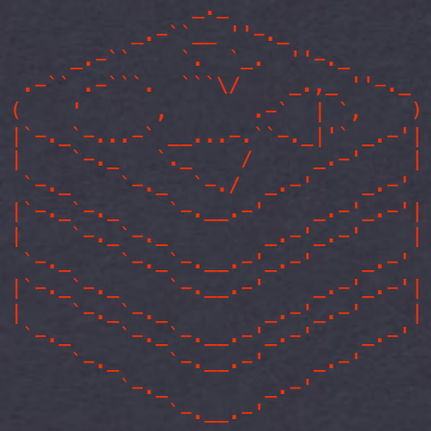

```
Redis 2.8.18 (00000000/0) 32 bit

Running in stand alone mode
Port: 6379
PID: 3807

      http://redis.io
```

走近REDIS

BY PEIXU ZHU

一个简单需求

- 实时统计某个网站的访问
  - 用一个巨大的字典？
    - std::map<std::string, VisitRecord> ip2rec;
  - 用一个数据库，比如 MySQL？
    - update VisitRecord set count=count+1 where visitor_ip = ‘xx.xx.xx.xx’;

- 如何保持这些数据？
- 如果这是一个访问量非常大的网站呢？
- 如果要提供这个服务给全球的网站使用呢？

我们需要什么

- 快速存取，快速修改，快速写入
- 可持久化
- 可扩展
- 最好还是高可用的

? => Redis

- 解决 http://www.lloogg.com 项目中的MySQL 扩展问题
- 一些故事 https://github.com/antirez/lloogg

Redis=?

- Redis = **RE**mote **DI**ctionary **S**erver
- Remote (可通过网络)
- Dictionary (Key/Value)
- Server (C/S)

Redis={…}

- In-memory, optional disk backend
- Simple key/value
- Key iteration
- Data structures (list, set, hash etc.)
- Channel Pub/Sub
- Lua scriptable
- Atomic

用户数据类型

- 字符串
- 列表
- 集合
- 有序集合
- 哈希(map / object / dictionary)
- 位元数组
- HyperLogLogs (HLL)

部署

- sentinel := replicate+
- partitioning := {replicate | stand_alone}+
- replicate := master ^ {slave}+ *# high availability*
- master, slave, stand_alone := redis_instance
- redis_instance = redis_process ^ {tcp:port}
- cluster := { service_node } + *# not production yet*
- service_node := replicate | stand_alone

开源Redis

- Open Source, commercial friendly license(3 clauses BSD license)
- https://github.com/antirez/redis （ Star:11385 , Fork:3615, Commits:1340, Contributors:188)
- https://github.com/ (9370个项目和redis密切相关)
- stackoverflow.com (Redis:5666 ，memcached: 3750)

- 活跃的项目


Mar 22, 2009 – Dec 19, 2014
Contributions: Commits
Contributions to unstable, excluding merge commits
60
40
20
0
2010
2011
2012
2013
2014

- 优秀的源代码

  src目录

```
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C                               57           5039           8836          30423
C/C++ Header                    33            345           1245           2920
make                             1             51             34            168
Bourne Shell                     1              0              0             11
-------------------------------------------------------------------------------
SUM:                            92           5435          10115          33522
-------------------------------------------------------------------------------
```

源代码分类（按照功能）

- 主模块
- 事件处理模块（fd, timer）
- 配置（configuration）
- 网络（C/S协议）模块和IO模块
- 内部数据结构（internal data structure）模块
- 序列化（serialization & restore）
- 集群管理模块
- 命令处理模块
- 工具模块

概念层次Concept Hierachy

- 集群
- 数据库（DB）
- 键/值（Key/Value）
- 值的存储、值的类型

SKIP LIST

一个分析例子

- 定时器
- 键超时处理

一个简单的应用例子

- 简单的分布式队列

“为什么Redis不使用多线程技术来处理业务？”

“当所有的努力都失败后，就看看手册吧。”

– 墨菲的技术法则

(找不到手册 | 手册不能满足要求) =>看源代码

在源代码中可以找到手册中没有写的秘密