文章
Nicky Zhu · 九月 27, 2021 阅读大约需 34 分钟

IRIS 2021 技术文档 First Look 17 Globals

*/

目录

技术概要： Globals 1

0. 什么是 Globals？ 1
1. 为什么要学习 Globals？ 1
2. 试一试： 访问 Globals 的三种方式 2
    0. 用前须知 2
    1. 导入和检查类定义 2
    2. 导入示例数据并检查 Globals 3
    3. 关系访问 Globals（Accessing Globals Relationaly） 5
    4. 将 Globals 作为对象访问（Accessing Globals as Objects） 6
    5. 直接访问 Globals（Accessing Globals Directly） 7
    6. 有关 Globals 的更多信息 8
3. 使用 InterSystems IRIS API 访问 Globals 9
4. 了解有关 Globals 的更多信息 10
    0. Globals 及其结构 10
    1. 多模型开发 10
    2. InterSystems Native API 10

技术概要: Globals

本文档向您介绍了 globals 的概念，globals是 InterSystems IRIS®数据平台的底层存储结构。我们将向您展示如何使用关系模型和对象模型访问 globals，以及如何直接访问 globals。

要浏览所有的技术概要（First Look），包括可以在 InterSystems IRIS 免费的评估实例上执行的那些，请参见 InterSystems First Looks（《InterSystems 技术概要》）。

什么是 Globals？

InterSystems IRIS 数据平台的特点之一是它能够一次性存储数据，并允许您使用多个范式（paradigm）来访问它。例如，您可以使用 InterSystems SQL 将数据可视化为行和列，或者您可以使用 ObjectScript 将数据视为具有属性和方法的对象。您的应用程序甚至可以混合使用这两种数据模型，对于给定的任务使用最简单和更有效的模型。但是无论您如何访问数据，InterSystems IRIS 都将其存储在被称为 globals 的底层数据结构中。

Globals 可以被认为是持久化多维稀疏数组（persistent multidimensional sparse arrays）：

- *持久化（Persistent）*------Globals 存储在数据库中，可以在任何时候被任何可以访问该数据库的进程（process）检索到。
- *多维（Multidimensional）*------global 节点可以有任意数量的下标。这些下标可以是整数、小数，或字符串。
- *稀疏（Sparse）*------节点下标不必是连续的，这意味着没有存储值的下标不会使用任何存储。

Global 节点可以存储多种类型的数据，包括：

- 字符串
- 数字数据
- 字符或二进制数据流
- 数据集合，如列表或数组
- 对其他存储位置的引用

即使是您编写的服务器端代码，最终也存储在 globals 中!

为什么要学习 Globals？

在了解很少或几乎不了解 globals 的情况下，虽然也可以在 InterSystems IRIS
平台上编写应用程序，但出于以下几个原因，您可能想了解有关它们的更多信息：

- 如果您直接访问 globals，有些操作可能会更容易或更有效。
- 您可能希望为不符合关系或对象数据模型的数据创建自定义数据结构。

试一试: 访问 Globals 的三种方式

- 有些系统管理任务是在 global 层面完成的，了解 globals 将使这些任务对您更有意义。

试一试：访问 Globals 的三种方式

在本文档中，您将检查用于存储 U.S. 州类对象的数据的
globals。该类的属性包括
州名、双字母邮政缩写、首府、建州年份和面积（
单位：平方英里）。然后，您将使用关系和对象技术，以及直接操作 globals 来访问和存储这个类的对象。

用前须知

要使用这个技术概要（First Look），您需要一个正在运行的 InterSystems IRIS 实例。您的选择包括几种类型的已授权的和免费的评估实例；该实例不需要由您正在工作的系统托管（尽管您的系统必须能够通过网络访问该实例）。关于如何部署每种类型的实例的信息（如果您还没有可使用的实例），请参见 *InterSystems IRIS Basics: Connecting an IDE*（《*InterSystems IRIS 基础：连接一个 IDE*》）中的 Deploying InterSystems IRIS（部署 InterSystems IRIS）。

您也需要知道：

- 实例的基于 web 的管理门户（Management Portal）的 URL，这是 InterSystems IRIS 的系统管理用户界面。
- 如何访问终端（Terminal），InterSystems IRIS 命令行工具。
- 实例的用户名和密码（InterSystems Labs 上的 web 实例不需要）。

您还需要从 InterSystems GitHub
repo ：https://github.com/intersystems/First- Look-Globals 下载两个示例文件。

- FirstLookGlobals.xml 包含 State 类的类定义。
- FirstLookGlobals.gof 包含了 U.S. 最初 13 个州的一些 global 示例数据。

关于如何访问管理门户（Management Portal ）或终端（Terminal）的更多信息，请参见 *InterSystems IRIS*

Basics:Connecting an IDE（《*InterSystems IRIS 基础：连接一个 IDE*》）中的"InterSystems IRIS Connection Information（InterSystems IRIS 连接信息）"。您实际上并不需要一个 IDE 来做这些练习。

导入和检查类（Class）定义

首先，将 State 类定义导入 InterSystems IRIS：

0. 从管理门户（Management Portal）的主页，选择 **System Explorer（系统资源管理器）** > **Classes（类）**。
1. 在 Classes 页面上，查看左栏，确保您在 USER 命名空间。您可以把命名空间看作是工作空间或目录。
2. 点击 **Import（导入）**。
3. 在 Import Classes 的对话框中： a. 如果您的 InterSystems IRIS 实例在远程服务器上运行，请指定是将示例文件下载到远程服务器还是下载到您的本地计算机。 b. 在 File 或 Directory 的 Import（导入）区域，点击 **File（文件）**。 c. 浏览您从 GitHub 下载的 FirstLookGlobals.xml 文件。

试一试: 访问 Globals 的三种方式

d. 选择 **Compile Imported Items（编译导入项）**。

e. 对于 **Compile Flags（编译标志）**，指定 cuk。

f. 点击 **Next（下一步）**。

g. 点击 **Import（导入）**。

h. 当出现加载成功的消息时，点击 **Done（完成）**。

现在，在 Classes 页面，您应该看到 FirstLook.State.cls 在类的列表中。在 InterSystems IRIS 中，包含类的包名称（FirstLook）附加有类的名称（State）。扩展名 .cls 用来表示类文件。

注意： 如果您的命名空间包含大量的类，可以在页面左栏的 **Class Name（类名）** 框中输入 F*.cls 来过滤列表。

在 FirstLook.State.cls 类的右侧，点击 **Documentation（文档）** 来查看为该类生成的文档。

您会注意到的第一件事是 FirstLook.State 类扩展了 %Persistent 类，这意味着该类的数据将存储在数据库中 (系统定义的类和方法通常以 % 开头) 。

persistent class FirstLook.State extends %Persistent

扩展 %Persistent 还提供了一些方法，您可以使用这些方法在该类上执行操作。例如，它允许您创建该类的新对象或从数据库访问对象并将其加载到内存中。

再往下看，您可以看到这个类有五个属性：Area、Capital、Established、Name 和 PostalAbbr。

property Area as %Integer; property Capital as %String; property Established as %Integer;

property Name as %String [Required]; property PostalAbbr as %String;

这个类还有两个索引，CapitalIndex 和 PostalAbbrIndex，这两个索引可以加快 SQL 查询的速度，并允许您通过首府和邮政缩写快速查找州。

index (CapitalIndex on Capital) [Unique];

index (PostalAbbrIndex on PostalAbbr) [Unique];

导入示例数据并检查 Globals

要了解存储 State 类对象的 globals，首先要导入一些数据：

0. 从管理门户（Management Portal）的主页，选择 **System Explorer（系统资源管理器）** > **Globals**。(或从 Classes 页面，点击 **Globals** 按钮。)
1. 在 Globals 页面上，查看左栏，确保您在 USER 命名空间。
2. 点击 **Import（导入）**。
3. 在 Import Globals 对话框中： a. 如果您的 InterSystems IRIS 实例在远程服务器上运行，请指定是将示例文件下载到远程服务器还是下载到您的本地计算机。 b. 浏览您从 GitHub 下载的 FirstLookGlobals.gof 文件。 c. 点击 **Next（下一步）**。 d. 点击 **Import（导入）**。

试一试: 访问 Globals 的三种方式

e. 当出现加载成功的消息时，点击 **Done（完成）**。

现在，在 Globals 页面上，您应该看到 FirstLook.StateD 和 FirstLook.StateI 这两个 globals 在列表中。默认情况下，类（class）的数据存储在名称后面加 Dappended 的 global 中，索引存储在名称后面加 Iappended 的 global 中。最常见的情况是，您会看到 globals 名称前有一个插入符号（\^）。

注意： 如果您的命名空间包含大量的 globals，可以在页面左栏的 **Global Name（Global 名称）** 框中输入 F* 来过滤这个列表。

在 FirstLook.StateD global 的右边，点击 **View（查看）** 来显示 global 内容的列表。

```
^FirstLook.StateD = 13

^FirstLook.StateD(1) = $lb("","Delaware","DE","Dover",1787,2489)

^FirstLook.StateD(2) = $lb("","Pennsylvania","PA","Harrisburg",1787,46054)

^FirstLook.StateD(3) = $lb("","New Jersey","NJ","Trenton",1787,8723)

^FirstLook.StateD(4) = $lb("","Georgia","GA","Atlanta",1788,59425)

^FirstLook.StateD(5) = $lb("","Connecticut","CT","Hartford",1788,5543)

^FirstLook.StateD(6) = $lb("","Massachusetts","MA","Boston",1788,10554)

^FirstLook.StateD(7) = $lb("","Maryland","MD","Annapolis",1788,12406)

^FirstLook.StateD(8) = $lb("","South Carolina","SC","Columbia",1788,32020)

^FirstLook.StateD(9) = $lb("","New Hampshire","NH","Concord",1788,9349)

^FirstLook.StateD(10) = $lb("","Virginia","VA","Richmond",1788,42775)

^FirstLook.StateD(11) = $lb("","New York","NY","Albany",1788,54555)

^FirstLook.StateD(12) = $lb("","North Carolina","NC","Raleigh",1789,53819)

^FirstLook.StateD(13) = $lb("","Rhode Island","RI","Providence",1790,1545)
```

您可以看到，每个状态的节点都有一个整数的下标，称为对象 ID（或简称 ID），它是在您存储这个类的新对象时由系统生成的。Global

的根节点，没有下标，包含一个计数器，该计数器递增以生成下一个 ID。

每个节点的数据按照在类定义中指定的顺序存储为属性列表。如果查看
\^FirstLook.StateD(3)，您可以看到州名是新泽西州（New Jersey），邮政缩写是
NJ，首府是特伦顿（Trenton），该州于 1787 年加入联邦（Union），面积为 8723 平方英里。

点击 **Cancel（取消）** 返回 globals 列表，然后点击 global FirstLook.StateI 旁边的 **View（查看）** 查看 State 类的索引。

```
\^FirstLook.StateI("CapitalIndex"," ALBANY",11) = ""

\^FirstLook.StateI("CapitalIndex"," ANNAPOLIS",7) = ""

\^FirstLook.StateI("CapitalIndex"," ATLANTA",4) = ""

\^FirstLook.StateI("CapitalIndex"," BOSTON",6) = ""

\^FirstLook.StateI("CapitalIndex"," COLUMBIA",8) = ""

\^FirstLook.StateI("CapitalIndex"," CONCORD",9) = ""

\^FirstLook.StateI("CapitalIndex"," DOVER",1) = ""

\^FirstLook.StateI("CapitalIndex"," HARRISBURG",2) = ""

\^FirstLook.StateI("CapitalIndex"," HARTFORD",5) = ""

\^FirstLook.StateI("CapitalIndex"," PROVIDENCE",13) = ""

\^FirstLook.StateI("CapitalIndex"," RALEIGH",12) = ""

\^FirstLook.StateI("CapitalIndex"," RICHMOND",10) = ""

\^FirstLook.StateI("CapitalIndex"," TRENTON",3) = ""

\^FirstLook.StateI("PostalAbbrIndex"," CT",5) = ""

\^FirstLook.StateI("PostalAbbrIndex"," DE",1) = ""

\^FirstLook.StateI("PostalAbbrIndex"," GA",4) = ""

\^FirstLook.StateI("PostalAbbrIndex"," MA",6) = ""

\^FirstLook.StateI("PostalAbbrIndex"," MD",7) = ""

\^FirstLook.StateI("PostalAbbrIndex"," NC",12) = ""

\^FirstLook.StateI("PostalAbbrIndex"," NH",9) = ""

\^FirstLook.StateI("PostalAbbrIndex"," NJ",3) = ""

\^FirstLook.StateI("PostalAbbrIndex"," NY",11) = ""

\^FirstLook.StateI("PostalAbbrIndex"," PA",2) = ""

\^FirstLook.StateI("PostalAbbrIndex"," RI",13) = ""

\^FirstLook.StateI("PostalAbbrIndex"," SC",8) = ""
```

\^FirstLook.StateI("PostalAbbrIndex"," VA",10) = ""

仔细看看 global \^FirstLook.StateI
中的第一个节点。这里，第一个下标（"CapitalIndex"
）是索引的名称，第二个下标是被索引的属性的值（"ALBANY"
），第三个下标是以奥尔巴尼（Albany）为首府的州的 ID（11）。如果您回顾一下 global 数据，您会发现 \^FirstLook.StateD(11) 是纽约州（New York）。

Globals 使用数组下标自动按排序顺序存储。

试一试: 访问 Globals 的三种方式

0. 关系访问 Globals （Accessing Globals Relationally）

现在，看看为 State 类生成的 InterSystems IRIS 表：

0. 从管理门户（Management Portal）的主页，选择 **System Explorer（系统资源管理器）** > **SQL**。
1. 在 SQL 页面上，查看页面的顶部，确保您在 USER 名称空间。
   如果不是，请点击 **Switch（切换）**来改更改命名空间。
2. 在左栏中，展开 **Tables（表）**部分来查看命名空间中的表。
3. 在 Tables 下，点击 **FirstLook.State** 表。
4. 在 **Catalog Details（目录详情）** 标签上，点击 **Fields（字段）**，您将看到为 State
   类的每个属性生成的字段，以及称为 ID（有时称为 RowID） 的额外字段。 这个字段类似于您在类的 globals 中看到的对象 ID。

   注意： 如果您的命名空间包含大量的表，可以在页面左栏的 **Filter（过滤）** 框中输入 F* 来过滤列表。

6. 点击 **Execute Query（执行查询）** 标签，打开一个文本区，您可以在这里编写针对 FirstLook.State
   表运行的查询。
7. 输入以下查询：

   SELECT * FROM FirstLook.State WHERE ID = 6

   并点击 **Execute（执行）**。该查询返回您请求的马萨诸塞州（Massachusetts ）的行。

8. 输入以下查询：

   INSERT INTO FirstLook.State(Area, Capital, Established, Name, PostalAbbr) VALUES (9616, 'Montpelier', 1791, 'Vermont', 'VT')

   并点击 **Execute（执行）**，将一行插入第 14 个州佛蒙特州（Vermont ）的表中。

   要查看在 FirstLook.State 表中插入一行的效果，请返回到命名空间 USER 的 Globals 页面，并再次查看 globals。

0. 从管理门户（Management Portal）的主页，选择 **System Explorer（系统资源管理器）** > **Globals**。
1. 在 Globals 页面上，查看左栏，确保您在 USER 命名空间。

   查看数据 global，\^FirstLook.StateD，您可以看到 ID 计数器已经增加：

   \^FirstLook.StateD = 14

   并且一个新的节点被添加到 global：

^FirstLook.StateD(14) = $lb("","Vermont","VT","Montpelier",1791,9616)

查看索引 global，^FirstLook.StateI，可以看到索引已经被更新为两个新节点：

^FirstLook.StateI("CapitalIndex"," MONTPELIER",14) = ""

和

^FirstLook.StateI("PostalAbbrIndex"," VT",14) = ""

试一试: 访问 Globals 的三种方式

将 Globals 作为对象访问（Accesing Globals as Objects）

请记住，当您创建 State 类时，扩展了类
%Persistent，这使您可以访问一些有用的方法，这些方法使您可以在类的 globals
中存储数据并再次检索它。接下来，通过在 InterSystems 终端（Terminal）中编写一些 ObjectScript
来测试其中一些方法。如果您以前没有使用过终端（Terminal），请参见 *InterSystems IRIS Basics:Connecting an IDE*（《*InterSystems IRIS 基础：连接一个 IDE*》）中的"InterSystems IRIS Connection Information（InterSystems IRIS 连接信息"。

在您启动 Terminal session （终端会话）后，您应该看到一个提示符，指示您所处的名称空间。如果您不在 USER 命名空间，执行以下命令：

```
set $namespace = "USER"
```

首先，将您刚刚用 SQL 查询添加的佛蒙特州（Vermont）的数据加载到内存中。调用 FirstLook.State
类的 **%OpenId()** 方法，并将返回值赋给变量
vt。**%OpenId()** 返回对象的"句柄（handle）"，更正式地称为对象引用或 OREF。

```
USER>set vt = ##class(FirstLook.State).%OpenId(14)
```

您可以通过访问对象的 Name 属性来查看刚刚加载的状态的名称：

```
USER>write vt.Name

Vermont
```

您可以使用 **zwrite** 命令获取所有对象属性的摘要。该命令提供的其他信息超出了本文档的范围。

```
USER>zwrite vt

vt=2@FirstLook.State ; <OREF>

+----------------- general information ---------------

¦ oref value: 2

¦ class name: FirstLook.State

¦ %%OID: $lb("14","FirstLook.State")

¦ reference count: 2

+----------------- attribute values ------------------

¦ %Concurrency = 1 <Set>
```

¦ Area = 9616

¦ Capital = "Montpelier"

¦ Established = 1791

¦ Name = "Vermont"

¦ PostalAbbr = "VT"

+--

要创建一个新的 State 对象，请使用 **%New()** 方法，该方法将一个 OREF 返回给新对象。

USER>set newstate = ##class(FirstLook.State).%New()

现在，设置第 15 个州肯塔基州（Kentucky）的属性。USER>set newstate.Name = "Kentucky" USER>set newstate.PostalAbbr = "KY" USER>set newstate.Capital = "Frankfort" USER>set newstate.Established = 1792 USER>set newstate.Area = 40408

检查所有属性。

试一试: 访问 Globals 的三种方式

USER>zwrite newstate

newstate=4@FirstLook.State ; <OREF>

+----------------- general information ---------------

¦ oref value: 4

¦ class name: FirstLook.State

¦ reference count: 2

+----------------- attribute values ------------------

¦ %Concurrency = 1 <Set>

¦ Area = 40408

¦ Capital = "Frankfort"

¦ Established = 1792

¦ Name = "Kentucky"

¦ PostalAbbr = "KY"

+--

如果您对一切看起来很满意，通过调用新对象的 **%Save()** 方法将对象保存到磁盘。**%Save()** 方法返回一个状态，如果保存成功，该状态的值为 1。

USER>set status = newstate.%Save()

```
USER>write status

1
```

再一次，通过进入管理门户（Management Portal）中的 Globals 页面来检查您的工作。查看数据 global，

^FirstLook.StateD，您可以看到 ID 计数器再次增加：

```
^FirstLook.StateD = 15
```

并且一个新的节点被添加到 global：

```
^FirstLook.StateD(15) = $lb("","Kentucky","KY","Frankfort",1792,40408)
```

查看索引 global，^FirstLook.StateI，可以看到索引已经被更新为两个新节点：

```
^FirstLook.StateI("CapitalIndex"," FRANKFORT",15) = ""
```

和

```
^FirstLook.StateI("PostalAbbrIndex"," KY",15) = ""
```

直接访问 Globals（Accessing Globals Directly）

虽然 InterSystems IRIS 中的数据最常见的是通过 SQL 或使用对象层来访问，但您也可以直接访问扩展 %Persistent 的类的 globals。这种方法比较棘手，因为您需要知道 global 的结构。

首先，找到 ID 为 15 的州名称。

在终端（Terminal）中，用下标 15 将变量 ky 赋给数据 global 中的节点。

```
USER>set ky = ^FirstLook.StateD(15)
```

您可以使用 **zwrite** 检查存储在这个节点上的值：

```
USER>zwrite ky

ky=$lb("","Kentucky","KY","Frankfort",1792,40408)
```

接下来，使用 **$list()** 函数来获取列表中的第二项：

```
USER>write $list(ky, 2)

Kentucky
```

根据您在使用 SQL 或对象方法添加状态后检查 globals 所学到的内容，编写一些代码来添加新状态。

首先，准备数据，将每个属性（从对象的角度考虑）或字段（从 SQL 的角度考虑）放入不同的变量。

试一试: 访问 Globals 的三种方式

```
USER>set name = "Tennessee"

USER>set postalabbr = "TN"

USER>set capital = "Nashville"
```

```
USER>set established = 1796

USER>set area = 42144
```

然后使用 **$listbuild()** 函数构建可以存储的属性列表。

```
USER>set properties = $listbuild("", name, postalabbr, capital, established, area)

USER>zwrite properties

properties=$lb("","Tennessee","TN","Nashville",1796,42144)
```

使用一个 ObjectScript 语句，增加 ^FirstLook.StateD global 的 ID 计数器，并将新值赋给变量 id。

```
USER>set id = $increment(^FirstLook.StateD)

USER>write ^FirstLook.StateD

16

USER>write id

16
```

现在，将数据存储在数据 global 中。

```
USER>set ^FirstLook.StateD(id) = properties

USER>zwrite ^FirstLook.StateD(id)

^FirstLook.StateD(16)=$lb("","Tennessee","TN","Nashville",1796,42144)
```

现在我们需要手动更新索引，否则在首府（Capital）或邮政缩写（PostalAbbr）上，带有 WHERE 子句的 SQL 查询将不包括田纳西州（Tennessee）。这个过程比存储数据要复杂一些。

对于字符串值（string value）的索引，InterSystems IRIS
将字符串转换为大写字母，并在
前面添加一个空格字符，以便于排序。使用 _
运算符将一个空格连接到大写字母的前面，然后使用 **$zconvert()**
函数将其转换为大写字母。然后对邮政缩写执行相同操作。

```
USER>set capital = $zconvert(" "_capital, "U")

USER>set postalabbr = $zconvert(" "_postalabbr, "U")
```

最后，将索引条目存储在索引 global 中。

```
USER>set ^FirstLook.StateI("CapitalIndex", capital, id) = ""

USER>set ^FirstLook.StateI("PostalAbbrIndex", postalabbr, id) = ""
```

为确保您正确完成工作，请进入管理门户（Management Portal）中的 Globals 页面，并查看

^FirstLook.StateD 和 ^FirstLook.StateI globals。您也可以使用 zwrite ^FirstLook.StateD 和

zwrite ^FirstLook.StateI 从终端（Terminal）显示每个 global 的完整内容。

3. 有关 Globals 的更多信息

0. 使用 SQL 创建 Globals

在本文档中，您所看到的 globals 是通过存储使用类定义（class definition）指定的类的对象（object）创建的。您可以通过使用 SQL 创建表和表的索引来创建结构非常相似的 globals。然后，InterSystems IRIS 将根据您创建的表为您生成一个类。

作为练习，进入管理门户（Management Portal）中 USER 命名空间的 SQL 页面，并使用 Execute Query（执行查询）标签执行下列语句。

使用 InterSystems IRIS API 访问 Globals

CREATE TABLE FirstLook.SQL (%CLASSPARAMETER USEEXTENTSET 0, %CLASSPARAMETER DEFAULTGLOBAL =

'\^FirstLook.SQL',

Name CHAR(30) NOT NULL, PostalAbbr CHAR(2), Capital CHAR(30), Established INT, Area INT)
CREATE UNIQUE INDEX CapitalIndex ON FirstLook.SQL (Capital)

CREATE UNIQUE INDEX PostalAbbrIndex ON FirstLook.SQL (PostalAbbr) INSERT INTO FirstLook.SQL (Name, PostalAbbr, Capital, Established, Area) VALUES ('Maine', 'ME', 'Augusta', 1820, 35380)

在 SQL 页面的左栏中，您现在应该看到 FirstLook.SQL 表。

进入 Classes 页面，找到类 FirstLook.SQL.cls
并查看其文档。您注意到有什么不同吗？这个类有一个额外的索引，称为位图扩展索引（Bitmap Extent Index）。然后进入 Globals 页面，找到 \^FirstLook.SQLD 和 \^FirstLook.SQLI globals。看看 \^FirstLook.SQLI，并看看您是否能找到额外的索引。

创建自定义 Globals

到目前为止，您已经看到了扩展 %Persistent 类时创建的 globals。但是，globals 可用于存储无模式的数据，这些数据可能不适合类或关系范式。例如，使用终端（Terminal）或管理门户（Management Portal），切换到 %SYS 命名空间，并检查 \^CONFIGglobal，它存储了一些 InterSystems IRIS 配置设置。下面是 \^CONFIGglobal 示例的一部分：

\^CONFIG("Cluster","CommIPAddress") = ""

\^CONFIG("Cluster","JoinCluster") = 0

\^CONFIG("ConfigFile","Version") = "2018.20"

\^CONFIG("Conversions","LastConvertTime") = "2019-03-13 08:39:45"

\^CONFIG("Databases","ENSLIB") = "/usr/irissys/mgr/enslib/"

\^CONFIG("Databases","IRISAUDIT") = "/usr/irissys/mgr/irisaudit/"

\^CONFIG("Databases","IRISLIB") = "/usr/irissys/mgr/irislib/"

\^CONFIG("Databases","IRISLOCALDATA") = "/usr/irissys/mgr/irislocaldata/"

\^CONFIG("Databases","IRISSYS") = "/usr/irissys/mgr/"

\^CONFIG("Databases","IRISTEMP") = "/usr/irissys/mgr/iristemp/"

```
\^CONFIG("Databases","USER") = "/usr/irissys/mgr/user/"
```

如果您想创建和存储自己的自定义数据结构，您可以使用 ObjectScript 轻松做到。使用下面的示例，您可以编写几行代码定义一个有向图来存储机场之间的机票价格：

```
USER>set \^Fares("BOS", "ORD") = 87

USER>set \^Fares("ORD", "BOS") = 63

USER>set \^Fares("BOS", "LAX") = 143

USER>set \^Fares("LAX", "BOS") = 143

USER>set \^Fares("ORD", "LAX") = 57

USER>set \^Fares("LAX", "ORD") = 94

USER>zwrite \^Fares

\^Fares("BOS","LAX")=143

\^Fares("BOS","ORD")=87

\^Fares("LAX","BOS")=143

\^Fares("LAX","ORD")=94

\^Fares("ORD","BOS")=63

\^Fares("ORD","LAX")=57
```

使用 InterSystems IRIS API 访问 Globals

如果您正在用 Java、.NET、Node.js 或 Python 编写应用程序，InterSystems IRIS 提供的 API 允许您使用本文中讨论的三种模型来操作您的数据库：

- 通过 JDBC、ADO.NET 或 PyODBC API 进行关系访问
- 通过 InterSystems XEP API 访问对象
- 通过 InterSystems Native API 直接访问 globals

了解有关 Globals 的更多信息

现在您知道了，无论您决定如何存储或访问数据，您所做的就是使用 globals。

注意： 并非所有语言都支持所有形式的访问。

了解有关 Globals 的更多信息

使用下面列出的参考资料来了解更多有关 globals 和如何访问它们的信息。

Globals 及其结构

- Globals QuickStart（Globals 快速入门）------提供 global 结构的可视化视图，并给出在自定义 global 结构中存储数据的示例。

- Using Globals（《使用 Globals》）------讨论 Globals 的结构、如何管理它们以及如何使用 SQL 或 ObjectScript 访问它们。
- Globals------*Defining and Using Classes（定义和使用类）*这部分解释了为扩展 %Persistent 类的类创建的 globals 的命名约定。

0. 多模型开发

- Multi-Model QuickStart（多模型快速入门）------描述了 InterSystems IRIS 如何允许您用各种不同的语言（包括 Java、.NET 和 ObjectScript）使用您选择的数据模型。
- Java QuickStart（Java 快速入门）------向您展示如何使用 Java API 对 InterSystems IRIS 数据库进行关系、对象和直接访问。
- .NET QuickStart（.NET 快速入门）------向您展示如何使用 .NET API 对 InterSystems IRIS 数据库进行关系、对象和直接访问。
- Python QuickStart（Python 快速入门）------向您展示如何使用 Python API 对 InterSystems IRIS 数据库进行关系、对象和直接访问。

0. InterSystems Native API

- Using the Native API for Java（《使用 Native API for Java》）（交互式课程）------向您展示如何使用 Native API for Java 访问 globals 以及调用类方法（call class method）和例程。
- Using the Native API for Java（《使用 Native API for Java 》）（书籍）------向您展示如何使用 Native API for Java 访问 globals 以及调用类方法（call class method）和例程。
- First Look: InterSystems IRIS Native API for Java（《技术概要：InterSystems IRIS Native API for Java》） --- 展示如何从 Java 应用程序访问 InterSystems IRIS globals。
- Using the Native API for .NET（《使用 Native API for .NET》）（交互式课程）------向您展示如何使用 Native API for .NET 访问 globals 以及调用类方法（call class method）和例程。
- Using the InterSystems Native API for .NET（《使用 InterSystems Native API for .NET》）（书籍）------向您展示如何使用 Native API for .NET 访问 globals 以及调用类方法（call class method）和例程。
- First Look: InterSystems IRIS Native API for .NET（《技术概要：InterSystems IRIS Native API for .NET》） ------展示如何从 .NET 应用程序访问 InterSystems IRIS globals。

了解有关 Globals 的更多信息

- Node.js QuickStart（Node.js 快速入门）------向您展示如何使用 Native API for Node.js 访问 globals 以及调用类方法（call class method）和例程。
- Using the Native API for Node.js（《使用 Native API for Node.js》）------向您展示如何使用 Native API for Python 访问 globals 以及调用类方法（call class method）和例程。
- First Look: InterSystems IRIS Native API for Node.js（《技术概要：InterSystems IRIS Native API for Node.js》）-展示如何从 Node.js 应用程序访问 InterSystems IRIS globals。
- Using the Native API for Python（《使用 Native API for Python》）（交互式课程）------向您展示如何使用 Native API for Python 访问 globals 以及调用类方法（call class method）和例程。
- Using the Native API for Python（《使用 Native API for Python》）（书籍）------向您展示如何使用 Native API for Python 访问 globals 以及调用类方法（call class method）和例程。
- First Look: InterSystems IRIS Native API for Python（《技术概要：InterSystems IRIS Native API for Python》）------展示如何从 Python 应用程序访问 InterSystems IRIS globals。

#InterSystems IRIS #InterSystems IRIS for Health

源 URL: https://cn.community.intersystems.com/post/iris-2021-%E6%8A%80%E6%9C%AF%E6%96%87%E6%A1%A3-first-look-17-globals