

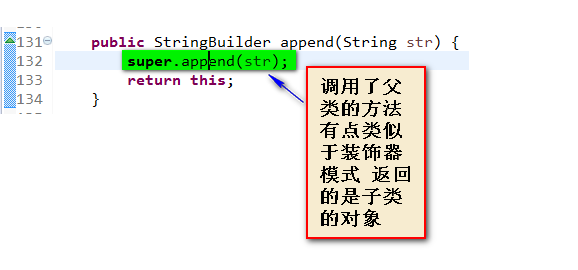
通常Director是要调用Builder的方法生成子组件 最后Director对外提供给接口 生成Build的对象

以StringBuilder#append(String) 为例

我认为StringBuilder已经把这个给简化成 Builder和Director二合一

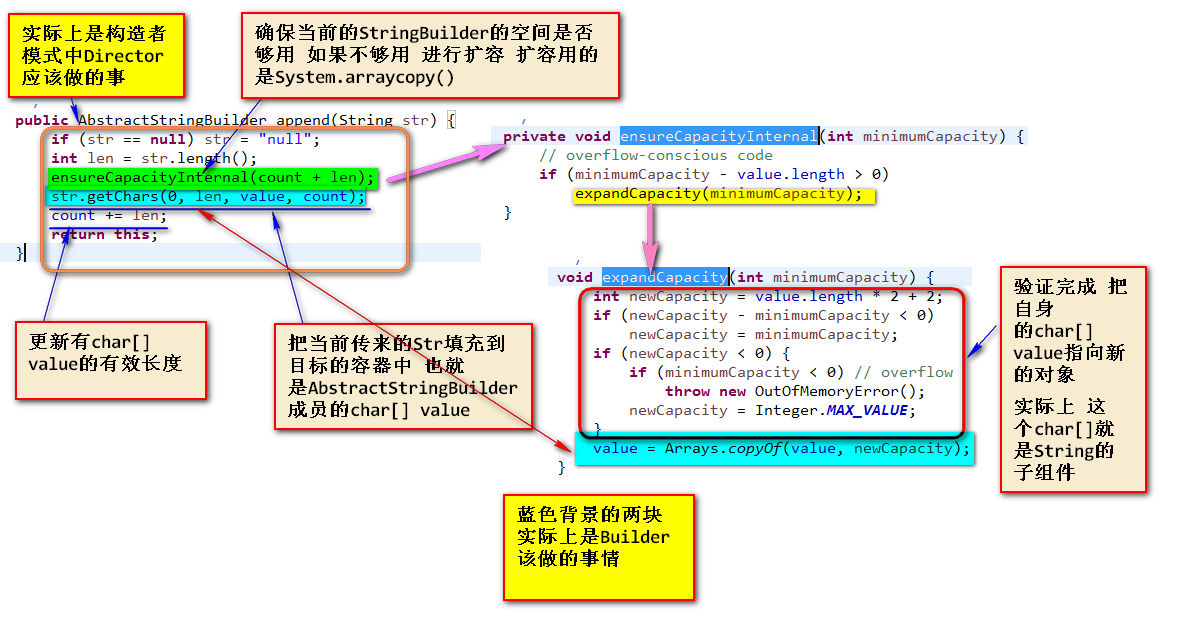
这样 Builder中既要负责生成子类的组件 也要有Director的装配作用

实际上 这个StringBuilder的父类是AbstractStringBuilder



进入到这个父类的append里面看

这个append实际上进行了两次调用



这样看来 AbstractBuilder中的append方法在流程中 实际上扮演了Director的作用 --- 装配最终对象的子组件

然后 ensureCapacityInternal()和str.getChars()就是Builder的作用

因为 String的核心对象就是字符数组char[]

常理来看 返回的应该是String String是最终的产品类

但是 因为要满足append这种链式法则 stringBuilder.append(xxx) .append(xxx) .append(xxx) .append(xxx)

**因此 我们就append最终生成的产品不是String 而还是StringBuilder**

所以 这里面 Director(AbstractStringBuilder中完成), Builder和Product都是StringBuilder

所以 返回的都是return this;

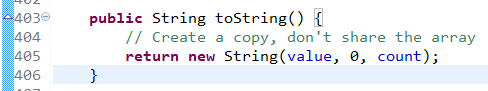
------- 可以看到 StringBuilder的核心组件就是char[] value和 int count

**这样StringBuilder就是自己构建了自己的组件并且组装了自己**

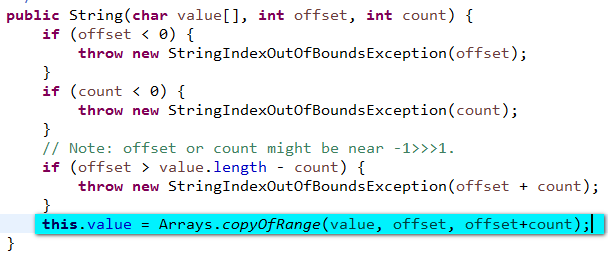
【还可以看到 这个StringBuilder在扩容的时候 如果空间不够 就会频繁复制数组 我认为效率不会太高】

============= StringBuilder的toString()方法 才是 Product是String

---- 这个地方更像具体工厂方法模式



new String的调用过程，最后同样使用了Arrays的数组复制



因为String的子组件同样是char[] value

所以 StringBuilder和 String都是StringBuilder类的产品类