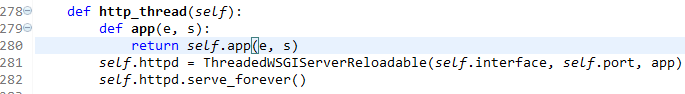
我们以http.py的Root类为分界点往前后看，首先Root类会在系统加载的时候把自己注册到wsgi\_server.py的module\_handlers中，这个列表会被wsgi\_server.py的application方法间接调用到，application方法会在系统启动的时候作为回调函数加入到ThreadedWSGIServerReloadable中，也就是说当请求到达的时候，系统会调用application方法，如下图是openerp/service/server.py中的代码片段：



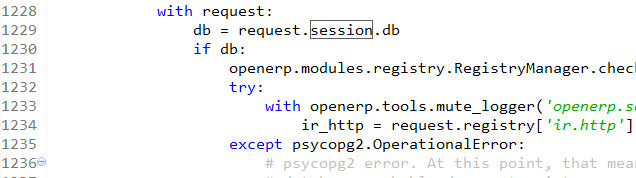
其中self.app就是wsgi\_server.py的application方法。

再回到Root类，Root类是一个自调用类，它的\_\_call\_\_方法会先检查组件是否已经被加载，没有加载则调用load\_addons方法加载，需要注意的是在load\_addons方法中，会把static下的文件路径添加到一个statics列表中，这些资源在被请求的时候odoo服务端不会对资源内容进行处理，而是直接把内容返回给浏览器。

加载完成后调用dispatch方法，初始化request

C:\Users\xiaoyang123\Desktop\QQ截图20140826153936.png

如果是JSON请求，则返回JsonRequest，否则返回HttpRequest方法，他们有一个共同的父类webRequest，之后处理请求



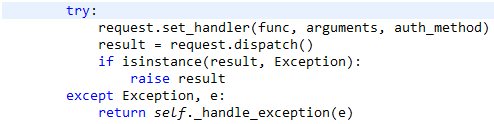
在webRequest的\_\_enter\_\_方法中会把当前request对象压入一个全局堆栈\_request\_stack，\_\_exit\_\_方法会把该对象出栈，而这个堆栈对应的是全局变量request，也就是说本次请求后面的调用中用到的request就是刚才入栈的webRequest。

在with之后，会获取ir\_http，如果执行过程中没有抛出异常，则会调用ir\_http.\_dispatch()方法，如下图：

C:\Users\xiaoyang123\Desktop\QQ截图20140826161519.png

ir\_http.\_dispatch()是一个很重要的方法，在该方法中，会设置处理该request的handler，指明应该由哪个endpoint去处理这个请求。并且调用request的dispatch方法，request的dispatch才会真正调用指定模块所对应的方法。

endpoint对象实例包含两个重要信息：请求路径和对应的处理方法。所有与路径绑定的endpoint都保存在ir\_http的一个名为\_routing\_map的属性中，这个属性首次初始化发生在第一次http请求到达的时候。具体调用是http.py的**routing\_map**方法。该方法会对每个继承自Controller的类进行初始化并把这些类中的具有routing属性的方法生成新的endpoint。在方法的后半部分，会告知request本次请求的handler，正如前文所述，在request的dispatch方法中会具体执行handler，如下图所示：



在request的dispatch方法中会真正执行绑定的处理方法。并且reqeust要求处理方法必须返回Response类型的对象。

如果引入website，website包中的ir\_http会继承并覆盖原有的ir\_http，并且website的ir\_http也重写了父类的\_dispatch()方法，在继承的\_dispatch()方法中，会给request增加一个website属性，这个website对应于数据表website中id为1的那条数据（硬编码）。

处理完request之后的返回值