总结XML内容（对内容写一个综述）

1.  eXtensible Markup Language  （可扩展的标记语言）。  
2.  Document Type Definition  （DTD，文档类型定义）  
3.  XML 中的每个元素都是成对出现的（有开始，有结束）， <student> </student>，  
XML 中的元素嵌套关系要保持正确性，即先开始的标记要先结束，后开始的  
标记要后结束。  
4.  每一个 XML 文档都有且只有一个根元素（Root Element）。 所谓根元素，就是  
唯一一个包含了其他所有元素的元素。  
5.  XML 描述的是文档的内容与语义，而不是文档应当如何显示。

6.  对于 XML 文档来说，<?处理指令必须要顶格写，前面不能有任何的空白。  
7.  XML 元素可以具有属性，属性的形式为：  
属性名=”属性值”，比如 gender=”male”  
属性值需要使用单引号或双引号括起来。多个属性之间使用空格分开。  
8.  通过样式，我们可以实现 XML 内容与展现形式的分离  
9  XML 的注释：<!-- comments  -->  ，注释不允许嵌套  
10.  在一个元素上，相同的属性只能出现一次。  
11.  如何定义和使用实体  
<?xml version="1.0"?>  
<!DOCTYPE company[  
<!ENTITY shengsiyuan "圣思园">  
<!ENTITY address "北京上地">  
]>  
<company>  
<name>&shengsiyuan;</name>  
<address>&address;</address>  
</company>  
12.  格式正规（well formed）的 XML 文档。符合 XML 语法要求的 XML 文档就是  
格式正规的 XML 文档。  
13.  有效的（valid）XML 文档。首先 XML 文档是个格式正规的 XML 文档，然后  
又需要满足 DTD 的要求，这样的 XML 文档称为有效的 XML 文档  
14.  #PCDATA  （Parsed Character Data） ，可解析的字符数据。  
15.  关于普通实体与参数实体  
1）  普通实体是在 dtd 中定义，xml 中使用，使用的时候格式为：&address;  
2）  参数实体是在 dtd 中定义，dtd 中使用，定义的时候使用%,使用的时候也  
需要使用%, %address;

3）  外部普通实体的定义方式是：  
<!ENTITY address SYSTEM “http://www.shengsiyuan.com/xml.xml”>  
表示使用 [http://baidu.com/xml.xml](http://www.shengsiyuan.com/xml.xml) 网址的内容来替换掉  
address，而不是使用 [http://www.google.com/xml.xml](http://www.shengsiyuan.com/xml.xml) 字符串本身来  
替换掉 address  
4）外部参数实体：  
<!ENTITY % address SYSTEM “http://www.shengsiyuan.com/xml.xml”>，使用  
的时候：%address;

16：SimpleType 与 ComplexType 的区别：  
1）  SimpleType 类型的元素没有子元素，也没有属性。  
2）  当需要定义的元素包含了子元素或者属性时，必须要使用 ComplexType。  
17.  SimpleContent，用于 ComplexType 元素上，用于限定该 ComplexType 的内容  
类型，表示该 ComplexType 没有子元素，同时该 ComplexType 需要有属性，否则它就成为 SimpleType 了。  
18：要严格区分 XML 文档树中的根结点与根元素结点：根节点（Document）代  
表的是 XML 文档本身，是我们解析 XML 文档的入口，而根元素结点则表示  
XML 文档的根元素，它对应于 XML 文档的 Root。

19：JAXP（Java API for XML Parsing）:用于 XML 解析的 Java API。  
20：DocumentBuilderFactory 与 DocumentBuilder 之间的关系。

21：SAX（Simple APIs for XML） ，面向 XML 的简单 APIs。  
22 ：使用 DOM 解析 XML 时，首先将 XML 文档加载到内存当中，然后可以通过随  
机的方式访问内存中的 DOM 树； SAX 是基于事件而且是顺序执行的，一旦经  
过了某个元素，我们就没有办法再去访问它了，SAX 不必事先将整个 XML 文  
档加载到内存当中，因此它占据内存要比 DOM 小，对于大型的 XML 文档来  
说，通常会使用 SAX 而不是 DOM 进行解析。  
23：  SAX 也是使用的观察者模式（类似于 GUI 中的事件） 。