链滴

SimpleDateFormat 解决SimpleDateFormat非线程安全

作者：cactus0509

原文链接：https://ld246.com/article/1469179020876

来源网站：链滴

大致意思：Tim Cull碰到一个SimpleDateFormat带来的严重的性能问题，该问题主要有SimpleDateormat引发，创建一个 SimpleDateFormat实例的开销比较昂贵，解析字符串时间时频繁创建生命周短暂的实例导致性能低下。即使将 SimpleDateFormat定义为静态类变量，貌似能解决这个问题，但SimpleDateFormat是非线程安全的，同样存在问题，如果用 ‘synchronized’ 线程同步同样面临题，同步导致性能下降（线程之间序列化的获取SimpleDateFormat实例）。

Tim Cull使用Threadlocal解决了此问题，对于每个线程SimpleDateFormat不存在影响他们之间协作状态，为每个线程创建一个SimpleDateFormat变量的拷贝或者叫做副本，代码如下：

[java] view plain copy

```
import java.text.DateFormat;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

/**
 * 使用ThreadLocal以空间换时间解决SimpleDateFormat线程安全问题。
 *
 * @author
 *
 */
public class DateUtil {

    private static final String DATE_FORMAT = "yyyy-MM-dd HH:mm:ss";

    @SuppressWarnings("rawtypes")
    private static ThreadLocal threadLocal = new ThreadLocal() {
        protected synchronized Object initialValue() {
            return new SimpleDateFormat(DATE_FORMAT);
        }
    };

    public static DateFormat getDateFormat() {
        return (DateFormat) threadLocal.get();
    }

    public static Date parse(String textDate) throws ParseException {
        return getDateFormat().parse(textDate);
    }
}
```

创建一个ThreadLocal类变量，这里创建时用了一个匿名类，覆盖了initialValue方法，主要作用是创时初始化实例。也可以采用下面方式创建；

[java] view plain copy

```
import java.text.DateFormat;
import java.text.SimpleDateFormat;

/**
 * 使用ThreadLocal以空间换时间解决SimpleDateFormat线程安全问题。
 *
```

```
 * @author
 *
 */
public class DateUtil {

    private static final String DATE_FORMAT = "yyyy-MM-dd HH:mm:ss";

    // 第一次调用get将返回null
    private static ThreadLocal threadLocal = new ThreadLocal();

    // 获取线程的变量副本，如果不覆盖initialValue，第一次get返回null，故需要初始化一个Simple
ateFormat，并set到threadLocal中
    public static DateFormat getDateFormat() {
        DateFormat df = (DateFormat) threadLocal.get();
        if (df == null) {
            df = new SimpleDateFormat(DATE_FORMAT);
            threadLocal.set(df);
        }
        return df;
    }

}
```

我们看下我们覆盖的initialValue方法：

[java] view plain copy

```
protected T initialValue() {
        return null;//直接返回null
}
```

当然也可以使用：

apache commons-lang包的DateFormatUtils或者FastDateFormat实现，apache保证是线程安全，并且更高效。

ThreadLocal参考：http://blog.csdn.net/partner4java/article/details/7107217