Bugfree数据交互流程

目的:了解从输入网址到页面显示过程中服务器的调用过程,以便方便二次开发

客户端浏览器输入网址:URL http://localhost/bugfree/index.php到显示页面拆解

1. 用户向入口脚本index.php 发起请求。
2. 入口脚本加载应用配置 并创建一个应用实例去处理请求。
3. 应用通过请求组件解析请求的路由。
4. 应用创建一个控制器实例去处理请求。
5. 控制器创建一个操作实例并针对操作执行过滤器。
6. 如果任何一个过滤器返回失败，则操作退出。
7. 如果所有过滤器都通过，操作将被执行。
8. 操作会加载一个数据模型，或许是来自数据库。
9. 操作会渲染一个视图，把数据模型提供给它。
10. 渲染结果返回给响应组件。
11. 响应组件发送渲染结果给用户浏览器。

index.php:

```
1  <?php
2
3  // change the following paths if necessary
4  $yii=dirname(__FILE__).'/lib/yii.php';
5  $config=dirname(__FILE__).'/protected/config/main.php';
6
7  // remove the following lines when in production mode
8  defined('YII_DEBUG') or define('YII_DEBUG',false);
9  // specify how many levels of call stack should be shown in each log message
10 defined('YII_TRACE_LEVEL') or define('YII_TRACE_LEVEL',3);
11 require_once($yii);
12 Yii::createWebApplication($config)->run();
```

第4行定义控制台应用的入口脚本目录,然后在第11行进行加载

第5行获取配置文件

第12行使用Yii类创建对象并运行

Yii::createWebApplication($config)-->

C:\xampp\htdocs\bugfree\lib\YiiBase.php

```
98      public static function createWebApplication($config=null)
99      {
100         return self::createApplication('CWebApplication',$config);
101     }
```

-->

```
126     public static function createApplication($class,$config=null)
127     {
128         return new $class($config);
129     }
```

相当于返回一个CWebApplication($config)对象

CWebApplication类存在C:\xampp\htdocs\bugfree\lib\web\CWebApplication.php中定义

注意:CWebApplication中有2个默认变量

public $defaultController='site';

public $layout='main';

由于CWebApplication中没有找到构造函数,因此调用父类的构造函数进行初始

```
60 class CWebApplication extends CApplication
61 {
```

C:\xampp\htdocs\bugfree\lib\base\CApplication.php

构造函数如下

```
117     public function __construct($config=null)
118     {
119         Yii::setApplication($this);
120 
121         // set basePath at early as possible to avoid trouble
122         if(is_string($config))
123             $config=require($config);
124         if(isset($config['basePath']))
125         {
126             $this->setBasePath($config['basePath']);
127             unset($config['basePath']);
128         }
129         else
130             $this->setBasePath('protected');
131         Yii::setPathOfAlias('application',$this->getBasePath());
132         Yii::setPathOfAlias('webroot',dirname($_SERVER['SCRIPT_FILENAME']));
133         Yii::setPathOfAlias('ext',$this->getBasePath().DIRECTORY_SEPARATOR.'extensions');
134 
135         $this->preinit();
136 
137         $this->initSystemHandlers();
138         $this->registerCoreComponents();
139 
140         $this->configure($config);
141         $this->attachBehaviors($this->behaviors);
142         $this->preloadComponents();
143 
144         $this->init();
145     }
```

119行:把该类$this赋值给Yii::$_app 到时候使用Yii::$_app即可调用cApplication类里面的函数和成员

122-130行 设置 web页面的根目录 这里在配置文文件$config中决定

C:\\xampp\\htdocs\\bugfree\\protected\\config\\.. 注意最后..表示父目录,即根目录为protected

131-133 设置路径别名

application=C:\\xampp\\htdocs\\bugfree\\protected

webroot=C:/xampp/htdocs/bugfree (这里即为index.php的目录)

ext=C:\\xampp\\htdocs\\bugfree\\protected\\extensions (扩展组件目录)

第135行 preinit 初始化准备工作

调用父类CModulepreinit(实际什么都没做)

C:\xampp\htdocs\bugfree\lib\base\CModule.php

137行加载异常和错误处理需要用到的类

其中常量YII_ENABLE_EXCEPTION_HANDLER YII_ENABLE_ERROR_HANDLER均为1

```
916     /**
917      * Initializes the class autoloader and error handlers.
918      */
919     protected function initSystemHandlers()
920     {
921         if(YII_ENABLE_EXCEPTION_HANDLER)
922             set_exception_handler(array($this,'handleException'));
923         if(YII_ENABLE_ERROR_HANDLER)
924             set_error_handler(array($this,'handleError'),error_reporting());
925     }
926 
```

138行 加载需要用到的组件信息,方便后面直接调用

```php
    protected function registerCoreComponents()
    {
        $components=array(
            'coreMessages'=>array(
                'class'=>'CPhpMessageSource',
                'language'=>'en_us',
                'basePath'=>YII_PATH.DIRECTORY_SEPARATOR.'messages',
            ),
            'db'=>array(
                'class'=>'CDbConnection',
            ),
            'messages'=>array(
                'class'=>'CPhpMessageSource',
            ),
            'errorHandler'=>array(
                'class'=>'CErrorHandler',
            ),
            'securityManager'=>array(
                'class'=>'CSecurityManager',
            ),
            'statePersister'=>array(
                'class'=>'CStatePersister',
            ),
            'urlManager'=>array(
                'class'=>'CUrlManager',
            ),
            'request'=>array(
                'class'=>'CHttpRequest',
            ),
            'format'=>array(
                'class'=>'CFormatter',
            ),
        );

        $this->setComponents($components);
    }
```

语言选择

数据库类名

URL路由管理

HTTP请求需要用到的类

显示格式

965行调用C:\xampp\htdocs\bugfree\lib\base\CModule.php中的设置组件信息函数

140行 加载配置

使用CModule中的configure函数进行加载赋值,只是把数组的value赋值给$this->$key

```php
    /**
     * Configures the module with the specified configuration.
     * @param array $config the configuration array
     */
    public function configure($config)
    {
        if(is_array($config))
        {
            foreach($config as $key=>$value)
                $this->$key=$value;
        }
    }
```

141 调用CModule父类CComponent中的函数C:\xampp\htdocs\bugfree\lib\base\CComponent.php 初始时$this->behaviors为空数组

142 preloadComponents加载提前需要加载的组件 这里初始化登录页面只有一个Array([0] => log)

143行 初始化,会获取所有的request信息

```
CHttpRequest::__set_state(array(
   'enableCookieValidation' => true,
   'enableCsrfValidation' => false,
   'csrfTokenName' => 'YII_CSRF_TOKEN',
   'csrfCookie' => NULL,
   '_requestUri' => NULL,
   '_pathInfo' => NULL,
   '_scriptFile' => NULL,
   '_scriptUrl' => NULL,
   '_hostInfo' => NULL,
   '_baseUrl' => NULL,
   '_cookies' => NULL,
   '_preferredLanguage' => NULL,
   '_csrfToken' => NULL,
   '_deleteParams' => NULL,
   '_putParams' => NULL,
   '_port' => NULL,
   '_securePort' => NULL,
   'behaviors' =>
  array (
  ),
   '_initialized' => true,
   '_e' => NULL,
   '_m' => NULL,
))
```

然后回到index.php中的run函数

调用C:\xampp\htdocs\bugfree\lib\base\CApplication.php中的run函数

```
    public function run()
    {

        if($this->hasEventHandler('onBeginRequest'))
            $this->onBeginRequest(new CEvent($this));
        $this->processRequest();
        if($this->hasEventHandler('onEndRequest'))
            $this->onEndRequest(new CEvent($this));
    }
```

关键159行processRequest函数为类CWebApplication的函数C:\xampp\htdocs\bugfree\lib\web\CWebApplication.php

主要得到$_GET请求数组 以及路由部分的值，初始化登录时$route值site/login

```
    /**
     * Processes the current request.
     * It first resolves the request into controller and action,
     * and then creates the controller to perform the action.
     */
    public function processRequest()
    {

        if(is_array($this->catchAllRequest) && isset($this->catchAllRequest[0]))
        {
            $route=$this->catchAllRequest[0];
            foreach(array_splice($this->catchAllRequest,1) as $name=>$value)
                $_GET[$name]=$value;
        }
        else
            $route=$this->getUrlManager()->parseUrl($this->getRequest());
        $this->runController($route);
    }
```

r=site/login

r=site表示调用的controller/SiteController.php的控制器

还表示在视图中使用的是site里面的内容

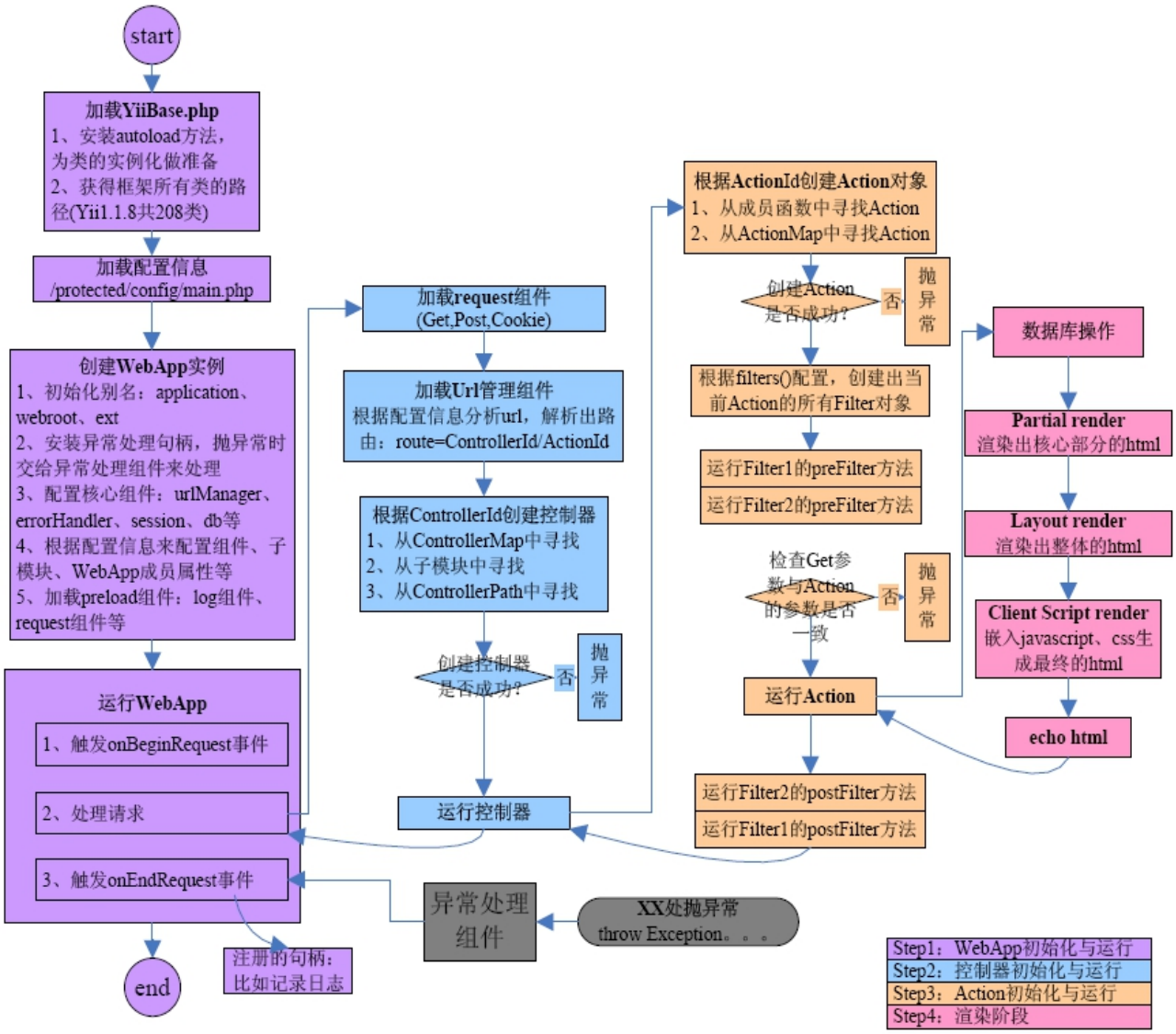
login表示的是SiteController.php里面的actionLogin()函数

login函数里面最后一句render渲染

$this->render('login', array('model' => $model));(这里的这个login表示到时候调用视图site中的为login.php文件)

这样就看见登录页面了

具体流程如下图所示:



http://localhost/bugfree/index.php?r=info/index&type=bug&product_id=383

r=info/index比如这句表示的controller/infoController.php view视图使用的是info/index.php models使用的是models/info.php模板

实际上r=info中的info M 表示模板中的info V表示views下的info文件夹 C表示controller/infoController控制器以及里面的action,即

使用一个变量将MVC关联起来

index表示调用的是infoController中的actionIndex函数

```
-- select severity,count(*) from bf_bugview WHERE product_id = 440 GROUP BY severity;
-- select module_name,count(*) from bf_bugview WHERE product_id = 440 AND severity = 1 GROUP BY module_name;
```