兄弟连教育

www.lampbrother.net

PHP视频教程

主讲：高洛峰

闭包函数（closures）

闭包函数

- PHP闭包实现主要就是靠匿名函数
- 将匿名函数在普通函数中当做参数传入，也可以被返回。这就实现了一个简单的闭包。
- 通俗的说：子函数可以使用父函数中的局部变量，这种行为就叫做闭包！
- 闭包的两个特点：
  - 1、作为一个函数变量的一个引用 - 当函数返回时，其处于激活状态。
  - 2、一个闭包就是当一个函数返回时，一个没有释放资源的栈区。
  - 其实上面两点可以合成一点,就是闭包函数返回时,该函数内部变量处于激活状态,函数所在栈区依然保留

例一

```
//在函数里定义一个匿名函数，并且调用它
function printStr() {
    $func = function( $str ) {
        echo $str;
    };
    $func( 'some string' );
}

printStr();
```

连接闭包和外界变量的关键字：USE

闭包可以保存所在代码块上下文的一些变量和值。PHP在默认情况下，匿名函数不能调用所在代码块的上下文变量，而需要通过使用use关键字。

例二

```
//在函数中把匿名函数返回，并且调用它
function getPrintStrFunc() {
    $func = function( $str ) {
        echo $str;
    };
    return $func;
}

$printStrFunc = getPrintStrFunc();
$printStrFunc( 'some string' );
```

例三

```
//把匿名函数当做参数传递，并且调用它
function callFunc( $func ) {
    $func( 'some string' );
}

$printStrFunc = function( $str ) {
    echo $str;
};
callFunc( $printStrFunc );

//也可以直接将匿名函数进行传递。这种写法可能会很熟悉（js）
callFunc( function( $str ) {
    echo $str;
} );
```

```
function getMoney() {
    $rmb = 1;
    $dollar = 6;
    $func = function() use ( $rmb ) {
        echo $rmb;
        echo $dollar;
    };
    $func();
}

getMoney();

//输出：
//1
//报错，找不到dorllar变量
```

可以看到，dollar没有在use关键字中声明，在这个匿名函数里也就不能获取到它，所以开发中要注意这个问题。

是否可以在匿名函数中改变上下文的变量---- 否

```
function getMoney() {
    $rmb = 1;
    $func = function() use ( $rmb ) {
        echo $rmb;
        //把$rmb的值加1
        $rmb++;
    };
    $func();
    echo $rmb;
}

getMoney();
//输出：
//1
//1
```

原来use所引用的也只不过是变量的一个副本而已

但是我想要完全引用变量，而不是复制。要达到这种效果，其实在变量前加一个 & 符号就可以了：

```
function getMoney() {
    $rmb = 1;
    $func = function() use ( &$rmb ) {
        echo $rmb;
        //把$rmb的值加1
        $rmb++;
    };
    $func();
    echo $rmb;
}

getMoney();

//输出：
//1
//2
```

如果将匿名函数返回给外界，匿名函数会保存use所引用的变量，而外界则不能得到这些变量，这样形成‘闭包'这个概念可能会更清晰一些。

```
function getMoneyFunc() {
    $rmb = 1;
    $func = function() use ( &$rmb ) {
        echo $rmb;
        //把$rmb的值加1
        $rmb++;
    };
    return $func;
}

$getMoney = getMoneyFunc();
$getMoney();
$getMoney();
$getMoney();

//输出：  1    2   3
```

1，闭包外层是个函数.
2，闭包内部都有函数.
3，闭包会return内部函数.
4，闭包返回的函数内部不能有return.(因为这样就真的结束了)
5，执行闭包后,闭包内部变量会存在,而闭包内部函数的内部变量不会存在.

闭包的应用场景

1、保护函数内的变量安全。以最开始的例子为例，外层函数中变量只有内部函数才能访问，而无法通过其他途径访问到，因此保护了外层函数中变量的安全性。

2、在内存中维持一个变量。依然如前例，由于闭包，外层函数中的变量一直存在于内存中，因此每次执行，都会使用到。

高老师建议:

PHP闭包的特性并没有太大惊喜，其实用 CLASS就可以实现类似甚至强大得多的功能，更不能和js的闭包相提并论，只能期待PHP以后对闭包支持的改进。不过匿名函数还是挺有用的，比如在使用 array_filter()等之类的函数可以不用在外部声明回调函数了。

目前还不稳定，不适用于正式开发。

THANK YOU!

扫描上面的二维码，关注我的新浪微博

http://weibo.com/gaoluofeng