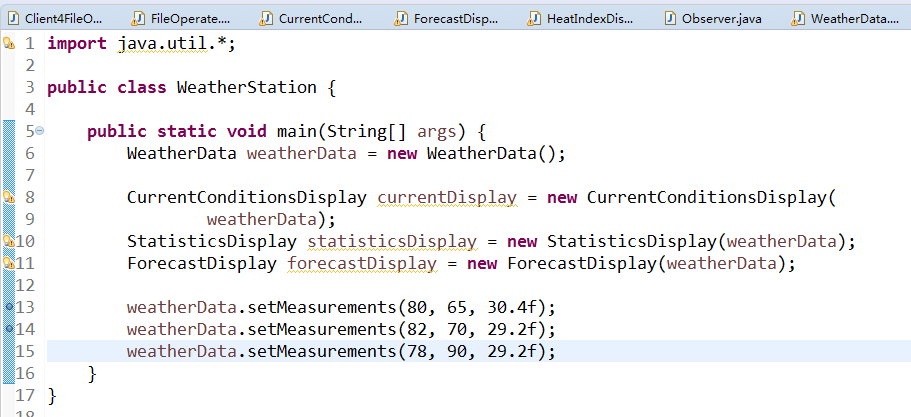
ArrayList中对象获取相关的调试过程

之前使用ArrayList一直是加泛型后使用的，没有注意到不加泛型的情况。上课时给大家的解释是对的，前提是基于泛型的ArrayList。没有强调加与不加泛型的不同之处，给大家造成的理解差异表示道歉，特此声明。

如果ArrayList不加泛型，存储对象的转型情况有所不同。为避免理解错误，特做本文档向大家说明ArrayList中不加泛型时，如何通过ArrayList 对象的get()获取对象。

具体如下：



CurrentConditionsDisplay currentDisplay = **new** CurrentConditionsDisplay(weatherData);

StatisticsDisplay statisticsDisplay = **new** StatisticsDisplay(weatherData);

ForecastDisplay forecastDisplay = **new** ForecastDisplay(weatherData);

以上三句中的构造函数都是如下格式：

public CurrentConditionsDisplay(Subject weatherData) {

this.weatherData = weatherData;

weatherData.registerObserver(this);

}

因此，都可以注册到WeatherData类中的 observers 列表中

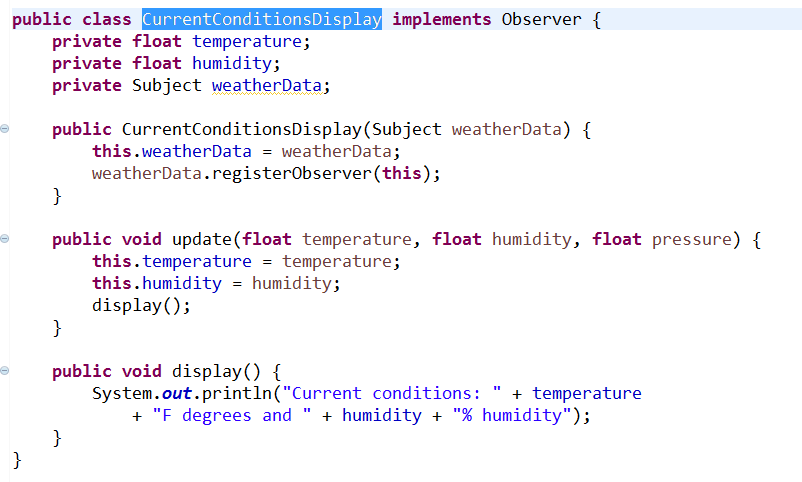
在如下语句中加上断点：

weatherData.setMeasurements(80, 65, 30.4f);

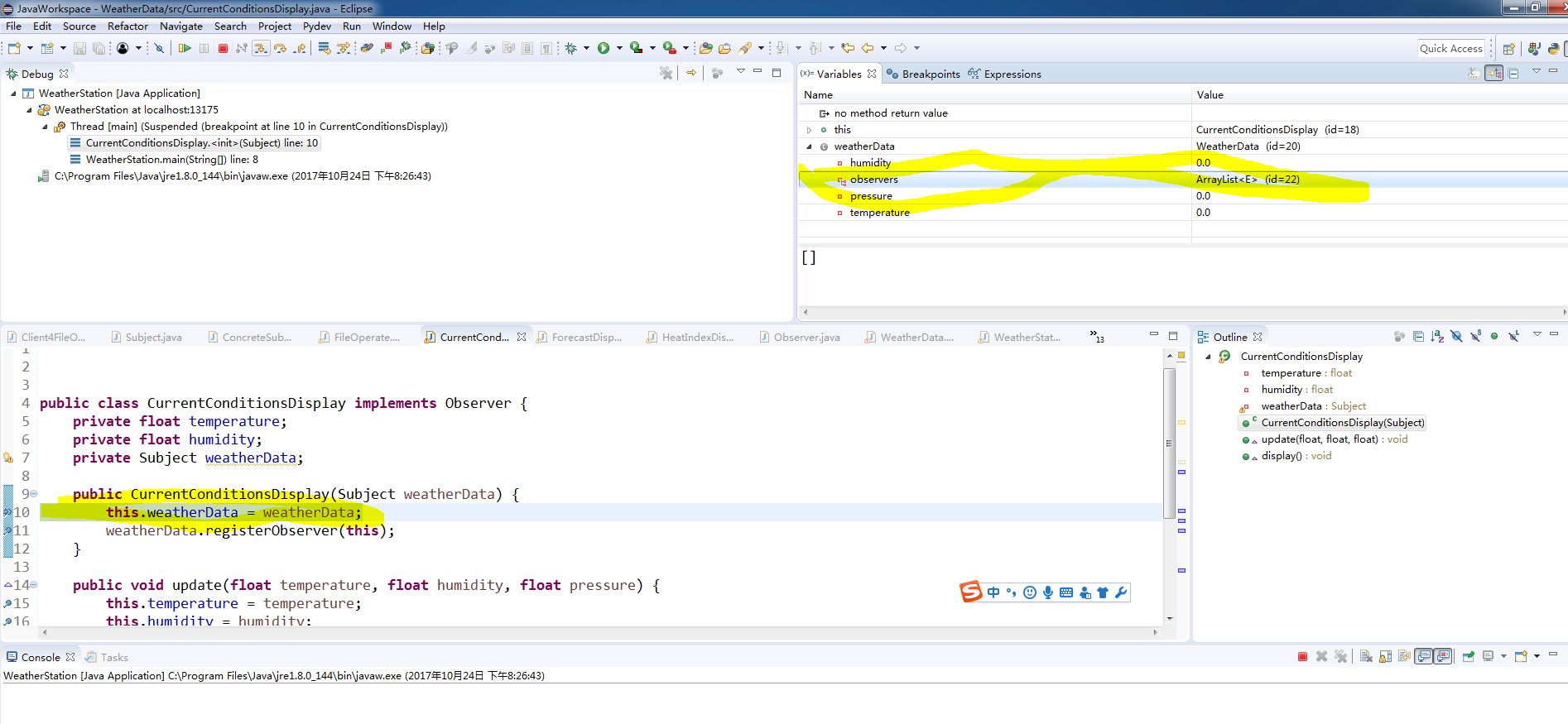
weatherData.setMeasurements(82, 70, 29.2f);

weatherData.setMeasurements(78, 90, 29.2f);

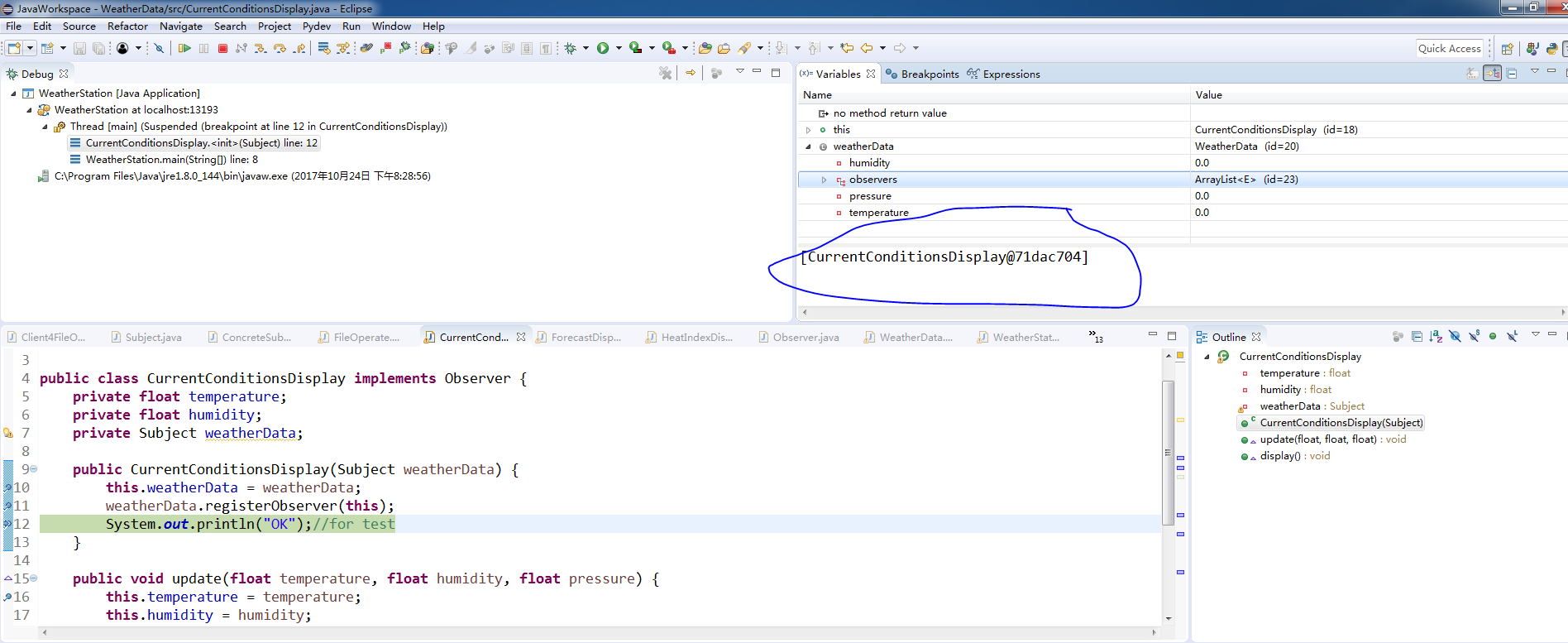
在CurrentConditionsDisplay中加上断点



Debug WeatherData.java

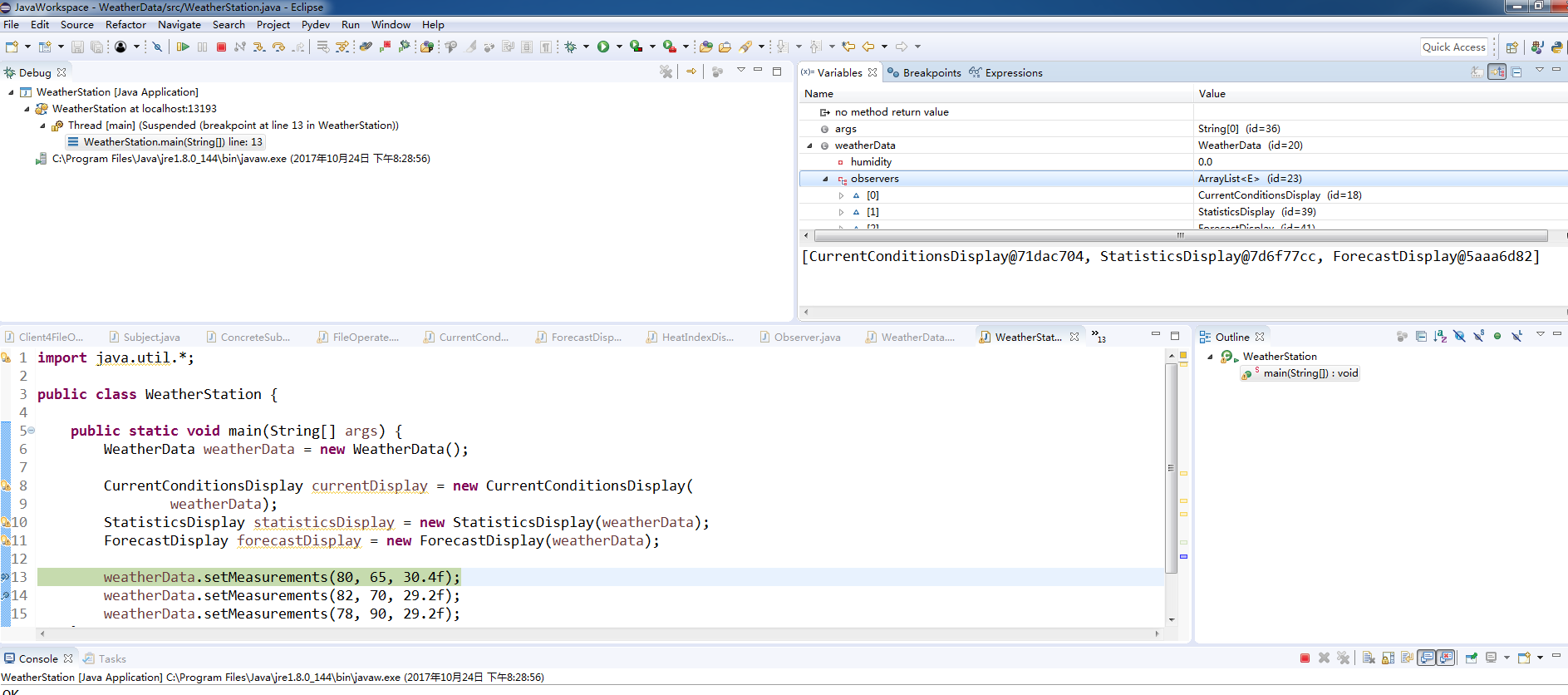


在CurrentConditionsDisplay中执行到构造函数中，执行之前如上图所示。

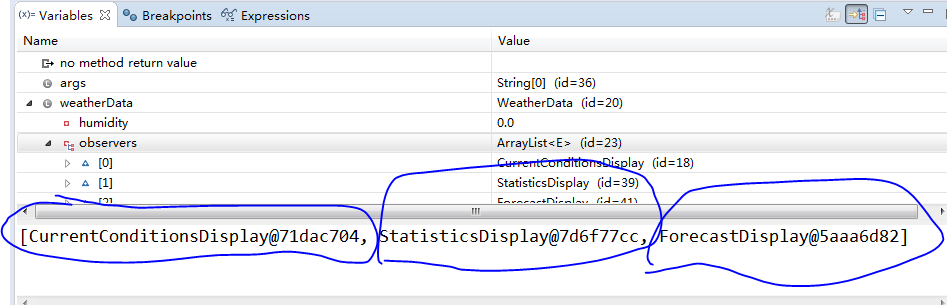


发现CurrentConditionsDisplay对象可以存储到ArrayList中的Observers列表中，并且类型是CurrentConditionsDisplay型

继续

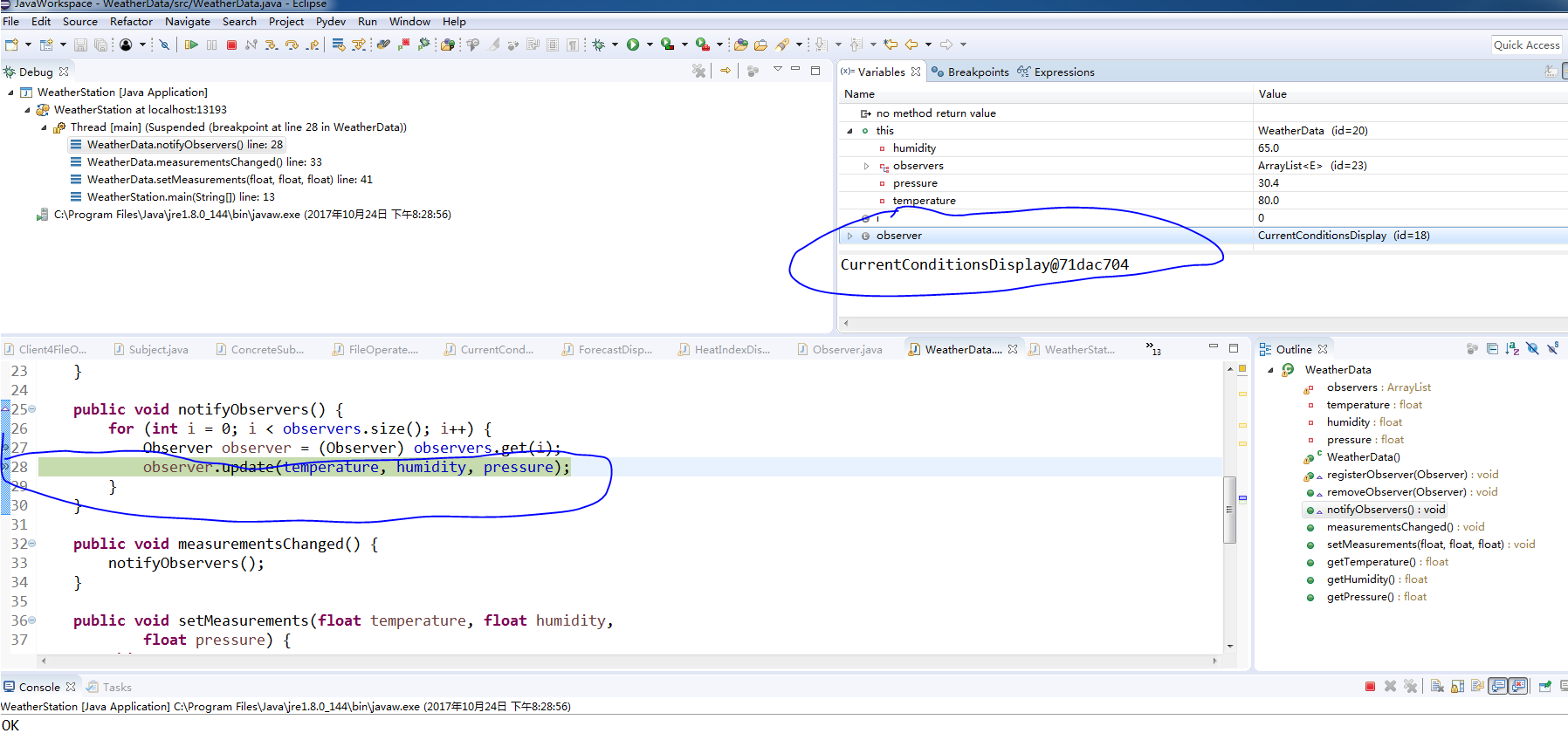


上图状态发现，ArrayList列表中存放了三个子类类型



这点和我原来的预期不一致。原来认为是存储成Observer类型，现在看可以存储成具体子类类型。

继续执行



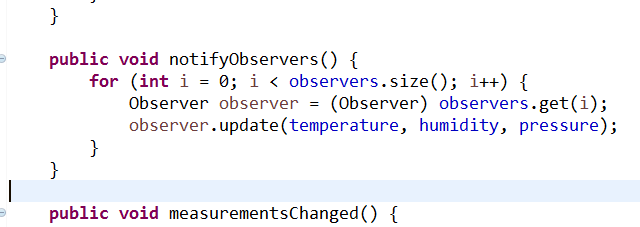
发现可以取出为子类类型！并且可以根据子类类型的定义，正确调用子类对象的方法。

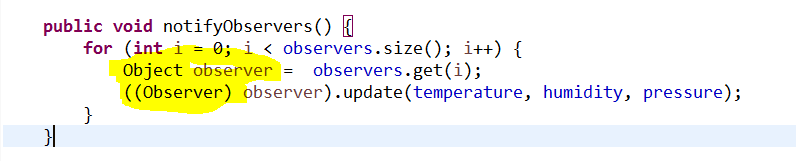
查看《effictive java》等资料，解释如下：

ArrayList内部封装了一个Object类型的数组，从一般的意义来说，它和数组没有本质的差别。如果没设定泛型的话，也可以添加任意不同的类型的对象到ArrayList中，但是不能添加int, long, short, char 等基本类型，如果一定要添加int类型，可先转换为Integer等包装类型。

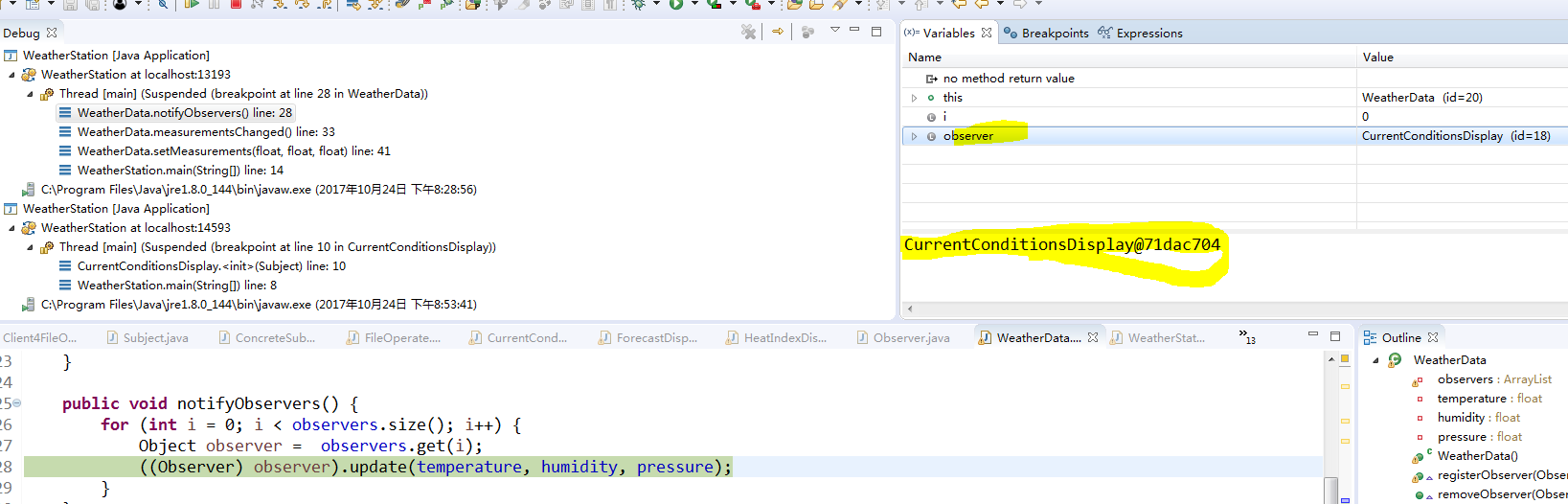
不仅仅是ArrayList，数组如果定义成Object类型，如Object[] objectArray就可以添加所有不同的对象。

但这样的话，数据就不安全，不建议这么做。





即便声明为Object类型，子类也可以在ArrayList中正确存储，并且可以把存储的子类类型对象取出来。



如果不指定类型，与使用Object作为类型相同。如果指定类型，则只能存放该类型以及它的子类型，否则编译器报错。

一般情况下，都是采用泛型定义ArrayList，因为泛型的本质是参数化类型，也就是说所操作的数据类型被指定为一个参数，保证类型安全，无需强转。

参考文献：

<http://www.cnblogs.com/ITtangtang/p/3948555.html#add>

effictive java