Perl 数组 Perl 条件语句

Perl 哈希

哈希是 **key/value** 对的集合。

Perl中哈希变量以百分号 (%) 标记开始。

访问哈希元素格式：**${key}**。

以下是一个简单的哈希实例：

实例

```
#!/usr/bin/perl
%data = ('google', 'google.com', 'runoob', 'runoob.com', 'taobao', 'taobao.com');
print "\$data{'google'} = $data{'google'}\n";
print "\$data{'runoob'} = $data{'runoob'}\n";
print "\$data{'taobao'} = $data{'taobao'}\n";
```

执行以上程序，输出结果为：

```
$data{'google'} = google.com
$data{'runoob'} = runoob.com
$data{'taobao'} = taobao.com
```

创建哈希

创建哈希可以通过以下两种方式：

一、为每个 key 设置 value

```
$data{'google'} = 'google.com';
$data{'runoob'} = 'runoob.com';
$data{'taobao'} = 'taobao.com';
```

二、通过列表设置

列表中第一个元素为 key，第二个为 value。

```
%data = ('google', 'google.com', 'runoob', 'runoob.com', 'taobao', 'taobao.com');
```

也可以使用 **=>** 符号来设置 key/value:

```
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
```

以下实例是上面实例的变种，使用 **-** 来代替引号：

```
%data = (-google=>'google.com', -runoob=>'runoob.com', -taobao=>'taobao.com');
```

使用这种方式 key 不能出现空格，读取元素方式为：

```
$val = $data{-google}
$val = $data{-runoob}
```

访问哈希元素

访问哈希元素格式：**${key}**，实例如下：

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
print "\$data{'google'} = $data{'google'}\n";
print "\$data{'runoob'} = $data{'runoob'}\n";
print "\$data{'taobao'} = $data{'taobao'}\n";
```

执行以上程序，输出结果为：

```
$data{'google'} = google.com
$data{'runoob'} = runoob.com
$data{'taobao'} = taobao.com
```

读取哈希值

你可以像数组一样从哈希中提取值。

哈希值提取到数组语法格式：**@{key1,key2}。**

实例

```
#!/uer/bin/perl
%data = (-taobao => 45, -google => 30, -runoob => 40);
@array = @data{-taobao, -runoob};
print "Array : @array\n";
```

执行以上程序，输出结果为：

```
Array : 45 40
```

读取哈希的 key 和 value

读取所有key

我们可以使用 **keys** 函数读取哈希所有的键，语法格式如下：

```
keys %HASH
```

该函数返回所有哈希的所有 key 的数组。

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
@names = keys %data;
print "$names[0]\n";
print "$names[1]\n";
print "$names[2]\n";
```

执行以上程序，输出结果为：

```
taobao
google
runoob
```

类似的我们可以使用 **values** 函数来读取哈希所有的值,语法格式如下：

```
values %HASH
```

该函数返回所有哈希的所有 value 的数组。

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
@urls = values %data;
print "$urls[0]\n";
print "$urls[1]\n";
print "$urls[2]\n";
```

执行以上程序，输出结果为：

```
taobao.com
runoob.com
google.com
```

检测元素是否存在

如果你在哈希中读取不存在的 key/value 对 ，会返回 **undefined** 值，且在执行时会有警告提醒。

为了避免这种情况，我们可以使用 **exists** 函数来判断key是否存在，存在的时候读取：

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
if( exists($data{'facebook'} ) ){
print "facebook 的网址为 $data{'facebook'} \n";
```

```
}
else
{
print "facebook 键不存在\n";
}
```

执行以上程序，输出结果为：

```
facebook 键不存在
```

以上代码中我们使用了 **IF...ELSE** 语句，在后面的章节我们会具体介绍。

获取哈希大小

哈希大小为元素的个数，我们可以通过先获取 key 或 value 的所有元素数组，再计算数组元素多少来获取哈希的大小，实例如下：

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
@keys = keys %data;
$size = @keys;
print "1 - 哈希大小: $size\n";
@values = values %data;
$size = @values;
print "2 - 哈希大小: $size\n";
```

执行以上程序，输出结果为：

```
1 - 哈希大小: 3
2 - 哈希大小: 3
```

哈希中添加或删除元素

添加 key/value 对可以通过简单的赋值来完成。但是删除哈希元素你需要使用 **delete** 函数：

实例

```
#!/usr/bin/perl
%data = ('google'=>'google.com', 'runoob'=>'runoob.com', 'taobao'=>'taobao.com');
@keys = keys %data;
$size = @keys;
print "1 - 哈希大小: $size\n";
# 添加元素
$data{'facebook'} = 'facebook.com';
@keys = keys %data;
$size = @keys;
print "2 - 哈希大小: $size\n";
# 删除哈希中的元素
```

```
delete $data{'taobao'};
@keys = keys %data;
$size = @keys;
print "3 - 哈希大小: $size\n";
```

执行以上程序，输出结果为：

```
1 - 哈希大小: 3
2 - 哈希大小: 4
3 - 哈希大小: 3
```

← Perl 数组 Perl 条件语句 →