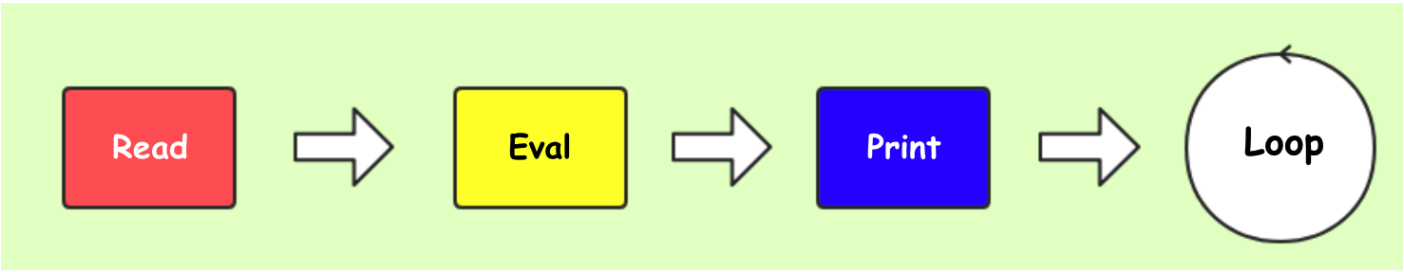
REPL



> 本文主要介绍Node提供的交互式运行环境REPL，包括常见操作以及基础命令等内容。

1.0 REPL介绍

在Node.js中，提供了一个交互式的运行环境-REPL(`Read-Eval-Print-Loop`)。

REPL解释器执行任务的方式

- ❏ Read　　**读取**用户输入并存储。
- ❏ Eval　　**执行**用户输入的代码。
- ❏ Print　　**输出**代码的执行结果。
- ❏ Loop　　**循环**执行以上步骤，直到退出。

在REPL环境中我们可以**操作变量**、**执行JavaScript文件**、**访问和使用Node内置的模块**、**声明和调用函数**甚至**开启服务器**，在该环境中可以方便的对JavaScript代码进行调试操作。

2.0 基本操作

在安装好Node.js之后，打开命令行窗口输入 `node` 指令即可进入到REPL环境。此时，命令行窗口中将会显示REPL运行环境的提示符 `>`

操作变量

```
wendingding$ node
> name = "wendingding"
'wendingding'
> age = 18
18
> console.log("姓名: "+name,"年龄: "+age)
姓名: wendingding 年龄: 18

> obj = {className:"太乙金仙",id:01};
{ className: '太乙金仙', id: 1 }
> obj.className
```

```
'太乙金仙'
> obj.id
1
```

上面的命令行中我们直接把字符串赋值给 `name` ，把数字18赋值给 `age` 。此外，还以字面量的方式创建了 `obj对象` ，而且通常观察发现REPL环境对JavaScript代码的支持还是比较好的。上面命令行在演示操作变量的时候，并没有使用 `var` 关键字，需要注意的是在REPL环境中如果**把表达式赋值给var关键字声明的变量，那么回车之后得到的结果将是undefined。**

```
wendingding:JQuery wendingding$ node
> var name = "LiuY"
undefined
> name
'LiuY'
> var obj = {name:"LiuY",age:18,birthday:"1993-08-22"};
undefined
> obj
{ name: 'LiuY', age: 18, birthday: '1993-08-22' }
> obj.showName = function(){ console.log(this.name) }
[Function]
> obj.showName
[Function]
> obj.showName();
LiuY
```

★ 为什么输出的是undefined而不是具体的值?

变量是否使用var声明其结果截然不同，是因为 `REPL环境内部使用eval函数来评估表达式的执行结果` 。

eval函数的作用主要是把字符串转换为JavaScript的代码并且马上执行，在早期开发中常用来处理JSON数据的反序列化处理(具体的详情可以参考JSON数据解析)，但因为eval函数的使用存在严重的安全隐患问题且会破坏JavaScript代码本身的词法作用域影响性能，所以不建议使用(在严格模式下禁用)。我们可以通过执行下面的代码来进行比较和验证。

```
console.log(eval("name='wendingding'"));
wendingding

console.log(eval("var newName='wendingding'"));
undefined
```

备注 在REPL环境中访问对象方法(函数)的时候，并不会完整的打印整个函数的内容而总是简单的显示和输出 `[Function]` ，这是因为函数内容可能又臭又长，做人做事呐还是简单点好：)

下划线字符

在REPL环境中，我们可以通过下划线字符(`_`)来访问最近使用的表达式。您可以通过(`_`)来访问最近的变量，对象甚至是对象的属性和方法。

```
wendingding:JQuery wendingding$ node
> index = 3
```

```
3
> _ + 4
7
>  obj = {name:"LiuY",age:18,show:function(){console.log(this.name,this.age)}}
{ name: 'LiuY', age: 18, show: [Function: show] }
> obj.name
'LiuY'
> _
'LiuY'
> obj.show
[Function: show]
> _
[Function: show]
```

> **注意** 在使用下划线字符(_)访问最近表达式的时候并不能修改变量的值。

多行输入

在REPL环境中支持输入多行代码，如果需要在REPL环境中声明和执行函数而且函数体较长，那么可以将函数分成多行来书写(直接回车即可)，当该表达式还没有完成的时候，REPL环境将为每一行添加 `...` 符号，下面给出简单的使用示例。

```
wendingding:JQuery wendingding$ node
> function sum(a,b){
... var result = a + b;
... console.log(result);
... return result;
... }
undefined
> sum(1,2);
3
3
```

3.0 基础命令

.break	返回命令提示符的起点，常用于重写代码(调整)。
.clear	返回命令提示符的起点，同 .break 命令。
.exit	退出当前的REPL运行环境。
.help	显示REPL环境中所有的基础命令。
.save	把REPL环境中输入的所有表达式保存到文件。
.load	把指定文件中的所有表达式依次加载到当前的REPL运行环境。

说明 在Node.js中提供了一些基础命令来帮助我们更好的使用REPL运行环境，这些基础命令都以点(.)开始，下面给出简单示例。

```
wendingding:JQuery wendingding$ cd /Users/文顶顶/Desktop/node
wendingding:node wendingding$ touch index.js
```

```
wendingding:node wendingding$ touch index.js
wendingding:node wendingding$ vim index.js
wendingding:node wendingding$ cat index.js
var name = "文顶顶";
var age = 18;
var obj = {
        name:"zs",
        class:"Node",
        show:function(){
                console.log("姓名: "+this.name+" 班级: "+ this.class)
        }
}

//001 测试.load命令
wendingding:node wendingding$ node
> .load ./index.js
var name = "文顶顶";
var age = 18;
var obj = {
        name:"zs",
        class:"Node",
        show:function(){
                console.log("姓名: "+this.name+" 班级: "+ this.class)
        }
}
> obj.name
'zs'
> name
'文顶顶'

//002 测试.help命令
> .help
.break    Sometimes you get stuck, this gets you out
.clear    Alias for .break
.editor   Enter editor mode
.exit     Exit the repl
.help     Print this help message
.load     Load JS from a file into the REPL session
.save     Save all evaluated commands in this REPL session to a file
> index = "我是测试的内容";
'我是测试的内容'

//003 测试.save命令
> .save ./save.js
Session saved to:./save.js

//004 测试.exit命令
> .exit
wendingding:node wendingding$ cat save.js
var name = "文顶顶";
var age = 18;
var obj = {
        name:"zs",
        class:"Node",
        show:function(){
          console.log("姓名: "+this.name+" 班级: "+ this.class)
        }
        }
```

```
obj.name
name
index = "我是测试的内容";
```

- Posted by 博客园·文顶顶 | 花田半亩
- 联系作者 简书·文顶顶 新浪微博·Coder_文顶顶
- 原创文章，版权声明：自由转载-非商用-非衍生-保持署名 | 文顶顶