PHP中__autoload()魔术方法详解

本文给大家介绍PHP中__autoload()魔术方法，希望对需要的朋友有所帮助！

__autoload()，尝试加载未定义的类

作用：

你可以通过定义这个函数来启用类的自动加载。

在魔术函数 __autoload() 方法出现以前，如果你要在一个程序文件中实例化100个对象，那么你必须用include或者require包含进来100个类文件，或者你把这100个类定义在同一个类文件中 —— 相信这个文件一定会非常大，然后你就痛苦了。

但是有了 __autoload() 方法，以后就不必为此大伤脑筋了，这个类会在你实例化对象之前自动加载制定的文件。

还是通过例子来看看吧：

先看看以往的方式：

```
/**
 * 文件non_autoload.php
 */

require_once('project/class/A.php');
require_once('project/class/B.php');
require_once('project/class/C.php');

if (条件A) {
    $a = new A();
    $b = new B();
    $c = new C();
    // … 业务逻辑
} else if (条件B) {
    $a = newA();
    $b = new B();
    // … 业务逻辑
}
```

```
18
```

看到了吗？不用100个，只是3个看起来就有点烦了。而且这样就会有一个问题：如果脚本执行“条件B”这个分支时，C.php这个文件其实没有必要包含。因为，任何一个被包含的文件，无论是否使用，均会被php引擎编译。

如果不使用，却被编译，这样可以被视作一种资源浪费。更进一步，如果C.php包含了D.php，D.php包含了E.php。并且大部分情况都执行“条件B”分支，那么就会浪费一部分资源去编译C.php,D.php,E.php三个“无用”的文件。

那么如果使用 __autoload() 方式呢？

```php
/**
 * 文件autoload_demo.php
 */
function  __autoload($className) {
    $filePath = “project/class/{$className}.php”;
    if (is_readable($filePath)) {
        require($filePath);
    }
}

if (条件A) {
    $a = new A();
    $b = new B();
    $c = new C();
    // … 业务逻辑
} else if (条件B) {
    $a = newA();
    $b = new B();
    // … 业务逻辑
}
```

ok,不论效率怎么用，最起码界面看起来舒服多了，没有太多冗余的代。

再来看看这里的效率如何，我们分析下：

当php引擎第一次使用类A，但是找不到时，会自动调用 __autoload 方法，并将类名“A”作为参数传入。所以，我们在 __autoload() 中需要的做的就是根据类名，找到相应的文件，并包含进来，如果我们的方法也找不到，那么php引擎就会报错了。

注意：

这里可以只用require，因为一旦包含进来后，php引擎再遇到类A时，将不会调用__autoload，而是直接使用内存中的类A，不会导致多次包含。

扩展：

其实php发展到今天，已经有将 `spl_autoload_register` — 注册给定的函数作为 __autoload 的实现了，但是这个不在啊本文讲解之内，有兴趣可以自行看手册。

推荐PHP实战教程：https://www.php.cn/k.html

以上就是PHP中__autoload()魔术方法详解的详细内容，更多请关注php中文网其它相关文章！