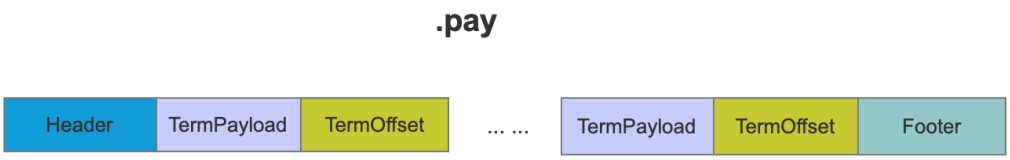
# pos&&pay文件

position在Lucene中描述的是一个term在一篇文档中的位置，并且存在一个或多个position。 payload是一个自定义的元数据(mete data)来描述term的某个属性，term在一篇文章中的多个位置可以一一对应多个payload，也可以只有部分位置带有payload。 offset是一对整数值(a pair of integers)，即startOffset跟endOffset，它们分别描述了term的第一个字符跟最后一个在文档中的位置。 每一个term在所有文档中的position、payload、offset信息在addDocument()的过程中计算出来，在内存中生成一张倒排表，最终持久化到磁盘时，通过读取倒排表，将position信息写入到.pos文件中，将payload、offset信息写入到.pay文件中。

# pay文件的数据结构

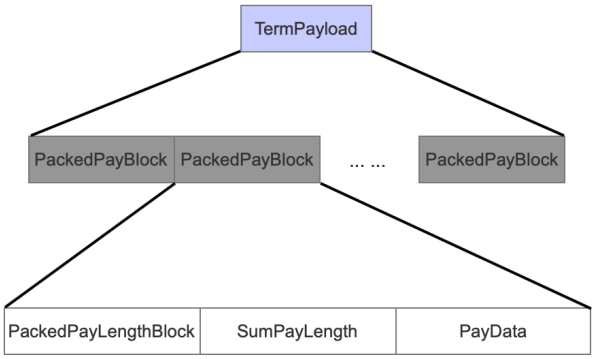
图1：




在.pay文件中，TermPayload、TermOffset分别记录一个term的payload、offset信息。

## TermPayload

图2：




### PackedPayBlock

每次处理一个term的128个position信息，就会将对应的128个payload信息（不一定每个position都对应一个payload）处理为一个PackedPayBlock。即除了最后一个PackedPayBlock，其他PackedPayBlock中都包含了当前term的128个payload信息。在后面的内容中，都默认PackedPayBlock中包含了128个payload信息。

### PackedPayLengthBlock

PackedPayLengthBlock存放了128个payload的长度数据，并且使用了PackedInts进行了压缩存储。

**这里注意是由于每一个payload的长度无法保证递增，而在使用PackedInts时会用统一大小的字节数来存放每一个payload长度，并且这个字节数取决于payload长度最长的那个。**

### SumPayLength

SumPayLength存放了这128个payload的数据长度(字节数)，在读取.pay文件用来确定128个payload的真实数据在.pay中的一个字节区间。

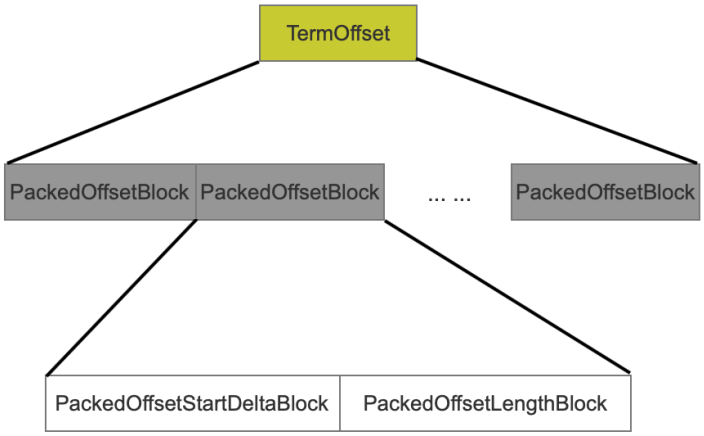
### PayData

PayData中存放了128个payload的真实数据。

注意的是，payload的真实数据没有使用压缩存储。

## TermOffset

图3：




### PackedOffsetBlock

跟TermPayload一样的是，都是每次处理一个term的128个position信息后，就会将对应的128个offset信息处理为一个block。

### PackedOffsetStartDeltaBlock

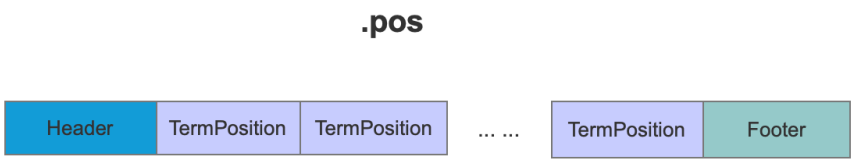
offset是一对整数值(a pair of integers)，startOffset跟endOffset分别描述了term的第一个字符跟最后一个在文档中的位置。PackedOffsetStartDeltaBlock存放了128个offset的startOffset值，并且使用了PackedInts进行压缩存储，由于这128个startOffset是个递增的值，所以实际存放了相邻两个offset的startOffset的差值。

### PackedOffsetLengthBlock

PackedOffsetLengthBlock存放了128个offset的startOffset跟endOffset差值，同样使用PackedInts进行压缩存储。

# pos文件的数据结构

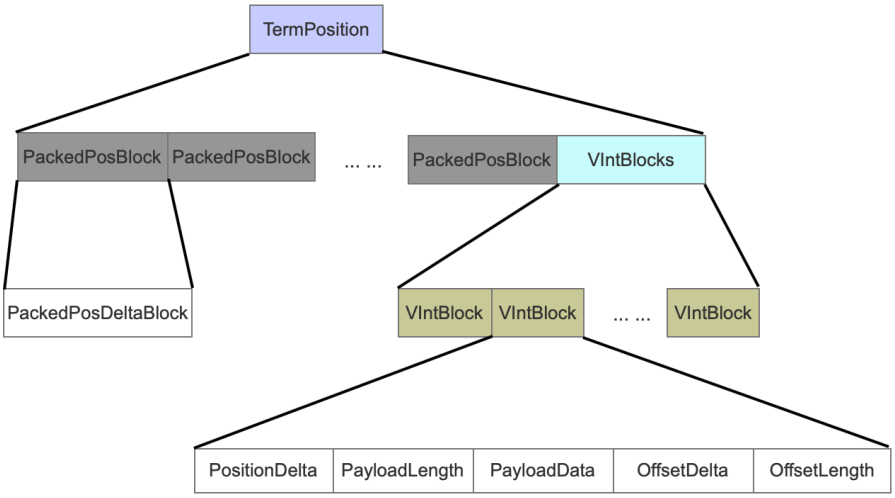
图4：




在.pos文件中，TermPosition记录一个term的position信息。

## TermPosition

图5：




## PackedPosBlock

每次处理一个term的128个position信息，就会将这些position处理为一个PackedPosBlock。

### PackedPosDeltaBlock

PackedPosDeltaBlock存放了128个位置信息，计算相邻两个position的差值后，利用PackedInts压缩存储。

## VIntBlocks && VIntBlock

如果position的个数不足128个，那么将每一个position处理为一个VIntBlock。(比如说某个term有200个position，那么前128个position处理为一个PackedPosBlock，剩余的72个position处理为72个VIntBlock，72个VIntBlock为一个VIntBlocks)。

### PositionDelta

term的position信息，这是一个差值。PositionDelta的最后一位用来标识当前position是否有payload信息。

### PayloadLength

当前position对应的payload信息的长度，在读取.pos时，用来确定往后读取的一个字节区间。

### PayloadData

当前position对应的payload真实数据。

### OffsetDelta

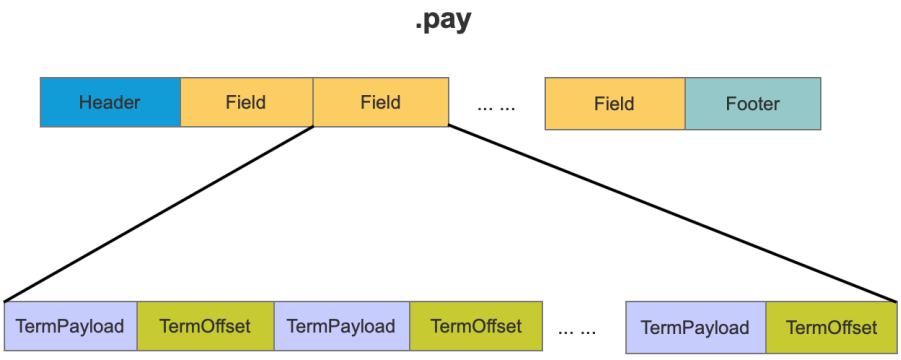
当前position对应的offset的startOffset值，同样是个差值

### OffsetLength

当前position对应的offset的endOffset与startOffset的差值。

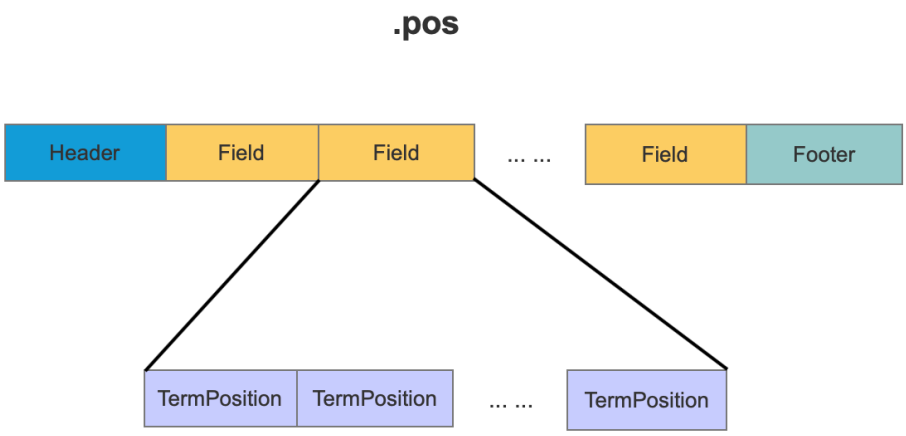
## 多个域的pay文件的数据结构

图6：




## 多个域的pos文件的数据结构

图7：




## 结语

.pos、.pay、.doc、.tim、.tip文件都是通过读取内存倒排表的过程中一起生成的，另外.doc跟.pos、.pay文件还有映射关系，在后面介绍.doc文件时候会涉及。

点击下载Markdown文件