iranphp articles

عنوان مقاله : الگويكتا و بكارگيري آن در php5
نگارنده : آرش ميكائيلي
آدرس پست الكترونيك : arash@iranphp.net
تاريخ نگارش : .....

در این مقاله به شرح الگوی یکتا (Singleton Pattern) و نحوه بکار گیری آن در PHP5 و همچنین بعضی از نکات و روشهای نوشتن یه کد خوب اشاره خواهد شد . مخاطبین این مقاله برنامه نویسانی هستند که مایلند از تکنیکهای برنامه نویسی شی گرا در کارهاشون استفاده کنند و البته نیاز دارن که با مبانی OOP در PHP5 آشنا باشند .
تمام کدهای این مقاله بر روی PHP 5.5-MySQL 4.1.14-Apache 2.0.54 و سیستم عامل FC4 تست شده است .

الگوی یکتا چیست ؟

پیش از اینکه به توضیح در مورد این الگوی خاص بپردازم ،توضیح مختصری راجع به Design Patterns می دم تا با ایده اصلی کمی آشنا بشین Design Patterns روشی برای برنامه نویساست تا از دوباره کاری ها اجتناب بشه و از دانش و مهارت افراد دیگه توی مسائلی که به صورت معمول پیش می آد و یا شبیه به هم هستند استفاده کنن و همچنین دانش و مهارت خودشون را با بقیه سهیم بشن .
هر الگو دستورالعمل انجام کار خاصیه بصورتی که موثر بودن آن به اثبات رسیده .

الگوی یکتا :

در خیلی از موارد بهتره که ما فقط یک instance از یک کلاس در کل اسکریپت داشته باشیم و این معمولا وقتی پیش می آد که شی مورد نظر یه component منحصر به فرد را در سیستم مشخص می کنن و یا اینکه بخوان یک نمونه (instance) را بین اجزا مختلف برنامه به اشتراک بذارن .مثلا وقتی که ما از دیتابیس تو برنامون استفاده می کنیم دیتابیس می تونه یه نمونه (instance) از کلاس DB باشه که تو جاهای مختلف برنامه ازش استفاده می کنیم. به عبارت دیگه با استفاده از این الگو می تونیم از تولید اشیاء تکراری جلوگیری کنیم و در نتیجه در استفاده از حافظه و زمانی که برای ساختن شی جدید می شه صرفه جویی می کنیم .
کلاسهایی که می خوایم در قالب این الگو باشند با ید شرایط زیر را داشته باشن :
هیچ راهی برای تولید نمونه دیگری از شئ مورد نظر نباید وجود داشته باشد .
تا زمانی که نیاز نیست نباید ساخته شود .
پیدا کردنش نیز باید راحت باشه .
توضیح مفصل این موارد خیلی طول می کشه و خسته کننده می شه در نتیجه فقط به ذکر عناوین بسنده می کنم و پیاده سازی یه مثال سعی می کنم منظورم را برسونم .

پیاده سازی الگوی یکتا در : PHP5

خوب با توجه به سه موردی که در بالا به اونا اشاره کردم ما باید مطمئن بشیم که راهی برای تولید شئ خارج از مداری که ما می خوایم وجود نداره پس ساختن (instance)گرفتن را ممنوع می کنم ؟! فقط کافیه که تابع سازنده را private کنیم اینجوری فقط یکی از متدهای خود کلاس می تونه نمونه جدید بسازه .

```
private function __construct($dbHost,$dbUser,$dbPass,$dbName) {
//do something
}
```

پس به یه متد نیاز داریم که بدون اینکه نیاز باشه نمونه ای از کلاس ساخته بشه اون را اجرا کنیم؟! برای این کار استفاده از یه متود static و با دسترسی public بهترین راه حله .

```
static function getInstance(){
        //create new instance in first call
        if (self::$instance==null) {
            new class();
        }
        return self::$instance;
    }
```

حالا فقط مونده که بتونیم جلوی تولید چند شی با استفاده از این متود را بگیریم به عبارت دیگه هروقت که این متود صدا می شه باید بتونه تشخیص بده که آیا قبلا صدا شده یا نه! اگه دفعه اول بود یه شی جدید بسازه و اگه قبلا ساخته بود بجای اینکه دوباره یه نمونه بسازه همون قبلی را برگردونه .برای این کار هم از یه ویژگی static (property) با دسترسی private استفاده می کنیم تا بتونیم همیشه مقدارش را چک کنیم و در عین حال امکان تغییر این مقدار هم خارج از کلاس وجود نداشته باشه .

```
static private $instance=null;
```

حالا فقط می مونه که جلوی clone شدن شی را هم بگیریم که اونم از طریق تعریف متود __clone() با دسترسی private حله .

```
private function __clone()
 {
 }
```

یک مثال واقعی از الگوی یکتا :

در این مثال نحوه ایجاد یک شی از کلاس MySQLi نشان داده می شود :

```
<?PHP
/**
* PHP version 5
*
* This source file is subject to version 2.1 of the GNU Lesser General Public
* License, that is bundled with this package in the file COPYING, available
* through the world wide web at the following URI:
* http://www.gnu.org/copyleft/lesser.html.
*
* <i>smysqli class </i> Singelton Improved Mysql<br />
* this is a very simple wrapper class that uses singleton pattern to avoid creating multiple links
* in a single application to mysql database<br />
*
* an example of how to use this class :
* <code>
* //creating DB object
* $dbObject=smysqli::getInstance('localhost','arash','arash','usersdb');
* //sending query to DB
* $dbObject->query('select * from tblUsers');
* //using object property
* echo $dbObject->affected_rows;
* </code>
* @package DataBase
* @author Arash Mikaeili <arash@iranphp.net>
* @copyright  2005 iranphp.net
* @license    LGPL License 2.1 <http://www.gnu.org/copyleft/lesser.html>
* @version    SVN $Id$
* @link       http://www.iranphp.net
*/
class smysqli {
/**
* Private constructor to avoid <i>smysqli</i> object creation.
*
```

```
* @access private
* @param string $dbHost the mysql hostname or ip
* @param string $dbUser
* @param string $dbPass
* @param string $dbName the DataBase we want to use
* @return void
* @version     SVN $Id$
*/
    private function __construct($dbHost,$dbUser,$dbPass,$dbName) {
        self::$instance=new mysqli($dbHost,$dbUser,$dbPass,$dbName) ;
        if (mysqli_connect_errno()) {
            throw new Exception('connect failed: '.mysqli_connect_error());
            exit();
        }
    }
/**
* Private clonator to avoid <i>smysqli</i> object clonation.
*
* @access private
*/
    private function __clone()
    {
    }
/**
* Create new mysqli object or just return the already existed one
*
* @static
* @access public
* @param string $dbHost the mysql hostname or ip
* @param string $dbUser
* @param string $dbPass
* @param string $dbName the DataBase we want to use
* @return object mysqli
* @version    SVN $Id$
*/
    static function getInstance($dbHost,$dbUser,$dbPass,$dbName){
        //create new smysqli object in first call
        if (self::$instance==null) {
            new smysqli($dbHost,$dbUser,$dbPass,$dbName);
        }
        return self::$instance;
    }
/**
* @var object mysqli
* @access private
* @static
*/
    static private $instance=null;
}
?>
```

با توجه به توضیحاتی که در بالا دادم و حاشیه نویسی که در خود کد وجود داره دیگه نیازی به شرح خط به خط کد نیست تنها کاری که این کلاس انجام می ده اینه که یه شی یکتا از کلاس mysqli بر می گردنه .

در انتها دوست دارم که چند نکته را یاد آوری کنم که شاید خیلی هم به بحث این مقاله مربوط نباشه :
تا جایی که می تونید تو کدتون از Comment استفاده کنید .این کار باعث می شه که هم خودتون بعدا راحتتر از کدی که نوشتین سر در بیارین و هم اینکه اگه یه روزی خواستین این کد را بصورت بازمتن منتشر کنید محبوبیت بیشتری پیدا کنه .

Refactoringرا فراموش نکنید بطور کلی هر وقت که مشغول برنامه نویسی هستین یا باید کد جدید بنویسید یا اینکه کدهای قبلیتون را بازسازی و بهینه کنید .مثلا همین مثال بالا اگه بجای اینکه مشخصات DB را بصورت پارامتر بگیره اونا بصورت ثابت تو خود کلاس باشن بهتره هم باعث می شه که استفاده از کلاس راحتتر بشه و هم اینکه اگه یه روزی خواستین تعداد این پارامترها را تغییر بدین مثلا بجای connectتنها از init هم استفاده کنین، دیگه لازم نیست کدتون را در تمام برنامه تغییر بدین فقط کافیه تغییرات را در همین کلاس انجام بدین خود به خود همه جا اعمال می شه .

حتما از CVS یا SUBVERSION استفاده کنید .حتی اگه می خواین یه کلاس کوچیک یا یه تابع بنویسید و تنها برنامه نویس هم خود شما هستین. تا استفاده نکنین اهمیتش را متوجه نمی شین .

پ.ن. در صورتی که هر گونه سوال یا بحثی در مورد مطالب با دارید می تونید اون را در قسمت انجمن سایت مطرح کنید .