**一、Base提供的路由映射方法**

- root(options={})

- mount(app, options=nil) 在指定路径挂载一个rack应用

**二、HttpHelpers提供的路由映射方法**

- get(\*args, options, &block) 定义get方法的路由

match(args, options.merge(via: get), &block)

- post(\*args, options, &block) 定义post方法的路由

match(args, options.merge(via: post), &block)

- patch(\*args, options, &block) 定义patch方法的路由

match(args, options.merge(via: patch), &block)

- put(\*args, options, &block) 定义put方法的路由

match(args, options.merge(via: put), &block)

- delete(\*args, options, &block) 定义delete方法的路由

match(args, options.merge(via: delete), &block)

**三、路由域**

**1路由域**

**路由域意义**：可为域内定义的路由提供统一配置

**区别定义域：**定义域内定义的变量覆盖父级定义域的变量，而路由域是合并父级

**2路由域类型**

**普通路由域：**scope等定义

定义域：任何路由域

提供配置：scope(options)

**资源路由域：**resource、resources定义

定义域：普通路由域、资源路由域

提供配置：scope({controller: parent\_resource.controller})

parent\_resource为当前所在资源路由域的资源对象(Resources::Resource)

**继承路由域：**nested定义，位于嵌套资源之间、非集合/成员的资源路由域

定义域：资源路由域

提供配置：scope({path: parent\_resource.nested\_scope, as: parent\_resource.member\_name})

例：scope(path: 'article/:article\_id', as: 'article')

**集合路由域：**collection定义

定义域：资源路由域

提供配置：{path: parent\_resource.collection\_scope} 例：scope(path: 'articles')

**成员路由域：**member定义的路由域

定义域：资源路由域

提供配置：scope({path: parent\_resource.member\_scope })

例：scope(path: 'articles/:id')

**浅射路由域：**shallow为true时，继承路由域、成员路由域将添加浅射路由域

定义域：继承路由域、成员路由域

提供配置：

继承路由域下浅射路由域

scope(as: @scope[:shallow\_prefix], path: @scope[:shallow\_path]) { nested{yield} }

成员路由域下浅射路由域

scope(as: @scope[:shallow\_prefix], path: @scope[:shallow\_path]) { member{yield} }

**3路由域对象(Mapper::Scope)**

parent 返回父路由域

scope\_level 路由域的级别

nested? 判断是否为继承级别

resources? 判断是否为资源级别

resource\_method\_scope? 判断是否为资源方法级别，如member

[](key) 返回(路由域配置 || 父级路由域配置)

[]=(k,v) 指定路由域的配置

**4 Scoping提供创建路由域方法**

- scope(\*args, &block) 建一个路由域

路径相关

:path 路由域的路径

:shallow\_path 路由域的浅射路径(见浅射路由域)

:module 路由域的模块

:path\_names action对应的路径别名

具名相关

:shallow\_prefix 路由域的浅射as名

:as 帮助方法名的组成

参数相关

:controller 路由域的控制器

:action 路由域的控制器方法

:constraints 路由域的限定，hash或&(:request)

:shallow 继承路由域和成员路由域是否使用浅映射

:defaults 指定路由的默认参数

- scope(\*paths, options={}, &block) 同scope(options.merge(path: paths.join('/'))})

- controller(controller, options, &block) 指定:controller参数的scope

- namespace(path, &block) 指定命名空间的scope

指定module、path、as、shallow\_path、shallow\_prefix为path的scope

- constraints(constraints={}, &block) 指定constraints参数的scope

- defaults(defaults={}, &block) 指定defaults参数的scope

**四、Resources提供的资源路由映射方法**

- resources(\*resources, options, &block) 映射多资源rest路由(下属options参数)

scope参数 见scope方法

on 用于添加对应的集合或成员路由域

via 指定路由的请求方法

only 指定使用默认路由

except 指定不使用的默认路由

- collection(&block) 创建collection级别路由域

- member(&block) 创建member级别路由域

- new(&block) 创建new级别路由域

- match(path, options) 定义一个路由映射，options参数如下

scope参数 见scope

resources参数 见resources

path 指定路径或action名，@scope[:path] + path

·若path为路径，则@scope[:path] + path

·若path为默认action(如index)，则@scope[:path]

·若path为非默认action，则@scope[:path] + @scope[:path\_names][path]

as 指定路由帮助方法名

·as指定，则as为指定的值

·as未指定且上述path为默认action，则as为nil

·as未指定且上述path不为默认action，则as为path的值

to 分发的(控制器#方法)或Rack应用

defaults参数 可指定url额外参数如下

·protocol（指定匹配的协议http、https）

·subdomain（指定匹配的子域名）

·domain（指定匹配的主域）

·host（指定匹配的主机ip）

·port（指定匹配的端口）

**五、Concerns提供的路由映射宏**

concern :commentable do

resources :comments

end

resources :messages, concerns: :commentable

等价于如下

resources :messages do

resources :comments

end

**六、Mapping类提供对路由的封装**

**封装mapping对象**

+ build(@scope, @set, URI.parser.escape(path), as, options)

**返回路由所需参数**

- to\_route => [app, conditions, requirements, defaults, as, anchor]

**添加到路由集**

@set.add\_route(app, conditions, requirements, defaults, as, anchor)