1. <?php
2. // +----------------------------------------------------------------------
3. // | ThinkPHP [ WE CAN DO IT JUST THINK IT ]
4. // +----------------------------------------------------------------------
5. // | Copyright (c) 2006-2012 http://thinkphp.cn All rights reserved.
6. // +----------------------------------------------------------------------
7. // | Licensed ( http://www.apache.org/licenses/LICENSE-2.0 )
8. // +----------------------------------------------------------------------
9. // | Author: liu21st <liu21st@gmail.com>
10. // +----------------------------------------------------------------------
12. /\*\*
13. \* ThinkPHP 运行时文件 编译后不再加载
14. \* @category   Think
15. \* @package  Common
16. \* @author   liu21st <liu21st@gmail.com>
17. \*/
18. //此文件只在第一次运行入口文件时会执行，如果不是第一次运行入口文件，则不会再执行此文件
19. //判断THNK\_PATH是否已经被定义，如果没被定义，则终止程序
20. defined('THINK\_PATH') **or** exit();
21. //判断当前php的版本是否大于5.2.0
22. **if**(version\_compare(PHP\_VERSION,'5.2.0','<'))  **die**('require PHP > 5.2.0 !');
24. //  版本信息
25. define('THINK\_VERSION', '3.1');
27. //   系统信息
28. **if**(version\_compare(PHP\_VERSION,'5.3.0','<')) {
29. set\_magic\_quotes\_runtime(0);//是否自动在某些特殊字符前面加上反斜线
30. define('MAGIC\_QUOTES\_GPC',get\_magic\_quotes\_gpc()?True:False);
31. }**else**{
32. define('MAGIC\_QUOTES\_GPC',True);
33. }
34. define('IS\_CGI',substr(PHP\_SAPI, 0,3)=='cgi' ? 1 : 0 );
35. define('IS\_WIN',strstr(PHP\_OS, 'WIN') ? 1 : 0 );
36. define('IS\_CLI',PHP\_SAPI=='cli'? 1   :   0);
38. // 项目名称，如果在入口文件中没有定义，则会将根目录作为项目的名称
39. defined('APP\_NAME') **or** define('APP\_NAME', basename(dirname($\_SERVER['SCRIPT\_FILENAME'])));
40. **if**(!IS\_CLI) {//默认值为0
41. // 当前文件名
42. **if**(!defined('\_PHP\_FILE\_')) {
43. **if**(IS\_CGI) {
44. //CGI/FASTCGI模式下
45. $\_temp  = explode('.php',$\_SERVER['PHP\_SELF']);
46. define('\_PHP\_FILE\_',    rtrim(str\_replace($\_SERVER['HTTP\_HOST'],'',$\_temp[0].'.php'),'/'));
47. }**else** {
48. //定义当前执行脚本的名称，如：/index.php    ，rtrim函数第二个参数：删除末端的'/'
49. define('\_PHP\_FILE\_',    rtrim($\_SERVER['SCRIPT\_NAME'],'/'));
50. }
51. }
52. **if**(!defined('\_\_ROOT\_\_')) {
53. // 网站URL根目录
54. **if**( strtoupper(APP\_NAME) == strtoupper(basename(dirname(\_PHP\_FILE\_))) ) {
55. $\_root = dirname(dirname(\_PHP\_FILE\_));
56. }**else** {//执行这里
57. $\_root = dirname(\_PHP\_FILE\_);
58. }
59. //定义网站的根目录
60. define('\_\_ROOT\_\_',   (($\_root=='/' || $\_root=='\\')?'':$\_root));
61. }
63. //支持的URL模式
64. define('URL\_COMMON',      0);   //普通模式
65. define('URL\_PATHINFO',    1);   //PATHINFO模式
66. define('URL\_REWRITE',     2);   //REWRITE模式
67. define('URL\_COMPAT',      3);   // 兼容模式
68. }
70. // 路径设置 可在入口文件中重新定义 所有路径常量都必须以/ 结尾
71. defined('CORE\_PATH')    **or** define('CORE\_PATH',      THINK\_PATH.'Lib/'); // 系统核心类库目录
72. defined('EXTEND\_PATH')  **or** define('EXTEND\_PATH',    THINK\_PATH.'Extend/'); // 系统扩展目录
73. defined('MODE\_PATH')    **or** define('MODE\_PATH',      EXTEND\_PATH.'Mode/'); // 系统扩展目录/模式扩展目录
74. defined('ENGINE\_PATH')  **or** define('ENGINE\_PATH',    EXTEND\_PATH.'Engine/'); // 系统扩展目录/引擎扩展目录
75. defined('VENDOR\_PATH')  **or** define('VENDOR\_PATH',    EXTEND\_PATH.'Vendor/'); // 系统扩展目录/第三方类库目录
76. defined('LIBRARY\_PATH') **or** define('LIBRARY\_PATH',   EXTEND\_PATH.'Library/'); // 系统扩展目录/扩展类库目录
77. defined('COMMON\_PATH')  **or** define('COMMON\_PATH',    APP\_PATH.'Common/'); // 项目公共目录
78. defined('LIB\_PATH')     **or** define('LIB\_PATH',       APP\_PATH.'Lib/'); // 项目类库目录
79. defined('CONF\_PATH')    **or** define('CONF\_PATH',      APP\_PATH.'Conf/'); // 项目配置目录
80. defined('LANG\_PATH')    **or** define('LANG\_PATH',      APP\_PATH.'Lang/'); // 项目语言包目录
81. defined('TMPL\_PATH')    **or** define('TMPL\_PATH',      APP\_PATH.'Tpl/'); // 项目模板目录
82. defined('HTML\_PATH')    **or** define('HTML\_PATH',      APP\_PATH.'Html/'); // 项目静态目录
83. defined('LOG\_PATH')     **or** define('LOG\_PATH',       RUNTIME\_PATH.'Logs/'); // 项目路径/runtime/项目日志目录
84. defined('TEMP\_PATH')    **or** define('TEMP\_PATH',      RUNTIME\_PATH.'Temp/'); // 项目路径/runtime/项目缓存目录
85. defined('DATA\_PATH')    **or** define('DATA\_PATH',      RUNTIME\_PATH.'Data/'); // 项目路径/runtime/项目数据目录
86. defined('CACHE\_PATH')   **or** define('CACHE\_PATH',     RUNTIME\_PATH.'Cache/'); // 项目路径/runtime/项目模板缓存目录
88. // 为了方便导入第三方类库 设置Vendor目录到include\_path
89. set\_include\_path(get\_include\_path() . PATH\_SEPARATOR . VENDOR\_PATH);//将第三方类库的路径加入到系统变量中
91. // 加载运行时所需要的文件 并负责自动目录生成
92. **function** load\_runtime\_file() {
93. // 加载系统基础函数库，D()函数、C()函数 、import()函数、vendor()函数、alias\_import()、require\_cache ()函数 、alias\_import()函数 等都在此文件中定义
94. **require** THINK\_PATH.'Common/common.php';//路径：ThinkPHP/Common/common.php
95. // 读取核心编译文件列表，//路径：ThinkPHP/Lib/Think.class.php
96. $list = **array**(
97. CORE\_PATH.'Core/Think.class.php',           // 这个文件很重要，主要用来读取配置文件，编译项目等等
98. CORE\_PATH.'Core/ThinkException.class.php',  // 异常处理类
99. CORE\_PATH.'Core/Behavior.class.php',        // 行为类
100. );
101. // 加载模式文件列表
102. **foreach** ($list **as** $key=>$file){
103. //require\_cache()函数会将成功加载的文件置入静态数组变量$\_importFiles[$filename]中
104. **if**(is\_file($file))  require\_cache($file);//加载核心编译文件：require\_cache函数在common.php文件中定义
105. }
107. // 加载系统类库别名定义，一个别名对应一个核心类库，如：【别名】'Db'=>【核心类库】 CORE\_PATH.'Core/Db.class.php',
108. /\*\*
109. \* 大概有如下几个别名
110. \* 'Model'         => CORE\_PATH.'Core/Model.class.php',
111. \* 'Db'            => CORE\_PATH.'Core/Db.class.php',
112. \* 'Log'            => CORE\_PATH.'Core/Log.class.php',
113. \* 'ThinkTemplate' => CORE\_PATH.'Template/ThinkTemplate.class.php',
114. \* 'TagLib'        => CORE\_PATH.'Template/TagLib.class.php',
115. \* 'Cache'         => CORE\_PATH.'Core/Cache.class.php',
116. \* 'Widget'        => CORE\_PATH.'Core/Widget.class.php',
117. \* 'TagLibCx'      => CORE\_PATH.'Driver/TagLib/TagLibCx.class.php',
118. \*/
119. alias\_import(**include** THINK\_PATH.'Conf/alias.php');
121. // 检查项目目录结构 如果不存在则自动创建
122. **if**(!is\_dir(LIB\_PATH)) {
123. // 如果不是目录，则创建项目目录结构
124. build\_app\_dir();
125. }**elseif**(!is\_dir(CACHE\_PATH)){
126. // 如果不是目录，则检查缓存目录，并创建缓存目录
127. check\_runtime();
128. }**elseif**(APP\_DEBUG){//在调试模式开启的情况下，在运行时会删除编译文件
129. // 调试模式切换删除编译缓存
130. **if**(is\_file(RUNTIME\_FILE))   unlink(RUNTIME\_FILE);
131. }
132. }
134. // 检查缓存目录(Runtime) 如果不存在则自动创建
135. **function** check\_runtime() {
136. **if**(!is\_dir(RUNTIME\_PATH)) {
137. //如果runtime目录不存在，则创建runtime目录
138. mkdir(RUNTIME\_PATH);
139. }**elseif**(!is\_writeable(RUNTIME\_PATH)) {
140. //如果runtime目录不可写，则给出提示
141. header('Content-Type:text/html; charset=utf-8');
142. exit('目录 [ '.RUNTIME\_PATH.' ] 不可写！');
143. }
144. mkdir(CACHE\_PATH);  // 创建项目模板缓存目录
145. **if**(!is\_dir(LOG\_PATH))   mkdir(LOG\_PATH);    // 日志目录
146. **if**(!is\_dir(TEMP\_PATH))  mkdir(TEMP\_PATH);   // 数据缓存目录
147. **if**(!is\_dir(DATA\_PATH))  mkdir(DATA\_PATH);   // 数据文件目录
148. **return** true;
149. }
151. // 在调试模式开启的情况下，不会执行此函数
152. // 创建编译缓存，此函数会将所有要编译的内容编译到~runtime.php文件中，此文件中没有php的结束标记
153. **function** build\_runtime\_cache($append='') {
154. // 生成编译文件
155. $defs           = get\_defined\_constants(TRUE);//返回当前定义的所有常量组成的数组，可通过输出$defs查看效果
156. $content        =  '$GLOBALS[\'\_beginTime\'] = microtime(TRUE);';
157. **if**(defined('RUNTIME\_DEF\_FILE')) { // 编译后的常量文件外部引入
158. file\_put\_contents(RUNTIME\_DEF\_FILE,'<?php '.array\_define($defs['user']));
159. $content   .=  'require \''.RUNTIME\_DEF\_FILE.'\';';
160. }**else**{
161. //array\_define():根据数组生成常量的定义
162. $content   .= array\_define($defs['user']);//只有用户定义的常量
163. }
165. //第三方类库写入到环境变量中
166. $content       .= 'set\_include\_path(get\_include\_path() . PATH\_SEPARATOR . VENDOR\_PATH);';
168. // 读取核心编译文件列表
169. $list = **array**(
170. THINK\_PATH.'Common/common.php',
171. CORE\_PATH.'Core/Think.class.php',
172. CORE\_PATH.'Core/ThinkException.class.php',
173. CORE\_PATH.'Core/Behavior.class.php',
174. );
176. **foreach** ($list **as** $file){
177. $content .= compile($file);//循环将这些文件中的内容拼接到$content后面【注意：每个文件的内容都去除了php标记】
178. }
180. // 系统行为扩展文件统一编译
181. **if**(C('APP\_TAGS\_ON')) {                                  //这里值为false
182. $content .= build\_tags\_cache();
183. }
185. //包含别名文件，别名文件返回的数组数据赋值给$alias
186. $alias      = **include** THINK\_PATH.'Conf/alias.php';
187. $content   .= 'alias\_import('.var\_export($alias,true).');';
188. // 编译框架默认语言包和配置参数
189. $content   .= $append."\nL(".var\_export(L(),true).");C(".var\_export(C(),true).');G(\'loadTime\');Think::Start();';
190. file\_put\_contents(RUNTIME\_FILE,strip\_whitespace('<?php '.$content));
191. }
193. // 编译系统行为扩展类库
194. **function** build\_tags\_cache() {
195. $tags = C('extends');//跟ThinkPHP/Conf/tags.php文件有关，可参考
196. $content = '';
197. **foreach** ($tags **as** $tag=>$item){
198. **foreach** ($item **as** $key=>$name) {//也挺重要的，记得看一下下
199. $content .= is\_int($key)?compile(CORE\_PATH.'Behavior/'.$name.'Behavior.class.php'):compile($name);
200. }
201. }
202. **return** $content;
203. }
205. // 创建项目目录结构
206. **function** build\_app\_dir() {
207. // 没有创建项目目录的话自动创建
208. **if**(!is\_dir(APP\_PATH)) mkdir(APP\_PATH,0777,true);
209. **if**(is\_writeable(APP\_PATH)) {
210. $dirs  = **array**(
211. LIB\_PATH,
212. RUNTIME\_PATH,
213. CONF\_PATH,
214. COMMON\_PATH,
215. LANG\_PATH,
216. CACHE\_PATH,
217. TMPL\_PATH,      //项目的模板目录，即：Tpl
218. TMPL\_PATH.C('DEFAULT\_THEME').'/',       //项目模板目录下默认风格目录
219. LOG\_PATH,
220. TEMP\_PATH,
221. DATA\_PATH,
222. LIB\_PATH.'Model/',                      //项目的模型目录
223. LIB\_PATH.'Action/',                     //项目的控制器目录
224. LIB\_PATH.'Behavior/',
225. LIB\_PATH.'Widget/',
226. );
227. **foreach** ($dirs **as** $dir){
228. **if**(!is\_dir($dir))  mkdir($dir,0777,true);//循环创建项目的各个目录
229. }
230. // 写入目录安全文件
231. build\_dir\_secure($dirs);
232. // 写入初始配置文件
233. **if**(!is\_file(CONF\_PATH.'config.php'))        // 配置文件
234. file\_put\_contents(CONF\_PATH.'config.php',"<?php\nreturn array(\n\t//'配置项'=>'配置值'\n);\n?>");
235. // 写入测试Action                           // 自动生成一个默认的控制器测试文件
236. **if**(!is\_file(LIB\_PATH.'Action/IndexAction.class.php'))
237. build\_first\_action();                   //如果没有IndexAction.class.php文件，则生成一个测试控制器文件
238. }**else**{
239. header('Content-Type:text/html; charset=utf-8');
240. exit('项目目录不可写，目录无法自动生成！<BR>请使用项目生成器或者手动生成项目目录~');
241. }
242. }
244. // 创建测试Action
245. **function** build\_first\_action() {                     //默认控制器文件中的内容
246. $content = file\_get\_contents(THINK\_PATH.'Tpl/default\_index.tpl');
247. file\_put\_contents(LIB\_PATH.'Action/IndexAction.class.php',$content);
248. }
250. // 生成目录安全文件
251. **function** build\_dir\_secure($dirs='') {
252. // 目录安全写入
253. **if**(defined('BUILD\_DIR\_SECURE') && BUILD\_DIR\_SECURE) {
254. defined('DIR\_SECURE\_FILENAME')  **or** define('DIR\_SECURE\_FILENAME',    'index.html');
255. defined('DIR\_SECURE\_CONTENT')   **or** define('DIR\_SECURE\_CONTENT',     ' ');
256. // 自动写入目录安全文件
257. $content = DIR\_SECURE\_CONTENT;
258. $files = explode(',', DIR\_SECURE\_FILENAME);
259. **foreach** ($files **as** $filename){
260. **foreach** ($dirs **as** $dir)
261. file\_put\_contents($dir.$filename,$content);
262. }
263. }
264. }
266. // 加载运行时所需文件
267. load\_runtime\_file();
268. // 记录加载文件时间
269. G('loadTime');
270. // 执行入口
271. Think::Start();