**涉及使用地图接口的文件：**

static/

baidu/

gaode.html

script/

widget.bmap.js

system/

admin/

view/

common/

header.html

data/

city/

create.html（仅是城市编码文件下载链接）

create.html.back（可能是create.html的备份文件）

edit.html（仅是城市编码文件下载链接）

group/

autopai.html

badweather.html

baseconfig.html

create.html

edit.html

set\_map.html

setarea.html

timeconfig.html

timeoutconfig.html

models/

magic/

baidu.mdl.php

themes/

dispatch/（**被禁用、废弃的模块，代码几乎被全部删除，看名字可能是“调度”增值模块**）

order/

paiorder.html

ploygion.html

waimai/

block/

header.html

order/

jiedan\_map.html

order\_map.html

shop/

map.html

static/

js/

common.js

ucenter/

gaodemap.html

gaodemap1.html（可能是上面这个文件的一个重命名备份）

demo.html（无用）

test.html（无用）

wmbiz/

newreg/

reg\_one.html

shop/

deliver/

gaode.html

**各文件功能整理：**

**路径：static/baidu/gaode.html**

地图坐标拾取器（一个modal对话框）会使用这个页面进行地图的展示和选取坐标点。

使用方式是通过给弹出拾取工具的元素（如a、button等）添加属性名map-marker，map-marker的属性值是接收选取结果的input的ID选择器，如：

<a map-marker="#LngInput,#LatInput">打开坐标选择器</a>

在使用到坐标拾取器的文件会使用JQ对具有此属性的页面元素进行事件绑定，现在暂时找到绑定处理事件的文件有以下几个：

admin/script/kt.admin.js 总管理后台使用

themes/default/biz/block/footer.html 未知，biz是商户后台管理

themes/default/biz/waimai/order/manage.html 未知，biz是商户后台管理，应该是外卖订单管理

themes/default/biz/weidian/order/manage.html 未知，biz是商户后台管理，应该是微店订单管理

themes/default/ditui/block/footer.html 未知，ditui是“地推”（地域/区推广?）

themes/default/wuye/block/footer.html 未知

**路径：admin/script/kt.admin.js**

几乎所有后台管理的页面都引入了此脚本，主要是总管理后台界面元素的事件处理，如警示框、确认框、modal对话框等，由于坐标拾取器也属于modal框的一种，这个文件中主要做的就是对具有map-marker属性的元素进行click事件绑定，以使得后台管理的页面只要定义一个具有map-marker属性的元素，就可以使用坐标拾取器。

**路径：system/admin/view/common/header.html**

这是总管理后台的共用页面段代码，其作用是封装两个函数geocoder()及placeapi()。

geocoder()的作用是根据经度(lng，第一个参数)、纬度(lat，第二个参数)向高德服务器获取对应最近的地理位置名称的信息。

placeapi()目前没有在后台的视图中找到使用的地方，但它使用的接口地址为“v3/place/text”，查高德API手册，这是“关键词搜索”的接口，用于在某个范围内搜索地理名称。

**路径：system/admin/view/group/autopai.html**

未知作用及功能，使用此模板的控制器是“group/group:autopai”，控制器的第一条语句已经被写为return false，禁用了引控制器的功能。

**路径：**

**system/admin/view/group/badweather.html** 外卖 》 配送站管理 》 编辑配送站的“恶劣天气配置”

**system/admin/view/group/baseconfig.html** 外卖 》 配送站管理 》 编辑配送站的“配送费基础设置”

**system/admin/view/group/timeconfig.html**  外卖 》 配送站管理 》 编辑配送站的“配送时间设置”

**system/admin/view/group/timeoutconfig.html** 外卖 》 配送站管理 》 编辑配送站的“超时规则设置”

这几个文件中仅是引入了这个官方JS连接，但可能并没有在页面中使用，因为删除后也没有发现对这个页面的功能产生什么影响。

**路径：**

**system/admin/view/group/create.html**  外卖 》 配送站管理 》 添加配送站

**system/admin/view/group/edit.html** 外卖 》 配送站管理 》编辑配送站的“基础设置”

**system/admin/view/group/set\_map.html** 外卖 》 配送站管理 》编辑配送站的“配送区域”

**system/admin/view/group/setarea.html** 外卖 》 配送中心 》 配送站区域

通过地图的Javascript API引入地图，以能够使得设置坐标点、设置配送的多边形范围的功能。

**路径：system/models/magic/baidu.mdl.php**

调用地图API以进行与地图相关的计算、解析功能，如设计两个经纬度间的距离，获取指定经纬最近的地理名称，根据地理名称查询对应的经纬度坐标等。

目前共有接口如下:

juli() 获取两点间的距离（应该是最短路程）

juliinfo() 获取两点间的距离（路程）信息，与juli不同的是juli只返回路程距离，这个还包含了路径信息（可能是中间经过的一些关键位置的信息）

decode\_by\_location() 通过经纬度获取最靠近的地理位置位置

geocode\_by\_addr() 通过地理位置名称查询经纬度

**路径：themes/waimai/block/header.html**

用户端很多页面共用的页面起始段代码，其中的参数key使用了固定值，而真正需要地图的页面，如“收货地址”中选择地址、“订单详情”中显示商家、收货地、配送员位置等页面又会使用变量设置的key来将地图脚本重新引入页面，所以**这个可能只是遗留代码**。

**路径：themes/waimai/order/jiedan\_map.html**

用于显示正在进行的订单详情中商家位置、收货位置、骑手位置等信息的地图。用法也是将官方JS引入到页面中，并在参数上指定可以画图（显示位置、距离）的插件。

**注：这是嵌入在订单详情中的地图**

**路径：themes/waimai/order/order\_map.html**

功能、用法同jiedan\_map.html，不同的是这是以一整个页面来显示地图和相关信息，而不是嵌入在订单详情中。

**路径：themes/waimai/shop/map.html**

用整个页面展示商家在地图上的位置，功能点在用户端中，进入店铺，再点“详情”，进入详情页面后再点商家地址，就会进入此页面。

**路径：themes/waimai/static/js/common.js**

由themes/waimai/block/header.html引入使用，是用户端通用的一个脚本文件，其作用也和system/admin/view/common/header.html一样，定义了两个函数geocoder()和placeapi()，函数作用也一样。

**路径：themes/wmbiz/newreg/reg\_one.html**

由控制器system/wmbiz/controllers/newreg:one使用，控制器注释为“申请入驻第一步”，但访问时无法访问，**可能该功能已经禁用**。

**路径：themes/wmbiz/shop/deliver/gaode.html**

商家管理后台的“店铺管理”》“配送管理”中查看、编辑配送区域（商家自行配送才可设置配送区域）需要使用地图交互进行设置，功能与总后设置配送区域相同，与此模板文件同一级目录的还有index.html和edit.html，但这两个模板现在已经没有控制器在使用。

**地图需要实现的功能：**

1、将地图嵌入到地图中，并地图上标识出若干个位置

2、可以地图上自定绘制图形，以标识一些自定义信息（如商家图标、骑手图标、距离等）

3、可以地图上绘制多边形，所可获得所绘制的多边形各点的经纬度坐标

4、可以地图上点击设定位置，并可获得所选定位置的经纬度坐标

5、可按需要设置是否可缩放地图，或设定固定的地图缩放等级

经查看谷歌地图的API文档，以上功能都具备，只是使用方式不同，但修改难度不高。

**地图相关关键功能整理：**

一、static/baidu/gaode.html坐标拾取器(注:以admin/script/kt.admin.js为例，其他类推)

kt.admin.js第172行进行了map-marker属性元素的点击事件绑定。

在元素被点击后，处理函数会将map-marker属性值读出，然后解析出经度、纬度两个input的ID选择器。

然后使用两个选择器分别读取当前是经纬度值（如果只有一个input，则表示为单个经纬度input，经纬度使用“,”分隔）。

将读取到的经纬度值及回调处理函数传递给Widget.BMap.Marker()进行处理。

处理完后，被回调的处理函数会得到一个拾取完成的“经度,纬度”的字符串，然后使用这个字符串设置接收结果的input。

Widget.BMap被定义在了static\script\widget.bmap.js和static\script\widget.gmap.js中，但当前所有的页面都只使用static\script\widget.bmap.js。

Widget.BMap只定义了一个接口(方法)，名为Marker，这个函数会接收一个经纬度标识对象结构为：

{

lng:经度值,

lat:纬度值

}

如果没有则忽略此参数，如果有，则会被拼接到url末尾，作为参数传递给static/baidu/gaode.html进行处理。

此外函数还接受一个回调函数，如果拾取器是被点击“确定”关闭的，则此回调函数会被调用，并收到一个“经度,纬度”表示选取结果的字符串。

在设定好这些参数后，Marker方法就会使用layout.open来打开一个modal（模态框），并使用static/baidu/gaode.html来显示地图，及拾取坐标。

模态框被点击“确定”关闭后，Marker传递给layout.open的回调函数被调用，此回调会查找模态框中用来存放经纬度坐标的Input中的值并读取出来，接着将读取结果作为参数，调用并传递给Marker所接收到的回调处理函数，然后使用layout.close来并闭模态框。

geode.html是如何将地图中拾取的坐标设置到layout中的input的？

实际上是绑定地图的click事件，当地图有click事件发生时，就会通过回调的事件获取到点击处的经纬度值，然后再清除地图、放置新的图标，最后使用获取到的经纬度值更新Input的值。

二、system/admin/view/common/header.html及其他地方封闭的geocoder()的实现过程【placeapi未被使用，所以暂时忽略，实现过程也是和geocoder一样的】

此函数接受一个经度值（第一个参数）、一个纬度值（第二个参数）和一个回调函数。

经纬度都将被直接以参数的形式传递给地图的JS API，而回调函数则使用闭包，封装在一个“临时函数”中进行回调。

临时函数的函数名是通过GGUID()获得的，然后使用window[GGUID()]的形式来定义这个临时的函数，然后通过callback=临时函数名作为参数传递到地图JS API，即使用JSONCALLBACK的方式进行调用。

JS API返回后就会执行设定的临时函数，临时函数又会调用第三个参数设定的回调函数，并把JS API传来的参数直接传给回调函数。

三、system/admin/view/group/create.html、

system/admin/view/group/set\_map.html、

system/admin/view/group/setarea.html、

themes/wmbiz/shop/deliver/gaode.html

中有关坐标、多边形显示及编辑的实现。

多边形的显示及编辑需要的是地图本身的功能来实现，而大多数地图都具有这样的功能，所以重点是在多边形数据的表单数据结构，如何将地图的多边形数据读出并转为后端接口所接受的数据格式。

从标的实现和坐标拾取器一样，只是这个功能被直接定义到了页面中而已，而坐标的表单数据依然是两个input共同提交。

1、放置、显示多边形

服务器保存区域多边形数据是能过一个key-value数组来保存的，所有的点都保存到一个数组中，格式是：

[

'polygon\_point' => [

'area\_1' =>[

0 =>[

'lat' => '22.864328',

'lng' => '108.182929',

],

1 => [

'lat' => '22.77717',

'lng' => '108.216231',

],

2 => [

'lat' => '22.791414',

'lng' => '108.376051',

],

3 => [

'lat' => '22.888685',

'lng' => '108.371931',

],

],

]

]

可以看到，一个点就是一个数组元素，使用lat和lng为键，分别标识了经纬度的值。

服务器返回后，将多边形显示在地图上，可能不同的地图会有不同的方式，但原理一样，就是将这些多边形数据取出，放置到地图上，如高德地图会接受一系列由经纬度组成的二维数组即：

[

[108.182929,22.864328],

[108.216231,22.77717],

[108.376051,22.791414],

[108.371931,22.888685]

]

2、多边形的编辑和获取编辑结果

可能不同的地图会有不同的方式，但所需要做的，就是将多边形各个点的经纬度，转为以下格式提交到后端

[

0 => [

'lng' => '108.137985',

'lat' => '22.884609',

],

1 => [

'lng' => '108.1494',

'lat' => '22.76431',

],

2 => [

'lng' => '108.490494',

'lat' => '22.622409',

],

3 => [

'lng' => '108.63675',

'lat' => '22.818816',

],

4 => [

'lng' => '108.480881',

'lat' => '22.968798',

],

]

可以看到，实际上就是一个由经纬度对象组成的数组，如果用表单表示就是：

form[0]['lng'] = "108.137985"

form[0]['lat'] = "22.884609"

form[1]['lng'] = "108.1494"

form[1]['lat'] = "22.76431"

. . .

而提交的数据表单名，查看相应的控制器获取的是什么数据即可，如总后台为“polygon\_point”。

**其他：**

像距离检测、是否超出范围之类的，都是与多边形数值点的相关计算，与地图和坐标系无关，虽然高德地图使用的经纬度坐标系和谷歌地图不同，但最终也还是一些坐标点，应该不影响使用。