Tedu.cn
达内教育

《分布式流媒体》实训项目

C/C++教学体系

Tedu.cn
达内教育

TNV DAY04

直播课

目录

跟踪服务器详细设计

全局变量

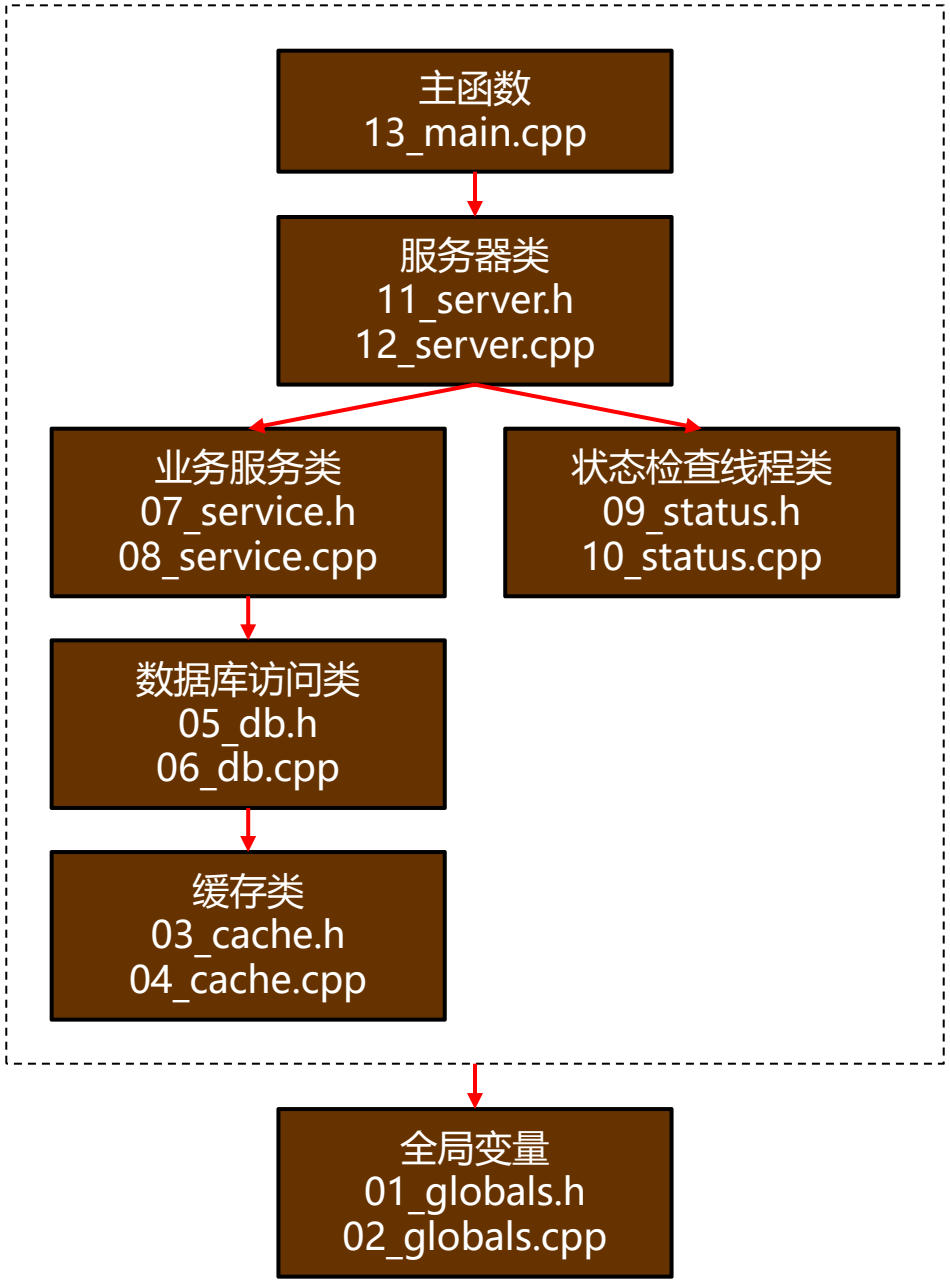
缓存类(cache_c)

Tedu.cn
达内教育

跟踪服务器详细设计

组织结构

- 02_tracker



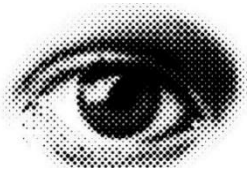
知识讲解

Tedu.cn 达内教育

开发计划

知识讲解

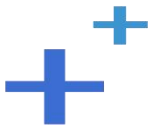
序号	内容	文档/代码	时间
10	声明全局变量	02_tracker/01_globals.h	5小时
11	定义全局变量	02_tracker/02_globals.cpp	
12	声明缓存类	02_tracker/03_cache.h	
13	实现缓存类	02_tracker/04_cache.cpp	
14	声明数据库访问类	02_tracker/05_db.h	
15	实现数据库访问类	02_tracker/06_db.cpp	
16	声明业务服务类	02_tracker/07_service.h	5小时
17	实现业务服务类	02_tracker/08_service.cpp	

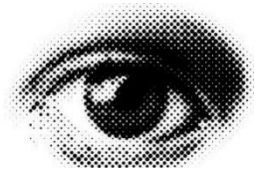


开发计划

知识讲解

序号	内容	文档/代码	时间
18	声明存储服务器状态检查线程类	02_tracker/09_status.h	5小时
19	实现存储服务器状态检查线程类	02_tracker/10_status.cpp	
20	声明服务器类	02_tracker/11_server.h	
21	实现服务器类	02_tracker/12_server.cpp	
22	定义主函数	02_tracker/13_main.cpp	
23	构建脚本	02_tracker/Makefile	
24	配置文件	etc/tracker.cfg	
25	建表脚本	sql/tracker.sql	





Tedu.cn
达内教育

全局变量

Tedu.cn
达内教育

配置信息

知识讲解

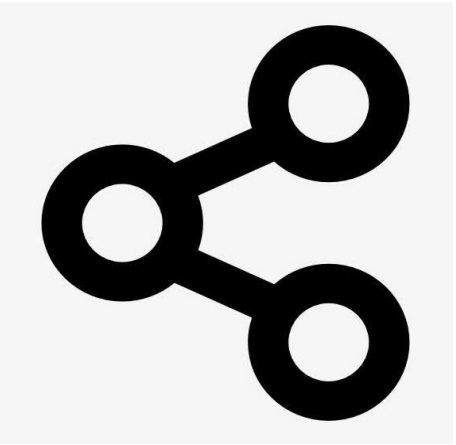
- 字符串配置表：cfg_str
  - 应用ID表：cfg_appids
  - MySQL地址表：cfg_maddrs
  - Redis地址表：cfg_raddrs
- 整型配置表：cfg_int
  - 存储服务器状态检查间隔秒数：cfg_interval
  - MySQL读写超时：cfg_mtimeout
  - Redis连接池最大连接数：cfg_maxconns
  - Redis连接超时：cfg_ctimeout
  - Redis读写超时：cfg_rtimeout
  - Redis键超时：cfg_ktimeout

Tedu.cn
达内教育

共享信息

知识讲解

- 配置共享信息
  - 应用ID表：g_appids
  - MySQL地址表：g_maddrs
  - Redis地址表：g_raddrs
- 其它共享信息
  - Redis连接池：g_rconns
  - 主机名：g_hostname
  - 组表：g_groups
  - 互斥锁：g_mutex



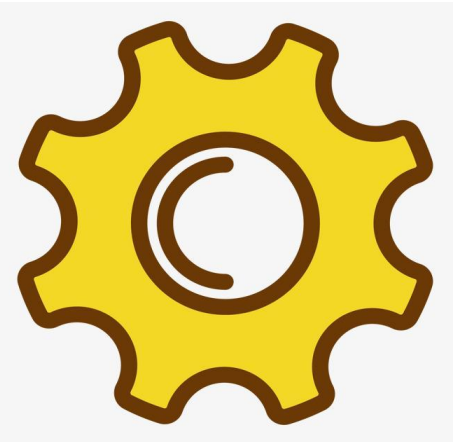
Tedu.cn
达内教育

缓存类(cache_c)

方法

知识讲解

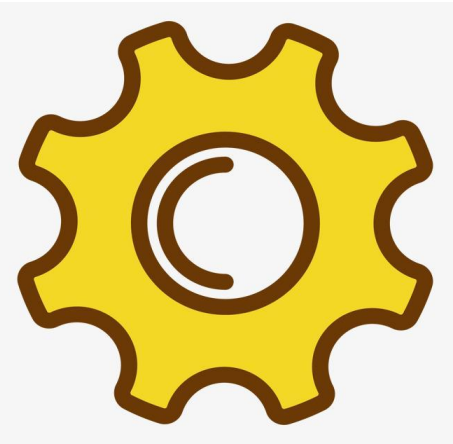
- 根据键获取其值：get
  - 构造键
  - 检查Redis连接池
  - 从连接池中获取一个Redis连接
  - 持有此连接的Redis对象即为Redis客户机
  - 借助Redis客户机根据键获取其值
  - 检查空值
  - 返回成功



方法

知识讲解

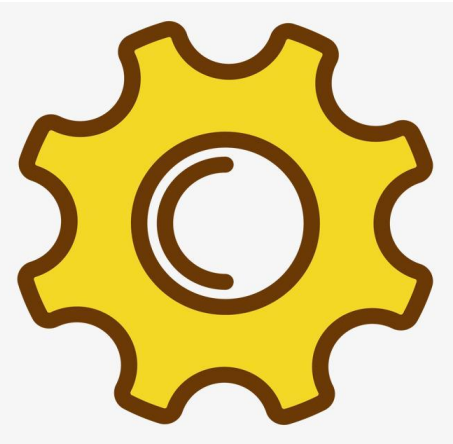
- 设置指定键的值：set
  - 构造键
  - 检查Redis连接池
  - 从连接池中获取一个Redis连接
  - 持有此连接的Redis对象即为Redis客户机
  - 借助Redis客户机设置指定键的值
  - 返回成功



方法

知识讲解

- 删除指定键值对：del
  - 构造键
  - 检查Redis连接池
  - 从连接池中获取一个Redis连接
  - 持有此连接的Redis对象即为Redis客户机
  - 借助Redis客户机删除指定键值对
  - 返回成功



Tedu.cn
达内教育

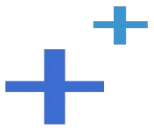
附录：程序清单

Tedu.cn
达内教育

TNV/src/02_tracker/01_globals.h

知识讲解

```
//跟踪服务器
//声明全局变量
//
#pragma once

#include <vector>
#include <string>
#include <map>
#include <list>
#include <lib_acl.hpp>
#include "01_types.h"
//
//配置信息
//
extern char* cfg_appids; //应用ID表
```

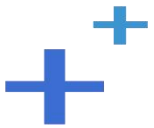


Tedu.cn
达内教育

TNV/src/02_tracker/01_globals.h

知识讲解

```
extern char* cfg_maddrs; // MySQL地址表
extern char* cfg_raddrs; // Redis地址表
extern acl::master_str_tbl cfg_str[]; // 字符串配置表

extern int cfg_interval; // 存储服务器状态检查间隔秒数
extern int cfg_mtimeout; // MySQL读写超时
extern int cfg_maxconns; // Redis连接池最大连接数
extern int cfg_ctimeout; // Redis连接超时
extern int cfg_rtimeout; // Redis读写超时
extern int cfg_ktimeout; // Redis键超时
extern acl::master_int_tbl cfg_int[]; // 整型配置表

extern std::vector<std::string> g_appids; // 应用ID表
extern std::vector<std::string> g_maddrs; // MySQL地址表
extern std::vector<std::string> g_raddrs; // Redis地址表
```

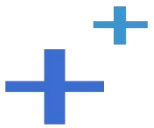


Tedu.cn
达内教育

TNV/src/02_tracker/01_globals.h

```
extern acl::redis_client_pool*  g_rconns; //Redis连接池
extern std::string g_hostname; //主机名
extern std::map<std::string,
        std::list<storage_info_t> > g_groups; //组表
extern pthread_mutex_t g_mutex; //互斥锁
```

知识讲解

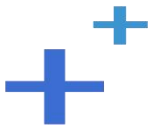


Tedu.cn
达内教育

TNV/src/02_tracker/02_globals.cpp

知识讲解

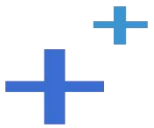
```
//跟踪服务器
//定义全局变量
//
#include "01_globals.h"
//
//配置信息
//
char* cfg_appids; //应用ID表
char* cfg_maddrs; //MySQL地址表
char* cfg_raddrs; //Redis地址表
acl::master_str_tbl cfg_str[] = { //字符串配置表
        {"tnv_apps_id", "tnvideo",         &cfg_appids},
        {"mysql_addrs", "127.0.0.1",       &cfg_maddrs},
        {"redis_addrs", "127.0.0.1:6379", &cfg_raddrs},
        {0, 0, 0}};
```



TNV/src/02_tracker/02_globals.cpp

知识讲解

```
int cfg_interval; //存储服务器状态检测间隔秒数
int cfg_mtimeout; //MySQL读写超时
int cfg_maxconns; //Redis连接池最大连接数
int cfg_ctimeout; //Redis连接超时
int cfg_rtimeout; //Redis读写超时
int cfg_ktimeout; //Redis键超时
acl::master_int_tbl cfg_int[] = { //整型配置表
        {"check_active_interval", 120, &cfg_interval, 0, 0},
        {"mysql_rw_timeout",       30, &cfg_mtimeout, 0, 0},
        {"redis_max_conn_num",    600, &cfg_maxconns, 0, 0},
        {"redis_conn_timeout",     10, &cfg_ctimeout, 0, 0},
        {"redis_rw_timeout",       10, &cfg_rtimeout, 0, 0},
        {"redis_key_timeout",      60, &cfg_ktimeout, 0, 0},
        {0, 0, 0, 0, 0}};
```

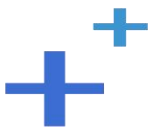


Tedu.cn
达内教育

TNV/src/02_tracker/02_globals.cpp

知识讲解

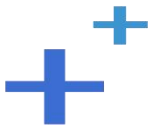
```
std::vector<std::string> g_appids; // 应用ID表
std::vector<std::string> g_maddrs; // MySQL地址表
std::vector<std::string> g_raddrs; // Redis地址表
acl::redis_client_pool*  g_rconns; // Redis连接池
std::string g_hostname; // 主机名
std::map<std::string,
        std::list<storage_info_t> > g_groups; // 组表
pthread_mutex_t g_mutex = PTHREAD_MUTEX_INITIALIZER; // 互斥锁
```



TNV/src/02_tracker/03_cache.h

知识讲解

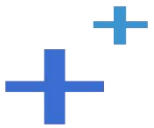
```
//跟踪服务器
//声明缓存类
//
#pragma once

#include <lib_acl.hpp>
//
//缓存类
//
class cache_c {
public:
        //根据键获取其值
        int get(char const* key, acl::string& value) const;

        //设置指定键的值
```



TNV/src/02_tracker/03_cache.h

```
        int set(char const* key, char const* value, int timeout = -1) const;

        //删除指定键值对
        int del(char const* key) const;
};
```

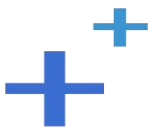
知识讲解



TNV/src/02_tracker/04_cache.cpp

知识讲解

```
//跟踪服务器
//实现缓存类
//
#include "01_globals.h"
#include "03_cache.h"

//根据键获取其值
int cache_c::get(char const* key, acl::string& value) const {
        //构造键
        acl::string tracker_key;
        tracker_key.format("%s:%s", TRACKER_REDIS_PREFIX, key);

        //检查Redis连接池
        if (!g_rconns) {
                logger_warn("redis connection pool is null, key: %s",
```



TNV/src/02_tracker/04_cache.cpp

知识讲解

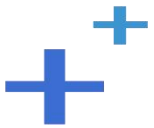
```
                tracker_key.c_str());
        return ERROR;
}

// 从连接池中获取一个Redis连接
acl::redis_client* rconn = (acl::redis_client*)g_rconns->peek();
if (!rconn) {
        logger_warn("peek redis connection fail, key: %s",
                tracker_key.c_str());
        return ERROR;
}

// 持有此连接的Redis对象即为Redis客户机
acl::redis redis;
redis.set_client(rconn);
```

Tedu.cn
达内教育

TNV/src/02_tracker/04_cache.cpp

知识讲解

```
// 借助Redis客户机根据键获取其值
if (!redis.get(tracker_key.c_str(), value)) {
        logger_warn("get cache fail, key: %s", tracker_key.c_str());
        g_rconns->put(rconn, false);
        return ERROR;
}

// 检查空值
if (value.empty()) {
        logger_warn("value is empty, key: %s", tracker_key.c_str());
        g_rconns->put(rconn, false);
        return ERROR;
}

logger("get cache ok, key: %s, value: %s",
```



TNV/src/02_tracker/04_cache.cpp

知识讲解

```
                tracker_key.c_str(), value.c_str());
        g_rconns->put(rconn, true);

        return OK;
}

//设置指定键的值
int cache_c::set(char const* key, char const* value,
        int timeout /* = -1 */) const {
        //构造键
        acl::string tracker_key;
        tracker_key.format("%s:%s", TRACKER_REDIS_PREFIX, key);

        //检查Redis连接池
        if (!g_rconns) {
```

TNV/src/02_tracker/04_cache.cpp

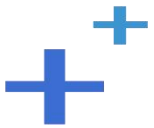
知识讲解

```
            logger_warn("redis connection pool is null, key: %s",
                    tracker_key.c_str());
            return ERROR;
    }

    // 从连接池中获取一个Redis连接
    acl::redis_client* rconn = (acl::redis_client*)g_rconns->peek();
    if (!rconn) {
            logger_warn("peek redis connection fail, key: %s",
                    tracker_key.c_str());
            return ERROR;
    }

    // 持有此连接的Redis对象即为Redis客户机
    acl::redis redis;
```

TNV/src/02_tracker/04_cache.cpp

知识讲解

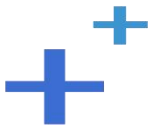
```
redis.set_client(rconn);

// 借助Redis客户机设置指定键的值
if (timeout < 0)
        timeout = cfg_ktimeout;
if (!redis.setex(tracker_key.c_str(), value, timeout)) {
        logger_warn("set cache fail, key: %s, value: %s, timeout: %d",
                tracker_key.c_str(), value, timeout);
        g_rconns->put(rconn, false);
        return ERROR;
}
logger("set cache ok, key: %s, value: %s, timeout: %d",
        tracker_key.c_str(), value, timeout);
g_rconns->put(rconn, true);
```



TNV/src/02_tracker/04_cache.cpp

知识讲解

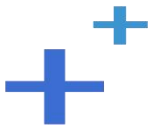
```
        return OK;
}

//删除指定键值对
int cache_c::del(char const* key) const {
        //构造键
        acl::string tracker_key;
        tracker_key.format("%s:%s", TRACKER_REDIS_PREFIX, key);

        //检查Redis连接池
        if (!g_rconns) {
                logger_warn("redis connection pool is null, key: %s",
                        tracker_key.c_str());
                return ERROR;
        }
```



TNV/src/02_tracker/04_cache.cpp

知识讲解

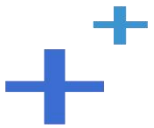
```
// 从连接池中获取一个Redis连接
acl::redis_client* rconn = (acl::redis_client*)g_rconns->peek();
if (!rconn) {
        logger_warn("peek redis connection fail, key: %s",
                tracker_key.c_str());
        return ERROR;
}

// 持有此连接的Redis对象即为Redis客户机
acl::redis redis;
redis.set_client(rconn);

// 借助Redis客户机删除指定键值对
if (!redis.del_one(tracker_key.c_str())) {
        logger_warn("delete cache fail, key: %s", tracker_key.c_str());
```



TNV/src/02_tracker/04_cache.cpp

```
            g_rconns->put(rconn, false);
            return ERROR;
        }
        logger("delete cache ok, key: %s", tracker_key.c_str());
        g_rconns->put(rconn, true);

        return OK;
    }
```

知识讲解



Tedu.cn
达内教育

复习课见