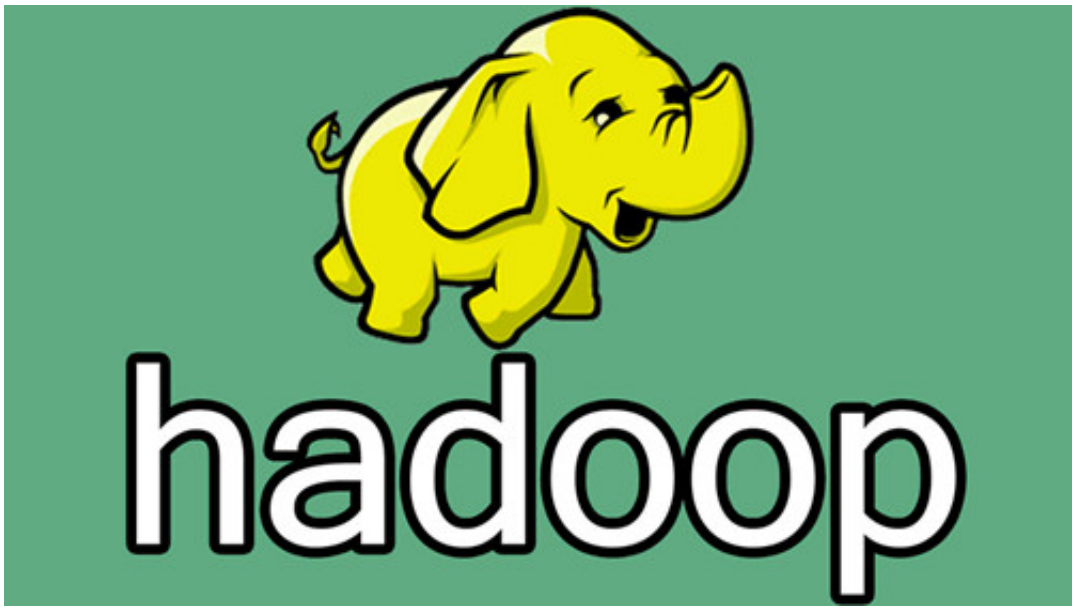
使用Hadoop Configuration一些需要注意的细节

我们在使用Hadoop、Spark或者是Hbase，最常遇到的问题就是进行相关系统的配置，比如集群的URL地址，MapReduce临时目录、最终输出路径等。这些属性需要有一个系统（类）进行管理。然而，Hadoop没有使用 Java.util.Properties 管理配置文件，也没有使用Apache Jakarta Commons Configuration管理配置文件，而是单独开发了一个配置文件管理类，这个类就是 org.apache.hadoop.conf.Configuration 。它可以解析如下格式的xml文件：

```
<configuration>
   <property>
      <name>color</name>
      <value>red</value>
   </property>
</configuration>
```



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

我们可以
将所有的配置都写
入这种格式的文件中；Hadoop就是这
么做的，可以参见：core-site.xml、mapred-site.xml以及hdfs-site.xml
文件。Hadoop生态系统中有很多API都需要传进一个 Configuration
对象，我们需要在这个对象中传进一些关于系统的配置。通过 Configuration

设置属性的方式主要有以下几种：

```
public void set(String name, String value)
public void addResource(String name)
public void addResource(URL url)
public void addResource(Path file)
public void addResource(InputStream in)
```

通过 set 方法传参只能通过在程序里面进行；而 addResource 方法允许我们将配置写入到文件中，然后后面会解析这个文件里面相关的配置。我们也可以同时通过这两种方式进行相关属性的配置。那么问题来了，同一个 属性通过这两种方式进行配置，最后哪种方式有效？如下：

```
Configuration conf = new Configuration(false);
conf.addResource(new Path("/iteblog.xml"));
conf.set("color", "blue");
System.out.println(conf.get("color"));
```

```
Configuration conf = new Configuration(false);
conf.set("color", "blue");
conf.addResource(new Path("/iteblog/test.xml"));
System.out.println(conf.get("color"));
```

上面两个代码片段主要的区别就是调用 set 和 addResource顺序不一样。大家先猜猜这两个程序的输出。

有趣的是，上面两个代码片段输出的结果一样，都是 blue！（原理见下面）大家再看看下面代码片段的输出是什么：

```
############## iteblog.xml ###############
<configuration>
   <property>
      <name>color</name>
      <value>red</value>
   </property>
</configuration>

############## iteblog1.xml ###############
```

```
<configuration>
  <property>
    <name>color</name>
    <value>yellow</value>
  </property>
</configuration>
```

```
Configuration conf = new Configuration(false);
conf.addResource(new Path("/iteblog.xml"));
conf.addResource(new Path("/iteblog1.xml"));
System.out.println(conf.get("color"));
```

```
Configuration conf = new Configuration(false);
conf.addResource(new Path("/iteblog1.xml"));
conf.addResource(new Path("/iteblog.xml"));
System.out.println(conf.get("color"));
```

上面的输出结果依次为 yellow 和 red ！因为通过 addResource 设置的属性，后面的设置会覆盖前面的，除非前面文件该属性被设置为final（true）。

在 Configuration 类中存在三个比较重要的数据结构：Properties overlay、HashMap updatingResource 以及 private Properties properties

- overlay：只有通过调用 set 方法设置的属性才会写入到这里面；
- updatingResource：这里面主要记录的是各个属性的来源。不管程序调用 addResource 和 set 的顺序如何，最后通过 set 设置的属性来源是 programatically（这个名字可以修改）；
- properties：这里面主要存的是通过 addResource 和 set 方式设置属性的交集。不管程序调用 addResource 和 set 的顺序如何，最后一定是通过 set 方式设置属性的值覆盖通过 addResource 复制的值。

最后总结如下：

- 通过 set 设置的属性优先级比通过 addResource 设置要高；
- 通过相同方式（都是通过 set 、或都是通过 addResource）设置属性，后面的会覆盖掉前面的设置；但是如果都是通过 addResource 设置，而且前面有些属性使用了final属性，那么后面的设置不能覆盖前面的设置；

- 如果系统加载的配置文件中有相关的属性被设置为final，那么用户程序里面不管是通过 set 还是 addResource 方式设置，都无法覆盖系统的设置。所以Hadoop管理员通常会将一些系统级别的属性设置为final，以防止用户自己修改。
- 如果本文对你有所帮助，请记得转发分享啊。