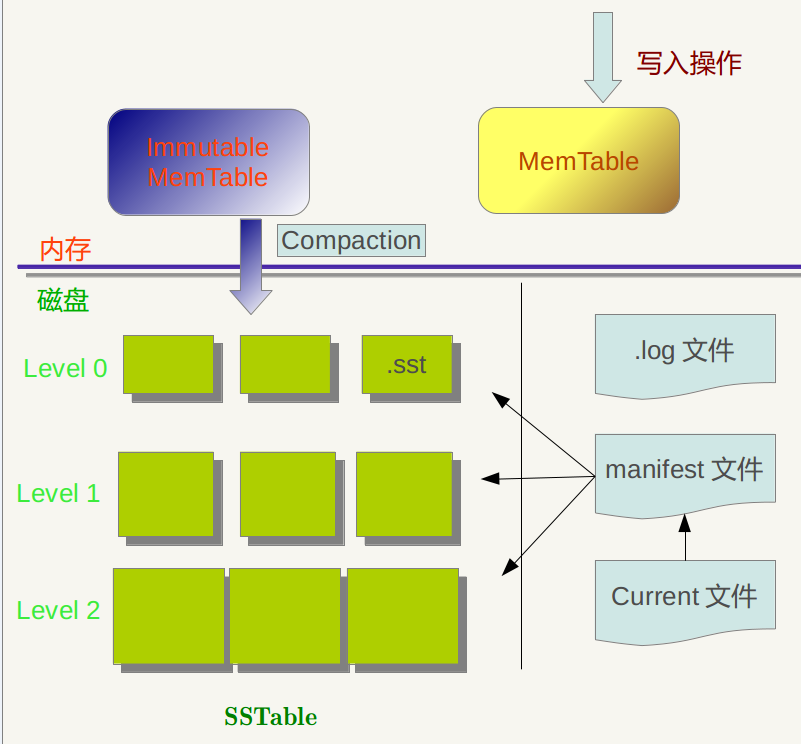
LevelDB

1. Demo Code
2. Overview
   1. 架构



* 1. 文件清单
     1. .log .sst memTable

都是用来存储K-V记录的

优先写入.log文件，崩溃是数据不丢失

MemTable满（write\_buffer\_size 4M）后会变成 im-memtable

.sst文件分级，level越高数据越旧；有序，levelo key可以重复

* + 1. CURRENT MANIFEST

MANIFEST记录文件的level，文件的最小key和最大key等信息，随着compaction变化

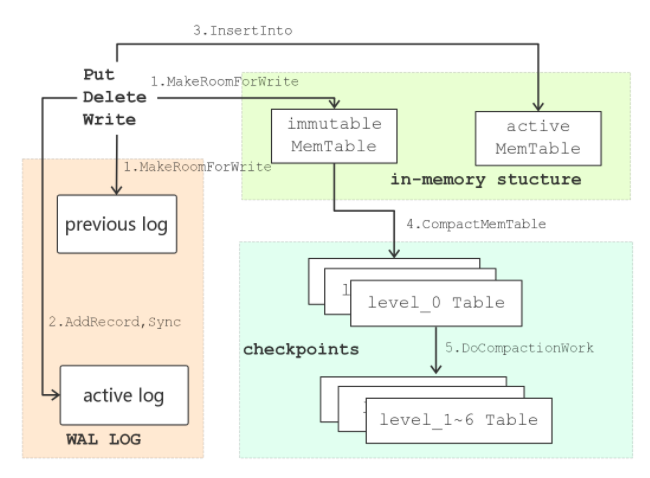
CURRENT记载当前最关心的的manifest文件名

* + 1. LOG

LOG就是操作日志文件，

* 1. Performance： https://github.com/google/leveldb

1. DB操作
   1. Write
      1. WriteOption(看代码)
      2. Delete其实就写空值



* + 1. Skip-list结构的插入操作
  1. Compaction
     1. Minor compaction

immutable MemTable 打包成level0的db文件，如果正在读取会等待

只有两个MemTable会留在内存中，打包之后会生成新的MemTable,删掉过时的log和MemTable

* + 1. Major compaction

将低层的leveldb文件压缩成高层的，

优先级低于Minor

Get(), Write(), Open(), 有有可能处罚打包操作

* 1. Read
     1. ReadOption (看代码)
     2. 在内存中依次查找memtable、immutable memtable；
     3. 如果配置了cache，查找cache；
     4. 根据mainfest索引文件，在磁盘中查找SST文件

1. Open
   1. OpenOption(看代码)
2. LevelMap

