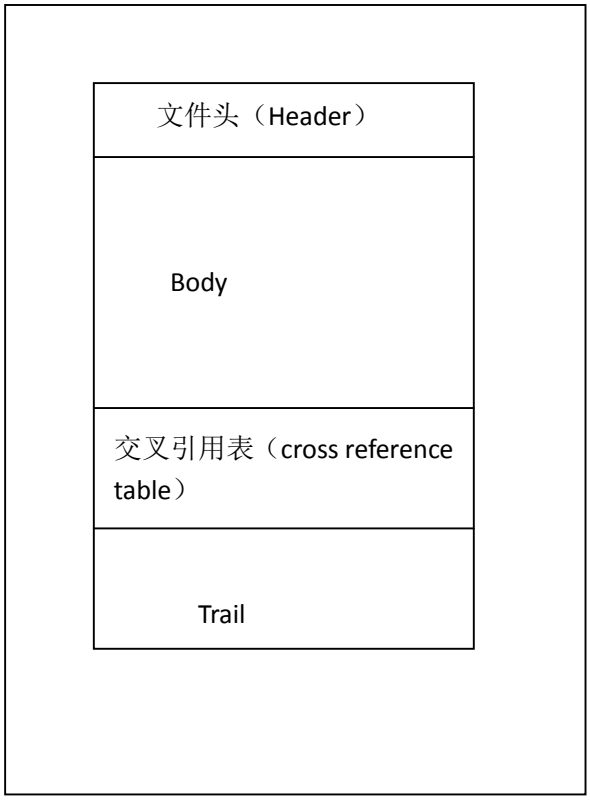
File Structure

一、一个 file 的总体结构

一个 PDF 文件，它的总体结构如下图：



文件头（Header）：确定这个 PDF 文件所依照的 PDF 版本规则。例如：%PDF-1.7 表示这个 PDF 文件所遵照的版本号是 PDF1.7.

Body：由一系列的代表文档内容的间接对象组成。这些对象，在 3.2 对象里描述的基本类型，代表文档的组成部分，像字体、页面和图片。

交叉引用表（cross reference table）：包含文件里的所有间接对象的信息。

Trail：能使一个阅读文件应用迅速的找到交叉引用表和某些特殊的对象。应用程序应该从尾部开始读。文件的最后一行仅包含文件尾标志，%%EOF.

二、File Header

一个 PDF 文件的第一行是一个用来检查这个 PDF 文件所遵循的 PDF 规范的版本的标识。如果一个文件需要遵循 PDF1.7 的规范版本，它的标题应该是%PDF-1.7。1.7 以前的版本也能被 1.7 所兼容，所以 PDF1.7 也能兼容 PDF1.0 至 PDF1.6.

从 PDF1.4 开始，Header 的版本号能够被文档的目录字典里的版本号所覆盖，这使一个 PDF 应用程序去使用增量更新的方法来更新版本。

三、File Body

PDF 文件身体由一系列的代表文档内容的间接对象组成。这些对象，在 3.2 对象里描述的基本类型，代表文档的组成部分，像字体、页面和图片。从 PDF1.5 开始，身体可以包含对象流，也就是包含一系列的对象。

四、Cross-Reference Table

交叉引用表包含的信息包括允许随机存取文件内的间接对象使得整个文件不需要去定位任何特殊对象。该表包含每个间接对象的一行输入，指定文件身体内部对象的位置。从 PDF1.5 开始，一些或者全部交叉引用信息或许都被包含在交叉引用流中了。

交叉引用表是一个 PDF 文件的唯一是固定格式的部分，这个部分允许被随机进入。这个表格包含一个或者多个交叉引用部分。最初，这整个表格由单个部分组成（或许两个部分，如果这个文件是线性化的，详见附录 F）。当文件每次被更改时，都会增加一个额外部分。

每一个交叉引用部分都是关键字 xref 开始的。紧跟这一行的是一个或者更多个依照任何命令出现的交叉引用部分。这个分段结构部分对增量更新很有用，自从它被 PDF 文件加入所需遵循的一个新的交叉引用部分，包含被加入或者删除的对象的数目。对于一个从来不会被更改的文件，它的交叉引用部分仅包含一个部分，它的对象编号从 0 开始。每一个交叉引用部分都包含一连串对象的编号。这个部分是从一条线开始，这条线是包含 2 个被空格隔开的编号的：一个是这一部分的第一个对象的对象编号，另一个是这个部分的编号数。例如：28 5 这一行，表示包含 5 个对象的部分，这部分是从 28 开始，依次连续，到 32.

交叉引用条目包含 2 种：一种是正在使用的对象，另一种是已经被删除的对象，这部分对象是可以被免费利用的。

交叉引用表里的空闲条目是一个链表形式，这个链表里每一个空闲条目包含下一个对象数。交叉引用表里的第一个条目（对象数是 0）总是空闲并且代号是 65535；它是交叉引用链表里的头。最后一个空闲条目（链表里的尾）连接回到对象 0。（需要强调的是，这个表里也许还会包含其他的空闲条目，这些空闲条目也能连接回对象 0，也有一个 65535 的代号，即使这些入口不在链表里。）

除了对象 0，其它的所有在交叉引用表里的对象最初的代号都是 0.当一个间接对象被删除，它的交叉引用条目被标记为空闲，并且会增加到空闲条目的链表里。当一个对象数被创建时，下一次使用这个对象的时候它的代号会自动加 1.然而，每一次条目被重复使用时，都会被赋予一个新的代号。最大的代号是 65535；当一个交叉引用条目达到这个值时，他就再不能被重复使用了。交叉引用表（由原来的交叉引用部分和所有更新的部分）必须包含文件里被使用的从 0 到最大的对象数的一个条目，即使在这个范围内的一个或者多个对象数没有存在于这个文件里。

五、File Trail

PDF Trail 能使一个阅读文件应用迅速的找到交叉引用表和某些特殊的对象。应用程序应该从尾部开始读。文件的最后一行仅包含文件尾标志，%%EOF. 之前的那两行包含关键字 startxref 和从文件开始到最后一个交叉引用部分的第一个关键字 xref。stratxref 被 trailer 字典推行，由关键字 trail 和尖括号里的一系列的键值对结束。

六、增量更新（Incremental Uptades）

一个 PDF 文件的内容在不需要重写整个文件的条件下可以增量更新。更新的部分会被加到文件的最后，并且可以完整的留下它的原始内容。用这种方式来更新文件最主要的优点就是在一个大文档里做一些小的改动能够很快被保存。下面还有一些额外的好处：

1、在某些情况下，对一个文档来说增量更新是唯一一种去存储更新的方法。当存储一个文档时，最小风险的减少数据丢失的办法是将它写到一个新的文件中并且重命名这个新

文件来代替这个旧文件。然而，某些情况下，像通过 HTTP 链接或者 OLE 嵌入的方式来编辑文档时，是不可能使用这种方式去覆盖原始文档的内容的。而在这些情况下可以使用增量更新的方式来存储更新。

2、一旦一个文档被签名，文档的所有改变必须使用增量更新的方式去保存更新，由于改变了文件里的任意存在字节，所以使得存在的签名变得无效。

在一个增量更新里，任意新的或者被更新的对象都被附加到文件里，一个交叉引用部分被添加，一个新的 trailer 被插入。

当一个文件被更新时，添加的交叉引用部分包括的条目仅仅是那些被改变的、被替代的或者被删除的那些对象。被删除的对象在文件未改变时就去掉了，因为在他们的交叉引用条目里被标记为删除。被添加的 trailer 包含所有的原有的 trailer，和一个 prev 条目一样，被原来的交叉引用部分定位了。一个文件被更新了几次，它就包含几个 trailers，每个 trailer 以自己的文件尾标志来宣告终止。

因为更新被加到一个 PDF 文件中，在一个文件里，一个对象会有一些复制品，这些复制品含有相同的对象标识符（对象号和代号）。这种情况会发生在以下情况，例如，如果一个文本注释被改变了几次并且这个文件是在这些改变之间保存的。因为文本注释对象没有被删除，它依旧包含了相同的对象号和代号。一个更新后的版本对象被列入到了一个文件的新加入的更新部分里。更新的交叉引用部分包含对于一个对象的新版本来说会有一个字节的偏移，覆盖旧的字节偏移量被包含在了原先的交叉引用部分里。当一个应用程序读文件时，他必须建立他自己的交叉引用信息，用这种方式才能使每个对象的最近的复制成为文件里访问的那一个。

在 PDF1.4 之前的版本，自从版本号要求仅在文件的开始的 header 被指定，不可能使用增量更新的方式去改变 PDF 的版本号来使文档合格。所以，使用增量更新，使使用一个文档里的目录字典里的 Version 条目来更改在 header 被定义的版本号成为可能。

七、对象流（Object Stream）

PDF1.5 介绍了一种新的流，对象流，包含一系列的 PDF 对象。对象流的目的是允许一个更大数目的 PDF 对象能够被压缩，来大幅度的减小 PDF 文件的大小。流里的对象作为被压缩的对象被释放。间接引用对象流里的对象使用正常语法：例如，140R。进入这些对象存储交叉引用信息需要不同的方式；详见 3.4.7，交叉引用流。虽然一个应用必须支持

PDF1.5 才能使用压缩对象，对象能够以这种方式来存储的话，就能够兼容 PDF1.4.不支持 PDF1.5 的应用能忽略对象；PDF 文件的创造者的灵活性在于确定哪些对象，如果有，储存在对象流里。例如，它能很方便的将拥有相同特性的对象储存在一起，像“第一页的字体”，或者“draft#3 的评论。”这些对象被称为一个集合。

为了避免性能降低，像当下载和解压一个大的对象流去接近一个单一的被压缩的对象，单个对象流的对象数应该被限制。他需要一组以集合方式来被连接的对象流，在字典里用 Extends 条目能够被实现。

当一个集合将要把新的对象收集起来更新时，可以用 Extends。宁可重新定义原先的对象流，他可能需要重新复制流数据，新对象在新的对象流里能够被储存。当在文档里添加一个更新部分时，这点尤为重要。

一个对象流的代数和任何被压缩对象的代数都被默认为零。不管是一个对象流还是一个被压缩对象被删除并且对象号被释放，那他的对象号就能被重新使用仅仅对于普通对象而言，而不是对象流。当新的对象流和新的压缩对象被创建时，他们必须一直被定义为新的对象号，就的对象好不会从 free list 里被拿走。

七、交叉引用流（Cross-Reference Streams）

从 PDF1.5 开始，交叉引用信息被储存在一个交叉引用流里来替代在一个交叉引用表里。交叉引用流提供了以下几个好处：

1、一个更紧凑的交叉引用信息；

2、进入压缩对象的能力被储存在对象流里，并且允许新的交叉引用条目类型的添加；

交叉引用流是流对象，它包含一个字典和一个数据流。每一个交叉引用流包含的信息等同于交叉引用表和一个交叉引用部分的结束标志。trailer 字典项被储存在流字典里，交叉引用表条目以一个流数据的形式来储存。

对于文件来说全部用交叉引用流，关键字 xref 和关键字 trailer 不再使用了。然而，除了 startxref 地址和%%EOF 段和注释外，一个 PDF1.5 文件完全是一个一系列的对象。对象流和交叉引用流的作用在线性的 PDF 中是被限制的，在说明书中有小的修改。

字典流（Cross-Reference Stream Dictionary）

交叉引用流包含的条目在Table3.15中体现除了所有流的常见条目和trailer字典项外。自从交叉引用流里的一些信息被应用程序去构建允许间接引用被解除的指标，交叉引用流里的条目有以下限定：

1、Table3.15 里的所有条目的值必须是直接对象，间接引用被禁止。对于数组，所有的元素也必须是直接对象。如果流被编码，Filter 和 DecodeParms 条目在 Table3.4 里的也必须是直接对象。

2、Table3.15 里没有被列出来的其它的交叉引用流条目也许是间接的，事实上，有一些被要求是间接的。

交叉引用流一定不能被加密，并且任何字符串也不能出现在交叉引用流字典里。它必须没有一个 Filter 条目，这个 filter 条目被指定是一个 Crypt filter。

交叉引用数据流（Cross-Reference Stream Data）

在交叉引用流里的每一个条目都拥有一个或者多个字段，第一个命名为条目类型。在PDF1.5 里，只有类型 1 和 2 被允许。任何其它的值被解释成一个空对象的引用，因此允许新的条目类型在将来被定义。

字段号是以递增的规律写的；每个字段的长度被在 W 条目里的对应值来被定义。需要超过一个字节的字段首先被以高字节储存。

像任何流一样，一个交叉引用流是一个间接对象。然而，他的一个条目必须存在在一个交叉引用流或者一个交叉引用表里。

兼容性（Compatibility）

不支持 PDF1.5 的应用程序不能进入被交叉引用流引用的对象。如果一个文件仅只用交叉引用流，那么它就不能被不支持 PDF1.5 的应用程序打开。

然而，有可能去建立一个被叫做 hybrid-reference 文件可以使用 PDF1.5 来阅读。这样的一个文件包含的对象被标准的交叉引用表引用，除了在对象流里的被交叉引用流引用的对象。

在这些文件中，trailer 字典能包括，除了 Table3.13 里列出来的 trailers，一个附加条

目，像 Table3.17 里列出来的。这个条目被 PDF1.4 用户们所忽略，因此没有进入交叉引用流里的条目所引用的条目。

trailer 的 Size 条目一定要足够大去包含所有的对象，包括这些被 XRefStm 条目在交叉引用流里被引用的所定义的。然而，为了允许随机进入，一个主要的交叉引用部分必须包含所有对象号由于 Size-1 而变成 0 的条目。然而，XRefStm 条目在 trailer 字典里被使用，这个 trailer 字典是交叉引用部分的核心但是仅仅在一个更新交叉引用部分。

当一个 PDF1.5 的用户打开了一个 hybrid（混合）-reference 文件，对象的条目在交叉引用流里没有被隐藏。当应用程序为一个对象搜索时，如果一个条目在提供的任何标准的交叉引用部分都没有被找到时，交叉引用流的搜索进程在现有的交叉引用部分里查找前被指定成 XRefStm 条目。

被隐藏对象，然而，有 2 个交叉引用条目。一个在交叉引用流里，另一个在一些现有的部分里是一个自由条目，被 Prev 条目引用的具有代表性的部分。一个 PDF1.5 的用户首先在交叉引用流里查找，找到对象在那，就忽略现有部分里的自由条目。一个 PDF1.4 用户忽略交叉引用流并且是在现有部分去查找，在那儿去查找自由条目。那个自由条目必须有一个超过 65535 的代数号，那样才能保证对象号不会被重复。文件里有混合引用对象时，在 PDF1.4 或更早的用户里一些对象可能会被隐藏并且不会让文件显示。特别之处在于，文件路径，文档目录一定不会被隐藏，包括路径里的任何可视对象。