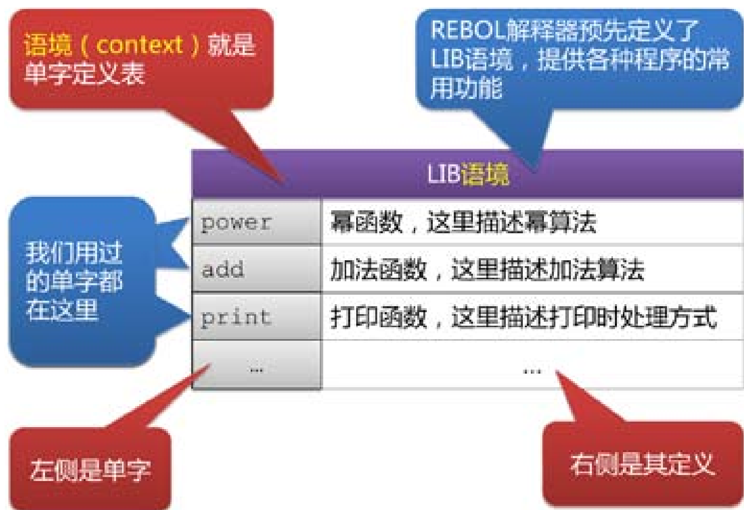
互动出版网
China-Pub.com
专业图书网上第一专营店
www.china-pub.com
独家提供样章

第6章

语境与单字

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章



当我们输入一段代码时，REBOL 解释器怎么知道如何执行这段代码?

其实 REBOL 解释器一启动，就预先准备好 LIB 语境。语境（context）是一张表，把单字对应到它的定义。这些 REBOL 解释器预先定义在 LIB 语境中的单字，我称为 REBOL 内置单字，都是一些常用单字。单字的定义可能是程序（函数）或者一般的值（例如整数）。

当 REBOL 解释器看到一个单字，就会去 LIB 语境中查询，得知如何处理。以 `power` 来说，REBOL 会从这个语境中找到它的定义，知道这个函数必须计算幂（以及如何计算幂），且需要两个参数，分别是底数与指数。

有些单字的定义是程序，这样的程序我们称为函数。有些则不是程序，而是静态的值，例如 `pi`（圆周率）就被定义为 `3.14159265358979`。有些单字的定义是完全一样的，例如 `q` 与 `quit`。执行程序时，有些单字不会影响其他值（也就是不需要参数），例如 `what-dir`、`LS`、`pi`。有些单字会影响后面的几个值（也就是需要参数），例如 `do`、`print`。有些单字会影响一前一后的值（也就是参数必须一个摆在前面，一个摆在后面），例如 `+` 与 `/`。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

通过?即可查看语境的定义

通过length?可得知语境内定义的单字个数

通过words-of可得知语境内定义的单字

想完整看到内容，写个foreach循环逐一打印

```
>> ? LIB
LIB is an object of value:
...省略
   red          tuple!   255.0.0
...省略

>> length? LIB
== 609

>> words-of LIB
== [end! unset! none! logic! integer!
decimal! percent! money! char! pair!
tuple ...省略

>> foreach w words-of lib [print w]
end!
unset!
none!
 ...省略
```

语境中有一个单字red，其值是255.0.0

foreach的用法到第3篇再详细介绍

如果我们想知道 LIB 中定义了多少个单字，可以通过 length? LIB 来查询。如果想知道 LIB 语境定义了哪些单字，可以在交互环境下输入 ? LIB 命令行来看看它内部的定义。仔细看看，有一个单字叫做 red，它的定义是 255.0.0。如果单字太多，我们无法看到全部的单字定义，可以考虑采用第 2 章介绍的 echo 函数，将结果输出到一个文件中，再慢慢研究。

如果只想看看有哪些单字，不想知道单字的定义，则可以通过 words-of 函数，来取得 LIB 语境内的所有单字，但依然会因为返回值的数据量太大，只能看到一部分。这时我们可以通过 foreach 循环的方式，将单字一一打印出来。如果你不懂 foreach 的用法，先不用担心，第 3 篇会详细介绍。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

这么设置，会影响LIB内的red定义吗？

居然不受影响

可明明改成功了，难道存在另一个语境？

LIB语境内没有my-name单字，这么设置会为LIB语境增添单字吗？

LIB语境内依然没有my-name单字，难道我们所操作的一切都在另一语境

```
>> red: 200.0.0
== 200.0.0

>> ? LIB
...省略
   red          tuple!    255.0.0
...省略

>> red
== 200.0.0

>> my-name: "Jerry"
== "Jerry"

>> ? LIB
（找不到my-name）


（接下一页操作）
```

如果我们改变 LIB 语境中已有单字的定义，会如何？ `red` 的值原本是 `255.0.0`（鲜红），我将 `red` 设置为 `200.0.0`（稍微暗一点的红）。再去观察 LIB，却发现我的设置没有效果。怎么会这样？是不是我的设置没有被记录下来。我在命令行中输入 `red`，得到的值确实是新的值。

我怀疑有另一个语境的存在。所有的定义更改只会影响这神秘的语境，而不会影响 LIB 语境。

接着做实验，我先确定 LIB 中没有 `my-name`，然后将 `my-name` 设置为 `"Jerry"`，再到 LIB 中寻找，竟然看不到 `my-name` 的存在。

上面两个实验之后，我推理出：应该有另一个语境，专门用来记录用户代码执行过程的单字定义变化。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

确实存在这样的语境：USER语境

没错，确实存在这样的语境，它就叫做 USER 语境！

USER 语境与 LIB 语境之间的分工很明确：

- LIB 语境提供各种常用的单字定义，方便我们写代码时采用。
- USER 语境用来记录用户代码执行过程中发生的变化。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章

我们所做的一切，都会影响USER语境。通过self单字，可以访问USER语境

对USER语境内的单字做修改，不会影响LIB语境

只会影响USER语境自己

```
（接上一页操作）
>> ? self
...省略
    red           tuple!     200.0.0
    my-name       string!    "Jerry"

>> print: 0
== 0

>> ? LIB
LIB is an object of value:
...省略
    print         native!    Outputs ...
...省略

>> ? self
SELF is an object of value:
...省略
    print         integer!   0
```

不受影响

受影响

通过 self 就可以访问 USER 语境。先通过 ? self 看看 USER 语境有哪些单字，以及这些单字的值是多少。我们发现，之前修改的 red 以及新增的 my-name 原来都在这里。

这时突发奇想，如果我把某个 REBOL 内置单字（比方说 print ）改掉其定义，会如何？通过实验我们发现，我们修改的是 USER 语境内的版本，而不是 LIB 内的版本。

互动出版网
China-Pub.com
专业图书网上第一专营店
www.china-pub.com
独家提供样章


```
>> ? self
SELF is an object of value:
...确定没有new-data

>> new-data: 10
== 10

>> ? self
SELF is an object of value:
...省略
   new-data     integer!   10

>> TYPO
** Script error: typo has no value

>> ? self
SELF is an object of value:
...省略
   TYPO         unset!     none
```

根据代码需求，USER
语境会随时增添单字

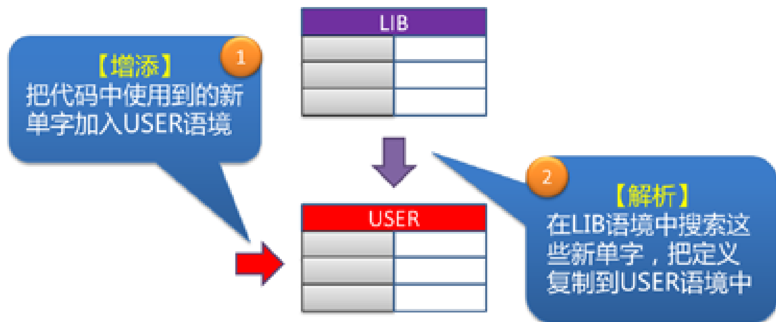
有些单字被增添到
USER语境，但无定义

这种类型表示
尚未设置定义

经过上面的实验与观察，USER 语境中并没有 LIB 中所有的单字，只有我们用到的单字。每次我们输入一行代码，新的单字就会被加进表中。

总而言之，代码在执行前，系统会把代码中所有的单字找出来，然后加到 USER 表中。对于 USER 中本来就存在的单字，不受影响。新加入的单字，会试图从 LIB 中取得定义，再复制到 USER 语境中。如此一来，我们的代码在执行时，只需要用到一个语境：USER 语境。不过 REBOL 解释器会自动帮我们处理这里所说的一切，所以我们不用为这些麻烦事担心。

互动出版网 China-Pub.com 专业图书网上第一专营店 www.china-pub.com 独家提供样章



总的来说，在代码切割完后，执行代码之前，REBOL 解释器有一些准备工作要进行。

1. 先把代码中使用到的新单字加入 USER 语境中，这些新单字的定义都是特殊值 `unset`，表示尚未设定。
2. 到 LIB 语境中搜索这些新单字的定义，把这些定义复制给 USER 语境中对应的单字。

做完了上面的准备工作，就可以开始执行代码了。通过上面的准备工作，执行时代码只需要参考 USER 语境即可，不需要参考 LIB 语境。

互动出版网 China-Pub.com
专业图书网上第一专营店
www.china-pub.com 独家提供样章

☑ 我知道USER与LIB语境存在的目的

如果你确定做到了，到本篇开始“学习目标”处打钩。

互动出版网
China-Pub.com
专业图书网上第一专营店
www.china-pub.com
独家提供样章