文章
姚 鑫 · 六月 10, 2021 阅读大约需 分钟

第三章 指定输出的字符集

第三章 指定输出的字符集

指定输出的字符集

若要指定要在输出文档中使用的字符集，可以设置Writer实例的Charset属性。选项包括“UTF-8”、“UTF-16”以及InterSystems IRIS支持的其他字符集。

Writing the Prolog

XML文件的序言(根元素之前的部分)可以包含文档类型声明、处理指令和 注 释。

影响Prolog的属性

在writer实例中，以下属性会影响prolog:

Charset

控制两件事:XML声明中的字符集声明和(相应的)输出中使用的字符集编码。

NoXmlDeclaration

控制输出是否包含XML声明。在大多数情况下，默认值是0，这意味着已经编写了声明。如果没有指定字符集，并且输出定向到字符串或字符流，则默认为1，并且不写入任何声明。

生成文档类型声明

在根元素之前，可以包含文档类型声明，该声明声明了文档中使用的模式。
要生成文档类型声明，需要使用WriteDocType()方法，该方法有一个必选参数和三个可选参数。
就本文档而言，文档类型声明包括以下可能的部分:

```
<!DOCTYPE doc_type_name external_subset [internal_subset]>
```

如这里所示，文档类型有一个名称，根据XML规则，该名称必须是根元素的名称。
声明可以包含外部子集、内部子集或两者。

external_subset 部分指向其他地方的DTD文件。
本节的结构是以下任何一种:

```
PUBLIC public_literal_identifier
PUBLIC public_literal_identifier system_literal_identifier
SYSTEM system_literal_identifier
```

这里public_literal_identifier和system_literal_identifier是包含DTD uri的字符串。
注 意, DTD可以同时具有公共标识符和系统标识符。
下面是一个文档类型声明示例, 它包含一个同时使用公共标识符和系统标识符的外部子集:

```
<!DOCTYPE hatches <!ENTITY open-hatch
         PUBLIC "-//Textuality//TEXT Standard open-hatch boilerplate//EN"
         "http://www.textuality.com/boilerplate/OpenHatch.xml">>
```

internal_subset部分是一组实体声明。
下面是一个文档类型声明的示例, 它只包含一组内部声明:

```
<!DOCTYPE teams [ <!ENTITY baseball team (name,city,player*)>
                    !ENTITY name (#PCDATA)>
                    !ENTITY city (#PCDATA)>
                    !ENTITY player (#PCDATA)>
  ] >
```

WriteDocType()方法有四个参数:

- 第一个参数指定文档类型的名称, 用于在这个XML文档中使用。
  这是必需而且必须是有效的XML标识符。
  还必须将此名称用作本文档中根级别元素的名称。
- 可选的第二个和第三个参数指定声明的外部部分, 如下所示:

WriteDocType参数
第二个参数

“publicURI”

“publicURI”

null

- 可选的第四个参数指定声明的内部部分。如果此参数非空, 则将其括在方括号[]中, 并适当地放在声明的末尾
  。没有添加其他字符。

编写处理指令

要将处理指令写入XML，请使用WriteProcessingInstruction()方法，该方法有两个参数：
1. 处理指令(也称为目标)的名称。
2. 指令本身是一个字符串。

该方法将以下内容写入XML：

```
<?name instructions?>
```

例如，要编写以下处理指令：

```
<?xml-stylesheet type="text/css" href="mystyles.css"?>
```

为此，可以按如下方式调用WriteProcessingInstruction()方法：

```
 set instructions="type=""text/css"" href=""mystyles.css"""
 set status=writer.WriteProcessingInstruction("xml-stylesheet", instructions)
```

指定默认命名空间

在编写器实例中，可以指定默认命名空间，该命名空间仅应用于没有Namespace参数设置的类。有几个选项：

- 可以在输出方法中指定默认命名空间。四个主要的输出方法(RootObject()、RootElement()、Object()或Element())都接受名称空间作为参数。只有在类定义中未设置Namespace参数时，才会将相关元素分配给Namespace。
- 可以为编写器实例指定总默认命名空间。为此，请为编写器实例的DefaultNamespace属性指定值。

```
Class Writers.BasicDemoPerson Extends (%RegisteredObject, %XML.Adaptor)
{

Parameter XMLNAME = "Person";

Property Name As %Name;

Property DOB As %Date;

}
```

默认情况下，如果我们只是导出此类的对象，我们会得到如下所示的输出：

```
<?xml version="1.0" encoding="UTF-8"?>
<Person>
  <Name>Persephone MacMillan</Name>
  <DOB>1976-02-20</DOB>
</Person>
```

相反，如果我们在编写器实例中将DefaultNamespace设置为"http://www.person.org"，然后导出一个对象，则会收到如下所示的输出：

```
<?xml version="1.0" encoding="UTF-8"?>
<Person xmlns="www.person.org">
  <Name>Persephone MacMillan</Name>
  <DOB>1976-02-20</DOB>
</Person>
```

在本例中，<Person> 元素使用默认名称空间，否则不会分配给名称空间。

#Caché #InterSystems IRIS

源 URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%89%E7%AB%A0-%E6%8C%87%E5%AE%9A%E8%BE%93%E5%87%BA%E7%9A%84%E5%AD%97%E7%AC%A6%E9%9B%86