

key-value-system 使用文档

一、介绍

项目名称: 单机 key-value 存储系统。

应用场景: 互联网企业中经常要存储各种较大的相对静态的数据, 例如图片、视频、音乐等类型的文件, 分布式 key-value 系统就是用来解决这类存储需求的, 单机 key-value 系统是整个系统的基础。

数据规模: 单机存储 1000 万条记录, 平均大小 100K, 单机存储 1T 数据, 单条最大长度 5MB。

key-value-system 目前提供 C 形式的共享库。程序员可以 `#include <kvs.h>` 来使用 `put`, `get`, `delete` 接口。理论上, 系统运行时内存 345MB + `buffer` 模块内存。`buffer` 模块最小内存由宏定义 `MINIMUM_BUFFER_SIZE` 指定。

二、安装

下载:

```
$git clone https://github.com/WadeLeng/key-value-system.git
```

进入 key-value-system 目录

```
~/key-value-system$ sudo make all
```

三、使用

安装完系统之后, 可以通过 C 语言的共享库在本机使用。

在程序中 `#include <kvs.h>`

通过填充 `KVS_ENV` 参数进行初始化设定。通过 `put`, `get`, `delete` 接口进行操作。

编译方式 (gcc 4.6.1): 加参数 `-lkvs -lpthread`

```
~/key-value-system$ gcc test_kvs.c -o test -lkvs -lpthread
```

四、接口详细说明

`KVS_ENV` 参数:

`init_type:INIT_TYPE_CREATE` //系统重新创建。

`INIT_TYPE_LOAD` //载入 `IMAGE_FILE`, 继续上次退出前状态运行。

`disk_file_path` //value 存放的大文件路径。

`IMAGE_file_path` //index 在内存中的镜像。下次系统启动可以载入。

`log_file_path` //log 信息输出文件。

`buffer_sleep_time` //buffer_lookout 线程刷新闻隔时间, 数据落地条件之一。

`buffer_horizon_size` //buffer_lookout 线程中数据包大小, 数据落地条件之一。

`buffer_size` //buffer 模块在内存中的大小。

接口说明:

/*系统初始化与退出接口, 初始化前先填充参数 `KVS_ENV*`*/

```
int kv_init(const KVS_ENV* kvs);
```

```
int kv_exit();
```

/*put/get/delete 操作，用户调用之前需要先申请 value 的空间，系统背景 value 最大值是 5MB*/

```
int kv_put(const char* key, int key_size, const char* value, int value_size);
```

```
int kv_get(const char* key, int key_size, char* buf, int* buf_size);
```

```
int kv_delete(const char* key, int key_size);
```