IBM SPSS Modeler 17 CLEF 开发人员指南

注释

在使用本资料及其支持的产品之前，请阅读第 321 页的『声明』中的信息。

产品信息

本版本适用于 IBM(r) SPSS(r) Modeler V17.0.0 及所有后续发行版和修订版，直到在新版本中另有声明为止。

目录

前言

IBM® SPSS® Modeler 是 IBM Corp. 企业级数据挖掘工作平台。SPSS Modeler 通过深度的数据分析帮助组织改进与客户和市民的关系。组织通过借助源自 SPSS Modeler 的洞察力可以留住优质客户，识别交叉销售机遇，吸引新客户，检测欺诈，降低风险，促进政府服务交付。

SPSS Modeler 的可视化界面让用户可以应用他们自己的业务专长，这将生成更加强有力的预测模型，缩减实现解决方案所需时间。SPSS Modeler 提供了多种建模技术，例如预测、分类、分割和关联检测算法。模型创建成功后，通过 IBM SPSS Modeler Solution Publisher，在广泛的企业内交付给决策者，或通过数据库交付。

关于 IBM Business Analytics

IBM Business Analytics 软件提供了完整、一致且正确的信息，决策人可以放心地根据此信息提高业务绩效。企业智能、预测分析、财务业绩和战略管理的完整产品组合，和分析应用程序一起提供对当前业绩的清晰、直接和实用的洞察力，以及预测未来结果的能力。结合丰富的行业解决方案，久经证明的实践和专业服务，各种规模的组织都能够实现最高生产力、确信地自动作出决策以及获得更好的结果。

作为此产品服务组合的组成部分，IBM SPSS Predictive Analytics 软件可以帮助组织预测未来事件，并在此洞察的基础上提前行动以实现更好的业务结果。全世界的商业、政府和学校客户都依靠 IBM SPSS 技术作为吸引、保留和增加客户的竞争优势，同时减少欺诈并降低风险。通过在其日常运营中融入 IBM SPSS 软件，组织将成为前瞻型企业 - 能够指引并实现决策自动化，以满足业务目标并实现可衡量的竞争优势。有关详细信息或要联系一位代表，请访问 http://www.ibm.com/spss。

技术支持

技术支持可供维护客户使用。客户可就 IBM Corp. 产品使用问题或某一受支持硬件环境的安装帮助寻求技术支持。要与技术支持联系，请访问 IBM Corp. Web 站点 (http://www.ibm.com/support)。寻求帮助时，请准备好标识您的身份、所在组织以及支持协议。

第 1 章 概述

CLEF 简介

组件级扩展框架 (CLEF) 是一种允许向 IBM SPSS Modeler 的标准功能添加用户提供的扩展的机制。扩展通常包含可添加到 IBM SPSS Modeler 中的共享库（例如数据处理例程或建模算法），并且可通过某个菜单上的新条目或节点选用板上的新节点访问该库。

要执行该操作，IBM SPSS Modeler 需要有关该定制程序的详细信息，例如其名称、应传递给该程序的命令参数以及 IBM SPSS Modeler 如何向程序显示选项和如何向用户显示结果等。要提供此信息，您应当提供 XML 格式的文件，即 **规范文件**。 IBM SPSS Modeler 会将该文件中的信息转换为新菜单条目或节点定义。

使用 CLEF 的好处包括:

- 提供简单易用、异常灵活且稳定的环境，可供工程师、顾问和最终用户将新功能集成到 IBM SPSS Modeler 中。
- 确保扩展模块的外观和功能与本地 IBM SPSS Modeler 模块的相同。
- 使扩展节点具有与本地 IBM SPSS Modeler 节点尽可能接近的执行速度和效率。

系统体系结构

与 IBM SPSS Modeler 自身相似，CLEF 使用两层的客户端/服务器体系结构，在该结构中，这两层可位于同一计算机中，也可位于两台不同的计算机上。

客户端组件

此处显示了客户端层的组件。

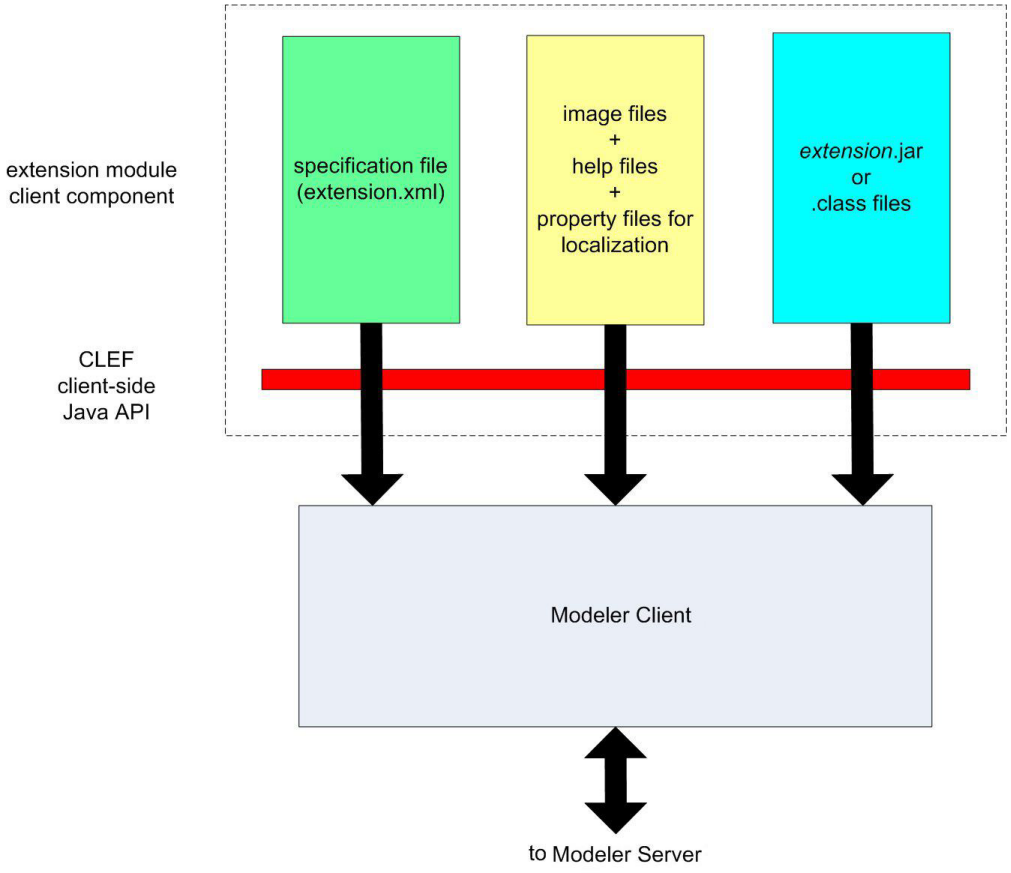

图 1. 客户端组件

- **规范文件**。列出了扩展定义的属性、格式、数据模型更改、控件和其他特征。
- **图像文件**。包含用于识别扩展中的节点的图像。
- **帮助文件**。用于显示有关扩展的帮助信息。
- **属性文件**。包含的文本字符串包括扩展在屏幕上显示的名称、标签和消息。
- **Java .jar 或 .class 文件**。包含了扩展使用的所有 Java 资源。
- **Java 应用程序编程接口 (API)**。可供需要未由规范文件直接提供的其他控件、用户界面组件或交互的扩展使用。

服务器端组件

此处显示了服务器层的组件。

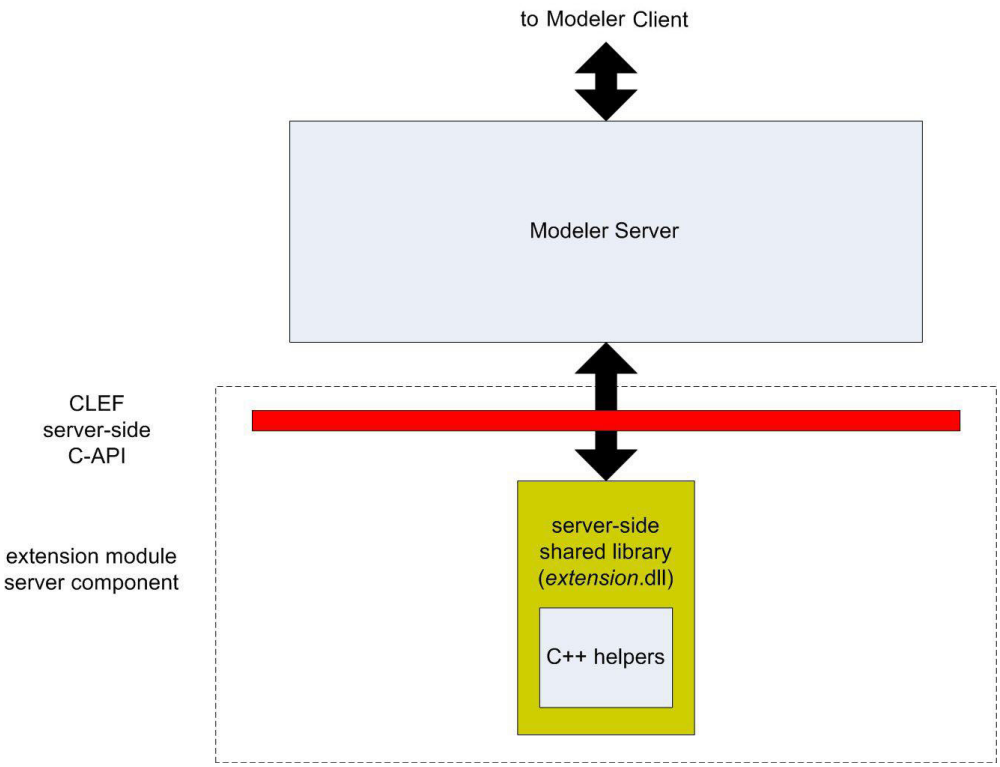

图 2. 服务器端组件

- **共享库的基于 C 的 API**。包含设置和获取执行设置、这些设置的持久性、执行反馈、作业控制（例如，中断执行）、SQL 生成和返回的对象等方面。
- **服务器端共享库**。一个支持节点执行的动态链接库 (DLL)。C++ 工具是一些基于 C 的 API 的包装器，这些基于 C 的 API 作为源代码提供，可以很容易地编译为 C++ CLEF 模块。

CLEF 的功能

下面的章节介绍了 CLEF 的许多主要功能:

- 规范文件
- 节点
- 数据模型
- 输入和输出文件
- 应用程序编程接口 (API)

规范文件

CLEF 规范文件是一个 XML 文件，它包含说明新扩展行为的结构规范。规范文件说明的内容包括:

- 此扩展所需的共享资源（例如，本地化文本束以及服务器端共享库）。
- 公共定义，例如文件类型或属性类型。
- 最终用户可以使用的新对象，例如节点和输出模型。

启动 IBM SPSS Modeler 后，将从规范文件所在的位置导入该文件，这样可以立即使用该文件中定义的功能。

有关更多信息，请参阅第 29 页的第 4 章，『规范文件』。

节点

将扩展添加到实现新节点的 IBM SPSS Modeler 中时，需要先确定要创建的节点类型（例如，节点是用于生成模型还是只用于转换数据）。有关更多信息，请参阅第 9 页的『节点概述』主题。

创建规范文件和所有必需的 Java 类和共享库后，将文件复制到 IBM SPSS Modeler 可读取这些文件的特定位置。下次启动 IBM SPSS Modeler 时，新节点将添加到相应的选用板中，并随时可用。

数据模型

数据模型表示流过 IBM SPSS Modeler 流的数据的结构。该模型描述流中该点处的数据，并与"类型"节点中显示的信息相对应。它列出了流中特定点处现有字段的名称，并说明了这些字段的类型。

将任何节点添加到 IBM SPSS Modeler 时，请考虑传输到此节点的数据模型对节点行为的影响。例如，一个衍生节点先获取输入数据模型，再为模型添加新字段，随后生成输出数据模型并将其传递到 IBM SPSS Modeler 流中的下一个节点中。与此相比，图形节点先获取输入数据模型，但不生成任何输出数据模型，因为该数据不会被传递到任何后续节点。 IBM SPSS Modeler 必须知道要对数据模型执行的操作，这样后续节点可以显示有关哪些字段可用的正确信息。规范文件中的数据模型信息是 IBM SPSS Modeler 用于在整个流中保持数据模型一致性所必需的信息。

根据数据是流入、流出还是流经节点，规范文件必须描述输入和/或输出的数据模型。CLEF 节点可为传递到节点中的任何字段添加新字段，或使用程序自身生成的新字段替代传输到节点中的字段，从而影响到数据模型。规范文件中的 `OutputDataModel` 元素说明了 CLEF 节点对数据模型的影响。有关更多信息，请参阅第 55 页的『输出数据模型』主题。

输入和输出文件

在运行 CLEF 节点之前，可以指定一个或多个要生成的临时文件。这些文件是 **输入文件**，因为它们被输入到服务器的节点执行中。例如，模型构建器节点中可能存在模型容器，执行该节点时，该容器中的内容将传输到指定的输入文件。有关更多信息，请参阅第 52 页的『输入文件』主题。

其他临时文件是在服务器上执行节点期间生成的；例如，执行模型构建器或文档构建器节点的结果。这些文件是**输出文件**，在执行节点后，它们将被传输回客户端。有关更多信息，请参阅第 53 页的『输出文件』主题。

应用程序编程接口 (API)

根据您期望此扩展执行的操作不同，您可能需要使用应用程序编程接口 (API)。对于简单的数据变换，可能能够在规范文件中完整地定义必要的处理。但是，如果有更高级的需求，将需要具有一个或多个 API 的接口:

- CLEF 客户端 API
- CLEF 服务器端 API
- Predictive Server API (PSAPI)

CLEF **客户端 API** 是 Java API，可供需要其他控件、用户界面组件或交互（未由规范文件直接提供）的扩展使用。

CLEF **服务器端 API** 是基于 C 的 API，涵盖了多个方面，例如设置和获取执行设置、这些设置的持久性、执行反馈、作业控制（例如中断执行）、SQL 生成和返回的对象。

Predictive Server API 是 Java API，显示了需要数据细化和预测分析功能的应用程序可以使用的 IBM SPSS Modeler 功能。

有关更多信息，请参阅第 161 页的第 9 章，『程序设计』。

文件结构

CLEF 扩展包含两组组件:

- 客户端组件
- 服务器端组件

客户端组件包含扩展规范文件、Java 类和 .jar 文件以及包含可本地化资源、图像和帮助文件的属性捆绑包。

服务器端组件是执行扩展节点时所需的共享库和 DLL。

客户端组件

客户端组件安装在 IBM SPSS Modeler 安装目录的 \ext\lib 文件夹下。服务器端组件包括:

- 规范文件
- Java 类和 .jar 文件
- 属性文件
- 图像文件
- 帮助文件

扩展文件夹

每个扩展都直接位于 \ext\lib 下自身的**扩展文件夹**中。

对扩展文件夹的建议命名规则为:

providerTag.id

其中，*providerTag* 是规范文件中 ExtensionDetails 元素的供应商标识，*id* 指相同元素的扩展标识。

例如，如果 ExtensionDetails 元素的开头为:

```
<ExtensionDetails providerTag="myco" id="sorter" ... />
```

则使用扩展文件夹名 myco.sorter。

规范文件

规范文件自身的名称必须为 extension.xml，且必须位于扩展子文件夹的根目录下。在前面提供的示例中，指向规范文件的路径必须在 IBM SPSS Modeler 安装目录下，如下所示:

```
\ext\lib\myco.sorter\extension.xml
```

Java 类和 .jar 文件

在使用了客户端 Java API 的扩展中，包含经过编译的 Java 代码。该代码可保留为 .class 文件集，也可以打包为 .jar 文件。

Java .class 文件位于相对于扩展文件夹的最上层。例如，实现 ActionHandler 接口的类的路径可能为:

```
com.my_example.my_extension.MyActionHandler
```

在该示例中，.class 文件应当在 IBM SPSS Modeler 安装目录的如下位置:

```
\extension_folder\com\my_example\my_extension\MyActionHandler.class
```

.jar 文件可位于扩展文件夹下的任何位置。可以通过规范文件中的 JarFile 元素指定 .jar 文件的实际位置。例如，如果扩展使用具有如下路径的 .jar 文件:

```
\extension_folder\lib\common-utilities.jar
```

规范文件应当在 Resources 元素中包含如下条目:

```
<Resources>
    <JarFile id="util" path="lib\common-utilities.jar"/>
     ...
</Resources>
```

有关更多信息，请参阅第 33 页的『Jar 文件』主题。

属性文件

本地化后的资源（例如，屏幕文本和错误消息及其外语翻译）可保存为扩展名为 .properties 的文件，该文件可位于扩展文件夹下的任意位置。有关更多信息，请参阅第 154 页的『属性文件』主题。

图像和帮助文件

包含用于显示图标的图形图像的文件以及包含帮助系统的文件可以位于扩展文件夹下的任意位置。如果将图像文件和帮助文件分别放在其各自的子文件夹下，将会非常方便。

可以通过规范文件中 Icon 元素的 imagePath 属性声明图像文件的位置。有关更多信息，请参阅第 98 页的『图标』主题。

与该方法相似，可以使用规范文件中 HelpInfo 元素的 path 属性声明帮助系统的位置。有关更多信息，请参阅第 149 页的『定义帮助系统的位置和类型』主题。

示例

基于这些组件的客户端文件可能具有以下结构:

```
\ext\lib\myco.sorter
\ext\lib\myco.sorter\extension.xml
\ext\lib\myco.sorter\sorter_en.properties
\ext\lib\myco.sorter\sorter_fr.properties
\ext\lib\myco.sorter\sorter_it.properties
\ext\lib\myco.sorter\com\my_example\my_extension\MyActionHandler.class
\ext\lib\myco.sorter\help\sorter.chm
\ext\lib\myco.sorter\images\lg_sorter.gif
\ext\lib\myco.sorter\images\sm_sorter.gif
\ext\lib\myco.sorter\lib\common-utilities.jar
```

服务器端组件

执行时所需的共享库必须位于 IBM SPSS Modeler 安装目录中 \ext\bin 文件夹下的某个文件夹中，例如:

```
installation_directory\ext\bin\myco.sorter\my_lib.dll
```

请注意，共享库不得直接位于 \ext\bin 文件夹中。

对于 IBM SPSS Modeler 在执行过程中直接调用的共享库，必须在规范文件的 SharedLibrary 元素中声明其位置。有关更多信息，请参阅第 33 页的『共享库』主题。

主共享库可能要使用其他库。也可以将所有相关共享库与主共享库放置在同一位置，以方便主共享库查找相关共享库。

示例

下面是服务器端文件结构的示例:

```
\ext\bin\myco.sorter\my_lib.dll
\ext\bin\myco.sorter\my_lib2.dll
```

第 2 章 节点

节点概述

创建实现新节点的扩展时，需要熟悉 IBM SPSS Modeler 节点的特征。这样做可帮助您在规范文件中正确定义这些节点。

IBM SPSS Modeler 节点根据它们的功能可以分为源节点、流程节点、输出节点和建模节点。在 CLEF 中，节点的分类方法略有不同。两个系统之间的映射如下表所示。

表 1. CLEF 节点类型.

IBM SPSS Modeler 分类	选用板	CLEF 节点类型
源节点	源	数据阅读器
过程节点	记录选项	数据变换器
	字段选项	
输出节点	图形	文档构建器
	输出（报告节点）	
	导出(E)	数据记录器
建模节点	建模	模型构建器

创建新的 CLEF 节点时，需要将它定义为 CLEF 节点类型之一。所选的节点类型取决于该节点的主要功能。

表 2. 节点类型和功能.

CLEF 节点类型	描述	对应的节点选用板	图标形状
数据阅读器	将不同格式的数据导入 IBM SPSS Modeler。	源	 *图 3. 源节点形状（圆形）*
数据变换器	从 IBM SPSS Modeler 获取数据，并以某种方式修改数据，然后将修改后的数据返回至该 IBM SPSS Modeler 流。	记录选项，字段选项	 *图 4. 选项节点形状（六边形）*
模型构建器	使用 IBM SPSS Modeler 中的数据生成模型。	建模	 *图 5. 模型构建器节点形状*

表 2. 节点类型和功能 (续).

CLEF 节点类型	描述	对应的节点选用板	图标形状
文档构建器	从 IBM SPSS Modeler 中的数据生成图形或报告。	图形	 *图 6. 图形节点形状（三角形）*
		输出（报告节点）	 *图 7. 输出节点形状（矩形）*
模型应用器（也称为“模型块”）	为恢复到 IBM SPSS Modeler 工作区的生成模型定义容器。	-	 *图 8. 模型应用器节点形状（金色菱形）*
数据记录器	将 IBM SPSS Modeler 格式的数据导出为适合于其他应用程序使用的格式。	导出(E)	 *图 9. 导出节点形状（矩形）*

在规范文件的 Node 元素中定义节点类型和其他属性，例如:

```
<Node name="sort_process" type="dataTransformer"
      palette="recordOp" ... >
     -- node elements --
</Node>
```

palette 属性定义用户可以从中访问节点的 IBM SPSS Modeler 主窗口中的选用板，此示例中定义的是“记录选项”选用板。如果省略此属性，节点会显示在“字段选项”选用板中。

IBM SPSS Modeler 提供了大量示例节点。有关更多信息，请参阅第 23 页的『关于示例』主题。

数据阅读器节点

通过数据阅读器节点，可以将外部源的数据读取到 IBM SPSS Modeler 流中。IBM SPSS Modeler“源”选用板上的节点与数据阅读器节点等效，并且用圆形图标形状标识。

在数据阅读器节点的规范中，可以包含下列内容的详细信息:

- 数据源（如文件或数据库）
- 对记录进行的所有预处理（如处理开头和结尾空格或用作记录定界符的字符）
- 是否过滤出所有记录字段
- 与每个字段关联的数据类型（如范围、集合或标志）和存储类型（如字符串、整数或实数）

- 输入数据模型是否发生了更改

数据阅读器节点可以包含用于读取源数据记录的逻辑。另外，这也可以通过 IBM SPSS Modeler 中的“类型”节点在下游完成。

IBM SPSS Modeler 提供了一个示例数据阅读器节点。有关更多信息，请参阅第 23 页的『关于示例』主题。

数据变换器节点

数据变换器节点从 IBM SPSS Modeler 流获取数据，并以某种方式修改数据，然后将修改后的数据返回至该流。IBM SPSS Modeler“记录选项”和“字段选项”选用板上的节点都是数据变换器节点，并且用六边形图标形状标识。

在数据变换器节点的规范中，可以包含下列内容的详细信息:

- 变换哪些记录或字段
- 修改数据的方式

IBM SPSS Modeler 提供了一个示例数据变换器节点。有关更多信息，请参阅第 23 页的『关于示例』主题。

模型构建器节点

有关在 IBM SPSS Modeler 中构建模型的概述，请参阅《*IBM SPSS Modeler 17 应用程序指南*》中的“建模简介”。

模型构建器节点生成的对象会显示在 IBM SPSS Modeler 主窗口中管理器窗格的“模型”或“输出”选项卡上。

IBM SPSS Modeler“建模”选用板上的节点是模型构建器节点的示例，并且用五边形图标形状标识。

执行后，模型构建器节点会在“模型”选项卡上生成**模型输出对象**（也称为“模型块”）。

将生成的模型添加到工作区时，它将采取模型应用器节点的形式。

在模型构建器节点的规范中，可以包含下列内容:

- 模型构建详细信息，如用于生成模型的算法以及哪些输入和输出字段可用于对模型的数据进行评分
- 模型使用的属性
- 用于容纳输出对象的容器
- 节点对话框的用户界面
- 执行节点时使用的属性和文件
- 执行节点对输入数据模型的影响
- 执行节点时生成的模型输出对象以及任何其他对象的标识
- 模型应用器节点的标识（请参阅 第 12 页的『模型应用器节点』）

注意: 定义模型构建器节点时，需要在同一规范文件中的其他位置定义实际模型输出对象和模型应用器节点。

IBM SPSS Modeler 提供了一个示例模型构建器节点。有关更多信息，请参阅第 23 页的『关于示例』主题。

文档构建器节点

文档构建器节点生成的对象会显示在 IBM SPSS Modeler 主窗口中管理器窗格的“输出”选项卡上。“图形”选用板上的节点是文档构建器节点的示例，并且用三角形图标形状标识。

执行后，文档构建器节点会在管理器窗格的“输出”选项卡上生成**文档输出对象**。

与模型输出对象相反，不能将文档输出对象添加回 IBM SPSS Modeler 工作区。

在文档构建器节点的规范中，可以包含下列内容:

- 文档构建详细信息，如要包含文档生成控件的节点对话框选项卡
- 文档使用的属性
- 用于容纳输出对象的容器
- 节点对话框的用户界面
- 执行节点时使用的属性和文件
- 执行节点时生成的文档输出对象和任何其他对象的标识

注意: 定义文档构建器节点时，需要在同一规范文件中的其他位置定义实际文档输出对象。

模型应用器节点

模型应用器节点定义用于容纳生成模型的容器，将生成的模型从管理器窗格的“模型”选项卡添加到 IBM SPSS Modeler 工作区时使用该容器。

在模型应用器节点的规范中，可以包含下列内容的详细信息:

- 容纳模型的容器（如果可以如文本和 HTML 等多种格式生成模型输出，那么为多个容器）
- 用户浏览“模型”选项卡上的应用器节点或在工作区打开应用器节点时显示的对话框的用户界面详细信息
- 输出数据模型
- 执行包含节点的流时所执行的处理
- 用于处理执行包含节点的流时所生成的对象的构造函数

数据记录器节点

数据记录器节点将 IBM SPSS Modeler 格式的数据导出为适合于其他应用程序使用的格式。IBM SPSS Modeler“导出”选用板上的节点是数据记录器节点，并且用矩形图标形状标识。

在数据记录器节点的规范中，可以包含下列内容:

- 流数据将被写入的文件或数据库的详细信息
- 是否发布整个流以使它可以嵌入到外部应用程序中（可选）

菜单、工具栏和选用板

用户可以从 IBM SPSS Modeler 菜单、工具栏或选用板访问扩展。扩展可以实现节点或执行指定的操作。

也可从工具栏访问可以从明确指定的菜单进行访问的扩展（节点或操作），反之亦然。

可以从“插入”菜单上的相应项目自动访问可从选用板访问的节点。

菜单和子菜单

用户可以从“插入”菜单访问标准的 IBM SPSS Modeler 节点。此菜单最后一组中的每一项（“模型”除外）均有一个子菜单，用于提供对相关节点集的访问。

这些项直接与节点选用板上的条目相对应。将节点添加到选用板时，也会自动将其添加到“插入”菜单的相应组中。

如果扩展定义的操作无法通过节点访问，那么可以通过添加下列一项或多项使该扩展可用:

- 系统菜单或子菜单的新项
- IBM SPSS Modeler 的新菜单
- 工具栏的新项（请参阅 『工具栏』）

新菜单或菜单项还可以显示与扩展关联的图标，如与“插入”菜单项中的某些项一样。

有关更多信息，请参阅第 99 页的『菜单』 和 第 101 页的『菜单项』。

工具栏

如果扩展定义的操作无法通过节点来访问，那么可以通过将其添加到 IBM SPSS Modeler 的主工具栏来使该扩展成为可用。

在这种情况下，建议隐藏该操作的标签。

您也可以将项目添加到节点对话框或输出窗口的工具栏中。可以选择显示或隐藏项目标签。

有关更多信息，请参阅第 102 页的『工具栏项』主题。

选用板和子选用板

如果扩展定义了新节点，那么可以将该节点放到一个标准的 IBM SPSS Modeler 选用板或子选用板的任意位置。

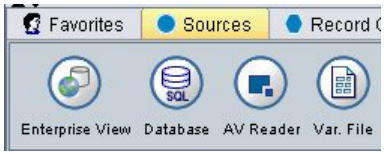

图 10. 标准选用板上的新节点

可以向标准子选用板添加一个条目，并可以从此处访问该节点。

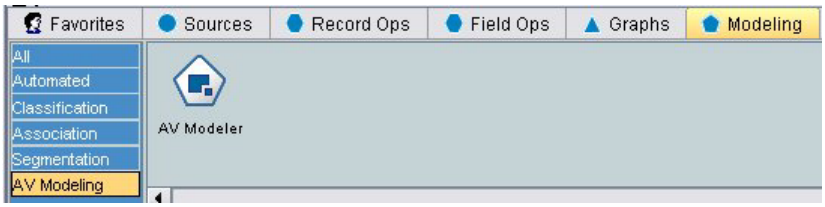

图 11. 标准子选用板的定制添加项上的新节点

可以定义定制选用板，并将新节点放入其中。

图 12. 定制选用板上的新节点

定制选用板可以包含定制子选用板。

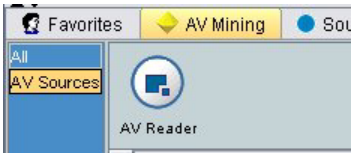

图 13. 定制选用板的定制子选用板上的新节点

有关更多信息，请参阅第 45 页的『节点』 和 第 40 页的『“用户界面（选用板）”部分』。

设计节点图标

对于在 CLEF 中创建的每一个新节点，均可以为在屏幕上标识该节点的图标提供一个中央图像。

注意: 您并非必须提供图像，如果您未指定图像，那么将显示 IBM SPSS Modeler 提供的缺省图像（这在开始开发节点时非常有用）。

图 14. CLEF 图标的缺省图像

标准 IBM SPSS Modeler 图标由以下三层组成:

- 边框
- 背景
- 中央图像

对于新节点，您只需提供中央图像（也称为**轮廓**），IBM SPSS Modeler 将对边框和背景进行处理。轮廓图像必须具有透明背景，以免遮挡图标背景层。在本节中，表示轮廓时采用的是彩色背景以便指示透明度。

这是典型的 IBM SPSS Modeler 建模图标的组成方式。

表 3. 节点图标和已生成模型图标的组成

	节点图标	生成的模型图标
边框	 *图 15. 图标边框*	无
背景	*图 16. 图标背景*	 *图 17. 生成的模型背景*

表 3. 节点图标和已生成模型图标的组成 (续)

	节点图标	生成的模型图标
轮廓	 *图 18. 图标轮廓*	 *图 19. 生成的模型轮廓*
图像显示方式	 *图 20. 显示的图标*	 *图 21. 显示的所生成模型图标*

边框

此节点的功能由图标边框的形状指示。有关更多信息，请参阅第 9 页的『节点概述』主题。

如果对节点启用了缓存，那么将在边框形状上添加文档符号缩图。节点上的白色文档图标指示其缓存为空。当高速缓存已满时，该文档图标将保持绿色不变。

表 4. 节点边框和缓存状态

缓存状态	示例
无缓存	 *图 22. 无缓存的节点*
启用缓存	 *图 23. 已启用缓存的节点*
缓存已满	 *图 24. 缓存已满的节点*

该系统提供了不同的边框符号，并且 IBM SPSS Modeler 会进行必要的处理以在正确的时间显示正确的符号。

背景

除生成的模型节点和模型应用器节点的图标以外，其他节点图标的背景均会更改颜色以指示状态。

表 5. 节点背景

状态	颜色(C)	示例
未选中的	灰色	*图 25. 图标背景（灰色）*
选定	蓝色	 *图 26. 图标背景（蓝色）*
错误	红色	 *图 27. 图标背景（红色）*
数据库中执行的操作	紫色	 *图 28. 图标背景（紫色）*

另外，该系统还提供了背景图像，并且 IBM SPSS Modeler 会进行必要的处理以在所有情况下都能显示正确的背景。

图形要求

对于每个新的 CLEF 节点，均可创建下列版本的轮廓层图像:

- 大尺寸（49 x 49 像素）图像，针对流工作区上的节点
- 小尺寸（38 x 38 像素）图像，针对位于屏幕底部选用板管理器中的节点

如果要在菜单、工具栏、浏览器或输出窗口的标题栏中显示图标，那么还需创建:

- 缩图尺寸（16 x 16 像素）图像

如果节点生成模型，那么还需创建:

- 小尺寸（38 x 38 像素）图像，设计为移到左下角以覆盖生成的模型图标（金色块）

注意: 大于这些尺寸的图像在 IBM SPSS Modeler 中显示时将进行剪裁。

有关更多信息，请参阅第 98 页的『图标』主题。

创建定制图像

为节点创建的图像应传达该节点的主要功能。考虑到国际用户，请注意使用的图像不要特定于某一个国家/地区，并且不会引起其他国家/地区用户的误解。

要创建用于 CLEF 的定制图像:

1. 通过使用支持透明度的图形包将绘图工作区设置为适当的尺寸并绘制图像版本。
2. 将每个版本（大尺寸、小尺寸等）保存为具有下列特征的不同 *.gif* 文件:
   - 透明背景
   - 16 色（4 位）或更高的颜色深度

   将图像背景设为透明的方法取决于您使用的图形包。例如，您可能可以将背景颜色直接设置为透明，或者可能需要先指定一种透明颜色，然后再用此颜色“描画”图像背景。

   对于图像文件，建议遵循 IBM SPSS Modeler 内部使用的文件命名约定，如下表所示。

表 6. 图像文件命名规则

图像类型	文件名
Large	lg_ *node* .gif
小(S)	sm_ *node* .gif
缩图尺寸	*node*16.gif
生成的模型	sm_gm_ *node* .gif

3. 通过在规范文件中引用图像文件（请参阅 『将图像文件添加至节点规范』）并将新节点添加至 IBM SPSS Modeler（请参阅 第 185 页的『测试 CLEF 扩展』）来测试图像的外观。

将图像文件添加至节点规范

创建图像文件后，将它们复制到将要运行 IBM SPSS Modeler 的计算机上的文件夹中。在规范文件中，需要声明图像相对于 IBM SPSS Modeler 安装目录中 \ext\lib\ *provider.nodename* 文件夹的路径，因此应将这些文件部署到易于查找的文件夹。有关更多信息，请参阅第 98 页的『图标』主题。

在规范文件中，可以通过 Node 规范的 UserInterface 部分中的 Icons 元素使大图标及小图标图形文件与定制节点相关联，例如:

```
<Icons>
     <Icon type="standardNode" imagePath="images/lg_mynode.gif" />
     <Icon type="smallNode" imagePath="images/sm_mynode.gif" />
</Icons>
```

对于模型构建器节点或文档构建器节点，还需要在 ModelOutput 规范（对于模型构建器节点）或 DocumentOutput 规范（对于文档构建器节点）的 UserInterface 部分中引用缩图（16 x 16 像素）版本，例如:

```
<Icons>
     <Icon type="standardWindow" imagePath="images/mynode16.gif" />
</Icons>
```

对于模型应用器节点，还需要在 Node 规范的 UserInterface 部分中引用生成的模型版本，例如:

```
<Icons>
     <Icon type="standardNode" imagePath="images/lg_gm_mynode.gif" />
     <Icon type="smallNode" imagePath="images/sm_gm_mynode.gif" />
</Icons>
```

设计对话框

本节介绍标准 IBM SPSS Modeler 节点对话框的特征，以帮助您在 CLEF 中设计一致的对话框。

关于节点对话框

节点对话框提供了使最终用户能够修改执行设置的界面。对话框的外观很重要，因为要在这里更正和修改节点行为。该界面必须包含所有必要的信息，而且要易于使用。

节点行为是通过使用各种基于对话框的 **控件**来进行更改的，这些控件是用户可以进行交互的用户界面元素。对话框可以包含许多控件，如单选按钮、复选框、文本框和菜单。 CLEF 提供了各种各样的控件，您可以将它们用于您的对话框设计。有关更多信息，请参阅第 111 页的『属性控件规范』主题。

由控件修改的参数的类型决定了出现在对话框中的控件，其中一些类型提供了备用控件。可以通过规范文件中的 `Tab` 元素对新选项卡上选项进行分组。有关更多信息，请参阅第 20 页的『选项卡区域』主题。

注意: 即使未指定扩展要执行的处理，也可以测试该扩展的用户界面外观。有关更多信息，请参阅第 185 页的『测试 CLEF 扩展』主题。

对话框的设计原则

定义对话框的控件时，请考虑下列原则:

- 仔细考虑控件具有显示标签时要使用的文本。该文本应简短适度，同时能够传达正确的信息。如果设计面向国际市场，请注意翻译后的文本长度可能会与原始文本长度有显著差异。
- 使用对参数正确的控件。例如，对于仅具有两个值的参数，最佳选择不一定总是复选框。IBM SPSS Modeler C5.0 节点对话框使用单选按钮使用户能够选择具有两个值之一的输出类型 - **决策树**或**规则集**。

  此设置可以用一个标签为**决策树**的复选框来表示。选中时，输出类型为决策树；取消选中时，输出为规则集。虽然结果一样，但在本例中使用单选按钮可使用户更易于理解选项。
- 文件名控件通常位于对话框的顶部。
- 构成节点重要部分的控件位于对话框的上端。例如，图形节点显示数据中的字段。选择这些字段是该对话框的主要功能，因此字段参数位于顶部。
- 复选框或单选按钮通常允许用户选择需要更多信息的选项。例如，在 C5.0 对话框中选择 **使用提高**时，需要在分析中包含一个表示 **尝试次数**的数字。

  进一步信息总是放置在选项选择之后，要么在其右边，要么在其正下方。

CLEF 对话框以标准 IBM SPSS Modeler 对话框的方式使用 IBM SPSS Modeler 的落实编辑: 直到用户单击**确定**、**应用**或**执行**（对于终端节点）之后，才将对话框中显示的值复制到节点。同样，只有用户取消或重新显示对话框或单击 **刷新**按钮后，对话框显示的信息才会被更新（例如，在当前节点的上游进行操作，导致节点的输入字段被更改时）。

对话框组件

对话框可包含下列组件:

- 标题栏
- 图标区域
- 工具栏和菜单区域可包含:

  -“文件”、“生成”、“查看”、“预览”、“刷新”以及其他按钮（取决于节点）

- 最大化/常规尺寸按钮
- 帮助按钮

- 状态区域
- 面板区域
- 选项卡区域
- 按钮区域

每个定制节点都需要一个对话框，该对话框将在用户打开该节点时显示。假设您的规范文件包含一个 Node 元素，且该元素的 UserInterface 部分包含一个 Tabs 元素，当您打开该节点时会看到上述所有的对话框组件。根据节点类型不同，选项卡区域及按钮区域至少包含下表所示的内容。

表 7. 不同类型的节点至少包含的选项卡区域和按钮区域的内容

节点类型	Tabs	按钮
数据阅读器	注解（工具栏区域含有“刷新”按钮）	确定、取消、应用和重置
数据变换器	注解	确定、取消、应用和重置
数据记录器	发布和注解	确定、取消、执行、应用和重置
模型构建器	注解	确定、取消、执行、应用和重置
文档构建器	注解	确定、取消、执行、应用和重置
模型应用器	汇总和注解	确定、取消、应用和重置

节点对话框最初的定位是，当用户打开该节点时，节点图标叠加在它所表示的节点上。用户可以移动该对话框，但下次打开该节点时，对话框并不会显示在这个新位置。如果用户移动了该对话框，并且该对话框随后被另一对话框部分或完全遮住，那么可以通过在工作区中双击原始节点使第一个对话框再次显示在前面。对话框是无模式的（即相同的用户输入始终会产生相同的操作）并可以调整大小。

此对话框中的所有可编辑字段都支持下表所示的键盘快捷键。

表 8. 对话框中的可编辑字段的键盘快捷键

快捷键	效应
Ctrl-C	复制(C)
Ctrl-V	粘贴(P)
Ctrl-X	剪切(T)

标题栏

节点对话框的标题栏包含缩图版的 IBM SPSS Modeler 块图标，后跟模型名称。文本来自模型名称控件的设置。缺省还在右上角提供“关闭”按钮 (X)。

图标区域

节点图标显示在对话框左上角附近的图标区域中。这是还可用于主窗口底部的节点选用板的小尺寸（38 x 38 像素）版本的图标，而不是显示在工作区中的大版本图标。

注意: 标题栏左端的小型块状图标硬编码到所有节点对话框中。

工具栏和菜单区域

对话框的最顶部区域保留为工具栏和菜单区域。

数据阅读器和数据变换器节点对话框在此区域中有一个“预览”按钮，用于显示输入数据的样本。

数据阅读器节点对话框还具有“刷新”按钮，它将更新节点显示的信息（例如，当节点的输入字段发生更改时）。

模型应用器节点具有“文件”、“生成”和“查看”菜单按钮，使用户能够执行各种操作，例如导出模型或生成新节点。模型应用器节点还有“预览”按钮，但在本例中显示输入数据的样本，以及应用节点时所创建的其他列。

此区域的右侧包含每一个节点对话框均有的两个按钮:

- 最大化/常规尺寸按钮
- 帮助按钮

最大化/常规尺寸按钮: 此按钮可以将对话框大小调整为全屏尺寸。再次使用此按钮时可将对话框收缩回最大化之前的尺寸。

帮助按钮: 此按钮用于打开节点的上下文相关帮助。对于选项卡式对话框或输出窗口，将显示该选项卡的帮助。还可以用 F1 键来访问帮助。

状态区域

对话框顶部的余下区域为显示信息、警告或错误文本而保留。源节点在此处显示源数据文件的完整路径和文件名。各个节点可以在此区域显示其他特定于节点的信息。为此区域指定的任何文本均应限制在两行内。

面板区域

这是对话框的主要区域，其中包含节点的所有控件和显示区域。每个选项卡都有不同的面板区域。每一个面板可以是下列类型之一:

- 文本浏览器
- 扩展对象
- 属性

还可以指定子面板，子面板是在新窗口中打开的单独对话框，并通过面板上的操作按钮进行调用。

有关更多信息，请参阅第 106 页的『面板规范』主题。

选项卡区域

节点对话框可以包含下列选项卡:

- 一个或多个用户提供的节点特定的选项卡
- 一个“摘要”选项卡（仅适用于模型输出对象和模型应用器节点）
- 一个“注解”选项卡

节点特定的选项卡定义在 CLEF 规范文件的 `Tabs` 部分中。有关更多信息，请参阅第 103 页的『Tabs』主题。

模型输出对象和模型应用器节点的对话框都包含一个由系统提供的“摘要”选项卡。该选项卡显示有关生成的模型的摘要信息，包括使用的字段、构建设置和模型估算过程。结果以树形视图显示，通过单击指定项可以扩展或合并树形视图。

系统为所有节点对话框提供了“注解”选项卡，用户可以通过该选项卡指定节点的相关信息。该选项卡包含节点名称、工具提示文本和一个较长的注释字段。

名称。缺省节点名称在规范文件的 Node 元素的 Label 属性中指定（请参阅第 45 页的『节点』）。用户可以重命名节点，方法是选择**定制**，在“定制”编辑字段中输入一个名称，然后单击**应用**或**确定**。虽然通过选择 **自动** 可恢复缺省名称，但新名称会跨各个会话保留。在“注解”选项卡上指定的定制名称会替代对话框中任何其他选项卡上指定的定制名称。

工具提示文本。此处指定的文本显示为工作区中节点的工具提示。如果未在此处指定工具提示文本，那么用户将光标悬停在此节点上时，将不会显示工具提示。

关键字。用户可以指定要用于工程报告的关键字，以及搜索或跟踪 IBM SPSS Collaboration and Deployment Services Repository 中存储的对象时使用的关键字。

注释面板。用户可在此区域输入注释文本。

创建和保存信息。这是一个不可编辑的文本区域，其中显示了创建信息和名称以及保存文件的日期/时间（日期/时间格式取决于语言环境）。如果未进行保存，此字段将显示“该项目尚未保存”。

按钮区域

每一个对话框的底部都会显示 **应用**、**重置**、**确定**和 **取消**按钮。如果节点是终端节点（处理流数据的可执行节点），那么还会显示 **执行**按钮。

确定。应用所有设置并关闭对话框。当第一次从节点打开对话框时，该按钮具有焦点（由按钮周围的蓝色矩形表示），按 Enter 键也可以执行“确定”操作。

取消。关闭对话框并保留打开对话框前或者上一次执行“应用”操作之后的设置。当整个对话框具有焦点时，按 Esc 键也可以执行“取消”操作。

执行。应用所有设置，关闭对话框并执行终端节点。

应用。保存对话框的设置，以便下游操作可以使用这些设置。

重置。将整个对话框重置为打开对话框时或上一次执行“应用”操作之后的值。

设计输出窗口

本节介绍标准 IBM SPSS Modeler 输出窗口的特征，以帮助您在 CLEF 中设计一致的输出窗口。

输出窗口显示的输出可以来源于:

- 模型，如对一组数据评分的模型（将模型应用于一组数据）
- 文档，如图形或报告

有关更多信息，请参阅第 95 页的『关于用户界面』主题。

输出窗口与节点对话框相似，但有下列不同之处:

- 标题栏具有节点特定的缩图图标，而不是通用的金色块图标
- 省略了主节点图标
- 在工具栏和菜单区域中，省略了最大化/常规按钮（在文档输出窗口中可由“关闭”和“删除”按钮代替），但仍然可以通过鼠标调整窗口大小
- 省略了状态区域

- 选项卡通常为:
  - 模型输出窗口的“模型”选项卡，用于显示预测变量重要性数据（如果在模型节点上选中此选项）
  - 用于输出的单个选项卡。
  - 模型输出窗口的“摘要”选项卡，用于显示模型的摘要详细信息。
  - “注解”选项卡（注解值来自生成输出的节点）。
- 按钮区域有“确定”、“取消”、“应用”和“重置”按钮

CLEF 提供了与以上插图所示的窗口相似的缺省模型输出窗口和文档输出窗口。在规范文件中使用 `ModelOutput` 或 `DocumentOutput` 元素时，通常会显示这两个窗口。有关更多信息，请参阅第 44 页的『对象标识』主题。

另外，可以通过指定 `ModelOutput` 或 `DocumentOutput` 元素，使用自己设计的定制窗口完全替换缺省的输出窗口。有关更多信息，请参阅第 147 页的『定制输出窗口』主题。

第 3 章 CLEF 示例

关于示例

为帮助您熟悉 CLEF，IBM SPSS Modeler 安装中包含了一组示例节点及其完整的源代码。这些基本节点功能有限，旨在帮助您理解 CLEF 的工作原理及其用法。您可以马上或随时测试这些节点。

示例包括:

- 数据阅读器节点（名为“Apache 日志阅读器”）
- 数据变换器节点（名为“URL 解析器”）
- 文档构建器节点（名为“Web 状态报告”）
- 模型构建器节点（名为“交互”）

在使用这些示例之前，必须先将其激活。

激活示例

这些示例作为 IBM SPSS Modeler 安装的一部分以压缩格式安装到 Demos 目录中。您需要激活这些示例，方法是将这些文件提取到如下所示的正确位置。

在安装 IBM SPSS Modeler 的计算机上:

1. 如果 IBM SPSS Modeler 正在运行，请退出。
2. 在 IBM SPSS Modeler 安装的 Demos 文件夹中找到文件 clef_examples_ext_lib.zip。
3. 将 clef_examples_ext_lib.zip 的内容提取到 IBM SPSS Modeler 安装目录的 \ext\lib 文件夹中。

在安装 IBM SPSS Modeler 和/或 IBM SPSS Modeler Server 的计算机上:

1. 将 clef_examples_ext_bin.zip 的内容提取到 IBM SPSS Modeler 和 IBM SPSS Modeler Server 安装目录的 \ext\bin 文件夹中。
2. 在非 Windows 系统上，使用 clef_examples_ext_bin.zip 中提供的 makefile 来编译感兴趣的示例的源代码。有关更多信息，请参阅第 26 页的『检查源代码』主题。

   或

   在 Windows 上，使用以下指示信息来编译感兴趣的示例的源代码（请注意，需要 Visual Studio 2008）:

   a. 在步骤 1 中解压缩 clef_examples_ext_bin.zip 的 \ext\bin 目录中，转至包含想要编译的示例 CLEF 扩展的子目录（例如，spss.apachelogreader）。
   b. 在 src 子目录中，双击 .sln 文件以在 Visual Studio 中打开 CLEF 扩展解决方案（例如，双击 \ext\bin\spss.apachelogreader\src\apachelogreader.sln）。
   c. 在 Visual Studio 中，转至**构建** > **配置管理器**。
   d. 对于“活动解决方案配置”，请选择**发行版**。
   e. 对于“活动解决方案平台”，请选择 **x64**。
   f. 单击**关闭**。
   g. 要构建项目，请转至**构建** > **构建解决方案**（或者，单击 F7）。

生成的 64 位 DLL 将写入以下位置（相对于 src 文件夹）：

```
..\bin\win64\release\spss.<extension-name>\<extension-name>.dll
```

（例如，`..\bin\win64\release\spss.apachelogreader\apachelogreader.dll`）。

最后，在所有情况下，请启动 IBM SPSS Modeler 并确保下表中显示的节点显示在节点选用板上。

表 9. 节点选用板上显示的节点.

选用板选项卡	节点
源	Apache 日志阅读器
字段选项	URL 解析器
建模	交互
输出	Web 状态报告

数据阅读器节点（Apache 日志阅读器）

数据阅读器节点示例是从 Apache HTTP Web 服务器的访问日志文件中读取数据的源节点。访问日志包含 Web 服务器已处理的所有请求的详细信息。日志记录的格式为综合日志格式，例如:

```
IP_address - - [09/Jul/2007:07:57:38 +0000] "GET /lsearch.php?county_id=3 HTTP/1.1" 200 16348
"http://www.google.co.uk/search?q=thunderbirds+cliveden&hl=en&start=10&sa=N"
"Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322)"
```

您可以使用该示例节点将日志记录转换为易于读取的表格格式。

要使用“Apache 日志阅读器”节点:

1. 如果尚未激活 CLEF 示例，请立即激活。有关更多信息，请参阅第 23 页的『激活示例』主题。
2. 打开 IBM SPSS Modeler。
3. 在节点选用板的“源”选项卡中选择 **Apache 日志阅读器**，并将该节点添加到工作区中。
4. 编辑此节点。在“选项”选项卡的“Apache 日志文件”字段中，输入:

   demos_folder\combined_log_format.txt

   其中 *demos_folder* 是指 *Demos* 文件夹在 IBM SPSS Modeler 安装目录中的位置（不要使用 *$CLEO_DEMOS* 格式）。
5. 单击**确定**。
6. 将“类型”节点添加到流中。
7. 编辑“类型”节点。单击 **读取值**以读取数据，然后单击 **确定**。
8. 将一个“表”节点附加到“类型”节点并执行流。日志文件的内容将以表格格式显示。
9. 保存流，以供下面的两个示例使用。

数据变换器节点（URL 解析器）

数据变换器节点示例用于对上一个示例返回的数据执行进一步处理。您选择一个标识字段（此字段应该包含每一行的相应唯一值）和一个包含 URL 的输入字段。此节点所生成的输出包含这两个字段，并且 URL 数据还解析到各个生成的字段中。例如，如果 URL 记录包含如下查询字符串:

```
http://www.dummydomain.co.uk/resource.php?res_id=89
```

此记录的解析方式如下表所示。

表 10. URL 记录解析示例.

生成的字段	内容
*URLfield*_server	*http://www.dummydomain.co.uk*
*URLfield*_path	*/resource.php*
*URLfield*_field	*res_id*
*URLfield*_value	*89*

要使用“URL 解析器”节点:

1. 如果前一示例中的流已经关闭，请立即将其打开。该流包含“Apache 日志阅读器”节点和“类型”节点。
2. 通过节点选用板的“字段选项”选项卡，将“URL 解析器”节点附加到“类型”节点。
3. 编辑该“URL 解析器”节点。从“标识字段”下拉列表中，选择 **ReturnedContentSize**。从“URL 字段”下拉列表中，选择 **ReferralURL**。单击**确定**。
4. 将“表”节点附加到“URL 解析器”节点并执行流。此时将显示 **ReturnedContentSize** 和 **ReferralURL** 字段，另外，**ReferralURL** 解析到生成的四个不同字段中：**ReferralURL_server**、**ReferralURL_path**、**ReferralURL_field** 和 **ReferralURL_value**。

文档构建器节点（Web 状态报告）

文档构建器节点示例会读取继承自 Web 服务器日志的数据，并生成 HTML 文件形式的报告。该报告中包含的表反映了返回各种 HTTP 状态码（例如，200、302、404 等）的日志记录的百分比。

要使用“Web 状态报告”节点:

1. 如果第一个示例中的流已关闭，请立即将其打开。该流包含“Apache 日志阅读器”节点和“类型”节点。如果您的流具有第二个示例中的“URL 解析器”节点，在本示例中将忽略该节点。
2. 通过节点选用板的“输出”选项卡，将“Web 状态报告”节点附加到“类型”节点。
3. 编辑这个“Web 状态报告”节点。从“状态码字段”下拉列表中，选择 **StatusCode**。单击**执行**。此时将显示包含报告内容的输出窗口。

模型构建器节点 (Interaction)

模型构建器节点示例的操作与其他示例无关，您可以用它以标准（非交互）方法构建简单的模型，也可以在生成该节点之前将其与模型交互。该模型可尝试预测电信公司的客户流失情况。

注: 此示例使用特定于 Windows 的 API 调用来创建和管理线程。因此，在非 Windows 平台上不支持。

要使用“交互”节点

1. 如果尚未激活 CLEF 示例，请立即激活。有关更多信息，请参阅第 23 页的『激活示例』。
2. 在 IBM SPSS Modeler 中创建新流。
3. 添加可从 *Demos* 目录导入文件 *telco.sav* 的 Statistics 文件源节点。
4. 在“类型”选项卡中，单击**读取值**，然后在消息框中单击**确定**以确认。
5. 将**流失**字段（列表中的最后一个）的角色设置为**目标**，然后单击**确定**。
6. 在节点选用板的“建模”选项卡中，将“交互”节点连接到源节点。

要测试标准（非交互）的模型构建

1. 运行流以在流和屏幕右上角的“模型”选用板中创建模型块。
2. 将一个“表”节点附加到模型块。
3. 执行这个“表”节点。滚动到表输出窗口右侧，以查看流失情况预测。字段 `$I-churn` 中包含了预测值，而 `$IP-churn` 则显示这些预测的置信度值（从 0.0 到 1.0）。

要测试交互的模型构建

1. 在“交互”模型构建器对话框的“模型”选项卡中，选择**启动交互式会话**。
2. 单击**执行**可显示“交互测试”对话框。
3. 在“交互测试”对话框中，单击**启动构建任务**以显示模型构建进程。
4. 完成模型构建操作后，在对话框中选择已添加到“构建任务”表中的行。
5. 在对话框顶部的工具栏区域中，单击带有黄色菱形图标的按钮。该操作将在屏幕右上角的“模型”选用板中生成模型输出对象（名为 **model_1**）。

除了名称有所不同以外，以交互方式生成的模型与第一个模型完全相同。重复从单击 **启动构建任务**开始的过程将生成另一个名为 **model_2** 的相同模型，依此类推。

检查规范文件

了解 CLEF 工作原理的一个好办法是检查所提供示例的规范文件。这些文件位于:

install_dir`\ext\lib\`*extension_folder*`\extension.xml`

其中 *install_dir* 是 IBM SPSS Modeler 安装目录，*extension_folder* 是指下列内容之一:

- *spss.apachelogreader*
- *spss.interaction*
- *spss.urlparser*
- *spss.webstatusreport*

您可能会看到 *\ext\lib* 下列出的其他扩展文件夹，这些文件夹与使用 CLEF 生成的系统提供的 IBM SPSS Modeler 节点有关。这些节点是否在安装中显示取决于许可的 IBM SPSS Modeler 模块。您可能还会发现浏览这些节点的规范文件很受启发，但是 **不要以任何方式更改这些文件**。如果更改了这些文件，这些节点将无法正常发挥作用，在这种情况下，您必须重新安装相关的 IBM SPSS Modeler 产品。IBM Corp. 不支持对系统提供的文件进行更改

检查源代码

为了便于参考，还提供了示例节点的完整源代码。所有示例代码都使用 C++ 服务器端库，但“交互”节点还使用客户端 Java 类。

您激活这些示例时，将自动解压缩源代码文件并将其安装在下表所示的位置。

表 11. 源代码文件安装

位置	内容
...\ext\lib\spss.interaction\src	父文件夹中 `ui.jar` 文件中的 `.class` 文件的 Java 源代码

表 11. 源代码文件安装 (续)

位置	内容
...\ext\bin\spss.apachelogreader\src ...\ext\bin\spss.interaction\src ...\ext\bin\spss.urlparser\src ...\ext\bin\spss.webstatusreport\src	父文件夹中 DLL 的 C++ 源代码和项目文件

删除示例

如果无需再查看 IBM SPSS Modeler 中的示例节点，那么可以将其删除，操作步骤如下所示:

1. 退出 IBM SPSS Modeler。
2. 从 IBM SPSS Modeler 安装的 *\ext\bin* 和 *\ext\lib* 目录中删除示例文件夹。不要误删任何标准 IBM SPSS Modeler 文件夹。如果误删，必须重新安装相关的 IBM SPSS Modeler 产品。要删除的文件夹包括:

- *spss.apachelogreader*
- *spss.urlparser*
- *spss.webstatusreport*
- *spss.interaction*

下次启动 IBM SPSS Modeler 时，这些更改才生效。

第 4 章 规范文件

规范文件概述

每个 CLEF 扩展都必须包含一个用于定义所有扩展特征的 XML 文件。这个文件称为 **规范文件**，通常名为 `extension.xml`。规范文件包括以下几个部分:

- **XML 声明**。有关 XML 版本的可选声明以及其他信息。
- **扩展元素**。文件的主要部分；包含后续所有部分。
- **“扩展详细信息”部分**。指定有关扩展的基本信息。
- **“资源”部分**。指定此扩展正常工作所需的外部资源，例如资源束、JAR 文件和共享库。
- **“公共对象”部分**。（可选）定义可以由扩展中的其他对象使用或引用的项目，例如模型、文档和属性类型。
- **“用户界面（选用板）”部分**。（可选）定义要在其中显示节点的定制选用板或子选用板。
- **“对象定义”部分**。标识由扩展定义的对象，如节点、模型输出和文档输出。

每个部分既可以包含静态声明（例如元素中的组件），也可以包含简单的动态流程（例如计算节点的输出数据模型），并可以同时包含这两者。完整的 CLEF 规范文件格式如下所示:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Extension ... >
   <ExtensionDetails ... />
   <Resources      "资源"部分
   </Resources>
   <CommonObjects>
      "公共对象"部分
   </CommonObjects>
   <UserInterface>
      "用户界面（选用板）"部分
   </UserInterface>
   "对象定义"部分
     对象定义
     对象定义
     对象定义
    ...
</Extension>
```

注释行

在规范文件的任意位置，您都可以添加如下格式的注释行:

```
<!- comment text ->
```

必需项还是可选项?

在后面各章节的元素定义中（通常由标题**格式**标识），除非元素属性及子元素指明为“（必需）”，否则均为可选项。要了解完整的元素语法，请参阅第 189 页的『CLEF XML 模式』。

规范文件的示例

下面是一个完整的 CLEF 规范文件示例，本例适用于简单的数据转换节点。

```
<?xml version="1.0" encoding="UTF-8"?>
<Extension version="1.0" debug="true">
     <ExtensionDetails id="urlparser" providerTag="spss" label="URL CLEF Module" version="1.0"
provider="IBM Corp." copyright="(c) 2005-2011 IBM Corp." description="A Url Transform CLEF Extension"/>
     <Resources>
          <SharedLibrary id="urlparser_library" path="spss.urlparser/urlparser" />
     </Resources>
     <Node id="urlparser_node" type="dataTransformer" palette="fieldOp" label="URL Parser">
          <Properties>
               <Property name="id_fieldname" valueType="integer" label="ID field" />
               <Property name="url_fieldname" valueType="string" label="URL field" />
          </Properties>
          <UserInterface>
               <Icons />
               <Tabs>
                    <Tab label="Types" labelKey="optionsTab.LABEL">
                         <PropertiesPanel>
                              <SingleFieldChooserControl property="id_fieldname"  storage="integer" />
                              <SingleFieldChooserControl property="url_fieldname"  storage="string" />
                         </PropertiesPanel>
                    </Tab>
               </Tabs>
               <Controls />
          </UserInterface>
          <Execution>
               <Module libraryId="urlparser_library" name="">
                    <StatusCodes>
                         <StatusCode code="0" status="error" message="Cannot initialise a peer" />
                         <StatusCode code="1" status="error" message="error reading input data" />
                         <StatusCode code="2" status="error" message="Internal Error" />
                         <StatusCode code="3" status="error" message="Input Field Does Not Exist" />
                    </StatusCodes>
               </Module>
          </Execution>
          <OutputDataModel mode="replace">
               <AddField name="${id_fieldname}" fieldRef="${id_fieldname}"/>
               <AddField name="${url_fieldname}" fieldRef="${url_fieldname}"/>
               <AddField name="${url_fieldname}_server" storage="string" />
               <AddField name="${url_fieldname}_path" storage="string" />
               <AddField name="${url_fieldname}_field" storage="string" />
               <AddField name="${url_fieldname}_value" storage="string" />
          </OutputDataModel>
     </Node>
</Extension>
```

ExtensionDetails 元素提供有关在 IBM SPSS Modeler 内部使用的扩展的基本信息。

Resources 元素用于指定服务器端库的位置，该位置稍后将在文件中引用。路径规范表明该库位于 \ext\bin\spss.urlparser\urlparser.dll 下的 IBM SPSS Modeler 安装目录中。

这个特定的规范文件并不包含 CommonObjects 元素。

Node 元素指定与节点本身有关的所有信息:

- 初始状态下，Properties 元素下具有两个已声明的属性，可供稍后在节点对话框的选项卡中使用。
- UserInterface 元素定义此扩展专用的节点对话框选项卡的外观和布局（其他选项卡由 IBM SPSS Modeler 提供）。
- Execution 元素则定义执行节点时所使用的项目。在本例中，这些项目是之前在文件中声明的服务器端库以及要在执行返回特定状态码时显示的一组消息。
- OutputDataModel 元素定义此节点执行的数据转换。此元素指定输入数据模型（输入到此节点中的字段集）将由此处定义的一组字段替换，这些字段构成输出数据模型（从此处传递到所有下游节点的字段集，除非此后对模型进行进一步修改）。在这个特定的示例中，节点并未更改传递的两个原始字段（ id_fieldname 和 url_fieldname），而是另外添加了四个字段，且其名称派生自 url_fieldname。

这个特定的规范文件是从作为 IBM SPSS Modeler 安装程序的一部分提供的示例节点之一中提取出来的。有关更多信息，请参阅第 24 页的『数据变换器节点（URL 解析器）』主题。

XML 声明

XML 声明是可选项，用于指定当前所用 XML 的版本，同时还提供字符编码格式的详细信息。

示例

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Extension 元素

Extension 元素是文件的主要组成部分，其中包含所有其他部分。格式为:

```
<Extension version="version_number" debug="true_false">
     "扩展详细信息"部分
     "资源"部分
     "公共对象"部分
     "用户界面（选用板）"部分
     "对象定义"部分
</Extension>
```

其中:

version 是此扩展的版本号。

debug 是可选项；如果设置为 true，那么将对任何与 CLEF 节点或输出相关联的对话框或框架中添加**调试**选项卡，并允许访问针对该对象定义的属性和容器。缺省值为 false。有关更多信息，请参阅第 186 页的『使用“调试”选项卡』主题。

“扩展详细信息”部分

“扩展详细信息”部分提供有关扩展的基本信息。

格式

```
<ExtensionDetails providerTag="extension_provider_tag"
     id="extension_unique_identifier"
     label="display_name" version="extension_version_number"
     provider="extension_provider" copyright="copyright_notice"
     description="extension_description"/>
```

其中:

providerTag（必需）是一个名称，用于唯一地标识此扩展的供应商。请注意，值不应包含字符串 spss，这是仅供内部使用的预留字符串。

id（必需）是一个名称，用于唯一地标识此扩展，并且在有关此扩展的系统消息中使用。按照惯例，扩展文件将放置在 IBM SPSS Modeler 安装目录中名为 \ext\lib\ *providerTag.id* 的文件夹中。

label（必需）是此扩展的显示标签。添加节点时，此文本显示在选用板管理器的“名称”字段中。有关更多信息，请参阅第 185 页的『测试 CLEF 扩展』主题。

version 是此扩展的版本号。

provider 是一个字符串，用于标识此扩展的供应商。添加节点时，此文本显示在选用板管理器的“供应商”字段中。缺省值为字符串 (unknown)。

copyright 是此扩展的版权声明。添加节点时，此文本显示在选用板管理器的“版权”字段中。

description 是有关此扩展的用途的简要描述。添加节点时，此文本显示在选用板管理器的“描述”字段中。

示例

```
<ExtensionDetails providerTag="myco" id="sorter" name="Sort Data" version="1.2"
     provider="My Company Inc." copyright="(c) 2005-2006 My Company Inc."
     description="An example extension that sorts data using built-in OS commands."/>
```

“资源”部分

本节定义要想此扩展正常发挥作用需要哪些外部资源。

格式

```
<Resources>
     <Bundle .../>
      ...
     <JarFile .../>
      ...
     <SharedLibrary .../>
      ...
     <HelpInfo .../>
</Resources>
```

其中:

Bundle 用于标识一组客户端本地化资源。有关更多信息，请参阅『数据包』主题。

JarFile 用于标识客户端 Java jar 文件。有关更多信息，请参阅第 33 页的『Jar 文件』主题。

SharedLibrary 用于标识服务器端库或 DLL。有关更多信息，请参阅第 33 页的『共享库』主题。

HelpInfo 用于指定此扩展的帮助信息（如果有）的类型。有关更多信息，请参阅第 149 页的『实现帮助系统』主题。

示例

```
<Resources>
     <SharedLibrary id="discriminantnode" path="spss.xd/Discriminant"/>
     <Bundle id="translations.discrim" type="properties" path="messages"/>
     <JarFile id="java" path="discriminant.jar"/>
     <HelpInfo id="help" type="native"/>
</Resources>
```

数据包

Bundle 元素指定位于客户端的资源束（例如一组用于本地化的消息文本），它可作为 .properties 文件或 Java .class 文件实现。有关更多信息，请参阅第 153 页的『本地化』主题。

格式

```
<Bundle id="identifier" path="path"/>
```

其中:

id（必需）是此捆绑包的唯一标识。

path（必需）指定捆绑包文件的位置（相对于此规范文件的父文件夹）。其中，捆绑包指 .properties 文件，路径不得包含语言扩展或 .properties 后缀。

示例

```
<Bundle id="translations.discrim" path="messages"/>
```

这表示在规范文件所在的同一个文件夹中，一个名为 messages.properties 的文件中存在资源束。

Jar 文件

JarFile 元素指定客户端 Java 存档 (.jar) 文件，该文件为此扩展提供 Java 类和其他客户端资源。

格式

```
<JarFile id="identifier" path="path"/>
```

其中:

id（必需）是此 .jar 文件的唯一标识。

path（必需）指定 .jar 文件的位置（相对于此规范文件的父文件夹）。

示例

```
<JarFile id="java" path="coxreg_model_terms.jar"/>
```

这表示此扩展的 .jar 文件位于规范文件所在的文件夹中。

共享库

SharedLibrary 元素指定服务器端共享库或 DLL。通常，只有在支持节点执行时，才需要使用此元素。在一个实现了多个模块的库中，节点规范“执行”部分中的 Module 元素用于标识库中的一个具体模块。

格式

```
<SharedLibrary id="identifier" path="path"/>
```

其中:

id（必需）是此共享库的唯一标识。

path（必需）指定共享库的位置（相对于服务器端安装目录中的 \ext\bin 文件夹）。请注意，该路径不能包含共享库文件扩展名（例如 .dll）。

示例

下面的共享库声明:

```
<SharedLibrary id="binning" path="spss.binning/Binning" />
```

指定共享库加载自:

install_dir\ext\bin\spss.binning\Binning.dll

其中，*install_dir* 是服务器端 CLEF 组件的安装目录。由于此库实现了多个模块，所需的特定模块 (supervisedBinning) 由构建节点规范中的 Module 元素标识，此时将引用如下库标识:

```
<Execution>
     <Module libraryId="binning" name="supervisedBinning" .../>
 ...
</Execution>
```

帮助信息

可选的 HelpInfo 元素指示能为此扩展提供哪些类型的帮助。有关更多信息，请参阅第 149 页的『实现帮助系统』主题。

“公共对象”部分

可选的“公共对象”部分用于定义可以在规范文件其他位置定义的元素间共享的对象。此部分中某些类型的对象（例如属性枚举）也可以根据需要在本地定义，但其他对象（例如模型和文档）只能在此定义。

格式

```
<CommonObjects>
     <PropertyTypes .../>
     <PropertySets .../>
     <ContainerTypes .../>
     <Actions .../>
     <Catalogs .../>
</CommonObjects>
```

其中:

PropertyTypes 允许在对象之间共享公共属性定义。有关更多信息，请参阅第 35 页的『属性类型』主题。

PropertySets 通常在模型构建器节点、模型输出对象和模型应用器节点包含同一组属性时使用。有关更多信息，请参阅第 36 页的『属性集』主题。

ContainerTypes 用于定义容器的类型（容器是可以容纳复杂数据结构的对象）。有关更多信息，请参阅第 36 页的『容器类型』主题。

Actions 用于定义有关用户交互（例如通过菜单或工具栏进行的交互）的基本信息。有关更多信息，请参阅第 38 页的『Actions』主题。

Catalogs 实现一个控件，该控件允许从服务器动态生成的值列表中选择一个或多个选项。有关更多信息，请参阅第 39 页的『Catalogs』主题。

示例

```
<CommonObjects>
     <ContainerTypes>
          <ModelType id="discriminant_model" format="utf8" />
          <DocumentType id="html_output" />
          <DocumentType id="zip_outputType" format="binary"/>
     </ContainerTypes>
</CommonObjects>
```

属性类型

可选的“属性类型”部分允许在对象之间共享公共属性定义。这在某种程度上可以减轻维护工作量，例如，某个属性的定义可以出现在单个位置，而不必在多个位置重复出现。定义共享也可用于确保当创建新的对象实例时，不同对象中复制了值的属性间相互兼容。

属性类型只能在“公共对象”部分中定义。

格式

```
<PropertyTypes>
     <PropertyType id="identifier" isKeyed="true_false" isList="true_false" max="max_value"
       min="min_value" valueType="value_type">
          <Enumeration ... />
          <Structure ... />
          <DefaultValue ... />
     </PropertyType>
     <PropertyType ... />
      ...
</PropertyTypes>
```

PropertyType 属性如下所示。

id（必需）是属性类型的唯一标识。

isKeyed，如果设置为 true，那么表示这是键控类型的属性。键控属性通过用户定制控件将一组操作关联到某个字段（请参阅第 127 页的『属性控件』）。如果 isKeyed 设置为 true，那么 valueType 属性必须设置为 structure。有关结构化属性的更多信息，请参阅第 58 页的『结构化属性』。

isList 指定该属性是具有指定值类型的值的列表 (true) 还是单个值 (false)。

max 和 min 表示范围的最大值和最小值。

valueType 可以是下列其中一项:

- string
- encryptedString
- fieldName
- integer
- double
- boolean
- date
- enum（请参阅第 57 页的『枚举属性』）
- structure（请参阅第 58 页的『结构化属性』）
- databaseConnection

Enumeration 和 Structure 子元素相互排斥。在特定的情况下，还会使用 Enumeration、Structure 和 DefaultValue 子元素，请参阅第 57 页的『枚举属性』、第 58 页的『结构化属性』和第 60 页的『缺省值』。

属性集

当模型构建器节点、模型输出对象和模型应用器节点包含同一组属性时，通常会用到属性集。例如，模型构建器节点可以定义一个可以在构建器中设定但直到模型应用时才实际使用的缺省属性集。为了能够自动传输属性集，还必须将它们包含在模型输出中。

格式

```
<PropertySets>
     <PropertySet id="identifier">
          <Property ... />
          <Property ... />
           ...
     </PropertySet>
       ...
</PropertySets>
```

其中，id 是此属性集的唯一标识。

有关 Property 元素的描述，请参阅第 48 页的『属性』。

示例

本示例演示的定义由两个属性组成：要生成的预测数以及是否包括概率。在“公共对象”部分，您将定义：

```
<PropertySets>
     <PropertySet id="common_model_properties">
          <Property name="prediction_count" valueType="integer" min="1" max="10"/>
          <Property name="include_probabilities" valueType="boolean" defaultValue="false"/>
     </PropertySet>
       ...
</PropertySets>
```

然后，在模型构建器节点、模型输出对象和模型应用器节点各自的定义中，都将包含一个 includePropertySets 属性，如下所示（此处仅举例说明模型构建器节点的定义）：

```
<Node id="my_builder" type="modelBuilder" ... >
     <Properties includePropertySets="[common_model_properties]">
      ...
     </Properties>
      ...
</Node>
```

容器类型

容器是充当复杂数据结构（例如模型和文档）的占位符的对象。容器定义总是包含具体的类型，而容器类型将在此处定义。可以定义下列容器类型:

- 模型类型
- 文档类型

容器类型可以在客户端与服务器之间传输，进行克隆，并保存到文件或内容存储库中。当从模型输出对象生成模型应用器节点时，将会对模型进行克隆。

每种类型的容器都有一组预定义的属性，不过，可以添加定制属性。容器类型只能在“公共对象”部分定义。

格式

容器类型部分的格式为:

```
<ContainerTypes>
     <ModelType ... />
      ...
     <DocumentType ... />
      ...
</ContainerTypes>
```

其中:

ModelType 指定特定模型类型的格式。有关更多信息，请参阅『模型类型』主题。

DocumentType 指定特定文档类型的格式。有关更多信息，请参阅『文档类型』主题。

示例

```
<ContainerTypes>
     <ModelType id="discriminant_model" format="utf8">
     <DocumentType id="html_output" />
     <DocumentType id="zip_outputType" format="binary"/>
</ContainerTypes>
```

模型类型

模型必须提供算法名称、模型类型以及输入和输出数据模型等信息。模型类型定义用于指定特定模型类型的格式。

模型类型信息可以在规范文件中的这一部分静态指定，也可以在模型构建器节点构建模型时动态指定。

格式

```
<ModelType id="identifier" format="model_type_format" />
```

其中:

- id（必需）是模型类型的唯一标识。
- format（必需）是模型类型的格式，这可以是 utf8（文本）或 binary。必须将模型格式指定为静态信息的一部分。

示例

```
<ModelType id="my_model" format="utf8" />
```

文档类型

document 是一种输出对象，如图形或报告。文档类型定义用于指定特定文档类型的格式。

格式

```
<DocumentType id="identifier" format="document_type_format" />
```

其中:

- id（必需）是文档类型的唯一标识。
- format（必需）是文档类型的格式，这可以是 utf8（文本）或 binary。

示例

```
<DocumentType id="html_output" format="utf8" />
<DocumentType id="zip_outputType" format="binary"/>
```

Actions

操作用于定义有关用户交互（例如通过菜单或工具栏进行的交互）的基本信息。每个操作定义标签、工具提示或图标等在用户界面中的表示方式。操作集合则由针对每个操作组定义的客户端 Java 类处理。操作也可以在具体的对象中定义。

格式

```
<Actions>
     <Action id="identifier" label="display_label" labelKey="label_key"
     description="action_description" descriptionKey="description_key" imagePath="image_path"
     imagePathKey="image_path_key" mnemonic="mnemonic_char" mnemonicKey="mnemonic_key"
     shortcut="shortcut_string" shortcutKey="shortcut_key" />
      ...
</Actions>
```

其中:

id（必需）是唯一的操作标识。

label（必需）是要在用户界面上显示的操作显示名称。

labelKey 用于标识标签以便进行本地化。

description 是对操作的描述，例如，对于定制菜单项或工具栏上的图标操作按钮，这将是该菜单项或按钮的工具提示文本。

descriptionKey 用于标识描述以便进行本地化。

imagePath 是图形文件（例如图标图像）的位置。提供的位置相对于规范文件的安装目录。

imagePathKey 用于标识图像路径以便进行本地化。

mnemonic 是与 Alt 键一起使用以激活此控件的字母符号（例如，如果您给其值为 S，那么用户可通过 Alt-S 激活此控件）。

mnemonicKey 用于标识助记符以便进行本地化。如果不使用 mnemonic 和 mnemonicKey，那么此控件无可用的助记符。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

shortcut 是指示可用于启动操作的键盘快捷键（例如，CTRL+SHIFT+A）的字符串。

shortcutKey 用于标识快捷键以便进行本地化。如果不使用 shortcut 和 shortcutKey ，那么该操作无可用的快捷键。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

示例

```
<Actions>
     <Action id="generateSelect" label="Select Node..." labelKey="generate.selectNode.LABEL"
      imagePath="images/generate.gif" description="Generates a select node"
      descriptionKey="generate.selectNode.TOOLTIP"/>
     <Action id="generateDerive" label="Derive Node..." labelKey="generate.deriveNode.LABEL"
      imagePath="images/generate.gif" description="Generates a derive node"
      descriptionKey="generate.deriveNode.TOOLTIP"/>
</Actions>
```

Catalogs

目录可以将某个属性与控件关联，以允许用户从由服务器动态生成的值列表中选择一个或多个选项。

当用户单击 **<Select>** 条目时，值以弹出列表形式显示在控件中。

当用户在列表中选择一行时，在 Catalog 元素中的指定列上的行值将被放入控件中。

格式

```
<CommonObjects>
     <Catalogs>
          <Catalog id="identifier" valueColumn="integer">
               <Attribute label="display_name" />
                ...
          </Catalog>
           ...
     </Catalogs>
</CommonObjects>
```

其中:

id（必需）是目录的唯一标识。

valueColumn（必需）是当用户选择某行时其值将被放入控件中的列的编号。列从 1 开始编号。

按列为每个列使用一个 Attribute 元素，请参阅下面的示例。

当用户激活与目录关联的控件时，通过调用 getCatalogInformation 函数从服务器检索到包含值列表的目录。此函数返回包含值列表的 XML 文档。有关更多信息，请参阅第 164 页的『对等函数』主题。

示例

此示例演示一些用于定义目录控件的代码。这里定义了三个目录，并使其与对话框选项卡上的三个不同控件相关联。

首先，在“公共对象”部分中定义目录:

```
<CommonObjects>
     <Catalogs>
          <Catalog id="cat1" valueColumn="1">
               <Attribute label="col1" />
               <Attribute label="col2" />
          </Catalog>
          <Catalog id="cat2" valueColumn="2">
               <Attribute label="col1" />
               <Attribute label="col2" />
               <Attribute label="col3" />
          </Catalog>
          <Catalog id="cat3" valueColumn="1">
               <Attribute label="col1" />
          </Catalog>
     </Catalogs>
</CommonObjects>
```

然后，在节点定义的“属性”部分定义与控件关联的属性:

```
<Node id="catalognode" type="dataReader" palette="import" label="Catalog">
     <Properties>
          <Property name="sometext" valueType="string" label="Some Text" />
```

```
        <Property name="selection1" valueType="string" label="Selection 1" />
        <Property name="selection2" valueType="string" isList="true" label="Selection 2" />
        <Property name="selection3" valueType="string" label="Selection 3" />
    </Properties>
```

在节点定义的“用户界面”部分中，对控件进行定义并通过属性引用将其关联到目录定义:

```
<UserInterface>
    <Tabs>
        <Tab label="Catalog Controls" labelKey="Catalog.LABEL" >
            <PropertiesPanel>
                <TextBoxControl property="sometext" />
                <SingleItemChooserControl property="selection1" catalog="cat1" />
                <MultiItemChooserControl property="selection2" catalog="cat2" />
                <SingleItemChooserControl property="selection3" catalog="cat3" />
            </PropertiesPanel>
        </Tab>
```

“用户界面（选用板）”部分

这是一个可选部分，且仅当要将此扩展用于定义在其上显示节点的定制选用板或子选用板时才会包括这一部分。

在定义了定制选用板或子选用板的扩展中，当后续加载的扩展用于定义要包括在同一个选用板或子选用板中的节点时，可以省略此“用户界面（选用板）”部分，此时只需 Node 元素包含引用此选用板的 customPalette 属性即可。扩展按照 *providerTag*. *id* 值的字母顺序加载，其中，providerTag 和 id 是此扩展中 ExtensionDetails 元素的 providerTag 和 id 属性的值（请参阅 第 31 页的『“扩展详细信息”部分』）。因此，将先加载扩展 myco.abc，再加载扩展 myco.def。

注意: “用户界面（选用板）”部分与主“用户界面”部分不同，后者作为单个对象定义的组成部分出现，相关的描述在第 95 页的第 6 章，『构建用户界面』中提供。

格式

“用户界面（选用板）”部分的格式为:

```
<UserInterface>
    <Palettes>
        <Palette id="name" systemPalette="palette_name" customPalette="palette_name"
           relativePosition="position" relativeTo="palette" label="display_label"
           labelKey="label_key" description="description" descriptionKey="description_key"
           imagePath="image_path" />
        <Palette ... />
     ...
    </Palettes>
</UserInterface>
```

表 12. 选用板属性.

属性	描述
id	（必需）要定义的选用板或子选用板的唯一标识。

表 12. 选用板属性 (续).

属性	描述
systemPalette	仅当在系统选用板中添加子选用板时使用，用于标识将显示此子选用板的系统选用板: import - 源 recordOp - 记录选项 fieldOp - 字段选项 graph - 图形 modeling - 建模（请参阅下文） dbModeling - 数据库建模 output - 输出 export - 导出
customPalette	仅当在定制选用板中添加子选用板时使用，用于标识将显示此子选用板的定制选用板。这是定义定制选用板的 Palette 元素的 id 属性的值。
relativePosition	仅当定义定制选用板时使用，用于指定选用板在屏幕底部的选用板栏中的位置。 可能的值包括: 第一个 last before after 如果值为 before 或 after，那么还要求具有 relativeTo 属性（请参阅下文）。 如果省略 relativePosition，此选用板将位于选用板栏的最后。
relativeTo	如果 relativePosition 的值为 before 或 after，那么 relativeTo 用于指定此定制选用板之前或之后的选用板的标识。选用板标识将作为 Node 元素的选用板属性值列出（请参阅 第 45 页的『节点』）。
label	（必需）当选用板或子选用板显示在用户界面上时所显示的名称。
labelKey	用于标识标签以便进行本地化。
description	这是光标悬停在选用板选项卡（不适用于子选用板）上时显示的工具提示文本。此值还可作控件的详细辅助描述。有关更多信息，请参阅第 159 页的『辅助功能选项』主题。
descriptionKey	用于标识说明以便进行本地化。
imagePath	用于标识选用板选项卡（不适用于子选用板）上使用的图像的位置。此位置相对于规范文件的安装目录进行指定。如果您省略此属性，将不使用任何图像。

示例 - 添加节点至系统选用板

假设您的组织发明了一种新的挖掘音频和视频率据的算法，您要将此算法集成到 IBM SPSS Modeler 中。您从定义定制数据阅读器节点开始，该节点将用于从音频和视频文件中读取输入数据。

开始时，您决定将新的数据阅读器节点添加到“源”系统选用板中。您只需通过 Node 元素的 palette 属性来标识源选用板。有关更多信息，请参阅第 45 页的『节点』主题。

使用此方法将此节点添加到“源”选用板中的“数据库”节点之后，可以使用:

```
<Node id="AVreader" type="dataReader" palette="import" relativePosition="after"
   relativeTo="database" label="AV Reader">
```

示例 - 添加定制选用板

虽然使用标准 IBM SPSS Modeler 选用板就能够满足要求，但您希望新节点显得更加突出。所以您决定为此节点定义一个定制选用板，并且将该选用板置于“收藏夹”选用板之后、“源”选用板之前。首先，您需要添加“用户界面（选用板）”部分，以便按照下列操作定义定制选用板:

```
<UserInterface>
     <Palettes>
          <Palette id="AV_mining" label="AV Mining" relativePosition="before"
      relativeTo="import" description="Audio video mining"  />
     </Palettes>
</UserInterface>
```

relativeTo 属性必须使用“源”选用板的内部标识，即 import。

然后按照下列操作更改 Node 定义:

```
<Node id="AVreader" type="dataReader" customPalette="AV_mining" label="AV Reader">
```

这样会将 **AV 挖掘**选用板置于“收藏夹”和“源”系统选用板之间。

示例 - 添加定制子选用板至定制选用板

继续使用上一个示例，您现在更希望将数据阅读器节点置于 **AV 挖掘**选用板上自己的 **AV 源**子选用板中。为此，您首先需要通过在“用户界面（选用板）”部分中添加另一个 Palette 元素来指定子选用板:

```
<UserInterface>
     <Palettes>
          <Palette id="AV_mining" label="AV Mining" description="Audio video mining" />
          <Palette id="AV_mining.sources" customPalette="AV_mining" label="AV Sources" />
     </Palettes>
<UserInterface>
```

然后，更改 Node 元素，以引用子选用板的标识:

```
<Node id="AVreader" type="dataReader" customPalette="AV_mining.sources" label="AV Reader">
```

现在，当用户单击 **AV 挖掘**选项卡时，他们将看到两个子选用板，一个标记为 **所有**，而另一个标记为 **AV 源**。这两个子选用板上均显示“AV 阅读器”节点。

如果您在 **AV 挖掘**的另一个新子选用板中再添加一个新的节点，这个新节点将同时显示在 **所有**和新的子选用板中，但不会显示在 **AV 源**子选用板中。

示例 - 添加节点至系统子选用板

为了处理音频和视频源数据，您现在定义了一个模型构建器节点。您决定将这个节点添加到标准的“建模”选用板中，该选用板包含多个标准子选用板。您选择将其添加到“分类”子选用板中，并置于“神经网络”节点之前，因此需要指定:

```
<Node id="AVmodeler" type="modelBuilder" palette="modeling.classification"
   relativePosition="before" relativeTo="neuralnet" label="AV Modeler">
```

请注意，还将在“建模”选用板的“所有”子选用板上的同一个相对位置添加该节点。

示例 - 添加定制子选用板至系统选用板

再次查看“分类”子选用板中模型构建器节点的数目，您意识到用户可能不容易注意到您添加的新节点。要让您的节点显得更为突出，一种方法就是在“建模”选用板中添加自己的子选用板并在其中放置该节点。

因此，您必须先通过在文件中添加“用户界面（选用板）”部分来定义定制子选用板:

```
<UserInterface>
     <Palettes>
          <Palette id="modeling.av_modeling" systemPalette="modeling" label="AV Modeling"
             labelKey="av_modeling.LABEL" description="Contains AV mining-related modeling
             nodes" descriptionKey="av_modeling.TOOLTIP"/>
     </Palettes>
</UserInterface>
```

请注意，您必须明确指定将 systemPalette 用于标识要扩展的系统选用板。

然后，在节点的主“用户界面”部分，您指定它在此子选用板中显示:

```
<Node id="my.avmodeler" type="modelBuilder" customPalette="modeling.av_modeling"
   label="AV Modeler">
```

定制子选用板始终位于系统子选用板之后。

注意: 如果要在“AV 建模”子选用板中添加更多节点，只要先加载了 AV Modeler 扩展，这些节点的规范文件就**不**再需要“用户界面（选用板）”部分。

隐藏或删除定制选用板或子选用板

如果不希望显示某个定制选用板或子选用板，可以通过 IBM SPSS Modeler 选用板管理器隐藏或删除它。

请注意，隐藏操作会在多个 IBM SPSS Modeler 会话中持续发挥作用，但此操作由复选框控制，因此可以恢复为正常显示。删除操作无法在同一会话中恢复，但重新启动 IBM SPSS Modeler 后，被删除的项目会重新显示，除非您已从规范文件中将其删除，或者删除了整个扩展。有关更多信息，请参阅第 188 页的『卸载 CLEF 扩展』主题。

要隐藏或删除选用板，请执行下列操作:

1. 在 IBM SPSS Modeler 主菜单中，选择:

   工具 > 管理选用板
2. 在“选用板名称”字段中选择所需选用板，然后:
   - 要隐藏选用板，请取消选中相应的“显示？”复选框
   - 要删除选用板，请单击“删除”选择按钮
3. 单击“确定”。

要隐藏或删除子选用板，请执行下列操作:

1. 在 IBM SPSS Modeler 主菜单中，选择:

   工具 > 管理选用板
2. 在“选用板名称”字段中选择一个选用板。
3. 单击“子选用板”按钮。
4. 在“子选用板名称”字段中选择所需子选用板，然后:
   - 要隐藏子选用板，请取消选中相应的“显示？”复选框

- 要删除子选用板，请单击“删除”选择按钮

5. 单击“确定”。

“对象定义”部分

在扩展中，元素是主要的可视部件。“对象定义”部分组成了 CLEF 规范文件的其余部分，用于定义扩展中的各种对象。可定义的对象类型包括:

- 节点
- 模型输出对象
- 文档输出对象
- 交互式输出对象

节点是显示在流中的对象。**模型输出对象**由模型构建器节点生成，显示在主窗口管理器窗格的“模型”选项卡下。同样，**文档输出对象**由文档构建器节点生成，显示在同一个窗格的“输出”选项卡下。**交互式输出对象**由交互式模型构建器节点生成，显示在管理器窗格的“输出”选项卡下。

“对象定义”部分由一个或多个此类对象定义组成。

下列各节对可以为不同类型对象定义的元素作了描述。在这些元素中，其中一部分是所有对象类型的公共元素，而其他元素则特定于节点或模型输出定义。对象特定的元素在文本中由以下项目表示。

- 对象标识
- 模型构建器
- 文档构建器
- 属性
- 容器
- 用户界面
- 执行
- 输出数据模型
- 构造函数

对象标识

对象标识用于标识对象的类型，包含以下几种:

```
<Node .../>

<ModelOutput .../>

<DocumentOutput .../>

<InteractiveModelBuilder .../>
```

对象标识还提供有关应该如何通过脚本编制透露对象的信息。scriptName 属性表示对象的唯一名称。脚本可以使用此属性指定特定的对象（例如，流中的一个节点，或“输出”选项卡中的输出）。

节点

节点定义用于描述可以在流中出现的对象。

格式

```
<Node id="identifier" type="node_type" palette="palette" customPalette="custom_palette"
     relativePosition="position" relativeTo="node" label="display_label" labelKey="label_key"
     scriptName="script_name" helpLink="topic_id" description="description"
     descriptionKey="description_key" delegate="Java_class">
     <ModelBuilder ... >
      ...
     </ModelBuilder>
     <DocumentBuilder ... >
      ...
     </DocumentBuilder>
     <ModelProvider ... />
     <Properties>
      ...
     </Properties>
     <Containers>
      ...
     </Containers>
     <UserInterface>
      ...
     </UserInterface>
     <Execution>
      ...
     </Execution>
     <OutputDataModel ...>
      ...
     </OutputDataModel>
     <Constructors>
      ...
     </Constructors>
</Node>
```

第 48 页的『属性』及后续章节对允许在节点定义中使用的元素作了描述。

表 13. 节点属性.

属性	描述
id	（必需）该节点的标识，采用文本字符串格式。

表 13. 节点属性 (续).

属性	描述
type	（必需）节点的类型包括: dataReader- 用于读取数据的节点（例如, “源”选用板节点） dataWriter - 用于记录数据的节点（例如, “导出”选用板节点） dataTransformer - 用于转换数据的节点（例如, “记录”/“字段选项”节点） modelBuilder - 模型构建器节点（例如, “建模”选用板节点） documentBuilder - 这是用于创建图形或报告的节点 modelApplier - 这是包含所生成的模型的节点 节点类型确定了选用板和工作区中节点图标的形状。有关更多信息，请参阅第 9 页的『节点概述』主题。 如果节点类型为 modelBuilder，那么节点定义必须包含 ModelBuilder 元素，请参阅第 48 页的『模型构建器』。 如果节点类型为 documentBuilder，那么节点定义必须包含 DocumentBuilder 元素，请参阅第 48 页的『文档构建器』。
palette	要在其上显示节点的标准 IBM SPSS Modeler 选用板或子选用板之一的标识，名为: import - 源 recordOp - 记录选项 fieldOp - 字段选项 graph - 图形 modeling - 建模（请参阅下文） dbModeling - 数据库建模 output - 输出 export - 导出 “建模”选用板包含多个标准子选用板: modeling.classification - 分类 modeling.association - 关联 modeling.segmentation - 分段 modeling.auto - 自动化 如果省略 palette 属性，节点将显示在“字段选项”选用板上。 注意: palette 属性仅用于模型构建器节点。
customPalette	这是要在其中显示节点的定制选用板或子选用板的标识。这是 Palette 元素 id 属性的值，该值在文件的“用户界面（选用板）”部分指定。有关更多信息，请参阅第 40 页的『“用户界面（选用板）”部分』主题。

表 13. 节点属性 (续).

属性	描述
relativePosition	指定节点在选用板中的位置。可能的值包括: 第一个 last before after 如果值为 before 或 after，那么还要求具有 relativeTo 属性（请参阅下文）。 如果省略 relativePosition，此节点将位于选用板的最后。
relativeTo	如果 relativePosition 的值为 before 或 after，那么 relativeTo 用于指定选用板中此节点之前或之后的节点的标识。relativeTo 的值是节点的脚本名称。 对于标准 IBM SPSS Modeler 节点，脚本名称可以在《 *IBM SPSS Modeler 脚本编写与自动化指南*》的“属性参考”部分找到，但不带 ...node 后缀（例如，对于“数据库”节点，需要使用 database，而不是 databasenode）。 对于 CLEF 节点，此为该节点 scriptName 属性的值。
label	（必需）当节点出现在选用板、工作区和对话框中时显示的节点名称。
labelKey	用于标识标签以便进行本地化。
scriptName	用于在脚本中引用节点时唯一地标识节点。有关更多信息，请参阅第 71 页的『在脚本中使用 CLEF 节点』主题。
helpLink	这是要在用户调用帮助系统（如果有）时显示的帮助主题的可选标识。此标识的格式取决于帮助系统的类型（请参阅第 149 页的第 7 章，『添加帮助系统』）: HTML Help - 帮助主题的 URL JavaHelp - 主题标识
description	节点的文本说明。
descriptionKey	用于标识说明以便进行本地化。
delegate	如果指定，请定义实现 NodeDelegate 界面的 Java 类的名称。将针对关联的节点的每个实例构造指定的类的实例。

第 48 页的『模型构建器』及后续章节对可以包含在节点定义中的元素作了描述。

示例

要查看节点定义的示例，请参阅第 29 页的『规范文件的示例』。

模型输出

模型输出定义对生成的模型进行描述，在这里，“生成的模型”是指执行流后显示在管理器窗格中“模型”选项卡上的对象。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 80 页的『模型输出』。

文档输出

文档输出定义描述生成的表或图形等对象，此类对象将在执行流后显示在管理器窗格中的“输出”选项卡下。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 90 页的『文档输出』。

交互式模型构建器

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 81 页的『构建交互式模型』。

模型构建器

此元素仅在 Node 元素定义中使用。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 73 页的第 5 章，『构建模型和文档』。

文档构建器

此元素仅在 Node 元素定义中使用。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 73 页的第 5 章，『构建模型和文档』。

模型提供器

此元素仅在 Node 元素定义中使用。

当定义模型输出对象和模型应用器节点时，您可以使用 ModelProvider 元素指定用于容纳该模型的容器。您还可以指定是否以 PMML 格式存储该模型。PMML 模型可以通过定制查看器查看，也可以通过标准 IBM SPSS Modeler 模型输出查看器查看，该标准查看器由 ModelViewerPanel 元素提供。有关更多信息，请参阅第 110 页的『模型查看器面板』主题。

格式

```
<ModelProvider container="container_name" isPMML="true_false" />
```

其中:

container 是存放模型的容器的名称。

isPMML 指示是否以 PMML 格式存储模型。

示例

```
<ModelProvider container="model" isPMML="true" />
```

要查看在模型应用器节点的上下文中使用 ModelProvider 的示例，请参阅 第 110 页的『模型查看器面板』下的示例。

属性

属性定义包含一组由名称和值组成的数据对。单个属性定义（可能会有多个）分别包含在单独的属性部分中。

注意: 如果已在“属性”部分中定义某个属性，那么无需为个别属性控件定义该属性，这是因为“属性”部分中的定义的优先级较高。因此，我们建议在属性部分定义属性。

此规则的一个例外是与 label 属性相关的情况。如果已经为属性控件定义了 label 属性，那么在属性控件声明中发现的 *任意* 属性定义（不仅仅是 label 定义）比属性部分中的相应定义具有较高的优先级。请注意，此

例外情况仅适用于属性控件，而不适用于其他类型的控件，例如菜单、菜单项目和工具栏项目。这些项目必须明确定义一个标签，可通过 Action 元素直接定义（菜单）或间接定义（菜单项和工具栏项目）。

格式

```
<Properties>
     <Property name="name" scriptName="script_name" valueType="value_type" isList="true_false"
          defaultValue="default_value" label="display_label" labelKey="label_key"
          description="description" descriptionKey="description_key" />
          <Enumeration ... />
          <Structure ... />
          <DefaultValue ... />
     ...
</Properties>
```

在特定的情况下，还会使用 Enumeration、Structure 和 DefaultValue 元素。有关更多信息，请参阅第 57 页的『值类型』主题。

下表显示了 Property 元素的属性。

表 14. Property 属性.

属性	描述
name	（必需）属性的唯一名称。
scriptName	属性在脚本中所引用的名称。有关更多信息，请参阅第 71 页的『在脚本中使用 CLEF 节点』主题。
valueType	指定此属性可以采用的值类型，包括: string encryptedString fieldName integer double boolean date enum structure databaseConnection 有关更多信息，请参阅第 57 页的『值类型』主题。
isList	指定该属性是特定值类型的一列值 (true) 还是一个值 (false)。
defaultValue	此属性的缺省值。这可以表示为一个简单的值属性或一个复杂的元素，而且必须与指定的有效值一致。
label	要在用户界面上显示的属性值显示名称。
labelKey	用于标识标签以便进行本地化。
description	属性的说明。
descriptionKey	用于标识说明以便进行本地化。

属性可以选择声明如何确定有效值:

- 对于数字值，这将是最小值和/或最大值。

- 对于字符串，这通常是字段选择（例如所有字段、所有数字字段和所有离散字段等等），但也可能是文件选择。
- 对于枚举，这将是有效值的集合。

键控属性也必须声明如何确定有效键。请注意，键控属性的键类型必须是字符串或枚举。有关更多信息，请参阅第 35 页的『属性类型』主题。

创建属性的相关对象之后，将对与该属性相关联的可选缺省值进行求值。例如，每次创建新节点实例时，都会对节点属性的缺省值进行求值；每次执行节点时，都会对执行属性进行求值。求值按照属性的声明顺序执行。

请注意，属性定义可以引用“公共对象”部分中声明的属性类型。

容器

容器就是在“构造函数”部分定义其生成的输出对象的占位符。

格式

```
<Containers>
     <Container name="container_name" />
      ...
</Containers>
```

其中:

name 对应于 CreateModel 或 CreateDocument 元素的目标属性值（请参阅第 91 页的『使用构造函数』），而且间接地使容器与“公共对象”部分中声明的其中一种容器类型相关联。

示例

首先，在“公共对象”部分中声明容器类型。为模型声明了一种容器类型，即采用文本格式；为文档输出对象声明了两种容器类型，一种是对 HTML 输出采用的缺省（文本）格式，一种是对压缩的输出采用的二值格式。

```
<CommonObjects>
     <ContainerTypes>
          <ModelType id="my_model" format="utf8" />
          <DocumentType id="html_output" />
          <DocumentType id="zip_outputType" format="binary" />
     </ContainerTypes>
</CommonObjects>
```

在节点定义的“执行”部分，将输出文件定义为包含容器类型（对应于“公共对象”部分中指定的标识）的容器文件:

```
<Node id="mynode" ... >
 ...
     <Execution>
      ...
          <OutputFiles>
               <ContainerFile id="pmml" path="${tempfile}.pmml" containerType="my_model" />
               <ContainerFile id="htmloutput" path="${tempfile}.html" containerType="html_
               output" />
               <ContainerFile id="zipoutput" path="${tempfile}.zip" containerType="zip_
               outputType" />
          </OutputFiles>
```

此后，“构造函数”部分定义执行节点时要生成的输出对象。此处，CreateModel 和 CreateDocument 元素都具有与容器文件对应的 sourceFile 属性，正如之前在“输出文件”部分中指定的一样:

```
          <Constructors>
               <CreateModelOutput type="myoutput">
                    <CreateModel target="model" sourceFile="pmml" />
                    <CreateDocument target="advanced_output" sourceFile="htmloutput" />
                    <CreateDocument target="zip_output" sourceFile="zipoutput" />
               </CreateModelOutput>
          </Constructors>
     </Execution>
</Node>
```

最后，“模型输出”部分使容器与模型输出对象或文档输出对象相关联。在 Container 元素中，name 属性相当于之前指定的 CreateModel 和 CreateDocument 元素中的 target 属性:

```
<ModelOutput id="myoutput" label="My Model">
     <Containers>
          <Container name="model" />
          <Container name="advanced_output" />
          <Container name="zip_output" />
     </Containers>
      ...
</ModelOutput>
```

用户界面

规范文件支持多种用户界面组件，以便显示各种对象以及修改各种控件和属性。有多种工具可供用户使用，用于指定组件的布局和调整大小行为，以及如果修改了其他控件之后，是否应该启用组件或显示组件。

“用户界面”部分用于指定对象的可视外观。规范可用于定制基本的用户界面组件，例如节点属性对话框或输出窗口。

“用户界面”部分是 Node 元素规范的必需部分。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 95 页的第 6 章，『构建用户界面』。

执行

此元素仅在 Node 元素定义中使用。

“执行”部分用于定义执行节点时使用的属性和文件。

格式

```
<Execution>
     <Properties>
      ...
     </Properties>
     <InputFiles>
          <ContainerFile ... />
           ...
     </InputFiles>
     <OutputFiles>
          <ContainerFile ... />
           ...
     </OutputFiles>
     <Module ... >
          <StatusCodes ... />
     </Module>
     <Constructors ... />
</Execution>
```

“执行”部分包括每次执行节点时都会重新创建的一组属性的定义，这些属性仅在执行节点时可用。

执行信息也可以定义要在执行节点之前生成的输入文件集，以及在执行时生成的任意输出文件。

可以指定任意数目的输入文件和输出文件。每个输入文件都与节点所定义的容器相关联。通常，每个输出文件都用于为生成的对象构造容器。输入或输出文件的格式由“公共对象”部分的容器声明确定。

示例

要查看“执行”部分的示例，请参阅第 29 页的『规范文件的示例』。

属性（运行时）

此部分定义仅当执行节点时才可用的运行时属性集。

格式

其格式与元素定义主要部分中的 Properties 部分相似。有关更多信息，请参阅第 48 页的『属性』主题。

当执行模型构建器节点或文档构建器节点时，将会创建一个 **服务器临时文件**以存储模型输出或文档输出对象。服务器将访问此文件，将对象传输到客户端，然后在客户端将对象包装在容器中。您需要在此指定此文件。

示例

本例说明如何指定服务器临时文件。

```
<Properties>
     <Property name="tempfile" valueType="string">
          <DefaultValue>
               <ServerTempFile basename="datatmp"/>
          </DefaultValue>
     </Property>
</Properties>
```

输入文件

此部分定义要在执行节点前生成的输入文件集。在此上下文中，输入文件就是在服务器上执行节点时使用的输入文件。例如，执行节点时，模型应用器节点具有将传输到指定输入文件中的模型容器。

格式

```
<InputFiles>
     <ContainerFile id="identifier" path="path" container="container">
      ...
</InputFiles>
```

在输入文件的 ContainerFile 元素中，其属性如下表所示。

表 15. 容器文件属性 - 输入文件.

属性	描述
id	容器文件的唯一标识。
path	服务器上输入文件的生成位置（例如，服务器临时文件的位置 - 请参阅『属性（运行时）』）。
container	容器（用于容纳要作为输入文件发送到服务器的对象）的标识。

示例

```
<InputFiles>
     <ContainerFile id="pmml" path="${tempfile}.pmml" container="model"/>
</InputFiles>
```

输出文件

此部分指定在服务器上执行节点期间生成的输出文件。输出文件（例如，执行模型构建器节点或文档构建器节点的结果）将在执行后传输回客户端。

格式

```
<OutputFiles>
     <ContainerFile id="identifier" path="path" containerType="container">
      ...
</OutputFiles>
```

在 ContainerFile 元素中，其属性如下表所示。

表 16. 容器文件属性 - 输出文件.

属性	描述
id	容器文件的唯一标识。
path	要传输到客户端的对象在服务器上的位置（例如，服务器临时文件的位置 - 请参阅第 52 页的『属性（运行时）』）。
containerType	对象的容器类型的标识（即，模型类型或文档类型的标识），用于使对象以正确格式传输。有关更多信息，请参阅第 36 页的『容器类型』主题。

示例

```
<OutputFiles>
     <ContainerFile id="pmml" path="${tempfile}.pmml" containerType="mynode_model" />
     <ContainerFile id="htmloutput" path="${tempfile}.html" containerType="html_output" />
     <ContainerFile id="zipoutput" path="${tempfile}.zip" containerType="zip_outputType" />
</OutputFiles>
```

模块

本节指定执行节点时要用的服务器端共享库（例如，要加载到内存中的 DLL）。

格式

```
<Module libraryId="shared_library_identifier" name="node_name">
     <StatusCodes ... />
</Module>
```

其中:

libraryId 是在“资源”部分的 Shared Library 元素中声明的库的标识。有关更多信息，请参阅第 33 页的『共享库』主题。

如果库由多个节点共享，请使用 name 来标识要执行的特定节点。如果只有一个节点在使用该库，名称字段可以为空。

示例

```
<Module libraryId="mynode1" name="mynode">
     <StatusCodes>
          <StatusCode code="0" status="error" message="An exception occurred" />
```

```
        <StatusCode code="1" status="error" message="Error reading input data" />
         ...
    </StatusCodes>
</Module>
```

状态码

大多数程序都会执行某种错误检查操作并向用户显示任何必要的消息，通常返回整数以表明成功完成状态或其他状态。服务器端 API 可以在执行包含节点的流后返回状态码。有关更多信息，请参阅第 177 页的『状态详细信息文档』主题。

状态码部分使您可以将一条消息与某个特定的状态码关联，以便向用户显示。

格式

```
<StatusCodes>
     <StatusCode code="codenum" status="status" message="message_text"
       messageKey="message_key" />
      ...
</StatusCodes>
```

状态码属性如下表所示。

表 17. 状态码属性.

属性	描述
code	与消息关联的状态码（一个整数值）。
status	状态分类: success - 执行节点成功 warning - 执行节点时发出了警告 error - 执行节点不成功
message	当返回此状态码时显示的消息。
messageKey	用于标识消息以便进行本地化。

示例

在本例中，错误消息的文本包含在 StatusCode 元素中:

```
<StatusCodes>
     <StatusCode code="0" status="error" message="Cannot initialise a peer" />
     <StatusCode code="1" status="error" message="Error reading input data" />
     <StatusCode code="2" status="error" message="Internal Error" />
     <StatusCode code="3" status="error" message="Input Field Does Not Exist" />
</StatusCodes>
```

在执行时，如果服务器端 API 返回状态码 3，那么将向用户显示以下消息。

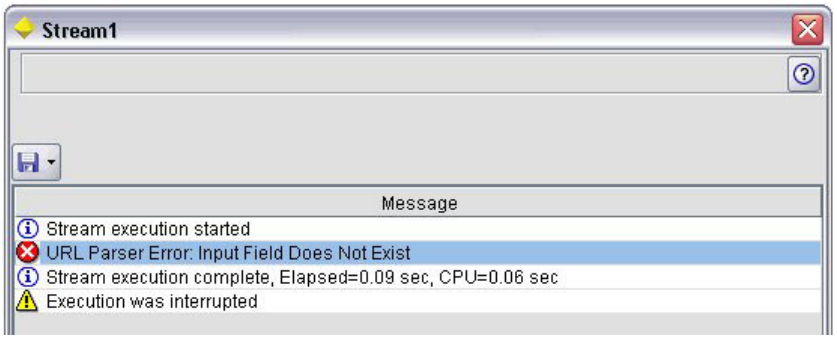

图 29. 显示的错误消息

在下一个示例中，错误消息的文本由 messageKey 属性引用:

```
<StatusCodes>
     <StatusCode code="0" status="error" messageKey="initErrMsg.LABEL"/>
     <StatusCode code="1" status="error" messageKey="inputErrMsg.LABEL"/>
     <StatusCode code="2" status="error" messageKey="internalErrMsg.LABEL"/>
     <StatusCode code="3" status="error" messageKey="invalidMetadataErrMsg.LABEL"/>
      ...
</StatusCodes>
```

属性文件（例如 messages.properties）位于规范文件所在的文件夹中，包含实际的消息文本以及其他显示文本:

```
 ...
initErrMsg.LABEL=Initialisation failed.
inputErrMsg.LABEL=Error when reading input data.
internalErrMsg.LABEL=Internal error.
invalidMetadataErrMsg.LABEL=Metadata (on input/output fields) not valid.
 ...
```

在需要对显示文本进行本地化以供海外市场使用时，此方法非常有用，这是因为所有要进行本地化的文本都可以在一个文件中找到。有关更多信息，请参阅第 153 页的『本地化』主题。

输出数据模型

此元素仅在 Node 元素定义中使用。

“输出数据模型”部分指定某些特定属性如何影响数据模型。

输出数据模型可以采用以下三种方式之一确定:

- 使用规范文件中的字段集定义功能。有关更多信息，请参阅第 65 页的『字段集』主题。
- 使用实现了数据模型提供者接口（此接口接收一组属性和输入数据模型，并返回数据模型实例）的客户端 Java 类。
- 使用接收一组属性和输入数据模型并返回元数据文档的服务器端共享库组件。

“输出数据模型”部分用于定义节点中的属性如何影响流经该节点的字段。输出数据模型可以:

- 保持输入数据模型不变
- 修改输入数据模型
- 使用另一个数据模型替换输入数据模型

例如，排序节点不会影响属性本身，但会对它们进行重新排序；导出节点可通过添加新字段修改数据模型；而汇总节点则完全替换数据模型。

在修改了输入数据模型的情况下，定义可以添加新字段或者修改或除去现有字段。替换数据模型时，只能添加新字段。规范文件支持这些基本操作（包括能够创建类型基于输入字段的新字段），并能够对输入字段集或者表示输入字段集中一组字段的键或列表属性进行迭代。

格式

“输出数据模型”部分的一般格式如下所示，但请参阅第 124 页的『多字段选择器控件』和第 131 页的『单字段选择器控件』章节以了解这些情况的特定格式。

```
<OutputDataModel mode="mode" libraryId="container_name">
     -- data model operations --
</OutputDataModel>
```

输出数据模型属性如下表所示。

表 18. 输出数据模型属性.

属性	描述
mode	有关数据模型的效果: extend - 对现有模型添加新字段 fixed - 不进行更改 modify - 更改现有字段（例如除去或重命名） replace - 替换现有模型
libraryId	从中获取数据模型的服务器端容器的名称。

数据模型操作是指添加新字段以及修改或除去现有字段之类的操作。有关更多信息，请参阅第 60 页的『数据模型操作』主题。

示例

OutputDataModel 元素包含在规范文件示例中。有关更多信息，请参阅第 29 页的『规范文件的示例』主题。

构造函数

构造函数用于定义在流中执行节点时或生成返回流的对象时产生的对象。

有关如何对文件的这一部分进行编码的全部详细信息，请参阅第 91 页的『使用构造函数』及后续章节。

公共特征

有些特征可以在规范文件的多个部分中使用，它们分别名为:

- 值类型
- 求值型字符串
- 操作
- 字段和字段元数据
- 字段集
- 角色
- 逻辑运算符

- 条件

值类型

值类型声明用于指定列、属性或属性类型规范可以采用的值类型。

字符串和加密字符串

`valueType="string"` 格式指定该值为文本字符串。如果用户输入的字段（例如密码字段）内容需要隐藏，`valueType="encryptedString"` 声明用于与这类字段相关的属性。

字段名

如果值以字段名的形式出现，请使用 `valueType="fieldName"` 格式。

数学、逻辑和日期表达式

如果值是一个数学（整数或双精度数）表达式、逻辑（true/false）表达式或日期表达式，请相应地将 `valueType` 设置为 `integer`、`double`、`boolean` 或 `date`。

枚举属性

枚举属性包含在 `valueType="enum"` 声明后紧跟的枚举部分中。有关更多信息，请参阅『枚举属性』主题。

结构声明

`valueType="structure"` 声明用于指定包含其他已知属性的复合值。属性与特性类似，但属性不支持结构化和键控。有关更多信息，请参阅第 58 页的『结构化属性』主题。

- **键控指示符**。指定属性是单个值还是散列表，对于散列表，表中的每个值都具有指定的值类型。
- **值集**。指定如何确定可用值的集合。
- **键集**。对于键控属性，这用于指定如何确定可用键的集合。此信息也用于向用户界面提供有关最适合使用的控制器类型的提示。

数据库连接

用户可以使用这些连接字符串（例如 `user1@testdb`）登录数据库。事先必须为数据库定义登录详细信息。有关更多信息，请参阅第 122 页的『数据库连接选择器控件』主题。

枚举属性

枚举属性就是可以从预定义的值列表中获取值的属性。

格式

枚举属性的格式使用 `Enumeration` 部分，并在其中定义值列表，如下所示:

```
<PropertyTypes>
     <PropertyType id="identifier" valueType="enum">
          <Enumeration>
               <Enum value="value" label="display_label" labelKey="label_key"
                    description="description" descriptionKey="description_key" />
           ...
          </Enumeration>
     </PropertyType>
<PropertyTypes>
```

其中，PropertyType 属性包括:

- id 是属性类型的唯一标识。
- valueType 指示这是枚举类型的属性。

而 Enum 属性包括:

- value（必需）是要在值列表中显示的属性值。
- label（必需）是要在用户界面上显示的属性值显示名称。
- labelKey 用于标识标签以便进行本地化。
- description 是对枚举值的描述。
- descriptionKey 用于标识描述以便进行本地化。

示例

```
<PropertyTypes>
     <PropertyType id="shared_enum1" valueType="enum">
          <Enumeration>
               <Enum value="value1" label="Value 5.1" labelKey="enum5.value1.LABEL" />
               <Enum value="value2" label="Value 5.2" labelKey="enum5.value2.LABEL" />
               <Enum value="value3" label="Value 5.3" labelKey="enum5.value3.LABEL" />
          </Enumeration>
     </PropertyType>
<PropertyTypes>
```

结构化属性

结构化属性就是在类似网格的结构（例如对话框中的表控件）中使用的属性。

格式

结构化属性的格式使用了一个定义结构的 Structure 部分，所以包含多个 Attribute 元素，如下所示:

```
<PropertyTypes>
     <PropertyType id="identifier" valueType="structure" isList="true_false">
          <Structure>
               <Attribute name="column_ID" valueType="value_type" isList="true_false"
                    label="column_label" labelKey="label_key" defaultValue="value"
                    description="description" descriptionKey="description_key" />
           ...
          </Structure>
     </PropertyType>
</PropertyTypes>
```

其中，PropertyType 元素属性包括:

- id 是属性类型的唯一标识。
- valueType 指示这是结构化属性。
- isList 指示该属性是具有指定值类型的值的列表 (true) 还是单个值 (false)。

而 Attribute 元素属性包括:

- name（必需）是列的标识。
- valueType 用于指定此列内容可以采用的值类型，这是下列其中一项:

  string

  encryptedString

integer

double

boolean

date

enum

- isList 指示该属性是具有指定值类型的值的列表 (true) 还是单个值 (false)。这样，键控属性就可以与固定的已知属性集（例如，表示要对特定字段执行的不同汇总操作的布尔属性）或值列表（例如，用于使字段名列表与另外某些字段名相关联的值列表）相关联。
- label（必需）是要在用户界面上显示的列显示名称。
- labelKey 用于标识标签以便进行本地化。
- defaultValue 是显示列时要在其中显示的值。
- description 是对列的描述。
- descriptionKey 用于标识描述以便进行本地化。

示例 - 表控件

要查看如何在表控件中使用结构化属性的示例，请参阅第 133 页的『表控件』。

示例 - 键控属性类型

这些示例中的第一个示例说明了键控属性类型的使用，其中每个关联值为表示从固定操作集应用到某个字段的汇总操作的结构:

```
<PropertyType id="aggregateOps" isKeyed="true" valueType="structure">
  <Structure>
    <Attribute name="MIN" valueType="boolean" label="Min" />
    <Attribute name="MAX" valueType="boolean" label="Max" defaultValue="true"/>
    <Attribute name="SUM" valueType="boolean" label="Sum" defaultValue="false"/>
    <Attribute name="MEAN" valueType="boolean" label="Mean" defaultValue="false"/>
    <Attribute name="SDEV" valueType="boolean" label="SDev" defaultValue="false"/>
  </Structure>
</PropertyType>
```

因此，被声明使用 aggregateOps 属性类型的属性可以是:

```
<Property name="aggregationSettings" scriptName="aggregation_settings" type="aggregateOps"/>
```

这里，属性包含多个值，并且每个值都具有不同的键。例如，键 name 是字段名称（MIN 和 MAX 等等）。

在接下来的键控属性类型示例中，每个相关联的值都是包含单个属性的结构。在本例中，属性为表示应用到某个字段的乘数的双精度表达式列表:

```
<PropertyType id="multiplierOps" isKeyed="true" valueType="structure">
  <Structure>
    <Attribute name="multipliers" valueType="double" isList="true"/>
  </Structure>
</PropertyType>
```

被声明使用 multiplierOps 属性类型的属性可以是:

```
<Property name="multiplierSettings" scriptName="multiplier_settings" type="multiplierOps"/>
```

缺省值

DefaultValue 元素用于指定服务器临时目录、文件或二者。它们的作用是存储模型输出对象或文档输出对象。

格式

```
<DefaultValue>
     <ServerTempDir basename="name"/>
     <ServerTempFile basename="name"/>
</DefaultValue>
```

其中，basename（必需）是临时目录或文件的名称。

示例

```
<DefaultValue>
     <ServerTempFile basename="datatmp"/>
</DefaultValue>
```

求值型字符串

在规范文件中声明的某些字符串可能会包含对属性名称的引用。这些字符串称为求值型字符串。

属性引用的语法为:

"${*property_name*}"

访问求值型字符串时，所有属性引用都将替换为所引用属性的值。如果该属性不存在，那么将发生错误。例如，当添加新字段时，节点定义中可能会有一个名为 my_new_field 的属性，而“用户界面”部分有一个控件，用户可以通过它来编辑这个属性的值。

示例

```
<AddField name="${my_new_field}" ... >
```

操作

规范文件的某些特定部分可支持多种不同的操作，例如添加字段、创建组件以及初始化属性。支持操作的部分包括:

- 输出数据模型（源节点和过程节点）
- 输入和输出数据模型（组件）
- 输出对象创建（模型构建器节点和文档构建器节点）
- 模型应用器创建（模型输出）

操作分为以下这几种类型:

- 数据模型操作: AddField、ChangeField 和 RemoveField
- 迭代: ForEach

数据模型操作

您可以在数据模型中执行的操作包括:

- 在现有数据模型中添加新字段
- 修改数据模型中的现有字段
- 删除数据模型中的字段

添加字段: AddField 元素可用于在现有数据模型中添加新字段。

格式

```
<AddField prefix="prefix" name="name" direction="field_role" directionRef="field_role_ref"
     fieldRef="field_ref" group="group_id" label="label" missingValuesRef="mval_ref"
     storage="storage_type" storageRef="storage_ref" targetField="target_field"
     type="data_type" typeRef="type_ref" role="role" tag="propensity_type" value="value"
depth="integer" valueStorage="storage_type">
     <Range min="min_value" max="max_value" />
</AddField>
```

AddField 的属性如下所示。

表 19. AddField 属性.

属性	描述
prefix	要对字段名称添加的前缀，例如用于表示模型输出字段。
name	（必需）要添加的字段的名称。这可以是硬编码字符串（例如 field8）或引用了字段的求值型字符串（例如 ${target}）。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
direction	字段的角色，例如字段是输入字段还是目标字段。包括 in、out、both、partition 和 none。
directionRef	指定字段方向是从引用了某个字段的求值型字符串（例如，${field1}）所确定的字段方向派生。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
fieldRef	指定所有引用值（directionRef、missingValuesRef、storageRef 和 typeRef）源自引用了字段的求值型字符串（例如，${field1}）所确定字段的对应值。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
group	指定字段是字段组的成员。有关更多信息，请参阅第 79 页的『模型字段』主题。
label	要添加的字段的标签。
missingValuesRef	指定缺失值处理方式源自由引用了字段的求值型字符串（例如 ${field1}）所确定字段的缺失值指定方式。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
storage	字段值的数据存储类型，包括 integer、real、string、date、time、timestamp、list 或 unknown。
storageRef	指定存储类型是从引用了某个字段的求值型字符串（例如，${field1}）所确定的字段存储类型派生。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
targetField	对于模型输出字段，指定从中派生新字段数据的目标字段。这可以是硬编码字符串（例如 field8）或引用了字段的求值型字符串（例如 ${target}）。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
type	字段的数据类型，包括 auto、range、discrete、set、orderedSet、flag、collection、geospatial 或 typeless。
typeRef	指定数据类型是从引用了某个字段的求值型字符串（例如，${field1}）所确定的字段数据类型派生。有关更多信息，请参阅第 60 页的『求值型字符串』主题。

表 19. AddField 属性 (续).

属性	描述
role	模型输出字段中保存的数据类型，包括 unknown、predictedValue、predictedDisplayValue、probability、residual、standardError、entityId、entityAffinity、upperConfidenceLimit、lowerConfidenceLimit、propensity、value 和 supplementary。有关更多信息，请参阅第 66 页的『角色』主题。
tag	仅当 role 的值为 propensity 时使用；表示倾向类型，可以是 RAW 或 ADJUSTED。
value	指定新字段包含的值要从引用了某个字段的求值型字符串（例如 ${field1}）所标识的字段派生。有关更多信息，请参阅第 60 页的『求值型字符串』主题。
depth	仅在 storage 值为 list 时使用，这定义必须 >= -1 的列表深度
valueStorage	仅在 storage 值为 list 时使用，这定义列表值的存储类型，包括 integer、real、string、date、time 或 timestamp。

Range 属性如下所示。

表 20. 范围属性

属性	描述
最小	字段可以接受的最小值。
最大	字段可以接受的最大值。

示例

以下示例添加一个名为 field8 的字符串字段:

```
<AddField name="field8" storage="string" />
```

下一个示例演示添加字段时如何使用对属性名称的引用。此处为字段添加的名称与先前定义的属性 prop1 的值相匹配:

```
<AddField name="${prop1}" ... />
```

在下面的示例中，如果目标字段名为 field1，模型将创建一个名为 $S-field1 的输出字段来存储 field1 的预测值:

```
<AddField prefix="$S" name="${target}" role="predictedValue" targetField="${target}"/>
```

下一个示例添加的模型输出字段用于存储介于 0.0 和 1.0 之间的概率评分:

```
<AddField prefix="$SC" name="${target}" storage="real" role="probability" targetField=
"${target}">
     <Range min="0.0" max="1.0"/>
</AddField>
```

在最后的示例中，为每个模型输出字段添加一个输出字段，用于存储介于 0.0 和 1.0 之间的概率评分，其值源自变量 fieldValue 的值:

```
<ForEach var="fieldValue" inFieldValues="${field}">
     <AddField prefix="$SP" name="${fieldValue}" storage="real" role="probability" targetField=
     "${field}" value="${fieldValue}">
          <Range min="0.0" max="1.0"/>
     </AddField>
</ForEach>
```

有关更多信息，请参阅第 60 页的『求值型字符串』主题。

更改字段: ChangeField 元素可用于修改数据模型中的现有字段。

格式

```
<ChangeField name="name" fieldRef="field_reference" direction="field_role" storage="storage_
type" type="data_type" >
     <Range min="min_value" max="max_value" />
</ChangeField>
```

ChangeField 的属性如下所示。

表 21. ChangeField 属性.

属性	描述
name	（必需）要更改的字段的名称。
fieldRef	字段的引用值。
direction	字段的角色，例如字段是输入字段还是目标字段。包括 in、out、both、partition 和 none。
storage	字段值的数据存储类型，包括 integer、real、string、date、time、timestamp 和 unknown。
type	字段的数据类型，包括 auto、range、discrete、set、orderedSet、flag 和 typeless。

Range 属性如下所示。

表 22. 范围属性

属性	描述
最小	字段可以接受的最小值。
最大	字段可以接受的最大值。

删除字段: RemoveField 元素可用于删除数据模型中的字段。

格式

```
<RemoveField fieldRef="field_reference" />
```

其中，fieldRef 是字段的参考值。

使用 ForEach 元素执行迭代

在某些位置，采用重复执行同一操作的方式处理一组值中的每个值非常有用。规范文件支持一种简单的 ForEach 迭代器，它会按顺序将一个临时属性与所提供数据组中的每个值绑定。可以将 ForEach 循环设置为以下几种迭代方式之一:

- 在两个具有可选步长的整数值之间
- 在列表属性的值之间

- 在键控属性的键之间
- 在字段组的字段之间

格式

```
<ForEach var="field_name" from="integer_exp" to="integer_exp" step="integer_exp"
inFields="fields" inFieldValues="field_name" inProperty="property_name" >
   -- data model operation --
</ForEach>
```

其中:

var（必需）指定一个字段，其中包含要对其执行迭代的值。

from 和 to 指定整数（或者求值结果为整数的表达式），分别表示迭代的下限和上限，同时使用可选属性 step 指示整数步长。

inFields、inFieldValues 和 inProperty 分别替代 from/to/step 格式:

- inFields 指定要对其执行迭代的字段集，这是下列其中一项:

  inputs - 节点的输入字段

  outputs - 节点的输出字段

  modelInput - 在模型签名中指定的输入字段

  modelOutput - 在模型签名中指定的输出字段
- inFieldValues 指定一个字段名（或者表示字段名的属性），并对该字段的元数据中的各个值执行迭代
- inProperty 指定要对其执行迭代的属性的名称

可以在 ForEach 元素中指定的数据模型操作是 AddField、ChangeField 或 RemoveField 元素之一。有关更多信息，请参阅第 60 页的『数据模型操作』主题。ForEach 元素也可以是嵌套元素。

示例

以下示例将执行一个操作十次:

```
<ForEach var="val" from="1" to="10">
 ...
</ForEach>
```

以下示例将按整数属性指定的次数执行操作:

```
<ForEach var="val" from="1" to="${history_count}">
 ...
</ForEach>
```

在下一个示例中，将对节点的输出字段中的所有值执行迭代:

```
<ForEach var="field" inFields="outputs">
 ...
</ForEach>
```

下面的示例指定字段为由 ${field} 标识的字段，并对该字段元数据中的所有值执行迭代:

```
<ForEach var="fieldValue" inFieldValues="${field}">
 ...
</ForEach>
```

下一个示例将对列表属性中的所有值执行迭代:

```
<ForEach var="val" inProperty="my_list_property">
 ...
</ForEach>
```

下面的示例将对键控属性中的所有键值执行迭代:

```
<ForEach var="key" inProperty="my_keyed_property">
 ...
</ForEach>
```

字段和字段元数据

节点、模型和数据源作为 **数据模型提供者**，它们可以定义可从其他对象访问的字段元数据。

数据模型提供者具有一个输入数据模型和一个输出数据模型。输出数据模型可以按照输入数据模型的方式定义，例如当通过添加字段扩展输入模型时，或者当更改现有模型时。

这些对象各自的要求稍有不同。

节点。您可以引用输入数据模型，但无法对其进行修改。输出数据模型可以基于输入数据模型，也可以将其替换。每当节点属性或输入数据模型发生更改时，都将重新计算输出数据模型。模型应用器节点的输出数据模型也可以引用模型组件的输出数据模型。

模型。缺省情况下，输入及输出数据模型（模型签名）均基于创建模型时使用的输入及输出字段设置。理想情况下，模型构建过程将返回一个元数据文件，该文件定义了必需的输入字段以及生成的输出字段。模型签名一旦定义便不可修改。但是，使用模型应用器节点中的属性可以修改来自应用器节点的数据模型输出。例如，这些属性可以定义聚类标识是作为字符串还是整数返回，或者定义要生成的序列标识的数目。另外，模型签名通常将输出的字段角色（方向）指定为"out"，而节点却似乎以字段角色"in"生成它们。

数据源。在数据阅读器节点中使用的数据源可以指定输出数据模型。输入数据模型始终为空。

字段集

可以在许多位置使用字段集，以便从数据模型提供者中选择字段子集。数据模型提供者既可以是宿主类对象，也可以是宿主类对象的容器。字段过滤器的初始状态既可以先包含所有可用字段，然后排除特定类型的字段；也可以先为空的字段集，然后再包含所需字段或添加新字段。

下面的示例演示扩展节点如何指定输出数据模型。键字段由一个名为 keys 的列表属性指定，该列表属性后跟随一个可选的可生成记录计数字段，该字段的名称也是由属性指定。

```
<OutputDataModel mode="replace">
     <ForEach var="field" inProperty="keys">
          <AddField name="${field}" fieldRef="${field}"/>
     </ForEach>
     <AddField name="${record_count_name}" storage="integer">
          <Condition property="include_record_count" op="equals" value="true"/>
     </AddField>
</OutputDataModel>
```

字段集和模型构建

下面的示例将演示模型应用器如何使用先前创建的模型组件中的信息生成自己的输出字段:

```
<OutputDataModel mode="modify">
     <AddField provider="model" dataModel="output">
</OutputDataModel>
```

AddField 和 ForEach 均用于指定数据模型提供者，同时还可以指定应该使用哪些输入或输出数据模型。它们提供了一种用于指定来自数据模型提供者的字段集（或子集）的机制。缺省提供者为表示外层元素（即，并非要创建的对象）的 this，并且缺省情况下使用输入字段集。如果未指定字段集，那么将使用所有的可用字段。

字段集可以基于存储、类型、字段角色或名称。如果字段集基于名称，那么需要引用列表属性。字段集可以为满（缺省）或为空；前者允许排除某些字段，而后者允许将某些字段包括在内。可以为每个单独的过滤器指定多个值，这些值将作为“intersection”或“and”运算符使用，例如:

```
<FieldSet include="none"> <Include direction="in" storage="string"/> </FieldSet>
```

此示例由空字段集开始（由 include="none" 指定），然后将包含字段角色（方向）为"in"以及存储类型为字符串的字段。

又例如:

```
<FieldSet include="all"> <Exclude type="typeless"/> </FieldSet>
```

此示例包含所有可用字段（由 include="all" 属性指定，这是缺省的行为），然后排除任何具有 typeless 类型的字段。这将包含方向设为 "in" 或 "both" 的字段。

也可指定多个过滤器，它们将作为“union”或“or”运算符使用，例如:

```
<FieldSet include="all">
     <Exclude type="discrete" storage="real"/>
     <Exclude type="discrete" storage="integer"/>
</FieldSet>
```

此示例排除那些具有离散实数存储或离散整数存储的字段。

请注意，当在初始为空的字段集中放入字段时，include 语句的顺序通常不会影响字段的放入顺序。也就是说，对于来自字段集提供者的字段，将以它们的自然顺序针对每个条件进行计算，以确定是否应将它们放入字段集中。

角色

角色用于描述数据模型输出字段中存储的数据的类型。角色可以由 AddField 元素指定，并由 Condition 元素进行测试。

可能的角色如下所示。

表 23. 模型输出角色

角色	含义
未知	未指定角色（不应发生）。
predictedValue	此字段包含目标字段的预测值。
概率	预测的概率或置信度。
残差	残差值。
standardError	预测的标准误差差。
entityId	实体标识，通常表示聚类模型中的聚类标识。
entityAffinity	实体仿射性，通常表示与模型聚类中心的距离。
upperConfidenceLimit	预测的置信度上限。
lowerConfidenceLimit	预测的置信度下限。
propensity	倾向评分。另一个 tag 属性指定这是指原始倾向还是指调整倾向。

表 23. 模型输出角色 (续)

角色	含义
value	用于表示与另一个输出相关联的模型的值（请参阅下文）。
supplementary	由其他角色不涉及的模型生成的信息。

作为 value 的示例，异常检测模型生成了多组字段，其中每组包含两个字段，一个表示字段名称，另一个用于指定衡量该字段异常程度的尺度。在本例中，value 将为字段名。

逻辑运算符

许多元素可以使用逻辑运算符 And、Or 和 Not 来指定各种类型的处理，例如在设置复合条件时执行此操作（请参阅第 70 页的『复合条件』）。

格式

And 元素的格式如下所示。Or 和 Not 元素的格式几乎相同，唯一区别在于分别使用外围标记 <Or >... </Or > 和 <Not >... </Not >。

```
<And>
     <Condition .../>
     <And ... />
     <Or ... />
     <Not ... />
</And>
```

Condition 元素用于指定要进行测试的条件。有关更多信息，请参阅『条件』主题。

请注意，And、Or 和 Not 子元素可以是嵌套元素。

条件

某些对象的行为可以通过使用由 Condition 元素（相当于 IF 语句）指定的条件进行修改。例如，由命令执行的节点可以对执行信息添加条件，比如在属性具有特定值时仅包括特定选项。同样，用户界面中的某个属性控件可能仅在另一个控件具有特定值时才会启用或可视。

这些条件既可以是简单条件，也可以是复合条件。**简单条件**由下列部分组成:

- 值源（可以是属性或控件）
- 测试
- 可选的测试值

复合条件允许其他条件进行组合，从而形成复杂的逻辑条件。复合条件可以使用:

- And
- Or
- Not

格式

```
<Condition container="container_name" control="prop_name" property="name" op="operator"
     value="value" />
```

其中:

container 用于指定值要由条件进行测试的特定容器的名称。

control 用于指定值要由条件进行测试的属性控件。*prop_name* 是在其中定义该控件（例如，在某个对话框选项卡的属性面板中）的元素的 property 属性值。

property 用于指定值要由条件进行测试的属性。*name* 是在其中定义此特性的 Property 元素的 name 属性值。

op 是条件运算符。有关更多信息，请参阅『条件运算符』主题。

value 是要由条件进行测试的特定值。

示例

有关条件设置的示例，请参阅第 70 页的『简单条件』和第 70 页的『复合条件』。

条件运算符

一组可用于满足大多数条件的运算符。

表 24. 任意值支持的测试

运算符	值	描述
equals	*value*	如果属性等于所提供的值，那么为真（字符串值需要区分大小写）。
notEquals	*value*	如果属性不等于所提供的值，那么为真（字符串值需要区分大小写）。
in	*值列表*	如果属性在所提供的值列表中，那么为真。

表 25. 数字值支持的测试

运算符	值	描述
lessThan	*number*	如果属性小于所提供的数字，那么为真。
lessOrEquals	*number*	如果属性小于或等于所提供的数字，那么为真。
greaterThan	*number*	如果属性大于所提供的数字，那么为真。
greaterOrEquals	*number*	如果属性大于或等于所提供的数字，那么为真。

表 26. 字符串值支持的测试

运算符	值	描述
isEmpty	-	如果属性具有长度为零的字符串，那么为真。
isNotEmpty	-	如果属性具有长度非零的字符串，那么为真。
startsWith	*string*	如果属性由所提供的字符串开头，那么为真（区分大小写）。
startsWithIgnoreCase	*string*	如果属性由所提供的字符串开头，那么为真，不区分大小写。
endsWith	*string*	如果属性由所提供的字符串结尾，那么为真（区分大小写）。
endsWithIgnoreCase	*string*	如果属性由所提供的字符串结尾，那么为真，不区分大小写。
equalsIgnoreCase	*string*	如果属性等于所提供的字符串，那么为真，不区分大小写。

表 26. 字符串值支持的测试 (续)

运算符	值	描述
hasSubstring	*string*	如果属性包含所提供的字符串，那么为真（区分大小写）。
hasSubstringIgnoreCase	*string*	如果属性包含所提供的字符串，那么为真，不区分大小写。
isSubstring	*string*	如果属性是所提供字符串的子字符串，那么为真（区分大小写）。
isSubstringIgnoreCase	*string*	如果属性是所提供字符串的子字符串，那么为真，不区分大小写。

表 27. 列表属性支持的测试

运算符	值	描述
isEmpty	-	如果列表中的项目数为 0，那么为真。
isNotEmpty	-	如果列表中的项目数不为 0，那么为真。
countEquals	*number*	如果列表中的项目数等于所提供的值，那么为真。
countLessThan	*number*	如果列表中的项目数小于所提供的值，那么为真。
countLessOrEquals	*number*	如果列表中的项目数小于或等于所提供的数字，那么为真。
countGreaterThan	*number*	如果列表中的项目数大于所提供的数字，那么为真。
countGreaterOrEquals	*number*	如果列表中的项目数大于或等于所提供的数字，那么为真。
contains	*value*	如果列表中含有所提供的项目，那么为真。

表 28. 字段属性支持的测试

运算符	描述
storageEquals	如果存储等于所提供的值，那么为真。
typeEquals	如果类型等于所提供的值，那么为真。
directionEquals	如果字段角色（方向）等于所提供的值，那么为真。
isMeasureDiscrete	如果字段数据类型为 discrete（即只能是 set、flag 或 orderedSet 中的一种），那么为真。
isMeasureContinuous	如果字段数据类型为 range，那么为真。
isMeasureTypeless	如果字段数据类型为 typeless，那么为真。
isMeasureUnknown	如果字段数据类型为 unknown，那么为真。
isStorageString	如果字段存储类型为 string，那么为真。
isStorageNumeric	如果字段存储类型为 numeric，那么为真。
isStorageDatetime	如果字段存储类型为 datetime，那么为真。
isStorageUnknown	如果字段存储类型为 unknown，那么为真。
isModelOutput	如果字段为模型输出字段，那么为真。
modelOutputRoleEquals	如果字段的角色为下表中显示的有效角色之一，那么为真。
modelOutputTargetFieldEquals	如果目标字段等于指定值（字符串），那么为真。
modelOutputHasValue	如果字段为模型输出字段并且具有相关值，那么为真。
modelOutputTagEquals	如果标记等于指定值（字符串），那么为真。

支持键控属性的条件运算符包括:

- isEmpty
- isNotEmpty
- countEquals
- countLessThan
- countLessOrEquals
- countGreaterThan
- countGreaterOrEquals
- contains

简单条件

简单条件包含要测试的初始值源（可以是属性或控制器名称，或者是求值表达式）、要执行的测试以及执行测试时要对比的值（可选）。

示例

如果名为 values_grouped 的布尔值属性为真，那么下列示例的值也将为真:

```
<Condition property="values_grouped" op="equals" value="true"/>
```

在下面的示例中，如果已选中显示布尔值的 values_grouped 控件，那么值为真:

```
<Condition control="values_grouped" op="equals" value="true"/>
```

在下面的示例中，如果名为 plot_fields 的列表属性包含至少一个值，那么值为真:

```
<Condition property="plot_fields" op="countGreaterThan" value="0"/>
```

在下面的示例中，如果名为 input_fields 的列表属性包含的值均已实例化，那么值为真:

```
<Condition property="input_fields" op="instantiated" listMode="all"/>
```

在最后这个示例中，如果名为 input_fields 的列表属性至少具有一个表示非实例化字段的值，那么值为真:

```
<Condition property="input_fields" op="uninstantiated" listMode="any "/>
```

复合条件

可以使用逻辑运算符将多组简单条件组合在一起。

示例

在下面的示例中，如果布尔值的 values_grouped 属性为真并且 group_fields 至少包含一个值，那么值为真:

```
<And>
     <Condition property="values_grouped" op="equals" value="true"/>
     <Condition property="group_fields" op="countGreaterThan" value="0"/>
</And>
```

在下面的示例中，如果布尔值的 values_grouped 属性为真或者如果 group_fields 至少包含一个值，那么值为真:

```
<Or>
     <Condition property="values_grouped" op="equals" value="true"/>
     <Condition property="group_fields" op="countGreaterThan" value="0"/>
</Or>
```

在下面的示例中，如果 group_fields 至少包含一个值，那么值为真:

```
<Not>
     <Condition property="group_fields" op="equals" value="0"/>
</Not>
```

对复合条件进行嵌套可以提供任意条件组合。

在脚本中使用 CLEF 节点

可以通过 Node 元素的 scriptName 属性，在脚本中引用 CLEF 节点。同样，通过 Property 元素的 scriptName 属性，可以在脚本中引用节点的属性。

在这两种情况下，scriptName 属性是可选的，不过我们建议使用此属性，以避免扩展或属性间发生名称冲突。

如果在节点定义中略去脚本名称，那么脚本可以通过以扩展名称为前缀的 id 属性的值来引用节点。例如，假定有一个名为 myext 的扩展，它使用标识 import 定义数据阅读器节点，脚本就可以使用 myextimport 引用此节点。

如果在属性定义中略去脚本名称，那么脚本可以通过其 name 属性的值来引用属性。

有关更多信息，请参阅《IBM SPSS Modeler 脚本编制和自动化指南》。

示例 - 编辑和执行节点

以下示例演示如何使用脚本来自动完成编辑和执行第 24 页的『数据阅读器节点（Apache 日志阅读器）』所示的示例数据阅读器节点的任务。

在“Apache 日志阅读器”节点的规范文件中，节点规范的开头内容为:

```
<Node id="apachelogreader" type="dataReader" palette="import" labelKey="apacheLogReader.LABEL">
     <Properties>
          <Property name="log_filename" valueType="string" labelKey="logfileName.LABEL" />
     </Properties>
```

在脚本中，可以像下面这样引用节点和属性:

```
create apachelogreader
set :apachelogreader.log_filename='installation_directory\Demos\combined_log_format.txt'
create tablenode at 200 100
connect :apachelogreader to :tablenode
execute :tablenode
```

其中，*installation_directory* 是 IBM SPSS Modeler 的安装目录。

运行此脚本:

- 创建数据阅读器节点
- 指定 *combined_log_format.txt* 作为要读取的 Apache 日志文件
- 创建“表”节点
- 将数据阅读器节点连接到“表”节点
- 执行“表”节点

示例 - 键控属性

键控属性支持标准脚本编写语法。例如，第 58 页的『结构化属性』中的第一个键控属性类型示例所示的结构在脚本中可以定义如下:

```
set :mynode.aggregation_settings.Age = {true true false false false}
```

单个属性可以修改如下:

```
set :mynode.aggregation_settings.Age.MIN = true
```

保持向后兼容性

当对现有扩展计划更新时，请注意保持与先前发布版本的扩展兼容。有些更改将不会产生不利的影响，有些更改涉及极大的风险，还有一些更改则会破坏兼容性，应避免此类更改。

无风险的更改

下列更改将不会影响向后兼容性:

- 添加新的 Node、ModelOutput、DocumentOutput 或 InteractiveModelBuilder 元素
- 对这些元素添加新的 Property 定义及其相关的新控件
- 对这些元素添加新容器*
- 对现有枚举属性添加新值

*请注意，任何使用这些新容器的代码都应允许以下事实，即对于使用早期版本的扩展创建的对象，这些容器将为空。

包含极大风险的更改

对现有声明进行更改意味着破坏兼容性这一巨大风险。这些更改应进行仔细测试后再发布。

需要避免的更改

下面这些更改是已知的会破坏兼容性的更改，应该避免:

- 更改 ExtensionDetail 元素的 id 属性或 providerTag 属性的值
- 更改 Node、ModelOutput、DocumentOutput 或 InteractiveModelBuilder 元素的 id 属性值
- 从扩展中删除 Node、ModelOutput、DocumentOutput 或 InteractiveModelBuilder 元素
- 更改 Property 或 PropertyType 元素的 valueType 属性值

第 5 章 构建模型和文档

模型和文档构建简介

标准的 IBM SPSS Modeler 模块包括允许用户生成（或“构建”）多种模型和图形的节点。通过 CLEF，您可以定义其他节点以构建未作为标准提供的其他模型和文档（图形和报告）。

在定义模型构建器或文档构建器节点时，还需要定义执行这些节点时生成的对象。您可以通过名为“构造函数”的项来执行此操作。

以下几节将详细介绍此过程。

模型

模型是一组规则、一个公式或一个方程，可用于根据输入字段集预测结果。预测输出结果的能力是预测性分析的核心目标。在 IBM SPSS Modeler 中，您可以通过以下方式实现此目标:

- 根据现有数据生成模型
- 将生成的模型应用到数据中以执行预测

生成模型的过程也称为“构建”模型，在 IBM SPSS Modeler 中，您可以通过建模节点来实现此过程。在 CLEF 中，建模节点被称为 **模型构建器节点**，该名称是从用于定义这些节点的 XML 语句的语法中派生而来的。有关更多信息，请参阅第 11 页的『模型构建器节点』主题。

将模型应用到数据的过程称为“评分数据”。这样，您便可以使用模型构建所获得的信息为新记录进行预测。在 IBM SPSS Modeler 中，可以通过将生成的模型图标添加到流工作区中实现此操作。图标采用金色块状形式，因此在 IBM SPSS Modeler 中所生成的模型称为“模型块”。在 CLEF 中，管理器窗格“模型”选项卡上的模型块称为**模型输出对象**，当将它添加到工作区中时，称为**模型应用器节点**。有关更多信息，请参阅第 12 页的『模型应用器节点』主题。

文档

在有些情况下，您可能希望生成对象而不是模型，例如图形或报告输出。在 IBM SPSS Modeler 中，我们将这些对象称为 **文档**，它们可通过 **文档构建器节点**生成。有关更多信息，请参阅第 11 页的『文档构建器节点』主题。

构造函数

构造函数用于定义在流中执行节点时或生成返回流的对象时产生的对象。

可以为以下任意内容定义构造函数:

- 模型构建器节点
- 文档构建器节点
- 模型应用器节点
- 模型输出对象

对于 **模型构建器**或 **文档构建器**节点，构造函数使这些节点能够定义在执行节点时生成输出对象的方式。输出对象定义可以包含多个属性和组成部分，而“构造函数”部分定义如何初始化或通过执行生成的对象创建这些属性和组成部分。

对于**模型应用器**节点，构造函数定义节点可以生成回流或“模型”选项卡的对象的类型。

对于为 **模型输出对象**定义的构造函数，它们可以:

- 指定在将模型输出对象拖放到流工作区时创建的模型应用器节点
- 通过用于创建模型输出对象的设置生成模型构建器节点

有关更多信息，请参阅第 91 页的『使用构造函数』主题。

构建模型

指定可以从中生成模型的节点（即，模型构建器节点）时，需要定义节点与 IBM SPSS Modeler 的模型构建器组件通信的方式，这就是实际创建模型的过程。您可以在规范文件的 Node 元素的定义中执行此操作。

除非另行指定，否则只要最终用户单击“模型构建器节点”对话框的**执行**按钮便开始构建模型。但是，也可以定义 **交互式模型**，通过它最终用户可以在单击 **执行**之后在实际构建模型之前精练或修改数据值。另外，交互式模型构建还要求包括用于定义交互的特定元素。有关更多信息，请参阅第 81 页的『构建交互式模型』主题。

定义模型构建器节点时，Node 元素必须包含:

- type="modelBuilder" 属性
- ModelBuilder 子元素
- 一个 Constructors 子元素，其中包含 CreateModelOutput 元素（请参阅第 91 页的『使用构造函数』）

要了解 Node 元素规范的格式，请参阅第 45 页的『节点』。

注意: 在后续章节的元素定义（通常由标题**格式**标识）中，除非指示为“（必需）”，否则元素属性和子元素均为可选项。要了解完整的元素语法，请参阅第 189 页的『CLEF XML 模式』。

对于模型构建，此扩展还需要 ModelOutput 元素，用于描述生成的模型（请参阅第 80 页的『模型输出』）。ModelOutput 元素需要包含 Constructors 子元素，后者包含 CreateModelApplier 定义。有关更多信息，请参阅第 94 页的『创建模型应用器』主题。

模型构建器

ModelBuilder 元素定义模型构建器节点的行为。可通过元素属性和一个或多个子元素来实现此操作。

格式

```
<ModelBuilder allowNoInputs="true_false" allowNoOutputs="true_false" nullifyBlanks="true_false"
     miningFunctions="[function1 function2 ... ]" >
     <Algorithm ... />
     <ModelingFields ... />
     <ModelGeneration ... />
     <ModelFields ... />
     <AutoModeling ... />
</ModelBuilder>
```

其中:

- 如果您想构建一个没有输入字段或没有输出字段的模型，那么必须分别显式地使用 allowNoInputs 和 allowNoOutputs。
- nullifyBlanks，如果设置为 false，那么在传递到 IBM SPSS Modeler 的模型构建器组件的数据中，将关闭用 Null 值（由 $null$ 表示）替换空白值的功能。缺省情况是使用 Null 来替换空白，但您可能希望取消激活此功能，例如，如果您的算法需要区别对待空白和 Null。

- miningFunctions（必需）标识模型所执行的一个或多个数据挖掘函数。

表 29. 数据挖掘函数.

函数	描述
classification	预测具有已知目标值的记录的未知目标属性的离散（即，set、flag 或 orderedSet 数据类型）值。
approximation	预测具有已知目标值的记录的未知目标属性的连续（即，range 数据类型）值。
clustering	标识相似记录的组并相应地进行标记。
association	标识数据中相关的事件或属性。
sequence	在按时间建立结构的数据中搜索顺序模式。
reduction	降低数据的复杂性，例如，通过对原始数据字段内容进行汇总的派生字段完成此任务。
conceptExtraction	在文本挖掘中使用。
categorize	在文本挖掘中使用。
timeSeries	根据过去数据中的模式预测未来值。
anomalyDetection	根据与聚类组正常值的偏差搜索异常值。
attributeImportance	标识对目标属性影响最大的属性。
supervisedMultiTarget	为多个概率中的某个概率估计一个（*yes* 或 *no*）结果的似然。

如果模型执行多个功能，那么在方括号中用空格分隔功能名称，如下例所示:

```
<ModelBuilder miningFunctions="[classification approximation]">
 ...
</ModelBuilder>
```

子元素

下表显示 ModelBuilder 元素的子元素。

表 30. 模型构建器声明的子元素.

子元素	描述	请参阅...
Algorithm	（必需）指定用于生成模型的算法。	『算法』
ModelingFields	指定随后在“用户界面”部分中用于定义模型的输入及输出字段控件位置的标识。控件本身在 ModelingFields 的 InputFields 和 OutputFields 子元素中定义。	第 76 页的『建模字段』
ModelGeneration	指定随后在“用户界面”部分用于为生成的模型定义模型名称控件的位置的标识。	第 78 页的『模型生成』
ModelFields	指定使用此模型评分数据的输入和输出字段组。	第 79 页的『模型字段』
AutoModeling	使此模型可以由整体建模节点（例如自动分类器、自动聚类或自动数字）使用。	第 85 页的『自动建模』

算法

Algorithm 元素可定义用于生成模型的算法的详细信息。

```
<Algorithm value="model_output_id" label="display_label" labelKey="label_key"/>
```

其中:

- value（必需）是算法的内部名称。规范文件中的许多其他位置都引用此名称。有关更多信息，请参阅第 79 页的『模型构建器示例』主题。
- label（必需）是算法的描述。
- labelKey 用于标识标签以便进行本地化。

建模字段

使用“类型”节点是指定模型输入及输出字段的标准方式。用户可根据需要将字段角色设置为 **in** 或 **out**。（可选）对于模型构建器节点，您可以允许用户选择覆盖上游“类型”节点中的设置并使用定制设置。

通过 ModelingFields 元素可以实现此操作。此元素指定一个标识，随后，在模型构建器节点声明的“用户界面”部分中，可以使用此标识来定义模型的输入及输出字段控件的位置。控件本身通过 InputFields 和 OutputFields 子元素定义。

格式

```
<ModelingFields controlsId="control_identifier" ignoreBOTH="true_false" >
      <InputFields ... />
      <OutputFields ... />
</ModelingFields>
```

其中:

- controlsId（必需）是随后在模型构建器节点声明的“用户界面”部分的 SystemControls 元素中使用的标识。这样，便可以标识节点对话框的哪个选项卡要包含模型的输入及输出字段控件。
- ignoreBOTH，如果设置为 true （缺省），那么指定模型忽略字段角色设置为 **both** 的字段。

『输入字段』及随后的章节对 InputFields 和 OutputFields 元素作了描述。

示例

本示例阐释模型构建器节点对话框的“字段”选项卡中一组建模字段控件的用法。首先，为该控件组指定标识:

```
<ModelBuilder miningFunctions="[classification]">
 ...
     <ModelingFields controlsId="modelingFields">
          <InputFields property="inputs" onlyNumeric="true" multiple="true" label="Inputs"
            labelKey="inputFields.LABEL"/>
          <OutputFields property="target" multiple="false" types="[set flag]" label="Target"
            labelKey="targetField.LABEL"/>
     </ModelingFields>
 ...
</ModelBuilder>
```

modelingFields 标识随后将在节点对话框的“用户界面”部分被引用，在定义“字段”选项卡的位置处:

```
<UserInterface ... >
     <Tabs defaultTab="1">
          <Tab label="Fields" labelKey="Fields.LABEL" helpLink="modeling_fieldstab.htm">
               <PropertiesPanel>
                    <SystemControls controlsId="modelingFields">
                    </SystemControls>
               </PropertiesPanel>
          </Tab>
 ...
</UserInterface>
```

输入字段: InputFields 元素定义用户可以为模型选择一个或多个输入字段（即预测变量）的一组字段。

此设置组成了可在此节点中看到的所有字段。如果字段从此节点上游进一步过滤，那么只有通过过滤器的字段才是可见的。该列表可以进一步对之进行限制，其方式是指定仅可选择特定存储和数据类型的字段。

```
<InputFields storage="storage_types" onlyNumeric="true_false" onlySymbolic="true_false"
     onlyDatetime="true_false" types="data_types" onlyRanges="true_false"
     onlyDiscrete="true_false" property="property_name" multiple="true_false" label="label"
     labelKey="label_key"/>
```

通过指定两个属性（其中一个属性必须来自于下面的列表），可以将要使用的字段列表限制为输入字段:

- storage 是列表属性，该属性指定在列表中所允许的字段的存储类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。有关可能存储类型的设置的信息，请参阅第 169 页的『数据和存储类型』 下的表格。
- onlyNumeric 如果设置为 true，那么指定只列出存储类型为数字的字段。
- onlySymbolic 如果设置为 true，那么指定只列出存储类型为符号（即字符串）的字段。
- onlyDatetime 如果设置为 true，那么指定只列出存储类型为日期时间的字段。

另一个指定的属性 必须来自下表:

- types 是列表属性，该属性指定在列表中所允许的字段的数据类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。可能的数据类型的设置如下:

  范围

  标志

  set

  orderedSet

  numeric

  discrete

  typeless
- onlyRanges 如果设置为 true，那么指定只列出数据类型为范围的字段。
- onlyDiscrete 如果设置为 true，那么指定只列出数据类型为离散（即标志、设置或无类型）的字段。

因此，例如一个指定 storage="[integer]" 和 types="[flag]" 的控制确保只有类型为标志的整数字段会显示在列表中。

其余属性如下:

- property 是指用来存储字段值的属性的标识。
- multiple 指定字段值是枚举列表 (true) 还是不是枚举列表 (false)。
- label 是控制的显示名称。
- labelKey 用于标识标签以便进行本地化。

输出字段: OutputFields 元素定义用户可以为模型选择一个或多个输出字段（即目标）的一组字段。

此设置组成了可在此节点中看到的所有字段。如果字段从此节点上游进一步过滤，那么只有通过过滤器的字段才是可见的。该列表可以进一步对之进行限制，其方式是指定仅可选择特定存储和数据类型的字段。

```
<OutputFields storage="storage_types" onlyNumeric="true_false" onlySymbolic="true_false"
     onlyDatetime="true_false" types="data_types" onlyRanges="true_false"
     onlyDiscrete="true_false" property="property_name" multiple="true_false" label="label"
     labelKey="label_key"/>
```

通过指定两个属性（其中一个属性必须来自于下面的列表），可以将要使用的字段列表限制为输出字段:

- storage 是列表属性，该属性指定在列表中所允许的字段的存储类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。有关可能存储类型的设置的信息，请参阅第 169 页的『数据和存储类型』 下的表格。
- onlyNumeric 如果设置为 true，那么指定只列出存储类型为数字的字段。
- onlySymbolic 如果设置为 true，那么指定只列出存储类型为符号（即字符串）的字段。
- onlyDatetime 如果设置为 true，那么指定只列出存储类型为日期时间的字段。

另一个指定的属性 必须来自下表:

- types 是列表属性，该属性指定在列表中所允许的字段的数据类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。可能的数据类型的设置如下:

  范围

  标志

  set

  orderedSet

  numeric

  discrete

  typeless
- onlyRanges 如果设置为 true，那么指定只列出数据类型为范围的字段。
- onlyDiscrete 如果设置为 true，那么指定只列出数据类型为离散（即标志、设置或无类型）的字段。

因此，例如一个指定 storage="[integer]" 和 types="[flag]" 的控制确保只有类型为标志的整数字段会显示在列表中。

其余属性如下:

- property 是指用来存储字段值的属性的标识。
- multiple 指定字段值是枚举列表 (true) 还是不是枚举列表 (false)。
- label 是控制的显示名称。
- labelKey 用于标识标签以便进行本地化。

模型生成

ModelGeneration 元素指定一个标识，在文件中的其他位置，可以使用此标识来定义模型构建器节点对话框的哪个选项卡要包含所生成模型的模型名称控件。

格式为:

```
<ModelGeneration controlsId="control_identifier" />
```

controlsId 属性指定随后在模型构建器节点规范的“用户界面”部分中的 SystemControls 元素中使用的标识。其规范包含此 SystemControls 元素的选项卡将包含模型名称控件。

模型字段

ModelFields 元素可用于构造 **模型签名** - 用于使用此模型评分数据的输入和输出字段组。

```
<ModelFields inputDirections="[in]" outputDirections="[out]">
     <AddField prefix="field_prefix" ... />
      ...
     <ForEach ... >
          <AddField prefix="field_prefix" ... />
     </ForEach>
      ...
</ModelFields>
```

其中 inputDirections 和 outputDirections 指定将如何构建模型签名；其值可以为 in、out 或 both。

这些字段本身由一个或多个 AddField 元素指定。prefix 属性指定要添加到字段名称中用来指示由模型所生成的字段的前缀。例如，如果字段名称是 field1 并且前缀值是 $S，那么生成的字段名称为 $S-field1。有关更多信息，请参阅第 60 页的『添加字段』主题。

通过 ForEach 元素可以进行迭代。有关更多信息，请参阅第 63 页的『使用 ForEach 元素执行迭代』主题。

字段组允许对两个或两个以上的模型输出字段进行分组，以供迭代使用。输出字段名称后面附加后缀以指示迭代，例如 $S-field1-1、$S-field1-2 等等。字段组的一种用途是使同一组字段在模型输出中多次出现。有关更多信息，请参阅第 80 页的『字段组示例』主题。

自动建模

AutoModeling 元素使模型可以由整体建模节点（例如自动分类器、自动聚类或自动数字）使用。有关更多信息，请参阅第 85 页的『自动建模』主题。

模型构建器示例

以下显示“交互”节点示例的规范文件中完整的模型构建器部分（请参阅第 25 页的『模型构建器节点 (Interaction)』）：

```
<Node id="interaction.builder" type="modelBuilder" palette="modeling" label="Interaction">
     <ModelBuilder miningFunctions="[classification]">
          <Algorithm value="robd" label="Robert's Algorithm" />
          <ModelingFields controlsId="modellingFields">
               <InputFields property="inputs" multiple="true" label="Inputs"
                onlyDiscrete="true" />
               <OutputFields property="target" multiple="false" label="Target"
                onlyDiscrete="true" />
          </ModelingFields>
          <ModelFields inputDirections="[in]" outputDirections="[out]">
               <ForEach var="field" inFields="outputs">
                    <AddField prefix="$I" name="${field}" fieldRef="${field}" role=
                     "predictedValue" targetField="${field}" />
                    <AddField prefix="$IP" name="${field}" storage="real" role="probability"
                     targetField="${field}">
                         <Range min="0.0" max="1.0"/>
                    </AddField>
               </ForEach>
          </ModelFields>
     </ModelBuilder>
 ...
</Node>
```

字段组示例

本示例取自 SLRM 节点，它将包含两个新字段的组添加到模型签名，以便包含对模型评分时生成的数据。对于每条输入记录，为每个新字段进行用户指定次数的数据评分，评分次数由 max_predictions 属性值决定。

两个新字段为:

- $S-*target* - 包含目标字段的预测值
- $SC-*target* - 包含此预测的概率值

为了将这两个字段分组在一起，当在 ModelFields 部分中声明它们时为其指定相同的组标识。组标识通过 AddField 元素的 group 属性进行指定。

因此，在模型构建器节点的声明中包含下列内容:

```
<Node ... type="modelBuilder" ... >
     <ModelBuilder ... >
     ...
          <ModelFields inputDirections="[in]" outputDirections="[out]">
               <AddField prefix="$S" name="${target}" fieldRef="${target}" role=
                "predictedValue" targetField="${target}" group="[1]"/>
               <AddField prefix="$SC" name="${target}" storage="real" role="probability"
                  targetField="${target}" group="[1]">
                    <Range min="0.0" max="1.0"/>
               </AddField>
          </ModelFields>
     </ModelBuilder>
</Node>
```

因此，在模型应用器节点的声明中包含下列内容:

```
<Node ... type="modelApplier" ... >
...
     <OutputDataModel mode="extend">
          <ForEach var="group" from="1" to="${max_predictions}">
               <ForEach var="field" inFields="modelOutputs" container="model">
                    <AddField name="${field}" group="[${group}]" fieldRef="${field}" />
               </ForEach>
          </ForEach>
     </OutputDataModel>
</Node>
```

目标字段命名为 **campaign**，用户在对应于 max_predictions 属性的字段中输入 2。执行模型构建器节点将导致以下字段被附加到模型:

- $S-campaign-1
- $SC-campaign-1
- $S-campaign-2
- $SC-campaign-2

模型输出

ModelOutput 元素所描述的模型输出对象将在执行流后，显示在管理器窗格的“模型”选项卡下。

格式

```
<ModelOutput id="identifier" label="display_label" labelKey="label_key" delegate="Java_class" >
     <ModelProvider ... />
     <Properties>
```

```
          <Property ... />
           ...
     </Properties>
     <Containers ... />
     <UserInterface ... />
     <Constructors ... />
</ModelOutput>
```

其中:

- id（必需）是生成的模型的唯一标识。
- label（必需）是所生成的模型将显示在“模型”选项卡上的显示名称。
- labelKey 用于标识标签以便进行本地化。

下表显示了可以包含在 ModelOutput 元素中的子元素。

表 31. 模型输出声明的子元素.

子元素	定义	请参阅...
ModelProvider	控制模型输出（无论是否为 PMML 格式）的容器。	第 48 页的『模型提供器』
Properties	生成的模型要使用的属性。	第 48 页的『属性』
Containers	放置生成的模型输出的容器。	第 50 页的『容器』
UserInterface	这是可以用来查看所生成的模型输出的用户界面，例如，模型输出对象的“模型”和“设置”选项卡。	第 51 页的『用户界面』
Constructors	生成的模型产生的对象。	第 91 页的『使用构造函数』
delegate	如果指定，请定义实现 OutputDelegate 界面的 Java 类的名称。将针对关联的输出的每个实例构造指定的类的实例	

示例

```
<ModelOutput id="interaction.model" label="Interaction Model">
     <Properties>
     </Properties>
     <Containers>
          <Container name="model" />
     </Containers>
     <UserInterface>
          <Tabs>
               <Tab label="Model">
                    <TextBrowserPanel container="model" textFormat="plainText" />
               </Tab>
          </Tabs>
     </UserInterface>
     <Constructors>
          <CreateModelApplier type="interaction.applier">
               <SetContainer target="model" source="model" />
          </CreateModelApplier>
     </Constructors>
</ModelOutput>
```

构建交互式模型

交互建模功能使您能够创建一个输出对象，最终用户可以在生成模型前与该对象进行交互。这个交互式输出对象将放在管理器窗格的“输出”选项卡上，并包含中间数据集。中间数据集可用于在模型生成之前对其进行精练或简化。通过将一些额外元素添加到常规模型构建器节点的规范中可实现交互建模:

- Node 定义的 Constructors 部分包含 CreateInteractiveModelBuilder 元素。
- 扩展包括专用的 InteractiveModelBuilder 元素。

通过名为 **交互窗口**的窗口（创建输出对象后即可显示该窗口），用户可以与中间数据集进行交互。

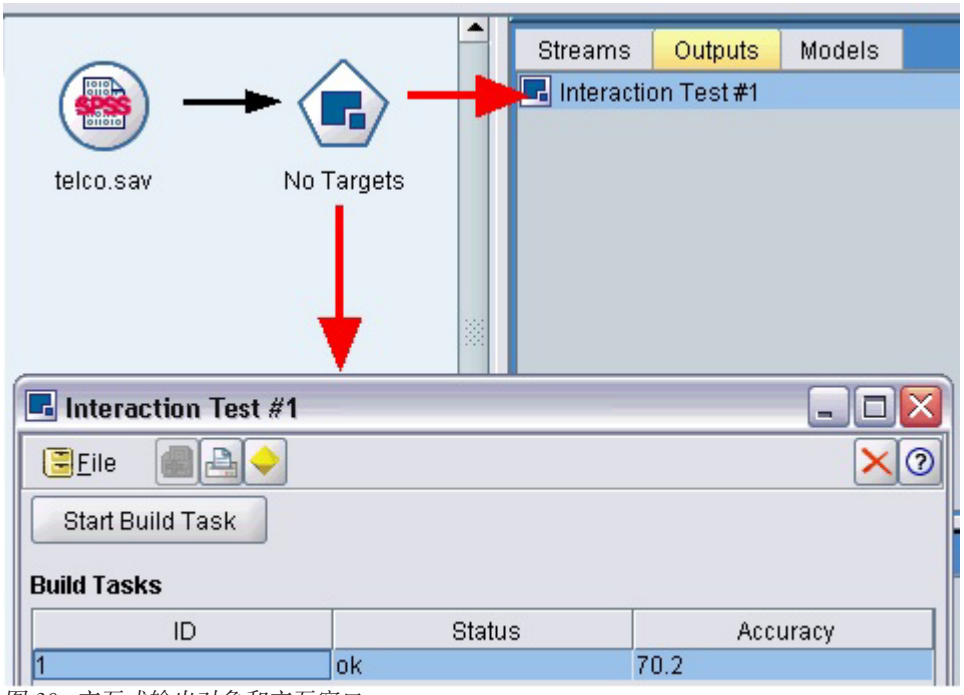

图 30. 交互式输出对象和交互窗口

交互随使用的算法不同而有所变化，并由扩展实现。交互窗口是在 InteractiveModelBuilder 元素的“用户界面”部分定义的。可以通过指定以下任一个内容定义交互窗口:

- 框架类（请参阅第 96 页的『“用户界面”部分』），用于完整地定义窗口
- 面板类，作为窗口中每个选项卡的扩展对象面板的属性指定（请参阅第 107 页的『扩展对象面板』）

关闭后，通过在“输出”选项卡上双击对象名称可以重新显示交互窗口。

交互窗口规范需要包括用来在用户完成交互后生成模型的代码。在图中所示的示例中，此操作通过带有金色块图标的工具栏按钮完成，该按钮与用于生成模型的操作相关联。此操作的代码显示在第 84 页的『交互建模示例』的 InteractiveModelBuilder 部分中。

创建交互式模型构建器

CreateInteractiveModelBuilder 元素描述了用户用于交互的输出对象。它是 CreateModelOutput 元素的一个有效交互版本。

格式

此元素可用于模型构建器节点定义的“执行”部分:

```
<Node ... type="modelBuilder" ... >
 ...
    <Execution>
     ...
        <Constructors>
         ...
            <CreateInteractiveModelBuilder ... >
```

```
                ...
            </CreateInteractiveModelBuilder>
        </Constructors>
    </Execution>
 ...
</Node>
```

元素本身的格式为:

```
<CreateInteractiveModelBuilder type="output_object_id">
    <Condition ... ./>
    <And>
    <Or>
    <Not>
    <CreateModel type="model_id" target="container_id" sourceFile="container_file_id" />
    <CreateDocument type="model_id" target="container_id" sourceFile="container_file_id" />
</CreateInteractiveModelBuilder>
```

其中 type（必需）是由 InteractiveModelBuilder 元素创建的输出对象的标识。

Condition 部分允许指定一个或多个条件。有关更多信息，请参阅第 67 页的『条件』主题。

还可以指定包含 And、Or 以及 Not 运算符的复杂条件。有关更多信息，请参阅第 67 页的『逻辑运算符』主题。

在 CreateModel 元素和 CreateDocument 元素中:

- type 是正在定义的模型或文档的标识。
- target（必需）是模型中容器的标识；在“模型输出”部分对此类容器进行了定义。有关更多信息，请参阅第 80 页的『模型输出』主题。
- sourceFile（必需）是节点执行期间生成的输出文件的标识；在“输出文件”部分对此类输出文件进行了定义。有关更多信息，请参阅第 53 页的『输出文件』主题。

示例

```
<CreateInteractiveModelBuilder type="my.interaction">
    <Condition property="interactive" op="equals" value="true" />
</CreateInteractiveModelBuilder>
```

此示例指定在执行规范文件中包含此元素的模型构建器节点时，会创建带有 my.interaction 标识的输出对象。输出对象本身由引用此标识的 InteractiveModelBuilder 元素在规范文件的其他地方定义，例如:

```
<InteractiveModelBuilder id="my.interaction" label=...>
 ...
</InteractiveModelBuilder>
```

交互式模型构建器

此元素可定义交互式输出对象，该对象允许最终用户在模型生成之前对其进行精练或简化。

InteractiveModelBuilder 元素符合包含相应 CreateInteractiveModelBuilder 元素的模型构建器节点的定义。有关更多信息，请参阅第 82 页的『创建交互式模型构建器』主题。

格式

InteractiveModelBuilder 元素的格式为:

```
<Node ... type="modelBuilder" ... >
 ...
    -- Create Interactive Model Builder section --
```

```
 ...
</Node>
 ...
<InteractiveModelBuilder id="identifier" label="display_label" labelKey="label_key">
     <Properties>
          <Property name=... />
           ...
     </Properties>
     <Containers>
          <Container name="container_name"/>
     </Containers>
     <UserInterface ... />
     <Constructors ... />
</InteractiveModelBuilder>
```

其中:

- id（必需）是生成的模型的唯一标识。
- label（必需）是所生成的模型将显示在“模型”选项卡上的显示名称。
- labelKey 用于标识标签以便进行本地化。

有关 Properties、Containers、UserInterface 和 Constructors 元素的更多信息，请参阅第 48 页的『属性』、第 50 页的『容器』、第 96 页的『“用户界面”部分』和第 91 页的『使用构造函数』。

交互建模示例

本示例阐释如何按以下方式定义模型构建器节点：用户可以通过一个简单的复选框选择是否在生成模型前进行交互。

要查看实际工作情况，请使用此发行版随附的“交互”示例节点。有关更多信息，请参阅第 25 页的『模型构建器节点 (Interaction)』主题。

首先，模型构建器节点指定布尔值属性:

```
<Node id="interaction.builder" type="modelBuilder" ... >
 ...
     <Properties>
      <Property name="interactive" valueType="boolean" />
    </Properties>
```

在节点规范的“用户界面”部分中，定义“模型”选项卡的部分包含对此属性的引用:

```
<Tab label="Model">
     <PropertiesPanel>
          <CheckBoxControl property="interactive" label="Start an interactive session" />
     </PropertiesPanel>
</Tab>
```

在同一节点的 CreateInteractiveModelBuilder 部分，将测试属性的设置，如果结果为 true，那么将创建交互式输出对象:

```
<CreateInteractiveModelBuilder type="my.interaction">
     <Condition property="interactive" op="equals" value="true" />
</CreateInteractiveModelBuilder>
```

它引用的输出对象在扩展的 InteractiveModelBuilder 部分中定义:

```
<InteractiveModelBuilder id="my.interaction" label="Interaction Test">
     <Properties>
     </Properties>
```

```
     <Containers>
     </Containers>
     <UserInterface actionHandler="ui.InteractionHandler">
          <Controls>
               <ToolbarItem action="generateModel" showLabel="false" />
          </Controls>
          <Tabs>
               <Tab label="Model">
                    <ExtensionObjectPanel id="model.panel" panelClass=
                     "ui.SampleInteractionPanel" />
               </Tab>
               <Tab label="Generic">
                    <ExtensionObjectPanel id="generic.panel" panelClass=
                     "ui.GenericInteractionPanel" />
               </Tab>
          </Tabs>
     </UserInterface>
</InteractiveModelBuilder>
```

生成模型的操作是由 ToolbarItem 元素定义的工具栏按钮控制的。

请注意，使用 ExtensionObjectPanel 元素的 panelClass 属性来指定 Java 类以控制交互窗口每个选项卡的用户界面。

自动建模

IBM SPSS Modeler 提供了一组标准整体建模节点，如自动分类器节点、自动聚类节点和自动数字节点。这些节点可同时自动构建多个不同的模型，从而允许最终用户比较结果并为其数据选择最佳模型。CLEF 提供了 AutoModeling 元素可允许任意这些整体节点使用由 ModelBuilder 元素指定的模型。

AutoModeling 元素的格式为:

```
<AutoModeling enabledByDefault="true_false">
     <SimpleSettings ... />
     <ExpertSettings ... />
     <Constraint ... />
     <Constraint ... />
      ...
</AutoModeling>
```

其中 enabledByDefault 指定缺省情况下是否允许在整体建模节点中使用模型（即，缺省情况下是否对特定模型选中**使用?** 列）。如果忽略该属性，那么假定值为真。

整体建模节点对话框的“专家”选项卡列出了用户可以选择构建的模型。

单击特定模型的**模型参数**字段中的**指定**可以显示“算法设置”对话框，用户可以在其中为该模型类型选择选项。

“算法设置”对话框包含“简单”选项卡和“专家”选项卡，它们分别对应于建模节点的“简单”和“专家”执行模式。“简单”选项卡和“专家”选项卡的内容由 SimpleSettings 和 ExpertSettings 元素控制，这些元素将在下面几节中进行介绍。

另外，通过 Constraint 元素，您可以指定允许最终用户编辑或约束（在某些方面）“算法设置”对话框中的参数的条件。有关更多信息，请参阅第 88 页的『约束』主题。

“算法设置”对话框中的某些参数可以具有多个值。在指定多个值的位置，整体节点将尝试为所有可能的参数值组合构建模型。例如，对于广义线性模型，如果用户指定两个分布（正态和伽玛）和三个链接功能（恒等式、对数和幂），自动数字节点将尝试构建六个广义线性模型，分别对应于每一个可能的参数组合。

简单设置

对于整体建模节点中的这种模型，SimpleSettings 元素可确定“算法设置”对话框的“简单”选项卡上将显示哪些参数。有关更多信息，请参阅第 85 页的『自动建模』主题。

格式

```
<SimpleSettings>
     <PropertyGroup label="group_name" labelKey="resource_key" properties="[prop_name1
      prop_name2 ...]"/>
     <PropertyGroup ... />
</SimpleSettings>
```

在 PropertyGroup 元素中（必须至少有一个）：

label 是属性组的显示标签，它作为副标题插入到该属性组第一个参数前的对话框中。

labelKey 用于标识标签以便进行本地化。如果既没有使用 label，也没有使用 labelKey，那么不插入副标题。

properties（必需）是要在选项卡中显示的一个或多个属性的列表。*prop_name1* 和 *prop_name2* 等的值 是 Property 元素的 name 属性的值，该属性是该元素中定义的。有关更多信息，请参阅第 48 页的『属性』主题。

示例

```
<SimpleSettings>
     <PropertyGroup properties="[method]"/>
</SimpleSettings>
```

本示例来自“判别”节点，指定只有 **Method** 参数将显示在相关整体建模节点模型的“算法设置”对话框的“简单”选项卡中（在此示例中，是自动分类器节点）。由于没有指定 label 或 labelKey 属性，因此没有在对话框中显示参数的子标题。

专家设置

对于整体建模节点中的这种模型，ExpertSettings 元素可确定“算法设置”对话框的“专家”选项卡上将显示哪些参数。有关更多信息，请参阅第 85 页的『自动建模』主题。

格式

```
<ExpertSettings>
     <Condition ... />
     <PropertyGroup label="group_name" labelKey="resource_key"
      properties="[property1 property2 ...]"/>
     <PropertyGroup ... />
      ...
</ExpertSettings>
```

Condition 元素指定这样一个条件：如果该条件为 true，将导致由后续的一个或多个 PropertyGroup 元素标识的参数被启用。有关更多信息，请参阅第 67 页的『条件』主题。

在 PropertyGroup 元素中（必须至少有一个）：

label 是属性组的显示标签，它作为副标题插入到该属性组第一个参数前的对话框中。

labelKey 用于标识标签以便进行本地化。如果既没有使用 label，也没有使用 labelKey，那么不插入副标题。

properties（必需）是要在选项卡中显示的一个或多个属性的列表。*prop_name1* 和 *prop_name2* 等的值 是 Property 元素的 name 属性的值，该属性是该元素中定义的。有关更多信息，请参阅第 48 页的『属性』主题。

示例

在下面的示例中，“算法设置”对话框的“专家”选项卡将**模式**参数最初设置为**简单**。

图 31. 禁用专家设置

下面的内容指定只有当用户将**模式**参数设置更改为**专家**时，才可以启用其他“专家”选项卡参数:

```
<ExpertSettings>
     <Condition property="mode" op="equals" value="Expert"/>
     <PropertyGroup properties="[mode prior_probabilities covariance_matrix]"/>
      ...
</ExpertSettings>
```

将**模式**设置更改为**专家**将在属性组中启用这两个参数。

图 32. 启用专家设置

下一个示例阐释属性组标签的用法:

```
<ExpertSettings>
      ...
     <PropertyGroup labelKey="automodel.stepping_criteria_options"
      properties="[stepwise_method V_to_enter criteria]"/>
      ...
</ExpertSettings>
```

这里，PropertyGroup 元素控制下图中突出显示的参数。

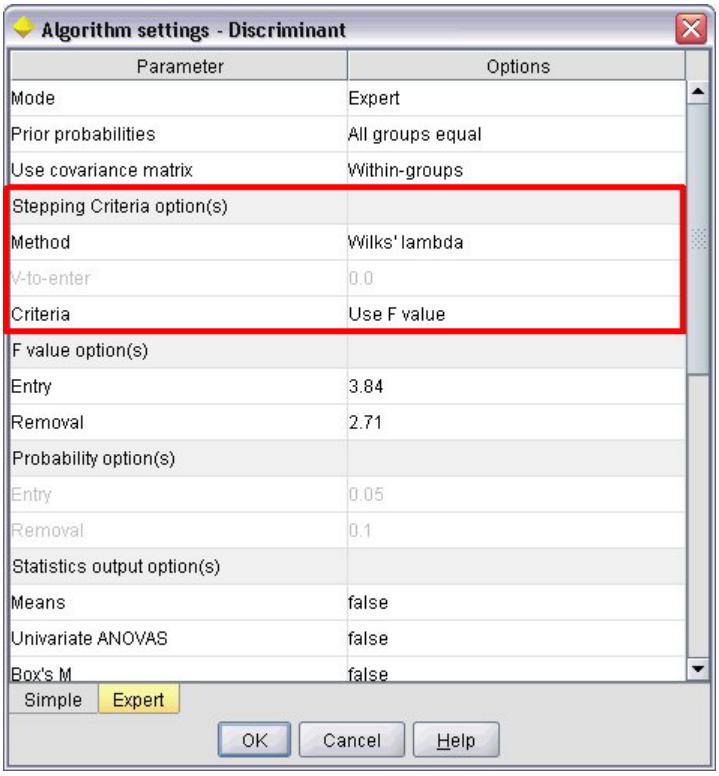

图 33. 禁用专家设置

labelKey 属性允许 CLEF 检索扩展的属性文件中对应条目的属性组子标题的显示文本:

automodel.stepping_criteria_options=Stepping Criteria option(s)

有关更多信息，请参阅第 154 页的『属性文件』主题。

约束

Constraint 元素指定允许编辑的条件，或在某些方面对在整体建模节点中某个模型的“算法设置”对话框中列出的参数的约束条件。例如，如果当前不允许最终用户修改某些参数，那么这些参数可能被禁用。

格式

```
<Constraint property="prop_name" singleSelection="true_false">
     <Condition property="prop_name" op="operator" value="value"/>
     <And ... />
     <Or ... />
     <Not ... />
</Constraint>
```

其中:

- property（必需）标识要编辑或约束的参数。*prop_name* 是在其中定义此参数的相应特性的 Property 元素的 name 属性值。有关更多信息，请参阅第 48 页的『属性』主题。
- singleSelection 控制最终用户是否可以为参数选择多个可用值。如果设置为 true，那么即使在“算法设置”对话框的该参数的“选项”字段中列出了多个值，也只能选择一个值。如果设置为 false（缺省），最终用户可以选择一个或多个可用值，如下面的示例所示。

Condition 元素指定实际的约束。有关更多信息，请参阅第 67 页的『条件』主题。

And、Or 和 Not 元素可用于指定复合条件。有关更多信息，请参阅第 67 页的『逻辑运算符』主题。

示例

本示例取自于“广义线性”节点的规范文件。缺省情况下，即使在“专家”模式中，某些参数也无法启用。

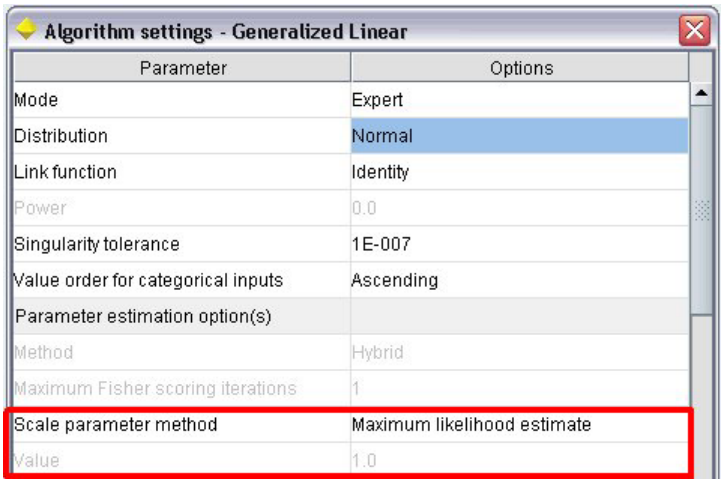

图 34. 约束的效果 - 将参数禁用

约束指定在其下启用值参数的条件:

```
<Constraint property="scale_value">
     <And>
          <Condition property="scale_method" op="equals" value="FixedValue"/>
          <Condition property="distribution" op="in" value="[IGAUSS GAMMA NORMAL]"/>
     </And>
</Constraint>
```

尺度参数方法参数（由 scale_method 属性标识）必须设置为 **固定值**，并且 **分布**必须在启用 **值**参数之前设置为 **正态**、**逆高斯**或 **伽玛**。

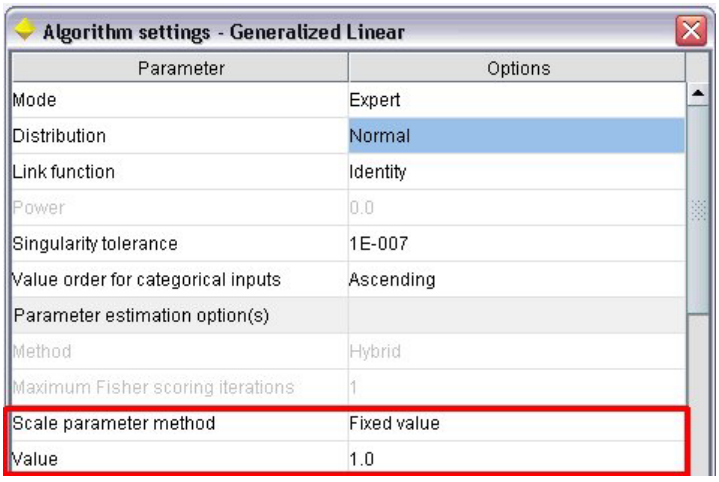

图 35. 约束的效果 - 将参数启用

应用模型

应用模型是指使用生成的模型对数据进行评分，即，使用模型构建所获得的信息为新记录创建预测。在 IBM SPSS Modeler 中，您可以通过模型应用器节点实现此操作。有关更多信息，请参阅第 12 页的『模型应用器节点』主题。

在规范文件中定义模型应用器节点将创建用于应用生成的模型的框架。在 IBM SPSS Modeler 中，通过将表示管理器窗格“模型”选项卡的模型输出对象的图标拖放到流工作区上，可创建模型应用器节点的实例。如果没有模型应用器节点定义，执行模型构建器节点将只生成非精练模型，该模型无法添加到流工作区中。

定义模型应用器节点时，Node 元素必须包括:

- type="modelApplier" 属性
- 一个 Constructors 子元素，其中包含 CreateModelOutput 元素（请参阅第 91 页的『使用构造函数』）

要了解 Node 元素规范的格式，请参阅第 45 页的『节点』。

构建文档

定义文档构建器节点时，Node 元素必须包含:

- type="documentBuilder" 属性
- DocumentBuilder 子元素

构建文档时，扩展还需要 DocumentOutput 元素来描述生成的文档。有关更多信息，请参阅第 80 页的『模型输出』主题。

要了解 Node 元素规范的格式，请参阅第 45 页的『节点』。

文档构建器

DocumentBuilder 元素定义文档构建器节点的行为。定义必须包含 DocumentGeneration 子元素以便指定文档构建器节点对话框中的哪个选项卡将包含文档生成控件。控件本身是在“用户界面”部分中定义的（请参阅第 95 页的第 6 章，『构建用户界面』）。

格式

```
<DocumentBuilder>
     <DocumentGeneration controlsId="control_identifier />
</DocumentBuilder>
```

其中 controlsId（必需）是系统控件用于指定文档生成控件出现位置的标识。

示例

```
<DocumentBuilder>
     <DocumentGeneration controlsId="1"/>
</DocumentBuilder>
```

文档输出

DocumentOutput 元素所描述的文档输出对象将在执行流后，显示在管理器窗格的“输出”选项卡下。

格式

```
<DocumentOutput id="identifier" label="display_label" labelKey="label_key"
delegate="Java_class" >
     <Properties>
          <Property ... />
           ...
     </Properties>
     <Containers>
          <Container ... />
           ...
     </Containers>
     <UserInterface ... />
     <Constructors ... />
</DocumentOutput>
```

其中:

- id（必需）是生成的文档的唯一标识。
- label（必需）是所生成的文档将显示在“输出”选项卡上的显示名称。
- labelKey 用于标识标签以便进行本地化。

下表显示了可以包含在 DocumentOutput 元素中的子元素。

表 32. 文档输出声明的子元素.

子元素	定义	请参阅...
Properties	生成的文档要使用的属性。	第 48 页的『属性』
Containers	将放置生成的文档输出的容器。	第 50 页的『容器』
UserInterface	通过其可以查看生成的文档输出的用户界面。	第 51 页的『用户界面』
Constructors	生成的文档产生的对象。	『使用构造函数』
delegate	如果指定，请定义实现 OutputDelegate 界面的 Java 类的名称。将针对关联的输出的每个实例构造指定的类的实例	

示例

```
<DocumentOutput id="webstatusreport">
     <Containers>
          <Container name="webstatusreportdata" />
     </Containers>
     <UserInterface>
          <Tabs>
               <Tab label="Advanced" labelKey="advancedTab.LABEL" >
                    <TextBrowserPanel container="webstatusreportdata" textFormat="html" />
               </Tab>
          </Tabs>
     </UserInterface>
</DocumentOutput>
```

使用构造函数

Constructors 元素可以显示在规范文件中以下两个位置中的任意一处:

- 在节点定义的“执行”部分（针对模型或文档输出对象）
- 在模型输出定义中（针对模型应用器节点）

一个节点只能生成一个输出对象。施加此限制是为了保证与现有节点、脚本编写以及这些“类型”节点的客户端 API 接口保持一致。 .

格式

在“执行”部分使用时，Constructors 元素的格式为:

```
<Constructors>
     <CreateModelOutput ... />
      ...
     <CreateDocumentOutput ... />
      ...
     <CreateInteractiveModelBuilder ... />
      ...
</Constructors>
```

在模型输出定义中，格式为:

```
<Constructors>
     <CreateModelApplier ... />
</Constructors>
```

创建模型输出

此部分定义如何在“模型”选项卡上创建模型输出对象，或如何在“输出”选项卡上创建文档输出对象。

格式

```
<CreateModelOutput type="output_object_id">
     <Condition ... ./>
     <And>
     <Or>
     <Not>
     <CreateModel type="model_id" target="container_id" sourceFile="container_file_id" />
      ...
     <CreateDocument type="document_id" target="container_id" sourceFile="container_file_id" />
      ...
</CreateModelOutput>
```

在 CreateModelOutput 元素中:

- type（必需）是在“模型输出”部分定义的模型输出对象的标识。有关更多信息，请参阅第 47 页的『模型输出』主题。

Condition 部分允许指定一个或多个条件。有关更多信息，请参阅第 67 页的『条件』主题。

还可以指定包含 And、Or 和 Not 运算符的复杂条件。有关更多信息，请参阅第 67 页的『逻辑运算符』主题。

在 CreateModel 元素和 CreateDocument 元素中:

- type 是正在定义的模型或文档的标识。
- target（必需）是模型中容器的标识；在“模型输出”部分对此类容器进行了定义。有关更多信息，请参阅第 80 页的『模型输出』主题。
- sourceFile（必需）是节点执行期间生成的输出文件的标识；在“输出文件”部分对此类输出文件进行了定义。有关更多信息，请参阅第 53 页的『输出文件』主题。

示例

```
<Constructors>
     <CreateModelOutput type="naivebayes">
          <CreateModel type="naivebayes_model" target="model" sourceFile="pmml"/>
          <CreateDocument type="html_output" target="advanced_output" sourceFile="htmloutput" />
          <CreateDocument type="zip_outputType" target="zip_output" sourceFile="zipoutput" />
     </CreateModelOutput>
</Constructors>
```

创建文档输出

此元素可在文档构建器节点定义的“执行”部分使用，并且可以标识正在创建的文档输出对象。

格式

```
<CreateDocumentOutput type="output_object_id">
     <Condition ... ./>
     <And>
     <Or>
     <Not>
     <CreateModel type="model_id" target="container_id" sourceFile="container_file_id" />
      ...
     <CreateDocument type="document_id" target="container_id" sourceFile="container_file_id" />
      ...
</CreateModelOutput>
```

其中 type（必需）是在“文档输出”部分定义的文档输出对象的标识。有关更多信息，请参阅第 48 页的『文档输出』主题。

Condition 部分允许指定一个或多个条件。有关更多信息，请参阅第 67 页的『条件』主题。

还可以指定包含 And、Or 和 Not 运算符的复杂条件。有关更多信息，请参阅第 67 页的『逻辑运算符』主题。

在 CreateModel 元素和 CreateDocument 元素中:

- type 是正在定义的模型或文档的标识。
- target（必需）是模型中容器的标识；在“模型输出”部分对此类容器进行了定义。有关更多信息，请参阅第 80 页的『模型输出』主题。
- sourceFile（必需）是节点执行期间生成的输出文件的标识；在“输出文件”部分对此类输出文件进行了定义。有关更多信息，请参阅第 53 页的『输出文件』主题。

示例

```
<Constructors>
     <CreateDocumentOutput type="webstatusreport">
          <CreateDocument type="webstatusreport" target="webstatusreportdata"
               sourceFile="webstatusreport_output_file" />
     </CreateDocumentOutput>
</Constructors>
```

创建交互式模型构建器

此元素在交互式模型构建器节点定义的“执行”部分中使用，用于标识与用户进行交互的输出对象。有关更多信息，请参阅第 82 页的『创建交互式模型构建器』主题。

创建模型应用器

此元素在模型构建器节点定义的“模型输出”部分中的 Constructors 元素内使用（请参阅第 80 页的『模型输出』）。CreateModelApplier 元素定义在将模型构建器节点生成的模型输出对象拖放到工作区时如何创建模型应用器节点。

格式

```
<CreateModelApplier type="apply_node_identifier">
     <Condition ... ./>
     <And>
     <Or>
     <Not>
     <CreateModel type="model_id" target="container_id" sourceFile="container_file_id" />
      ...
     <CreateDocument type="document_id" target="container_id" sourceFile="container_file_id" />
      ...
</CreateModelApplier>
```

在 CreateModelApplier 元素中:

- type（必需）是要创建的模型应用器节点的标识；实际上，此节点是随后在文件的 Node ... type="modelApplier" 元素中定义的。

Condition 部分允许指定一个或多个条件。有关更多信息，请参阅第 67 页的『条件』主题。

还可以指定包含 And、Or 以及 Not 运算符的复杂条件。有关更多信息，请参阅第 67 页的『逻辑运算符』主题。

在 CreateModel 元素和 CreateDocument 元素中:

- type 是正在定义的模型或文档的标识。
- target（必需）是模型中容器的标识；在“模型输出”部分对此类容器进行了定义。有关更多信息，请参阅第 80 页的『模型输出』主题。
- sourceFile（必需）是节点执行期间生成的输出文件的标识；在“输出文件”部分对此类输出文件进行了定义。有关更多信息，请参阅第 53 页的『输出文件』主题。

示例

在下面的示例中，CreateModelApplier 元素包含一个对名为 myapplier 的模型应用器节点的前向引用。该节点在随后的 Node 元素中定义。

```
<ModelOutput>
     <Constructors>
          <CreateModelApplier type="myapplier"></CreateModelApplier>
     </Constructors>
</ModelOutput>
<Node id="myapplier" type="modelApplier">
 ...
</Node>
```

第 6 章 构建用户界面

关于用户界面

当添加新的 CLEF 节点时，需要定义最终用户与节点、任何模型输出和文档输出进行交互的方式，或者定义如何应用扩展所指定的节点。每个对象的用户界面是在扩展的规范文件的 `UserInterface` 部分进行定义的。本章将详细介绍 `UserInterface` 部分的内容。

注意: 根据文件中所定义的对象类型，单个规范文件中可能存在多个 `UserInterface` 部分。

CLEF 对象的用户界面由以下一项或多项组成:

- 菜单项
- 选用板项
- 图标
- 一个或多个对话窗口
- 一个或多个输出窗口

通过**菜单项**和**选用板项**，用户可以从 IBM SPSS Modeler 菜单系统和节点选用板分别访问对象。**图标**可识别菜单、画布以及多种节点选用板中的对象。**对话窗口**包含选项卡和控件，允许用户在执行流之前指定各种选项，还可以在执行完成后指定要生成的输出。**输出窗口**用于显示模型输出（如，对数据集应用模型时生成的结果）或文档输出（如图表）。

CLEF 允许您将新的对象类型添加到由 IBM SPSS Modeler 提供的标准类型中，并且还支持使用一致的方法为这些新对象定义用户界面。有关在 CLEF 中创建图标和对话框的指导准则，请参阅第 14 页的『设计节点图标』和 第 18 页的『设计对话框』。

图标采用了图形的格式，这些图形可标识特殊节点，并且显示在标识节点类型的几何形状内部。

对话框和**输出窗口**具有以下特征:

- 包含缩图图标和对象标题的标题栏
- 菜单栏内容包括:
  - •菜单
  - •特定于对象的操作按钮
  - •常用操作按钮（例如，最大化或帮助）
- 主要内容区域
- 用于将 UI 组件组织到逻辑组中的多个选项卡
- 重新调整容量大小

选项卡可以以多种不同的方式更改窗口的主要内容区域。对于对话窗口，不同的选项卡显示不同对象属性的控件集。可以对这些控件进行修改，并且当用户单击 **应用**或 **确定**按钮时，修改的结果将应用到基础对象中。

对于输出窗口，用户使用选项卡可以执行与输出相关的不同操作，例如显示结果汇总或将对结果添加注解。

“用户界面”部分

对象的用户界面是在规范文件中对象定义内的 UserInterface 元素部分进行声明的。在同一规范文件中，可以有一个或多个 UserInterface 元素，这取决于文件中所定义对象（例如，模型构建器节点、模型输出对象和模型应用器节点）的数量。

在每个“用户界面”部分中，您都可以定义以下内容:

- 在工作区或选用板中显示的图标
- 对话框或输出窗口中显示的控件（定制菜单和工具栏项）
- 定义属性控件集的选项卡（针对于对话框或输出窗口）

注意: 在下面的元素定义（通常由标题**格式**标识）中，除非指示为“（必需）”，否则元素属性和子元素均为可选项。有关所有元素的完整语法信息，请参阅第 189 页的『CLEF XML 模式』。

对于每一个用户界面，可以指定要执行的处理操作。您可以通过操作处理程序或帧类属性执行此项操作，这两种方法都可以选择。如果操作处理程序或帧类属性都未作指定，将在文件的其他位置指定要执行的处理操作。

格式

UserInterface 元素的基本格式为:

```
<UserInterface>
     <Icons>
          <Icon ... />
           ...
     </Icons>
     <Controls>
          <Menu ... />
          <MenuItem ... />
           ...
          <ToolbarItem ... />
           ...
     </Controls>
     <Tabs>
          <Tab ... />
           ...
     </Tabs>
</UserInterface>
```

扩展对象 UI 代表与一个节点对话框或者一个模型或文档输出窗口相关联，并且允许扩展处理在对话框上调用的定制操作。UI 代表指定扩展提供的 Java 类，该类与 IBM SPSS Modeler 窗口同时创建，并且是 ExtensionObjectUIDelegate 类的实现。请参阅主题第 161 页的『客户端 API 类』，了解更多信息。

除第一行以外，UI 代表的格式与基本格式相同:

```
<UserInterface uiDelegate="Java_class" >
      ...
</UserInterface>
```

其中，uiDelegate 是 UI 代表 Java 类的名称。

请注意，扩展不应假定 UI 可调用。例如，可以在“headless”（非 UI）环境中运行扩展，例如批处理方式和应用程序服务器。

当您将定制操作添加至标准 IBM SPSS Modeler 窗口时可使用 **操作处理程序**。其类似于**扩展对象 UI 代表**，但是在没有扩展节点或输出窗口时也可使用操作处理程序，例如，在定义可直接从 IBM SPSS Modeler主窗口

调用的新工具时。操作处理程序可以指定 Java 类，这些 Java 类将在用户选择节点对话框、模型输出窗口或文档输出窗口中的定制菜单选项或工具栏按钮时被调用。它可以实现 ExtensionObjectFrame 类或 ActionHandler 类。无论在何种情况下，都将自动包含标准窗口组件，如标准菜单、选项卡以及工具栏按钮。有关更多信息，请参阅第 161 页的『客户端 API 类』主题。

操作处理程序的格式也与基本格式相同:

```
<UserInterface actionHandler="Java_class" >
      ...
</UserInterface>
```

其中 actionHandler 是操作处理程序 Java 类的名称。

建议的实践是在扩展添加可直接从 IBM SPSS Modeler 主窗口调用的工具时使用**操作处理程序**，并将**扩展对象 UI 代表**用于与扩展所定义的节点和输出相关联的窗口，因为 UI 代表提供更简单的底层节点或输出对象访问。扩展不应在相同的 UserInterface 元素中同时定义 uiDelegate 和 actionHandler。

帧类可用于模型输出或文档输出对象，其中扩展将提供其自身的窗口而无需定制标准 IBM SPSS Modeler 窗口。帧类是一种可完全指定整个窗口及其处理的 Java 类。不会自动包含标准窗口组件 - 类必须逐个指定这些组件。帧类只能用于模型输出或文档输出对象，而不能用于节点（其经常使用 IBM SPSS Modeler 对话框）。有关更多信息，请参阅第 147 页的『定制输出窗口』主题。

帧类的格式比较简单:

```
<UserInterface frameClass="Java_class" />
```

其中 frameClass 是模型输出或文档输出对象的帧类名称。任何图标、控件和选项卡都由帧类自身指定，因此不以此格式使用那些元素。

UserInterface 元素的子元素将在随后的章节中进行介绍。

示例

第一个示例显示定义 UI 代表的模型构建器节点的用户界面:

```
<UserInterface uiDelegate="com.spss.myextension.MyNodeUIDelegate">
     <Icons>
          <Icon type="standardNode" imagePath="images/lg_discriminant.gif" />
          <Icon type="smallNode" imagePath="images/sm_discriminant.gif" />
     </Icons>
     <Tabs defaultTab="1">
      ...
     </Tabs>
</UserInterface>
```

对应的模型输出对象部分为:

```
<UserInterface>
     <Icons>
          <Icon type="standardWindow" imagePath="images/browser_discriminant.gif" />
     </Icons>
     <Tabs>
          <Tab label="Advanced" labelKey="advancedTab.LABEL"
           helpLink="discriminant_output_advancedtab.htm">
               <ExtensionObjectPanel id="DiscriminantPanel"
```

```
                panelClass="com.spss.clef.discriminant.DiscriminantPanel"/>
          </Tab>
     </Tabs>
</UserInterface>
```

模型应用器节点的“用户界面”部分如下所示:

```
<UserInterface>
     <Icons>
          <Icon type="standardNode" imagePath="images/lg_gm_discriminant.gif" />
          <Icon type="smallNode" imagePath="images/sm_gm_discriminant.gif" />
     </Icons>
     <Tabs>
          <Tab label="Advanced" labelKey="advancedTab.LABEL"
           helpLink="discriminant_output_advancedtab.htm">
               <ExtensionObjectPanel id="DiscriminantPanel"
                panelClass="com.spss.clef.discriminant.DiscriminantPanel"/>
          </Tab>
     </Tabs>
</UserInterface>
```

图标

本部分定义与对象相关的图标。

对于节点，本部分应定义以下两个 Icon 元素:

- 在工作区中显示的较大图标
- 在选用板上显示的较小图标

对于模型输出和文档输出，这将定义显示在窗口框架左上角的缩图图标。

有关在 CLEF 中创建图标的指导准则，请参阅第 14 页的『设计节点图标』。

格式

```
<Icons>
     <Icon type="icon_type" imagePath="image_path" />
      ...
</Icons>
```

其中:

type（必需）是下表所示的其中一种图标类型。

表 33. 图标类型.

类型	示例	描述
standardNode	 *图 36. 标准节点图标*	工作区中以节点形状显示的大尺寸（49 x 49 像素）图标。

表 33. 图标类型 (续).

类型	示例	描述
smallNode	 *图 37. 小节点图标*	节点选用板上以节点形状显示的小尺寸（38 x 38 像素）图标。
window	 *图 38. 窗口图标*	浏览器或输出窗口中显示的缩图（16 x 16 像素）图标。

imagePath（必需）可标识图标中所用图像的位置。此位置相对于规范文件的安装目录进行指定。

示例

```
<Icon type="standardNode" imagePath="images/lg_mynode.gif" />
<Icon type="smallNode" imagePath="images/sm_mynode.gif" />
<Icon type="window" imagePath="images/mynode16.gif" />
```

Controls

本部分定义定制菜单和工具栏项，这些项可用于调用在规范文件的“公共对象”部分声明的操作。有关更多信息，请参阅第 38 页的『Actions』主题。

格式

```
<Controls>
     <Menu ... />
      ...
     <MenuItem ... />
      ...
     <ToolbarItem ... />
      ...
</Controls>
```

Menu、MenuItem 和 ToolbarItem 元素将在随后的章节中进行介绍。

示例

以下示例将一个项添加到“生成”菜单以及对话框（该对话框的规范中包含该项）的工具栏中。两个项目都将完成先前定义的名为 generateDerive（生成“导出”节点）的操作。

```
<Controls>
     <MenuItem action="generateDerive" systemMenu="generate"/>
     <ToolbarItem action="generateDerive" showLabel="false"/>
      ...
</Controls>
```

菜单

可以定义定制菜单以添加到其中一个标准菜单中。

格式

```
<Menu id="name" label="display_label" labelKey="label_key" systemMenu="menu_name"
showLabel="true_false" showIcon="true_false" separatorBefore="true_false" separatorAfter="true_
false" offset="integer" mnemonic="mnemonic_char" mnemonicKey="mnemonic_key"/>
```

其中:

id（必需）是正在添加的菜单的唯一标识。

label（必需）是菜单出现在用户界面上的显示名称（前提是将 showLabel 设置为 true）。

labelKey 用于标识标签以便进行本地化。

systemMenu（必需）标识定制菜单要添加到的标准菜单。*menu_name* 的值为以下其中一项:

- 文件
- edit
- insert*
- view*
- tools*
- window*
- generate
- help*
- file.project
- file.outputs
- file.models
- edit.stream
- edit.node
- edit.outputs
- edit.models
- edit.project
- tools.repository
- tools.options
- tools.streamProperties

* 仅当添加到主要的 IBM SPSS Modeler 窗口时有效

showLabel 指定用户界面上是否显示项目的显示标签。

showIcon 指定用户界面上是否显示项目的相关图标。

separatorBefore 指定是否应将分隔符（例如，菜单项的水平条或工具栏按钮的空格）添加到菜单中新项目的前面。

separatorAfter 指定是否应将分隔符添加到菜单中新项目的后面。

offset 是定义新项目位置的非负整数，例如将 0 偏移量添加它可作为第一个项目（将 缺省偏移量添加它作为最后一个项目）。

mnemonic 是与 Alt 键一起使用以激活此控件的字母符号（例如，如果您给其值为 S，那么用户可通过 Alt-S 激活此控件）。

mnemonicKey 用于标识助记符以便进行本地化。如果不使用 mnemonic 和 mnemonicKey，那么此控件无可用的助记符。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

菜单项

可以定义定制菜单项以添加到其中一个标准或定制菜单中。

格式

```
<MenuItem action="identifier" systemMenu="menu_name" customMenu="menu_name"
     showLabel="true_false" showIcon="true_false" separatorBefore="true_false"
     separatorAfter="true_false" offset="integer" />
```

其中:

action（必需）是在“公共对象”部分定义的操作的标识，可以指定该菜单项要执行的操作。

systemMenu 指定将出现在由 *menu_name* 标识的标准菜单上的项，它可以是以下任意一项:

- 文件
- edit
- insert*
- view*
- tools*
- window*
- generate
- help*
- file.project
- file.outputs
- file.models
- edit.stream
- edit.node
- edit.outputs
- edit.models
- edit.project
- tools.repository
- tools.options
- tools.streamProperties

* 仅当添加到主要的 IBM SPSS Modeler 窗口时有效

customMenu 是 Menu 元素中的标识，可以指定在该定制菜单上出现的菜单项。

showLabel 指定用户界面上是否显示项目的显示标签。

showIcon 指定用户界面上是否显示项目的相关图标。

separatorBefore 指定是否应将分隔符（例如，菜单项的水平条或工具栏按钮的空格）添加到菜单中新项目的前面。

separatorAfter 指定是否应将分隔符添加到菜单中新项目的后面。

offset 是定义新项目位置的非负整数，例如将 0 偏移量添加它可作为第一个项目（将 缺省偏移量添加它作为最后一个项目）。

示例

```
<MenuItem action="generateSelect" systemMenu="generate" showIcon="true"/>
```

工具栏项

您可以为对话框或输出窗口定义定制工具栏项。

格式

```
<ToolbarItem action="action" showLabel="true_false" showIcon="true_false"
     separatorBefore="true_false" separatorAfter="true_false" offset="integer" />
```

其中:

action（必需）是在“公共对象”部分定义的操作的标识，可以指定该工具栏项要执行的操作。

showLabel 指定用户界面上是否显示项目的显示标签。

showIcon 指定用户界面上是否显示项目的相关图标。

separatorBefore 指定是否应将分隔符（例如，菜单项的水平条或工具栏按钮的空格）添加到菜单中新项目的前面。

separatorAfter 指定是否应将分隔符添加到菜单中新项目的后面。

offset 是定义新项目位置的非负整数，例如将 0 偏移量添加它可作为第一个项目（将 缺省偏移量添加它作为最后一个项目）。

示例

```
<ToolbarItem action="generateDerive" showLabel="true"/>
```

示例: 添加至主窗口

这是将一个新项目添加至主窗口中的“工具”菜单的规范文件示例。本示例不定义任何标准对象（例如，节点、模型输出窗口或文档输出窗口），但具有顶层 Extension（表示“更改主窗口”）中的 UserInterface 元素。所有其他 UserInterface 部分都将在其中一个标准对象定义中出现，并且会影响与这些对象相关的对话框。

```
<?xml version="1.0" encoding="UTF-8"?>
<Extension version="1.0">
     <ExtensionDetails providerTag="example" id="main_window" label="Main Window" version="1.0"
          provider="IBM Corp." copyright="(c) 2005-2006 IBM Corp."
          description="An example extension that plugs into the main window"/>
     <Resources/>
     <CommonObjects>
          <Actions>
               <Action id="customTool1" label="Custom Tool..." labelKey="customTool.LABEL"
                    imagePath="images/generate.gif" description="Invokes the custom tool"
                    descriptionKey="customTool.TOOLTIP"/>
```

```
            <Action id="customTool2" label="Custom Tool..." labelKey="customTool.LABEL"
                imagePath="images/generate.gif" description="Invokes the custom tool"
                descriptionKey="customTool.TOOLTIP"/>
        </Actions>
    </CommonObjects>
    <UserInterface actionHandler="com.spss.cleftest.MainWindowActionHandler">
        <Controls>
            <Menu systemMenu="tools" id ="toolsExtension" separatorBefore="true"
                label="Extension Items" offset="3"/>
            <MenuItem action="customTool2" customMenu="toolsExtension" showIcon="true"/>
            <MenuItem action="customTool1" systemMenu="file.models" showIcon="true"/>
            <ToolbarItem action="customTool1" offset="0"/>
        </Controls>
    </UserInterface>
</Extension>
```

这样做的结果是将名为**扩展项**的新子菜单添加到“工具”菜单中。该新子菜单有一个名为 **定制工具**的单个项。

您可以通过将 XML 代码保存在一个名为 *extension.xml* 的文件中来尝试本示例，然后将扩展名添加到 CLEF。有关更多信息，请参阅第 185 页的『测试 CLEF 扩展』主题。

Tabs

Tabs 部分定义了在以下位置可能出现的选项卡:

- 当用户打开工作区中的节点时显示的对话框
- 模型输出窗口
- 文档输出窗口

每个 Tabs 部分可以包含多个 Tab 元素，每个元素均可声明一个要显示的定制选项卡:

```
<Tabs defaultTab="integer">
    <Tab ... />
    <Tab ... />
     ...
</Tabs>
```

其中 defaultTab 是一个非负整数，可指定在流中首次打开节点对话框或窗口时要显示的选项卡。当该流处于活动状态时，如果用户在关闭或重新打开对话框或窗口时选择了其他选项卡，那么显示最近一次查看的选项卡而非缺省选项卡。选项卡从 0 开始计数。

请注意，其他选项卡可能自动包括在对话框或窗口上，例如，所有对话框和输出窗口中均包括 **注解**选项卡，而所有数据源节点对话框中均包括 **过滤器**和 **类型**选项卡。

Tab 元素必须声明选项卡标签（选项卡自身上显示的文本），而且还应包含标签键，以便在翻译选项卡标签时用于进行查找。

在每个 Tab 元素中有一个面板规范，可以定义选项卡主内容区域的布局方式。面板规范可为以下三种类型之一:文本浏览器、扩展对象或属性。有关更多信息，请参阅第 106 页的『面板规范』主题。

单个 Tab 元素的格式为:

```
<Tab id="identifier" label="display_label" labelKey="label_key" helpLink="help_ID"
        mnemonic="mnemonic_char" mnemonicKey="mnemonic_key" >
    <TextBrowserPanel ... />
```

```
    <ExtensionObjectPanel ... />
    <PropertiesPanel ... />
    <ModelViewerPanel ... />
</Tab>
```

其中:

id 是用于在 Java 代码中引用选项卡的唯一标识。

label（必需）是要在用户界面上显示的选项卡显示名称。

labelKey 用于标识标签以便进行本地化。

helpLink 是用户调用帮助系统时显示的帮助主题的标识（如果有的话）。标识的格式取决于帮助系统的类型（请参阅第 149 页的『定义帮助系统的位置和类型』）:

HTML Help: 帮助主题的 URL

JavaHelp: 主题标识

mnemonic 是与 Alt 键一起使用以激活此控件的字母符号（例如，如果您给其值为 S，那么用户可通过 Alt-S 激活此控件）。

mnemonicKey 用于标识助记符以便进行本地化。如果不使用 mnemonic 和 mnemonicKey，那么此控件无可用的助记符。有关更多信息，请参阅『访问键和键盘快捷键』主题。

示例

有关 Tab 元素的示例，请参阅不同类型面板规范上的各个部分，其中包括: 第 106 页的『文本浏览器面板』、第 107 页的『扩展对象面板』、第 108 页的『属性面板』 和 第 110 页的『模型查看器面板』。

访问键和键盘快捷键

作为鼠标访问用户界面上功能的替代方式，您可以指定各个按键组合以使用户能通过键盘访问各项功能。

访问键

访问键可与 Alt 键一起使用。对于菜单项、对话框选项卡和各种其他对话框控件，您可以通过使用以下元素的 mnemonic 属性指定访问键。

表 34. 用户界面上的功能部件.

功能	元素	请参阅...
屏幕操作（例如对于菜单项）	action	第 38 页的『Actions』
菜单	menu	第 99 页的『菜单』
对话框选项卡	tab	第 103 页的『Tabs』
控制器	各种	第 117 页的『控制器』

例如，以如下的菜单为例:

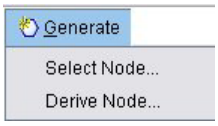

图 39. 菜单项

要指定此菜单的访问键，您可以使用以下代码:

```
<Actions>
     <Action id="generateSelect" label="Select Node..." labelKey="generate.selectNode.LABEL"
          imagePath="images/generate.gif" description="Generates a select node"
          descriptionKey="generate.selectNode.TOOLTIP" mnemonic="S" />
     <Action id="generateDerive" label="Derive Node..." labelKey="generate.deriveNode.LABEL"
          imagePath="images/generate.gif" description="Generates a derive node"
          descriptionKey="generate.deriveNode.TOOLTIP"  mnemonic="D" />
   ...
</Actions>
```

这将对菜单项中的字符加上下划线。

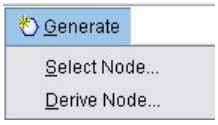

图 40. 使用访问键访问菜单项

用户现在可以通过 Alt-S 和 Alt-D 以及鼠标单击来访问菜单项。

快捷键

快捷键是使最终用户能在用户界面中导航的按键组合。IBM SPSS Modeler 提供多个标准的键盘快捷键。在 CLEF 中，您可以只对您添加的定制菜单项添加快捷键。

例如，要指定访问键示例中所示菜单的菜单项的快捷键，您可以使用以下代码:

```
<Actions>
     <Action id="generateSelect" label="Select Node..." labelKey="generate.selectNode.LABEL"
          imagePath="images/generate.gif" description="Generates a select node"
          descriptionKey="generate.selectNode.TOOLTIP" mnemonic="S" shortcut="CTRL+SHIFT+S" />
     <Action id="generateDerive" label="Derive Node..." labelKey="generate.deriveNode.LABEL"
          imagePath="images/generate.gif" description="Generates a derive node"
          descriptionKey="generate.deriveNode.TOOLTIP"  mnemonic="D"  shortcut="CTRL+SHIFT+D" />
     ...
</Actions>
```

快捷组合键现在添加到菜单项中。

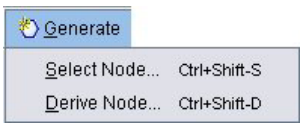

图 41. 使用快捷键访问菜单项

用户现在可以通过键盘快捷键以及鼠标单击来访问菜单项。可以同时指定快捷键和访问键，如此示例所示。

请注意，不要使用已在相同对话框上指定的快捷键，或任一标准 IBM SPSS Modeler 快捷键。

面板规范

每个 Tab 元素可以包含单个面板的规范，这些规范可以是下表显示的其中一种类型。

表 35. 面板规范类型

面板	显示...	请参阅...
文本浏览器面板	指定的容器的文本内容。	『文本浏览器面板』
扩展对象面板	由指定的 Java 类定义的内容。	第 107 页的『扩展对象面板』
属性面板	属性控件（例如按钮、复选框和输入字段）。	第 108 页的『属性面板』
模型查看器面板	来自指定容器的 PMML 格式的模型输出。	第 110 页的『模型查看器面板』

文本浏览器面板

文本浏览器面板显示扩展中指定容器的文本内容。受支持的格式（UTF-8 编码）有纯文本、HTML 和富文本格式 (RTF)。

以下示例显示包含 HTML 格式的文本浏览器面板的模型输出窗口。

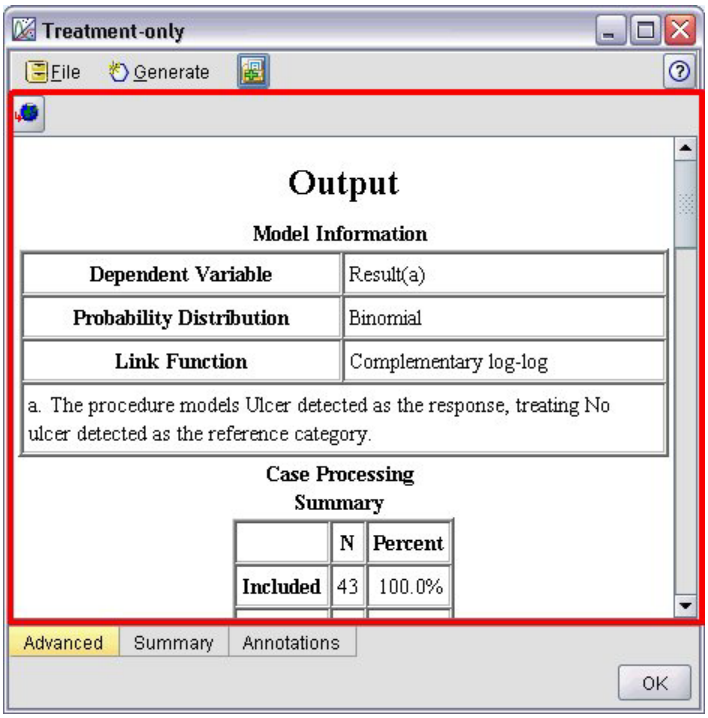

图 42. 突出显示文本浏览器面板的模型输出窗口

格式

```
<TextBrowserPanel container="name" textFormat="text_format" rows="integer"
     columns="integer" wrapLines="true_false" >
     -- advanced custom layout options --
</TextBrowserPanel>
```

其中:

container（必需）是可从中获取面板内容的容器的名称。

textFormat（必需）指定面板中所显示文本的格式，具体格式有:

- plainText
- html
- rtf

rows 和 columns 指定当包含面板的窗口打开时显示的文本行数和列数。

wrapLines 指定对长文本行是使用换行 (true)，还是需要水平滚动以阅读长文本行 (false)。缺省值为 false。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

以下示例演示了“选项卡”部分，该部分定义之前显示的文本浏览器面板:

```
<Tab label="Advanced" labelKey="advancedTab.LABEL" helpLink="genlin_output_advancedtab.htm">
     <TextBrowserPanel container="advanced_output" textFormat="html" />
</Tab>
```

将模型输出发送至容器，该容器在与选项卡规范相同的 ModelOutput 部分中定义。

```
<ModelOutput id="generalizedlinear" label="Generalized Linear">
     <Containers>
      ...
          <Container name="advanced_output"/>
     </Containers>
     <UserInterface>
      ...
          <Tabs>
               <Tab label="Advanced" ... >
                    <TextBrowserPanel container="advanced_output" ... />
               </Tab>
          </Tabs>
     </UserInterface>
</ModelOutput>
```

在“执行”部分的 CreateDocument 元素中，为相关构建节点引用该容器:

```
<Execution>
 ...
     <Constructors>
          <CreateModelOutput type="generalizedlinear">
               <CreateModel type="generalizedlinear_model" target="model" sourceFile="pmml"/>
               <CreateDocument type="html_output" target="advanced_output" sourceFile=
                "htmloutput"/>
          </CreateModelOutput>
     </Constructors>
</Execution>
```

扩展对象面板

扩展对象面板和文本浏览器面板的工作方式相似，但不显示容器的文本内容，而是创建特定 Java 类的实例，该 Java 类可实现由 CLEF Java API 定义的 ExtensionObjectPanel 界面。

以下示例显示包含扩展对象面板的模型应用节点对话框:

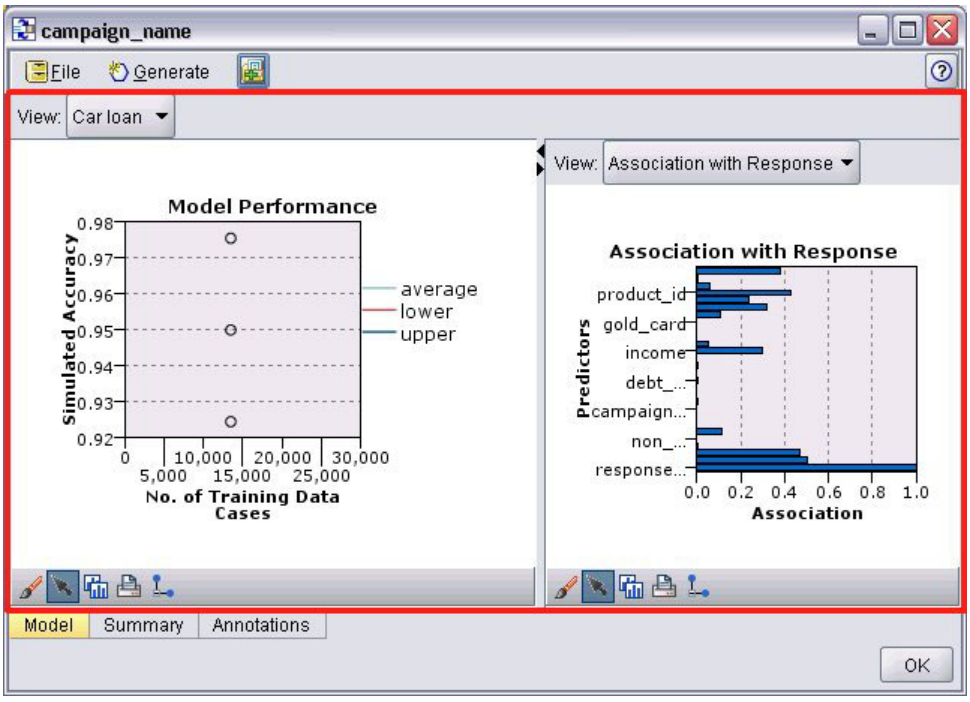

图 43. 突出显示扩展对象面板的模型输出窗口

格式

```
<ExtensionObjectPanel id="identifier" panelClass="Java_class" >
     -- advanced custom layout options --
</ExtensionObjectPanel>
```

其中:

`id` 是用于在 Java 代码中引用面板的唯一标识。

`panelClass`（必需）是面板实现的 Java 类的名称。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

以下示例演示了“选项卡”部分，该部分定义之前显示的扩展对象面板:

```
<Tab label="Model" labelKey="Model.LABEL" helpLink="selflearnnode_output.htm">
     <ExtensionObjectPanel id="SelfLearningPanel"
          panelClass="com.spss.clef.selflearning.SelfLearningPanel"/>
</Tab>
```

属性面板

属性面板允许选项卡或属性子面板（请参阅 第 115 页的『属性子面板』）显示 **属性控件**，这些控件是一些屏幕组件（如按钮、复选框和输入字段），可用于修改屏幕显示对象的属性。当用户单击 **确定**或 **应用**后，属性面板将自动应用使用这些控件所作的更改。当用户单击 **取消**或 **重置**后，面板将放弃自上次应用操作后进行的所有更改。

以下示例显示了包含属性面板的节点对话框。

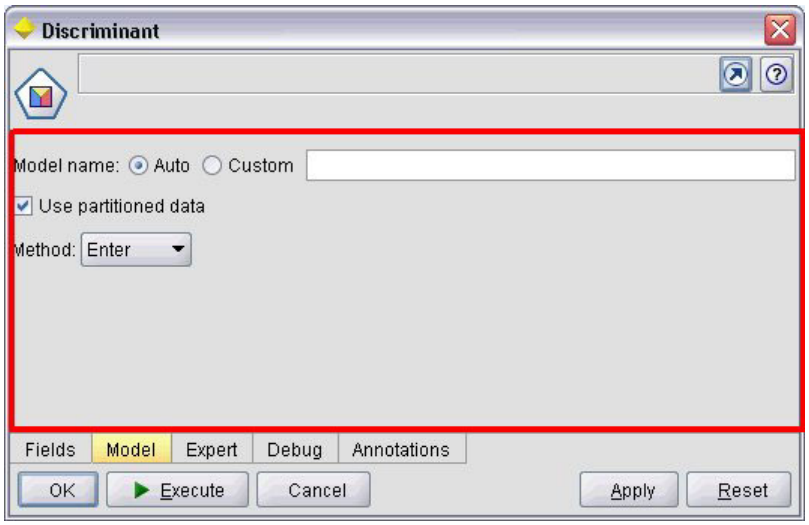

图 44. 突出显示了属性面板的节点对话框

下一示例演示了包含三个属性面板的属性子面板。

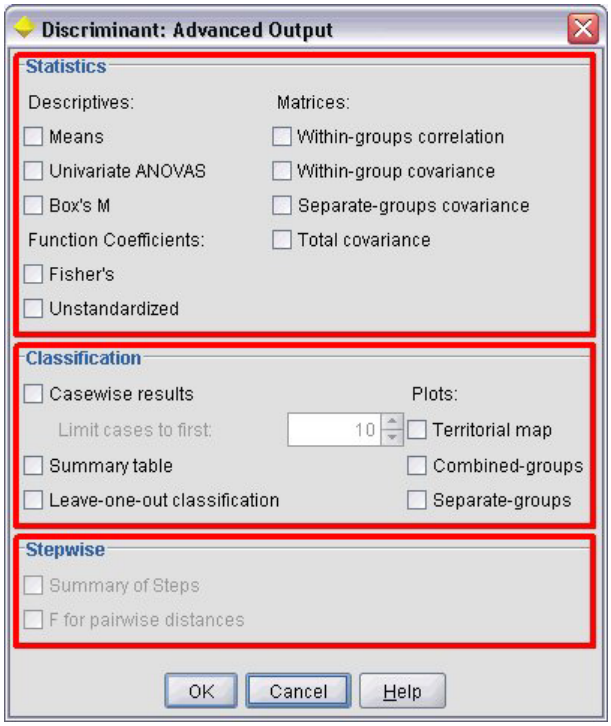

图 45. 突出显示了属性面板的属性子面板

格式

```
<PropertiesPanel id="identifier" label="display_label" labelKey="label_key">
     -- advanced custom layout options --
     -- property control specifications --
</PropertiesPanel>
```

其中:

id 是用于在 Java 代码中引用面板的唯一标识。

label 是属性控件组（例如上一示例中的**统计**、**分类**和**逐步**）的显示标题。

labelKey 用于标识标签以便进行本地化。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

有关各个属性控件规范的信息，请参阅 第 111 页的『属性控件规范』

示例

```
<Tab label="Model" labelKey="Model.LABEL" helpLink="discriminant_node_model.htm">
     <PropertiesPanel>
          <SystemControls controlsId="ModelGeneration" />
          <ComboBoxControl property="method">
               <Layout fill="none" />
          </ComboBoxControl>
     </PropertiesPanel>
</Tab>
```

模型查看器面板

模型查看器面板可显示扩展中指定容器的任意 PMML 格式的模型输出。

以下示例显示包含模型查看器面板的模型应用器节点窗口:

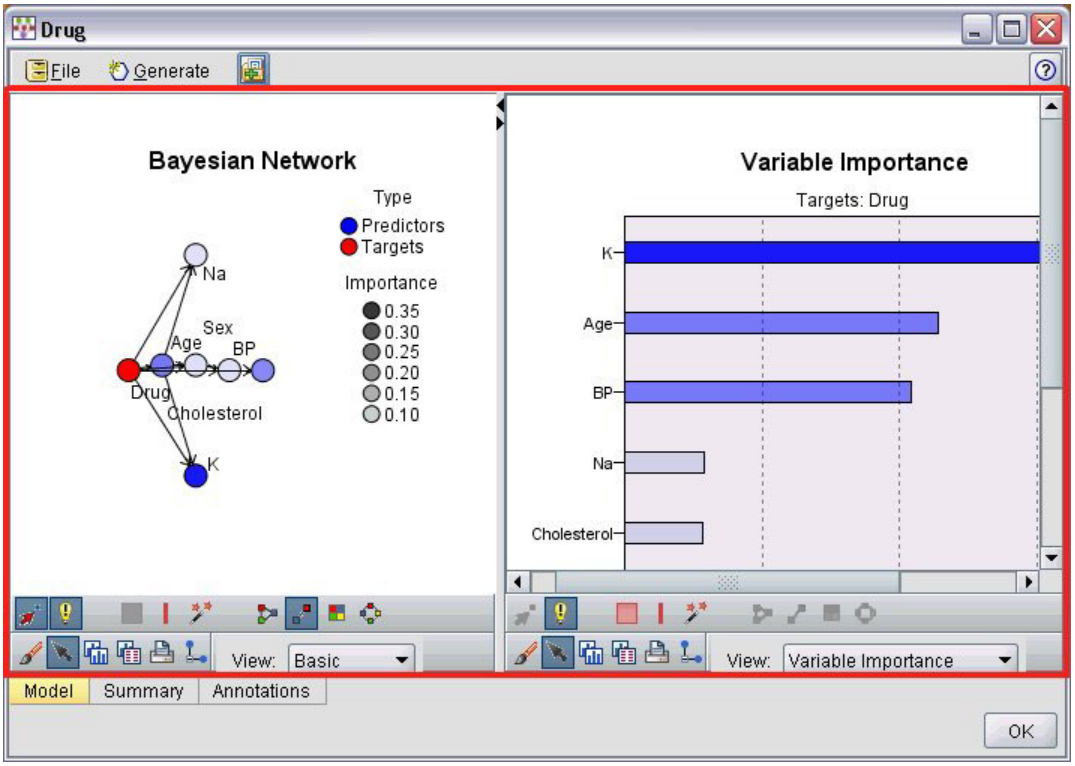

图 46. 突出显示模型查看器面板的输出窗口

格式

```
<ModelViewerPanel container="container_name">
     -- advanced custom layout options --
</ModelViewerPanel>
```

其中 container（必需）是向其中分配模型输出的容器的名称。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

本示例演示模型查看器面板在模型应用器规范中的用法。该模型输出之前已被分配到的名为 model 的容器中。此处模型应用器规范将选取此容器，并将其与模型查看器面板相关联:

```
<Node id="applyBN" type="modelApplier">
     <ModelProvider container="model" isPMML="true" />
      ...
     <Containers>
          <Container name="model" />
     </Containers>
     <UserInterface>
      ...
     <Tabs>
          <Tab label="Model" labelKey="modelTab.LABEL" helpLink="BN_output_modeltab.htm">
               <ModelViewerPanel container="model"/>
          </Tab>
      ...
     </Tabs>
     </UserInterface>
      ...
</Node>
```

属性控件规范

属性控件是指一些像按钮、复选框和输入字段这样的屏幕组件，可用于修改屏幕显示对象的属性。属性控件规范的格式由属性控件类型决定，可以是以下类型之一:

- UI 组件
- 属性面板
- 控制器

UI 组件控件可以是操作按钮、显示屏上的静态文本和系统控件（一组对所有对话框都通用的处理属性的控件）。

属性面板控件是指属性面板规范中的各个面板。

控制器可形成最大的属性控件组，并包含许多项，如复选框、组合框和微调控件等。

UI 组件控件

下表显示了 UI 组件控件。

表 36. UI 组件控件

控件	描述
ActionButton	屏幕按钮，单击时可执行预定义的操作。

表 36. UI 组件控件 (续)

控件	描述
StaticText	显示在屏幕上的非变量文本字符串。
SystemControls	处理所有模型通用属性的标准控件组。

操作按钮

定义可执行“公共对象”部分指定操作的对话框按钮或工具栏按钮。用户单击此按钮后，将执行该操作（例如，显示新屏幕）。

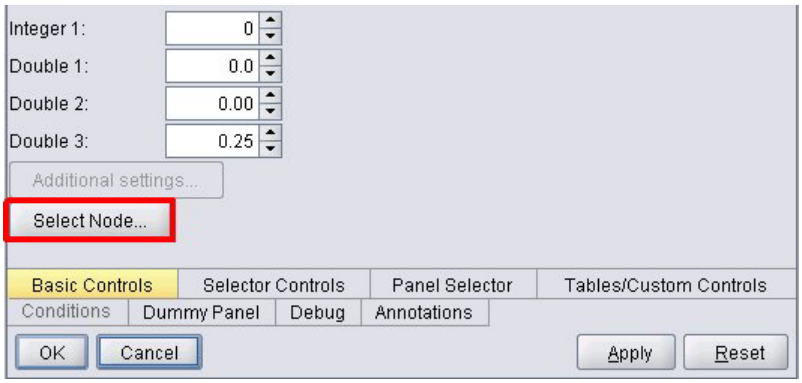

图 47. 突出显示操作按钮的对话框

格式

```
<ActionButton action="action" showLabel="true_false" showIcon="true_false" >
     -- advanced custom layout options --
</ActionButton>
```

其中:

action（必需）是要执行的操作的标识。

showLabel 指定是显示 (true) 按钮标签还是隐藏 (false) 按钮标签（例如，对于工具栏按钮，您可以选择显示图标而不是标签）。缺省值为 true。

showIcon 指定是显示 (true) 还是隐藏 (false) 与该按钮关联的图标。缺省值为 false。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

创建之前所显示的操作按钮的代码为:

```
<ActionButton action="generateSelect"/>
```

在“公共对象”部分定义该操作，如下所示（请注意，此处还定义了按钮标签）:

```
<CommonObjects extensionListenerClass="com.spss.cleftest.TestExtensionListener">
     ...
    <Actions>
      <Action id="generateSelect" label="Select Node..." labelKey="generate.selectNode.LABEL"
           imagePath="images/generate.gif" description="Generates a select node"
```

```
            descriptionKey="generate.selectNode.TOOLTIP"/>
        ...
    </Actions>
</CommonObjects>
```

静态文本

通过此元素，您可以在对话框或输出窗口中包含一个非变量文本字符串。以下示例演示了包含静态文本的属性面板:

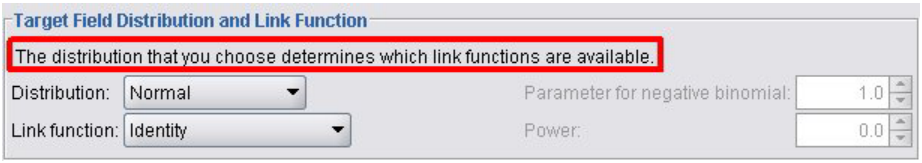

图 48. 突出显示静态文本的属性面板

格式

```
<StaticText text="static_text" textKey="text_key" >
     -- advanced custom layout options --
</StaticText>
```

其中:

`text` 是要使用的文本字符串。

`textKey` 用于标识标签以便进行本地化。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

以下示例显示之前显示的静态文本所使用的声明:

```
<StaticText text="The distribution that you choose determines which link functions are
     available." textKey="Genlin_staticText1"/>
```

系统控件

有些属性所有模型都通用。在模型构建器节点中，系统控件是处理这些属性的标准控件组。

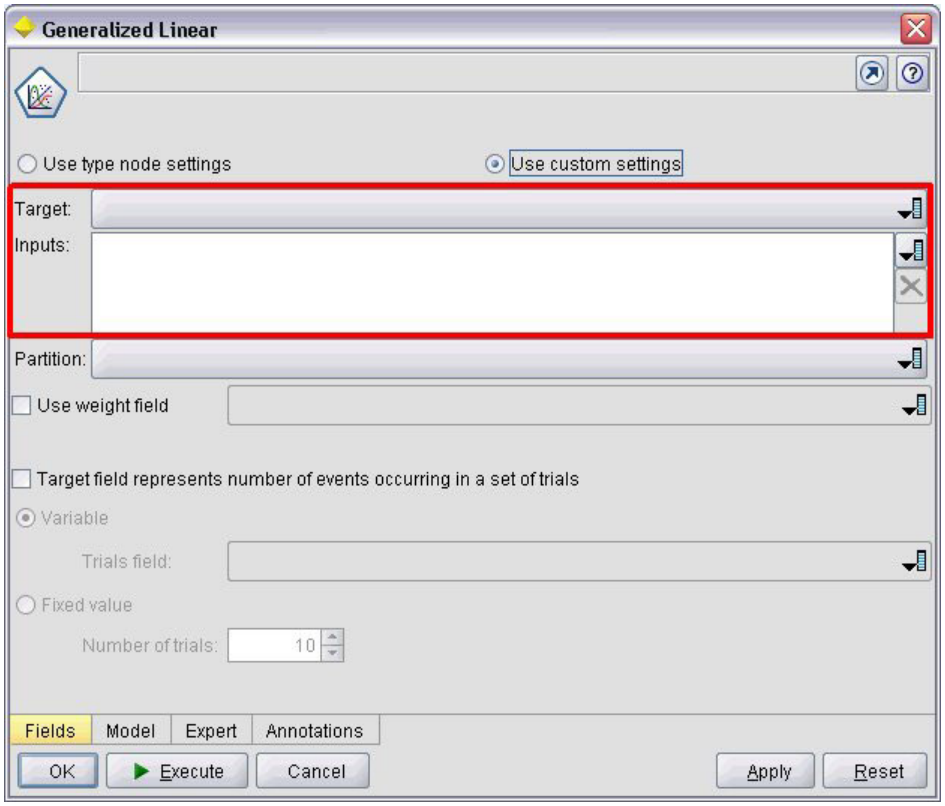

图 49. 突出显示系统控件的对话框示例

格式

```
<SystemControls controlsID="identifier" >
     -- advanced custom layout options --
</SystemControls>
```

其中 controlsID 是控制组的标识。此标识必须与在“模型构建器”声明中的 ModelingFields 元素的 controlsID 属性中指定的标识相同（请参阅第 48 页的『模型构建器』）。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

此示例显示用于上图中的系统控件的声明。

在节点规范的模型构建器部分，以下代码定义了一组系统控件，本示例中包含了模型输入输出字段的字段选取器:

```
<ModelBuilder ... >
     <ModelingFields controlsId="modelingFields">
          <InputFields property="inputs" multiple="true" label="Inputs" types="[range set
orderedSet flag]" labelKey="inputFields.LABEL"/>
          <OutputFields property="target" multiple="false" types="[range flag]"
label="Target" labelKey="targetField.LABEL"/>
     </ModelingFields>
      ...
</ModelBuilder>
```

在随后的文件中，该控件组将在它们所在的模型构建器节点对话框的选项卡定义中被引用:

```
<Tab label="Fields" labelKey="Fields.LABEL" helpLink="genlin_node_fieldstab.htm">
     <PropertiesPanel>
          <SystemControls controlsId="modelingFields">
          </SystemControls>
      ...
     </PropertiesPanel>
</Tab>
```

属性面板控件

下表显示了属性面板控件。

表 37. 属性面板控件

控件	描述
PropertiesSubPanel	在用户单击属性面板上的按钮时显示的独立对话框。
PropertiesPanel	嵌套在属性子面板声明或嵌套在顶层属性面板声明中的属性面板。

属性子面板

定义在用户单击属性面板上的按钮时显示的独立对话框。属性子面板声明将成为选项卡的主属性面板规范中的一部分。

格式

```
<PropertiesSubPanel buttonLabel="display_label" buttonLabelKey="label_key"
     dialogTitle="display_title" dialogTitleKey="title_key" helplink="help_ID"
     mnemonic="mnemonic_char" mnemonicKey="mnemonic_key" >
     -- advanced custom layout options --
     -- property control specifications --
</PropertiesSubPanel>
```

其中:

buttonLabel 是单击即可访问子面板的按钮的标签。

buttonLabelKey 用于标识按钮标签以便进行本地化。

dialogTitle 是显示在子面板对话框标题栏上的文本。

dialogTitleKey 用于标识子面板对话框标题以便进行本地化。

helpLink 是用户调用帮助系统时显示的帮助主题的标识（如果有的话）。标识的格式取决于帮助系统的类型（请参阅第 149 页的『定义帮助系统的位置和类型』）:

HTML Help: 帮助主题的 URL

JavaHelp: 主题标识

mnemonic 是与 Alt 键一起使用以激活此控件的字母符号（例如，如果您给其值为 S，那么用户可通过 Alt-S 激活此控件）。

mnemonicKey 用于标识助记符以便进行本地化。如果不使用 mnemonic 和 mnemonicKey，那么此控件无可用的助记符。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

有关各个属性控件规范的信息，请参阅 第 111 页的『属性控件规范』

示例

以下示例演示了当用户单击选项卡主属性面板上的 **输出**按钮时显示的属性子面板。

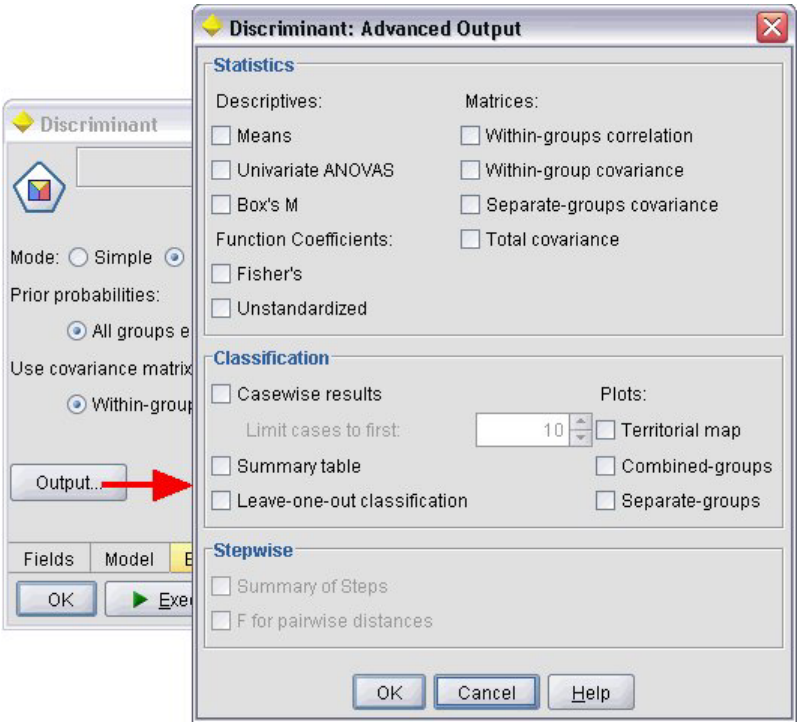

图 50. 属性子面板

以下代码显示了声明中最重要的部分，该声明可用于获取上一插图中的属性子面板。请注意，在子面板声明中，各个字段组（**统计量**、**分类**和**逐步**）都具有其自身的属性面板规范:

```
<PropertiesSubPanel buttonLabel="Output..." buttonLabelKey="OutputSubPanel.LABEL"
     dialogTitle="Discriminant: Advanced Output" dialogTitleKey="AdvancedOutputSubDialog.LABEL"
     helpLink="discriminant_node_outputdlg.htm">
     ...
     <PropertiesPanel>
          <PropertiesPanel label="Statistics" ... >
           ...
          </PropertiesPanel>
          <PropertiesPanel label="Classification" ... >
           ...
          </PropertiesPanel>
          <PropertiesPanel label="Stepwise" ... >
           ...
          </PropertiesPanel>
     </PropertiesPanel>
</PropertiesSubPanel>
```

属性面板（嵌套）

可以在属性子面板声明中嵌套属性面板规范，以定义从子面板中显示的对话框内容。有关更多信息，请参阅第 115 页的『属性子面板』主题。

也可以在顶层属性面板声明中嵌套属性面板规范。在以下情况下您可能想要执行此操作，根据是否已选择选项卡上的某个特定按钮来启用或禁用整个选项卡（包括多个属性面板）的内容。在此情况下，选项卡规范类似于以下格式:

```
<Tab .... >
     <PropertiesPanel>
      --- button specification ---
          <PropertiesPanel>
               <Enabled>
                --- condition involving button value ---
               </Enabled>
           ...
          </PropertiesPanel>
          <PropertiesPanel>
               <Enabled>
                --- condition involving button value ---
               </Enabled>
           ...
          </PropertiesPanel>
           ...
     </PropertiesPanel>
</Tab>
```

嵌套属性面板规范的格式与顶层元素的格式相同。有关更多信息，请参阅第 108 页的『属性面板』主题。

控制器

控制器组成最大的属性控件组。

表 38. 控制器

控件	描述
CheckBoxControl	复选框。
CheckBoxGroupControl	复选框组，每个 enum 值表示一个复选框。
ClientDirectoryChooserControl	单行文本字段和关联按钮，可允许用户选择客户端上的目录。
ClientFileChooserControl	单行文本字段和关联按钮，可允许用户选择客户端上的文件。
ComboBoxControl	包含 enum 值的下拉组合框。
DBConnectionChooserControl	允许用户选择数据源并连接到数据库。
DBTableChooserControl	允许用户在成功连接到数据库后选择数据库表。
MultiFieldChooserControl	（仅用于节点）允许用户从中选择一个或多个字段的字段名称列表。
MultiItemChooserControl	允许用户从值列表中选择一项或多项。
PasswordBoxControl	隐藏了输入字符的单行文本字段。
PropertyControl	用户可定义的属性控件。
RadioButtonGroupControl	单选按钮组，一次只能选择一个按钮。对于 enum 属性，每个 enum 值有一个单选按钮；对于布尔属性，那么显示两个单选按钮。
ServerDirectoryChooserControl	单行文本字段和关联按钮，可允许用户选择服务器上的目录。
ServerFileChooserControl	单行文本字段和关联按钮，可允许用户选择服务器上的文件。
SingleFieldChooserControl	（仅用于节点）允许用户从中选择单个字段的字段名称列表。

表 38. 控制器 (续)

控件	描述
SingleItemChooserControl	允许用户从值列表中选择单独一项。
SpinnerControl	微调控件（使用向上和向下箭头更改值的数字字段）。
TableControl	向对话框或窗口中添加表。
TextAreaControl	多行文本字段。
TextBoxControl	单行文本字段。

控制器属性

控制器规范可以包含以下属性:

```
property="value" showLabel="true_false" label="display_label" labelKey="label_key"
labelWidth="label_width" labelAbove="true_false" description="description"
descriptionKey="description_key" mnemonic="mnemonic_char" mnemonicKey="mnemonic_key"
```

其中:

property（必需）是属性控件的唯一标识。

showLabel 指定是显示 (true) 还是隐藏 (false) 属性控件的显示标签。缺省值为 true。

label 是要在用户界面上显示的属性控件显示名称。此值还可视为属性控件的简短辅助说明。有关更多信息，请参阅第 159 页的『辅助功能选项』主题。

labelKey 用于标识标签以便进行本地化。

labelWidth 是标签横跨显示网格列的数目。缺省值为 1。

labelAbove 指示控件标签是出现在控件上方 (true) 还是旁边 (false)。缺省值为 false。

description 是光标悬停在控件上时所显示的工具提示文本。此值还可作为控件的详细辅助说明。有关更多信息，请参阅第 159 页的『辅助功能选项』主题。

descriptionKey 用于标识描述以便进行本地化。

mnemonic 是与 Alt 键一起使用以激活此控件的字母符号（例如，如果您给其值为 S，那么用户可通过 Alt-S 激活此控件）。

mnemonicKey 用于标识助记符以便进行本地化。如果不使用 mnemonic 和 mnemonicKey，那么此控件无可用的助记符。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

复选框控件

定义复选框。

图 51. 复选框

格式

```
<CheckBoxControl controller_attributes invert="true_false" >
     -- advanced custom layout options --
</CheckBoxControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

invert 很少使用，但如果设置为 true，那么将反转复选框选中和取消选中的效果。缺省值为 false。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

以下示例显示用于排列前面所显示的复选框的代码（复选框标签已在规范文件的其他位置定义）。有关 Layout 元素的信息，请参阅 第 138 页的『高级定制布局』。

```
<CheckBoxControl property="means">
     <Layout rowIncrement="0" gridWidth="1" />
</CheckBoxControl>
<CheckBoxControl property="within_groups_correlation">
     <Layout gridColumn="2" />
</CheckBoxControl>
<CheckBoxControl property="univariate_anovas">
     <Layout gridWidth="1" rowIncrement="0" />
</CheckBoxControl>
<CheckBoxControl property="within_group_covariance">
     <Layout gridColumn="2" />
</CheckBoxControl>
<CheckBoxControl property="box_m">
     <Layout gridWidth="1" rowIncrement="0" />
</CheckBoxControl>
<CheckBoxControl property="separate_groups_covariance">
     <Layout gridColumn="2" />
</CheckBoxControl>
```

复选框组控件

定义分为一组并作为单个单元处理的复选框集。此控件只能与定义组成员的枚举列表属性一起使用。

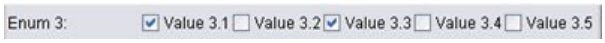

图 52. 复选框组

格式

```
<CheckBoxGroupControl controller_attributes rows="integer" layoutByRow="true_false"
     useSubPanel="true_false" >
     -- advanced custom layout options --
</CheckBoxGroupControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

rows 是一个正整数，可指定复选框组在屏幕上所占用的行数。缺省值为 1。

layoutByRow 指定复选框是先沿着行 (true) 还是先顺着列 (false) 排列。缺省值为 true。有关单选按钮组的 layoutByRow 的类似用法，请参阅 第 138 页的『更改控件顺序』。

useSubPanel 指定是 (true) 否 (false) 将复选框作为子面板进行放置。缺省值为 true。

复选框组通常被作为子面板进行放置，其中包含组中的所有复选框。但是，如果复选框组与相邻的文本字段相关联，那么可能会导致出现对齐问题。将 useSubPanel 设置为 false 可以解决此问题。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

创建之前所显示的复选框组的代码为:

```
<CheckBoxGroupControl property="enum3" label="Enum 3" labelKey="enum3.LABEL"/>
```

与各个复选框相关的标签和值在相应节点的“属性”部分进行定义:

```
<Property name="enum3" valueType="enum" isList="true" defaultValue="[value1 value3]">
     <Enumeration>
          <Enum value="value1" label="Value 3.1" labelKey="enum3.value1.LABEL"/>
          <Enum value="value2" label="Value 3.2" labelKey="enum3.value2.LABEL"/>
          <Enum value="value3" label="Value 3.3" labelKey="enum3.value3.LABEL"/>
          <Enum value="value4" label="Value 3.4" labelKey="enum3.value4.LABEL"/>
          <Enum value="value5" label="Value 3.5" labelKey="enum3.value5.LABEL"/>
     </Enumeration>
</Property>
```

客户端目录选择器控件

定义允许用户选择客户端上目录的单行文本字段和关联按钮。该目录必须已经存在。用户可以从该目录打开文件或将文件保存到该目录中，具体取决于模式设置。

图 53. 客户端目录选择器控件

用户可以直接在文本字段中输入目录路径和名称，也可以单击旁边的按钮从所显示的对话框中选择一个目录。

格式

```
<ClientDirectoryChooserControl controller_attributes mode="chooser_mode" >
     -- advanced custom layout options --
</ClientDirectoryChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

mode 确定用户可以从中选择目录的对话框中所显示的按钮，可以是以下值之一:

- open（缺省）显示**打开**按钮。
- save显示**保存**按钮。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<ClientDirectoryChooserControl property="directory2" label="Client Directory"
     labelKey="directory2.LABEL"/>
```

客户端文件选择器控件

定义允许用户选择客户端上文件的单行文本字段和关联按钮。该文件必须已经存在。用户可以打开该文件，也可以保存它，具体取决于模式设置。

图 54. 客户端文件选择器控件

用户可以在文本字段中直接输入文件路径和名称，也可以单击旁边的按钮从所显示的对话框中选择一个文件。

格式

```
<ClientFileChooserControl controller_attributes mode="chooser_mode" >
     -- advanced custom layout options --
</ClientFileChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

mode 确定用户可以从中选择文件的对话框中所显示的按钮，可以是以下值之一:

- open（缺省）显示**打开**按钮。
- save显示**保存**按钮。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<ClientFileChooserControl property="file2" label="Client File" labelKey="file2.LABEL"/>
```

组合框控件

定义组合框下拉列表。

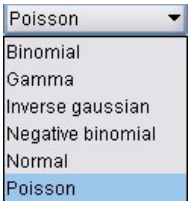

图 55. 组合框

格式

```
<ComboBoxControl controller_attributes >
     -- advanced custom layout options --
</ComboBoxControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

下面的示例显示用于设置上一插图中组合框下拉列表布局的代码:

```
<ComboBoxControl property="distribution" >
     <Layout rowIncrement="0" gridWidth="1" fill="none"/>
</ComboBoxControl>
```

有关 Layout 元素的信息，请参阅 第 138 页的『高级定制布局』。

注意: 实际的列表条目是在相应节点的"属性"部分定义的; 以下示例为 distribution 属性声明中的一个枚举列表:

```
<Property name="distribution" valueType="enum" label="Distribution" labelKey="distribution.
LABEL" defaultValue="NORMAL">
     <Enumeration>
          <Enum value="BINOMIAL" label="Binomial" labelKey="distribution.BINOMIAL.LABEL"/>
          <Enum value="GAMMA" label="Gamma" labelKey="distribution.GAMMA.LABEL"/>
          <Enum value="IGAUSS" label="Inverse gaussian" labelKey="distribution.IGAUSS.LABEL"/>
          <Enum value="NEGBIN" label="Negative binomial" labelKey="distribution.NEGBIN.LABEL"/>
          <Enum value="NORMAL" label="Normal" labelKey="distribution.NORMAL.LABEL"/>
          <Enum value="POISSON" label="Poisson" labelKey="distribution.POISSON.LABEL"/>
     </Enumeration>
</Property>
```

数据库连接选择器控件

定义使用户可以选择数据源并连接到数据库的控件。

图 56. 数据库连接选择器控件

用户不能在该文本字段中输入文本，他们必须单击按钮才能显示标准的 IBM SPSS Modeler 数据库连接对话框。

成功连接后，连接详情会显示在数据库连接选择器控件的文本字段中。

格式

```
<DBConnectionChooserControl controller_attributes >
     -- advanced custom layout options --
</DBConnectionChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

以下示例演示控件需要如何定义字符串属性，该字符串属性可以用于连接字符串。

```
<Node ... >
     <Properties>
      ...
          <Property name="dbconnect" valueType="databaseConnection" />
     </Properties>
      ...
     <UserInterface>
      ...
          <Tabs>
               <Tab label="Database">
                    <PropertiesPanel>
                         <DBConnectionChooserControl property="dbconnect" label="Database
                          Connection"/>
                     ...
                    </PropertiesPanel>
                     ...
     </UserInterface>
</Node>
```

数据库表选择器控件

定义使用户在成功连接数据库后可以选择数据库表的文本字段和关联按钮。

图 57. 数据库表选择器控件

用户可以在文本字段中直接输入表名，也可以单击旁边的按钮并从列表中选择所需的表名。

格式

```
<DBTableChooserControl connectionProperty="DB_connection_property" controller_attributes >
     -- advanced custom layout options --
</DBTableChooserControl>
```

其中:

`connectionProperty` 是已经定义的数据库连接属性的名称。这是先前为该节点定义的 `DBConnectionChooserControl` 元素的 `property` 属性值。

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

本示例是 `DBConnectionChooserControl` 的一个后续示例，显示如何再包含一个 `DBTableChooserControl` 元素，以在成功建立数据库连接后选择数据库表。

```
<Node ... >
     <Properties>
      ...
          <Property name="dbconnect" valueType="databaseConnection" />
          <Property name="dbtable" valueType="string" />
     </Properties>
       ...
     <UserInterface>
```

```
      ...
          <Tabs>
               <Tab label="Database">
                    <PropertiesPanel>
                         <DBConnectionChooserControl property="dbconnect" label="Database
                          connection"/>
                         <DBTableChooserControl property="dbtable" connectionProperty=
                          "dbconnect" label="Database Table" />
                     ...
                    </PropertiesPanel>
      ...
     </UserInterface>
</Node>
```

多字段选择器控件

定义允许用户从列表中选择一个或多个字段名的控件。

图 58. 多字段选择器控件

用户单击此控件时，会显示一个字段列表，用户可以从中选择一个或多个字段。

此设置组成了可在此节点中看到的所有字段。如果字段从此节点上游进一步过滤，那么只有通过过滤器的字段才是可见的。该列表可以进一步对之进行限制，其方式是指定仅可选择特定存储和数据类型的字段。

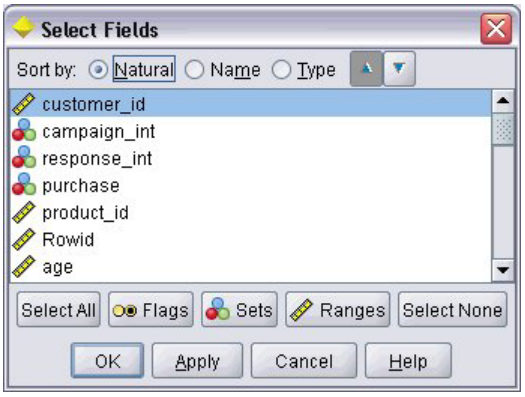

图 59. 多字段列表

每一个多字段选择器控件均可指定一个属性特性（该属性特性在文件的其他位置进行声明），并且可定义列表在节点对话框中的显示方式。

格式

```
<MultiFieldChooserControl controller_attributes storage="storage_types" onlyNumeric="true_false"
     onlySymbolic="true_false" onlyDatetime="true_false" types="data_types"
     onlyRanges="true_false" onlyDiscrete="true_false" >
     -- advanced custom layout options --
</MultiFieldChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

另外，还可以通过指定另外两个属性来进一步限制字段列表，其中一个属性必须是下列属性之一:

- storage 是列表属性，该属性指定在列表中所允许的字段的存储类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。有关可能存储类型的设置的信息，请参阅第 169 页的『数据和存储类型』 下的表格。
- onlyNumeric 如果设置为 true，那么指定只列出存储类型为数字的字段。
- onlySymbolic 如果设置为 true，那么指定只列出存储类型为符号（即字符串）的字段。
- onlyDatetime 如果设置为 true，那么指定只列出存储类型为日期时间的字段。

另一个指定的属性 必须来自下表:

- types 是列表属性，该属性指定在列表中所允许的字段的数据类型，例如，storage="[integer real]" 表示只列出上述存储类型的字段。可能的数据类型的设置如下:

  范围

  标志

  set

  orderedSet

  numeric

  discrete

  typeless
- onlyRanges 如果设置为 true，那么指定只列出数据类型为范围的字段。
- onlyDiscrete 如果设置为 true，那么指定只列出数据类型为离散（即标志、设置或无类型）的字段。

因此，例如一个指定 storage="[integer]" 和 types="[flag]" 的控制确保只有类型为标志的整数字段会显示在列表中。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

注意: 此控件仅用于节点元素定义。要在输出数据模型定义中指定一个多字段选择器，请使用以下格式:

```
<OutputDataModel mode="mode">
 ...
    <ForEach var="field" inProperty="prop_name">
         <AddField name="${field_name}_NEW" fieldRef="${field_name}" />
    </ForEach>
 ...
</OutputDataModel>
```

有关更多信息，请参阅第 55 页的『输出数据模型』主题。有关 ForEach 元素的信息，请参阅第 63 页的『使用 ForEach 元素执行迭代』。第 60 页的『添加字段』对 AddField 作了描述。

示例

下面的示例显示用于指定上图中的多字段选择器控件的代码:

```
<MultiFieldChooserControl property="inputs" >
     <Enabled>
          <Condition control="custom_fields" op="equals" value="true"/>
     </Enabled>
</MultiFieldChooserControl>
```

Enabled 部分使该控件仅在选择了 custom_fields 控件后才能启用。

注意: 此列表的内容受相应节点属性部分的 inputs 属性声明的控制。

```
<Property name="inputs" valueType="string" isList="true" label="Inputs" labelKey="inputs.
LABEL"/>
```

多项目选择器控件

定义允许用户从值列表中选择一个或多个项目的控件。将属性与包含值列表的目录相关联。有关更多信息，请参阅第 39 页的『Catalogs』主题。

图 60. 多项目选择器控件

格式

```
<MultiItemChooserControl controller_attributes catalog="catalog_name" >
     -- advanced custom layout options --
</MultiItemChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

catalog（必需）是要关联的目录的名称。从中获取目录的库在“执行”部分的 Module 元素中指定。有关更多信息，请参阅第 53 页的『模块』主题。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<MultiItemChooserControl property="selection2" catalog="cat2" />
```

property 属性（在本例中为 selection2）引用的属性必须具有 isList="true" 特性。有关 MultiItemChooserControl 用法的解释和示例，请参阅第 39 页的『Catalogs』。

密码框控件

定义可以隐藏输入字符的单行文本字段。

图 61. 密码框控件

格式

```
<PasswordBoxControl controller_attributes columns="integer" >
     -- advanced custom layout options --
</PasswordBoxControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

columns 是一个正整数，定义密码框应占用的字符列数。缺省值为 20。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<PasswordBoxControl property="encrypted_string1" label="Encrypted string 1" labelKey=
"encryptedString1.LABEL"/>
```

可以采用与相应节点的属性部分中定义为加密字符串的属性相关联这样的方式对文本字段加密:

```
<Property name="encrypted_string1" valueType="encryptedString"/>
```

属性控件

属性控件是一个完全可由用户来定义的控件，通过该控件用户可以输入节点的属性。处理操作由用户编写的 Java 类执行。下图是属性控件的示例。

图 62. 突出显示属性控件示例的对话框部分

格式

```
<PropertyControl controller_attributes controlClass="Java_class" >
     -- advanced custom layout options --
</PropertyControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

controlClass（必需）是 *.jar* 文件内指向实现属性控件的 Java 类的路径。（*注意*: *.jar* 文件在“资源”部分的 JarFile 元素内声明。有关更多信息，请参阅第 33 页的『Jar 文件』主题。）

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<PropertyControl property="target_field_values_specify" labelAbove="true"
     controlClass="com.spss.clef.selflearning.propertycontrols.list.CustomListControl" label=""
     labelKey="target_field_values_specify.LABEL">
     <Enabled>
          <Condition control="target_field_values" op="equals" value="specify"/>
     </Enabled>
     <Layout rowIncrement="2" />
</PropertyControl>
```

属性控件与相应节点的属性部分定义的一个属性相关联:

```
<Property name="target_field_values_specify" valueType="string" isList="true" label=""
     labelKey="target_field_values_specify.LABEL"/>
```

单选按钮组控件

定义一组单选按钮，一次仅能选择一个按钮。

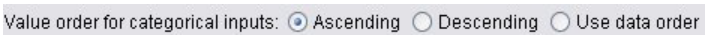

图 63. 单选按钮组控件

每一个单选按钮组控件均有一个将该组与特定属性相关联的属性特性。此属性是在文件的其他位置定义的，可指定组成该组的按钮。

关联的属性可以是一个枚举列表，也可以是一个布尔属性。对于枚举列表（其中属性特性为 `valueType="enum"`），显示的每一个单选按钮均对应于一个 `enum` 值。对于布尔属性（其中 `valueType="boolean"`），始终显示两个单选按钮。

格式

```
<RadioButtonGroupControl controller_attributes
     rows="integer" layoutByRow="true_false" useSubPanel="true_false"
     falseLabel="button_label" falseLabelKey="label_key" trueLabel="?button_label"
     trueLabelKey="label_key" trueFirst="true_false" >
     -- advanced custom layout options --
</RadioButtonGroupControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

`rows` 是一个正整数，可指定单选按钮组显示时要占用的屏幕行数。缺省值为 1。

`layoutByRow` 指定单选按钮是先沿着行 (`true`) 还是先顺着列 (`false`) 排列。缺省值为 `true`。有关单选按钮组的 `layoutByRow` 用法的示例，请参阅 第 138 页的『更改控件顺序』。

`useSubPanel` 指定是 (`true`) 否 (`false`) 将单选按钮作为子面板进行放置。缺省值为 `true`。

单选按钮通常被作为子面板放置，其中包含组中的所有按钮。但是，如果复选框组与相邻的文本字段相关联，那么可能会导致出现对齐问题。将 `useSubPanel` 设置为 `false` 可以解决此问题。

`falseLabel` 是布尔属性“false”值的标签（请参阅下面的第二个示例）。仅能与布尔属性一起使用，在此情况下它是必需的。

`falseLabelKey` 用于标识“false”标签以便进行本地化。

trueLabel 是布尔属性“true”值的标签（请参见下面的第二个示例）。仅能与布尔属性一起使用，在此情况下它是必需的。

trueLabelKey 用于标识“true”标签以便进行本地化。

trueFirst，如果设置为 true，会反转布尔属性按钮的显示顺序，因此会先显示对应于“true”值的按钮。缺省值为 false，意味着会先显示对应于“false”值的按钮。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

第一个示例演示之前显示的单选按钮组所使用的代码。

```
<RadioButtonGroupControl property="value_order" labelWidth="2">
     <Layout gridWidth="4"/>
</RadioButtonGroupControl>
```

有关 Layout 元素的信息，请参阅 第 138 页的『高级定制布局』。

注意: 按钮数及其标签是在相应节点的属性部分定义的；以下示例为 value_order 属性声明中的一个枚举列表。此声明还包括单选按钮组本身的标签:

```
<Property name="value_order" valueType="enum" label="Value order for categorical
   inputs" labelKey="value_order.LABEL">
     <Enumeration>
          <Enum value="Ascending" label="Ascending" labelKey="value_order.Ascending.LABEL"/>
          <Enum value="Descending" label="Descending" labelKey="value_order.Descending.
           LABEL"/>
          <Enum value="DataOrder" label="Use data order" labelKey="value_order.UseDataOrder.
           LABEL"/>
     </Enumeration>
</Property>
```

第二个示例演示如何对控制布尔属性的单选按钮组使用 falseLabel 和 trueLabel，如一个控制启用标准还是定制设置。

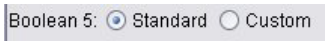

图 64. 控制布尔属性的单选按钮组

实现上述目的的代码是:

```
<RadioButtonGroupControl property="boolean5" label="Boolean 5" labelKey="boolean5.LABEL"
     falseLabel="Standard" falseLabelKey="boolean5.false.LABEL" trueLabel="Custom"
     trueLabelKey="boolean5.true.LABEL" />
```

在这种情况下，在 RadioButtonGroupControl 元素本身中定义按钮标签和组标签。在节点的属性部分中定义与组相关联的属性:

```
<Property name="boolean5" valueType="boolean" defaultValue="false"/>
```

服务器目录选择器控件

定义允许用户选择服务器上目录的单行文本字段和关联按钮。该目录必须已经存在。用户可以从该目录打开文件或将文件保存到该目录中，具体取决于模式设置。

图 65. 服务器目录选择器控件

用户可以直接在文本字段中输入目录路径和名称，也可以单击旁边的按钮从所显示的对话框中选择一个目录。

格式

```
<ServerDirectoryChooserControl controller_attributes mode="chooser_mode" >
     -- advanced custom layout options --
</ServerDirectoryChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

mode 确定用户可以从中选择目录的对话框中所显示的按钮，可以是以下值之一:

- open（缺省）显示**打开**按钮。
- save显示**保存**按钮。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<ServerDirectoryChooserControl property="directory1" label="Server Directory"
     labelKey="directory1.LABEL"/>
```

服务器文件选择器控件

定义允许用户选择服务器上文件的单行文本字段和关联按钮。该文件必须已经存在。用户可以打开该文件，也可以保存它，具体取决于模式设置。

图 66. 服务器文件选择器控件

用户可以在文本字段中直接输入文件路径和名称，也可以单击旁边的按钮从所显示的对话框中选择一个文件。

格式

```
<ServerFileChooserControl controller_attributes mode="chooser_mode" >
     -- advanced custom layout options --
</ServerFileChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

mode 确定用户可以从中选择文件的对话框中所显示的按钮，可以是以下值之一:

- open（缺省）显示**打开**按钮。
- save显示**保存**按钮。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<ServerFileChooserControl property="file1" label="Server File" labelKey="file1.LABEL"/>
```

单字段选择器控件

定义允许用户从列表中选择单个字段的控件。

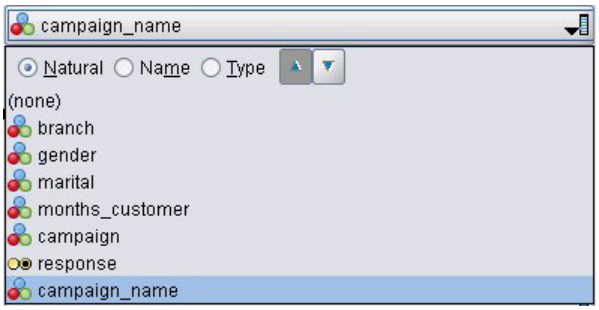

图 67. 单字段选择器控件

当用户单击此控件时，会显示一个字段列表，他们可以从该列表中选择单个字段。

此设置组成了可在此节点中看到的所有字段。如果字段从此节点上游进一步过滤，那么只有通过过滤器的字段才是可见的。该列表可以进一步对之进行限制，其方式是指定仅可选择特定存储和数据类型的字段。

格式

```
<SingleFieldChooserControl controller_attributes storage="storage_types" onlyNumeric="true_
     false" onlySymbolic="true_false" onlyDatetime="true_false" types="data_types"
     onlyRanges="true_false" onlyDiscrete="true_false" >
     -- advanced custom layout options --
</SingleFieldChooserControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

另外，还可以通过指定另外两个属性来进一步限制字段列表，其中一个属性必须是下列属性之一:

- `storage` 是列表属性，该属性指定在列表中所允许的字段的存储类型，例如，`storage="[integer real]"` 表示只列出上述存储类型的字段。有关可能存储类型的设置的信息，请参阅第 169 页的『数据和存储类型』 下的表格。
- `onlyNumeric` 如果设置为 `true`，那么指定只列出存储类型为数字的字段。
- `onlySymbolic` 如果设置为 `true`，那么指定只列出存储类型为符号（即字符串）的字段。
- `onlyDatetime` 如果设置为 `true`，那么指定只列出存储类型为日期时间的字段。

另一个指定的属性 必须来自下表:

- `types` 是列表属性，该属性指定在列表中所允许的字段的数据类型，例如，`storage="[integer real]"` 表示只列出上述存储类型的字段。可能的数据类型的设置如下:

  范围

  标志

  `set`

orderedSet

numeric

discrete

typeless

- onlyRanges 如果设置为 true，那么指定只列出数据类型为范围的字段。
- onlyDiscrete 如果设置为 true，那么指定只列出数据类型为离散（即标志、设置或无类型）的字段。

因此，例如一个指定 storage="[integer]" 和 types="[flag]" 的控制确保只有类型为标志的整数字段会显示在列表中。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

注意：此控件仅用于节点定义。要在输出数据模型定义中指定一个多字段选择器，请使用以下格式：

```
<OutputDataModel mode="mode">
 ...
    <ForEach var="field" from="1" to="${integer}">
         <AddField name="${string}_${field}" fieldRef="${field_ref}" />
    </ForEach>
 ...
</OutputDataModel>
```

有关更多信息，请参阅第 55 页的『输出数据模型』主题。有关 ForEach 元素的信息，请参阅第 63 页的『使用 ForEach 元素执行迭代』。第 60 页的『添加字段』对 AddField 作了描述。

示例

下面的示例显示用于指定上图中的单字段选择器控件的代码：

```
<SingleFieldChooserControl property="target"  storage="string" onlyDiscrete="true"/>
```

注意：列表的实际内容是在相应节点的属性部分定义的；在本示例中位于 target 属性的声明中：

```
<Property name="target" valueType="string" label="Target field" labelKey="target.LABEL"/>
```

单项目选择器控件

定义允许用户从值列表中选择单个项目的控件。将属性与包含值列表的目录相关联。有关更多信息，请参阅第 39 页的『Catalogs』主题。

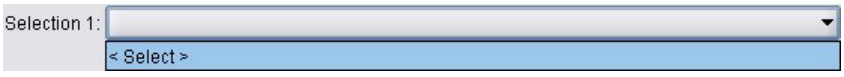

图 68. 单项目选择器控件

格式

```
<SingleItemChooserControl controller_attributes catalog="catalog_name" >
     -- advanced custom layout options --
</MultiItemChooserControl
```

其中：

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

catalog（必需）是要关联的目录的名称。从中获取目录的库在“执行”部分的 Module 元素中指定。有关更多信息，请参阅第 53 页的『模块』主题。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

```
<SingleItemChooserControl property="selection1" catalog="cat1" />
```

有关该控件用法的解释和示例，请参阅第 39 页的『Catalogs』。

微调控件

定义微调框（使用上下箭头更改字段值的数字字段）。

图 69. 微调框

格式

```
<SpinnerControl controller_attributes columns="integer" stepSize="increment"
     minDecimalDigits="number" maxDecimalDigits="number" >
     -- advanced custom layout options --
</SpinnerControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

columns 是一个正整数，指定控件所跨越的字符列数。缺省值为 5。

stepSize 是一个十进制数，指定用户单击其中一个箭头时字段值的更改量。缺省值为 1.0。

minDecimalDigits 是字段值所显示的最小小数位数。缺省值为 1。

maxDecimalDigits 是字段值所显示的最大小数位数。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

下面的示例显示用于指定上图中的旋钮控件的代码:

```
<SpinnerControl property="double1" label="Double 1" labelKey="double1.LABEL"/>
```

该数字字段内容的精确度和有效范围是在相应节点的属性部分定义的；在本示例中位于 double1 属性的声明中:

```
<Property name="double1" valueType="double" min="0" max="100"/>
```

表控件

定义要显示在节点对话框或输出窗口中的表布局项目。

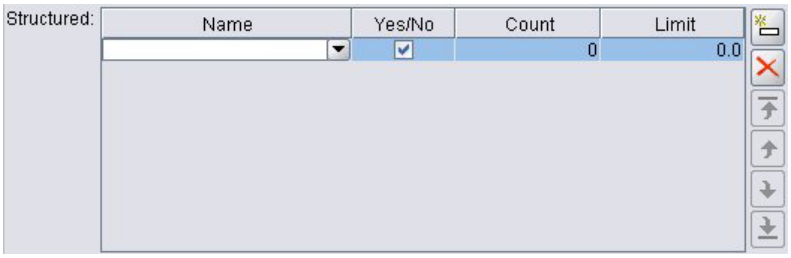

图 70. 表控件

格式

```
<TableControl controller_attributes rows="integer" columns="integer" columnWidths="list" >
     -- advanced custom layout options --
</TableControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

`rows` 是一个正整数，指定屏幕上显示的表行数。缺省值为 8。

`columns` 是一个正整数，指定表所跨越的字符列数。缺省值为 20。

`columnwidths` 是指定相应列宽的一列值。因此，如果值为 `[30 5 10]`，那么说明第 1 列的宽度是第 3 列的三倍。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

第 135 页的『列控件』对下例中的 *ColumnControl* 属性做了描述。

示例

用于指定上图中的表控件的代码如下所示:

```
<TableControl property="structure1" allowReorder="true" label="Structured"
  labelKey="structure1.LABEL" columnWidths="[20 6 10 10]">
     <ColumnControl column="0" editor="fieldValue" fieldControl="field1"/>
</TableControl>
```

表控件的结构在规范文件的“公共对象”部分定义为属性类型:

```
<PropertyType id="shared_structure1" valueType="structure" isList="true">
     <Structure>
          <Attribute name="id" valueType="string" label="Name" labelKey="structure1.id.LABEL"/>
          <Attribute name="yesno" valueType="boolean" label="Yes/No" labelKey="structure1.
           yesno.LABEL" defaultValue="true"/>
          <Attribute name="count" valueType="integer" label="Count" labelKey="structure1.
           count.LABEL" defaultValue="0"/>
          <Attribute name="limit" valueType="double" label="Limit" labelKey="structure1.
           limit.LABEL" defaultValue="0.0"/>
     </Structure>
</PropertyType>
```

在节点规范中，此属性类型的标识随后通过属性声明与表控件的标识相关联:

```
<Property name="structure1" type="shared_structure1"/>
```

如果引用脚本中的节点，那么可以使用方括号 [] 为列表设置属性值，或使用大括号 {} 为结构设置属性值。例如，可以为 structure1 属性设置一个由两个结构组成的网格:

```
set :node_ID.structure1 = [{"hello" true 4 0.21} {"bye" false 5 0.95}]
```

请注意，值的顺序必须与 Attribute 的定义顺序一致。

列控件: 定义表中的列的布局。

表控件的所有列共享同一数据类型；基于此，您可以针对某列指定一个编辑器用于所有行。因此，每列只需一个编辑器进行编辑。例如，如果列 X 需要用户输入整数，那么您可以针对列 X 设置整数编辑器。

属性 *editor* 指定列的编辑器类型。编辑器有四种类型: *default*、*field*、*fieldValue* 和 *enumeration*，每种编辑器都是可编辑的组合框。

对于 *fieldValue* 类型编辑器，下拉列表中将包含您在 *fieldControl* 中指定的字段的所有值。因此，下列 XML 元素定义: 当您编辑列 0 时，编辑器将为组合框，下拉列表中将包含 field1 的所有值:

```
<ColumnControl column="0" editor="fieldValue" fieldControl="field1" />
```

您可以使用 *fieldDirection* 替换 *fieldControl*。例如: *fieldDirecton="[in out]"* 表示组合框下拉列表中将包含第一个方向为 *in* 或 *out* 的字段的所有值。

对于 *field* 类型编辑器，下拉列表中将包含满足字段过滤条件的所有字段。下例定义: 列 0 将使用字段组合框作为编辑器，下拉列表中将包含所有实数和整数字段:

```
<ColumnControl column="0" editor="field" storage="[real integer]" />
```

另外，您还可以使用属性 *types* 来指定下拉列表中的字段应遵从的度量类型。适用于字段的布尔属性有: *onlyRanges*、*onlyDiscrete*, *onlyNumeric*, *onlySymbolic* 和 *onlyDatetime*。

文本区域控件

定义多行文本输入字段。

图 71. 文本区域控件

格式

```
<TextAreaControl controller_attributes rows="integer" columns="integer" wrapLines="true_false" >
     -- advanced custom layout options --
</TextAreaControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

rows 是一个正整数，指定文本区域占用的屏幕行数。缺省值为 8。

columns 是一个正整数，指定文本区域所跨越的字符列数。缺省值为 20。

wrapLines 指定对长文本行是使用换行 (true)，还是需要水平滚动以阅读长文本行 (false)。缺省值为 true。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

用于创建之前所示示例的代码如下所示:

```
<TextAreaControl property="string2" label="String 2" labelKey="string2.LABEL"/>
```

在本示例中，文本区域的标签定义在文本区域控件声明中，而输入数据类型则定义在相关节点的属性部分，位于 string2 属性的声明中:

```
<Property name="string2" valueType="string"/>
```

文本框控件

定义单行文本输入字段。

图 72. 文本框控件

格式

```
<TextBoxControl controller_attributes columns="integer" >
     -- advanced custom layout options --
</TextBoxControl>
```

其中:

第 118 页的『控制器属性』对 *controller_attributes* 作了描述。

columns 是一个正整数，指定文本框所跨越的字符列数。缺省值为 20。

高级定制布局选项对屏幕组件的配置和显示提供了精度控制。有关更多信息，请参阅第 138 页的『高级定制布局』主题。

示例

用于创建之前所示文本框的代码如下所示:

```
<TextBoxControl property="string1" label="String 1" labelKey="string1.LABEL"/>
```

文本框的输入数据类型在相应节点的属性部分定义；在本示例中位于 string1 属性的声明中:

```
<Property name="string1" valueType="string"/>
```

属性控件布局

本节介绍用于对话框和窗口的标准布局方法，以及对标准布局方法进行修改以获得定制布局的方法。

标准控件布局

可将属性面板视为单元格的二维网格。每一行均可以有不同的高度，每一列均可以有不同的宽度。UI 组件可以分配给多个连续的单元格，但通常情况下每个单元格仅分配一个 UI 组件。

缺省情况下，将一个属性控件分配给一行，且每一个控件占两列：其中一列用于标签，另一列用于控件组件。包含标签的列宽度会扩展到最宽标签的宽度。例如，在规范文件中给出以下元素:

```
<TextBoxControl property="string1" label="String 1"/>
<PasswordBoxControl property="encryptedString1" label="Encrypted string 1"/>
<TextAreaControl property="string2" label="String 2"/>
```

生成的面板如下图所示。

图 73. 简单的属性面板

注意: 标签末尾的":"字符是自动添加的。

包含多个用户界面组件的属性控件创建自己的不可视矩形区域以放置这些组件。RadioButtonGroupControl 和 CheckBoxGroupControl 元素是这种组件的示例。

注意，根据属性控件，放置组件的矩形区域形状可能有所不同。因此，不同控件的布局可能不会总是严格对齐。

一些属性控件包括完全填充组件列的组件，当窗口宽度放大或缩小时水平调整大小。例如，由 TextBoxControl、PasswordBoxControl 和 TextAreaControl 元素指定的控件。但是，并非所有组件都可以执行此操作。例如，复选框和微调控件仅占用固定的水平空间量，甚至当窗口宽度放大时也是如此:

定制控件布局

可以采用多种方式修改控件的标准布局，其中有些方法比较简单，而有些方法比较复杂。

简单定制布局

定制控件布局有以下三种简单方法:

- 将标签放置到其组件的上方
- 更改控件占用的行数
- 更改控件的放置顺序

将标签放置到其组件的上方: 可以通过将控件的 labelAbove 属性设置为 true 来将标签放置到其组件上方的单独一行。例如:

```
<TextBoxControl property="string0" label="String 0" labelAbove="true"/>
<TextBoxControl property="string1" label="String 1"/>
<PasswordBoxControl property="encryptedString1" label="Encrypted string 1"/>
```

将标签放置到其组件上方的同时，实际的 UI 组件或组件会被分配到标签显示列。这会生成如下所示的面板，其中标签 **String 0** 将显示在其对应字段的上方:

图 74. 字段标签位于单独行的面板

更改行数: 缺省情况下，在单行中放置单选按钮和复选框组，并调整对话框的宽度以容纳它们。如果单选按钮或复选框组有大量选项，可能导致对话框非常宽。您可以通过更改用于放置控件的行数来避免这一情况。通过将控件定义的 rows 属性设置为所需的值，可以实现上述配置。例如:

```
<RadioButtonGroupControl property="enum1" label="Enum 1" rows="2"/>
```

这会生成如下所示的面板，其中单选按钮组占用了两行。

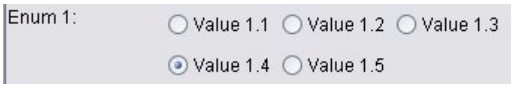

图 75. 单选按钮组占用两行的面板

更改控件顺序: 对于单选按钮组和复选框组，还可以针对每一个 enum 值更改控件添加到面板的顺序。

缺省情况下，按行添加控件，如上一示例所示，前三个值添加到了第一行，而第四、第五个值添加到了第二行。另外，可以通过将 layoutByRow 设置为 false 来在指定的行数内按列添加控件。例如:

```
<RadioButtonGroupControl property="enum1" label="Enum 1" rows="2" layoutByRow="false"/>
```

值仍然显示在两行中，但现在第一和第二个值添加到了第一列，第三和第四个值添加到了第二列，第五个值添加到了第三列:

图 76. 按列放置单选按钮组的面板

对于以两个单选按钮显示的布尔属性，缺省的顺序行为是“False”按钮在前，“True”按钮在后。可以通过将 trueFirst 属性设置为 true 来颠倒此顺序。

还可以通过将 useSubPanel 属性设置为 false 来禁止单选按钮组和复选框组使用子面板。但是，这会产生一些不理想的布局行为，除非上述用法与 Layout 元素一起使用（请参阅 『使用 Layout 元素指定精确的控件位置』）。

高级定制布局

在每个控件声明中，您可以使用各种元素指定复杂的控件布局。可以进行下列操作:

- 使用 Layout 元素指定控件在屏幕上的精确位置
- 使用 Enabled 元素控制显示特征
- 使用 Visible 元素控制屏幕组件的可见性

使用 Layout 元素指定精确的控件位置: 通过指定明确的 Layout 元素并将其与控件相关联可以实现精确的布局定位。

格式

```
<property_control ... >
     <Layout attributes
      --- cell specification ---
          ...
</property_control>
```

其中:

property_control 是属性控件之一（请参阅第 111 页的『属性控件规范』）。

attributes 是下表中显示的任何属性。

表 39. 布局属性.

属性	值(V)	描述
anchor	north northeast east southeast south southwest west northwest center	定义控件的定位点。
columnWeight	0.0-1.0	定义窗口在水平方向发生的大小变化会如何影响控件的宽度。面板中所有 columnWeight 属性的总和不应超过 1.0。
fill	none horizontal vertical both	定义控件是否以及如何填充分配给它的单元格。
gridColumn	integer >= 0	定义开始放置控件的第一列。
gridHeight	integer	定义放置控件时控件应占用的行数。值 0（缺省值）会跨所有剩余的行分配控件。
gridRow	integer >= 0	定义放置控件的第一行。缺省情况下，网格行索引会自动增加。
gridWidth	integer	定义放置控件时控件应占用的列数。值 0（缺省值）会跨所有剩余的列分配控件。
leftIndent	integer	定义控件相对于其缺省位置的缩进像素数。
rowWeight	0.0-1.0	定义窗口在垂直方向发生的大小变化会如何影响控件的高度。面板中所有 rowWeight 属性的总和不应超过 1.0。

通过 **cell specification** 可以指定控件在屏幕上的精确位置。格式如下所示:

```
<Cell row="integer" column="integer" width="integer" />
```

其中:

row（必需）是一个非负整数，可指定控件的起始行位置。

column（必需）是一个非负整数，可指定控件的起始列位置。

width（必需）是一个非负整数，可指定控件占用的屏幕网格列数。

因此，假设有一个三列三行的屏幕网格，以下面的格式定制控件布局。

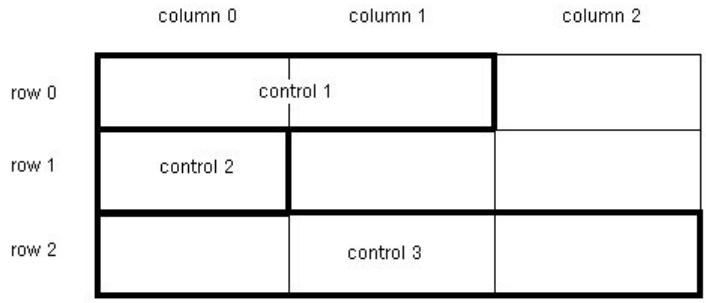

图 77. 使用单元格的控件布局示例

此格式需要 Layout 元素具有以下单元格规范:

```
<Layout ... >
     <Cell row="0" column="0" width="2">
     <Cell row="1" column="0" width="1">
     <Cell row="2" column="0" width="3">
</Layout>
```

下面有一些详细的示例将进一步说明如何使用 Layout 元素。

示例: 启用文本字段的复选框: 本示例演示如何在使用复选框时使文本字段位于相同的显示行中。

使用复选框时要使另一控件位于相同的行中，需要使用一个简单的 Layout 元素以正确显示该控件。（*注意*: 有关启用和禁用控件的机制，请参阅第 145 页的『使用 Enabled 元素控制显示特征』）。

假设我们希望实现以下面板显示结果。

图 78. 启用文本字段的复选框

这里有两个控件:

- 一个带标签的复选框，该标签也是文字字段的标签
- 文本字段本身

开始部分是对这两个控件进行常规声明:

```
<CheckBoxControl property="boolean3" label="Check box 3"/>
<TextBoxControl property="string3" label="String 3"/>
```

将显示以下面板。

图 79. 复选框和文本字段位于单独的行中

首先，我们希望禁止显示文本字段标签 **String 3** 。这可通过将文本字段控件的 showLabel 属性设置为 false 来实现:

```
<CheckBoxControl property="boolean3" label="Check box 3"/>
<TextBoxControl property="string3" label="String 3" showLabel="false"/>
```

文本字段将扩展以填充标签先前占用的区域。

图 80. 复选框和不带标签的文本字段

现在，我们希望在同一行显示文本字段和复选框。为此，我们在 CheckBoxControl 元素内添加一个 Layout 元素，以将行增量设置为 0（缺省情况下，每一个控件的行增量为 1）：

```
<CheckBoxControl property="boolean3" label="Check box 3">
     <Layout rowIncrement="0"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false"/>
```

但是，显示结果如下所示。

图 81. 文本字段覆盖了复选框

文本字段向上移动了一行，但它仍然占用了整个行，因此覆盖了复选框。

注意: 如果显示内容如下所示，那么复选框将显示在文本字段之后。

图 82. 复选框覆盖了文本字段

文本字段的前几个字符被覆盖了。

无论先绘制哪一个对象，将多个 UI 组件分配到同一个单元格都会得到不理想的行为或未定义的行为，应加以避免。要解决此问题，我们需要在 TextBoxControl 元素内添加第二个 Layout 元素，以强制从第二个显示列开始显示文本字段:

```
<CheckBoxControl property="boolean3" label="Check box 3">
     <Layout rowIncrement="0"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false">
     <Layout gridColumn="1"/>
</TextBoxControl>
```

但是，该解决方案仅解决了部分问题。两个控件的位置放置正确，但文本字段太短，如下所示。

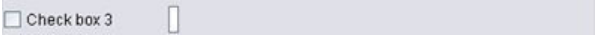

图 83. 位置正确但文本字段非常短

问题是定制布局与控件关联后，会覆盖与每种类型的控件相关联的“智能”缺省值。在此情况下，Layout 元素的缺省填充行为（即组件填充其可用单元格的方式）将不填充可用的单元格，而是占用尽可能少的屏幕空间。为改变此情况，我们只需让文本字段去填充水平空间:

```
<CheckBoxControl property="boolean3" label="Check box 3">
     <Layout rowIncrement="0"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

需要增加一个较小的 columnWeight 值以使 Java 正确分配填充的空间。

这样我们就会得到预期的布局。

图 84. 启用文本字段的复选框

以上布局表面看上去没有错误，但仍然有一个问题需要解决。当前，复选框试图占用整个行，即使我们现在在同一行中放置了另外一个控件。当前看不到该问题，因为复选框标签相对比较短，面板上的其他标签（插图中未显示）已经移出了第二个显示列，因此不存在重叠现象。如果将复选框标签变长，该问题就会变得很明显:

```
<CheckBoxControl property="boolean3" label="Check box 3 with a much longer label than we had">
     <Layout rowIncrement="0"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

这样做会得到以下显示结果。

图 85. 文本字段覆盖长复选框标签

我们唯一需要做的是让复选框将其可用的宽度限制到单一列中:

```
<CheckBoxControl property="boolean3" label="Check box 3 with a much longer label than we had">
     <Layout rowIncrement="0" gridWidth="1"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

这样就可以得到我们最终想要的结果。

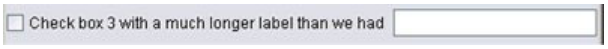

图 86. 长复选框标签完整显示

示例: 单选按钮组和文本字段: 本示例演示将单选按钮组中的每个按钮与其自身的文本字段相关联的方式。

我们想定义一个如下所示的面板。

图 87. 具有文本字段的单选按钮组

这次我们有四个控件:

- 一个单选按钮组，用于一个包含三个值的枚举列表
- 三个文本字段，每个文本字段对应一个值

与前面的示例一样，我们首先对这些控件进行简单声明:

```
<RadioButtonGroupControl property="enum4" label="Enum 4"/>
<TextBoxControl property="string4" label="String 4"/>
<TextBoxControl property="string5" label="String 5"/>
<TextBoxControl property="string6" label="String 6"/>
```

这样做会得到以下显示结果。

图 88. 具有文本字段和标签的单选按钮组

我们希望使用单选按钮标签标识文本字段，因此我们的第一个任务是在一个具有三行的单列中排列单选按钮，并隐藏文本字段标签:

```
<RadioButtonGroupControl property="enum4" label="Enum 4" rows="3"/>
<TextBoxControl property="string4" label="String 4" showLabel="false"/>
<TextBoxControl property="string5" label="String 5" showLabel="false"/>
<TextBoxControl property="string6" label="String 6" showLabel="false"/>
```

显示结果如下所示。

图 89. 单选按钮和文本字段位于同一列中

这里我们已经可以看到一个小小的问题，即单选按钮组标签未与第一个单选按钮对齐。稍后我们将修复此问题，现在我们需要使文本字段与每个对应的单选按钮大致位于一行。

操作步骤与我们在示例 1 中执行的操作类似。我们需要执行以下操作:

- 将单选按钮组的行增量更改为 0。
- 限制网格宽度，使文本字段与单选按钮不重叠。
- 将每个文本字段与其对应的单选按钮放置到同一行。

因此，就像在上一示例中执行的操作一样，我们添加一些 Layout 元素。在本示例中，我们按如下所示更改规范文件:

```
<RadioButtonGroupControl property="enum4" label="Enum 4" rows="3">
     <Layout rowIncrement="0" gridWidth="1"/>
</RadioButtonGroupControl>
<TextBoxControl property="string4" label="String 4" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string5" label="String 5" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string6" label="String 6" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

不幸的是，我们现在得到以下显示。

图 90. 文本字段覆盖了单选按钮

如果我们使用与示例 1 完全相同的 Layout 元素，会得到什么样的结果呢？

答案是，与上一示例中的复选框控件不同，单选按钮组（与大多数控件一样）有单独的标签和实际控件。这意味着单选按钮组需要额外的列，因此我们需要从后面的列（第 2 列而不是第 1 列）开始放置文本字段。这样在文本字段的 Layout 元素中，我们将 gridColumn 的值设置为 2:

```
<RadioButtonGroupControl property="enum4" label="Enum 4" rows="3">
     <Layout rowIncrement="0" gridWidth="1"/>
</RadioButtonGroupControl>
<TextBoxControl property="string4" label="String 4" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string5" label="String 5" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string6" label="String 6" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

请注意，虽然我们将文本字段网格列增加到了 2，但我们并没有增加单选按钮组的网格宽度 1。这是由于对于属性控件，大多数 Layout 属性仅影响组成控件可编辑区域的 UI 组件，而不影响控件标签。

我们现在得到以下显示。

图 91. 文本字段不再覆盖单选按钮

这已经非常接近于我们想要的结果。但是，单选按钮和文本字段之间仍然存在一些对齐问题。

主要问题是单选按钮放置到了单独的子面板中，因此单选按钮与其文本字段之间不存在实际的布局关系。我们唯一需要做的是停止单选按钮组使用子面板:

```
<RadioButtonGroupControl property="enum4" label="Enum 4" rows="3" useSubPanel="false">
     <Layout rowIncrement="0" gridWidth="1"/>
</RadioButtonGroupControl>
<TextBoxControl property="string4" label="String 4" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string5" label="String 5" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
<TextBoxControl property="string6" label="String 6" showLabel="false">
     <Layout gridColumn="2" fill="horizontal" columnWeight="0.001"/>
</TextBoxControl>
```

最后，我们得到了所需的布局。

图 92. 具有文本字段的单选按钮组

使用 Enabled 元素控制显示特征: 通常可以使用 Enabled 元素依据是否满足特定的条件来启用或禁用控件。

面板和属性控件可以拥有与其相关联的条件，以确定各种显示特征。例如，复选框可用于启用关联的文本字段，单选按钮可使另一组隐藏的字段变成可见。

用户界面中的条件通常取决于另一控件的值，而非属性的值。基于属性的条件仅在所作的更改应用到基础对象（例如节点、模型输出或文档输出）时才起作用。在用户界面中，控件需要在相关控件发生变化后立即启用。

格式

```
<Enabled>
     <Condition .../>
     <And ... />
     <Or ... />
     <Not ... />
</Enabled>
```

Condition 元素指定要测试的以确定是否启用控件的条件。

通过 And、Or 和 Not 元素可以指定复合条件。

有关更多信息，请参阅第 67 页的『条件』主题。

示例: 使用简单条件启用控件: 在 第 140 页的『示例: 启用文本字段的复选框』中，我们开发了一个复选框，选择该框时可以启用一个文本字段。

我们希望在复选框被选中后立即启用文本字段，而不是在基础对象的属性发生变化时启用文本字段。为此，我们需要添加一个 Enabled 条件:

```
<CheckBoxControl property="boolean3" label="Check box 3 with a much longer label than we had">
     <Layout rowIncrement="0" gridWidth="1"/>
</CheckBoxControl>
<TextBoxControl property="string3" label="String 3" showLabel="false">
     <Layout gridColumn="1" fill="horizontal" columnWeight="0.001"/>
```

```
        <Enabled>
             <Condition control="boolean3" op="equals" value="true"/>
        </Enabled>
</TextBoxControl>
```

这可确保文本字段仅在与复选框关联的布尔值为真时才启用。

示例: 使用复杂条件启用控件: 为了说明复杂条件的编码，我们将观察用 CLEF 开发的“广义线性”节点的一个对话框选项卡。

该节点对话框有一个“专家”选项卡，其中包含专为用户设计的选项，选项里是这类模型的详细信息。该选项卡上的所有选项最初都是禁用的。

将 **模式**复选框设置为**专家**可启用其中一些选项。

但是，仍有些选项被禁用，例如对话框底部的**迭代**控件。只有同时满足以下 **两个**条件时才启用此控件:

- **分布**设置为**正态**
- **关联函数**设置为**恒等**

此组合是“专家”模式中此选项卡的实际缺省设置，更改任意这两个组合框的设置都会启用**迭代**。

实现上述目的的代码包含在**迭代**按钮的 PropertiesSubPanel 声明中，如下所示:

```
<PropertiesSubPanel buttonLabel="Iterations..." buttonLabelKey= ...
     <Enabled>
          <And>
               <Condition control="mode" op="equals" value="Expert"/>
                    <Not>
                         <And>
                              <Condition control="distribution" op="equals" value="NORMAL"/>
                              <Condition control="link_function" op="equals" value="IDENTITY"/>
                         </And>
                    </Not>
          </And>
     </Enabled>
      ...
</PropertiesSubPanel>
```

外层 And 部分中的 Condition 元素指定在进行任何更改之前必须将**模式**设置为**专家**。假设满足此条件，那么 Not 部分指定仅在同时满足内层 And 部分的*两个*条件时才禁用此按钮。因此，在“专家”模式中，如果**分布**或**连接函数**的设置值与其缺省值不同，那么启用**迭代**。

使用 Visible 元素控制显示特征: 还可以根据指定的情况使用条件显示或隐藏控件。通过 Visible 元素可实现上述目的。

格式

```
<Visible>
     <Condition .../>
     <And ... />
     <Or ... />
     <Not ... />
</Visible>
```

Condition 元素指定要测试的以确定是否显示控件的条件。

通过 And、Or 和 Not 元素可以指定复合条件。

有关更多信息，请参阅第 67 页的『条件』主题。

示例

下面的示例演示仅在满足 source_language 条件时才显示指定的属性面板:

```
<PropertiesPanel>
     <Visible>
          <Condition control="source_language" op="equals" value="eng" />
     </Visible>
       ...
</PropertiesPanel>
```

定制输出窗口

对于模型输出、文档输出和交互式输出对象（而非节点），扩展时可以使用定制窗口完全代替缺省的输出窗口。这是作为标准的 java.awt.Frame 类实现的.

要提供定制窗口，请指定一个 Java 类作为 UserInterface 元素的 frameClass 属性，如下所示:

```
<DocumentOutput id="my_modelling_node" type="modelBuilder" ...>
     <Properties>
          <Property name="use_custom_type" valueType="boolean" .../>
           ...
     </Properties>
     <UserInterface frameClass="com.myextension.MyOutputFrame"/>
      ...
  </DocumentOutput>
```

所指定的类必须实现通过 CLEF 客户端 API 定义的 ExtensionObjectFrame 接口。这包含窗口的生命周期:

- 访问基础 java.awt.Frame
- 初始化窗口，包括访问输出对象和会话
- 在要保存或删除对象时同步窗口和基础对象
- 窗口处理

有关更多信息，请参阅第 161 页的『客户端 API 类』主题。

第 7 章 添加帮助系统

帮助系统的类型

开发 CLEF 扩展时，通常希望包含联机帮助系统以说明如何使用此扩展。 CLEF 支持下列类型的帮助系统:

- HTML 帮助
- JavaHelp

HTML 帮助

HTML 帮助是 Microsoft 开发的专有格式，只能在 Windows 平台上运行。HTML 帮助系统由单个 .htm 或 .html 文件编译成压缩格式，形成扩展名为 .chm 的独立文件。IBM SPSS Modeler 自己的帮助系统以 HTML 帮助格式提供。

HTML 帮助支持目录、索引和全文搜索功能（词汇表可通过弹出窗口的形式实现）。可以使用 HTML 编辑器或商用帮助创作工具创建 .htm 或 .html 的源主题文件。要生成 .chm 文件，可以选择使用 HTML Help Workshop，该工具可从“Microsoft 下载中心”网站免费下载（有关生成 .chm 文件的详细信息，请参见 HTML Help Workshop 的帮助系统）。也可以使用支持 HTML 帮助格式的帮助创作工具将所用的主题文件和所有图形文件编译为 .chm 文件。

JavaHelp

JavaHelp 是一种开放式源代码帮助格式，由 Sun Microsystems 开发，可以在任何支持 Java 的平台上运行。JavaHelp 系统中包含以下文件:

- .htm 或 .html 的源主题文件
- 主题中使用的所有图形文件
- 控制帮助系统的帮助集文件（扩展名为 .hs）
- map.xml 文件，用于将主题标识和主题文件相关联，并定义显示帮助主题的窗口
- index.xml 文件，其中包含索引条目
- toc.xml 文件，其中包含目录条目

JavaHelp 支持目录、索引、全文搜索和词汇表功能。可以使用 HTML 编辑器或商用帮助创作工具创建 .htm 或 .html 的源文件。您将需要 JavaHelp 软件，可从 Sun Developer Network 网站免费下载该工具（有关详细信息，请参见 *JavaHelp System User's Guide*，此文档也可以从该网站获取。）

实现帮助系统

本节介绍如何在规范文件中定义帮助系统的相关组件。

定义帮助系统的位置和类型

用于扩展的帮助系统类型（如果有）在扩展的规范文件“资源”部分的 HelpInfo 元素中定义。

格式

```
<Resources>
 ...
     <HelpInfo id="name" type="help_type" path="help_path" helpset="helpset_loc"
      default="default_topicID" />
 ...
</Resources>
```

其中:

id（必需）是此扩展的帮助信息的标识。

type（必需）指示帮助类型，这是下列其中一项:

- htmlhelp - HTML 帮助，包含在 path 属性所标识的 .chm 文件中。
- javahelp - JavaHelp，使用帮助集（.hs）文件（由 helpset 属性标识）和帮助源文件及相关文件。

如果帮助类型为 htmlhelp，那么需要下列附加属性:

- path - 这是包含帮助系统的 .chm 文件的位置（相对于规范文件）和名称。

如果帮助类型为 javahelp，那么需要下列属性:

- helpset - 要使用的 .hs 帮助集文件的位置（相对于规范文件）和名称。
- default - 在没有为特定选项卡指定主题时，要显示的缺省主题的标识。

如果未指定 HelpInfo 元素，那么没有帮助与此扩展关联。

示例

第一个示例说明 HTML 帮助的 HelpInfo 元素:

```
<HelpInfo id="help" type="htmlhelp" path="help/mynode.chm" />
```

JavaHelp 系统的等价元素是:

```
<HelpInfo id="help" type="javahelp" helpset="help/mynode.hs"/>
```

请注意，如果使用 JavaHelp，关联文件（图像、映射文件、索引和内容文件）必须与 .hs 帮助集文件位于同一文件夹中。

指定要显示的特定帮助主题

您可以指定用户在节点对话框、特定选项卡或属性子面板中调用帮助时要显示的特定帮助主题。要实现该功能，可使用节点、选项卡或属性子面板规范的 helpLink 属性。

如果未指定 helpLink 属性，在用户调用帮助时将显示帮助系统的缺省主题。

有关详细信息，请参见 第 45 页的『节点』、第 103 页的『Tabs』和 第 115 页的『属性子面板』中对 helpLink 属性的介绍。

示例

本示例假设您要使用 HTML 帮助，介绍了如何操作才能显示不同的上下文主题，具体取决于用户选择帮助时选中的窗口。

```
<Resources>
     ...
    <HelpInfo id="help" type="htmlhelp" path="help/mynode.chm"/>
     ...
```

```
</Resources>
 ...
<Node id="mynode" scriptName="my_node" type="dataTransformer" palette="recordOp"
  label="Sorter" description="Sorts a data file" >
  ...
     <Tabs defaultTab="1">
          <Tab label="Basic Controls" labelKey="basicControlsTab.LABEL"
            helpLink="basic_controls.htm">
              <PropertiesPanel>
               ...
                   <PropertiesSubPanel buttonLabel="Additional settings..."
                     buttonLabelKey="AdditionalOptions.LABEL" dialogTitle="Additional
                     Settings" dialogTitleKey="AdditionalOptionsDialog.LABEL" helpLink=
                     "addsettingsdlg.htm">
  ...
          </Tab>
          <Tab label="Selector Controls" labelKey="selectorControlsTab.LABEL"
           helpLink="selector_controls.htm">
  ...
          </Tab>
  ...
</Node>
```

这指定了: 如果焦点位于“基本控制”选项卡时用户选择帮助，那么显示 mynode.chm 帮助文件中 basic_controls.htm 中的主题。如果用户随后单击**其他设置**按钮打开“其他设置”对话框，并选择该对话框中的**帮助**，将显示 addsettingsdlg.htm 中的主题。如果用户随后关闭“其他设置”对话框，而选择“选择器控件”选项卡，并再次选择**帮助**，将显示 selector_controls.htm 中的主题。

对于 JavaHelp，helpLink 属性的值必须与 map.xml 文件中 target 属性的值相匹配。例如，如果 map.xml 文件包含下列内容:

```
<map version="1.0">
 ...
     <mapID target="basic_controls" url="basic_controls.htm"/>
 ...
</map>
```

必须为相应的 helpLink 属性赋予下列值:

```
helpLink="basic_controls"
```

这是因为当 JavaHelp 被调用时，它将读取 target 属性的值并将其映射到相关的 url 值，以找到要显示的正确文件。

第 8 章 本地化和辅助功能选项

介绍

本地化是指调整软件、帮助和文档，使其适应特定语言环境的过程。这包括在相应语言环境中翻译用户界面、帮助和文档以及测试系统。如果您要将扩展部分分发给非本地区域的用户，应该分发扩展部分的本地化版本。

在此上下文中，**辅助功能**是指在用户界面中包括相应的功能，以使身体有残疾（例如视力欠佳或手灵活度受限）的用户能够更方便地访问系统。

本地化

IBM SPSS Modeler 自身已针对全球众多地区进行本地化。对于所有受支持的语言版本，如果用户将 Windows 区域选项设置为自己所在的区域，那么标准 IBM SPSS Modeler UI 组件将以相应的语言显示，例如:

- 系统菜单和菜单条目
- 系统按钮（“生成”、“确定”、“执行”、“取消”、“应用”和“重置”）
- 标准对话框选项卡（如果使用，那么使用“注解”和“调试”）
- 错误和系统消息（例如“尚未保存该对象”。）

在扩展部分使用这些标准 IBM SPSS Modeler 组件的位置，这些组件将自动以选定的语言（如果该语言受支持）显示。

对于扩展部分的其他组件，CLEF 提供了有助于本地化的功能。可以本地化的内容包括:

- 节点名称（在选用板和工作区上）
- 模型名称（在管理器窗格的“模型”选项卡上）
- 文档名称（在管理器窗格的“输出”选项卡上）
- 与操作相关的图标图像的位置
- 工具提示文本
- 帮助系统
- 节点对话框:
  - 标题栏文本
  - 定制菜单和菜单条目
  - 字段、属性、按钮和选项卡标签
  - 静态文本
- 系统和错误消息

在翻译项目时，文本字符串应尽量缩短，以便能够保留较长的文本。

系统和错误消息可以通过规范文件、属性文件和服务器端 API 的组合进行本地化。有关更多信息，请参阅第 177 页的『状态详细信息文档』主题。

属性文件

可以本地化的项目的文本字符串存储在作为 **属性文件**的文件中，该文件使用标准 Java 格式存储本地化资源束。每个属性文件都包含一连串的记录，每个记录分别对应于扩展的每个本地化项目。每个记录中的字段都与规范文件中的 labelKey 属性相对应，从而使 CLEF 可以从属性文件中读取相应的文本字符串，并将其显示在正确的位置。

属性文件的扩展名必须是 .properties，且该文件必须与其相关节点的规范文件位于同一目录中。 IBM SPSS Modeler 开始会查找缺省的属性文件，名为:

path.properties

其中 *path* 是定义了属性捆绑包的 Bundle 元素（在“资源”部分）中的 path 属性。例如:

```
<Bundle id="bundle" path="my_resources"/>
```

如果没有缺省的属性文件，IBM SPSS Modeler 会从规范文件中的定义读取文本字符串。

必须有本地化支持的各种语言的属性文件。非缺省语言的文件通过文件名的后缀进行区分。例如:

```
my_resources.propertiesmy_resources_de.propertiesmy_resources_fr.properties
```

后缀符合语言代码的双字符 ISO 639-1 标准。

属性文件中的每个记录具有如下格式:

id=text_string

其中:

id 是规范文件中 buttonLabelKey、descriptionKey、dialogTitleKey、falseLabelKey、imagePathKey、labelKey、messageKey、textKey 或 trueLabelKey 属性中的标识。为便于区分，该标识的后缀通常是 .LABEL。当然，它可以有任何后缀或没有后缀，具体取决于它在规范文件中的出现形式。

text_string 是项目的文本。

示例: 将对话框选项卡本地化

本节点对话框中本地化选项卡示例使用两个属性文件，即缺省（英语）版本和法语版本，位置如下:

*extension_folder***my_resources**.properties
*extension_folder***my_resources**_fr.properties

其中，*extension_folder* 指包含规范文件的文件夹。

在规范文件中，属性文件通过 Resources 部分中的 Bundle 元素引用:

```
<Resources>
     <Bundle id="bundle" type="properties" path="my_resources"/>
</Resources>
```

请注意，path 属性一定不能包含语言扩展名或 .properties 后缀。

规范文件的其他相关部分是:

```
<Node id="uiTest" scriptName="ui_test" type="dataTransformer" palette="recordOp" label=
"UI Test" ... >
     <Properties>
          <Property name="enum1" valueType="enum" defaultValue="value4">
               <Enumeration>
```

```
                    <Enum value="value1" label="Value 1.1" labelKey="enum1.value1.LABEL"/>
                    <Enum value="value2" label="Value 1.2" labelKey="enum1.value2.LABEL"/>
                    <Enum value="value3" label="Value 1.3" labelKey="enum1.value3.LABEL"/>
                    <Enum value="value4" label="Value 1.4" labelKey="enum1.value4.LABEL"/>
                    <Enum value="value5" label="Value 1.5" labelKey="enum1.value5.LABEL"/>
               </Enumeration>
          </Property>
     </Properties>
 ...
     <UserInterface ... >
          <Tabs defaultTab="1">
               <Tab label="Basic Controls" labelKey="basicControlsTab.LABEL" ... >
      ...
     </UserInterface>
 ...
</Node>
```

在属性文件中，英语版属性文件包括以下记录:

```
basicControlsTab.LABEL=Basic Controls
enum1.value1.LABEL=Value 1.1
enum1.value2.LABEL=Value 1.2
enum1.value3.LABEL=Value 1.3
enum1.value4.LABEL=Value 1.4
enum1.value5.LABEL=Value 1.5
```

下图突出显示了受这些记录影响的对话框部分。

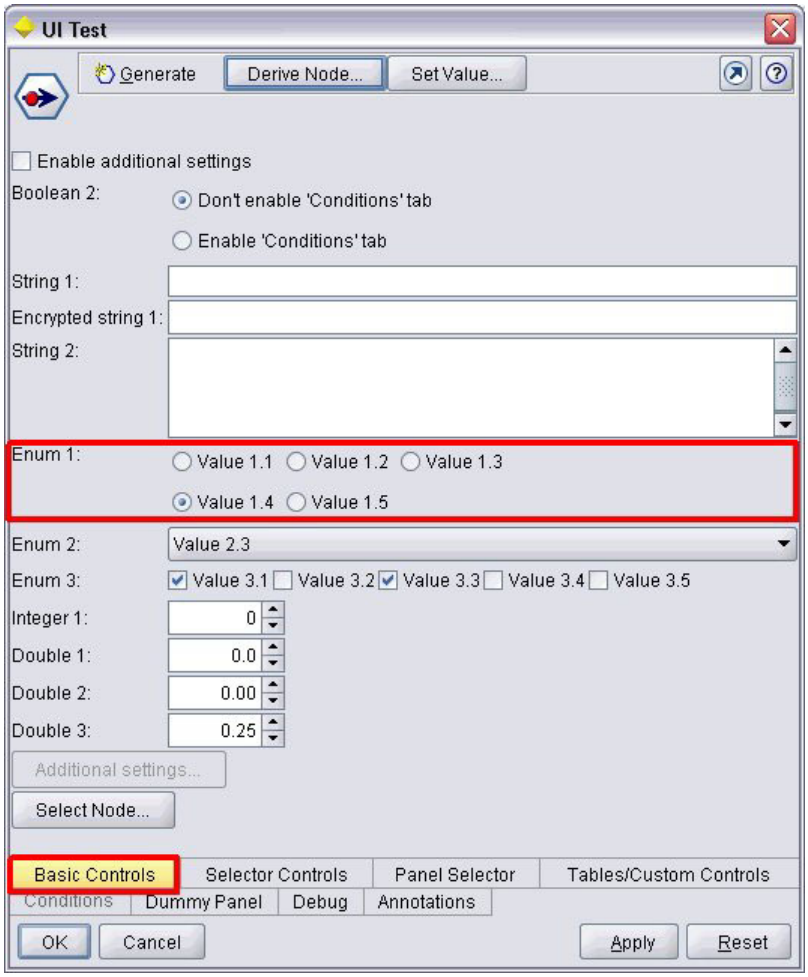

图 93. 未本地化的选项卡

法语版属性文件 (my_resources_fr.properties) 的相应部分是:

```
basicControlsTab.LABEL=Contrôles de Base
enum1.value1.LABEL=Valeur 1,1
enum1.value2.LABEL=Valeur 1,2
enum1.value3.LABEL=Valeur 1,3
enum1.value4.LABEL=Valeur 1,4
enum1.value5.LABEL=Valeur 1,5
```

这些记录将导致屏幕的相关部分显示翻译后的文本，如下图所示。

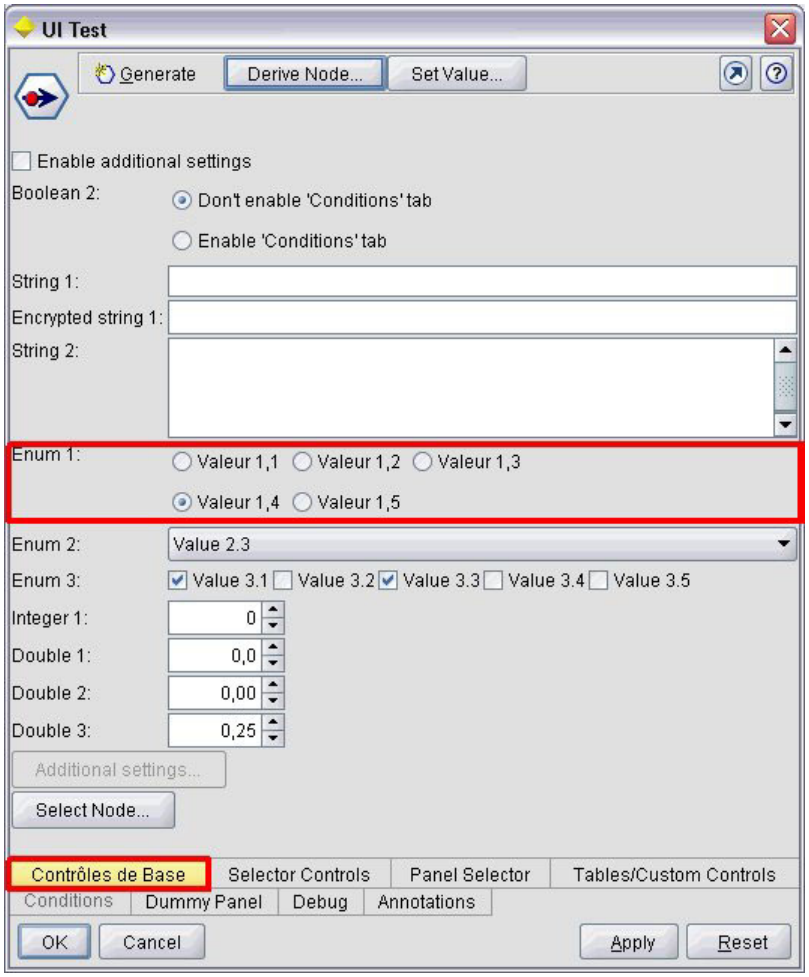

图 94. 本地化后的选项卡

请注意，屏幕底部四个按钮的本地化由 IBM SPSS Modeler 功能（而不是 CLEF）处理。

示例: 使用特殊字符

在属性文件中，需要对标准 ASCII 文本字符串的所有特殊字符使用 Unicode 转义序列。下面是已本地化为法语的部分属性文件:

```
Genlinnode.LABEL=Lin\u00e9aire g\u00e9n\u00e9ralis\u00e9

Fields.LABEL=Champs
Model.LABEL=Mod\u00e8le
Expert.LABEL=Expert

inputFields.LABEL=Entr\u00e9es
targetField.LABEL=Cible
 ...
```

对于使用非拉丁语字符的语言（例如日语或中文），需要对整个文本字符串使用 Unicode 转义序列。此处是针对日语本地化后的相同记录集:

```
Genlinnode.LABEL=\u4e00\u822c\u5316\u7dda\u578b

Fields.LABEL=\u30d5\u30a3\u30fc\u30eb\u30c9
Model.LABEL=\u30e2\u30c7\u30eb
Expert.LABEL=\u30a8\u30ad\u30b9\u30d1\u30fc\u30c8
```

```
inputFields.LABEL=\u5165\u529b
targetField.LABEL=\u5bfe\u8c61
 ...
```

帮助文件

如果要将具有帮助系统的扩展本地化，那么您还应提供帮助系统的本地化版本。应当针对每个本地化的扩展部分提供一个本地化帮助系统。

本地化 HTML 帮助

如果您要本地化的扩展部分使用 HTML 帮助文件（后缀为 .chm），可以使用本地化后的版本替换缺省的 .chm 文件。有关 HTML 帮助系统的更多信息，请参阅第 149 页的『HTML 帮助』。

要创建本地化后的 .chm 文件:

1. 创构建成帮助系统的各个帮助主题（ .htm 或 .html）文件的翻译版本，同时保持文件名不变。
2. 如果需要，可以使用包含在帮助系统中的本地化版本的图形（例如，屏幕快照）。
3. 使用 Microsoft HTML Help Workshop 或其他帮助创作工具将上述文件编译为本地化后的 .chm 文件。
4. 对帮助系统和本地化节点进行协同测试。有关更多信息，请参阅『测试本地化后的 CLEF 节点』主题。

本地化 JavaHelp

如果要本地化的扩展使用了 JavaHelp 系统，那么您需要针对每种受支持的语言提供帮助源文件的本地化版本。JavaHelp 负责显示正确的本地化版本（如果存在）。有关更多信息，请参阅第 149 页的『JavaHelp』主题。

要创建本地化后的 JavaHelp 系统:

1. 创构建成帮助系统的各个帮助主题（ .htm 或 .html）文件的翻译版本，同时保持文件名不变。
2. 如果需要，可以使用包含在帮助系统中的本地化版本的图形（例如，屏幕快照）。
3. 生成帮助集和其他所需文件（映射文件、内容和索引文件）。
4. 对帮助系统和本地化节点进行协同测试。有关更多信息，请参阅『测试本地化后的 CLEF 节点』主题。

测试本地化后的 CLEF 节点

要测试本地化后的节点及其帮助系统:

1. 在本地化节点的规范文件中的“资源”部分，更改 HelpInfo 元素的 path 属性以引用本地化后的 .chm 或 .hs 文件。例如，对于 HTML 帮助，您可以使用:

   ```
   <Resources>
    ...
       <HelpInfo id="help" type="HTMLHelp" path="help/mynode_fr.chm "/>
   </Resources>
   ```

   对于 JavaHelp，可以使用:

   ```
   <Resources>
    ...
       <HelpInfo id="help" type="javahelp" helpset="help/mynode_fr.hs "/>
   </Resources>
   ```

2. 将本地化后的 .chm 或 .jar 文件复制到 path 属性标明的位置。
3. 设置 Windows 区域的所需语言环境:

控制面板 > 区域和语言选项 > 区域选项 > 标准和格式 > <语言>

4. 启动 IBM SPSS Modeler，确保它以所需的语言显示。
5. 将本地化后的节点添加到 IBM SPSS Modeler。有关更多信息，请参阅第 185 页的『测试 CLEF 扩展』主题。
6. 将节点副本放入工作区。

   打开节点对话框，检查它是否以所需的语言正常显示。
7. 单击对话框中的“帮助”按钮，确保以所需语言显示正确的帮助主题。

辅助功能选项

所有的标准 IBM SPSS Modeler 辅助功能（例如，与鼠标操作等效的键盘操作以及屏幕朗读器支持）都可以使 CLEF 节点受益。

另外，出于提供辅助功能选项的考虑，可以提供带定制工具提示文本的 CLEF 节点。

您也可以指定键盘组合为最终用户提供访问您在 CLEF 中添加的各个用户界面功能的替代访问方式。有关更多信息，请参阅第 104 页的『访问键和键盘快捷键』主题。

对于操作按钮和被分类为控制器的各个屏幕组件（例如复选框或单选按钮组），可以定义:

- 标签
- 描述

标签是在屏幕上作为组件的名称显示的文本，可用屏幕朗读器软件读取。对于有视力障碍的用户，可通过“用户选项”对话框“显示”选项卡上的控件更改标签的显示字体大小，该对话框的访问方法是:

工具 > 用户选项

描述是鼠标指针悬停在组件上时显示的工具提示文本。与单独使用名称所能够传达的信息相比，工具提示能够提供更多有关组件的信息。工具提示也只能由配置为读取它们的屏幕朗读器读取。

标签和说明通过元素（在规范文件中定义组件）中的 `label` 和 `description` 属性定义。这两项分别可以通过 `labelKey` 和 `descriptionKey` 属性进行本地化。

示例

该操作按钮示例介绍了标签和说明功能的用途。

```
<Action id="setValue" label="Set Value..." labelKey="setValue.LABEL"
     description="Sets a value" descriptionKey="setValue.TOOLTIP"/>
```

第 9 章 程序设计

关于 CLEF 节点的程序设计

为了使节点能够执行无法在规范文件中定义的处理，CLEF 提供了下列应用程序编程接口 (API)，您的程序可以调用这些接口:

- **客户端 API**。一种 Java API，可供需要规范文件中没有直接提供的其他控件、用户界面组件或交互的扩展使用。
- **Predictive Server API (PSAPI)**。一种 Java API，公开可供需要数据挖掘和预测分析功能的应用程序使用的 IBM SPSS Modeler 功能。PSAPI 和 IBM SPSS Modeler 数据挖掘平台共享相同的底层技术。
- **服务器端 API**。一种基于 C 的 API，它涵盖许多方面，如设置和获取执行设置、这些设置的持久性、执行反馈、作业控制（如中断执行）、SQL 生成和返回的对象。

CLEF API 文档

下列各节提供有关客户端 API 和服务器端 API 的概述。更完整的 API 文档以 zip 文件的形式包含在 IBM SPSS Modeler 安装中，必须提取后才能使用。

要提取 API 文档:

1. 在产品 DVD 的 *\Documentation\en* 文件夹中找到文件 *clef_apidoc.zip*。
2. 使用 WinZip 或类似工具，将 zip 文件内容解压缩到任何方便使用的目录中。这样做会在该目录中创建一个包含所有 API 文档的 *clef_apidoc* 子文件夹。

要查看 API 文档:

1. 定位到 *clef_apidoc* 子文件夹，然后打开 *clef_apidoc.htm* 文件。
2. 根据需要选择 PSAPI/客户端或服务器端选项。

客户端 API

CLEF 提供了众多的 Java 类，其中包含可用于进行客户端处理的方法。例如，通过 `DataModelProvider` 类，可以计算由于对输入数据模型的更改太复杂而无法使用规范文件提供的功能的输出数据模型。

客户端 API 类

客户端类如下。

表 40. 客户端 API 类

类	描述
`NodeDelegate`	定义节点代表支持的方法。针对每个 `ExtensionProcessor` 实例创建一个节点代表实例。在 `extension.xml` 文件中相关的 `Node` 元素的“delegate”属性中指定实现类的路径。
`OutputDelegate`	定义输出代表支持的方法。针对每个 `ExtensionOutput` 实例创建一个输出代表实例。在 `extension.xml` 文件中相关的 `DocumentOutput` 或 `ModelOutput` 元素的“delegate”属性中指定实现类的路径。

表 40. 客户端 API 类 (续)

类	描述
ExtensionObjectUIDelegate	定义扩展对象 UI 代表支持的方法。针对每个扩展节点对话框或输出窗口实例创建一个 UI 代表实例。在 extension.xml 文件中相关的 UserInterface 元素的“uiDelegate”属性中指定实现类的路径。
ExtensionDelegate	定义扩展代表支持的方法。针对过程中共享的每个扩展创建一个扩展代表实例。因为单个过程可支持不同语言环境中的会话，因此没有可用于扩展代表的语言环境信息。在 extension.xml 文件中 CommonObjects 元素的“extensionDelegate”属性中指定实现类的路径。
ActionHandler	允许扩展管理用户通过菜单选项和工具栏按钮请求的操作
DataModelProvider	允许对数据模型进行复杂更改的节点使用 Java 计算输出数据模型
ExtensionObjectFrame	定义与用于显示模型或文档输出的窗口相关联的功能
ExtensionObjectPanel	定义与扩展对象面板相关联的功能
PropertyControl	定义与属性面板中的定制属性控件相关联的功能

客户端 API 文档提供了有关这些类的完整详细信息。有关更多信息，请参阅第 161 页的『CLEF API 文档』主题。

使用客户端 API

要将客户端函数调用包含到 CLEF 节点中:

1. 创建包含函数调用的 *.java* 源文件。
2. 将源文件编译到 *.class* 文件中。
3. 可以将多个 *.class* 文件合并到一个 *.jar* 文件中，然后在规范文件中包括对该 *.jar* 文件的引用，例如:

```
<Resources>
 ...
     <JarFile id="selfjar" path="selfllearning.jar"/>
 ...
</Resources>
```

某些 CLEF 元素允许您显式引用某一个类。例如，可以在规范文件的 PropertyControl 元素的 controlClass 属性中包含一个类引用:

```
<PropertyControl property="target_field_values_specify" ...
     controlClass="com.spss.clef.selflearning.propertycontrols.list.CustomListControl" ... />
```

其中，CustomListControl 是实现属性控件的类的名称；com.spss.clef.selflearning.propertycontrols.list 是该类在 JarFile 元素中声明的 *.jar* 文件内的路径。

有关更多信息，请参阅第 32 页的『“资源”部分』主题。

您还可能会发现，查看此发行版随附的示例节点的源代码会很有帮助。有关更多信息，请参阅第 26 页的『检查源代码』主题。

Predictive Server API (PSAPI)

PSAPI 提供了面向底层 Predictive Server 技术的编程接口。PSAPI 的主要元素均被指定为 Java 接口。其中大部分接口通过由 PSAPI 提供但并非 PSAPI 规范组成部分的内部类实现。此方法旨在保护 PSAPI 用户不受 Predictive Server 技术更改（例如体系结构更改以及专用客户端/服务器协议更改等等）的影响。

PSAPI 文档提供了有关这些类的完整详细信息。有关更多信息，请参阅第 161 页的『CLEF API 文档』主题。

服务器端 API

服务器端 API 定义为 C 语言 API，但仍支持以 C++ 实现。扩展模块的开发者可以选择使用 C 或 C++ 直接针对 C 语言 API 进行编程。如果开发者有绑定到 C API 的方法，那么也可以使用其他语言。 CLEF 还提供了多个 C++ 帮助程序源文件，这些文件相当于某些 C API 的包装器。

体系结构

客户端上的扩展 **节点**由服务器上的扩展 **对等**补充。对等由以服务器承载的共享库形式实现的 **扩展模块**定义。节点与其对等之间的通信由服务器管理的扩展 **资源**传递。资源调用由扩展模块定义的 **服务函数**可创建并操作其对等，同时对等使用 **回调函数**可向其主机请求信息和服务。

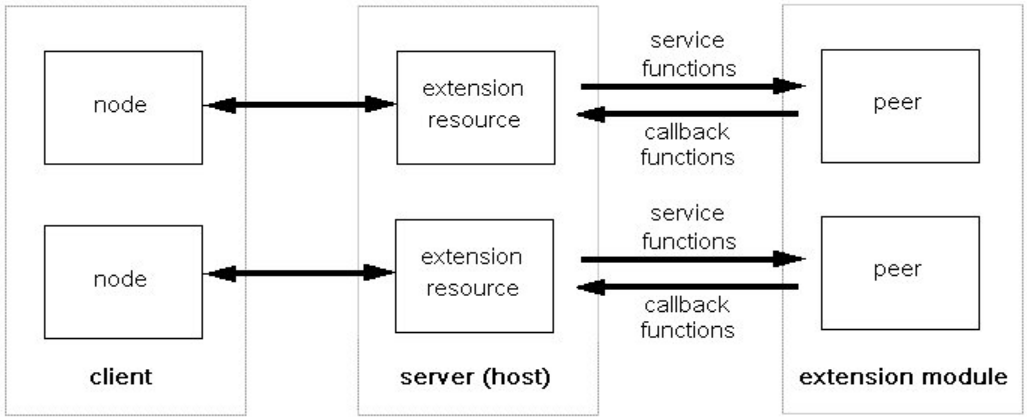

图 95. CLEF API 体系结构

服务函数

服务函数由扩展模块实现。扩展模块必须实现标记为必需的所有函数，并且可以实现任一或所有未作此标记的函数。

有两种类型的服务函数:

- 模块函数
- 对等函数

以下各节提供了服务函数的概述。有关这些函数的更详细说明可以在服务器端 API 文档中找到，如下所示:

1. 在“CLEF API 文档”屏幕中，选择**服务器端 API 概述**。
2. 单击 **模块**选项卡。
3. 选择 **由扩展模块实现的 API 服务函数**。

有关访问 CLEF API 文档的信息，请参阅 第 161 页的『CLEF API 文档』。

模块函数

模块函数从单线程调用。

表 41. 模块函数

函数	必需?	描述
clemext_initialise	是	初始化扩展模块
clemext_cleanup	是	释放由扩展模块分配的资源
clemext_getModuleInformation	否	获取有关扩展模块的信息
clemext_create_peer	是	为扩展节点创建对等实例
clemext_destroy_peer	是	释放对等实例

对等函数

对等函数应用于先前调用 clemext_create_peer 所返回的对等实例句柄。仅当各个对等句柄互不相同时，才可以从不同线程中并发调用对等函数。但存在一种例外情况，即可以随时从任何线程调用 clemext_peer_cancelExecution 函数（如果已定义）以中断另一线程上运行时间很长的执行。

表 42. 对等函数

函数	必需?	描述
clemext_peer_configure	是	指示对等实例处理其参数和数据模型
clemext_peer_getDataModel	是	获取对等实例的输出数据模型
clemext_peer_getCatalogueInformation	否	获取模块的分类信息
clemext_peer_getExecutionRequirements	否	获取对等实例的执行要求
clemext_peer_beginExecution	是	开始执行对等实例
clemext_peer_nextRecord	是	移到对等实例的结果集中的下一条记录
clemext_peer_getRecordValue	是	返回上次提取的结果记录中指定字段的值
clemext_peer_endExecution	是	指示执行已完成的对等实例
clemext_peer_cancelExecution	否	从单独的线程调用以中断运行时间很长的 beginExecution 或 nextRecord 函数调用
clemext_peer_getSQLGeneration	否	从对等实例生成 SQL
clemext_peer_getErrorDetail	是	检索对等上有关上一个模块特定的错误的补充信息

下表所列的对等函数设计为配合交互式模型构建器使用。

表 43. 与交互式模型构建器配合使用的对等函数

函数	必需?	描述
clemext_peer_beginInteraction	否	开始与对等实例交互
clemext_peer_request	否	执行对等上的交互请求
clemext_peer_getRequestReply	否	检索上一个成功执行的请求的回复
clemext_peer_endInteraction	否	指示交互已完成的对等实例

回调函数

当扩展模块需要来自主机进程的信息或服务时，它必须通过 **回调**来进行此操作。回调应用于 **句柄**，句柄是标识请求目标的指针。

通过在该调用所定向的 IBM SPSS Modeler 对象的句柄中进行传递可调用回调。句柄作为服务函数的参数传递到扩展模块中。

如果回调函数失败，它应在相关联的模块特定的错误码（由 CLEMEXTErrorCode 表示）中返回一些更详细的信息。反过来，模块可以通过返回回调错误并传递此详细信息来管理失败的调用函数，以便主机能够检测到它。

可用的回调函数类型有:

- 主机函数
- 节点函数
- 迭代器函数
- 进度函数
- 通道函数（仅适用于交互式模型）

以下各节提供了回调函数的概述。有关这些函数的更详细说明可以在服务器端 API 文档中找到，如下所示:

1. 在“CLEF API 文档”屏幕中，选择**服务器端 API 概述**。
2. 单击 **模块**选项卡。
3. 选择 **一般回调**。

有关访问 CLEF API 文档的信息，请参阅 第 161 页的『CLEF API 文档』。

主机函数

主机函数定义在通过 clemext_initialise 传递的主机句柄中。

表 44. 主机函数

函数	描述
clemext_host_getHostInformation	获取有关主机环境的静态信息

节点函数

节点函数定义在通过 clemext_create_peer 传递的节点句柄中。

表 45. 节点函数

函数	描述
clemext_node_getNodeInformation	获取有关节点的静态信息
clemext_node_getParameters	获取节点的参数
clemext_node_getDataModel	获取节点的输入数据模型
clemext_node_getOutputDataModel	获取节点的输出数据模型
clemext_node_getSQLGeneration	获取节点的上游 SQL 生成信息
clemext_node_getPassword	检索密码标识的纯文本
clemext_node_getFilePath	检索执行期间在客户端和服务器之间交换的文件的路径

迭代器函数

迭代器函数定义在通过 clemext_peer_beginExecution 传递的迭代器句柄中。迭代器公开扩展模块的输入数据集。

表 46. 迭代器函数

函数	描述
clemext_iterator_nextRecord	移到输入数据集中的下一个记录
clemext_iterator_getRecordValue	返回上次提取的输入记录中指定字段的值
clemext_iterator_rewind	恢复输入数据集的状态，以便下一次调用 nextRecord 时会移到数据集中的第一个记录

进度函数

进度函数定义在通过 clemext_peer_beginExecution 传递的进度句柄中。

表 47. 进度函数

函数	描述
clemext_progress_report	向主机回报进度

通道函数

通道函数只能与交互式模型一起使用，并且定义在通过 clemext_peer_beginInteraction 传递的通道句柄中。

表 48. 通道函数

函数	描述
clemext_channel_send	向客户端发送异步消息，启用一个对等线程以报告进度和状态信息

过程流

扩展模块调用许多服务和回调函数以执行其处理。实际调用的函数取决于模块需要执行的处理。

示例

典型模块执行的时序图如下所示。

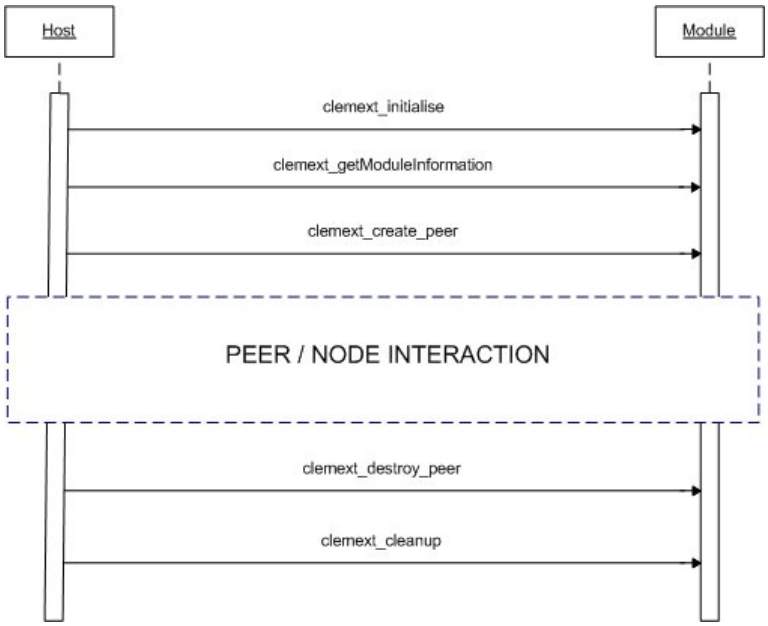

图 96. 典型过程流

对等/节点交互块如下图所示。

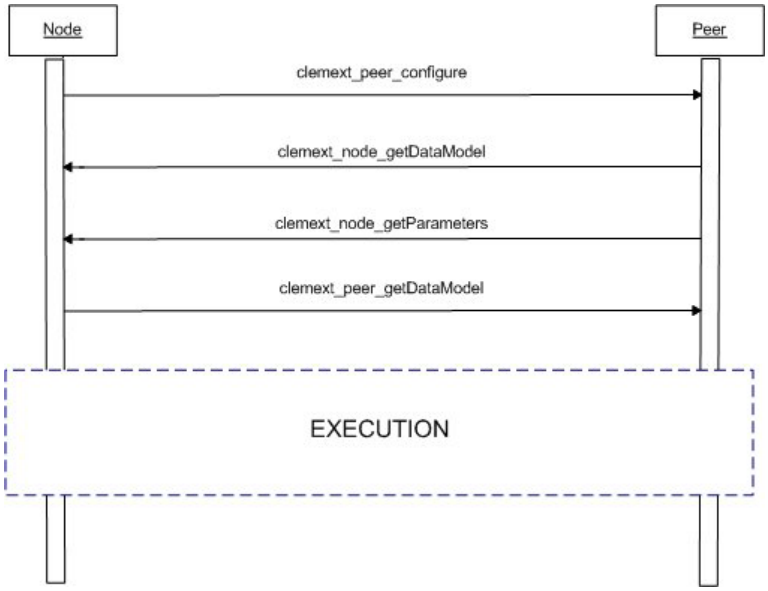

图 97. 典型对等/节点交互块

典型执行块如下图所示。

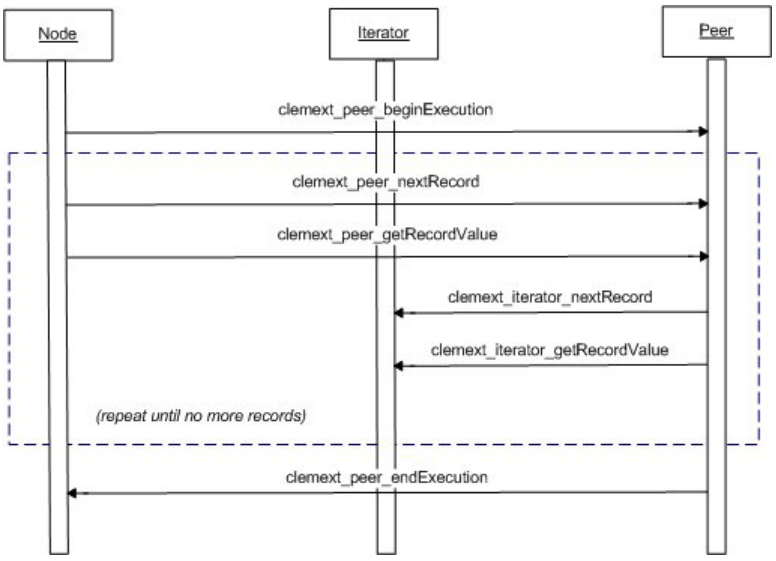

图 98. 典型执行块

注意:

- 模块可能会在服务器启动时加载到 IBM SPSS Modeler 服务器进程中，也可能会在以后首次需要其服务时根据需要加载。
- 模块加载后，主机调用一次服务函数 `clemext_initialise`。
- 模块加载并初始化后，主机可能会使用服务函数 `clemext_getModuleInformation` 查询该模块。
- 模块加载后，将通过该模块提供的对等对象来调用其服务。在模块内，对等对象由服务函数 `clemext_create_peer` 作为主机节点对象的对应对象创建，以按照主机应用程序的指示管理任务的执行。可以存在多个同一类型的对等对象，这些对象可以同时在一个进程中并发执行。
- 创建对等对象后，可以通过服务函数 `clemext_peer_configure` 来配置它。
- 此时，对等可以执行回调函数以获取客户端的信息，如 `clemext_node_getDataModel` 和 `clemext_node_getParameters`。
- IBM SPSS Modeler 通过 `clemext_peer_getDataModel` 服务函数来获取对等实例的输出数据模型。
- 可通过 `clemext_peer_beginExecution` 服务函数开始执行对等实例。
- `clemext_peer_nextRecord` 服务函数将焦点移到对等结果集中的下一个记录（如果是首次调用该函数，那么移到第一个记录）。然后是 `clemext_peer_getRecordValue` 服务函数，它返回当前记录中指定字段的值。
- 可通过 CLEF 模块调用迭代器回调函数 `clemext_iterator_nextRecord` 和 `clemext_iterator_getRecordValue`，以对输入记录进行排序并返回指定的字段值。
- 可通过调用 `clemext_peer_endExecution` 服务函数来结束对等函数的执行。
- 可通过调用 `clemext_destroy_peer` 来删除对等实例。
- 卸载模块之前，主机调用服务函数 `clemext_cleanup`。
- 可在服务器进程关闭时或在不再需要其服务之前卸载模块。

服务器端 API 功能

本节着重介绍服务器端 API 的某些功能:

- 节点类型信息
- 表示不同类型的数据存储的数据类型

- 服务器端共享库
- 文件空间和临时文件
- 用于执行数据库中 SQL 指令的 SQL 回送
- IBM SPSS Modeler 与扩展之间的数据模型信息的交换
- 输出文档
- C++ 帮助程序

节点类型

在规范文件中，节点定义采用下列格式:

```
<Node id="identifier" type="node_type" .../>
```

id 属性是一个字符串，用于唯一地标识节点。

type 属性将节点标识为下列类型之一:

- 数据阅读器
- 数据记录器
- 数据变换器
- 模型构建器
- 模型应用器
- 文档构建器

有关更多信息，请参阅第 9 页的『节点概述』主题。

clemext_create_peer 函数将 Node 元素的 id 和 type 两个属性的值作为参数。

单个扩展模型可以实现各种类型的节点，从而在每种类型中执行各种功能。例如，一个模块可以实现:

- 数据源的数据阅读器和数据记录器
- 各种建模算法的模型构建器和模型应用器
- 各种图形类型的文档构建器

数据和存储类型

对等实例通过在开始执行时在为其提供的迭代器上调用 clemext_iterator_getRecordValue 来获取输入数据，并提供输出数据以响应来自主机的 clemext_peer_getRecordValue 请求。数据以二进制格式在内存中传输，并且对等实例和主机必须在数据类型方面达成一致。

二进制数据类型由数据模型确定，并与字段的存储属性相关。

下表列出了可能的存储类型以及用于表示这些存储类型的数据类型。

表 49. 存储类型

存储类型	表示方法	附注
string	char *	字符串始终是 UTF-8 编码
real	CLEMEXTReal	双精度浮点数
integer	CLEMEXTInteger	64 位有符号整数
date	CLEMEXTDate	64 位有符号整数，表示自 1970 年 1 月 1 日起的天数
时间	CLEMEXTTime	64 位有符号整数，表示自午夜起的秒数

表 49. 存储类型 (续)

存储类型	表示方法	附注
时间戳	CLEMEXTTimestamp	64 位有符号整数，表示自 1970 年 1 月 1 日午夜起的秒数
未知	-	表示未知数据类型 - 无法表示值

库

可以在规范文件中声明服务器端共享库，以支持节点执行。共享库路径用于定位动态加载到主机进程中的共享库。共享库必须定义所有的必需 API 函数。有关更多信息，请参阅第 33 页的『共享库』主题。

如果在规范文件中提供了模块名称（在节点定义的 执行部分），该名称将传递到服务函数 clemext_create_peer 的 nodeId 参数以创建对等对象。这样，扩展即可创建适当类型的对等模块。nodeType 参数值也可能会影响创建的对等的种类。模块名称可以为空白，因为一个共享库不会实现一个以上的同一类型的模块。

实现了扩展模块的共享库可能会需要从属库。这些从属库应位于与扩展共享库相同的目录中。

临时文件

客户端规范文件和服务器扩展模块可以指定与 **文件空间**相关的路径名，文件空间是临时专用空间，对等可以在文件空间中创建执行时使用的文件。文件空间是在服务器临时目录中为对等实例创建的子目录。将根据需要来创建它，并在对等销毁时删除它。

存在文件空间时，对等实例对其具有完全的控制权。文件空间的完整路径名包含在 **节点信息文档**中。这是 XML 格式的信息，作为执行 clemext_node_getNodeInformation 回调函数的结果返回。有关更多信息，请参阅第 176 页的『节点信息文档』主题。

SQL 回送

在 SQL 回送中，IBM SPSS Modeler 流从 SQL 数据库读取数据并执行数据处理，高级用户可以通过回送 SQL 指令以在数据库本身中执行来提高此操作的效率。

多个标准 IBM SPSS Modeler 节点均支持 SQL 回送，并且服务器端 API 包含函数调用以使 CLEF 节点也可支持 SQL 回送。

clemext_peer_getSQLGeneration 服务函数从对等实例生成 SQL，并且可用于将 SQL 执行回送到数据库。对于数据阅读器节点，生成的 SQL 必须能够独立创建对等结果集。对于任何其他类型的节点，生成的 SQL 将很有可能依赖于针对那些向对等实例提供输入的上游节点生成的 SQL。对等可以通过在其关联节点句柄上调用 clemext_node_getSQLGeneration 回调函数来获取上游 SQL。

数据模型处理

某些服务器端 API 调用与 IBM SPSS Modeler 和扩展模块之间的数据模型信息交换有关:

- clemext_node_getDataModel 获取节点的输入数据模型
- clemext_peer_getDataModel 从对等实例获取输出数据模型
- clemext_node_getOutputDataModel 获取节点的输出数据模型

其他调用与数据传入和传出模块的方法相关。数据模型确定用于在下列函数中查找字段值的索引，这些函数返回上一次提取的输入记录中指定字段的值:

- clemext_peer_getRecordValue
- clemext_iterator_getRecordValue

IBM SPSS Modeler 调用 clemext_node_getDataModel 以获取有关输入数据模型的字段的信息。信息以 XML 格式返回，例如:

```
<DataModel>
     <Fields>
          <Field name="abc" storage="string" type="set" />
          <Field name="uvw" storage="integer" type="range" />
          <Field name="xyz" storage="real" type="range" />
     <Fields>
</DataModel>
```

在使用 clemext_iterator_getRecordValue 函数从输入记录中检索值时，模块可以使用此信息来提供字段索引。

模块影响输入数据模型的方式受规范文件中 OutputDataModel 元素的 mode 属性的值控制。模块可以:

- 通过向模型中添加新字段来扩展模型。
- 通过除去或重命名现有字段来修改模型。
- 使用新字段替换现有模型。
- 保留模型不变。

下列示例演示扩展和替换模型。

示例 - 扩展输入数据模型: 最简单的案例是: 使模块能够添加新字段并设置这些字段的值，但不除去或更改现有字段的值。

假设规范文件在节点定义中包含下列指令:

```
<OutputDataModel mode="extend">
     <AddField name="field1" storage="string" ... />
     <AddField name="field2" storage="real" ... />
     ...
</OutputDataModel>
```

此处，将输出数据模型定义为包含输入数据模型中的所有字段以及 OutputDataModel 元素中指定的两个附加字段。因此，输出数据模型由五个字段组成。

clemext_peer_getDataModel 函数返回仅与添加字段相关的信息，例如:

```
<DataModel>
     <Fields>
          <Field name="field1" storage="string" ... />
          <Field name="field2" storage="real" ... />
     <Fields>
</DataModel>
```

返回的信息必须与规范文件中 <AddField > 元素的类型和数值（不是名字）相匹配。

模块可以使用回调函数 clemext_node_getOutputDataModel 获取 IBM SPSS Modeler 期望添加的字段的详细信息。此信息可以直接传递回 IBM SPSS Modeler 以响应对 clemext_peer_getDataModel 的调用。如果用于创建和命名输出字段的规范文件的逻辑比较复杂，这样做会很有用。

当 IBM SPSS Modeler 调用 clemext_peer_getRecordValue 时，该模块为每个输出记录提供新值。新字段的字段索引在输入字段的最后一个索引之后开始。在本示例中，输入数据模型包含三个字段（位于指数位置 0、1 和 2），因此两个输出字段的字段指数为 3 和 4。 IBM SPSS Modeler 在调用 clemext_peer_getRecordValue 时，不会使字段指数对应于输入字段，因为该模块无法更改这些字段。

示例 - 替换输入数据模型 (1): 在本示例中，扩展模块将废弃其输出中的所有输入数据模型字段，从而将其替换为新字段。

规范文件包含下列内容:

```
<OutputDataModel mode="modify">
     <AddField name="key" storage="integer" ... />
     <AddField name="field1" storage="real" ... />
     <AddField name="field2" storage="real" ... />
     ...
</OutputDataModel>
```

这次，通过调用 clemext_peer_getDataModel 返回的 XML 数据描述输出数据模型中的所有字段:

```
<DataModel>
     <Fields>
          <Field name="key" storage="integer" ... />
          <Field name="field1" storage="real" ... />
          <Field name="field2" storage="real" ... />
     <Fields>
</DataModel>
```

调用 clemext_peer_getRecordValue 时使用的字段指数从第一个输出字段 (key) 的字段指数 0 开始，然后是下一个字段 (field1) 的字段指数 1，依此类推。

示例 - 替换输入数据模型 (2): 在本示例中，由扩展提供的输出数据模型还将替换输入数据模型，如上一示例所示。但是，在本例中，未在规范文件中定义输出数据模型，而是在运行时由服务器上的扩展模块计算输出模型。规范文件包含下列内容:

```
<OutputDataModel mode="modify" method="sharedLibrary" libraryId="myLibraryId" />
```

要计算输出数据模型，IBM SPSS Modeler 首先调用 clemext_peer_configure，然后调用 clemext_peer_getDataModel。如上一示例所示，由 clemext_peer_getDataModel 的响应完全定义的输出数据模型中不会自动包含输入数据模型中的任何字段。

注意: 在此类情况（服务器上的输出数据模型由扩展模块定义）下，该模块无法使用 clemext_node_getOutputDataModel 来获取输出数据模型，因为这样会导致“操作无效”错误。

XML 输出文档

某些服务和回调函数在主机和扩展模块之间以 XML 输出文档的形式传送信息。存在众多不同的文档，如下表所示。

表 50. XML 输出文档

XML 输出文档	附注	由对...的调用返回
目录文档	包含与目录相关联的控件值列表。	clemext_peer_getCatalogInformation
数据模型文档	描述进/出节点的字段集	clemext_peer_getDataModel clemext_node_getDataModel clemext_node_getOutputDataModel
错误详细信息文档	提供有关错误或其他状态的信息	clemext_peer_getErrorDetail
执行需求文档	描述对等实例所需的执行支持，如数据缓存或必填输入字段	clemext_peer_getExecutionRequirements
主机信息文档	提供有关主机环境的信息，包括: 产品标识、描述、版本、供应商、版权和平台详细信息	clemext_host_getHostInformation

表 50. XML 输出文档 (续)

XML 输出文档	附注	由对...的调用返回
模块信息文档	提供有关扩展模块的信息，包括：模块标识、描述、版本、供应商、版权和许可证详细信息	clemext_getModuleInformation
节点信息文档	提供有关与对等实例相关联的节点的信息，包括：节点标识、类型和文件空间详细信息	clemext_node_getNodeInformation
参数文档	包含客户端节点的配置参数；内容由扩展确定	clemext_node_getParameters
SQL 生成文档	描述如何将对等的执行转换为 SQL	clemext_peer_getSQLGeneration clemext_node_getSQLGeneration
状态详细信息文档	提供有关进度以及执行期间出现的警告或其他状态的补充信息	clemext_progress_report

目录文档: 目录文档描述目录内容，其中包含可以从 UI 控件中显示的值列表。

CLEF 模块执行调用 getCatalogInformation 如下:

```
CLEMEXTStatus
getCatalogInformation(
          const char *catalogId,
            char* buffer,
            size_t buffer_size,
            size_t* data_size,
            CLEMEXTErrorCode* errorCode) {
      ...
}
```

其中 catalogId 是特定目录的标识，如规范文件的 Catalog 元素中的定义。

此函数返回目录文档。

示例

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<CatalogInformation>
     <CatalogEntry>
          <CatalogValue>apples</CatalogValue>
          <CatalogValue>0</CatalogValue>
     </CatalogEntry>
     <CatalogEntry>
          <CatalogValue>oranges</CatalogValue>
          <CatalogValue>1</CatalogValue>
     </CatalogEntry>
     <CatalogEntry>
          <CatalogValue>bananas</CatalogValue>
          <CatalogValue>2</CatalogValue>
     </CatalogEntry>
</CatalogInformation>
```

数据模型文档: 数据模型文档描述进入/离开节点的**数据模型**，即包含字段名称、类型和相关信息的字段集。它封装了类型节点的可用信息。

不使用输入的对等实例（源节点）的输入模型为空，而不生成输出的对等实例（终端节点）的输出模型为空。使用输入并生成输出的对等（过程节点）必须知道如何从其输入计算其输出模型。

对等可以通过在其关联节点句柄上调用 clemext_node_getDataModel 来获取其输入数据模型。对等提供其输出数据模型以响应来自主机的 clemext_peer_getDataModel 请求。

任何数据模型均可以直接表示为数据字典，该字典对数据模型中的所有字段及其属性进行枚举。节点向对等实例提供的输入数据模型始终采用这种形式。对等生成的输出数据模型可能会具有相同的形式，也可能会表示为应用于输入模型的一系列操作（添加字段、删除字段、修改字段）。这将大大简化某些节点的输出模型。

数据模型文档中的字段顺序非常重要，该顺序决定了数据在相应输入数据集或输出数据集中的出现顺序。

数据模型可能不完整，即，只提供数据的部分规范。完整指定的以允许对等计算执行计划的输入模型对于该对等来说是 **可执行**的。可执行的数据模型必须包含每一个字段的二进制类型，以便能够正确整理输入和输出数据。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<DataModel>
  <Fields>
    <Field name="Age" type="range" storage="integer" direction="in">
      <Range minValue="15" maxValue="74"/>
    </Field>
    <Field name="Sex" type="flag" storage="string">
      <Values>
        <Value value="F" flagValue="false" displayLabel="Female"/>
        <Value value="M" flagValue="true" displayLabel="Male"/>
      </Values>
    </Field>
    <Field name="BP" type="orderedSet" storage="integer">
      <Values>
        <Value value="-1" />
        <Value value="0" />
        <Value value="1" />
      </Values>
    </Field>
    <Field name="Cholesterol" type="flag" storage="string">
      <Values>
        <Value value="NORMAL" flagValue="false"/>
        <Value value="HIGH" flagValue="true"/>
      </Values>
    </Field>
    <Field name="Na" type="range" storage="real" displayLabel="Blood sodium">
      <Range minValue="0.500517" maxValue="0.899774"/>
    </Field>
    <Field name="K" type="range" storage="real" displayLabel="Potassium concentration">
      <Range minValue="0.020152" maxValue="0.079925"/>
    </Field>
    <Field name="Drug" type="set" storage="string" direction="out">
      <Values>
        <Value value="drugA"/>
        <Value value="drugB"/>
        <Value value="drugC"/>
        <Value value="drugX"/>
        <Value value="drugY"/>
      </Values>
    </Field>
  </Fields>
</DataModel>
```

错误详细信息文档: 错误详细信息文档用于将消息（错误、警告、信息）发送回 IBM SPSS Modeler，并提供有关错误或其他状态的信息。扩展模块可以提供错误详细信息文档以解释模块特定错误，从而响应来自主机的 clemext_peer_getErrorDetail 请求。

错误详细信息是一个或多个 Diagnostic 元素的集合，其中每一个诊断均至少包含一个错误码、一条消息，以及一个或多个包含要在消息中插入的进一步信息的参数集合。错误码与规范文件中 StatusCode 元素的值相匹配。

消息可能具有不同的语言变体，另外，客户端也可以使用错误码从资源束中选择本地化消息。一系列诊断元素描述导致错误的因果链。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<ErrorDetail>
  <Diagnostic code="123" severity="error">
    <Message>You can't do that ({0})</Message>
    <Parameter>Permission denied</Parameter>
  </Diagnostic>
  <Diagnostic code="456" severity="warning">
    <Message>That was silly!</Message>
    <Message lang="fr">Quel idiot!</Message>
  </Diagnostic>
</ErrorDetail>
```

执行需求文档: 执行需求文档用于描述对等实例所需的执行支持。对等实例可以提供执行需求文档以响应来自主机的 clemext_peer_getExecutionRequirements 请求。主机在对等上调用 clemext_peer_beginExecution 之前查询要求文档，以便提供正确的执行环境。

主机可以提供数据缓存服务，以使模块能够使用 clemext_iterator_rewind 函数跨输入数据进行多次传递。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<ExecutionRequirements>
  <Cache/><!-- this ensures that the CLEF module can make multiple passes over the input
    data -->
</ExecutionRequirements>
```

主机信息文档: 主机信息文档用于描述主机环境。扩展模块可以通过在主机句柄上调用 clemext_host_getHostInformation 来获取主机信息。

返回的信息包含产品标识、描述、版本、供应商、版权和平台的详细信息。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<HostInformation>
  <Host name="clemlocal" externalEncoding="cp1252" language="english_us"
        locale="English_United Kingdom.1252" provider="IBM Corp." version="17" platform=
        "Windows XP SP2" copyright="Copyright 1995-2011 IBM Corp. All rights reserved.">
    <VersionDetail major="12" minor="0"/>
    <PlatformDetail osType="windows" osName="WindowsNT" osMajor="5" osMinor="1"/>
    <LibraryDetail path="C:\Program Files\IBM\SPSS\Modeler\17\ext\bin\my.module\myModule.dll"/>
  </Host>
</HostInformation>
```

模块信息文档: 模块信息文档描述扩展模块。扩展模块必须提供模块信息文档以响应来自主机的 clemext_getModuleInformation 请求。

返回的信息包含模块标识、描述、版本、供应商、版权和许可的详细信息。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<ModuleInformation>
  <Module name="MyModule" provider="My Company Inc." version="10.1.0.329"
          copyright="Copyright 2006 My Company Inc. All rights reserved.">
    <VersionDetail major="10" minor="1" release="0" build="329"/>
    <Licence code="1234" type="mandatory"/>
    <Description>Provides a thorough test of the new extensions framework.</Description>
  </Module>
</ModuleInformation>
```

节点信息文档: 节点信息文档用于描述与对等实例相关联的节点。对等实例可以通过在节点句柄上调用 clemext_node_getNodeInformation 来获取节点信息。节点信息包含节点标识、类型和文件空间的详细信息。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<NodeInformation>
  <Node name="databaseImport" type="dataReader">
    <FileSpace path="C:\Program Files\IBM SPSS Modeler Server
17\tmp\ext-8005-6711-01"/>
  </Node>
</NodeInformation>
```

参数文档: 参数文档包含规范文件中定义的每一个 Property 元素的详细信息。这些详细信息以配置参数的形式返回，对等可以通过在节点句柄上调用 clemext_node_getParameters 来获取这些配置参数。

参数具有名称和值，其中的值可以为:

- 简单值（字符串）
- 键控值（键和值）
- 结构化值（指名值的集合）
- 值列表

参数文档的内容完全由扩展软件包确定。参数定义在客户端规范文件中，并由服务器扩展模块解释。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<Parameters>
  <Parameter name="linesToScan" value="50"/>
  <Parameter name="useCaption" value="true"/>
  <Parameter name="caption" value="My Caption"/>
  <Parameter name="captionPosition" value="north"/>
  <Parameter name="defaultAggregation">
    <ListValue>
      <Value value="min"/>
      <Value value="max"/>
      <Value value="mean"/>
      <Value value="stddev"/>
    </ListValue>
  </Parameter>
</Parameters>
```

SQL 生成文档: SQL 生成文档描述如何将对等的执行转换为 SQL。

对等可以提供 SQL 生成文档，以响应来自主机的 clemext_peer_getSQLGeneration 请求。主机将尝试先执行 SQL，然后再以内部方式执行对等实例。

使用输入的对等可以通过在其关联节点句柄上调用 clemext_node_getSQLGeneration 来获取其输入 SQL。

SQL 生成文档的主要组成部分是复制节点或流段的执行行为的 SQL 语句。对于生成数据的节点（即数据阅读器节点或数据变换器节点），该语句必须是 SELECT 语句，并且必须与将数据模型中的字段名称映射到 SELECT 语句中的列名称的字典同时使用。

SQL 生成文档还可以包含执行语句所针对的数据库连接的属性，例如数据源名称和产品名称。对等可以使用这些属性帮助确定其生成的 SQL。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<SqlGeneration>
  <Properties
    datasourceName="SQL Server"
    databaseName="DataMining"
    serverName="GB1-RDUNCAN1"
    passwordKey="PW0"
    userName="fred"
    dbmsName="Microsoft SQL Server"
    dbmsVersion="09.00.1399"/>
  <Statement>
    <Bindings>
      <Binding columnName="C0" fieldName="ID"/>
      <Binding columnName="C1" fieldName="START_DATE"/>
    </Bindings>
    <TableParameters>
 <TableParameter name="${TABLE26}" value="dbo.DRUG4N"/>
    </TableParameters>
    <Sql>
      SELECT
      T0.ID AS C0,T0."START_DATE" AS C1
      FROM ${TABLE26} T0
      WHERE (T0."START_DATE" > '2003-01-01')
      ORDER BY 2 ASC
    </Sql>
  </Statement>
</SqlGeneration>
```

状态详细信息文档: 状态详细信息文档提供有关进度以及执行期间出现的非致命警告或其他条件的信息。扩展模块可以使用 clemext_progress_report 回调函数异步调度状态详细信息文档。

状态详细信息文档由一个或多个 Diagnostic 元素的集合组成，其中每一个诊断至少包含一个状态码、一条消息（除非属性文件中提供），以及一个或多个包含要在消息中插入的进一步信息的参数集合。StatusDetail 元素也可以有可选 destination 属性，将消息传递到下列位置之一的目标:

- 由 IBM SPSS Modeler 管理的本地跟踪文件
- 客户端（适用于与用户有关的消息）
- 所有（发动到所有可能的目标）

Diagnostic 元素的格式为:

```
<Diagnostic code="integer" severity="severity_level">
     <Message>message_text</Message>
     <Parameter>value</Parameter>
</Diagnostic>
```

其中:

code（必需）是一个整数，用于表示条件代码。

severity 指示条件的严重性，这是下列其中一项: unknown、information、warning、error 或 fatal。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<StatusDetail destination="client">
  <Diagnostic code="654" severity="information">
    <Message>Processed {0} records</Message>
    <Parameter>10000</Parameter>
  </Diagnostic>
</StatusDetail>
```

使用本地化消息

如果您想从属性文件中使用本地化消息，您从状态详细信息文档中忽略 Message 元素，并使用规范文件中的消息密钥，如以下示例所示:

```
    ...
    <Execution ...>
    ...
        <StatusCodes>
            ...
            <StatusCode code="21" status="warning" messageKey="fieldIgnoredMsg.LABEL"/>
            ...
        </StatusCodes>
    </Execution>
    ...
```

messages.properties 文件将会包含:

```
fieldIgnoredMsg.LABEL=Field "{0}" cannot be used for model building and was ignored
```

在状态详细信息文档中，您然后可以将要替换的参数（如字段名）发送到本地化消息中，例如:

```
<?xml version="1.0" encoding="utf-8"?>
<StatusDetail>
    <Diagnostic code="21">
        <Parameter>BP</Parameter>
    </Diagnostic>
</StatusDetail>
```

C++ 帮助程序

某些 CLEF 示例节点包含许多预定义的 C++ 源文件，称为 **帮助程序**。这些帮助程序相当于某些基于 C 的服务器端 API 的包装器，可以轻松地将它们编译到 C++ CLEF 中。

表 51. C++ 帮助程序

帮助程序	描述
BufferHelper	管理 C API 中使用的可调整大小的内存缓冲区
DataHelper	帮助包装数据读取和写入操作

表 51. C++ 帮助程序 (续)

帮助程序	描述
HostHelper	包装 CLEMEXT HostHandle 对象
XMLHelper	包装 XML 处理

这些帮助程序采用 *.cpp* 和 *.h* 文件对形式，例如 *BufferHelper.cpp* 和 *BufferHelper.h*。

有关查看这些帮助程序文件的信息，请参阅第 26 页的『检查源代码』。

有关这些文件的更详细说明可以在服务器端 API 文档中找到，如下所示:

1. 在“CLEF API 文档”屏幕中，选择**服务器端 API 概述**。
2. 单击 **文件**选项卡。
3. 单击与您要查看其信息的帮助程序相对应的 **.h** 文件名。
4. 在 **数据结构**下，单击对应的类名即可显示该文档。

有关访问 CLEF API 文档的信息，请参阅 第 161 页的『CLEF API 文档』。

错误处理

每一个 API 函数调用均返回一个状态码 (CLEMEXTStatus) 和一个可选的模块特定错误码 (CLEMEXTErrorCode)。状态码可以是 success（无错误）或 API 函数枚举的错误码之一；这些错误码几乎总是包含“模块特定错误”。模块特定错误码可以为 0，表示“无模块特定错误”。

状态码消息由 IBM SPSS Modeler 提供。通用状态码的详细信息可在服务器端 API 文档中找到，如下所示:

1. 在“CLEF API 文档”屏幕中，选择**服务器端 API 概述**。
2. 单击 **模块**选项卡。
3. 选择 **通用状态码**。

有关访问 CLEF API 文档的信息，请参阅 第 161 页的『CLEF API 文档』。

模块特定错误消息可以通过下列方式提供:

- 在规范文件中（在模块部分的 StatusCodes 元素中）
- 在规范文件中引用的资源束中
- 通过扩展模块

对于特定于模块的错误码，模块可以提供额外的错误详细信息，其中包括缺省错误消息（适用于与规范文件和资源束无关的错误）以及要插入到消息中的参数。导致错误的因果链可以由多条错误消息描述。

客户端上的错误报告采用下列格式:

node_label:*message*

其中:

- *node_label* 是在其中指定了此模块的 Node 元素的 label 属性值。
- *message* 是消息的文本，此文本既可以由服务器提供，也可以在规范文件（或者用于本地化的 *.properties* 文件）中定义。

XML 解析 API

IBM SPSS Modeler 提供了 Apache 的 Xerces-C XML 解析器，从而提供了许多允许模块读写 XML 数据的回调。如果需要，可以替代自己的 XML 解析器。

使用服务器端 API

要在节点中包含服务器端函数调用:

1. 创建包含函数调用的 C++ *.cpp* 和 *.h* 源文件。
2. 将源文件编译到动态链接库 (*.dll*) 文件中。
3. 在规范文件中包括对 *.dll* 文件的引用，例如:

```
<Resources>
.
     <SharedLibrary id="mylib1" path="mycorp.mynode/mylib" />
.
</Resources>
```

有关更多信息，请参阅第 33 页的『共享库』主题。

您还可能会发现，查看此发行版随附的示例节点的源代码会很有帮助。有关更多信息，请参阅第 26 页的『检查源代码』主题。

服务器端编程准则

CLEF 模块的服务器端动态链接库 (DLL) 部分应该遵守许多准则，以确保模块功能正确并避免影响 IBM SPSS Modeler 的操作。CLEF 模块应该:

- 确保对等执行自包含
- 支持单个过程中的多个对等实例
- 确保线程安全
- 避免修改线程或过程环境
- 限制模块内的线程用途
- 正确处理请求以取消执行
- 重新启动中断的系统调用 (UNIX)
- 调用 CoInitialize 或 CoUninitialize 时务必注意 (Windows)
- 避免假设卸载模块的时间
- 启动子过程时务必注意
- 避免写入标准输出或标准误差

以下部分更加详细介绍了这些区域中的每一个。

确保对等执行自包含

对等实例不得假设在 IBM SPSS Modeler 服务器过程内存在其他对等实例。 IBM SPSS Modeler 可能计划执行，以便在流中直接相邻的与节点对应的对等实例实际上按不同阶段执行，以便实例的存在和执行不会重叠。

因此，对等实例应该是独立的，不得尝试直接与其他对等实例进行通信，例如通过管道或套接字进行通信。不同对等实例之间的所有通信应该通过读取或将数据写入流中直接执行，或通过一些外部代理（例如管理对等之间共享数据的数据库服务器）间接执行。

支持单个过程中的多个对等实例

最终用户可以当执行流时在服务器过程中创建特定 CLEF 模块的多个对等实例（例如相同类型的多个节点）。因此 CLEF 模块中的任何静态数据在多个对等实例中共享，不得用于存储对等对象的专有数据。静态数据的示例是 C++ 类的静态成员，以及 C 编译单位中的全局或静态变量。

CLEF 模块 API 函数必须可重入，并且避免进行任何不可重入的系统调用。例如，当对等实例使用 `clemext_iterator_nextRecord` 从其输入迭代器中获取输入数据时，这可能在位于第一个对等上游并且生成第一个对等最终使用的数据的第二个对等实例上依次调用 `clemext_peer_nextRecord`。

`strtok` 等系统调用不是重新进入，不得使用。有关重新进入选项的详细信息，请参考操作系统文档了解您使用的平台。

确保线程安全

IBM SPSS Modeler 可以交替执行来自不同执行线程的多个对等实例。因此，对所有在对等对象之间共享的资源进行的访问应该受保护，例如通过与互斥体（互斥对象）同步或类似的线程库服务实现保护。

CLEF 模块必须避免进行任何不具有线程安全性的系统调用。有关更多信息，请参考您的操作系统文档或 UNIX `man` 页面。

避免修改线程或过程环境

避免使用可能更改调用线程或过程环境的系统调用。

这种调用的一些示例在此列出 - 列表绝不会是穷举:

- `setlocale`（在用于更改语言环境，而非读取语言环境信息时）
- `SetCurrentDirectory` (Windows) 或 `chdir` (UNIX)
- `LogonUser` (Windows) 或 `seteuid` (UNIX)
- `putenv`
- `exit`
- `signal`

注意: 在 Windows 上，一个将会更改线程环境但可能有必要执行的调用是 `CoInitialize`。有关更多信息，请参阅第 182 页的『调用 CoInitialize 或 CoUninitialize 时务必注意 (Windows)』主题。

限制模块内的线程用途

通常，模块在内部可以自由地使用线程技术。然而，应该只对 IBM SPSS Modeler 用于调用 CLEF 模块函数的线程调用回调函数（`clemext_peer_cancelExecution` 除外）。

可以从在模块内执行的任何线程中异步调用以下回调函数:

- `clemext_progress_report`
- `clemext_channel_send`

对等实例应该确保多个线程不会同时调用每个调用。

正确处理请求以取消执行

当最终用户请求取消执行对等实例时，IBM SPSS Modeler 对模块的 `clemext_peer_cancelExecution` 函数进行异步调用。开发者应该尝试实现此调用。注意此函数将被异步调用，并在正由模型执行另一个 CLEF API 函数调用时调用。

重新启动中断的系统调用 (UNIX)

在 UNIX 上，IBM SPSS Modeler 应用程序使用信号和信号处理程序。如果过程在执行调用期间收到一个信号，一些 UNIX 系统调用可能返回代码 EINTR--查看 man 页面了解特定 UNIX 平台上的系统调用。

如果发生此事件，那么执行调用的代码应该检查 EINTR 返回码并重新启动调用。达到这点的一种方法是创建一个简单包装器函数 (open_safe) 并且使包装器应用程序调用:

```
int
open_safe(const char* path, int oflag, mode_t mode) {
  int res;
  while ((res = ::open(path, oflag, mode)) == -1
                                    && errno == EINTR) {
  }
  return res;
}
```

调用 CoInitialize 或 CoUninitialize 时务必注意 (Windows)

在 Windows 上，需要使用 Windows 组件对象模型 (COM) 库服务的线程应该调用系统 API 函数 CoInitialize 然后使用 COM 服务，并且应该在完成时调用 CoUninitialize。IBM SPSS Modeler 为一个模块调用 CLEF API 的线程可能或不可能已经拥有调用的 CoInitialize。

希望从这些线程中使用 COM 服务的 CLEF 模块应该调用 CoInitialize，通常在 clemext_create_peer 或 clemext_peer_beginExecution 函数中。如果该调用成功，模块必须稍后当线程完成执行时还调用 CoUninitialize，通常分别在 clemext_destroy_peer 或 clemext_peer_endExecution 中。

有关 CoInitiaize 调用的更多信息，请参阅 Microsoft Developer Network (MSDN) 站点 http://msdn.microsoft.com 上的文档。

避免假设卸载模块的时间

当前，始终加载 CLEF 模块，直到会话结束（例如模块不能按需要卸载和重新加载）。函数 clemext_cleanup 始终未被调用，即使从加载模块的 IBM SPSS Modeler 服务器过程中退出。因此开发者任何时候都不得假定模块将被卸载，其资源已释放。

启动子过程时务必注意

通过 CreateProcess (Windows) 或 fork (UNIX) 启动子过程可以引入大量复杂情况，使父过程和子过程进行交互，使子过程继承父过程中开放的资源。

如果 CLEF 模块需要调用超出过程的执行，请考虑使用一个适当的其他体系结构。例如，CLEF 模块可能使用应用程序服务器提供的服务以执行所需任务。

具体来说，Windows 过程应该避免使用 CreateProcess 函数启动子过程，将 bInheritHandles 参数设为 TRUE。这样做使子过程继承在父 (IBM SPSS Modeler 服务器) 过程中开放的所有文件描述符。

避免写入标准输出或标准误差

如果 CLEF 模块写入一个过程的标准输出或标准误差流（可能出于调试目的），通常最终用户不会看到。然而，当使用 IBM SPSS Modeler Solution Publisher 部署并从命令行命令解释程序中执行包含 CLEF 节点的流时（无论是在 Windows 还是 UNIX 中），此输出将可见，可能使用户感到混乱。

CLEF 模块可以改为通过调用主机回调函数 clemext_host_trace 并以字符串形式传递所要显示的消息来调用跟踪服务。还需要通过在 IBM SPSS Modeler 安装目录中的 IBM SPSS Modeler Server 配置选项文件 (*/config/options.cfg* 中使用以下设置在 IBM SPSS Modeler 安装中启用跟踪:

```
trace_extension, 1
```

跟踪的消息输出到 IBM SPSS Modeler 安装目录中的 */log/trace-<process_ID>-<process_ID>.log* 文件，其中 *process_ID* 是 IBM SPSS Modeler Server 进程的标识。请避免同时跟踪多个会话，否则它们将共享同一个日志文件，这将引起问题。

第 10 章 测试和分发

测试 CLEF 扩展

强烈建议先在本地测试新扩展，然后再将其分发给其他用户。

创建完规格文件和任何关联资源束、.jar 文件、共享库和用户帮助文件后，您可以将文件排列为所需文件结构并复制它们到您的本地 IBM SPSS Modeler 安装中以测试扩展。下次启动 IBM SPSS Modeler 时，您会在 IBM SPSS Modeler 用户界面中看到新扩展。

测试 CLEF 扩展

1. 如果 IBM SPSS Modeler 为打开的话，请关闭它。
2. 如果扩展定义 CLEF 节点或输出，我们建议激活扩展对话框的“调试”选项卡，直到扩展正确工作。有关更多信息，请参阅第 186 页的『使用“调试”选项卡』主题。
3. 将客户端和服务器端文件排列为所需结构。确保节点所需的所有关联资源（例如，`.jar` 或 `.dll` 文件）都已复制到正确的位置。有关更多信息，请参阅第 5 页的『文件结构』主题。
4. 复制客户端目录到 IBM SPSS Modeler 安装目录的 *\ext\lib* 文件夹。
5. 复制服务器端目录到 IBM SPSS Modeler 安装目录的 *\ext\bin* 文件夹。
6. 开始 IBM SPSS Modeler。
7. 如果该扩展定义了菜单或菜单项，请确保这些内容可以在主菜单系统中正确显示。如果扩展定义了新节点，确保节点显示在规范文件中的正确节点选用板上的所需位置。
8. 充分测试扩展。

   例如，确保:

   - 节点性能不随字段数量和记录数量的增加而下降
   - 对空值采用一致的处理方式
   - 如果需要，可支持不同语言环境（例如，欧洲和远东）
9. 一旦添加了扩展，您还可以对其规格文件进行更改。然而，您所进行的任何更改只在您重新启动 IBM SPSS Modeler 后生效。

调试 CLEF 扩展

CLEF 提供了以下辅助调试扩展的功能:

- XML 语法错误消息
- 外部执行
- 调试选项卡

XML 语法错误

XML 解析器中的错误消息标记了规范文件中的错误 XML 语法。

此消息显示错误所在的大概行号以及错误的类型。

要从这种情况下恢复:

1. 更正文件中的错误。
2. 按照 第 185 页的『测试 CLEF 扩展』中的过程重新测试文件。
3. 重复此过程，直到规范文件中没有语法错误。

外部执行

通常，用户所编写的 CLEF 扩展均是在其自身的过程中执行，而不是在 IBM SPSS Modeler 过程执行。这样做有助于调试，即使某个外部过程失败，也不会破坏掉整个 IBM SPSS Modeler Server 过程。

注意: 这可能覆盖此缺省设置。有关更多信息，请参阅『更改执行选项』主题。

使用“调试”选项卡

对于所有与 CLEF 节点或输出关联的对话框或框，都可以通过激活“调试”选项卡检查相应对象的属性设置。也可以查看在扩展中定义的所有容器的内容，并可将这些内容保存到文件中，以供进一步检查。有关更多信息，请参阅第 50 页的『容器』主题。

可以通过将规范文件中 Extension 元素的 debug 属性值设置为 true 来激活“调试”选项卡。有关更多信息，请参阅第 31 页的『Extension 元素』主题。

选项卡上的字段包括:

元素标识。扩展的唯一标识 - 规范文件中 ExtensionDetails 元素的 id 属性的值。

脚本名称。在脚本中引用节点时的唯一标识 - Node 元素 scriptName 属性的值。

扩展标识。扩展文件夹的名称，该扩展文件夹中包含扩展文件和目录资源。该值由 ExtensionDetails 元素的 providerTag 和 id 属性（由“.”字符分隔）连接而成。请注意，对于标识的 providerTag 部分，该值不应该包含供内部使用的字符串 spss。

属性。此表显示节点的 Property 声明的选定详细信息:

- **属性**。属性的唯一标识 - Property 元素 name 字段的值。
- **脚本名称**。在脚本中引用属性的唯一标识 - Property 元素 scriptName 属性的值。
- **值类型**。此属性可采用的值类型，由 Property 元素的 valueType 属性定义。
- **列表?** 指示属性是否为指定值类型的值列表 - Property 元素的 isList 属性的值。
- **共享?** 如果选中，表示扩展中不止一处使用了此属性（例如: 模型构建器节点、模型输出和模型应用器等）。
- **值**。属性的缺省值（如果有）。

容器。显示选定容器的内容（例如: 模型数据）。单击该字段可显示为扩展定义的所有其他容器的列表，选择不同容器可显示不同的内容。单击相邻的 **保存容器**按钮可以 XML 格式保存选定容器的内容，以供进一步检查。

跟踪。显示执行节点时可查看跟踪输出的对话框。

更改执行选项

缺省情况下，用户编写的 CLEF 扩展模块在独立于主 IBM SPSS Modeler 过程的过程中执行。因此，即使扩展过程失败，也不会导致 IBM SPSS Modeler 过程失败。与此不同，IBM Corp. 提供的模块缺省在主过程中执行。

通过提供两种服务器配置选项，系统管理员可以将一个或更多已命名的模块从其中一种模式更改为相反的模式。两个选项都采用逗号分隔的模块标识列表来指示哪些模块受到了更改的影响。

注意: 正常情况下，仅在客户支持代表请求时才更改这些选项之一。

选项如下所示:

过程执行选项

此选项允许一般会加载到外部过程的扩展模块（通常为用户编写的模块）可直接加载 IBM SPSS Modeler。格式为:

```
clef_inprocess_execution, "moduleID1[,moduleID2[,...moduleIDn]]"
```

其中，*moduleID* 是相关规范文件中 ExtensionDetails 元素的 id 属性值。示例如下:

```
clef_inprocess_execution, "test.example_filereader"
```

外部执行选项

此选项允许一般被直接加载到 IBM SPSS Modeler 的扩展模块（通常为 IBM Corp. 提供的模块）加载到外部过程。格式为:

```
clef_external_execution, "moduleID1[,moduleID2[,...moduleIDn]]"
```

其中，*moduleID* 与 clef_inprocess_execution 的相同。（虚构的）示例如下:

```
clef_external_execution, "spss.naivebayes,spss.terminator"
```

添加或更改执行选项

要添加或更改执行选项，请遵循《*IBM SPSS Modeler 17 服务器管理与性能指南*》上“使用 options.cfg 文件”中所述的步骤。

分发 CLEF 扩展

新的扩展全部测试完毕并可进行分发，过程如下:

1. 如果“调试”选项卡已激活的话，取消激活。有关更多信息，请参阅第 186 页的『使用“调试”选项卡』主题。
2. 创建一个文件结构，准确说明要安装扩展文件的方式。有关更多信息，请参阅第 5 页的『文件结构』主题。
3. 将该文件结构压缩为 .zip 文件。您会发现为客户端和服务器端文件单独制作单独 .zip 文件更为简便。
4. 向最终用户分发 .zip 文件。

安装 CLEF 扩展

要安装 CLEF 扩展:

1. 收到包含扩展文件结构的 .zip 文件时，将客户端文件解压到 IBM SPSS Modeler 安装目录下的 \ext\lib 文件夹。
2. 将服务器端文件解压到 IBM SPSS Modeler 安装目录（如果使用 IBM SPSS Modeler Server，那么为相应的安装目录）下的 \ext\bin 文件夹中。
3. 启动 IBM SPSS Modeler 并检查在节点选用板上指定的位置是否显示新节点。

卸载 CLEF 扩展

要卸载 CLEF 扩展:

1. 在 IBM SPSS Modeler 安装目录下的 \ext\lib 目录中找到扩展文件夹。

   如果扩展还安装了服务器端扩展文件夹，可在 IBM SPSS Modeler 或 IBM SPSS Modeler Server 安装目录下的 \ext\bin 目录中找到此文件夹。

2. 删除一个或多个扩展文件夹。

下次启动 IBM SPSS Modeler 时，更改才生效。

附录. CLEF XML 模式

CLEF 元素参考

本节提供 CLEF 中所有元素的参考。

每个主题都列出了元素及其父元素和子元素的有效属性。图显示了该元素的所有子元素。请注意，图中的箭头指示可以在其他元素间共享的元素。这些元素作为此主题（“CLEF 元素参考”）的子主题，而不是作为父主题的子主题列在目录中。

元素

Action 元素

表 52. Action 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
id	**必需**		*string*
imagePath	可选		*string*
imagePathKey	可选		*string*
label	**必需**		*string*
labelKey	可选		*string*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
shortcut	可选		*string*
shortcutKey	可选		*string*

XML 表示法

```
<xs:element name="Action">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="imagePath" type="xs:string" use="optional"/>
  <xs:attribute name="imagePathKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="shortcut" type="xs:string" use="optional"/>
  <xs:attribute name="shortcutKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Actions

ActionButton 元素

定义可用于调用操作的按钮。操作通常由 UI 代表或操作侦听器来实现。

表 53. ActionButton 的属性

属性	使用	描述	有效值
action	**必需**		*string*
showIcon	可选		*布尔值*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ActionButton">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="action" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="showIcon" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

Actions 元素

XML 表示法

```
<xs:element name="Actions">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="Action"/>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

Action

AddField 元素

表 54. AddField 的属性

属性	使用	描述	有效值
depth	可选		*整数*
direction	可选		**in** **out** **both** **none** **partition**
directionRef	可选		
fieldRef	可选		
group	可选		*string*
label	可选		*string*
missingValuesRef	可选		
name	**必需**		
prefix	可选		*string*
role	可选		**unknown** **predictedValue** **predictedDisplayValue** **概率** **residual** **standardError** **entityId** **entityAffinity** **upperConfidenceLimit** **lowerConfidenceLimit** **propensity** **value** **supplementary**
storage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**
storageRef	可选		
tag	可选		*string*
targetField	可选		*string*

表 54. AddField 的属性 (续)

属性	使用	描述	有效值
类型	可选		**auto** **range** **discrete** **set** **orderedSet** **flag** **typeless** **collection** **geospatial**
typeRef	可选		
value	可选		*string*
valueStorage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**

XML 表示法

```
<xs:element name="AddField">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Range" minOccurs="0"/>
      <xs:element ref="Values" minOccurs="0"/>
      <xs:element ref="NumericInfo" minOccurs="0"/>
      <xs:element name="MissingValues" minOccurs="0">
        <xs:sequence>
          <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="Range" minOccurs="0"/>
        </xs:sequence>
      </xs:element>
      <xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
      </xs:element>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
  <xs:attribute name="storage" type="FIELD-STORAGE">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
  <xs:attribute name="type" type="FIELD-TYPE">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="range"/>
    <xs:enumeration value="discrete"/>
    <xs:enumeration value="set"/>
    <xs:enumeration value="orderedSet"/>
    <xs:enumeration value="flag"/>
    <xs:enumeration value="typeless"/>
    <xs:enumeration value="collection"/>
    <xs:enumeration value="geospatial"/>
  </xs:attribute>
  <xs:attribute name="direction" type="FIELD-DIRECTION">
    <xs:enumeration value="in"/>
```

```
    <xs:enumeration value="out"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="partition"/>
  </xs:attribute>
  <xs:attribute name="label" type="xs:string"/>
  <xs:attribute name="depth" type="xs:integer" use="optional" default="-1"/>
  <xs:attribute name="valueStorage" type="FIELD-STORAGE" use="optional">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
  <xs:attribute name="fieldRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="storageRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="typeRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="directionRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="missingValuesRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="role" type="MODEL-FIELD-ROLE" use="optional">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="predictedValue"/>
    <xs:enumeration value="predictedDisplayValue"/>
    <xs:enumeration value="probability"/>
    <xs:enumeration value="residual"/>
    <xs:enumeration value="standardError"/>
    <xs:enumeration value="entityId"/>
    <xs:enumeration value="entityAffinity"/>
    <xs:enumeration value="upperConfidenceLimit"/>
    <xs:enumeration value="lowerConfidenceLimit"/>
    <xs:enumeration value="propensity"/>
    <xs:enumeration value="value"/>
    <xs:enumeration value="supplementary"/>
  </xs:attribute>
  <xs:attribute name="targetField" type="xs:string" use="optional"/>
  <xs:attribute name="value" type="xs:string" use="optional"/>
  <xs:attribute name="group" type="xs:string" use="optional"/>
  <xs:attribute name="tag" type="xs:string" use="optional"/>
  <xs:attribute name="prefix" type="xs:string" use="optional"/>
</xs:element>
```

父元素

ForEach 和 ModelFields

子元素

MissingValues、ModelField、NumericInfo、Range、Range、Values 和 Values

相关元素

ChangeField

MissingValues 元素:

表 55. MissingValues 的属性

属性	使用	描述	有效值
treatNullAsMissing	可选		*布尔值*
treatWhitespaceAsMissing	可选		*布尔值*

XML 表示法

```
<xs:element name="MissingValues" minOccurs="0">
  <xs:sequence>
    <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="Range" minOccurs="0"/>
```

```
  </xs:sequence>
  <xs:attribute name="treatWhitespaceAsMissing" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="treatNullAsMissing" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

AddField

子元素

Range、Range、Values 和 Values

ModelField 元素:

表 56. ModelField 的属性

属性	使用	描述	有效值
group	可选		
role	**必需**		**unknown** **predictedValue** **predictedDisplayValue** 概率 **residual** **standardError** **entityId** **entityAffinity** **upperConfidenceLimit** **lowerConfidenceLimit** **propensity** **value** **supplementary**
tag	可选		*string*
targetField	可选		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
  <xs:attribute name="role" type="MODEL-FIELD-ROLE" use="required">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="predictedValue"/>
    <xs:enumeration value="predictedDisplayValue"/>
    <xs:enumeration value="probability"/>
    <xs:enumeration value="residual"/>
    <xs:enumeration value="standardError"/>
    <xs:enumeration value="entityId"/>
    <xs:enumeration value="entityAffinity"/>
    <xs:enumeration value="upperConfidenceLimit"/>
    <xs:enumeration value="lowerConfidenceLimit"/>
    <xs:enumeration value="propensity"/>
    <xs:enumeration value="value"/>
    <xs:enumeration value="supplementary"/>
  </xs:attribute>
  <xs:attribute name="targetField" type="xs:string"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="group" type="MODEL-FIELD-GROUP"/>
  <xs:attribute name="tag" type="xs:string"/>
</xs:element>
```

父元素

AddField

And 元素

XML 表示法

```
<xs:element name="And">
  <xs:sequence minOccurs="2" maxOccurs="unbounded">
    <xs:group ref="CONDITION-EXPRESSION">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

And、Command、Constraint、CreateDocument、CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModel、CreateModelApplier、CreateModelOutput、Enabled、Not、Option、Or、Run 和 Visible

子元素

And、Condition、Not 和 Or

Attribute 元素

表 57. Attribute 的属性

属性	使用	描述	有效值
defaultValue	可选		
描述	可选		*string*
descriptionKey	可选		*string*
isList	可选		*布尔值*
label	**必需**		*string*
labelKey	可选		*string*
name	**必需**		*string*
valueType	可选		**string** **encryptedString** **整数** **double** **布尔值** **date** **enum**

XML 表示法

```
<xs:element name="Attribute">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
```

```
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="valueType" type="ATTRIBUTE-VALUE-TYPE">
    <xs:enumeration value="string"/>
    <xs:enumeration value="encryptedString"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="boolean"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="enum"/>
  </xs:attribute>
  <xs:attribute name="defaultValue" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="isList" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

Catalog 和 Structure

BinaryFormat 元素

XML 表示法

```
<xs:element name="BinaryFormat"/>
```

父元素

FileFormatType

Catalog 元素

表 58. Catalog 的属性

属性	使用	描述	有效值
id	必需		*string*
valueColumn	必需		*整数*

XML 表示法

```
<xs:element name="Catalog">
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:element ref="Attribute"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="valueColumn" type="xs:integer" use="required"/>
</xs:element>
```

父元素

Catalogs

子元素

Attribute

Catalogs 元素

XML 表示法

```
<xs:element name="Catalogs">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="Catalog"/>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

Catalog

ChangeField 元素

表 59. ChangeField 的属性

属性	使用	描述	有效值
depth	可选		*整数*
direction	可选		**in** **out** **both** **none** **partition**
directionRef	可选		
fieldRef	**必需**		
label	可选		*string*
missingValuesRef	可选		
name	**必需**		
storage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**
storageRef	可选		
类型	可选		**auto** **range** **discrete** **set** **orderedSet** **flag** **typeless** **collection** **geospatial**
typeRef	可选		

表 59. *ChangeField 的属性 (续)*

属性	使用	描述	有效值
valueStorage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**

XML 表示法

```
<xs:element name="ChangeField">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Range" minOccurs="0"/>
      <xs:element ref="Values" minOccurs="0"/>
      <xs:element ref="NumericInfo" minOccurs="0"/>
      <xs:element name="MissingValues" minOccurs="0">
        <xs:sequence>
          <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="Range" minOccurs="0"/>
        </xs:sequence>
      </xs:element>
      <xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
      </xs:element>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
  <xs:attribute name="storage" type="FIELD-STORAGE">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
  <xs:attribute name="type" type="FIELD-TYPE">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="range"/>
    <xs:enumeration value="discrete"/>
    <xs:enumeration value="set"/>
    <xs:enumeration value="orderedSet"/>
    <xs:enumeration value="flag"/>
    <xs:enumeration value="typeless"/>
    <xs:enumeration value="collection"/>
    <xs:enumeration value="geospatial"/>
  </xs:attribute>
  <xs:attribute name="direction" type="FIELD-DIRECTION">
    <xs:enumeration value="in"/>
    <xs:enumeration value="out"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="partition"/>
  </xs:attribute>
  <xs:attribute name="label" type="xs:string"/>
  <xs:attribute name="depth" type="xs:integer" use="optional" default="-1"/>
  <xs:attribute name="valueStorage" type="FIELD-STORAGE" use="optional">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
  <xs:attribute name="fieldRef" type="EVALUATED-STRING" use="required"/>
  <xs:attribute name="storageRef" type="EVALUATED-STRING" use="optional"/>
```

```
  <xs:attribute name="typeRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="directionRef" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="missingValuesRef" type="EVALUATED-STRING" use="optional"/>
</xs:element>
```

父元素

ForEach 和 ModelFields

子元素

MissingValues、ModelField、NumericInfo、Range、Range、Values 和 Values

相关元素

AddField

MissingValues 元素:

表 60. MissingValues 的属性

属性	使用	描述	有效值
treatNullAsMissing	可选		*布尔值*
treatWhitespaceAsMissing	可选		*布尔值*

XML 表示法

```
<xs:element name="MissingValues" minOccurs="0">
  <xs:sequence>
    <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="Range" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="treatWhitespaceAsMissing" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="treatNullAsMissing" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

AddField

子元素

Range、Range、Values 和 Values

ModelField 元素:

表 61. ModelField 的属性

属性	使用	描述	有效值
group	可选		

表 61. ModelField 的属性 (续)

属性	使用	描述	有效值
role	**必需**		**unknown** **predictedValue** **predictedDisplayValue** **概率** **residual** **standardError** **entityId** **entityAffinity** **upperConfidenceLimit** **lowerConfidenceLimit** **propensity** **value** **supplementary**
tag	可选		*string*
targetField	可选		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
  <xs:attribute name="role" type="MODEL-FIELD-ROLE" use="required">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="predictedValue"/>
    <xs:enumeration value="predictedDisplayValue"/>
    <xs:enumeration value="probability"/>
    <xs:enumeration value="residual"/>
    <xs:enumeration value="standardError"/>
    <xs:enumeration value="entityId"/>
    <xs:enumeration value="entityAffinity"/>
    <xs:enumeration value="upperConfidenceLimit"/>
    <xs:enumeration value="lowerConfidenceLimit"/>
    <xs:enumeration value="propensity"/>
    <xs:enumeration value="value"/>
    <xs:enumeration value="supplementary"/>
  </xs:attribute>
  <xs:attribute name="targetField" type="xs:string"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="group" type="MODEL-FIELD-GROUP"/>
  <xs:attribute name="tag" type="xs:string"/>
</xs:element>
```

父元素

AddField

CheckBoxControl 元素

定义可用于修改布尔值的复选框控件。

表 62. CheckBoxControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
invert	可选		*布尔值*
label	可选		*string*

表 62. CheckBoxControl 的属性 (续)

属性	使用	描述	有效值
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="CheckBoxControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="invert" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

CheckBoxGroupControl 元素

定义可用于从枚举列表类型中指定值选择的复选框控件组。

表 63. CheckBoxGroupControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*

表 63. CheckBoxGroupControl 的属性 (续)

属性	使用	描述	有效值
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
layoutByRow	可选		*布尔值*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
行	可选		*positiveInteger*
showLabel	可选		*布尔值*
useSubPanel	可选		*布尔值*

XML 表示法

```
<xs:element name="CheckBoxGroupControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="rows" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="layoutByRow" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="useSubPanel" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

ClientDirectoryChooserControl 元素

定义可用于选择客户端上的目录的控件。

表 64. ClientDirectoryChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
mode	**必需**		**open** **save** **import** 导出
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ClientDirectoryChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="mode" type="FILE-CHOOSER-MODE" use="required">
    <xs:enumeration value="open"/>
    <xs:enumeration value="save"/>
    <xs:enumeration value="import"/>
    <xs:enumeration value="export"/>
  </xs:attribute>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

ClientFileChooserControl 元素

定义可用于选择客户端上的文件的控件。

表 65. ClientFileChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
mode	**必需**		**open** **save** **import** **导出**
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ClientFileChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="mode" type="FILE-CHOOSER-MODE" use="required">
    <xs:enumeration value="open"/>
    <xs:enumeration value="save"/>
    <xs:enumeration value="import"/>
    <xs:enumeration value="export"/>
  </xs:attribute>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

ComboBoxControl 元素

表 66. ComboBoxControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ComboBoxControl" type="CONTROLLER">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
</xs:element>
```

表 67. 扩展类型

类型	描述
ItemChooserControl	

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

Command 元素

表 68. Command 的属性

属性	使用	描述	有效值
path	必需		

XML 表示法

```
<xs:element name="Command">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

Run

子元素

And、Condition、Not 和 Or

CommonObjects 元素

为扩展的全局定义提供位置

表 69. CommonObjects 的属性

属性	使用	描述	有效值
extensionDelegate	可选		*string*
extensionListenerClass	可选		*string*

XML 表示法

```
<xs:element name="CommonObjects">
  <xs:all>
    <xs:element ref="PropertyTypes" minOccurs="0"/>
    <xs:element ref="PropertySets" minOccurs="0"/>
    <xs:element ref="FileFormatTypes" minOccurs="0"/>
    <xs:element ref="ContainerTypes" minOccurs="0"/>
    <xs:element ref="Actions" minOccurs="0"/>
```

```
    <xs:element ref="Catalogs" minOccurs="0"/>
  </xs:all>
  <xs:attribute name="extensionDelegate" type="xs:string" use="optional"/>
  <xs:attribute name="extensionListenerClass" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Actions、Catalogs、ContainerTypes、FileFormatTypes、PropertySets 和 PropertyTypes

Condition 元素

表 70. Condition 的属性

属性	使用	描述	有效值
container	可选		*string*
control	可选		*string*
expression	可选		
listMode	可选		**all** 任意

表 70. Condition 的属性 (续)

属性	使用	描述	有效值
op	必需		**equals** **notEquals** **isEmpty** **isNotEmpty** **lessThan** **lessOrEquals** **greaterThan** **greaterOrEquals** **equalsIgnoreCase** **isSubstring** **startsWith** **endsWith** **startsWithIgnoreCase** **endsWithIgnoreCase** **isSubstring** **hasSubstring** **isSubstringIgnoreCase** **hasSubstringIgnoreCase** **in** **countEquals** **countLessThan** **countLessOrEquals** **countGreaterThan** **countGreaterOrEquals** **contains** **storageEquals** **typeEquals** **directionEquals** **isMeasureDiscrete** **isMeasureContinuous** **isMeasureCollection** **isMeasureGeospatial** **isMeasureTypeless** **isMeasureUnknown** **isStorageString** **isStorageNumeric** **isStorageDatetime** **isStorageList** **isStorageUnknown** **isModelOutput** **modelOutputRoleEquals** **modelOutputTargetFieldEquals** **modelOutputHasValue** **modelOutputTagEquals** **enumRestriction**
property	可选		*string*

表 70. Condition 的属性 (续)

属性	使用	描述	有效值
value	可选		

XML 表示法

```
<xs:element name="Condition">
  <xs:attribute name="expression" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="control" type="xs:string" use="optional"/>
  <xs:attribute name="property" type="xs:string" use="optional"/>
  <xs:attribute name="container" type="xs:string" use="optional"/>
  <xs:attribute name="op" type="CONDITION-TEST" use="required">
    <xs:enumeration value="equals"/>
    <xs:enumeration value="notEquals"/>
    <xs:enumeration value="isEmpty"/>
    <xs:enumeration value="isNotEmpty"/>
    <xs:enumeration value="lessThan"/>
    <xs:enumeration value="lessOrEquals"/>
    <xs:enumeration value="greaterThan"/>
    <xs:enumeration value="greaterOrEquals"/>
    <xs:enumeration value="equalsIgnoreCase"/>
    <xs:enumeration value="isSubstring"/>
    <xs:enumeration value="startsWith"/>
    <xs:enumeration value="endsWith"/>
    <xs:enumeration value="startsWithIgnoreCase"/>
    <xs:enumeration value="endsWithIgnoreCase"/>
    <xs:enumeration value="isSubstring"/>
    <xs:enumeration value="hasSubstring"/>
    <xs:enumeration value="isSubstringIgnoreCase"/>
    <xs:enumeration value="hasSubstringIgnoreCase"/>
    <xs:enumeration value="in"/>
    <xs:enumeration value="countEquals"/>
    <xs:enumeration value="countLessThan"/>
    <xs:enumeration value="countLessOrEquals"/>
    <xs:enumeration value="countGreaterThan"/>
    <xs:enumeration value="countGreaterOrEquals"/>
    <xs:enumeration value="contains"/>
    <xs:enumeration value="storageEquals"/>
    <xs:enumeration value="typeEquals"/>
    <xs:enumeration value="directionEquals"/>
    <xs:enumeration value="isMeasureDiscrete"/>
    <xs:enumeration value="isMeasureContinuous"/>
    <xs:enumeration value="isMeasureCollection"/>
    <xs:enumeration value="isMeasureGeospatial"/>
    <xs:enumeration value="isMeasureTypeless"/>
    <xs:enumeration value="isMeasureUnknown"/>
    <xs:enumeration value="isStorageString"/>
    <xs:enumeration value="isStorageNumeric"/>
    <xs:enumeration value="isStorageDatetime"/>
    <xs:enumeration value="isStorageList"/>
    <xs:enumeration value="isStorageUnknown"/>
    <xs:enumeration value="isModelOutput"/>
    <xs:enumeration value="modelOutputRoleEquals"/>
    <xs:enumeration value="modelOutputTargetFieldEquals"/>
    <xs:enumeration value="modelOutputHasValue"/>
    <xs:enumeration value="modelOutputTagEquals"/>
    <xs:enumeration value="enumRestriction"/>
  </xs:attribute>
  <xs:attribute name="value" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="listMode" use="optional" default="all">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="all"/>
        <xs:enumeration value="any"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:element>
```

父元素

And、Command、Constraint、CreateDocument、CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModel、CreateModelApplier、CreateModelOutput、Enabled、ExpertSettings、Not、Option、Or、Run 和 Visible

Constraint 元素

表 71. Constraint 的属性

属性	使用	描述	有效值
property	必需		*string*
singleSelection	可选		*布尔值*

XML 表示法

```
<xs:element name="Constraint">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="singleSelection" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

AutoModeling

子元素

And、Condition、Not 和 Or

Constructors 元素

XML 表示法

```
<xs:element name="Constructors">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="CreateModelOutput"/>
      <xs:element ref="CreateDocumentOutput"/>
      <xs:element ref="CreateInteractiveModelBuilder"/>
      <xs:element ref="CreateInteractiveDocumentBuilder"/>
      <xs:element ref="CreateModelApplier"/>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

DocumentOutput、Execution、InteractiveDocumentBuilder、InteractiveModelBuilder、ModelOutput 和 Node

子元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

Container 元素

表 72. Container 的属性

属性	使用	描述	有效值
name	必需		*string*

表 72. Container 的属性 (续)

属性	使用	描述	有效值
类型	可选		*string*

XML 表示法

```
<xs:element name="Container">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Containers、Containers、Containers、Containers 和 Containers

ContainerFile 元素

表 73. ContainerFile 的属性

属性	使用	描述	有效值
container	可选		
containerType	可选		
path	**必需**		

XML 表示法

```
<xs:element name="ContainerFile" type="SERVER-CONTAINER-FILE">
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
  <xs:attribute name="container" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="containerType" type="EVALUATED-STRING" use="optional"/>
</xs:element>
```

父元素

InputFiles 和 OutputFiles

ContainerTypes 元素

XML 表示法

```
<xs:element name="ContainerTypes">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="DocumentType"/>
      <xs:element ref="ModelType"/>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

DocumentType 和 ModelType

Controls 元素

定义可添加到用户界面对象（例如，菜单和工具栏项）的特定控件。

XML 表示法

```
<xs:element name="Controls">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="Menu"/>
      <xs:element ref="MenuItem"/>
      <xs:element ref="ToolbarItem"/>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

UserInterface

子元素

Menu、MenuItem 和 ToolbarItem

CreateDocument 元素

表 74. CreateDocument 的属性

属性	使用	描述	有效值
sourceFile	必需		*string*
target	必需		*string*
类型	可选		*string*

XML 表示法

```
<xs:element name="CreateDocument">
  <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
    <xs:choice>
      <xs:element ref="Condition"/>
      <xs:element ref="And"/>
      <xs:element ref="Or"/>
      <xs:element ref="Not"/>
    </xs:choice>
  </xs:group>
  <xs:attribute name="sourceFile" type="xs:string" use="required"/>
  <xs:attribute name="target" type="xs:string" use="required"/>
  <xs:attribute name="type" type="xs:string" use="optional"/>
</xs:element>
```

父元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

子元素

And、Condition、Not 和 Or

相关元素

CreateModel

CreateDocumentOutput 元素

表 75. CreateDocumentOutput 的属性

属性	使用	描述	有效值
类型	必需		*string*

XML 表示法

```
<xs:element name="CreateDocumentOutput">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="SetProperty"/>
        <xs:element ref="SetContainer"/>
        <xs:element ref="CreateModel"/>
        <xs:element ref="CreateDocument"/>
      </xs:choice>
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required"/>
</xs:element>
```

父元素

Constructors

子元素

And、Condition、CreateDocument、CreateModel、Not、Or、SetContainer 和 SetProperty

相关元素

CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

CreateInteractiveDocumentBuilder 元素

表 76. CreateInteractiveDocumentBuilder 的属性

属性	使用	描述	有效值
类型	必需		*string*

XML 表示法

```
<xs:element name="CreateInteractiveDocumentBuilder">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="SetProperty"/>
        <xs:element ref="SetContainer"/>
        <xs:element ref="CreateModel"/>
        <xs:element ref="CreateDocument"/>
      </xs:choice>
```

```
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required"/>
</xs:element>
```

父元素

Constructors

子元素

And、Condition、CreateDocument、CreateModel、Not、Or、SetContainer 和 SetProperty

相关元素

CreateDocumentOutput、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

CreateInteractiveModelBuilder 元素

表 77. CreateInteractiveModelBuilder 的属性

属性	使用	描述	有效值
类型	**必需**		*string*

XML 表示法

```
<xs:element name="CreateInteractiveModelBuilder">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="SetProperty"/>
        <xs:element ref="SetContainer"/>
        <xs:element ref="CreateModel"/>
        <xs:element ref="CreateDocument"/>
      </xs:choice>
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required"/>
</xs:element>
```

父元素

Constructors

子元素

And、Condition、CreateDocument、CreateModel、Not、Or、SetContainer 和 SetProperty

相关元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateModelApplier 和 CreateModelOutput

CreateModel 元素

表 78. CreateModel 的属性

属性	使用	描述	有效值
signatureFile	可选		*string*
sourceFile	**必需**		*string*
target	**必需**		*string*
类型	可选		*string*

XML 表示法

```
<xs:element name="CreateModel">
  <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
    <xs:choice>
      <xs:element ref="Condition"/>
      <xs:element ref="And"/>
      <xs:element ref="Or"/>
      <xs:element ref="Not"/>
    </xs:choice>
  </xs:group>
  <xs:attribute name="sourceFile" type="xs:string" use="required"/>
  <xs:attribute name="target" type="xs:string" use="required"/>
  <xs:attribute name="type" type="xs:string" use="optional"/>
  <xs:sequence>
    <xs:element name="ModelDetail" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="signatureFile" type="xs:string" use="optional"/>
</xs:element>
```

父元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

子元素

And、Condition、ModelDetail、Not 和 Or

相关元素

CreateDocument

ModelDetail 元素:

表 79. ModelDetail 的属性

属性	使用	描述	有效值
algorithm	**必需**		*string*

XML 表示法

```
<xs:element name="ModelDetail" maxOccurs="unbounded">
  <xs:attribute name="algorithm" type="xs:string" use="required"/>
</xs:element>
```

父元素

CreateModel

CreateModelApplier 元素

表 80. CreateModelApplier 的属性

属性	使用	描述	有效值
类型	**必需**		*string*

XML 表示法

```
<xs:element name="CreateModelApplier">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="SetProperty"/>
        <xs:element ref="SetContainer"/>
        <xs:element ref="CreateModel"/>
        <xs:element ref="CreateDocument"/>
      </xs:choice>
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required"/>
</xs:element>
```

父元素

Constructors

子元素

And、Condition、CreateDocument、CreateModel、Not、Or、SetContainer 和 SetProperty

相关元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder 和 CreateModelOutput

CreateModelOutput 元素

表 81. CreateModelOutput 的属性

属性	使用	描述	有效值
类型	**必需**		*string*

XML 表示法

```
<xs:element name="CreateModelOutput">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="SetProperty"/>
        <xs:element ref="SetContainer"/>
        <xs:element ref="CreateModel"/>
        <xs:element ref="CreateDocument"/>
      </xs:choice>
```

```
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required"/>
</xs:element>
```

父元素

Constructors

子元素

And、Condition、CreateDocument、CreateModel、Not、Or、SetContainer 和 SetProperty

相关元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder 和 CreateModelApplier

DBConnectionChooserControl 元素

定义可用于选择数据库连接的控件。

表 82. DBConnectionChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="DBConnectionChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

DBTableChooserControl 元素

定义可用于选择数据库表的控件。

表 83. DBTableChooserControl 的属性

属性	使用	描述	有效值
connectionProperty	**必需**		*string*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="DBTableChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="connectionProperty" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

DataFile 元素

表 84. DataFile 的属性

属性	使用	描述	有效值
path	必需		

XML 表示法

```
<xs:element name="DataFile" type="SERVER-DATA-FILE">
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
  <xs:choice>
    <xs:element ref="DelimitedDataFormat"/>
  </xs:choice>
</xs:element>
```

父元素

InputFiles 和 OutputFiles

子元素

DelimitedDataFormat

DataFormat 元素

XML 表示法

```
<xs:element name="DataFormat">
  <xs:group ref="DATA-FORMAT-TYPE">
    <xs:choice>
      <xs:element ref="DelimitedDataFormat"/>
      <xs:element ref="SPSSDataFormat"/>
    </xs:choice>
  </xs:group>
</xs:element>
```

父元素

FileFormatType

子元素

DelimitedDataFormat 和 SPSSDataFormat

DataModel 元素

进/出节点的数据模型。输入/输出数据模型是一个字段集。

XML 表示法

```
<xs:element name="DataModel" type="DATA-MODEL">
  <xs:sequence>
    <xs:element name="FieldFormats" type="FIELD-FORMATS" minOccurs="0">
      <xs:sequence>
        <xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION" minOccurs="0" maxOccurs="unbounded">
        </xs:element>
      </xs:sequence>
    </xs:element>
    <xs:element name="FieldGroups" type="FIELD-GROUPS" minOccurs="0">
      <xs:sequence>
        <xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION" minOccurs="0" maxOccurs="unbounded">
          <xs:sequence>
            <xs:element name="FieldName">
            </xs:element>
          </xs:sequence>
        </xs:element>
      </xs:sequence>
    </xs:element>
    <xs:element name="Fields" type="FIELDS">
      <xs:sequence>
        <xs:element name="Field" type="FIELD" minOccurs="0" maxOccurs="unbounded">
          <xs:group ref="FIELD-CONTENT">
            <xs:sequence>
              <xs:element ref="DisplayLabel"/>
              <xs:choice minOccurs="0">
                <xs:element ref="Range"/>
                <xs:element ref="Values"/>
              </xs:choice>
              <xs:element ref="MissingValues"/>
            </xs:sequence>
          </xs:group>
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
</xs:element>
```

子元素

FieldFormats、FieldGroups 和 Fields

FieldFormats 元素: 定义缺省字段格式。在输出中显示值时将使用字段格式，例如，一般格式（标准数字、科学或货币格式）、要显示的小数位数和十进制分隔符等。当前字段格式仅用于数字字段，但这可能在未来的版本中进行更改。

表 85. FieldFormats 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*
defaultNumberFormat	**必需**		*string*

XML 表示法

```
<xs:element name="FieldFormats" type="FIELD-FORMATS" minOccurs="0">
  <xs:sequence>
    <xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="defaultNumberFormat" type="xs:string" use="required"/>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

DataModel

子元素

NumberFormat

NumberFormat 元素: 为数字字段定义格式信息。

表 86. NumberFormat 的属性

属性	使用	描述	有效值
decimalPlaces	必需		*nonNegativeInteger*
decimalSymbol	必需		句号 逗号
formatType	必需		**standard** 科学表示法 货币
groupingSymbol	必需		**none** 句号 逗号 **space**
name	必需		*string*

XML 表示法

```
<xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="formatType" type="NUMBER-FORMAT-TYPE" use="required">
    <xs:enumeration value="standard"/>
    <xs:enumeration value="scientific"/>
    <xs:enumeration value="currency"/>
  </xs:attribute>
  <xs:attribute name="decimalPlaces" type="xs:nonNegativeInteger" use="required"/>
  <xs:attribute name="decimalSymbol" type="DECIMAL-SYMBOL" use="required">
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
  </xs:attribute>
  <xs:attribute name="groupingSymbol" type="NUMBER-GROUPING-SYMBOL" use="required">
    <xs:enumeration value="none"/>
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
    <xs:enumeration value="space"/>
  </xs:attribute>
</xs:element>
```

父元素

FieldFormats

FieldGroups 元素: 定义字段组。字段组用于关联相关的字段。例如，要求调查对象从一组选项中选择他们已访问过哪些位置的调查问题将使用一组标记字段来表示。字段组可用于确定哪些字段与该调查问题相关联。

表 87. FieldGroups 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="FieldGroups" type="FIELD-GROUPS" minOccurs="0">
  <xs:sequence>
    <xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION" minOccurs="0" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="FieldName">
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

DataModel

子元素

FieldGroup

FieldGroup 元素: 定义字段组。字段组由字段名称列表和关于字段组的信息（例如，组名称和可选标签、组类型和多命题组的类型以及计数值（例如，标识“true”的值））组成。

表 88. FieldGroup 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*
countedValue	可选		*string*
displayLabel	可选		*string*
groupType	**必需**		**fieldGroup** **multiCategorySet** **multiDichotomySet**
name	**必需**		

XML 表示法

```
<xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION" minOccurs="0" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="FieldName">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-GROUP-NAME" use="required"/>
  <xs:attribute name="displayLabel" type="xs:string"/>
  <xs:attribute name="groupType" type="FIELD-GROUP-TYPE" use="required">
    <xs:enumeration value="fieldGroup"/>
    <xs:enumeration value="multiCategorySet"/>
    <xs:enumeration value="multiDichotomySet"/>
  </xs:attribute>
  <xs:attribute name="countedValue" type="xs:string"/>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

FieldGroups

子元素

FieldName

FieldName 元素:

表 89. FieldName 的属性

属性	使用	描述	有效值
name	必需		

XML 表示法

```
<xs:element name="FieldName">
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
</xs:element>
```

父元素

FieldGroup

Fields 元素:

表 90. Fields 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="Fields" type="FIELDS">
  <xs:sequence>
    <xs:element name="Field" type="FIELD" minOccurs="0" maxOccurs="unbounded">
      <xs:group ref="FIELD-CONTENT">
        <xs:sequence>
          <xs:element ref="DisplayLabel"/>
          <xs:choice minOccurs="0">
            <xs:element ref="Range"/>
            <xs:element ref="Values"/>
          </xs:choice>
          <xs:element ref="MissingValues"/>
        </xs:sequence>
      </xs:group>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

DataModel

子元素

Field

Field 元素:

表 91. Field 的属性

属性	使用	描述	有效值
direction	可选		**in** **out** **both** **none** **partition**
displayLabel	可选		*string*

表 91. Field 的属性 (续)

属性	使用	描述	有效值
name	**必需**		*string*
storage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**
类型	可选		**auto** **range** **discrete** **set** **orderedSet** **flag** **typeless** **collection** **geospatial**

XML 表示法

```
<xs:element name="Field" type="FIELD" minOccurs="0" maxOccurs="unbounded">
  <xs:group ref="FIELD-CONTENT">
    <xs:sequence>
      <xs:element ref="DisplayLabel"/>
      <xs:choice minOccurs="0">
        <xs:element ref="Range"/>
        <xs:element ref="Values"/>
      </xs:choice>
      <xs:element ref="MissingValues"/>
    </xs:sequence>
  </xs:group>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="type" type="FIELD-TYPE" default="auto">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="range"/>
    <xs:enumeration value="discrete"/>
    <xs:enumeration value="set"/>
    <xs:enumeration value="orderedSet"/>
    <xs:enumeration value="flag"/>
    <xs:enumeration value="typeless"/>
    <xs:enumeration value="collection"/>
    <xs:enumeration value="geospatial"/>
  </xs:attribute>
  <xs:attribute name="storage" type="FIELD-STORAGE" default="unknown">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
  <xs:attribute name="direction" type="FIELD-DIRECTION" default="in">
    <xs:enumeration value="in"/>
    <xs:enumeration value="out"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="partition"/>
  </xs:attribute>
  <xs:attribute name="displayLabel" type="xs:string"/>
</xs:element>
```

父元素

字段

子元素

DisplayLabel、MissingValues、Range、Range、Values 和 Values

DatabaseConnectionValue 元素

用于指定数据库连接详细信息的值。

表 92. DatabaseConnectionValue 的属性

属性	使用	描述	有效值
connectionType	必需		*string*
datasourceName	必需		*string*
密码	必需		*string*
userName	必需		*string*

XML 表示法

```
<xs:element name="DatabaseConnectionValue" type="DATABASE-CONNECTION-VALUE">
  <xs:attribute name="connectionType" type="xs:string" use="required"/>
  <xs:attribute name="datasourceName" type="xs:string" use="required"/>
  <xs:attribute name="userName" type="xs:string" use="required"/>
  <xs:attribute name="password" type="xs:string" use="required"/>
</xs:element>
```

父元素

Attribute、Attribute、ListValue、ListValue、ListValue 和 Parameter

DefaultValue 元素

XML 表示法

```
<xs:element name="DefaultValue">
  <xs:choice>
    <xs:element name="ServerTempFile">
    </xs:element>
    <xs:element name="ServerTempDir">
    </xs:element>
    <xs:element name="Identifier">
    </xs:element>
  </xs:choice>
</xs:element>
```

父元素

Property 和 PropertyType

子元素

Identifier、ServerTempDir 和 ServerTempFile

ServerTempFile 元素:

表 93. ServerTempFile 的属性

属性	使用	描述	有效值
basename	必需		

XML 表示法

```
<xs:element name="ServerTempFile">
  <xs:attribute name="basename" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

DefaultValue

ServerTempDir 元素:

表 94. ServerTempDir 的属性

属性	使用	描述	有效值
basename	必需		

XML 表示法

```
<xs:element name="ServerTempDir">
  <xs:attribute name="basename" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

DefaultValue

Identifier 元素:

表 95. Identifier 的属性

属性	使用	描述	有效值
basename	必需		

XML 表示法

```
<xs:element name="Identifier">
  <xs:attribute name="basename" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

DefaultValue

DelimitedDataFormat 元素

表 96. DelimitedDataFormat 的属性

属性	使用	描述	有效值
delimeter	可选		**tab** 逗号 **semicolon** **colon** **verticalBar** **other**

表 96. DelimitedDataFormat 的属性 (续)

属性	使用	描述	有效值
eol	可选		**cr** **crlf** **lf** **other**
includeFieldNames	可选		*布尔值*
otherDelimiter	可选		*string*
otherEol	可选		*string*
quoteStrings	可选		*布尔值*
stringQuote	可选		*string*

XML 表示法

```
<xs:element name="DelimitedDataFormat">
  <xs:attribute name="delimeter" use="optional" default="tab">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="tab"/>
        <xs:enumeration value="comma"/>
        <xs:enumeration value="semicolon"/>
        <xs:enumeration value="colon"/>
        <xs:enumeration value="verticalBar"/>
        <xs:enumeration value="other"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="otherDelimiter" type="xs:string" use="optional"/>
  <xs:attribute name="eol" use="optional" default="cr">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="cr"/>
        <xs:enumeration value="crlf"/>
        <xs:enumeration value="lf"/>
        <xs:enumeration value="other"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="otherEol" type="xs:string" use="optional"/>
  <xs:attribute name="includeFieldNames" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="quoteStrings" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="stringQuote" type="xs:string" use="optional" default="""/>
</xs:element>
```

父元素

DataFile 和 DataFormat

DisplayLabel 元素

指定语言中的字段或值的显示标签。可以在特定语言没有标签的地方使用 displayLabel 属性。

表 97. DisplayLabel 的属性

属性	使用	描述	有效值
lang	可选		*NMTOKEN*
value	**必需**		*string*

XML 表示法

```
<xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="lang" type="xs:NMTOKEN" default="en"/>
</xs:element>
```

父元素

Field

DocumentBuilder 元素

XML 表示法

```
<xs:element name="DocumentBuilder">
  <xs:sequence>
    <xs:element name="DocumentGeneration">
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

DocumentGeneration

DocumentGeneration 元素:

表 98. DocumentGeneration 的属性

属性	使用	描述	有效值
controlsId	必需		*string*

XML 表示法

```
<xs:element name="DocumentGeneration">
  <xs:attribute name="controlsId" type="xs:string" use="required"/>
</xs:element>
```

父元素

DocumentBuilder

DocumentOutput 元素

表 99. DocumentOutput 的属性

属性	使用	描述	有效值
delegate	可选		*string*
deprecatedScriptNames	可选		*string*
id	必需		*string*
scriptName	可选		*string*

XML 表示法

```
<xs:element name="DocumentOutput">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="Properties"/>
      <xs:element name="Containers" minOccurs="0">
        <xs:sequence maxOccurs="unbounded">
          <xs:element ref="Container"/>
        </xs:sequence>
      </xs:element>
      <xs:element ref="UserInterface"/>
      <xs:element ref="Constructors" minOccurs="0"/>
      <xs:element ref="ModelProvider" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="delegate" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Constructors、Containers、ModelProvider、Properties 和 UserInterface

相关元素

InteractiveDocumentBuilder、InteractiveModelBuilder、ModelOutput 和 Node

Containers 元素:

XML 表示法

```
<xs:element name="Containers" minOccurs="0">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Container"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Container

DocumentType 元素

定义新的文档类型

表 100. DocumentType 的属性

属性	使用	描述	有效值
format	必需		**utf8** **binary**
id	必需		*string*

表 100. DocumentType 的属性 (续)

属性	使用	描述	有效值
类型	可选		**unknown** **rowSet** **report** **graph**

XML 表示法

```
<xs:element name="DocumentType">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="format" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="utf8"/>
        <xs:enumeration value="binary"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="type" type="DOCUMENT-TYPE" use="optional">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="rowSet"/>
    <xs:enumeration value="report"/>
    <xs:enumeration value="graph"/>
  </xs:attribute>
</xs:element>
```

父元素

ContainerTypes

相关元素

ModelType

Enabled 元素

定义 UI 组件应启用或可编辑的条件。

XML 表示法

```
<xs:element name="Enabled">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

ActionButton、CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、ComboBoxControl、DBConnectionChooserControl、DBTableChooserControl、ExtensionObjectPanel、FieldAllocationList、ItemChooserControl、ModelViewerPanel、MultiFieldAllocationControl、MultiFieldChooserControl、MultiItemChooserControl、PasswordBoxControl、PropertiesPanel、PropertiesSubPanel、PropertyControl、RadioButtonGroupControl、SelectorPanel、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、

SingleFieldChooserControl、SingleFieldValueChooserControl、SingleItemChooserControl、SpinnerControl、StaticText、SystemControls、TabbedPanel、TableControl、TextAreaControl、TextBoxControl 和 TextBrowserPanel

子元素

And、Condition、Not 和 Or

Enumeration 元素

XML 表示法

```
<xs:element name="Enumeration">
  <xs:sequence>
    <xs:element name="Enum" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

PropertyType

子元素

Enum

Enum 元素:

表 101. Enum 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
imagePath	可选		*string*
imagePathKey	可选		*string*
label	**必需**		*string*
labelKey	可选		*string*
value	**必需**		*string*

XML 表示法

```
<xs:element name="Enum" maxOccurs="unbounded">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="imagePath" type="xs:string" use="optional"/>
  <xs:attribute name="imagePathKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Enumeration

ErrorDetail 元素

关于错误或其他情况的补充信息。

XML 表示法

```
<xs:element name="ErrorDetail" type="ERROR-DETAIL">
  <xs:sequence>
    <xs:element name="Diagnostic" type="DIAGNOSTIC" minOccurs="0" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
        </xs:element>
        <xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
</xs:element>
```

子元素

Diagnostic

Diagnostic 元素:

表 102. Diagnostic 的属性

属性	使用	描述	有效值
code	**必需**		*整数*
severity	可选		**unknown** **信息** **warning** **error** **fatal**
source	可选		*string*
subCode	可选		*整数*

XML 表示法

```
<xs:element name="Diagnostic" type="DIAGNOSTIC" minOccurs="0" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
    </xs:element>
    <xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="code" type="xs:integer" use="required"/>
  <xs:attribute name="subCode" type="xs:integer" default="0"/>
  <xs:attribute name="severity" type="DIAGNOSTIC-SEVERITY" default="error">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="information"/>
    <xs:enumeration value="warning"/>
    <xs:enumeration value="error"/>
    <xs:enumeration value="fatal"/>
  </xs:attribute>
  <xs:attribute name="source" type="xs:string"/>
</xs:element>
```

父元素

ErrorDetail

子元素

Message 和 Parameter

Message 元素:

表 103. Message 的属性

属性	使用	描述	有效值
lang	可选		*NMTOKEN*

XML 表示法

```
<xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
  <xs:attribute name="lang" type="xs:NMTOKEN"/>
</xs:element>
```

父元素

Diagnostic

Parameter 元素:

XML 表示法

```
<xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
```

父元素

Diagnostic

Executable 元素

XML 表示法

```
<xs:element name="Executable">
  <xs:sequence>
    <xs:element ref="Run" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

Execution

子元素

Run

Execution 元素

XML 表示法

```
<xs:element name="Execution">
  <xs:sequence>
    <xs:element ref="Properties" minOccurs="0"/>
    <xs:element ref="InputFiles"/>
    <xs:element ref="OutputFiles"/>
    <xs:choice>
      <xs:element ref="Executable"/>
      <xs:element ref="Module"/>
    </xs:choice>
    <xs:element ref="Constructors" minOccurs="0"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Constructors、Executable、InputFiles、Module、OutputFiles 和 Properties

Extension 元素

定义顶级扩展容器。

表 104. Extension 的属性

属性	使用	描述	有效值
debug	可选		*布尔值*
version	**必需**		*string*

XML 表示法

```
<xs:element name="Extension">
  <xs:sequence>
    <xs:element ref="ExtensionDetails"/>
    <xs:element ref="Resources"/>
    <xs:element ref="License" minOccurs="0"/>
    <xs:element ref="CommonObjects"/>
    <xs:element ref="UserInterface" minOccurs="0"/>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element ref="Node"/>
        <xs:element ref="ModelOutput"/>
        <xs:element ref="DocumentOutput"/>
        <xs:element ref="InteractiveModelBuilder"/>
        <xs:element ref="InteractiveDocumentBuilder"/>
      </xs:choice>
    </xs:sequence>
  </xs:sequence>
  <xs:attribute name="version" type="xs:string" use="required"/>
  <xs:attribute name="debug" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

子元素

CommonObjects、DocumentOutput、ExtensionDetails、InteractiveDocumentBuilder、InteractiveModelBuilder、License、ModelOutput、Node、Resources 和 UserInterface

ExtensionDetails 元素

定义关于扩展的信息，例如，扩展标识、扩展提供者和版本信息。

表 105. ExtensionDetails 的属性

属性	使用	描述	有效值
copyright	可选		*string*
描述	可选		*string*
id	**必需**		*string*
label	**必需**		*string*
provider	可选		*string*
providerTag	**必需**		*string*
version	可选		*string*

XML 表示法

```
<xs:element name="ExtensionDetails">
  <xs:attribute name="providerTag" type="xs:string" use="required"/>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
```

```
  <xs:attribute name="version" type="xs:string"/>
  <xs:attribute name="provider" type="xs:string" use="optional" default="(unknown)"/>
  <xs:attribute name="copyright" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

ExtensionObjectPanel 元素

定义定制面板。panelClass 属性所确定的类必须实现 ExtensionObjectPanel 界面。

表 106. ExtensionObjectPanel 的属性

属性	使用	描述	有效值
id	可选		*string*
panelClass	**必需**		*string*

XML 表示法

```
<xs:element name="ExtensionObjectPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="panelClass" type="xs:string" use="required"/>
  <xs:attribute name="id" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel、PropertiesSubPanel 和 Tab

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

Field 元素

表 107. Field 的属性

属性	使用	描述	有效值
depth	可选		*整数*
direction	可选		**in** **out** **both** **none** **partition**
label	可选		*string*

表 107. Field 的属性 (续)

属性	使用	描述	有效值
name	**必需**		
storage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**
类型	可选		**auto** **range** **discrete** **set** **orderedSet** **flag** **typeless** **collection** **geospatial**
valueStorage	可选		**unknown** **整数** **real** **string** **date** **time** **timestamp** **list**

XML 表示法

```
<xs:element name="Field" type="FIELD-DECLARATION">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Range" minOccurs="0"/>
      <xs:element ref="Values" minOccurs="0"/>
      <xs:element ref="NumericInfo" minOccurs="0"/>
      <xs:element name="MissingValues" minOccurs="0">
        <xs:sequence>
          <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
          <xs:element ref="Range" minOccurs="0"/>
        </xs:sequence>
      </xs:element>
      <xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
      </xs:element>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
  <xs:attribute name="storage" type="FIELD-STORAGE">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
```

```
  <xs:attribute name="type" type="FIELD-TYPE">
    <xs:enumeration value="auto"/>
    <xs:enumeration value="range"/>
    <xs:enumeration value="discrete"/>
    <xs:enumeration value="set"/>
    <xs:enumeration value="orderedSet"/>
    <xs:enumeration value="flag"/>
    <xs:enumeration value="typeless"/>
    <xs:enumeration value="collection"/>
    <xs:enumeration value="geospatial"/>
  </xs:attribute>
  <xs:attribute name="direction" type="FIELD-DIRECTION">
    <xs:enumeration value="in"/>
    <xs:enumeration value="out"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="partition"/>
  </xs:attribute>
  <xs:attribute name="label" type="xs:string"/>
  <xs:attribute name="depth" type="xs:integer" use="optional" default="-1"/>
  <xs:attribute name="valueStorage" type="FIELD-STORAGE" use="optional">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="real"/>
    <xs:enumeration value="string"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="time"/>
    <xs:enumeration value="timestamp"/>
    <xs:enumeration value="list"/>
  </xs:attribute>
</xs:element>
```

子元素

MissingValues、ModelField、NumericInfo、Range、Range、Values 和 Values

MissingValues 元素:

表 108. MissingValues 的属性

属性	使用	描述	有效值
treatNullAsMissing	可选		*布尔值*
treatWhitespaceAsMissing	可选		*布尔值*

XML 表示法

```
<xs:element name="MissingValues" minOccurs="0">
  <xs:sequence>
    <xs:element ref="Values" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="Range" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="treatWhitespaceAsMissing" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="treatNullAsMissing" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

AddField

子元素

Range、Range、Values 和 Values

ModelField 元素:

表 109. ModelField 的属性

属性	使用	描述	有效值
group	可选		

表 109. ModelField 的属性 (续)

属性	使用	描述	有效值
role	**必需**		**unknown** **predictedValue** **predictedDisplayValue** **概率** **residual** **standardError** **entityId** **entityAffinity** **upperConfidenceLimit** **lowerConfidenceLimit** **propensity** **value** **supplementary**
tag	可选		*string*
targetField	可选		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="ModelField" type="MODEL-FIELD-INFORMATION" minOccurs="0">
  <xs:attribute name="role" type="MODEL-FIELD-ROLE" use="required">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="predictedValue"/>
    <xs:enumeration value="predictedDisplayValue"/>
    <xs:enumeration value="probability"/>
    <xs:enumeration value="residual"/>
    <xs:enumeration value="standardError"/>
    <xs:enumeration value="entityId"/>
    <xs:enumeration value="entityAffinity"/>
    <xs:enumeration value="upperConfidenceLimit"/>
    <xs:enumeration value="lowerConfidenceLimit"/>
    <xs:enumeration value="propensity"/>
    <xs:enumeration value="value"/>
    <xs:enumeration value="supplementary"/>
  </xs:attribute>
  <xs:attribute name="targetField" type="xs:string"/>
  <xs:attribute name="value" type="xs:string"/>
  <xs:attribute name="group" type="MODEL-FIELD-GROUP"/>
  <xs:attribute name="tag" type="xs:string"/>
</xs:element>
```

父元素

AddField

FieldAllocationList 元素

定义包含可用字段列表的面板。单字段和多字段分配控件可从此面板分配字段。

表 110. FieldAllocationList 的属性

属性	使用	描述	有效值
enabledProperty	可选		*string*
enabledValue	可选		*string*
id	**必需**		*string*

XML 表示法

```
<xs:element name="FieldAllocationList">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="enabledProperty" type="xs:string" use="optional"/>
  <xs:attribute name="enabledValue" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、ModelViewerPanel、SelectorPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

FieldFormats 元素

定义缺省字段格式。在输出中显示值时将使用字段格式，例如，一般格式（标准数字、科学或货币格式）、要显示的小数位数和十进制分隔符等。当前字段格式仅用于数字字段，但这可能在未来的版本中进行更改。

表 111. FieldFormats 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*
defaultNumberFormat	必需		*string*

XML 表示法

```
<xs:element name="FieldFormats" type="FIELD-FORMATS">
  <xs:sequence>
    <xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="defaultNumberFormat" type="xs:string" use="required"/>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

子元素

NumberFormat

NumberFormat 元素: 为数字字段定义格式信息。

表 112. NumberFormat 的属性

属性	使用	描述	有效值
decimalPlaces	必需		*nonNegativeInteger*
decimalSymbol	必需		句号 逗号

表 112. NumberFormat 的属性 (续)

属性	使用	描述	有效值
formatType	必需		**standard** 科学表示法 货币
groupingSymbol	必需		**none** 句号 逗号 **space**
name	必需		*string*

XML 表示法

```
<xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="formatType" type="NUMBER-FORMAT-TYPE" use="required">
    <xs:enumeration value="standard"/>
    <xs:enumeration value="scientific"/>
    <xs:enumeration value="currency"/>
  </xs:attribute>
  <xs:attribute name="decimalPlaces" type="xs:nonNegativeInteger" use="required"/>
  <xs:attribute name="decimalSymbol" type="DECIMAL-SYMBOL" use="required">
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
  </xs:attribute>
  <xs:attribute name="groupingSymbol" type="NUMBER-GROUPING-SYMBOL" use="required">
    <xs:enumeration value="none"/>
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
    <xs:enumeration value="space"/>
  </xs:attribute>
</xs:element>
```

父元素

FieldFormats

FieldGroup 元素

定义字段组。字段组由字段名称列表和关于字段组的信息（例如，组名称和可选标签、组类型和多命题组的类型以及计数值（例如，标识“true”的值））组成。

表 113. FieldGroup 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*
countedValue	可选		*string*
displayLabel	可选		*string*
groupType	必需		**fieldGroup** **multiCategorySet** **multiDichotomySet**
name	必需		

XML 表示法

```
<xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION">
  <xs:sequence>
    <xs:element name="FieldName">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-GROUP-NAME" use="required"/>
  <xs:attribute name="displayLabel" type="xs:string"/>
  <xs:attribute name="groupType" type="FIELD-GROUP-TYPE" use="required">
    <xs:enumeration value="fieldGroup"/>
    <xs:enumeration value="multiCategorySet"/>
    <xs:enumeration value="multiDichotomySet"/>
  </xs:attribute>
  <xs:attribute name="countedValue" type="xs:string"/>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

子元素

FieldName

FieldName 元素:

表 114. FieldName 的属性

属性	使用	描述	有效值
name	必需		

XML 表示法

```
<xs:element name="FieldName">
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
</xs:element>
```

父元素

FieldGroup

FieldGroups 元素

定义字段组。字段组用于关联相关的字段。例如，要求调查对象从一组选项中选择他们已访问过哪些位置的调查问题将使用一组标记字段来表示。字段组可用于确定哪些字段与该调查问题相关联。

表 115. FieldGroups 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="FieldGroups" type="FIELD-GROUPS">
  <xs:sequence>
    <xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION" minOccurs="0" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="FieldName">
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

子元素

FieldGroup

FieldGroup 元素: 定义字段组。字段组由字段名称列表和关于字段组的信息（例如，组名称和可选标签、组类型和多命题组的类型以及计数值（例如，标识“true”的值））组成。

表 116. FieldGroup 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*
countedValue	可选		*string*
displayLabel	可选		*string*
groupType	**必需**		**fieldGroup** **multiCategorySet** **multiDichotomySet**
name	**必需**		

XML 表示法

```
<xs:element name="FieldGroup" type="FIELD-GROUP-DECLARATION" minOccurs="0" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="FieldName">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="FIELD-GROUP-NAME" use="required"/>
  <xs:attribute name="displayLabel" type="xs:string"/>
  <xs:attribute name="groupType" type="FIELD-GROUP-TYPE" use="required">
    <xs:enumeration value="fieldGroup"/>
    <xs:enumeration value="multiCategorySet"/>
    <xs:enumeration value="multiDichotomySet"/>
  </xs:attribute>
  <xs:attribute name="countedValue" type="xs:string"/>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

FieldGroups

子元素

FieldName

FieldName 元素:

表 117. FieldName 的属性

属性	使用	描述	有效值
name	**必需**		

XML 表示法

```
<xs:element name="FieldName">
  <xs:attribute name="name" type="FIELD-NAME" use="required"/>
</xs:element>
```

父元素

FieldGroup

FileFormatType 元素

表 118. FileFormatType 的属性

属性	使用	描述	有效值
name	可选		

XML 表示法

```
<xs:element name="FileFormatType">
  <xs:sequence>
    <xs:group ref="FILE-FORMAT">
      <xs:choice>
        <xs:element ref="UTF8Format"/>
        <xs:element ref="BinaryFormat"/>
        <xs:element ref="DataFormat"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
  <xs:attribute name="name" type="EVALUATED-STRING" use="optional"/>
</xs:element>
```

父元素

FileFormatTypes

子元素

BinaryFormat、DataFormat 和 UTF8Format

FileFormatTypes 元素

XML 表示法

```
<xs:element name="FileFormatTypes">
  <xs:sequence>
    <xs:element ref="FileFormatType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

FileFormatType

ForEach 元素

表 119. ForEach 的属性

属性	使用	描述	有效值
container	可选		*string*
from	可选		*string*
inFieldValues	可选		*string*
inFields	可选		*string*
inProperty	可选		*string*
step	可选		*string*
to	可选		*string*

表 119. ForEach 的属性 (续)

属性	使用	描述	有效值
变量	**必需**		*string*

XML 表示法

```
<xs:element name="ForEach">
  <xs:sequence maxOccurs="unbounded">
    <xs:group ref="DATA-MODEL-EXPRESSION">
      <xs:choice>
        <xs:element ref="ForEach"/>
        <xs:element ref="AddField"/>
        <xs:element ref="ChangeField"/>
        <xs:element ref="RemoveField"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
  <xs:attribute name="var" type="xs:string" use="required"/>
  <xs:attribute name="inProperty" type="xs:string" use="optional"/>
  <xs:attribute name="inFields" type="xs:string" use="optional"/>
  <xs:attribute name="inFieldValues" type="xs:string" use="optional"/>
  <xs:attribute name="from" type="xs:string" use="optional"/>
  <xs:attribute name="to" type="xs:string" use="optional"/>
  <xs:attribute name="step" type="xs:string" use="optional"/>
  <xs:attribute name="container" type="xs:string" use="optional"/>
</xs:element>
```

父元素

ForEach 和 ModelFields

子元素

AddField、ChangeField、ForEach 和 RemoveField

Icon 元素

表 120. Icon 的属性

属性	使用	描述	有效值
imagePath	**必需**		*string*
resourceID	可选		*string*
类型	**必需**		**standardNode** **smallNode** **standardWindow**

XML 表示法

```
<xs:element name="Icon">
  <xs:attribute name="type" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="standardNode"/>
        <xs:enumeration value="smallNode"/>
        <xs:enumeration value="standardWindow"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="imagePath" type="xs:string" use="required"/>
  <xs:attribute name="resourceID" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Icons 和 Palette

Icons 元素

XML 表示法

```
<xs:element name="Icons">
  <xs:sequence>
    <xs:element ref="Icon" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

UserInterface

子元素

图标

InputFiles 元素

XML 表示法

```
<xs:element name="InputFiles">
  <xs:group ref="RUNTIME-FILES">
    <xs:sequence>
      <xs:element ref="DataFile"/>
      <xs:element ref="ContainerFile" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:group>
</xs:element>
```

父元素

Execution 和 Module

子元素

ContainerFile 和 DataFile

InteractiveDocumentBuilder 元素

表 121. InteractiveDocumentBuilder 的属性

属性	使用	描述	有效值
delegate	可选		*string*
deprecatedScriptNames	可选		*string*
id	**必需**		*string*
scriptName	可选		*string*

XML 表示法

```
<xs:element name="InteractiveDocumentBuilder">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="Properties"/>
      <xs:element name="Containers" minOccurs="0">
        <xs:sequence maxOccurs="unbounded">
          <xs:element ref="Container"/>
        </xs:sequence>
```

```
      </xs:element>
      <xs:element ref="UserInterface"/>
      <xs:element ref="Constructors" minOccurs="0"/>
      <xs:element ref="ModelProvider" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="delegate" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Constructors、Containers、ModelProvider、Properties 和 UserInterface

相关元素

DocumentOutput、InteractiveModelBuilder、ModelOutput 和 Node

Containers 元素:

XML 表示法

```
<xs:element name="Containers" minOccurs="0">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Container"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Container

InteractiveModelBuilder 元素

表 122. InteractiveModelBuilder 的属性

属性	使用	描述	有效值
delegate	可选		*string*
deprecatedScriptNames	可选		*string*
id	**必需**		*string*
scriptName	可选		*string*

XML 表示法

```
<xs:element name="InteractiveModelBuilder">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="Properties"/>
      <xs:element name="Containers" minOccurs="0">
        <xs:sequence maxOccurs="unbounded">
          <xs:element ref="Container"/>
        </xs:sequence>
      </xs:element>
      <xs:element ref="UserInterface"/>
```

```
      <xs:element ref="Constructors" minOccurs="0"/>
      <xs:element ref="ModelProvider" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="delegate" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Constructors、Containers、ModelProvider、Properties 和 UserInterface

相关元素

DocumentOutput、InteractiveDocumentBuilder、ModelOutput 和 Node

Containers 元素:

XML 表示法

```
<xs:element name="Containers" minOccurs="0">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Container"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Container

Layout 元素

定义 UI 组件在面板中的放置方式。布局基于网格组装布局，并且每个控件占据网格中的一个或多个单元。布局允许指定调整大小、填充、锚点和缩进行为。

表 123. Layout 的属性

属性	使用	描述	有效值
anchor	可选		**north** **northeast** **east** **southeast** **south** **southwest** **west** **northwest** **center**
columnWeight	可选		*double*

表 123. Layout 的属性 (续)

属性	使用	描述	有效值
fill	可选		**horizontal** **vertical** **both** **none**
gridColumn	可选		*nonNegativeInteger*
gridHeight	可选		*nonNegativeInteger*
gridRow	可选		*nonNegativeInteger*
gridWidth	可选		*nonNegativeInteger*
leftIndent	可选		*nonNegativeInteger*
rowIncrement	可选		*nonNegativeInteger*
rowWeight	可选		*double*

XML 表示法

```
<xs:element name="Layout">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Cell">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="gridRow" type="xs:nonNegativeInteger" use="optional"/>
  <xs:attribute name="gridColumn" type="xs:nonNegativeInteger" use="optional"/>
  <xs:attribute name="rowIncrement" type="xs:nonNegativeInteger" use="optional"/>
  <xs:attribute name="gridWidth" type="xs:nonNegativeInteger" use="optional" default="1"/>
  <xs:attribute name="gridHeight" type="xs:nonNegativeInteger" use="optional" default="1"/>
  <xs:attribute name="rowWeight" type="xs:double" use="optional"/>
  <xs:attribute name="columnWeight" type="xs:double" use="optional"/>
  <xs:attribute name="fill" type="UI-COMPONENT-FILL" use="optional" default="none">
    <xs:enumeration value="horizontal"/>
    <xs:enumeration value="vertical"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
  </xs:attribute>
  <xs:attribute name="anchor" type="UI-COMPONENT-ANCHOR" use="optional" default="west">
    <xs:enumeration value="north"/>
    <xs:enumeration value="northeast"/>
    <xs:enumeration value="east"/>
    <xs:enumeration value="southeast"/>
    <xs:enumeration value="south"/>
    <xs:enumeration value="southwest"/>
    <xs:enumeration value="west"/>
    <xs:enumeration value="northwest"/>
    <xs:enumeration value="center"/>
  </xs:attribute>
  <xs:attribute name="leftIndent" type="xs:nonNegativeInteger" use="optional"/>
</xs:element>
```

父元素

ActionButton、CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、ComboBoxControl、DBConnectionChooserControl、DBTableChooserControl、ExtensionObjectPanel、FieldAllocationList、ItemChooserControl、ModelViewerPanel、MultiFieldAllocationControl、MultiFieldChooserControl、MultiItemChooserControl、PasswordBoxControl、PropertiesPanel、PropertiesSubPanel、PropertyControl、RadioButtonGroupControl、SelectorPanel、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SingleItemChooserControl、SpinnerControl、StaticText、SystemControls、TabbedPanel、TableControl、TextAreaControl、TextBoxControl 和 TextBrowserPanel

子元素

Cell

Cell 元素:

表 124. Cell 的属性

属性	使用	描述	有效值
列	**必需**		*nonNegativeInteger*
row	**必需**		*nonNegativeInteger*
width	**必需**		*nonNegativeInteger*

XML 表示法

```
<xs:element name="Cell">
  <xs:attribute name="row" type="xs:nonNegativeInteger" use="required"/>
  <xs:attribute name="column" type="xs:nonNegativeInteger" use="required"/>
  <xs:attribute name="width" type="xs:nonNegativeInteger" use="required"/>
</xs:element>
```

父元素

Layout

License 元素

保留供系统使用。

XML 表示法

```
<xs:element name="License">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="OptionCode"/>
  </xs:sequence>
</xs:element>
```

父元素

Extension

子元素

OptionCode

ListValue 元素

值序列。所有值的内容类型必须相同，但是不会对此进行检查。

XML 表示法

```
<xs:element name="ListValue" type="LIST-VALUE">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
</xs:element>
```

父元素

Attribute、Attribute、ListValue、ListValue、ListValue 和 Parameter

子元素

DatabaseConnectionValue、ListValue、MapValue、StructuredValue 和 Value

MapValue 元素

映射条目的集合，每个映射条目由一个键和一个值组成。

XML 表示法

```
<xs:element name="MapValue" type="MAP-VALUE">
  <xs:sequence>
    <xs:element name="MapEntry" type="MAP-ENTRY" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="KeyValue" type="KEY-VALUE">
        </xs:element>
        <xs:element name="StructuredValue" type="STRUCTURED-VALUE">
          <xs:sequence>
            <xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
              <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
                <xs:choice>
                  <xs:element ref="MapValue"/>
                  <xs:element ref="StructuredValue"/>
                  <xs:element ref="ListValue"/>
                  <xs:element ref="Value"/>
                  <xs:element ref="DatabaseConnectionValue"/>
                </xs:choice>
              </xs:group>
              <xs:sequence>
                <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
                  <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
                    <xs:choice>
                      <xs:element ref="MapValue"/>
                      <xs:element ref="StructuredValue"/>
                      <xs:element ref="ListValue"/>
                      <xs:element ref="Value"/>
                      <xs:element ref="DatabaseConnectionValue"/>
                    </xs:choice>
                  </xs:group>
                </xs:element>
              </xs:sequence>
            </xs:element>
          </xs:sequence>
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

Attribute、Attribute、ListValue、ListValue、ListValue 和 Parameter

子元素

MapEntry

MapEntry 元素: 键属性映射中的条目。每个条目由一个键和一个关联的值组成。

XML 表示法

```
<xs:element name="MapEntry" type="MAP-ENTRY" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="KeyValue" type="KEY-VALUE">
    </xs:element>
    <xs:element name="StructuredValue" type="STRUCTURED-VALUE">
      <xs:sequence>
        <xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
          <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
```

```
            <xs:choice>
              <xs:element ref="MapValue"/>
              <xs:element ref="StructuredValue"/>
              <xs:element ref="ListValue"/>
              <xs:element ref="Value"/>
              <xs:element ref="DatabaseConnectionValue"/>
            </xs:choice>
          </xs:group>
          <xs:sequence>
            <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
              <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
                <xs:choice>
                  <xs:element ref="MapValue"/>
                  <xs:element ref="StructuredValue"/>
                  <xs:element ref="ListValue"/>
                  <xs:element ref="Value"/>
                  <xs:element ref="DatabaseConnectionValue"/>
                </xs:choice>
              </xs:group>
            </xs:element>
          </xs:sequence>
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

MapValue

子元素

KeyValue 和 StructuredValue

KeyValue 元素: 映射条目中的键值。

表 125. KeyValue 的属性

属性	使用	描述	有效值
value	必需		*string*

XML 表示法

```
<xs:element name="KeyValue" type="KEY-VALUE">
  <xs:attribute name="value" type="xs:string" use="required"/>
</xs:element>
```

父元素

MapEntry

StructuredValue 元素: 指定值（“属性”）的序列。

XML 表示法

```
<xs:element name="StructuredValue" type="STRUCTURED-VALUE">
  <xs:sequence>
    <xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
      <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
        <xs:choice>
          <xs:element ref="MapValue"/>
          <xs:element ref="StructuredValue"/>
          <xs:element ref="ListValue"/>
          <xs:element ref="Value"/>
          <xs:element ref="DatabaseConnectionValue"/>
        </xs:choice>
      </xs:group>
      <xs:sequence>
        <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
          <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
            <xs:choice>
```

```
            <xs:element ref="MapValue"/>
            <xs:element ref="StructuredValue"/>
            <xs:element ref="ListValue"/>
            <xs:element ref="Value"/>
            <xs:element ref="DatabaseConnectionValue"/>
          </xs:choice>
        </xs:group>
      </xs:element>
    </xs:sequence>
  </xs:element>
 </xs:sequence>
</xs:element>
```

父元素

MapEntry

子元素

Attribute

Attribute 元素:

表 126. Attribute 的属性

属性	使用	描述	有效值
name	必需		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
  <xs:sequence>
    <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
      <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
        <xs:choice>
          <xs:element ref="MapValue"/>
          <xs:element ref="StructuredValue"/>
          <xs:element ref="ListValue"/>
          <xs:element ref="Value"/>
          <xs:element ref="DatabaseConnectionValue"/>
        </xs:choice>
      </xs:group>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:string"/>
</xs:element>
```

父元素

StructuredValue

子元素

DatabaseConnectionValue、ListValue、ListValue、MapValue、StructuredValue 和 Value

ListValue 元素: 值序列。所有值的内容类型必须相同，但是不会对此进行检查。

XML 表示法

```
<xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
</xs:element>
```

父元素

Attribute

子元素

DatabaseConnectionValue、ListValue、MapValue、StructuredValue 和 Value

Menu 元素

定义菜单。这可以是菜单栏中新的顶级菜单，也可以使现有菜单的子菜单。

表 127. Menu 的属性

属性	使用	描述	有效值
id	**必需**		*string*
label	**必需**		*string*
labelKey	可选		*string*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
offset	可选		*nonNegativeInteger*
separatorAfter	可选		*布尔值*
separatorBefore	可选		*布尔值*
showIcon	可选		*布尔值*
showLabel	可选		*布尔值*

表 127. Menu 的属性 (续)

属性	使用	描述	有效值
systemMenu	必需		**file** **edit** **insert** **view** **tools** **window** **help** **generate** **file.project** **file.outputs** **file.models** **edit.stream** **edit.node** **edit.outputs** **edit.models** **edit.project** **tools.repository** **tools.options** **tools.streamProperties**

XML 表示法

```
<xs:element name="Menu">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="systemMenu" type="STANDARD-MENU" use="required">
    <xs:enumeration value="file"/>
    <xs:enumeration value="edit"/>
    <xs:enumeration value="insert"/>
    <xs:enumeration value="view"/>
    <xs:enumeration value="tools"/>
    <xs:enumeration value="window"/>
    <xs:enumeration value="help"/>
    <xs:enumeration value="generate"/>
    <xs:enumeration value="file.project"/>
    <xs:enumeration value="file.outputs"/>
    <xs:enumeration value="file.models"/>
    <xs:enumeration value="edit.stream"/>
    <xs:enumeration value="edit.node"/>
    <xs:enumeration value="edit.outputs"/>
    <xs:enumeration value="edit.models"/>
    <xs:enumeration value="edit.project"/>
    <xs:enumeration value="tools.repository"/>
    <xs:enumeration value="tools.options"/>
    <xs:enumeration value="tools.streamProperties"/>
  </xs:attribute>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="showIcon" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="separatorBefore" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="separatorAfter" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="offset" type="xs:nonNegativeInteger" use="optional" default="0"/>
</xs:element>
```

父元素

Controls

MenuItem 元素

定义可添加到菜单的项。

表 128. MenuItem 的属性

属性	使用	描述	有效值
action	**必需**		*string*
customMenu	可选		*string*
offset	可选		*nonNegativeInteger*
separatorAfter	可选		*布尔值*
separatorBefore	可选		*布尔值*
showIcon	可选		*布尔值*
showLabel	可选		*布尔值*
systemMenu	可选		**file** **edit** **insert** **view** **tools** **window** **help** **generate** **file.project** **file.outputs** **file.models** **edit.stream** **edit.node** **edit.outputs** **edit.models** **edit.project** **tools.repository** **tools.options** **tools.streamProperties**

XML 表示法

```
<xs:element name="MenuItem">
  <xs:attribute name="action" type="xs:string" use="required"/>
  <xs:attribute name="systemMenu" type="STANDARD-MENU" use="optional">
    <xs:enumeration value="file"/>
    <xs:enumeration value="edit"/>
    <xs:enumeration value="insert"/>
    <xs:enumeration value="view"/>
    <xs:enumeration value="tools"/>
    <xs:enumeration value="window"/>
    <xs:enumeration value="help"/>
    <xs:enumeration value="generate"/>
    <xs:enumeration value="file.project"/>
    <xs:enumeration value="file.outputs"/>
    <xs:enumeration value="file.models"/>
    <xs:enumeration value="edit.stream"/>
    <xs:enumeration value="edit.node"/>
    <xs:enumeration value="edit.outputs"/>
    <xs:enumeration value="edit.models"/>
    <xs:enumeration value="edit.project"/>
    <xs:enumeration value="tools.repository"/>
    <xs:enumeration value="tools.options"/>
    <xs:enumeration value="tools.streamProperties"/>
  </xs:attribute>
  <xs:attribute name="customMenu" type="xs:string" use="optional"/>
```

```
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="showIcon" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="separatorBefore" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="separatorAfter" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="offset" type="xs:nonNegativeInteger" use="optional" default="0"/>
</xs:element>
```

父元素

Controls

MissingValues 元素

表 129. MissingValues 的属性

属性	使用	描述	有效值
treatNullAsMissing	可选		*布尔值*
treatWhitespaceAsMissing	可选		*布尔值*

XML 表示法

```
<xs:element name="MissingValues" type="MISSING-VALUES" minOccurs="0">
  <xs:sequence>
    <xs:element name="Range" type="RANGE">
    </xs:element>
    <xs:element name="Values" type="FIELD-VALUES">
      <xs:sequence>
        <xs:element name="Value" type="FIELD-VALUE" minOccurs="0" maxOccurs="unbounded">
          <xs:sequence>
            <xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
            </xs:element>
          </xs:sequence>
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="treatNullAsMissing" type="xs:boolean" default="true"/>
  <xs:attribute name="treatWhitespaceAsMissing" type="xs:boolean" default="false"/>
</xs:element>
```

父元素

Field

子元素

Range 和 Values

Range 元素:

表 130. Range 的属性

属性	使用	描述	有效值
maxValue	必需		*string*
minValue	必需		*string*

XML 表示法

```
<xs:element name="Range" type="RANGE">
  <xs:attribute name="minValue" type="xs:string" use="required"/>
  <xs:attribute name="maxValue" type="xs:string" use="required"/>
</xs:element>
```

父元素

MissingValues

Values 元素:

表 131. Values 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="Values" type="FIELD-VALUES">
  <xs:sequence>
    <xs:element name="Value" type="FIELD-VALUE" minOccurs="0" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
        </xs:element>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

父元素

MissingValues

子元素

值

Value 元素:

表 132. Value 的属性

属性	使用	描述	有效值
code	**必需**		*整数*
displayLabel	可选		*string*
flagValue	可选		*布尔值*
value	**必需**		*string*

XML 表示法

```
<xs:element name="Value" type="FIELD-VALUE" minOccurs="0" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="code" type="xs:integer" use="required"/>
  <xs:attribute name="flagValue" type="xs:boolean"/>
  <xs:attribute name="displayLabel" type="xs:string"/>
</xs:element>
```

父元素

值

子元素

DisplayLabel

DisplayLabel 元素: 指定语言中的字段或值的显示标签。可以在特定语言没有标签的地方使用 displayLabel 属性。

表 133. DisplayLabel 的属性

属性	使用	描述	有效值
lang	可选		*NMTOKEN*
value	**必需**		*string*

XML 表示法

```
<xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="lang" type="xs:NMTOKEN" default="en"/>
</xs:element>
```

父元素

值

ModelBuilder 元素

表 134. ModelBuilder 的属性

属性	使用	描述	有效值
allowNoInputs	可选		*布尔值*
allowNoOutputs	可选		*布尔值*
miningFunctions	**必需**		*任意*
nullifyBlanks	可选		*布尔值*

XML 表示法

```
<xs:element name="ModelBuilder">
  <xs:sequence>
    <xs:element name="Algorithm">
    </xs:element>
    <xs:element name="ModelingFields" minOccurs="0">
      <xs:sequence>
        <xs:element name="InputFields" minOccurs="0">
        </xs:element>
        <xs:element name="OutputFields" minOccurs="0">
        </xs:element>
      </xs:sequence>
    </xs:element>
    <xs:element name="ModelGeneration">
    </xs:element>
    <xs:element name="ModelFields">
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
        <xs:group ref="DATA-MODEL-EXPRESSION">
          <xs:choice>
            <xs:element ref="ForEach"/>
            <xs:element ref="AddField"/>
            <xs:element ref="ChangeField"/>
            <xs:element ref="RemoveField"/>
          </xs:choice>
        </xs:group>
      </xs:sequence>
    </xs:element>
    <xs:element name="ModelEvaluation" minOccurs="0">
      <xs:sequence>
        <xs:element name="RawPropensity" minOccurs="0">
        </xs:element>
```

```
        <xs:element name="AdjustedPropensity" minOccurs="0">
        </xs:element>
        <xs:element name="VariableImportance" minOccurs="0">
        </xs:element>
      </xs:sequence>
    </xs:element>
    <xs:element name="AutoModeling" minOccurs="0">
      <xs:sequence>
        <xs:element name="SimpleSettings">
          <xs:sequence>
            <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:element>
        <xs:element name="ExpertSettings" minOccurs="0">
          <xs:sequence>
            <xs:element ref="Condition" minOccurs="0"/>
            <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:element>
        <xs:element name="PropertyMap" minOccurs="0">
          <xs:sequence>
            <xs:element name="PropertyMapping" maxOccurs="unbounded">
            </xs:element>
          </xs:sequence>
        </xs:element>
        <xs:element ref="Constraint" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="miningFunctions" use="required"/>
  <xs:attribute name="allowNoInputs" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="allowNoOutputs" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="nullifyBlanks" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

Node

子元素

Algorithm、AutoModeling、ModelEvaluation、ModelFields、ModelGeneration 和 ModelingFields

Algorithm 元素:

表 135. Algorithm 的属性

属性	使用	描述	有效值
label	**必需**		*string*
labelKey	可选		*string*
value	**必需**		*string*

XML 表示法

```
<xs:element name="Algorithm">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

ModelBuilder

ModelingFields 元素:

表 136. ModelingFields 的属性

属性	使用	描述	有效值
controlsId	**必需**		*string*
ignoreBOTH	可选		*布尔值*

XML 表示法

```
<xs:element name="ModelingFields" minOccurs="0">
  <xs:attribute name="controlsId" type="xs:string" use="required"/>
  <xs:sequence>
    <xs:element name="InputFields" minOccurs="0">
    </xs:element>
    <xs:element name="OutputFields" minOccurs="0">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="ignoreBOTH" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

ModelBuilder

子元素

InputFields 和 OutputFields

InputFields 元素:

表 137. InputFields 的属性

属性	使用	描述	有效值
label	**必需**		*string*
labelKey	可选		*string*
多个	**必需**		*布尔值*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="InputFields" minOccurs="0">
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="multiple" type="xs:boolean" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

ModelingFields

OutputFields 元素:

表 138. OutputFields 的属性

属性	使用	描述	有效值
label	**必需**		*string*
labelKey	可选		*string*
多个	**必需**		*布尔值*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="OutputFields" minOccurs="0">
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="multiple" type="xs:boolean" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

ModelingFields

ModelGeneration 元素:

表 139. ModelGeneration 的属性

属性	使用	描述	有效值
controlsId	**必需**		*string*

XML 表示法

```
<xs:element name="ModelGeneration">
  <xs:attribute name="controlsId" type="xs:string" use="required"/>
</xs:element>
```

父元素

ModelBuilder

ModelFields 元素:

XML 表示法

```
<xs:element name="ModelFields">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:group ref="DATA-MODEL-EXPRESSION">
      <xs:choice>
        <xs:element ref="ForEach"/>
        <xs:element ref="AddField"/>
        <xs:element ref="ChangeField"/>
        <xs:element ref="RemoveField"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

ModelBuilder

子元素

AddField、ChangeField、ForEach 和 RemoveField

ModelEvaluation 元素:

表 140. ModelEvaluation 的属性

属性	使用	描述	有效值
container	必需		*任意*
controlsId	必需		*string*
outputContainer	可选		*任意*

XML 表示法

```
<xs:element name="ModelEvaluation" minOccurs="0">
  <xs:attribute name="controlsId" type="xs:string" use="required"/>
  <xs:sequence>
    <xs:element name="RawPropensity" minOccurs="0">
    </xs:element>
    <xs:element name="AdjustedPropensity" minOccurs="0">
    </xs:element>
    <xs:element name="VariableImportance" minOccurs="0">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="container" use="required"/>
  <xs:attribute name="outputContainer" use="optional"/>
</xs:element>
```

父元素

ModelBuilder

子元素

AdjustedPropensity、RawPropensity 和 VariableImportance

RawPropensity 元素:

表 141. RawPropensity 的属性

属性	使用	描述	有效值
availabilityProperty	可选		*string*
defaultValue	可选		*布尔值*
property	可选		*string*

XML 表示法

```
<xs:element name="RawPropensity" minOccurs="0">
  <xs:attribute name="property" type="xs:string" use="optional"/>
  <xs:attribute name="availabilityProperty" type="xs:string" use="optional"/>
  <xs:attribute name="defaultValue" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

ModelEvaluation

AdjustedPropensity 元素:

表 142. AdjustedPropensity 的属性

属性	使用	描述	有效值
availabilityProperty	可选		*string*
defaultValue	可选		*布尔值*
property	可选		*string*

XML 表示法

```
<xs:element name="AdjustedPropensity" minOccurs="0">
  <xs:attribute name="property" type="xs:string" use="optional"/>
  <xs:attribute name="availabilityProperty" type="xs:string" use="optional"/>
  <xs:attribute name="defaultValue" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

ModelEvaluation

VariableImportance 元素:

表 143. VariableImportance 的属性

属性	使用	描述	有效值
availabilityProperty	可选		*string*
defaultValue	可选		*布尔值*
property	可选		*string*

XML 表示法

```
<xs:element name="VariableImportance" minOccurs="0">
  <xs:attribute name="property" type="xs:string" use="optional"/>
  <xs:attribute name="availabilityProperty" type="xs:string" use="optional"/>
  <xs:attribute name="defaultValue" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

ModelEvaluation

***AutoModeling* 元素:**

表 144. AutoModeling 的属性

属性	使用	描述	有效值
enabledByDefault	可选		*任意*

XML 表示法

```
<xs:element name="AutoModeling" minOccurs="0">
  <xs:sequence>
    <xs:element name="SimpleSettings">
      <xs:sequence>
        <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:element>
    <xs:element name="ExpertSettings" minOccurs="0">
      <xs:sequence>
        <xs:element ref="Condition" minOccurs="0"/>
        <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:element>
    <xs:element name="PropertyMap" minOccurs="0">
      <xs:sequence>
        <xs:element name="PropertyMapping" maxOccurs="unbounded">
        </xs:element>
      </xs:sequence>
    </xs:element>
    <xs:element ref="Constraint" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="enabledByDefault" use="optional"/>
</xs:element>
```

父元素

ModelBuilder

子元素

Constraint、ExpertSettings、PropertyMap 和 SimpleSettings

SimpleSettings 元素:

XML 表示法

```
<xs:element name="SimpleSettings">
  <xs:sequence>
    <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

AutoModeling

子元素

PropertyGroup

ExpertSettings 元素:

XML 表示法

```
<xs:element name="ExpertSettings" minOccurs="0">
  <xs:sequence>
    <xs:element ref="Condition" minOccurs="0"/>
    <xs:element ref="PropertyGroup" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

AutoModeling

子元素

Condition 和 PropertyGroup

PropertyMap 元素:

XML 表示法

```
<xs:element name="PropertyMap" minOccurs="0">
  <xs:sequence>
    <xs:element name="PropertyMapping" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

AutoModeling

子元素

PropertyMapping

PropertyMapping 元素:

表 145. PropertyMapping 的属性

属性	使用	描述	有效值
property	**必需**		*string*
systemProperty	**必需**		*string*

XML 表示法

```
<xs:element name="PropertyMapping" maxOccurs="unbounded">
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="systemProperty" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertyMap

ModelOutput 元素

表 146. ModelOutput 的属性

属性	使用	描述	有效值
delegate	可选		*string*
deprecatedScriptNames	可选		*string*
helpLink	可选		*string*
id	**必需**		*string*
label	可选		*string*
labelKey	可选		*string*
scriptName	可选		*string*

XML 表示法

```
<xs:element name="ModelOutput">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="Properties"/>
      <xs:element name="Containers" minOccurs="0">
        <xs:sequence maxOccurs="unbounded">
          <xs:element ref="Container"/>
        </xs:sequence>
      </xs:element>
      <xs:element ref="UserInterface"/>
      <xs:element ref="Constructors" minOccurs="0"/>
      <xs:element ref="ModelProvider" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="delegate" type="xs:string" use="optional"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="helpLink" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Constructors、Containers、ModelProvider、Properties 和 UserInterface

相关元素

DocumentOutput、InteractiveDocumentBuilder、InteractiveModelBuilder 和 Node

Containers 元素:

XML 表示法

```
<xs:element name="Containers" minOccurs="0">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Container"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Container

ModelProvider 元素

表 147. ModelProvider 的属性

属性	使用	描述	有效值
container	必需		*string*
isPMML	可选		*布尔值*

XML 表示法

```
<xs:element name="ModelProvider">
  <xs:attribute name="container" type="xs:string" use="required"/>
  <xs:attribute name="isPMML" type="xs:boolean" use="optional" default="true"/>
</xs:element>
```

父元素

DocumentOutput、InteractiveDocumentBuilder、InteractiveModelBuilder、ModelOutput 和 Node

ModelType 元素

定义新的模型类型

表 148. ModelType 的属性

属性	使用	描述	有效值
format	必需		**utf8** **binary**
id	必需		*string*
inputDirection	可选		*string*
outputDirection	可选		*string*

XML 表示法

```
<xs:element name="ModelType">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="format" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="utf8"/>
        <xs:enumeration value="binary"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="inputDirection" type="xs:string" use="optional" default="[in]"/>
  <xs:attribute name="outputDirection" type="xs:string" use="optional" default="[out]"/>
</xs:element>
```

父元素

ContainerTypes

相关元素

DocumentType

ModelViewerPanel 元素

表 149. ModelViewerPanel 的属性

属性	使用	描述	有效值
container	必需		*string*

XML 表示法

```
<xs:element name="ModelViewerPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
```

```
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="container" type="xs:string" use="required"/>
</xs:element>
```

父元素

Tab

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、SelectorPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

Module 元素

表 150. Module 的属性

属性	使用	描述	有效值
libraryId	可选		*string*
name	可选		*string*
systemModule	可选		**SmartScore**

XML 表示法

```
<xs:element name="Module">
  <xs:sequence>
    <xs:element ref="InputFiles"/>
    <xs:element ref="OutputFiles"/>
    <xs:element ref="StatusCodes" minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="systemModule" use="optional" default="SmartScore">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="SmartScore"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="libraryId" type="xs:string" use="optional"/>
  <xs:attribute name="name" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Execution

子元素

InputFiles、OutputFiles 和 StatusCodes

MultiFieldAllocationControl 元素

定义可用于从分配程序属性所确定的字段分配列表控件中选择字段的控件。

表 151. MultiFieldAllocationControl 的属性

属性	使用	描述	有效值
allocator	**必需**		*string*
buttonColumn	**必需**		*整数*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
showLabel	可选		*布尔值*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="MultiFieldAllocationControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="allocator" type="xs:string" use="required"/>
  <xs:attribute name="buttonColumn" type="xs:integer" use="required"/>
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

MultiFieldChooserControl 元素

定义可用于从当前数据模型选择字段的控件。

表 152. MultiFieldChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
showLabel	可选		*布尔值*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="MultiFieldChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
```

```
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

MultiItemChooserControl 元素

定义可用于从选择中选择多个值的控件。

表 153. MultiItemChooserControl 的属性

属性	使用	描述	有效值
catalog	**必需**		*string*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="MultiItemChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
```

```
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="catalog" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

SingleItemChooserControl

Node 元素

表 154. Node 的属性

属性	使用	描述	有效值
delegate	可选		*string*
deprecatedScriptNames	可选		*string*
描述	可选		*string*
descriptionKey	可选		*string*
helpLink	可选		*string*
id	**必需**		*string*
label	**必需**		*string*
labelKey	可选		*string*
palette	可选		**import** **fieldOp** **recordOp** **建模** **dbModeling** **graph** **输出** **导出** **modeling.classification** **modeling.association** **modeling.segmentation** **modeling.auto**
relativePosition	可选		*string*
relativeTo	可选		*string*
scriptName	可选		*string*

表 154. Node 的属性 (续)

属性	使用	描述	有效值
类型	**必需**		**dataReader** **dataWriter** **dataTransformer** **modelApplier** **modelBuilder** **documentBuilder**

XML 表示法

```
<xs:element name="Node">
  <xs:sequence maxOccurs="unbounded">
    <xs:choice maxOccurs="unbounded">
      <xs:element ref="Properties"/>
      <xs:element name="Containers" minOccurs="0">
        <xs:sequence maxOccurs="unbounded">
          <xs:element ref="Container"/>
        </xs:sequence>
      </xs:element>
      <xs:element ref="UserInterface"/>
      <xs:element ref="Constructors" minOccurs="0"/>
      <xs:element ref="ModelProvider" minOccurs="0"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="delegate" type="xs:string" use="optional"/>
  <xs:sequence>
    <xs:element ref="ModelBuilder" minOccurs="0"/>
    <xs:element ref="DocumentBuilder" minOccurs="0"/>
    <xs:element ref="Execution"/>
    <xs:element ref="OutputDataModel" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="type" type="NODE-TYPE" use="required">
    <xs:enumeration value="dataReader"/>
    <xs:enumeration value="dataWriter"/>
    <xs:enumeration value="dataTransformer"/>
    <xs:enumeration value="modelApplier"/>
    <xs:enumeration value="modelBuilder"/>
    <xs:enumeration value="documentBuilder"/>
  </xs:attribute>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="palette" type="SYSTEM-PALETTE" use="optional">
    <xs:enumeration value="import"/>
    <xs:enumeration value="fieldOp"/>
    <xs:enumeration value="recordOp"/>
    <xs:enumeration value="modeling"/>
    <xs:enumeration value="dbModeling"/>
    <xs:enumeration value="graph"/>
    <xs:enumeration value="output"/>
    <xs:enumeration value="export"/>
    <xs:enumeration value="modeling.classification"/>
    <xs:enumeration value="modeling.association"/>
    <xs:enumeration value="modeling.segmentation"/>
    <xs:enumeration value="modeling.auto"/>
  </xs:attribute>
  <xs:attribute name="helpLink" type="xs:string" use="optional"/>
  <xs:attribute name="relativeTo" type="xs:string" use="optional"/>
  <xs:attribute name="relativePosition" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Extension

子元素

Constructors、Containers、DocumentBuilder、Execution、ModelBuilder、ModelProvider、OutputDataModel、Properties 和 UserInterface

相关元素

DocumentOutput、InteractiveDocumentBuilder、InteractiveModelBuilder 和 ModelOutput

Containers 元素:

XML 表示法

```
<xs:element name="Containers" minOccurs="0">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Container"/>
  </xs:sequence>
</xs:element>
```

父元素

Node

子元素

Container

Not 元素

XML 表示法

```
<xs:element name="Not">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

And、Command、Constraint、CreateDocument、CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModel、CreateModelApplier、CreateModelOutput、Enabled、Not、Option、Or、Run 和 Visible

子元素

And、Condition、Not 和 Or

NumberFormat 元素

为数字字段定义格式信息。

表 155. NumberFormat 的属性

属性	使用	描述	有效值
decimalPlaces	必需		*nonNegativeInteger*

表 155. NumberFormat 的属性 (续)

属性	使用	描述	有效值
decimalSymbol	必需		句号 逗号
formatType	必需		**standard** 科学表示法 货币
groupingSymbol	必需		**none** 句号 逗号 **space**
name	必需		*string*

XML 表示法

```
<xs:element name="NumberFormat" type="NUMBER-FORMAT-DECLARATION">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="formatType" type="NUMBER-FORMAT-TYPE" use="required">
    <xs:enumeration value="standard"/>
    <xs:enumeration value="scientific"/>
    <xs:enumeration value="currency"/>
  </xs:attribute>
  <xs:attribute name="decimalPlaces" type="xs:nonNegativeInteger" use="required"/>
  <xs:attribute name="decimalSymbol" type="DECIMAL-SYMBOL" use="required">
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
  </xs:attribute>
  <xs:attribute name="groupingSymbol" type="NUMBER-GROUPING-SYMBOL" use="required">
    <xs:enumeration value="none"/>
    <xs:enumeration value="period"/>
    <xs:enumeration value="comma"/>
    <xs:enumeration value="space"/>
  </xs:attribute>
</xs:element>
```

NumericInfo 元素

表 156. NumericInfo 的属性

属性	使用	描述	有效值
mean	可选		*double*
standardDeviation	可选		*double*

XML 表示法

```
<xs:element name="NumericInfo">
  <xs:attribute name="mean" type="xs:double"/>
  <xs:attribute name="standardDeviation" type="xs:double"/>
</xs:element>
```

父元素

AddField、ChangeField 和 Field

Option 元素

表 157. Option 的属性

属性	使用	描述	有效值
ifProperty	可选		*string*
unlessProperty	可选		*string*
value	**必需**		

XML 表示法

```
<xs:element name="Option">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
  <xs:attribute name="value" type="EVALUATED-STRING" use="required"/>
  <xs:attribute name="ifProperty" type="xs:string" use="optional"/>
  <xs:attribute name="unlessProperty" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Run

子元素

And、Condition、Not 和 Or

OptionCode 元素

表 158. OptionCode 的属性

属性	使用	描述	有效值
code	可选		*long*
描述	可选		*string*
类型	可选		**mandatory** **optional**

XML 表示法

```
<xs:element name="OptionCode">
  <xs:attribute name="code" type="xs:long"/>
  <xs:attribute name="type" type="LicenseType">
    <xs:enumeration value="mandatory"/>
    <xs:enumeration value="optional"/>
  </xs:attribute>
  <xs:attribute name="description" type="xs:string"/>
</xs:element>
```

父元素

License

Or 元素

XML 表示法

```
<xs:element name="Or">
  <xs:sequence minOccurs="2" maxOccurs="unbounded">
    <xs:group ref="CONDITION-EXPRESSION">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

And、Command、Constraint、CreateDocument、CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModel、CreateModelApplier、CreateModelOutput、Enabled、Not、Option、Or、Run 和 Visible

子元素

And、Condition、Not 和 Or

OutputDataModel 元素

表 159. OutputDataModel 的属性

属性	使用	描述	有效值
libraryId	可选		*string*
method	可选		**xml** **delegate** **dataModelProvider** **sharedLibrary**
mode	可选		**fixed** **modify** **extend** **replace**
providerClass	可选		*string*

XML 表示法

```
<xs:element name="OutputDataModel">
  <xs:attribute name="mode" use="optional" default="fixed">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="fixed"/>
        <xs:enumeration value="modify"/>
        <xs:enumeration value="extend"/>
        <xs:enumeration value="replace"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="method" use="optional" default="xml">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="xml"/>
        <xs:enumeration value="delegate"/>
        <xs:enumeration value="dataModelProvider"/>
```

```
        <xs:enumeration value="sharedLibrary"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="providerClass" type="xs:string" use="optional"/>
  <xs:attribute name="libraryId" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Node

OutputFiles 元素

XML 表示法

```
<xs:element name="OutputFiles">
  <xs:group ref="RUNTIME-FILES">
    <xs:sequence>
      <xs:element ref="DataFile"/>
      <xs:element ref="ContainerFile" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:group>
</xs:element>
```

父元素

Execution 和 Module

子元素

ContainerFile 和 DataFile

Palette 元素

表 160. Palette 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
id	**必需**		*string*
label	**必需**		*string*
labelKey	可选		*string*
position	可选		**atStart** **atEnd** **before** **after**

表 160. Palette 的属性 (续)

属性	使用	描述	有效值
systemPalette	可选		**import** **fieldOp** **recordOp** 建模 **dbModeling** **graph** 输出 导出 **modeling.classification** **modeling.association** **modeling.segmentation** **modeling.auto**

XML 表示法

```
<xs:element name="Palette">
  <xs:sequence>
    <xs:element ref="Icon"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="position" use="optional">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="atStart"/>
        <xs:enumeration value="atEnd"/>
        <xs:enumeration value="before"/>
        <xs:enumeration value="after"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="systemPalette" type="SYSTEM-PALETTE" use="optional">
    <xs:enumeration value="import"/>
    <xs:enumeration value="fieldOp"/>
    <xs:enumeration value="recordOp"/>
    <xs:enumeration value="modeling"/>
    <xs:enumeration value="dbModeling"/>
    <xs:enumeration value="graph"/>
    <xs:enumeration value="output"/>
    <xs:enumeration value="export"/>
    <xs:enumeration value="modeling.classification"/>
    <xs:enumeration value="modeling.association"/>
    <xs:enumeration value="modeling.segmentation"/>
    <xs:enumeration value="modeling.auto"/>
  </xs:attribute>
</xs:element>
```

子元素

图标

Parameters 元素

扩展节点中的配置参数。

表 161. Parameters 的属性

属性	使用	描述	有效值
count	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="Parameters" type="PARAMETERS">
  <xs:sequence>
    <xs:element name="Parameter" type="PARAMETER" minOccurs="0" maxOccurs="unbounded">
      <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
        <xs:choice>
          <xs:element ref="MapValue"/>
          <xs:element ref="StructuredValue"/>
          <xs:element ref="ListValue"/>
          <xs:element ref="Value"/>
          <xs:element ref="DatabaseConnectionValue"/>
        </xs:choice>
      </xs:group>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="count" type="xs:nonNegativeInteger"/>
</xs:element>
```

子元素

Parameter

Parameter 元素: 参数具有名称和值。可以使用 value 属性来表示简单值; 复合值使用 ParameterContent 描述的内容模型。会对嵌套值重复此属性和内容组合。

表 162. Parameter 的属性

属性	使用	描述	有效值
name	**必需**		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="Parameter" type="PARAMETER" minOccurs="0" maxOccurs="unbounded">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:string"/>
</xs:element>
```

父元素

Parameters

子元素

DatabaseConnectionValue、ListValue、MapValue、StructuredValue 和 Value

PasswordBoxControl 元素

定义可用于修改不应显示值的字符串值的单行密码控件。

表 163. PasswordBoxControl 的属性

属性	使用	描述	有效值
列	可选		*positiveInteger*
描述	可选		*string*
descriptionKey	可选		*string*

表 163. PasswordBoxControl 的属性 (续)

属性	使用	描述	有效值
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="PasswordBoxControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="columns" type="xs:positiveInteger" use="optional" default="20"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

Properties 元素

XML 表示法

```
<xs:element name="Properties">
  <xs:sequence>
    <xs:element ref="Property" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

DocumentOutput、Execution、InteractiveDocumentBuilder、InteractiveModelBuilder、ModelOutput 和 Node

子元素

属性

PropertiesPanel 元素

定义顶级属性面板。

表 164. PropertiesPanel 的属性

属性	使用	描述	有效值
id	可选		*string*
label	可选		*string*
labelKey	可选		*string*

XML 表示法

```
<xs:element name="PropertiesPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:sequence maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="CheckBoxControl"/>
      <xs:element ref="TextBoxControl"/>
      <xs:element ref="PasswordBoxControl"/>
      <xs:element ref="TextAreaControl"/>
      <xs:element ref="RadioButtonGroupControl"/>
      <xs:element ref="CheckBoxGroupControl"/>
      <xs:element ref="ComboBoxControl"/>
      <xs:element ref="SpinnerControl"/>
      <xs:element ref="ServerFileChooserControl"/>
      <xs:element ref="ServerDirectoryChooserControl"/>
      <xs:element ref="ClientFileChooserControl"/>
      <xs:element ref="ClientDirectoryChooserControl"/>
      <xs:element ref="TableControl"/>
      <xs:element ref="SingleFieldChooserControl"/>
      <xs:element ref="SingleFieldAllocationControl"/>
      <xs:element ref="MultiFieldChooserControl"/>
      <xs:element ref="MultiFieldAllocationControl"/>
      <xs:element ref="SingleFieldValueChooserControl"/>
      <xs:element ref="SingleItemChooserControl"/>
      <xs:element ref="MultiItemChooserControl"/>
      <xs:element ref="DBConnectionChooserControl"/>
      <xs:element ref="DBTableChooserControl"/>
      <xs:element ref="PropertyControl"/>
      <xs:element ref="StaticText"/>
      <xs:element ref="SystemControls"/>
      <xs:element ref="ActionButton"/>
      <xs:element ref="FieldAllocationList"/>
      <xs:element ref="PropertiesPanel"/>
      <xs:element ref="PropertiesSubPanel"/>
      <xs:element ref="SelectorPanel"/>
      <xs:element ref="ExtensionObjectPanel"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="optional"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel、PropertiesSubPanel 和 Tab

子元素

ActionButton、CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、ComboBoxControl、DBConnectionChooserControl、DBTableChooserControl、Enabled、ExtensionObjectPanel、FieldAllocationList、Layout、MultiFieldAllocationControl、MultiFieldChooserControl、MultiItemChooserControl、PasswordBoxControl、PropertiesPanel、PropertiesSubPanel、PropertyControl、RadioButtonGroupControl、SelectorPanel、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SingleItemChooserControl、SpinnerControl、StaticText、SystemControls、TableControl、TextAreaControl、TextBoxControl 和 Visible

相关元素

PropertiesSubPanel

PropertiesSubPanel 元素

定义可用于将相关 UI 组件和控件分组到一起的子面板。

表 165. PropertiesSubPanel 的属性

属性	使用	描述	有效值
buttonLabel	可选		*string*
buttonLabelKey	可选		*string*
dialogTitle	可选		*string*
dialogTitleKey	可选		*string*
helpLink	可选		*string*
mnemonic	可选		*string*
mnemonicKey	可选		*string*

XML 表示法

```
<xs:element name="PropertiesSubPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:sequence maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="CheckBoxControl"/>
      <xs:element ref="TextBoxControl"/>
      <xs:element ref="PasswordBoxControl"/>
      <xs:element ref="TextAreaControl"/>
      <xs:element ref="RadioButtonGroupControl"/>
      <xs:element ref="CheckBoxGroupControl"/>
      <xs:element ref="ComboBoxControl"/>
      <xs:element ref="SpinnerControl"/>
      <xs:element ref="ServerFileChooserControl"/>
      <xs:element ref="ServerDirectoryChooserControl"/>
      <xs:element ref="ClientFileChooserControl"/>
      <xs:element ref="ClientDirectoryChooserControl"/>
      <xs:element ref="TableControl"/>
      <xs:element ref="SingleFieldChooserControl"/>
      <xs:element ref="SingleFieldAllocationControl"/>
      <xs:element ref="MultiFieldChooserControl"/>
      <xs:element ref="MultiFieldAllocationControl"/>
      <xs:element ref="SingleFieldValueChooserControl"/>
      <xs:element ref="SingleItemChooserControl"/>
      <xs:element ref="MultiItemChooserControl"/>
      <xs:element ref="DBConnectionChooserControl"/>
      <xs:element ref="DBTableChooserControl"/>
```

```
      <xs:element ref="PropertyControl"/>
      <xs:element ref="StaticText"/>
      <xs:element ref="SystemControls"/>
      <xs:element ref="ActionButton"/>
      <xs:element ref="FieldAllocationList"/>
      <xs:element ref="PropertiesPanel"/>
      <xs:element ref="PropertiesSubPanel"/>
      <xs:element ref="SelectorPanel"/>
      <xs:element ref="ExtensionObjectPanel"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="buttonLabel" type="xs:string" use="optional"/>
  <xs:attribute name="buttonLabelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="dialogTitle" type="xs:string" use="optional"/>
  <xs:attribute name="dialogTitleKey" type="xs:string" use="optional"/>
  <xs:attribute name="helpLink" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

ActionButton、CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、ComboBoxControl、DBConnectionChooserControl、DBTableChooserControl、Enabled、ExtensionObjectPanel、FieldAllocationList、Layout、MultiFieldAllocationControl、MultiFieldChooserControl、MultiItemChooserControl、PasswordBoxControl、PropertiesPanel、PropertiesSubPanel、PropertyControl、RadioButtonGroupControl、SelectorPanel、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SingleItemChooserControl、SpinnerControl、StaticText、SystemControls、TableControl、TextAreaControl、TextBoxControl 和 Visible

相关元素

PropertiesPanel

Property 元素

表 166. Property 的属性

属性	使用	描述	有效值
defaultValue	可选		
defaultValueKey	可选		*string*
deprecatedScriptNames	可选		*string*
描述	可选		*string*
descriptionKey	可选		*string*
isList	可选		*布尔值*
label	可选		*string*
labelKey	可选		*string*
最大	可选		*string*
最小	可选		*string*
name	**必需**		*string*
scriptName	可选		*string*
类型	可选		*string*

表 166. Property 的属性 (续)

属性	使用	描述	有效值
valueType	可选		**string** **encryptedString** **fieldName** 整数 **double** 布尔值 **date** **enum** **structure** **databaseConnection**

XML 表示法

```
<xs:element name="Property">
  <xs:choice>
    <xs:element ref="DefaultValue" minOccurs="0"/>
  </xs:choice>
  <xs:attribute name="valueType" type="PROPERTY-VALUE-TYPE">
    <xs:enumeration value="string"/>
    <xs:enumeration value="encryptedString"/>
    <xs:enumeration value="fieldName"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="boolean"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="enum"/>
    <xs:enumeration value="structure"/>
    <xs:enumeration value="databaseConnection"/>
  </xs:attribute>
  <xs:attribute name="isList" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="min" type="xs:string" use="optional"/>
  <xs:attribute name="max" type="xs:string" use="optional"/>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="scriptName" type="xs:string" use="optional"/>
  <xs:attribute name="deprecatedScriptNames" type="xs:string" use="optional"/>
  <xs:attribute name="type" type="xs:string" use="optional"/>
  <xs:attribute name="defaultValue" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="defaultValueKey" type="xs:string" use="optional"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Properties 和 PropertySets

子元素

DefaultValue

相关元素

PropertyType

PropertyControl 元素

定义定制属性控件。controlClass 属性所确定的类必须实现 PropertyControl 界面。

表 167. PropertyControl 的属性

属性	使用	描述	有效值
controlClass	**必需**		*string*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="PropertyControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="controlClass" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

PropertyGroup 元素

表 168. PropertyGroup 的属性

属性	使用	描述	有效值
label	可选		*string*
labelKey	可选		*string*
属性	**必需**		*string*

XML 表示法

```
<xs:element name="PropertyGroup">
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="properties" type="xs:string" use="required"/>
</xs:element>
```

父元素

ExpertSettings 和 SimpleSettings

PropertySets 元素

XML 表示法

```
<xs:element name="PropertySets">
  <xs:sequence>
    <xs:element ref="Property" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

属性

PropertyType 元素

表 169. PropertyType 的属性

属性	使用	描述	有效值
id	**必需**		*string*
isKeyed	可选		*布尔值*
isList	可选		*布尔值*
最大	可选		*string*
最小	可选		*string*

表 169. PropertyType 的属性 (续)

属性	使用	描述	有效值
valueType	可选		**string** **encryptedString** **fieldName** 整数 **double** 布尔值 **date** **enum** **structure** **databaseConnection**

XML 表示法

```
<xs:element name="PropertyType">
  <xs:choice>
    <xs:element ref="DefaultValue" minOccurs="0"/>
  </xs:choice>
  <xs:attribute name="valueType" type="PROPERTY-VALUE-TYPE">
    <xs:enumeration value="string"/>
    <xs:enumeration value="encryptedString"/>
    <xs:enumeration value="fieldName"/>
    <xs:enumeration value="integer"/>
    <xs:enumeration value="double"/>
    <xs:enumeration value="boolean"/>
    <xs:enumeration value="date"/>
    <xs:enumeration value="enum"/>
    <xs:enumeration value="structure"/>
    <xs:enumeration value="databaseConnection"/>
  </xs:attribute>
  <xs:attribute name="isList" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="min" type="xs:string" use="optional"/>
  <xs:attribute name="max" type="xs:string" use="optional"/>
  <xs:choice>
    <xs:element ref="Enumeration" minOccurs="0"/>
    <xs:element ref="Structure" minOccurs="0"/>
  </xs:choice>
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="isKeyed" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

PropertyTypes

子元素

DefaultValue、Enumeration 和 Structure

相关元素

属性

PropertyTypes 元素

XML 表示法

```
<xs:element name="PropertyTypes">
  <xs:sequence>
    <xs:element ref="PropertyType" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

CommonObjects

子元素

PropertyType

RadioButtonGroupControl 元素

定义可用于指定 true/false 布尔值或枚举类型值的单选按钮控件组。

表 170. RadioButtonGroupControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
falseLabel	可选		*string*
falseLabelKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
layoutByRow	可选		*布尔值*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
行	可选		*positiveInteger*
showLabel	可选		*布尔值*
trueFirst	可选		*布尔值*
trueLabel	可选		*string*
trueLabelKey	可选		*string*
useSubPanel	可选		*布尔值*

XML 表示法

```
<xs:element name="RadioButtonGroupControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="rows" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="layoutByRow" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="useSubPanel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="falseLabel" type="xs:string" use="optional"/>
  <xs:attribute name="falseLabelKey" type="xs:string" use="optional"/>
```

```
  <xs:attribute name="trueLabel" type="xs:string" use="optional"/>
  <xs:attribute name="trueLabelKey" type="xs:string" use="optional"/>
  <xs:attribute name="trueFirst" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

Range 元素

表 171. Range 的属性

属性	使用	描述	有效值
最大	可选		*string*
最小	可选		*string*

XML 表示法

```
<xs:element name="Range">
  <xs:attribute name="min" type="xs:string"/>
  <xs:attribute name="max" type="xs:string"/>
</xs:element>
```

父元素

AddField、ChangeField、Field、Field、MissingValues、MissingValues 和 MissingValues

Range 元素

表 172. Range 的属性

属性	使用	描述	有效值
maxValue	必需		*string*
minValue	必需		*string*

XML 表示法

```
<xs:element name="Range" type="RANGE">
  <xs:attribute name="minValue" type="xs:string" use="required"/>
  <xs:attribute name="maxValue" type="xs:string" use="required"/>
</xs:element>
```

父元素

AddField、ChangeField、Field、Field、MissingValues、MissingValues 和 MissingValues

RemoveField 元素

表 173. RemoveField 的属性

属性	使用	描述	有效值
fieldRef	必需		

XML 表示法

```
<xs:element name="RemoveField">
  <xs:attribute name="fieldRef" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

ForEach 和 ModelFields

Resources 元素

定义诸如客户端库、资源束和服务器端库之类的公共资源。

XML 表示法

```
<xs:element name="Resources">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element name="Bundle" minOccurs="0">
      </xs:element>
      <xs:element name="JarFile" minOccurs="0">
      </xs:element>
      <xs:element name="SharedLibrary" minOccurs="0">
      </xs:element>
      <xs:element name="HelpInfo" minOccurs="0">
      </xs:element>
    </xs:choice>
  </xs:sequence>
</xs:element>
```

父元素

Extension

子元素

Bundle、HelpInfo、JarFile 和 SharedLibrary

Bundle 元素:

表 174. Bundle 的属性

属性	使用	描述	有效值
id	必需		*string*
path	必需		
类型	必需		**list** 属性

XML 表示法

```
<xs:element name="Bundle" minOccurs="0">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="type" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
```

```
        <xs:enumeration value="list"/>
        <xs:enumeration value="properties"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

Resources

JarFile 元素:

表 175. JarFile 的属性

属性	使用	描述	有效值
id	必需		*string*
path	必需		

XML 表示法

```
<xs:element name="JarFile" minOccurs="0">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

Resources

SharedLibrary 元素:

表 176. SharedLibrary 的属性

属性	使用	描述	有效值
id	必需		*string*
path	必需		

XML 表示法

```
<xs:element name="SharedLibrary" minOccurs="0">
  <xs:attribute name="id" type="xs:string" use="required"/>
  <xs:attribute name="path" type="EVALUATED-STRING" use="required"/>
</xs:element>
```

父元素

Resources

HelpInfo 元素:

表 177. HelpInfo 的属性

属性	使用	描述	有效值
缺省	可选		*string*
helpset	可选		
id	可选		*string*
missing	可选		*string*
path	可选		

表 177. HelpInfo 的属性 (续)

属性	使用	描述	有效值
类型	必需		**native** **javahelp** **html**

XML 表示法

```
<xs:element name="HelpInfo" minOccurs="0">
  <xs:attribute name="id" type="xs:string" use="optional"/>
  <xs:attribute name="type" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="native"/>
        <xs:enumeration value="javahelp"/>
        <xs:enumeration value="html"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="path" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="helpset" type="EVALUATED-STRING" use="optional"/>
  <xs:attribute name="default" type="xs:string" use="optional"/>
  <xs:attribute name="missing" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Resources

Run 元素

XML 表示法

```
<xs:element name="Run">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
    <xs:element ref="Command" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="Option" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="StatusCodes" minOccurs="0"/>
  </xs:sequence>
</xs:element>
```

父元素

Executable

子元素

And、Command、Condition、Not、Option、Or 和 StatusCodes

SPSSDataFormat 元素

XML 表示法

```
<xs:element name="SPSSDataFormat"/>
```

父元素

DataFormat

SelectorPanel 元素

定义可包含多个子面板但是一次只能显示其中一个子面板的 UI 组件。

表 178. SelectorPanel 的属性

属性	使用	描述	有效值
control	必需		*string*

XML 表示法

```
<xs:element name="SelectorPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Selector">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="control" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout、Selector 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、StaticText、SystemControls、TabbedPanel 和 TextBrowserPanel

Selector 元素:

表 179. Selector 的属性

属性	使用	描述	有效值
controlValue	必需		*string*
panelId	必需		*string*

XML 表示法

```
<xs:element name="Selector">
  <xs:attribute name="panelId" type="xs:string" use="required"/>
  <xs:attribute name="controlValue" type="xs:string" use="required"/>
</xs:element>
```

父元素

SelectorPanel

ServerDirectoryChooserControl 元素

定义可用于选择服务器上的目录的控件。

表 180. ServerDirectoryChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
mode	**必需**		**open** **save** **import** 导出
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ServerDirectoryChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="mode" type="FILE-CHOOSER-MODE" use="required">
    <xs:enumeration value="open"/>
    <xs:enumeration value="save"/>
    <xs:enumeration value="import"/>
    <xs:enumeration value="export"/>
  </xs:attribute>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

ServerFileChooserControl 元素

定义可用于选择服务器上的文件的控件。

表 181. ServerFileChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
mode	**必需**		**open** **save** **import** **导出**
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ServerFileChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="mode" type="FILE-CHOOSER-MODE" use="required">
    <xs:enumeration value="open"/>
    <xs:enumeration value="save"/>
    <xs:enumeration value="import"/>
    <xs:enumeration value="export"/>
  </xs:attribute>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

SetContainer 元素

表 182. SetContainer 的属性

属性	使用	描述	有效值
source	必需		*string*
target	必需		*string*

XML 表示法

```
<xs:element name="SetContainer">
  <xs:attribute name="source" type="xs:string" use="required"/>
  <xs:attribute name="target" type="xs:string" use="required"/>
</xs:element>
```

父元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

相关元素

SetProperty

SetProperty 元素

表 183. SetProperty 的属性

属性	使用	描述	有效值
source	必需		*string*
target	必需		*string*

XML 表示法

```
<xs:element name="SetProperty">
  <xs:attribute name="source" type="xs:string" use="required"/>
  <xs:attribute name="target" type="xs:string" use="required"/>
</xs:element>
```

父元素

CreateDocumentOutput、CreateInteractiveDocumentBuilder、CreateInteractiveModelBuilder、CreateModelApplier 和 CreateModelOutput

相关元素

SetContainer

SingleFieldAllocationControl 元素

定义可用于从分配程序属性所确定的字段分配列表控件中选择字段的控件。

表 184. SingleFieldAllocationControl 的属性

属性	使用	描述	有效值
allocator	**必需**		*string*
buttonColumn	**必需**		*整数*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
showLabel	可选		*布尔值*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="SingleFieldAllocationControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="allocator" type="xs:string" use="required"/>
```

```
  <xs:attribute name="buttonColumn" type="xs:integer" use="required"/>
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

SingleFieldChooserControl 元素

定义可用于从当前数据模型中选择字段的控件。

表 185. SingleFieldChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
onlyDatetime	可选		*布尔值*
onlyDiscrete	可选		*布尔值*
onlyNumeric	可选		*布尔值*
onlyRanges	可选		*布尔值*
onlySymbolic	可选		*布尔值*
property	**必需**		*string*
showLabel	可选		*布尔值*
storage	可选		*string*
types	可选		*string*

XML 表示法

```
<xs:element name="SingleFieldChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="storage" type="xs:string" use="optional"/>
  <xs:attribute name="onlyNumeric" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlySymbolic" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDatetime" type="xs:boolean" use="optional"/>
  <xs:attribute name="types" type="xs:string" use="optional"/>
  <xs:attribute name="onlyRanges" type="xs:boolean" use="optional"/>
  <xs:attribute name="onlyDiscrete" type="xs:boolean" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldValueChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

SingleFieldValueChooserControl 元素

定义可用于从 fieldControl 所确定的控件选中的字段中选择字段值的控件。

表 186. SingleFieldValueChooserControl 的属性

属性	使用	描述	有效值
描述	可选		*string*
descriptionKey	可选		*string*
fieldControl	可选		*string*
fieldDirection	可选		**in** **out** **both** **none** **partition**
label	可选		*string*
labelAbove	可选		*布尔值*

表 186. SingleFieldValueChooserControl 的属性 (续)

属性	使用	描述	有效值
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="SingleFieldValueChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="fieldControl" type="xs:string" use="optional"/>
  <xs:attribute name="fieldDirection" type="FIELD-DIRECTION" use="optional">
    <xs:enumeration value="in"/>
    <xs:enumeration value="out"/>
    <xs:enumeration value="both"/>
    <xs:enumeration value="none"/>
    <xs:enumeration value="partition"/>
  </xs:attribute>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SpinnerControl、TableControl、TextAreaControl 和 TextBoxControl

SingleItemChooserControl 元素

定义可用于从选择中选择值的控件。

表 187. SingleItemChooserControl 的属性

属性	使用	描述	有效值
catalog	**必需**		*string*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="SingleItemChooserControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="catalog" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

MultiItemChooserControl

SpinnerControl 元素

定义可用于修改数字值的微调框控件。

表 188. SpinnerControl 的属性

属性	使用	描述	有效值
列	可选		*positiveInteger*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
maxDecimalDigits	可选		*positiveInteger*
minDecimalDigits	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*
stepSize	可选		*decimal*

XML 表示法

```
<xs:element name="SpinnerControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="columns" type="xs:positiveInteger" use="optional" default="5"/>
  <xs:attribute name="stepSize" type="xs:decimal" use="optional" default="1.0"/>
  <xs:attribute name="minDecimalDigits" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="maxDecimalDigits" type="xs:positiveInteger" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl

、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、TableControl、TextAreaControl 和 TextBoxControl

StaticText 元素

定义包含静态文本部分的 UI 组件。通常用于子面板以描述面板目的。

表 189. StaticText 的属性

属性	使用	描述	有效值
文本	可选		*string*
textKey	可选		*string*

XML 表示法

```
<xs:element name="StaticText">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="text" type="xs:string" use="optional"/>
  <xs:attribute name="textKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、SystemControls、TabbedPanel 和 TextBrowserPanel

StatusCode 元素

表 190. StatusCode 的属性

属性	使用	描述	有效值
code	**必需**		*整数*
message	可选		*string*
messageKey	可选		*string*
状态	可选		**success** **warning** **error**

XML 表示法

```
<xs:element name="StatusCode">
  <xs:attribute name="code" type="xs:integer" use="required"/>
  <xs:attribute name="status" use="optional" default="success">
```

```
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="success"/>
        <xs:enumeration value="warning"/>
        <xs:enumeration value="error"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="message" type="xs:string" use="optional"/>
  <xs:attribute name="messageKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

StatusCodes

StatusCodes 元素

表 191. StatusCodes 的属性

属性	使用	描述	有效值
defaultMessage	可选		*string*
defaultMessageKey	可选		*string*

XML 表示法

```
<xs:element name="StatusCodes">
  <xs:sequence>
    <xs:element ref="StatusCode" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="defaultMessage" type="xs:string" use="optional"/>
  <xs:attribute name="defaultMessageKey" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Module 和 Run

子元素

StatusCode

StatusDetail 元素

关于进度或其他情况的补充信息。

表 192. StatusDetail 的属性

属性	使用	描述	有效值
destination	可选		客户机 **tracefile** **console**

XML 表示法

```
<xs:element name="StatusDetail" type="STATUS-DETAIL">
  <xs:sequence>
    <xs:element name="Diagnostic" type="DIAGNOSTIC" minOccurs="0" maxOccurs="unbounded">
      <xs:sequence>
        <xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
        </xs:element>
        <xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:element>
```

```
  </xs:sequence>
  <xs:attribute name="destination" type="STATUS-DESTINATION" default="client">
    <xs:enumeration value="client"/>
    <xs:enumeration value="tracefile"/>
    <xs:enumeration value="console"/>
  </xs:attribute>
</xs:element>
```

子元素

Diagnostic

Diagnostic 元素:

表 193. Diagnostic 的属性

属性	使用	描述	有效值
code	**必需**		*整数*
severity	可选		**unknown** **信息** **warning** **error** **fatal**
source	可选		*string*
subCode	可选		*整数*

XML 表示法

```
<xs:element name="Diagnostic" type="DIAGNOSTIC" minOccurs="0" maxOccurs="unbounded">
  <xs:sequence>
    <xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
    </xs:element>
    <xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="code" type="xs:integer" use="required"/>
  <xs:attribute name="subCode" type="xs:integer" default="0"/>
  <xs:attribute name="severity" type="DIAGNOSTIC-SEVERITY" default="error">
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="information"/>
    <xs:enumeration value="warning"/>
    <xs:enumeration value="error"/>
    <xs:enumeration value="fatal"/>
  </xs:attribute>
  <xs:attribute name="source" type="xs:string"/>
</xs:element>
```

父元素

StatusDetail

子元素

Message 和 Parameter

Message 元素:

表 194. Message 的属性

属性	使用	描述	有效值
lang	可选		*NMTOKEN*

XML 表示法

```
<xs:element name="Message" type="DIAGNOSTIC-MESSAGE" minOccurs="0">
  <xs:attribute name="lang" type="xs:NMTOKEN"/>
</xs:element>
```

父元素

Diagnostic

Parameter 元素:

XML 表示法

```
<xs:element name="Parameter" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
```

父元素

Diagnostic

Structure 元素

XML 表示法

```
<xs:element name="Structure">
  <xs:sequence>
    <xs:element ref="Attribute" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:element>
```

父元素

PropertyType

子元素

Attribute

StructuredValue 元素

指定值（“属性”）的序列。

XML 表示法

```
<xs:element name="StructuredValue" type="STRUCTURED-VALUE">
  <xs:sequence>
    <xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
      <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
        <xs:choice>
          <xs:element ref="MapValue"/>
          <xs:element ref="StructuredValue"/>
          <xs:element ref="ListValue"/>
          <xs:element ref="Value"/>
          <xs:element ref="DatabaseConnectionValue"/>
        </xs:choice>
      </xs:group>
      <xs:sequence>
        <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
          <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
            <xs:choice>
              <xs:element ref="MapValue"/>
              <xs:element ref="StructuredValue"/>
              <xs:element ref="ListValue"/>
              <xs:element ref="Value"/>
              <xs:element ref="DatabaseConnectionValue"/>
            </xs:choice>
          </xs:group>
        </xs:element>
```

```
      </xs:sequence>
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

Attribute、Attribute、ListValue、ListValue、ListValue 和 Parameter

子元素

Attribute

Attribute 元素:

表 195. Attribute 的属性

属性	使用	描述	有效值
name	必需		*string*
value	可选		*string*

XML 表示法

```
<xs:element name="Attribute" type="ATTRIBUTE" maxOccurs="unbounded">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
  <xs:sequence>
    <xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
      <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
        <xs:choice>
          <xs:element ref="MapValue"/>
          <xs:element ref="StructuredValue"/>
          <xs:element ref="ListValue"/>
          <xs:element ref="Value"/>
          <xs:element ref="DatabaseConnectionValue"/>
        </xs:choice>
      </xs:group>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="value" type="xs:string"/>
</xs:element>
```

父元素

StructuredValue

子元素

DatabaseConnectionValue、ListValue、ListValue、MapValue、StructuredValue 和 Value

ListValue 元素: 值序列。所有值的内容类型必须相同，但是不会对此进行检查。

XML 表示法

```
<xs:element name="ListValue" type="LIST-VALUE" minOccurs="0" maxOccurs="1">
  <xs:group ref="PARAMETER-CONTENT" minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="MapValue"/>
      <xs:element ref="StructuredValue"/>
      <xs:element ref="ListValue"/>
      <xs:element ref="Value"/>
```

```
      <xs:element ref="DatabaseConnectionValue"/>
    </xs:choice>
  </xs:group>
</xs:element>
```

父元素

Attribute

子元素

DatabaseConnectionValue、ListValue、MapValue、StructuredValue 和 Value

SystemControls 元素

表 196. SystemControls 的属性

属性	使用	描述	有效值
controlsId	**必需**		*string*

XML 表示法

```
<xs:element name="SystemControls">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="controlsId" type="xs:string" use="required"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、TabbedPanel 和 TextBrowserPanel

Tab 元素

定义选项卡式面板中的选项卡。

表 197. Tab 的属性

属性	使用	描述	有效值
helpLink	可选		*string*
id	可选		*string*
label	**必需**		*string*
labelKey	可选		*string*
mnemonic	可选		*string*
mnemonicKey	可选		*string*

XML 表示法

```
<xs:element name="Tab">
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:choice>
      <xs:element ref="PropertiesPanel"/>
      <xs:element ref="ExtensionObjectPanel"/>
      <xs:element ref="TextBrowserPanel"/>
      <xs:element ref="ModelViewerPanel"/>
      <xs:element ref="TabbedPanel"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="id" type="xs:string" use="optional"/>
  <xs:attribute name="label" type="xs:string" use="required"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="helpLink" type="xs:string" use="optional"/>
</xs:element>
```

父元素

Tabs

子元素

ExtensionObjectPanel、ModelViewerPanel、PropertiesPanel、TabbedPanel 和 TextBrowserPanel

TabbedPanel 元素

定义选项卡式面板。可向选项卡式面板中的选项卡添加其他面板。

表 198. TabbedPanel 的属性

属性	使用	描述	有效值
style	可选		**standard** **sidebar**

XML 表示法

```
<xs:element name="TabbedPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Tabs"/>
  </xs:sequence>
  <xs:attribute name="style" use="optional">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="standard"/>
        <xs:enumeration value="sidebar"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:element>
```

父元素

Tab

子元素

Enabled、Layout、Tabs 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、SystemControls 和 TextBrowserPanel

TableControl 元素

定义可用于添加、修改和除去结构列表值的表格控件。

表 199. TableControl 的属性

属性	使用	描述	有效值
columnWidths	可选		*string*
列	可选		*positiveInteger*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
行	可选		*positiveInteger*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="TableControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="rows" type="xs:positiveInteger" use="optional" default="8"/>
  <xs:attribute name="columns" type="xs:positiveInteger" use="optional" default="20"/>
  <xs:attribute name="columnWidths" type="xs:string" use="optional"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TextAreaControl 和 TextBoxControl

Tabs 元素

定义选项卡式面板中选项卡的顺序。

表 200. Tabs 的属性

属性	使用	描述	有效值
defaultTab	可选		*nonNegativeInteger*

XML 表示法

```
<xs:element name="Tabs">
  <xs:sequence>
    <xs:element ref="Tab" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="defaultTab" type="xs:nonNegativeInteger" use="optional" default="0"/>
</xs:element>
```

父元素

TabbedPanel 和 UserInterface

子元素

Tab

TextAreaControl 元素

定义可用于修改字符串值的多行文本区域。

表 201. TextAreaControl 的属性

属性	使用	描述	有效值
列	可选		*positiveInteger*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
monospaced	可选		*布尔值*
property	**必需**		*string*
行	可选		*positiveInteger*
showLabel	可选		*布尔值*

表 201. TextAreaControl 的属性 (续)

属性	使用	描述	有效值
wrapLines	可选		*布尔值*

XML 表示法

```
<xs:element name="TextAreaControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="rows" type="xs:positiveInteger" use="optional" default="8"/>
  <xs:attribute name="columns" type="xs:positiveInteger" use="optional" default="20"/>
  <xs:attribute name="wrapLines" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="monospaced" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl 和 TextBoxControl

TextBoxControl 元素

定义可用于修改字符串值的单行文本控件。

表 202. TextBoxControl 的属性

属性	使用	描述	有效值
列	可选		*positiveInteger*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*

表 202. TextBoxControl 的属性 (续)

属性	使用	描述	有效值
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="TextBoxControl">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="property" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="true"/>
  <xs:attribute name="label" type="xs:string" use="optional"/>
  <xs:attribute name="labelKey" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonic" type="xs:string" use="optional"/>
  <xs:attribute name="mnemonicKey" type="xs:string" use="optional"/>
  <xs:attribute name="labelWidth" type="xs:positiveInteger" use="optional" default="1"/>
  <xs:attribute name="labelAbove" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="description" type="xs:string" use="optional"/>
  <xs:attribute name="descriptionKey" type="xs:string" use="optional"/>
  <xs:attribute name="columns" type="xs:positiveInteger" use="optional" default="20"/>
</xs:element>
```

父元素

PropertiesPanel 和 PropertiesSubPanel

子元素

Enabled、Layout 和 Visible

相关元素

CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、DBConnectionChooserControl、DBTableChooserControl、MultiFieldAllocationControl、MultiFieldChooserControl、PasswordBoxControl、PropertyControl、RadioButtonGroupControl、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SpinnerControl、TableControl 和 TextAreaControl

TextBrowserPanel 元素

表 203. TextBrowserPanel 的属性

属性	使用	描述	有效值
列	可选		*string*
container	**必需**		*string*
行	可选		*string*
textFormat	**必需**		**plainText** **html** **rtf**

表 203. TextBrowserPanel 的属性 (续)

属性	使用	描述	有效值
wrapLines	可选		*布尔值*

XML 表示法

```
<xs:element name="TextBrowserPanel">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
  <xs:attribute name="container" type="xs:string" use="required"/>
  <xs:attribute name="textFormat" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="plainText"/>
        <xs:enumeration value="html"/>
        <xs:enumeration value="rtf"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="rows" type="xs:string" use="optional"/>
  <xs:attribute name="columns" type="xs:string" use="optional"/>
  <xs:attribute name="wrapLines" type="xs:boolean" use="optional" default="false"/>
</xs:element>
```

父元素

Tab

子元素

Enabled、Layout 和 Visible

相关元素

ActionButton、ComboBoxControl、ExtensionObjectPanel、FieldAllocationList、ModelViewerPanel、SelectorPanel、StaticText、SystemControls 和 TabbedPanel

ToolbarItem 元素

定义可添加到窗口工具栏的项。

表 204. ToolbarItem 的属性

属性	使用	描述	有效值
action	**必需**		*string*
offset	可选		*nonNegativeInteger*
separatorAfter	可选		*布尔值*
separatorBefore	可选		*布尔值*
showIcon	可选		*布尔值*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:element name="ToolbarItem">
  <xs:attribute name="action" type="xs:string" use="required"/>
  <xs:attribute name="showLabel" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="showIcon" type="xs:boolean" use="optional" default="true"/>
```

```
  <xs:attribute name="separatorBefore" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="separatorAfter" type="xs:boolean" use="optional" default="false"/>
  <xs:attribute name="offset" type="xs:nonNegativeInteger" use="optional" default="0"/>
</xs:element>
```

父元素

Controls

UTF8Format 元素

XML 表示法

```
<xs:element name="UTF8Format"/>
```

父元素

FileFormatType

UserInterface 元素

定义扩展对象或工具的用户界面。

表 205. UserInterface 的属性

属性	使用	描述	有效值
actionHandler	可选		*任意*
frameClass	可选		*任意*
uiDelegate	可选		*任意*

XML 表示法

```
<xs:element name="UserInterface">
  <xs:sequence>
    <xs:element ref="Icons" minOccurs="0"/>
    <xs:element ref="Controls" minOccurs="0"/>
    <xs:element ref="Tabs" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute name="uiDelegate" use="optional"/>
  <xs:attribute name="frameClass" use="optional"/>
  <xs:attribute name="actionHandler" use="optional"/>
</xs:element>
```

父元素

DocumentOutput、Extension、InteractiveDocumentBuilder、InteractiveModelBuilder、ModelOutput 和 Node

子元素

Controls、Icons 和 Tabs

Value 元素

简单值。

表 206. Value 的属性

属性	使用	描述	有效值
value	必需		*string*

XML 表示法

```
<xs:element name="Value" type="SIMPLE-VALUE">
  <xs:attribute name="value" type="xs:string" use="required"/>
</xs:element>
```

父元素

Attribute、Attribute、ListValue、ListValue、ListValue 和 Parameter

Values 元素

XML 表示法

```
<xs:element name="Values">
  <xs:sequence>
    <xs:element name="Value" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
</xs:element>
```

父元素

AddField、ChangeField、Field、Field、MissingValues、MissingValues 和 MissingValues

子元素

值

Value 元素:

表 207. Value 的属性

属性	使用	描述	有效值
flagProperty	可选		**trueValue** **falseValue**
value	**必需**		*string*
valueLabel	可选		*string*

XML 表示法

```
<xs:element name="Value" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="valueLabel" type="xs:string" use="optional"/>
  <xs:attribute name="flagProperty">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="trueValue"/>
        <xs:enumeration value="falseValue"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:element>
```

父元素

值

Values 元素

表 208. Values 的属性

属性	使用	描述	有效值
code	**必需**		*整数*
displayLabel	可选		*string*
flagValue	可选		*布尔值*
value	**必需**		*string*

XML 表示法

```
<xs:element name="Values" type="FIELD-VALUE">
  <xs:sequence>
    <xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
    </xs:element>
  </xs:sequence>
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="code" type="xs:integer" use="required"/>
  <xs:attribute name="flagValue" type="xs:boolean"/>
  <xs:attribute name="displayLabel" type="xs:string"/>
</xs:element>
```

父元素

AddField、ChangeField、Field、Field、MissingValues、MissingValues 和 MissingValues

子元素

DisplayLabel

DisplayLabel 元素: 指定语言中的字段或值的显示标签。可以在特定语言没有标签的地方使用 displayLabel 属性。

表 209. DisplayLabel 的属性

属性	使用	描述	有效值
lang	可选		*NMTOKEN*
value	**必需**		*string*

XML 表示法

```
<xs:element name="DisplayLabel" type="DISPLAY-LABEL" minOccurs="0" maxOccurs="unbounded">
  <xs:attribute name="value" type="xs:string" use="required"/>
  <xs:attribute name="lang" type="xs:NMTOKEN" default="en"/>
</xs:element>
```

父元素

值

Visible 元素

定义 UI 组件应可视的条件。

XML 表示法

```
<xs:element name="Visible">
  <xs:sequence>
    <xs:group ref="CONDITION-EXPRESSION" minOccurs="0">
      <xs:choice>
        <xs:element ref="Condition"/>
```

```
        <xs:element ref="And"/>
        <xs:element ref="Or"/>
        <xs:element ref="Not"/>
      </xs:choice>
    </xs:group>
  </xs:sequence>
</xs:element>
```

父元素

ActionButton、CheckBoxControl、CheckBoxGroupControl、ClientDirectoryChooserControl、ClientFileChooserControl、ComboBoxControl、DBConnectionChooserControl、DBTableChooserControl、ExtensionObjectPanel、FieldAllocationList、ItemChooserControl、ModelViewerPanel、MultiFieldAllocationControl、MultiFieldChooserControl、MultiItemChooserControl、PasswordBoxControl、PropertiesPanel、PropertiesSubPanel、PropertyControl、RadioButtonGroupControl、SelectorPanel、ServerDirectoryChooserControl、ServerFileChooserControl、SingleFieldAllocationControl、SingleFieldChooserControl、SingleFieldValueChooserControl、SingleItemChooserControl、SpinnerControl、StaticText、SystemControls、TabbedPanel、TableControl、TextAreaControl、TextBoxControl 和 TextBrowserPanel

子元素

And、Condition、Not 和 Or

扩展类型

扩展类型通过添加属性和子元素来在 XML 文档中扩展元素。要在 XML 文档中使用扩展类型，请使用元素的 xsi:type 属性来指定扩展类型。然后您可以使用由扩展类型定义的属性和元素。

ItemChooserControl 类型

表 210. ItemChooserControl 的属性

属性	使用	描述	有效值
catalog	**必需**		*string*
描述	可选		*string*
descriptionKey	可选		*string*
label	可选		*string*
labelAbove	可选		*布尔值*
labelKey	可选		*string*
labelWidth	可选		*positiveInteger*
mnemonic	可选		*string*
mnemonicKey	可选		*string*
property	**必需**		*string*
showLabel	可选		*布尔值*

XML 表示法

```
<xs:complexType name="ItemChooserControl" mixed="false">
  <xs:sequence>
    <xs:choice>
      <xs:element ref="Layout" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="Enabled" minOccurs="0" maxOccurs="1"/>
```

```
      <xs:element ref="Visible" minOccurs="0" maxOccurs="1"/>
    </xs:choice>
  </xs:sequence>
</xs:complexType>
```

扩展

ComboBoxControl

子元素

Enabled、Layout 和 Visible

相关类型

ItemChooserControl

声明

这些信息开发用于在全球提供的产品和服务。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节 (DBCS) 信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本出版物中描述的产品进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的许可证持有者如果要了解有关程序的信息以达到如下目的: (i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Software Group
ATTN: Licensing

200 W. Madison St.
Chicago, IL; 60606
U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。所有这些名字都是虚构的，若现实生活中实际业务企业使用的名字和地址与此相似，纯属巧合。

如果您正在查看本信息的软拷贝，图片和彩色图例可能无法显示。

商标

IBM、IBM 徽标和 ibm.com 是 International Business Machines Corp. 在全球许多行政管辖地区的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。Web 页面“Copyright and trademark information” (www.ibm.com/legal/copytrade.shtml) 提供了 IBM 商标的最新列表。

Intel、Intel 徽标、Intel Inside、Intel Inside 徽标、Intel Centrino、Intel Centrino 徽标、Celeron、Intel Xeon、Intel SpeedStep、Itanium 和 Pentium 是 Intel Corporation 或其子公司在美国和其他国家或地区的商标或注册商标。

Linux 是 Linus Torvalds 在美国和/或其他国家或地区的注册商标。

Microsoft、Windows、Windows NT 和 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。

Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其子公司的商标或注册商标。

其他产品和服务名称可能是 IBM 或其他公司的商标。

索引

[K]

[L]

[M]

[P]

[Q]

[R]

[S]

[T]

[W]

[X]

[Y]

[Z]

A

B

C

D

E

F

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

[特别字符]

Printed in China