Golang time包计时器的创建

计时器在底层是一个struct结构体，他包含了一个通道和运行时时间这么一个结构体，这个结构体中，包含了when，f，args，period这几个比较关键的字段，如果创建普通的计时器，那么会将when赋值为执行时间，f是一个sendtime函数，参数是chan，当定时器触发的时候会调用sendtime函数，给chan参数传值。每一个定时器的创建都是放在一个小顶堆（节点长度是4）的数据结构中，这个小顶堆是在process中的，总共有64个这样的process。每次有新的计时器被创建的时候，都会获取当前所在的process进行加锁操作，先先处理小顶堆中一些需要处理的状态，如删除或者移动状态，然后在插入这个定时器。插入的过程是向上调整堆（跟父节点比较，如果比父节点大就交换），而删除的过程是将最后一个元素放置在堆顶，向下调整堆（先找出孩子结点中最大的，如果比最大的孩子结点小就交换，继续向下比较）结构。计时器的删除不是立即删除的，而是通过计时器的状态，在某些特定的情况下进行删除，比如新建的时候。

计时器的执行时通过判断堆顶元素的状态、when字段和当前时间的大小关系，只有时间到了才会执行，在执行的过程中会调用timer中的f函数，并去判断timer是否是个计时器，如果是更新timer的执行时间然后调整堆，否则删除堆顶元素。然后去更新下一次堆顶元素的执行时间。

Golang gin包。有关路由的注册和匹配

Gin在路由注册部分，其实是一颗类似字典树的结构。并且gin为每一种http方法都生成了一棵树，放在一个容量为9的slice中。

在路由注册的时候，先进行路由拼接，获得一个完成的url路径，并将路由组中的方法组跟自身的方法组进行结合，获取该路由从头到尾要执行的全部方法。最后将路由注册到相应的字典树中。在注册的过程中如果扫描到具有相同路径的部分会进行结点分裂，并生成indices。最后如果没有重复就注册插入。

在路由匹配的过程中，会先申请context对象，并进行初始化。根据http请求的方法找到对应的方法树，然后进行路由匹配，获取路由方法以及解析出路径中的param参数。如果没匹配到会进行斜杠添加然后重定向。

Golang context包

Context是一个接口，他实现了4个方法分别是Value、deadline、done、err方法。context定义了三种不同类型的context分别是可取消的、包含过期时间的、以及带值类型的。可取消的和包含过期时间的在创建的时候会返回一个终止函数，如果调用该函数，会将context中的done通道关闭，并且通知所有子节点进行取消删除的操作。包含过期时间的context在创建的时候会判断过期时间是否小于父节点。如果小于的话 他会退化成一个可取消的context。需要注意的是在新建的时候会将context会他的祖先结点绑定而不一定是父节点，因为父节点有可能是带值的context，他是不包含children结构的，在context取消的时候无法向下进行取消。

Xorm坑爹的地方

1. 软删除

Xorm支持软删除，软删除的操作方式就是给映射结构体，定义一个deleted字段，当删除的时候只更新删除时间而不是直接删除，然后查询的时候不会查到相应的记录。不过如果查询的过程中我们使用自定义结构体而没有使用映射的结构体变量的时候，由于没有携带deleted字段，会将软删除的记录查询出来。如果使用map做参数就没有这个问题

原因是：xorm在查询的过程中会根据参数传进来的变量，如果是一个结构体的话会根据结构体生成一个reftable，如果是map不会生成reftable。如果reftable存在，他会根据reftable里面的字段生成查询的语句以及所需要的相应参数，只有携带了deleted的自定义结构体才会在查询的过程中，将这个字段解析出来进行软删除的查询。

如果想使用自定义结构体实现软查询的话，就需要为每个结构体携带deleted字段，并且对其设置json的tag为”-”，横杠的目的是为了不把该字段返还给前端。

1. 与数据库映射的结构体中嵌套了结构体。会报”no primary key for col”的错误

是因为在插入的过程中，xorm会对插入的数据进行映射解析生成args，如果字段类型是结构体的话，他会去xorm的store存储中获取缓存，如果没有就插入获取，如果插入的结构体没有primary key的话会报错，即使有了primary key，他只会返回primary key数组的第一个 而不是整个结构体。插入过程中会出错。

我个人觉得可能是源码部分的错误。在结构体的解析中，他会先判断列类型是否为json类型，如果不是json类型就进行上述解析，只有为json类型的才会跳过上述解析，判断是否为text类型，如果是text类型就转换成json字符串返回

Restful风格

REST：表现层状态转化

资源有很多种外在表现的形式，我们把资源具体呈现出来的形成，叫做它的表现层。URI只代表资源实体，不代表它的形式。它具体的表现形式应该在HTTP的请求头信息中用Accept和Content-Type字段指定。客户端对资源的操作不应该在URI中显示，URI是唯一的，客户端对资源的操作都通过http的方法来表现。Get（获取资源）、Post（新建资源）、Put（更新资源）、Delete（删除资源）

为什么QQ可以使用但是不能访问网页？

可能是DNS服务器的问题。QQ是通过IP来访问服务器的，而浏览器一般是通过域名来访问的。浏览器在访问服务器之前需要将域名转化为IP地址。

如果IP也不能访问，那么可能是浏览器出现了故障。

常访问的网站不能访问，没访问过的网站可以访问？

本地的DNS缓存出现问题。DNS会将域名和IP的映射关系放在本地DNS缓冲中。再次进行访问，不在通过DNS服务器，而是从本地的DNFS缓存中获取IP进行访问。如果本地的DNS缓存出现问题会导致网站不能访问。