# LocalDirAllocator

实现轮询的检查所有的目录，如果当前目录满足需求，则一直使用这个目录，如果当前目录不能满足需求了，则继续向下轮询，知道找到了满足需求的目录。

这个实现可以提供一个JVM中多个allocators（one for each unique functionality or context, like mapred, dfs-client, etc.）确保jvm中的每个context都有唯一的allocator

上面提到的contexts，其实是配置项"mapred.local.dir"

属性：

1. **private** **static** Map <String, AllocatorPerContext> *contexts* = **new** TreeMap<String, AllocatorPerContext>();
2. **private** String contextCfgItemName;

方法主要是操作AllocatorPerContext

## AllocatorPerContext

属性：

1. **private** **int** dirNumLastAccessed;
2. **private** Random dirIndexRandomizer = **new** Random();
3. **private** FileSystem localFS;
4. **private** DF[] dirDF;
5. **private** String contextCfgItemName;
6. **private** Path[] localDirsPath;
7. **private** String savedLocalDirs = "";

方法：

1. **private** **synchronized** **void** confChanged(Configuration conf)

This method gets called everytime before any read/write to make surethat any change to localDirs is reflected immediately.

* 1. 查询String newLocalDirs = conf.get(contextCfgItemName);（contextCfgItemName是类似mapred.local.dir这样的配置项）
  2. 如果savedLocalDirs和newLocalDirs不同，即配置发生了变化：
     1. 从newLocalDirs解析出每个dir，填充到localDirsPath和dirDF，savedLocalDirs
     2. 获取一个随机的索引dirNumLastAccessed = dirIndexRandomizer.nextInt(dirs.size());

1. **private** Path createPath(Path path, **boolean** checkWrite)
   1. 在一个随机的位置上新建Path file = **new** Path(localDirsPath[dirNumLastAccessed], path);
   2. 如果checkWrite为true，要使用DiskChecker检查写权限
   3. 返回file
2. **public** **synchronized** Path getLocalPathForWrite(String pathStr, **long** size, Configuration conf, **boolean** checkWrite)
   1. 更新confChanged(conf);
   2. 如果size == *SIZE\_UNKNOWN*（-1）：
      1. 随机选取一个位置
   3. 否则（即文件大小已知）：轮询所有目录，直到找到了一个容量满足条件的目录
3. **public** **synchronized** Path getLocalPathToRead(String pathStr, Configuration conf)
   1. 更新confChanged(conf);
   2. 遍历所有localDirsPath，直到找到 pathStr对应的文件
4. **public** **synchronized** **boolean** ifExists(String pathStr,Configuration conf)

同4

### PathIterator

迭代所有传入的目录