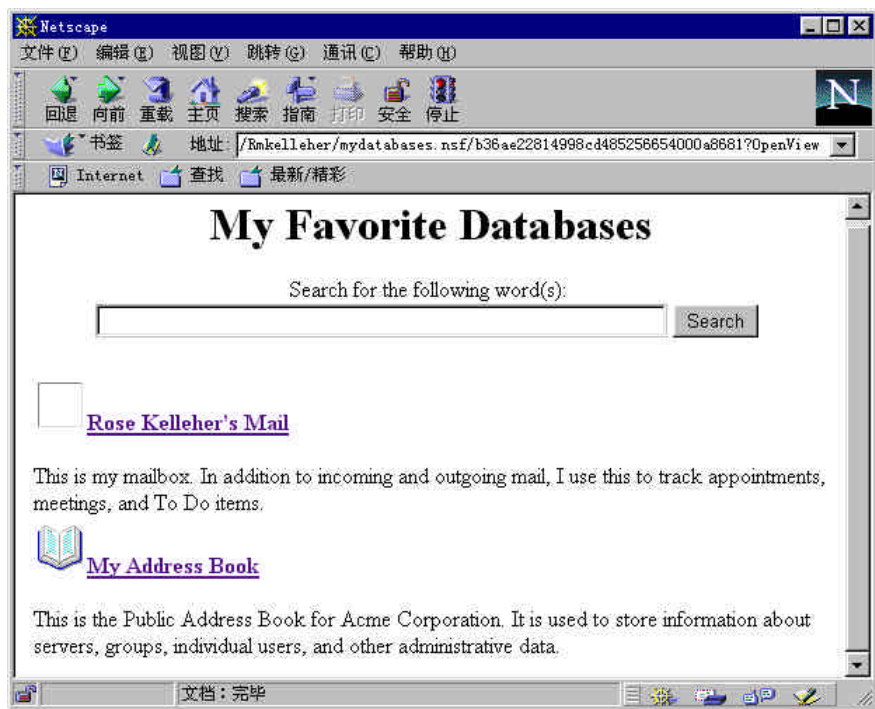


图3-7 这个视图的默认连接已经被使用 HTML 通用文本覆盖，在数据库标题上
点击不打开相应的文档而是打开特定的数据库

如果单击 My Address Book 的链接标签，你将打开 Names.nsf 数据库而不是打开相应的文档。但是如果单击图标时将打开文档。这是因为我已经使用 HTML 通用文本创建了自己的链接。

这个数据库视图只有一个列，它的列公式使用 `<A HREF>` 标签连接数据库名称到一个 OpenDatabase URL。让我们看一下这个怪异的公式：

```
"[<img src=\"http://" + Hostname + "/" + DBFilePath +
"/$icon?OpenIcon\" border=0 height=32 width=32 alt=\"\" +
Title + "\" align=left>\" +
```

公式的第一部分创建了一个 `` 标签，在此我们显示了相应的数据库的图标，例如：

```
<img src="http://1.2.3.4/names.nsf/$icon?OpenIcon" border=0
```



```
height=32 width=32 alt="My Address Book" align=left>
```

这部分公式不包括任何 <A HREF> 标签。因此，当你单击图标的时候，激活的链接是由 Domino 产生的默认链接，它打开视图中相应的文档。

下面是公式的第二部分：

```
"<a href=\"http://\" + Hostname + "/" + DBFilePath +
"?OpenDatabase&Login\">" +
"<h4>" + Title + "</h4></a>]" + Description
```

第二部分创建了一个到数据库的 <A HREF> 链接，例如：

```
<a href="http://1.2.3.4/names.nsf?OpenDatabase"><h4>My Address Book</h4></a>
```

在你的服务器上对这个例子进行测试之前，你需要更改这些文档以便它们指向的是你能够访问的数据库。

可以使用类似的方法改变一个视图连接的动作。例如可以：

- 创建一个视图，在视图中每个文档代表一个代理，单击代理名将运行这个代理。
- 创建一个视图，在视图中每个文档代表一个查询，单击查询名将执行这个查询。
- 创建一个视图，视图的每个 <A HREF> 标签的 TARGET 属性被设置为 "_new"，使每个文档都可以打开一个分离的窗口。

只要你拥有一点关于 HTML 标签的知识和一点想像力，可能性是无限的。

3.9.5 控制视图连接的另外一种方式

- 使用视图列属性，“把列中的值显示为链接”
- 为了创建一个不包含连接的视图，创建一个列，设置列的属性为“把列中的值显示为链接”，并把这个列设置为隐藏。（另外，在连接列的开始添加标签 []，就像下面的一个例子）
- 使用定制的连接代替默认的连接，你可以使用列公式终止它，例如：

```
"[</a><a href=\"my_custom_url\">" + Subject + "</a>]"
```

为了看到关于这个技术的例子，在 MyDatabase.nsf 中打开 "Databases2"。在这个视图中的每一个链接指向同一个 URL (<http://www.erols.com/kelleher>)。

3.9.6 例子：在视图使用 JavaScript

Navigator3.04™	Navigator4.05™	Domino4.6.1™
Explorer3.0™	Explorer4.01™	Domino5.0™

在第5章中包含了几个关于在视图使用 JavaScript 的例子，因此在此我只讲述一个简单的例子。

在 SimpleJSView.nsf 数据库中的 Colors 视图包含几个文档。每个文档包含一个文本域存储颜色的名字：Blue，Yellow，Green 等等。这个视图只有一列，其使用如下的列公式：

```
"[<input type=\"button\" value=\"\" + Color + "\" onClick=\""
```

```
showColor(\' + Color + "\');\>"]
```

例如，如果Color域的值是Blue，这个公式返回如下字符串：

```
[<input type="button" value="Blue" onClick="showColor('Blue');">]
```

这个HTML代码创建了一个按钮，当它被单击的时候执行 JavaScript showColor函数。但是函数在哪儿呢？它在 PickAColor表单中，该表单被用来显示 Color视图。图 3-8显示了当 Netscape Navigator使用OpenForm URL打开PickAColor表单的界面。

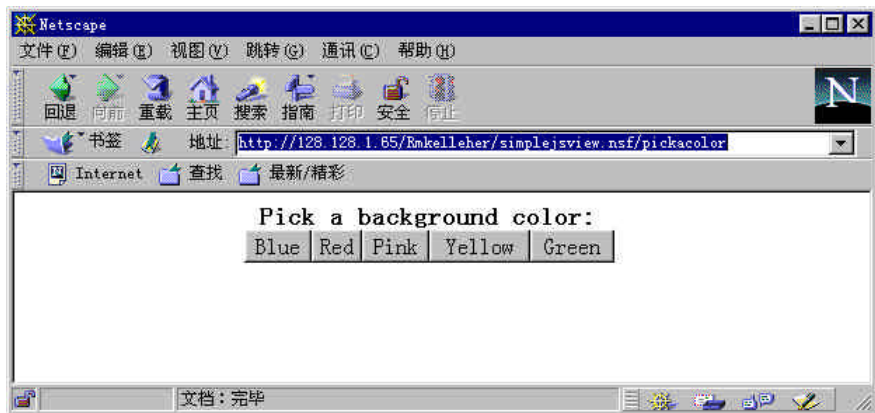


图3-8 PickColor表单通过点击相应的按钮设置背景色

PickAColor表单的设计如图 3-9。ShowColor函数在表单的顶部被设置为 HTML通用文本格式，嵌入Color视图使用了较古老的方式：使用 Colors的默认值插入一个 \$\$ViewBody域。

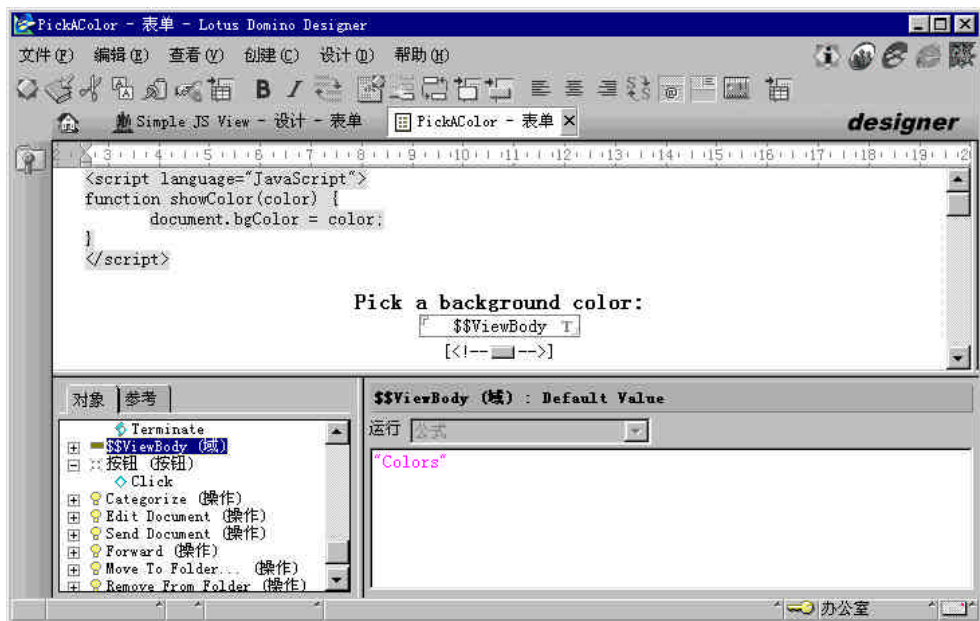


图3-9 指向视图列公式的JavaScript 函数被插入表单的设计中

注意这些按钮是并排排列而不是每行一个。这是因为视图的属性启动了“把视图内容看做HTML”。这个属性导致视图的列公式在编译时不会得到服务器的帮助。正常情况下，Domino服务器把视图内容转换为HTML，适当地添加行控制符（在表单中或表格的行中）。当“把视图内容看做HTML”属性启动后，Domino把上述功能禁用从而让开发者自己完全控制视图的显示。

3.10 在表单中使用视图

有几种方式可创建并使用表单显示视图：

- 创建一个包含内嵌的视图的表单。
- 创建一个表单并使它总是显示特定的视图。
- 创建一个表单用来作为显示所有视图的默认表单。

3.10.1 在表单中嵌入视图

最为直接的把视图和表单联系在一起的方法是简单地创建一个表单并在其中插入一个内嵌式视图元素。当你使用 OpenForm URL在Web浏览器中显示表单，嵌入的视图与其他表单中的元素，如 Submit按钮同样被显示在浏览器中。要想得到一个例子，打开 ViewIcons.nsf表单中的Embed表单使用一个类似于下面的URL

<http://server/RMKelleher/ViewIcons.nsf/Embed?OpenForm>

结果是Web页由View Icon视图和一个表单提交按钮组成。

从R5开始，嵌入的视图元素包含一个你可以选择的属性，可以把视图显示为 applet表单而不是HTML，在嵌入的表单中这个属性禁用。

另外一个在表单中嵌入视图的方法是创建一个名为 \$\$ViewBody的域，它的默认值是视图的名字。在早期的Domino版本中，这是在表单中嵌入视图的惟一方法。这种方法的缺点是当你在Notes客户端打开表单的时候视图不能显示出来。对我们来说，这不是一个问题，因为这本书的焦点在于Web浏览器应用程序而非Notes客户端应用程序或混合应用。

3.10.2 创建视图模板

在表单中显示视图的另外一种方法是使用一个视图模板，你可以为某个独立的视图创建一个模板或为所有的视图创建一个默认的模式。

1. 为一个独立的视图创建一个视图模板

在原来的例子中，只有当用户打开表单的时候视图在表单中显示。在此我们需要一个用户随时可以使用默认的Domino格式打开一个视图而不必先打开一个表单。你可以要求Domino总是使用一个表单显示特定的视图，只需要把表单命名为一个特定的名称：

```
$$ViewTemplate forviewname
```

其中viewname是视图的名字或别名。例如，一个被用来显示“ All By Date ”视图的表单

被命名为 \$\$ViewTemplate for All By Date。

在表单中，既可以插入一个嵌入式视图元素，也可以插入一个名为 \$\$ViewBody 的域，并在域中显示你想显示的视图内容。因此，每次当用户在浏览器中打开 All By Date 视图的时候，它将被嵌入到 \$\$ViewTemplate for All By Date 表单中。

2. 创建一个默认的视图模板

假设你的 Domino 数据库包含 20 个不同的视图，并且希望对所有的视图提供统一的风格和外观。可以创建一个 \$\$ViewTemplate 表单并做 19 份拷贝，为每个视图创建一个表单，但是这将花费很多的时间，另外，每次对某个表单的升级将不得不执行 20 次。因此，你应该采取另外一种方式，只创建一个默认显示所有数据库中视图的表单，并将其命名为 \$\$ViewTemplate Default。

在表单中运行下面两个动作之一：

- 插入一个嵌入式视图元素，在公式中使用空字符串 (" ") 代表视图名称。
- 插入一个名为 \$\$ViewBody 的域名但是不指定一个特定的值。

当一个用户在数据库中打开一个视图的时候，Domino 首先检查是否视图包含它自己的 \$\$ViewTemplate 表单。如果是，则使用这个表单显示视图。否则，使用 \$\$ViewTemplate Default 表单显示视图。

3.10.3 例子：使用表单在视图添加功能

Navigator3.04™	Navigator4.05™	Domino4.6.1™
Explorer3.0™	Explorer4.01™	Domino5.0™

在这个例子中，搜索表单和视图显示在一起，允许用户在不离开页面的情况下快速搜索视图。图 3-10 显示了 MyDatabases.nsf 数据库中的数据库视图，在前面这个视图被用来作为一个说明如何使用 HTML 通用文本对视图链接重定向的例子。在视图的顶部，一个小的搜索表单允许你输入一个单词或短语在视图中进行搜索。

这个例子包含下列组件：

- 一个 \$\$ViewTemplateDefault 表单。
- 数据库视图。

\$\$ViewTemplateDefault 表单有双重作用，它的名字，\$\$ViewTemplateDefault，使得当用户使用浏览器打开视图的时候使用这个表单来显示。它的别名，\$\$SearchTemplateDefault，告诉 Domino 每次要显示搜索结果的时候也使用这个表单。

在表单的顶部，一个名为 QUERY_STRING 的域捕捉 URL 的最后一部分以便显示表单。QUERY_STRING 是一个 CGI 环境变量，它保存了 URL 问号后面的部分。

在这个域下面是标题和一些 HTML 通用文本：

```
<center>
<FORM METHOD="POST" ACTION="/RMKelleher/MyDatabases.nsf/Databases?
SearchView">
Search for the following word(s):<br>
```

```
<input name="Query" size = 50 VALUE="">
<input type="submit" value = "Search">
</FORM>
<a href="/[Db]/Databases?OpenView">View All Databases</a>
</center>
```

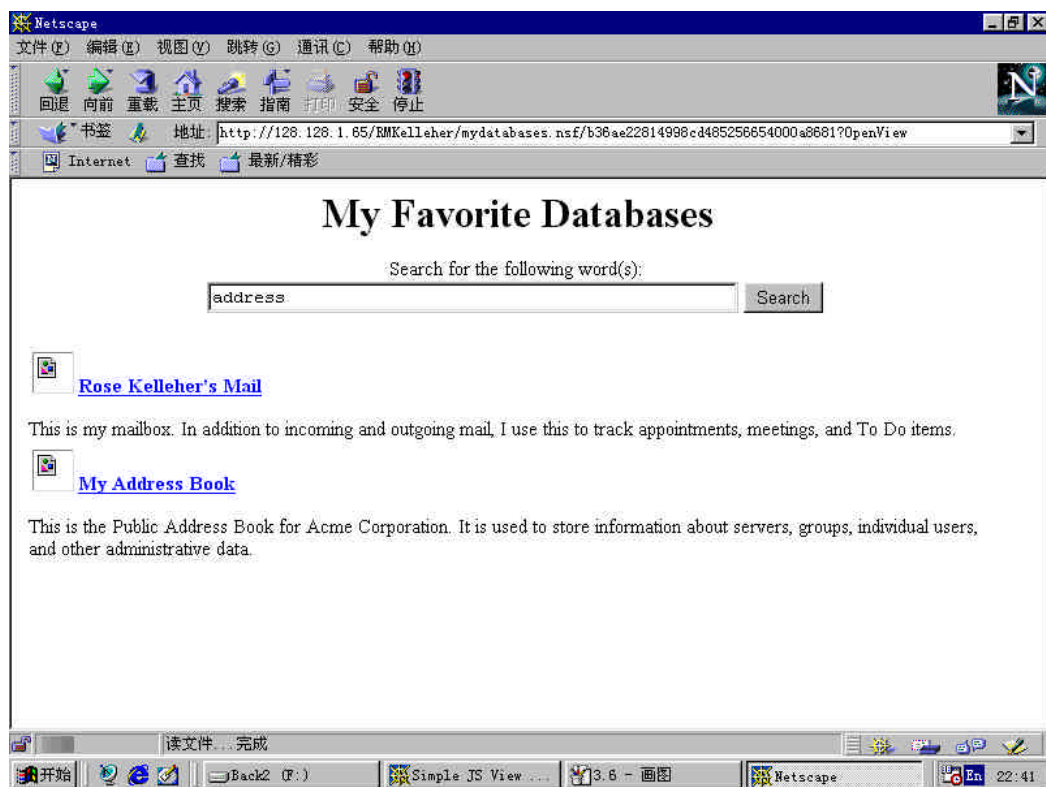


图3-10 视图顶部的搜索表单使你不必打开其他链接就可以搜索

HTML标签创建了一个小的搜索表单它模仿由 Domino产生的默认搜索表单。当你填写了Query域并提交表单的时候，Domino进行全文搜索，得到包含你输入的单词或短语的文档。

在表单下面是链接标签View All Database，这个标签打开数据库视图。在搜索数据库视图的之后，用户可以简单地单击这个链接重新显示视图。由于同样的表单被用来显示视图和搜索结果。当视图被打开的时候使用下面的Hide When公式覆盖链接。

```
@Contains(QUERY_STRING; "OpenView")
```

View All Database连接从搜索结果画面带回原来的视图画面。这类似与 Notes客户端的搜索条上的Reset按钮（在对某个视图进行搜索之后，你单击Reset按钮返回显示视图中的所有文档）。

图3-11是当用户在Query域中输入单词“address”并单击Search按钮的时候显示的画面。

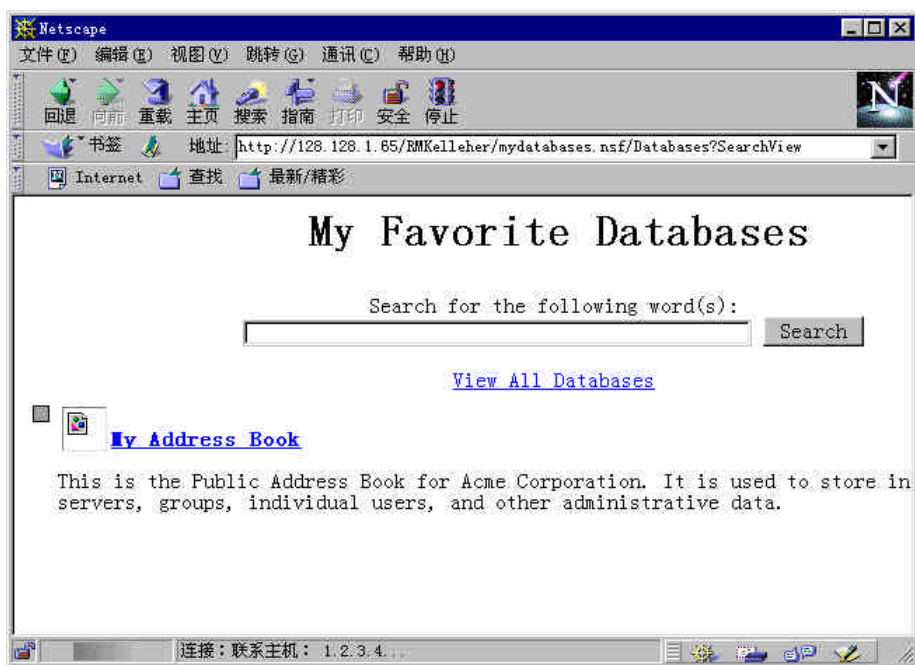


图3-11 View All Database连接类似于Notes客户端的搜索条中的Reset按钮

3.10.4 例子：使用视图在表单中添加功能

Navigator3.04™ Navigator4.05™ Domino4.6.1™
 Explorer3.0™ Explorer4.01™ Domino5.0™

在这个例子中，我们将创建一个表单，在表单中用户可以输入文档的 ID 以便可以从数据库中删除它。为了收集这些 ID，我们将使用一个嵌入式视图在表单提交时提供捕捉到的域值。

图3-12显示了Bulletins.nsf数据库中的Bulletins表单。你可以通过单击在Delete行中的复选框选择要删除的文档。当你提交表单的时候，特定的文档被删除，Bulletins表单被重新显示，此时被删除的文档将不被显示在视图中。

Bulletin视图使用HTML通用文本创建复选框，图3-13显示了Bulletins视图的设计图。复选框按钮包含下列公式：

```
unid := @Text(@DocumentUniqueID);
" [<input type=\"checkbox\" name=\"Delete\" value=\"\" + unid + "\"> ]"
```

这样，在复选框列的每一行包含类似与如下的值：

```
<input type="checkbox" name="Delete" value="8A9909467C1923DB852566  
D0005F8632">
```

为了当表单被提交的时候Domino处理复选框，Bulletins表单必须有一个和复选框同名的域：Delete。因此这个隐藏的，允许多值的Delete域在Bulletins表单的底部。如果在表单中没有设计这样一个域的话，Domino在收到提交表单的时候会产生一个错误信息。因此，我们在此设计了

一个这样的域，当你选择了复选框并提交表单的时候，多值的Delete域就会包含选定文档的ID。

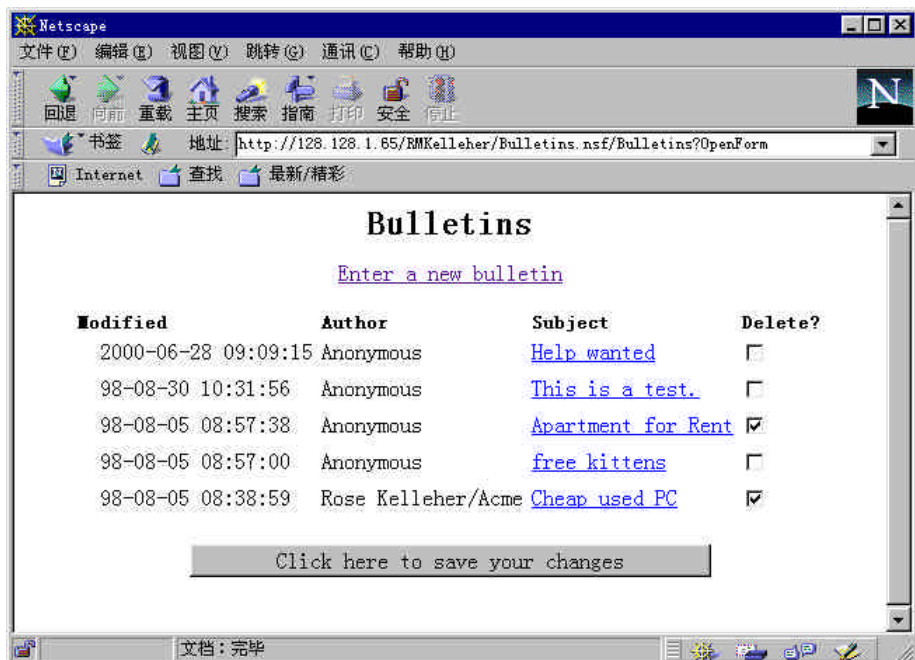


图3-12 这个表单使用一个内嵌视图提供域值，在这个例子中这些值是数据库中将要删除的文档的ID

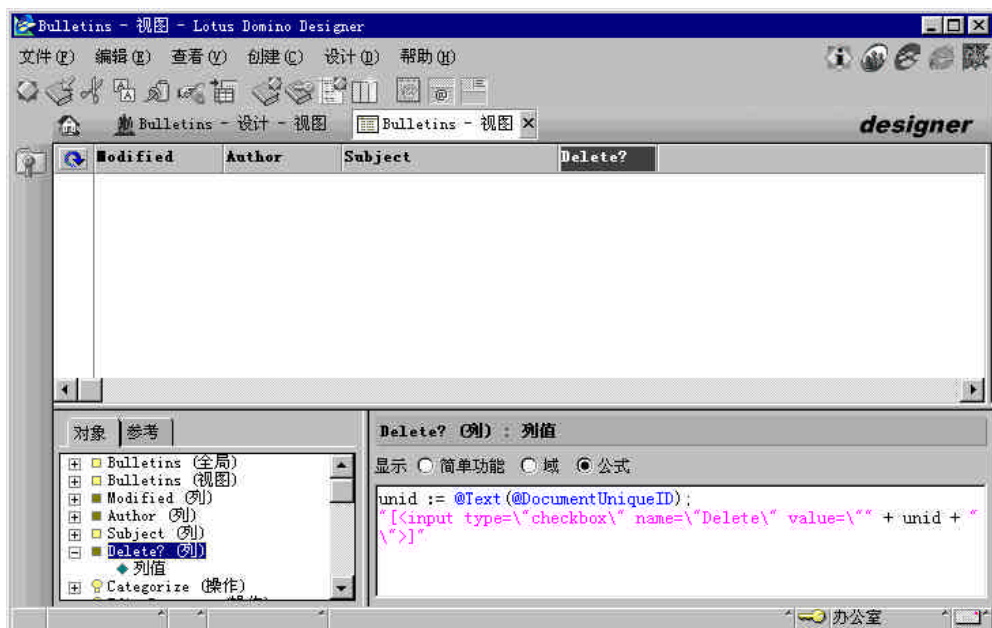


图3-13 Bulletins 视图使用HTML通用文本为每一个文档创建一个删除复选框

Bulletins表单还包含另外两个隐藏域。HTTP_REFERER是可以捕捉原来的URL的CGI变量。当表单提交时WebQuerySave代理将使用这个值处理删除过程。SaveOptions域表明是否提交的文档被保存在硬盘中。在这个例子中，这个表单仅仅被用来捕捉要删除的文档的ID，因此域中的值被设为0，意味着被提交的文档将不会被保存。

在这个例子中使用的另外一个HTML元素是在头部的取消缓冲程序。下面的HTML公式保证页不存在缓冲。（否则将导致在删除某些文档后用户看到的仍然是原来的页面版本）：

```
"<meta http-equiv=\"Pragma\" content=\"no-cache\">"
```

最后也是最重要的一个组件是WebQuerySave代理，这是一个当表单被提交时的代理。下面是DeleteDocuments代理的LotusScript代码：

```
Sub Initialize
    On Error Goto ErrorHandler

    Dim session As New NotesSession
    Dim db As NotesDatabase
    Dim doc As NotesDocument
    Dim item As NotesItem
    Dim docToDelete As NotesDocument

    ' Delete the specified documents
    Set db = session.CurrentDatabase
    Set doc = session.DocumentContext
    Set item = doc.GetFirstItem("Delete")
    Forall unid In item.Values
        Set docToDelete = db.GetDocumentByUNID(unid)
        Call docToDelete.Remove(True)
    End Forall

    ' Now redirect the browser to the previous URL
    Print "[" & doc.HTTP_REFERER(0) & "]"
    Exit Sub

ErrorHandler:
    Print "<h1>Whoops!</h1>"
    Print "Error " & Str(Err) & ": " & Error$
    Resume Next
End Sub
```

代理在Delete域的值列表中进行循环。每个值是一个文档的独特的ID，使用NotesDatabase.GetDocumentByUnid()方法得到它的值，调用它的Remove()方法进行删除。一旦所有的特定文档已经被删除，代理使用Print语句把浏览器指回原来的URL，换句话说，使用?OpenForm URL打开Bulletins表单。当表单被重新加载的时候，视图内容被重新转换为HTML，由代理造成的删除这时被反应了出来。你可以从第6章和第8章得到关于WebQuerySave代理的更多的内容，第6章还包含很多关于LotusScript的例子。

在这个代理中对错误处理并非非常复杂。如果发生了某些错误，代理简单地显示一些 HTML 文本显示错误代码和错误描述，然后指出出错的地方。显示错误信息导致浏览器的重定向不能工作。要对此进行测试，试着把语句：

```
Set docToDelete = db.GetDocumentByUNID(unid)
```

改为：

```
Set docToDelete = db.GetDocumentByUNID("12345")
```

把不存在的文档 ID 传送到 GetDocumentByUNID() 方法产生了一个“无效的通用 ID”错，假设用户选择要删除两个文档的话，结果类似于下面：

```
Whoops!
Error 4091: Invalid universal id
Whoops!
Error 91: Object variable not set
Whoops!
Error 4091: Invalid universal id
Whoops!
Error 91: Object variable not set
[http://1.2.3.4/RMKelleher/Bulletins.nsf/c1a220947d23ad10852566d000
5f81f9?OpenForm]
```

一方面，浏览器不能重定向的优点是可以提醒用户发生了错误，但是另外一个方面，两个方括号之间的 OpenForm URL 对用户来说不代表任何事情，即使知道一些关于“通用 ID”的初步知识也没用。如果你能够预测到将发生某些错误的话，对错误的处理将能够更好一点，例如：

```
ErrorHandler:
Print "<h1>Whoops!</h1>"
Print "Error " & Str(Err) & ": " & Error$
If (Err = 4091) Then
Print "<br>The document you tried to delete does not exist in
the database."
Print "<br>The document ID is: " & unid
End If
Resume Next
```

3.10.5 例子：创建多个类的视图

Navigator 3.04™	Navigator 4.05™	Domino 4.6.1™
EXplorer 3.0™	Explorer 4.01™	Domino 5.0™

在这个例子中，实际的视图从来没有被显示出来，而是使用表单显示从视图使用 @DbColumn 和 @DbLookup 函数得到的信息。这个例子与单个类的视图例子的区别在于它允许用户一次选择超过一个类别。它为了选择类别和显示文档链接使用了一个表单，由于这个技术依靠于函数 @DbColumn 和 @DbLookup，因此它的返回数据限制在 64K 以内。你只能在相对较小的文档数量的时候使用它。

图3-14显示了 在SmartView.nsf数据库中的SelectCategory表单，当你第一次打开这个表单的时候，它显示了一个类别的列表。这个“列表”实际上是一个名为 Category的关键字域，它使用了如下的关键字公式：

```
@DbColumn("Notes":"NoCache"; "" ; "All"; 1)
```

这个公式返回了All视图的第一列的值的列表。在这个例子中，第一列包含了类别，你可以在选择的同时通过按下 Shift或Ctrl键而一次选择多个类别。为了保证提交表单的时候最少选择了一个类别，Category域使用了下面的默认公式来返回 All视图的第一列的第一个值：

```
@Subset(@DbColumn("Notes":"NoCache"; "" ; "All"; 1); 1)
```

当你提交表单的时候，\$\$Return域把浏览器指回 SelectCategory表单，但这次在URL中包含了一个参数：

```
db := @ReplaceSubstring(@Subset(@DbName; -1); "\\\" ; "/" );  
cat := @ReplaceSubstring(@Implode(Category; "" ); " " ; "+");  
url := "/" + db + "/SelectCategory?OpenForm&Category=" + cat;  
 "[" + url + "]"
```

这个公式返回一个包含Category的URL，这个参数可能包含多个值，各个值之间使用分隔符(~)分开。例如：

```
/db.nsf/SelectCategory?OpenForm&Category=Black+and+White~Brown
```

当使用URL打开表单的时候，Category参数被QUERY_STRING域捕获。(QUERY_STRING是一个CGI变量，它包含在URL问号后面的查询信息)

当表单被重新打开的时候，QUERY_STRING域的值看起来像下面一样：

```
OpenForm&Category=Black+and+White~Brown
```

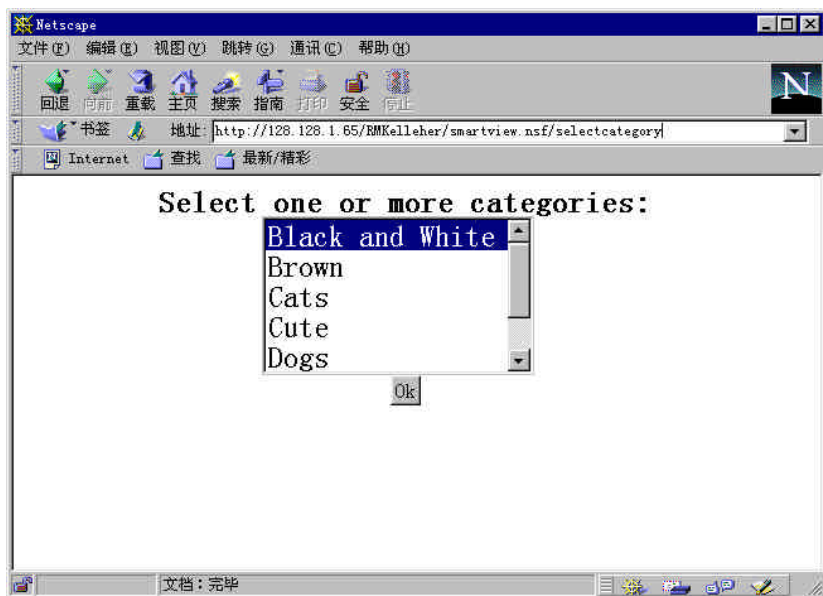


图3-14 SelectCategory表单在你没有选择一个类的时候会提示你选择一个类

QUERY_STRING域在表单中的几个Hide When公式中使用, 因此根据用户是否选择一个类显示的信息不同。如果用户没有选择一个类, QUERY_STRING不包含字符串 "&Category=", 因此信息如图3-14所示。如果用户选择了一个类, QUERY_STRING包含字符串 "&Category=" 因此显示信息如图3-15。

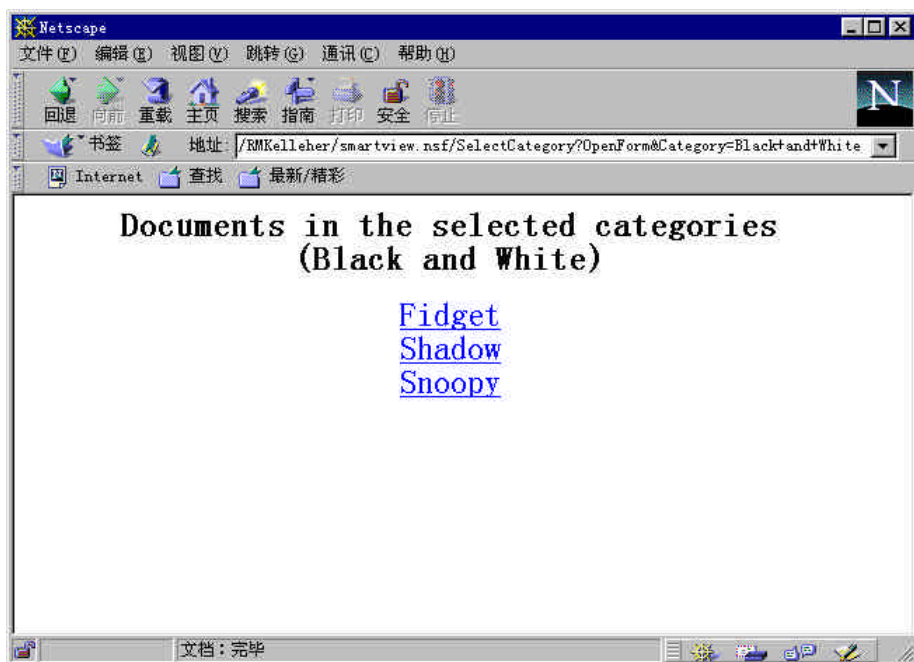


图3-15 如果在URL中指定一个类, SelectCategory表单
显示到这个类中的所有文档的链接

圆括号中的文本事实上是一个名为 SelectedCategory 的计算域, 它包含下面的公式值:

```
cat := @Right(QUERY_STRING; "&Category=");
@Explode(@ReplaceSubstring(cat; "+"; " "); "~")
```

第一行从 OpenDocument URL 中的 Category 参数中提取值, 在这个例子中, 值为 Black+and+White~Brown。第二行使用空格代替加号, 把原来用分割符 (~) 连接的字符串分割为一个数组或列表 (仅仅是由于 URL 中不允许空格, 因此你使用了加号, 你还可以使用 %20, 这是 URL 内部空格键的代码)。

文档的链接是其他计算域公式的结果:

```
list := @Unique(@DbLookup("Notes"; "NoCache"; ""; "Links";
SelectedCategory; 3));
@if(@IsError(list); ""; list)
```

这个公式在 Links 视图中查找选定的类, 并在第三列为每个属于这些类的文档返回一个值, 在 Links 视图的第三列包含 HTML 通用文本, 该文本使用下列公式:

```
db := @ReplaceSubstring(@Subset(@DbName; -1); "\\\"; "/");
```

```
unid := @Text(@DocumentUniqueID);  
url := "/" + db + "/All/" + unid + "?OpenDocument";  
"[<a href=\"\" + url + "\">\" + Subject + "</a>"]"
```

这个公式为每个文档返回一个 <A HREF>链接，在 URL 中使用文档的独特的 ID 作为关键值，例如：

```
<a href="/RMKelleher/SmartView.nsf/All/  
825E296A3CFBC6F68525665B00829594?OpenDocument">Shadow</a>
```

这个例子的优点是它全部由 Notes 公式组成而不依赖于 LotusScript 或者 Java 代理。缺点是，它受 @DbColumn 和 @DbLookup 函数可以返回的文本的限制（根据 Notes 版本的不同其限制不同，在 4.6.1 版本中限制为 64 千字节）和文本域可以容纳的数据的限制。可以使用可计算的 RTF 域而不是文本域避开 15 千字节的限制，但是还是无法解决 64 千字节的问题。这个例子的另外一个缺点是从多个类导出的链接导致一系列的链接以不正确的字母表序列排序。

3.11 在小应用程序表单中显示一个视图

Navigator3.04™ Navigator4.05™ Domino4.6.1™

Explorer3.0™ Explorer4.01™ Domino5.0™

在视图添加这种时髦的功能的早期方法是把它嵌入到表单中，然后在嵌入的视图对象中启动“显示小应用程序”属性。图 3-16 显示了根据 R5 讨论数据库模板建立的数据库的 By Category 视图。

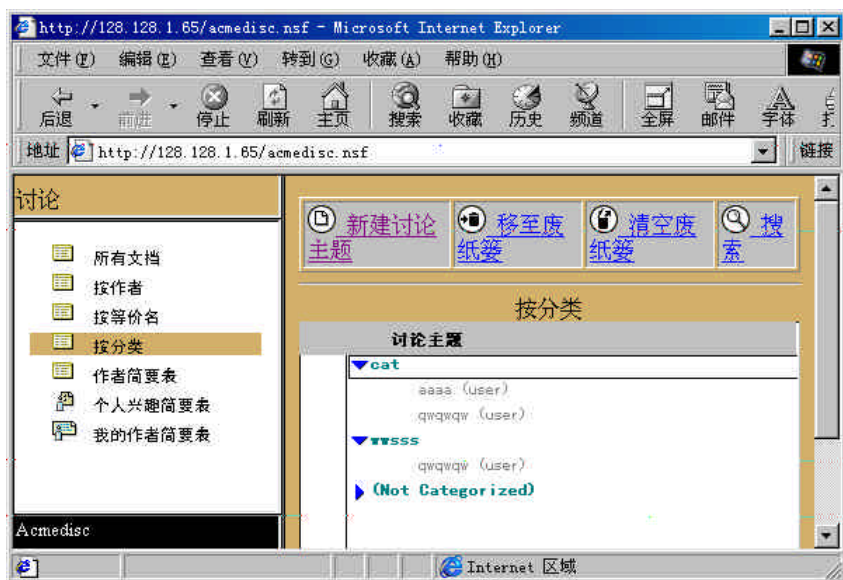


图3-16 Domino小应用程序提供Notes客户端视图的Web视图

视图小应用程序，一个新的 R5 功能已经被提供给你。它从视图读到数据，与此同时读到关于视图设计的信息，并使视图更像在 Notes 客户端的显示。用户可以重新规划列的大小，改

变排序顺序，选择多个文档，滚动屏幕等等。缺点是用户必须等到小应用程序下载之后，才能看到这些小应用程序。为了在 Netscape中更新视图，你必须按下 Shift+Reload去重新初始化这些applet。当然你的浏览器必须启动 Java以便小应用程序可以工作。

除了提供给用户精美的界面以外，例如可以重新设定列的大小，视图小应用程序还在 HTML视图上提供其他功能。与 HTML页面不同，小应用程序可以接受键盘事件，这使用户可以使用 Del删除选定文档，使用 F9刷新视图。另外，你可以使用操作按钮或热点操纵视图，如表3-7。

表3-7 操作视图的热点按钮

公 式	描 述
@Command([ViewCollapse])	折叠选定的文档
@Command([ViewExpand])	展开选定的文档
@Command([ViewCollapseAll])	折叠所有文档
@Command([ViewExpandAll])	展开所有文档
@Command([EmptyTrash])	把标记为删除的文档删除掉

注意这个列表框随时可能增加。为了实验视图小应用程序的一些操作热点，打开 SmartView.nsf数据库中的 ViewApplet表单。如果你把视图嵌入到 HTML格式，Previous和Next热点被转换为 OpenForm URL。Start参数指定在视图的顶部显示那些文档。Expand和Collapse热点被转换为下面的 JavaScript URL:

```
Javascript:document.view.expandAll();  
Javascript:document.view.collapseAll();
```

这些URL使用Netscape的“LiveConnect”技术在JavaScript中调用小应用程序方法。文档对象代表当前Web页，视图对象代表小应用程序，expandAll()和collapseAll()代表小应用程序方法。

参考信息

- 在Iris Today中提供了一篇极好的文章“Combining forms and views for friendlier Web applications”，作者Mark Gordon 和Andrew Broyles，需要请访问<http://www.notes.net>。
- 在10月11日发行的Lotes Notes & Domino Advisor描述了大量使用Web的视图的有用的技术，还包括了一个非常聪明的办法去创建一个灵活的 Expand，Collapse，Previous和Next按钮，作者Keith Reichley，网址<http://www.advisor.com>。

本章小结

视图不但在为用户提供信息上是非常有用的，而且可以从应用程序中获得信息。许多视图在Notes数据库中已经可以通过 Web访问。默认情况下，Domino在视图的顶部和底部产生 Expand、Collapse、Previous、Next和Search链接。视图中的文档被表示为到文档 URL的 HTML的链接。Domino可以把日历视图转换为 HTML表格，而且视图把组件格式为 HTML标签（例如标签）。

视图列公式在创建一个复杂的视图，在每一列中显示超过一个域的值的时候是非常有用的。表单公式允许你确定在视图中显示给定的文档使用哪个表单，因此在决定怎样显示内容的时候它非常有用。

视图列图标为用户提供了一个可视化线索，最普通的应用是显示附件。你也可以使用列图标显示文档的状态或类别，编辑它的文字处理器的类型，确定是否对文档的访问进行限制等等。

视图的操作按钮对一些简单的操作，如打开一个表单或显示另外一个视图是非常有用的。

当为每个视图中的文档产生一个 HTML 的时候在视图中的 HTML 通用文本显得非常有用。应用包括有条件地改变单个行的格式化信息，根据文档的状态或其他准则，显示图像或其他 Java 小应用程序，改变在视图中的连接的动作。在行的基础上使用 JavaScript。典型的是在视图列公式中添加 HTML 通用文本。在可以使用 JavaScript 的视图中，你可以添加一些普通的 JavaScript 函数。为此，可以在表单中嵌入视图，和在视图的上面或下面添加一些函数作为 HTML 通用文本。

在表单中添加一个视图允许把表单的功能添加到视图中，增强视图的外观和功能。你可以通过将表单命名为 `$$ViewTemplate for <view-name>` 从而为某个视图指定一个默认的表单。你还可以数据库中所有的视图指定一个默认的表单通过将其命名为 `$$ViewTemplate Default`。使用表单为视图添加功能的方法是在表单顶部显示一个搜索表单。使用视图升级表单的一种方法是使用视图为关键字域提供选项。

你可以使用 `@DbColumn` 和 `@DbLookup` 函数创建一个模拟的视图显示一个属于选定类的文档的链接。

你可以把一个视图显示为一个 Java 小应用程序通过在一个表单中嵌入它并启动它的小应用程序显示属性。这个小应用程序提供了一个类似与 Notes 客户端的界面视图。