文章
姚 鑫 · 五月 6, 2021 阅读大约需 5 分钟

第三章 使用多维存储(全局变量)（二）

第三章 使用多维存储(全局变量)（二）

遍历全局变量中的数据

有许多方法可以遍历(迭代)存储在全局变量中的数据。

$ORDER(下一个/上一个)函数

ObjectScript $Order函数允许顺序访问全局中的每个节点。

$ORDER函数返回给定级别(下标编号)的下一个下标的值。例如，假设定义了以下全局设置：

```
Set ^Data(1) = ""
Set ^Data(1,1) = ""
Set ^Data(1,2) = ""
Set ^Data(2) = ""
Set ^Data(2,1) = ""
Set ^Data(2,2) = ""
Set ^Data(5,1,2) = ""
```

要查找第一个第一级下标，我们可以使用：

```
SET key = $ORDER(^Data(""))
```

这将返回空字符串(“ ”)之后的第一个第一级下标。(空字符串用于表示第一个条目之前的下标值；作为返回值，它用于指示没有后续的下标值。)。在本例中，key现在将包含值1。

我们可以通过在$ORDER表达式中使用1或键来查找下一个第一级下标：

```
SET key = $ORDER(^Data(key))
```

如果key的初始值为1，则此语句将其设置为2(因为^Data(2)是下一个第一级下标)。再次执行此语句会将key设置为5，因为这是下一个第一级下标。请注意，即使没有直接存储在^Data(5)中的数据，也会返回5。再次执行此语句将把key设置为空字符串(“ ”)，表示没有更多的一级下标。

通过将附加下标与$ORDER函数一起使用，可以迭代不同的下标级别。$order返回其参数列表中最后一个下标的下一个值。使用上述数据，该语句如下：

```
SET key = $ORDER(^Data(1,""))
```

将关键字设置为1，因为^Data(1，1)是下一个二级下标。再次执行此语句会将KEY设置为2，因为这是下一个二级下标。再次执行此语句将把key设置为“ ”，表示在节点^Data(1)下没有更多的二级下标。

使用$ORDER循环

下面的ObjectScript代码定义了一个简单的全局变量，然后循环遍历其所有第一级子脚本：

```
/// w ##class(PHA.TEST.Global).ReadGlobalSimpleFor()
ClassMethod ReadGlobalSimpleFor()
{
    // ??^Data???????
    Kill ^Data

    // ????????^Data
    For i = 1:1:100 {
        // ???????????
        Set ^Data(i) = ##class(%PopulateUtils).Name()
    }

    // ???????? ???????
    Set key = $Order(^Data(""))

    While (key '= "") {
        Write "#", key, " ", ^Data(key),!
        // F???????
        Set key = $Order(^Data(key))
    }

    q ""
}
```

```
DHC-APP>w ##class(PHA.TEST.Global).ReadGlobalSimpleFor()
#1 Edwards,Barbara T.
#2 Ragon,Kevin K.
#3 Avery,Josephine U.
#4 Townsend,Buzz R.
#5 Joyce,Quentin V.
#6 Xenia,Ted F.
#7 Chadwick,Wilma N.
#8 Duquesnoy,Orson A.
#9 Uberoth,Orson X.
#10 Jones,Joe O.
#11 Hills,Barb R.
#12 Yakulis,Pat J.
#13 Tesla,Al P.
#14 Goncharuk,Sam J.
#15 Presley,Amanda D.
#16 Olsen,Kristen I.
#17 Roentgen,John T.
#18 Minichillo,Elmo N.
#19 Koivu,Patrick R.
#20 Harrison,Lawrence I.
#21 Page,Agnes P.
#22 Wijnschenk,Hannah L.
#23 Chesire,Bart S.
#24 Klingman,Liza K.
```

```
#25 Smyth,Imelda J.
#26 Alton,Filomena L.
#27 Minichillo,Charles U.
#28 Nichols,Jeff W.
#29 O'Rielly,Thelma X.
#30 Schaefer,Kristen G.
#31 Black,Filomena R.
#32 Vivaldi,Xavier B.
#33 Allen,Phyllis U.
#34 Mastrolito,Zelda Z.
#35 Quilty,Jane V.
#36 Zevon,Maureen H.
#37 O'Rielly,Maureen C.
#38 Olsen,Robert W.
#39 Page,Milhouse D.
#40 Nelson,Dick R.
#41 Ironhorse,Danielle I.
#42 Tweed,Rhonda T.
#43 Quincy,Terry L.
#44 Tsatsulin,Jocelyn C.
#45 Yeats,Michelle E.
#46 Jackson,Paul V.
#47 Humby,Dave I.
#48 Kelvin,Natasha R.
#49 Kelvin,Kyra R.
#50 Yoders,Agnes R.
#51 Tesla,Amanda F.
#52 Harrison,Christen T.
#53 Allen,Nataliya J.
#54 Xenia,Diane W.
#55 Xenia,Phyllis E.
#56 Isaksen,Pam D.
#57 Waterman,Charles M.
#58 Peters,Sophia N.
#59 Peterson,Bart B.
#60 Eastman,Edward S.
#61 Young,Belinda F.
#62 White,Fred G.
#63 Ubertini,Lola U.
#64 Uhles,Xavier T.
#65 Quine,Phyllis T.
#66 Hernandez,Umberto B.
#67 Allen,Zelda S.
#68 Harrison,David Z.
#69 Harrison,Danielle T.
#70 Ott,Dick D.
#71 Lennon,Joe Y.
#72 Quigley,Alfred M.
#73 Klausner,Mario J.
#74 Tsatsulin,Emily S.
#75 Anderson,Edward R.
#76 Lennon,Fred H.
#77 DeSantis,Molly J.
#78 Browne,Dave H.
#79 Cunningham,Buzz L.
#80 Ingersol,Edgar G.
#81 Paraskiv,Linda O.
#82 Beatty,Kim H.
#83 Quilty,Wilma P.
```

```
#84 Dunlap,Jules I.
#85 Waterman,Buzz D.
#86 Edison,Kim C.
#87 Eagleman,Michael N.
#88 Huff,Hannah K.
#89 Vanzetti,Maria E.
#90 Zampitello,Angela Q.
#91 Anderson,Angela Z.
#92 Isaacs,Charlotte Q.
#93 O'Donnell,Paul A.
#94 Underman,Zeke R.
#95 Schultz,James I.
#96 Chadbourne,Janice N.
#97 Lennon,William T.
#98 Vonnegut,Pam V.
#99 Miller,Patricia T.
#100 Hills,Charles C.
```

其他$ORDER参数

ObjectScript $ORDER函数接受可选的第二个和第三个参数。
第二个参数是一个方向标志，指示希望在哪个方向上遍历全局变量。
默认值1指定正向遍历，而-1指定反向遍历。

第三个参数(如果存在)包含一个局部变量名。
如果$ORDER找到的节点包含数据，则将找到的数据写入这个本地变量。
当您在一个全局循环中，并且您对节点值和下标值感兴趣时，这样操作更有效。

$QUERY函数

如果需要访问全局变量中的每个节点和子节点，在子节点上上下移动，请使用ObjectScript $Query函数。(或者，可以使用嵌套的$ORDER循环)。

$Query函数接受全局变量引用，并返回一个字符串，其中包含全局变量中下一个节点的全局引用(如果没有后续节点，则返回"")。若要使用$QUERY返回的值，必须使用ObjectScript间接运算符(@)。

例如，假设定义了以下全局设置：

```
Set ^Data(1) = ""
Set ^Data(1,1) = ""
Set ^Data(1,2) = ""
Set ^Data(2) = ""
Set ^Data(2,1) = ""
Set ^Data(2,2) = ""
Set ^Data(5,1,2) = ""
```

以下是对$QUERY的调用：

```
SET node = $QUERY(^Data(""))
```

将节点设置为字符串“ ^Data(1) ”，即全局中第一个节点的地址。然后，要获取全局中的下一个节点，请再次调用$QUERY并在节点上使用间接运算符：

此时，节点包含字符串 “ ^Data(1,1) ” 。

以下示例定义一组全局变量节点，然后使用$QUERY遍历它们，同时写入每个节点的地址：

```
/// w ##class(PHA.TEST.Global).ReadGlobalSimpleQuery()
ClassMethod ReadGlobalSimpleQuery()
{
    Kill ^Data // ??^Data??

    // ???????^Data //
    Set ^Data(1) = ""
    Set ^Data(1,1) = ""
    Set ^Data(1,2) = ""
    Set ^Data(2) = ""
    Set ^Data(2,1) = ""
    Set ^Data(2,2) = ""
    Set ^Data(5,1,2) = ""

    // ????^Data ???????
    Set node = $Query(^Data(""))
    While (node '= "") {
        Write node,!
        // ???????
        Set node = $Query(@node)
    }
    q ""
}
```

```
DHC-APP>w ##class(PHA.TEST.Global).ReadGlobalSimpleQuery()
^Data(1)
^Data(1,1)
^Data(1,2)
^Data(2)
^Data(2,1)
^Data(2,2)
^Data(5,1,2)
```

#SQL #Caché #InterSystems IRIS #InterSystems IRIS for Health

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%89%E7%AB%A0-%E4%BD%BF%E7%94%A8%E5%A4%9A%E7%BB%B4%E5%AD%98%E5%82%A8%E5%85%A8%E5%B1%80%E5%8F%98%E9%87%8F%EF%BC%88%E4%BA%8C%EF%BC%89