DataCenter

datacenter 可用来在整个 skynet 网络做跨节点的数据共享。

当你需要跨节点通讯时，虽然只要持有其它节点的地址，就可以发送消息。但地址如何获得，却是一个问题。

早期的 skynet 提供了**具名服务**的特性，可以给一个服务起一个唯一的名字，用名字即可**发送消息**。但目前更推荐的做法是通过 datacenter 或 UniqueService。**即早期采用具名服务发送消息。**

datacenter 类似一个**全网络共享的注册表**。它是一个**树结构**，任何人都可以向其中写入一些合法的 lua 数据，其它服务可以从中取出来。所以你可以把一些需要跨节点访问的服务，自己把其地址记在 datacenter 中，需要的人可以读出。

即 datacenter 类似一个全网共享注册表，树结构。

datacenter 是一个 lua 库，使用：

```
local datacenter = require "skynet.datacenter"
```

可以进入。

一共有三个方法：

datacenter.set(key1, key2, ... , value) 可以向 key1.key2 设置一个值 value 。这个 api 至少需要两个参数，没有特别限制树结构的层级数。

datacenter.get(key1, key2, ...) 从 key1.key2 读一个值。这个 api 至少需要一个参数，如果传入多个参数，则用来读出树的一个分支。

datacenter.wait(key1, key2, ...) 同 get 方法，但如果读取的分支为 nil 时，这个函数会阻塞，直到有人更新这个分支才返回。当读写次序不确定，但你需要读到其它地方写入的数据后再做后续事情时，用它比循环尝试读取要好的多。wait 必须作用于一个叶节点，不能等待一个分支。

注意：**这三个 api 都会阻塞住当前 coroutine ，留心异步重入的问题。**

和 UniqueService 相比较，datacenter 使用起来更加灵活。你还可以通过它来交换 Multicast 的频道号等各种信息。但是，datacenter 其实是通过在 master 节点上部署了一个专门的数据中心服务来共享这些数据的。所有对 datacenter 的查询，都需要和中心节点通讯（如果你是多节点的架构的话），这有时会造成瓶颈。如果你只需要简单的服务地址管理，UniqueService 做的更好，它会在每个节点都缓存查询结果。

在 cluster 模式下使用

该功能在 cluster 模式下不能直接使用，需要自己实现。