**目录规范与域名配置**

**roadmap.path**

解释：定制项目文件属性，包括但不限于 **产出路径，访问url，资源id，默认依赖，文件类型**。

值类型：Array

默认值：无

说明：roadmap.path配置是fis编译系中非常核心的机制，使用它可以控制文件编译后发布的路径或访问的url、操纵文件属性、为fis产出的资源表添加扩展信息，它的 [实现原理](https://github.com/fouber/fis-kernel/blob/master/lib/uri.js#L45-L67) 也很简单：当fis创建一个内部的 [file对象](https://github.com/fouber/fis-kernel/blob/master/lib/file.js#L130) 时，会利用roadmap.path来匹配文件路径，如果命中，则将当前规则下的属性及其值赋给file对象，从而影响file对象的相关信息(发布路径、访问url、资源表属性等)。roadmap.path是fis系统中资源定位的核心能力，具有非常重要的意义。由于fis自动化工具接管了js、css和类html语言的 **资源定位能力**，因此，用户在开发时只需使用相对路径对资源进行引用，fis编译时会根据roadmap.path的配置调整引用内容，并将代码产出到配置指定的位置，一切都配合的非常完美！

支持的配置项：

reg：用于匹配文件路径的正则(RegExp)或通配(String)。文件路径是相对项目根目录的路径，以 / 开头。

release：设置文件的产出路径。默认是文件相对项目根目录的路径，以 / 开头。该值可以设置为 false ，表示为不产出（unreleasable）文件。

url：指定文件的资源定位路径，以 / 开头。默认是 release 的值，url可以与发布路径 release 不一致。

query：指定文件的资源定位路径之后的query，比如'?t=123124132'。

id：指定文件的资源id。默认是 namespace + subpath 的值。

charset：指定文本文件的输出编码。默认是 utf8，可以制定为 gbk 或 gb2312等。

isHtmlLike：指定对文件进行html相关的 [语言能力扩展](http://fis.baidu.com/docs/more/fis-standard.html)

isJsLike：指定对文件进行js相关的 [语言能力扩展](http://fis.baidu.com/docs/more/fis-standard.html)

isCssLike：指定对文件进行css相关的 [语言能力扩展](http://fis.baidu.com/docs/more/fis-standard.html)

useCompile：指定文件是否经过fis的编译处理，如果为false，则该文件不会做任何编译处理。

useHash：指定文件产出后是否添加md5戳。默认只有js、css、图片文件会添加。

useDomain：指定文件引用处是否添加域名。

useCache：指定文件编译过程中是否创建缓存，默认值是 true。

useMap：指定fis在打包阶段是否将文件加入到map.json中索引。默认只有isJsLike、isCssLike、isMod的文件会加入表中

useParser：指定文件是否经过parser插件处理。默认为true，值为 false 时才会关闭。

usePreprocessor：指定文件是否经过preprocessor插件处理。默认为true，值为 false 时才会关闭。

useStandard：指定文件是否经过内置的三种语言标准化流程处理。默认为true，值为 false 时才会关闭。

usePostprocessor：指定文件是否经过postprocessor插件处理。默认为true，值为 false 时才会关闭。

useLint：指定文件是否经过lint插件处理。默认为true，值为 false 时才会关闭。

useTest：指定文件是否经过test插件处理。默认为true，值为 false 时才会关闭。

useOptimizer：指定文件是否经过optimizer插件处理

useSprite：指定文件是否进行csssprite处理。默认是 false，即不对单个文件进行csssprite操作的，而只对合并后的文件进行。fis release中使用 --pack 参数即可触发csssprite操作。

isMod：标记文件为组件化文件。被标记成组件化的文件会入map.json表。并且会对js文件进行组件化包装。

extras：在map.json中的附加数据，用于扩展map.json表的功能。

requires：默认依赖的资源id表，类型为Array。

用法：

fis.config.merge({

roadmap **:** {

path **:** [

{

*//所有widget目录下的js文件*

reg **:** 'widget/\*\*.js',

*//是模块化的js文件（标记为这种值的文件，会进行amd或者闭包包装）*

isMod **:** **true**,

*//默认依赖lib.js*

requires **:** [ 'lib.js' ],

*//向产出的map.json文件里附加一些信息*

extras **:** { say **:** '123' },

*//编译后产出到 /static/widget/xxx 目录下*

release **:** '/static$&'

},

{

*//所有的js文件*

reg **:** '\*\*.js',

*//发布到/static/js/xxx目录下*

release **:** '/static/js$&'

},

{

*//所有的ico文件*

reg **:** '\*\*.ico',

*//发布的时候即使加了--md5参数也不要生成带md5戳的文件*

useHash **:** **false**,

*//发布到/static/xxx目录下*

release **:** '/static$&'

},

{

*//所有image目录下的.png，.gif文件*

reg **:** /^\/images\/(.\*\.(?:png|gif))/i,

*//访问这些图片的url是 '/m/xxxx?log\_id=123'*

url **:** '/m/$1?log\_id=123',

*//发布到/static/pic/xxx目录下*

release **:** '/static/pic/$1'

},

{

*//所有template目录下的.php文件*

reg **:** /^\/template\/(.\*\.php)/i,

*//是类html文件，会进行html语言能力扩展*

isHtmlLike **:** **true**,

*//发布为gbk编码文件*

charset **:** 'gbk',

*//发布到/php/template/xxx目录下*

release **:** '/php/template/$1'

},

{

*//前面规则未匹配到的其他文件*

reg **:** /.\*/,

*//编译的时候不要产出了*

release **:** **false**

}

]

}

});

**roadmap.ext**

解释：指定后缀名与标准化语言的映射关系。

值类型：Object

默认值：无

说明：fis允许在前端开发中使用less、coffee、utc等非标准语言，并能利用插件将它们编译成标准的js、css语言。这个过程是由modules.parser配置的插件处理的。编译之后，less会变成css文件，那么，后续对于css的处理应该同样可以适用于less的生成文件，因此，这个时候需要通过配置告诉fis，less文件会编译为css文件，并在后续的处理过程中当做css文件对待。

用法：

*//fis-conf.js*

fis.config.merge({

roadmap **:** {

ext **:** {

*//less后缀的文件将输出为css后缀*

*//并且在parser之后的其他处理流程中被当做css文件处理*

less **:** 'css',

*//coffee后缀的文件将输出为js文件*

*//并且在parser之后的其他处理流程中被当做js文件处理*

coffee **:** 'js',

*//md后缀的文件将输出为html文件*

*//并且在parser之后的其他处理流程中被当做html文件处理*

md **:** 'html'

}

}

});

**roadmap.domain**

解释：设置静态资源的域名前缀。

值类型：Object | string

默认值：无

说明：fis扩展了html、js、css的[三种语言能力](http://fis.baidu.com/docs/more/fis-standard.html)，并支持对资源的定位，定位包括 **开发路径与发布路径的映射关系** 以及 **静态资源服务器域名设置**。roadmap.domain节点就是用于控制该能力的配置。

注意：domain的值如果不是特殊需要，请 **不要以"/"结尾**。

用法：

*//fis-conf.js*

*//用法一*

fis.config.merge({

roadmap **:** {

*//所有静态资源文件都使用 http://s1.example.com 或者 http://s2.example.com 作为域名*

domain **:** 'http://s1.example.com, http://s2.example.com'

}

});

*//用法二*

fis.config.merge({

roadmap **:** {

domain **:** {

*//widget目录下的所有css文件使用 http://css1.example.com 作为域名*

'widget/\*\*.css' **:** 'http://css1.example.com',

*//所有的js文件使用 http://js1.example.com 或者 http://js2.example.com 作为域名*

'\*\*.js' **:** ['http://js1.example.com', 'http://js2.example.com']

}

}

});

编译时使用fis release的 --domains 参数来控制是否添加domain

$ fis release --domains --dest ../output

**roadmap.domain.image**

解释：设置图片资源的域名前缀。

值类型：Array | string

默认值：无

说明：由于使用配置roadmap.domain.ext方式来配置图片资源太麻烦，fis提供了image字段，对于符合 [project.fileType.image](http://fis.baidu.com/docs/api/fis-conf.html#fileTypeimage)规则的文件，设置相应domain配置。

用法：

*//fis-conf.js*

fis.config.merge({

roadmap **:** {

domain **:** {

*//所有图片文件，使用 http://img.example.com 作为域名*

'image' **:** ['http://img.example.com']

}

}

});

编译时使用fis release的 --domains 参数来控制是否添加domain

$ fis release --domains --dest ../output

**部署配置**

**deploy**

解释：设置项目的发布方式。

值类型：Object

默认值：无

说明：当使用 fis release 命令时，参数 **--dest <name>** 可以指定项目发布配置。deploy配置是一个key-value的object对象，--dest参数的值如果与配置的key相同，则执行该配置的部署设置。fis支持使用post请求向http服务器发送文件，服务器端可以使用php、java等后端逻辑进行接收，[fis-command-release](https://github.com/fis-dev/fis-command-release)插件中提供了一个这样的 [php版示例](https://github.com/fis-dev/fis-command-release/blob/master/tools/receiver.php)，用户可以直接部署此文件于接收端服务器上。

用法：

*//fis-conf.js*

fis.config.merge({

deploy **:** {

*//使用fis release --dest remote来使用这个配置*

remote **:** {

*//如果配置了receiver，fis会把文件逐个post到接收端上*

receiver **:** 'http://www.example.com/path/to/receiver.php',

*//从产出的结果的static目录下找文件*

from **:** '/static',

*//保存到远端机器的/home/fis/www/static目录下*

*//这个参数会跟随post请求一起发送*

to **:** '/home/fis/www/',

*//通配或正则过滤文件，表示只上传所有的js文件*

include **:** '\*\*.js',

*//widget目录下的那些文件就不要发布了*

exclude **:** /\/widget\//i,

*//支持对文件进行字符串替换*

replace **:** {

from **:** 'http://www.online.com',

to **:** 'http://www.offline.com'

}

},

*//名字随便取的，没有特殊含义*

local **:** {

*//from参数省略，表示从发布后的根目录开始上传*

*//发布到当前项目的上一级的output目录中*

to **:** '../output'

},

*//也可以是一个数组*

remote2 **:** [

{

*//将static目录上传到/home/fis/www/webroot下*

*//上传文件路径为/home/fis/www/webroot/static/xxxx*

receiver **:** 'http://www.example.com/path/to/receiver.php',

from **:** '/static',

to **:** '/home/fis/www/webroot'

},

{

*//将template目录内的文件（不包括template一级）*

*//上传到/home/fis/www/tpl下*

*//上传文件路径为/home/fis/www/tpl/xxxx*

receiver **:** 'http://www.example.com/path/to/receiver.php',

from **:** '/template',

to **:** '/home/fis/www/tpl',

subOnly **:** **true**

}

]

}

});

小贴士：--dest参数支持使用逗号（,）分割多个发布配置，比如上面的例子，我们可以使用fis release --dest**remote,local,remote2** 命令在一次编译中同时发布多个目标。

**打包配置**

**pack**

解释：配置要打包合并的文件。

值类型：Object

默认值：无

说明：fis内置的 [打包策略](http://fis.baidu.com/docs/more/fis-base.html#pack) 与传统的打包概念不同，fis的打包实际上是在建立一个资源表，并将其描述并产出为一份map.json文件，用户应该围绕着这份描述文件来设计前后端运行框架，从而实现运行时判断打包输出策略的架构。

用法：

*//fis-conf.js*

fis.config.merge({

pack **:** {

*//打包所有的demo.js, script.js文件*

*//将内容输出为static/pkg/aio.js文件*

'pkg/aio.js' **:** ['\*\*/demo.js', /\/script\.js$/i],

*//打包所有的css文件*

*//将内容输出为static/pkg/aio.css文件*

'pkg/aio.css' **:** '\*\*.css'

}

});

输出结果：使用命令 fis release **--pack** --md5 --dest ./output 编译项目，然后到output目录下查看产出的map.json内容得到：

{

"res": {

"demo.css": {

"uri": "/static/css/demo\_7defa41.css",

"type": "css",

"pkg": "p1"

},

"demo.js": {

"uri": "/static/js/demo\_33c5143.js",

"type": "js",

"deps": [

"demo.css"

],

"pkg": "p0"

},

"index.html": {

"uri": "/index.html",

"type": "html",

"deps": [

"demo.js",

"demo.css"

]

},

"script.js": {

"uri": "/static/js/script\_32300bf.js",

"type": "js",

"pkg": "p0"

},

"style.css": {

"uri": "/static/css/style\_837b297.css",

"type": "css",

"pkg": "p1"

}

},

"pkg": {

"p0": {

"uri": "/static/pkg/aio\_5bb04ef.js",

"type": "js",

"has": [

"demo.js",

"script.js"

],

"deps": [

"demo.css"

]

},

"p1": {

"uri": "/static/pkg/aio\_cdf8bd3.css",

"type": "css",

"has": [

"demo.css",

"style.css"

]

}

}

}