




北京圣思园科技有限公司
<http://www.shengsiyuan.com>

主讲人：张龙

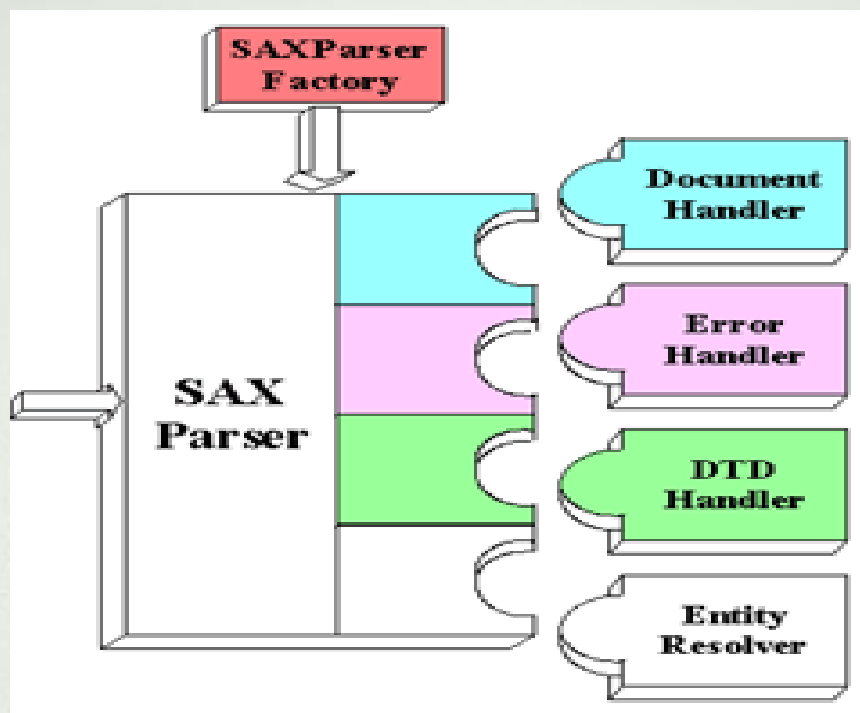
SAX

SAX的全称是**Simple APIs for XML**，也即**XML**简单应用程序接口。与**DOM**不同，**SAX**提供的访问模式是一种顺序模式，这是一种快速读写**XML**数据的方式。当使用**SAX**分析器对**XML**文档进行分析时，会触发一系列事件，并激活相应的事件处理函数，应用程序通过这些事件处理函数实现对**XML**文档的访问，因而**SAX**接口也被称作事件驱动接口。

SAX分析器在对**XML**文档进行分析时，触发了一系列的事件，由于事件触发本身是有时序性的，因此，**SAX**提供的是一种顺序访问机制，对于已经分析过的部分，不能再倒回去重新处理。**SAX**之所以被叫做"简单"应用程序接口，是因为**SAX**分析器只做了一些简单的工作，大部分工作还要由应用程序自己去做。也就是说，**SAX**分析器在实现时，它只是顺序地检查**XML**文档中的字节流，判断当前字节是**XML**语法中的哪一部分、是否符合**XML**语法，然后再触发相应的事件，而事件处理函数本身则要由应用程序自己来实现。同**DOM**分析器相比，**SAX**分析器缺乏灵活性。然而，由于**SAX**分析器实现简单，对内存要求比较低，因此实现效率比较高，对于那些只需要访问**XML**文档中的数据而不对文档进行更改的应用程序来说，**SAX**分析器更为合适。



SAX分析器的大体构成框架



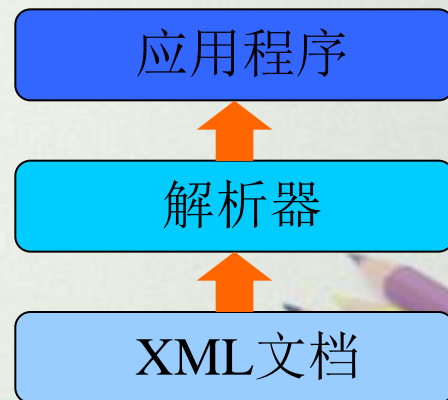
图中最上方的**SAXParserFactory**用来生成一个分析器实例。XML文档是从左侧箭头所示处读入，当分析器对文档进行分析时，就会触发在**DocumentHandler**, **ErrorHandler**, **DTDHandler**以及**EntityResolver**接口中定义的回调方法。

解析器基础

- XML解析器实际上就是一段代码，它读入一个XML文档并分析其结构。

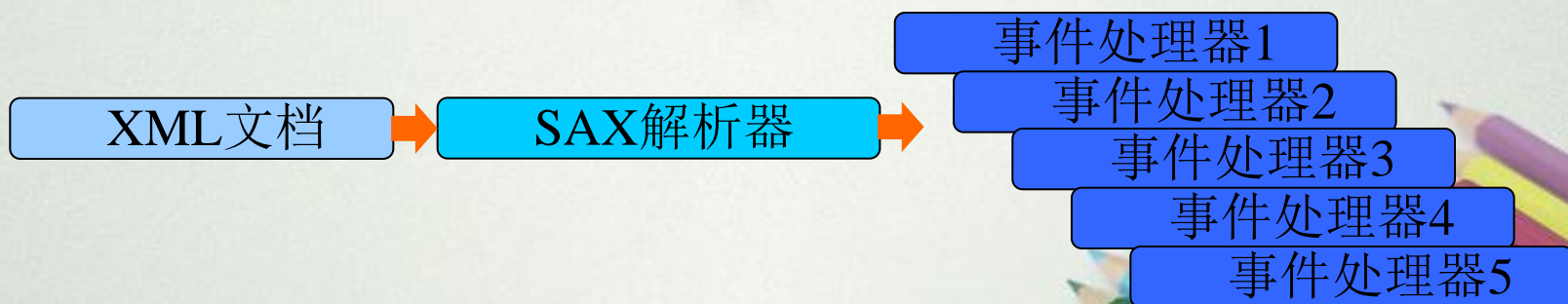
- 分类：

- 带校验的解析器
- 不校验的解析器（效率高）
- 支持DOM的解析器（W3C的官方标准）
- 支持SAX的解析器（事实上的工业标准）



SAX基础

- **SAX**是事件驱动的，文档的读入过程就是**SAX**的解析过程。
- 在读入的过程中，遇到不同的项目，解析器会调用不同的处理方法。



常用的事件处理方法


org.xml.sax.helpers.DefaultHandler 类的方法

项目	处理方法
文档开始	startDocument()
<PEOPLE>	startElement()
“Tony Blair”	characters()
</PEOPLE>	endElement()
文档结束	endDocument()



SAX方式提取XML文档内容

```
import javax.xml.parsers.*;
import org.xml.sax.*;
import org.xml.sax.helpers.*;
public class sax extends DefaultHandler {
    Stack tags=new Stack();
    public void endDocument() throws SAXException { System.out.println("-----
        Parse End-----"); }
    public void startDocument() throws SAXException { System.out.println("-----
        Parse Begin-----"); }
    public void startElement(String p0, String p1, String p2, Attributes p3) throws
        SAXException {
        tags.push(p2);
    }
    public void endElement(String p0, String p1, String p2) throws SAXException {
        tags.pop();
        if (p2.equals("PERSON")) printout();
    }
    public void characters(char[] p0, int p1, int p2) throws SAXException {
        String tag=(String) tags.peek();
        if (tag.equals("NAME")) name=new String(p0,p1,p2);
        else if (tag.equals("ADDRESS")) address=new String(p0,p1,p2);
        .....
    }
}
```



SAX方式提取XML文档内容

```
static public void main(String[] args) {  
    String filename = "candidate.xml";  
    SAXParserFactory spf =  
        SAXParserFactory.newInstance();  
    SAXParser saxParser=null;  
    try {  
        saxParser = spf.newSAXParser();  
        saxParser.parse(new File(filename),new  
sax());  
    } catch (Exception e) {  
        System.out.println(e);  
        System.exit(1);  
    }  
}  
}
```

