handler注解开发配置说明

代码分析

```
@Controller
public class ItemList3 {

 @RequestMapping("/queryItem.action")
 public ModelAndView queryItem() {
  // 商品列表
  List<Items> itemsList = new ArrayList<Items>();

  Items items_1 = new Items();
  items_1.setName("联想笔记本");
  items_1.setPrice(6000f);
  items_1.setDetail("ThinkPad T430 联想笔记本电脑！");

  Items items_2 = new Items();
  items_2.setName("苹果手机");
  items_2.setPrice(5000f);
  items_2.setDetail("iphone6苹果手机！");

  itemsList.add(items_1);
  itemsList.add(items_2);

  // 创建modelAndView准备填充数据、设置视图
  ModelAndView modelAndView = new ModelAndView();

  // 填充数据
  modelAndView.addObject("itemsList", itemsList);
  // 视图
  modelAndView.setViewName("order/itemsList");

  return modelAndView;
 }
}
```

1. @controller：标识该类是一个handler。标记在类的上面
2. @RequestMapping
    1. URL路径映射：
       @RequestMapping(value="/item")或@RequestMapping("/item）value的值是数组，可以将多个url映射到同一个方法
    2. 窄化请求映射
       在class上添加@RequestMapping(url)指定通用请求前缀， 限制此类下的所有方法请求url必须以请求前缀开头，通过此方法对url进行分类管理。
       如下：
       @RequestMapping放在类名上边，设置请求前缀
       @Controller
       @RequestMapping("/item")
       方法名上边设置请求映射url：
       @RequestMapping放在方法名上边，如下：
       @RequestMapping("/queryItem ")
       访问地址为：/item/queryItem
    3. 请求方法限定
       限定GET方法
       @RequestMapping(method = RequestMethod.GET)
       如果通过Post访问则报错：
       HTTP Status 405 - Request method 'POST' not supported

例如：
@RequestMapping(value="/editItem",method=RequestMethod.GET)
限定POST方法
@RequestMapping(method = RequestMethod.POST)
如果通过Post访问则报错：
HTTP Status 405 - Request method 'GET' not supported
GET和POST都可以
@RequestMapping(method={RequestMethod.GET,RequestMethod.POST})

3. controller方法返回值

    1. 返回ModelAndView
    controller方法中定义ModelAndView对象并返回，对象中可添加model数据、指定view。
    2. 返回void
    在controller方法形参上可以定义request和response，使用request或response指定响应结果：
        - 使用request转向页面，如下：request.getRequestDispatcher("页面路径").forward(request, response);
        - 也可以通过response页面重定向：response.sendRedirect("url")

    也可以通过response指定响应结果，例如响应json数据如下：
    response.setCharacterEncoding("utf-8");
    response.setContentType("application/json;charset=utf-8");
    response.getWriter().write("json串");

    3. 返回string字符串
        - 逻辑视图名
        controller方法返回字符串可以指定逻辑视图名，通过视图解析器解析为物理视图地址。 //指定逻辑视图名，经过视图解析器解析为jsp物理路径：/WEB-INF/jsp/item/editItem.jsp return "item/editItem";
        - Redirect重定向
        Contrller方法返回结果重定向到一个url地址，如下商品修改提交后重定向到商品查询方法，参数无法带到商品查询方法中。//重定向到queryItem.action地址,request无法带过去 return "redirect:queryItem.action";
        - forward转发
        controller方法执行后继续执行另一个controller方法，如下商品修改提交后转向到商品修改页面，修改商品的id参数可以带到商品修改方法中。//结果转发到editItem.action，request可以带过去 return "forward:editItem.action";

补充：

1. redirect方式相当于“response.sendRedirect()”，转发后浏览器的地址栏变为转发后的地址，因为转发即执行了一个新的request和response。由于新发起一个request原来的参数在转发时就不能传递到下一个url，如果要传参数可以/item/queryItem.action后边加参数，如下：/item/queryItem?...&.....
2. forward方式相当于“request.getRequestDispatcher().forward(request,response)”，转发后浏览器地址栏还是原来的地址。转发并没有执行新的request和response，而是和转发前的请求共用一个request和response。所以转发前请求的参数在转发后仍然可以读取到。

完