# 2018

## 0225-万门——高效读书

### 思维：

目的是休息还是学习？

带着问题去读书

从身边的问题引发读书的兴趣

误区：为了读书而读书，非要从头开始看完。

### 方法：

### 读书顺序：

从哪里开始读？从目录开始，不要从头开始看，重要的内容肯定不是放在开头的。如果在中间遇到关于书中不懂的东西，可以跳到开始找内容。

不要把中学应试读书用到学习中。

### 组合思维：

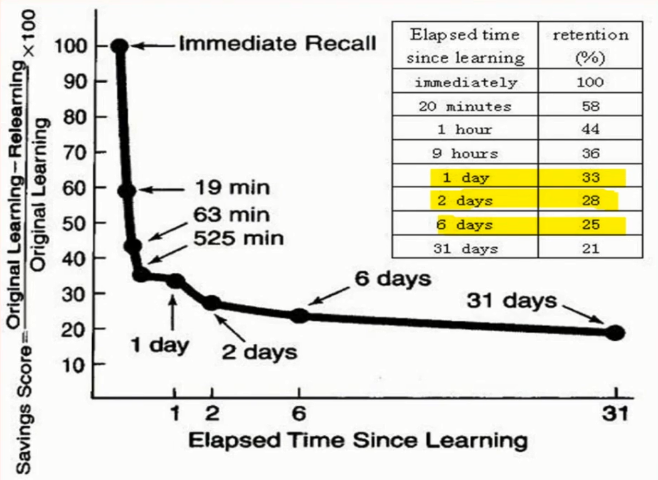
信息包：不同的信息包组合解决问题，形成一本书。

### 输出强化输入：

读书不仅仅是读，否则很容易就忘记了。



1. 笔记：对抗遗忘，检验自己是否真正理解；不要只摘抄原文，用自己的话总结；



1. 分享：说和写
2. 实践：知**行**合一
3. 教别人：

### 工具：

1. 笔记软件
2. 豆瓣
3. 电子书

## 0225-万门——怎样管理好你的时间

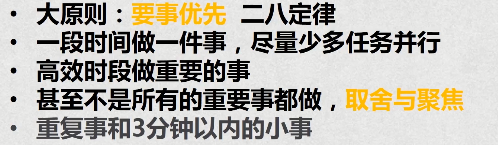
### 时间管理的方法：

### 记录时间去向，盘点时间库存

时间花在哪里不知道

主管时间&客观时间

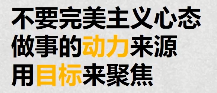
### 事情排优先级，优先重要的事情



### 项目拆分成可执行的时间

### 做好代办事情清单

### 动力来源&目标



## 0225-关于自律

自律也应该有让自己娱乐放松的时间；

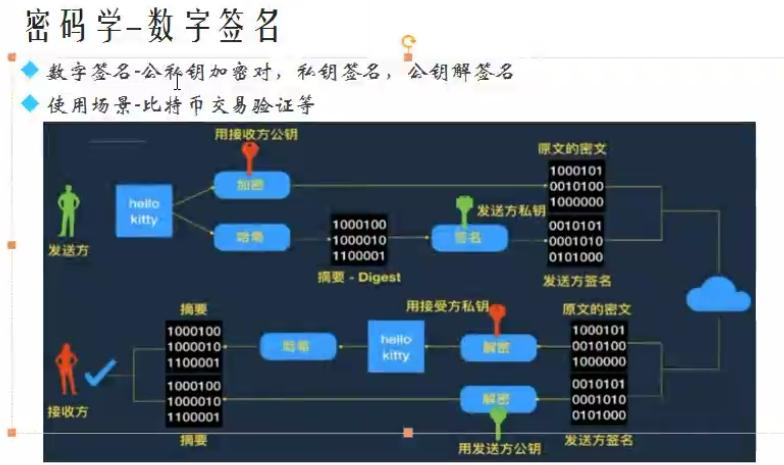
自律这种事情不是照搬照抄别人的，要死人定制；

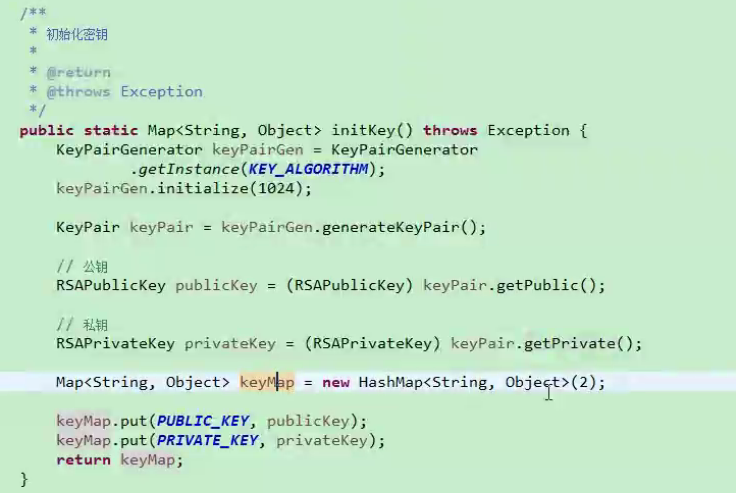
自律不是自虐，不要因为自律让自己身心俱疲。

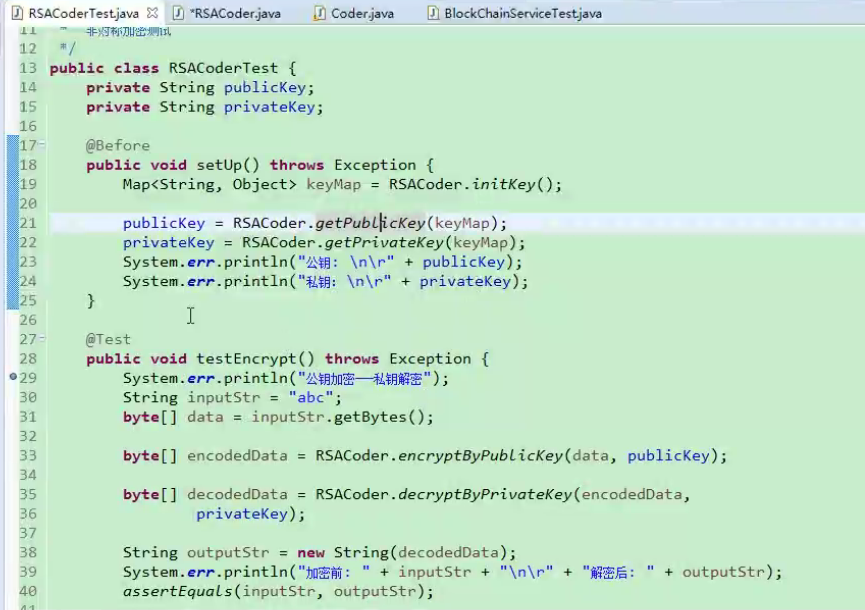
## 0225-图灵学院——密码学



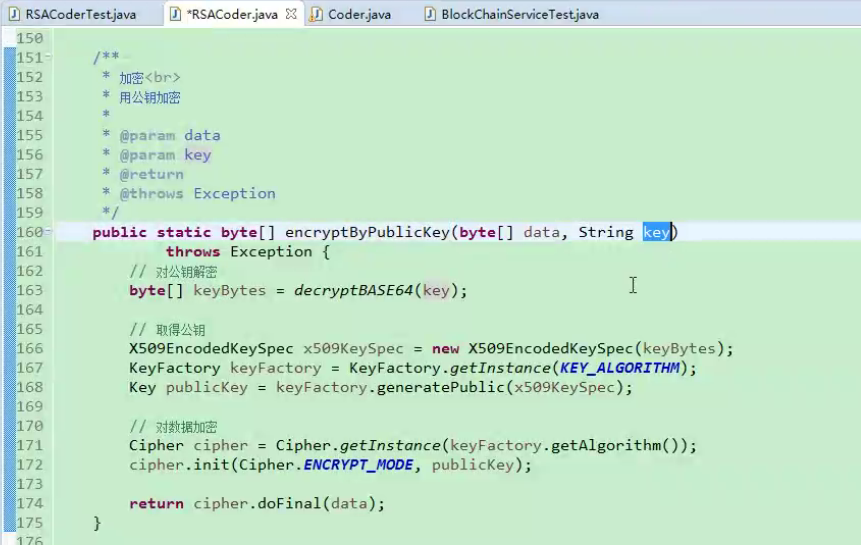




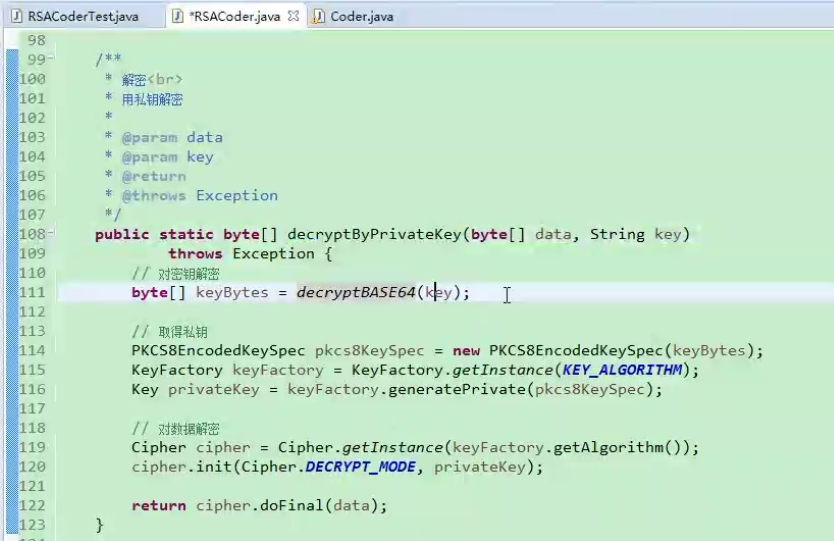




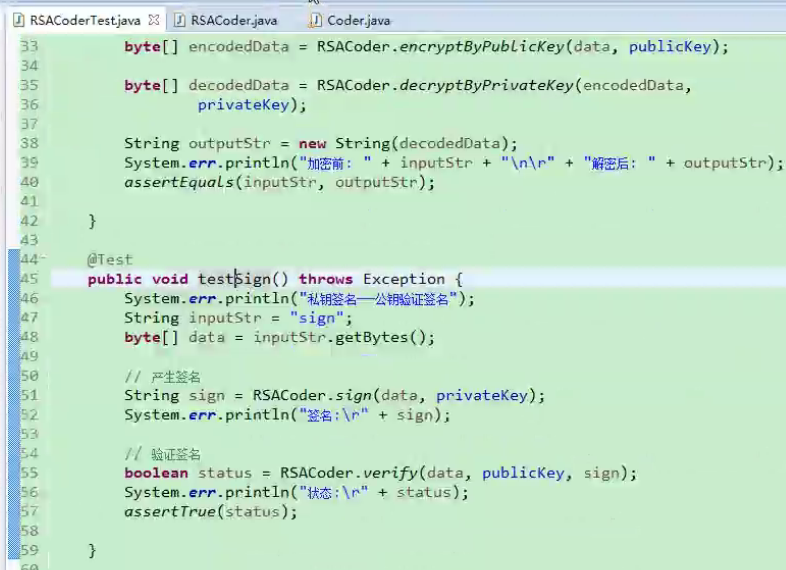
加密方法



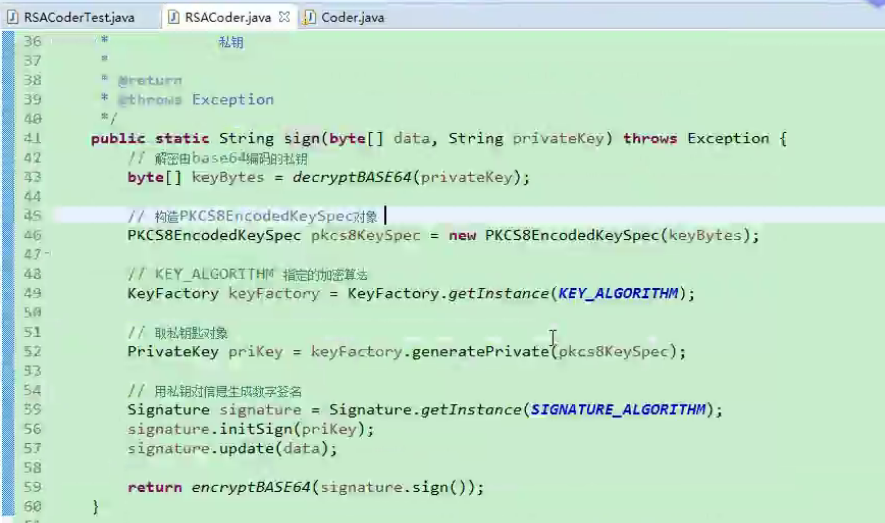
解密方法



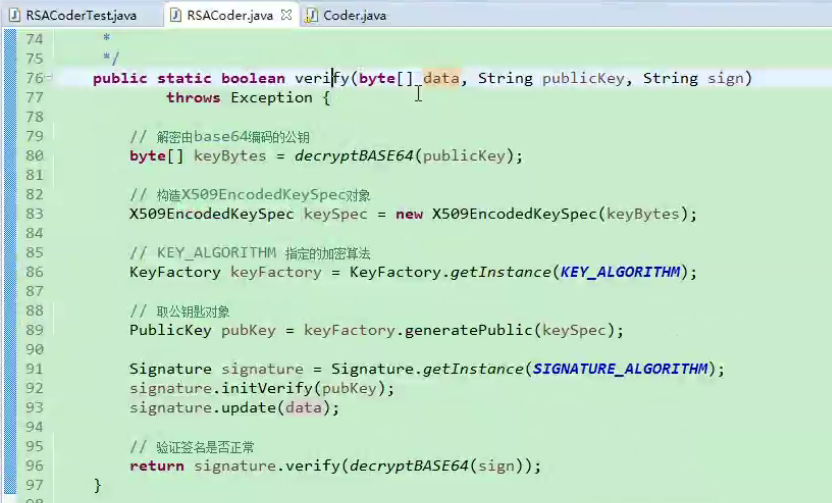
数字签名测试



签名方法



验证签名



## 0226-敏捷开发

敏捷开发思想是短时间高效迭代，人员不多，可以在充分交流的情况下不停的发现问题解决问题。

## 0226-燃尽图

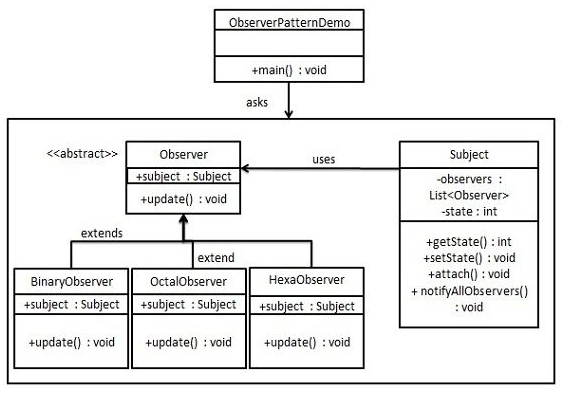


## 0227-观察者模式

首先[文章](http://www.runoob.com/design-pattern/observer-pattern.html)清晰易懂，代码已经加入自己的GitHub（NK\_Utils）中。

被观察者维护一个观察对象的list，因为它要有广播操作；还有约定统一被观察的值。

被观察者可以将观察者每个实例对象放入所维护的list中，并且当被观察者的值变化的时候可以遍历整个list通知每个观察者；每个观察对象都继承自同一个父类，它们有相同的更新方法，关联相同的被观察者；



## 0227-回调函数

Acallbackis**afunction**thatispassedasanargumenttoanotherfunctionandis**executedafteritsparentfunctionhascompleted**。

一篇很好的介绍[文章](https://www.cnblogs.com/lishuxue/p/5999682.html)：

例子：约会结束后你送你女朋友回家，离别时，你肯定会说：“到家了给我发条信息，我很担心你。”对不，然后你女朋友回家以后还真给你发了条信息。小伙子，你有戏了。其实这就是一个回调的过程。你留了个参数函数（要求女朋友给你发条信息）给你女朋友，然后你女朋友回家，回家的动作是主函数。她必须先回到家以后，主函数执行完了，再执行传进去的函数，然后你就收到一条信息了。

代码举例如下：

functionmainFunc(subFunc){

console.log("Main!");

subFunc();

console.log("Main!");

}

functionsubFunc(){

setTimeout("console.log('subFunc!')",3000);

}

mainFunc(subFunc);

## 0227-万门——程序员成长之路

程序员——主管——经理——CTO

每个阶段所需的能力是不一样的，而且没有这种看得见的梯子去跳跃，首先要能够到达这样的能力，再遇到合适的机会，才能成就一些事情。

## 0228-《设计的原则》：

有关设计（同时也是有关生活）的一个更一般的指导原则是：不要畏畏缩缩。

四大原则：

* 亲密性
* 对齐

规则都是用来打破的。但是必须记住，关于打破规则本身也有一条Robin规则：在打破规之前必须清楚规则是什么

什么才是最重要的呢？倾听你的眼睛—眼睛看向哪里，路线是什么，这是最佳路径吗？

对齐的根本目的是使页面**统一而且有条理**。

要避免在页面上混合使用多种文本对齐方式（也就是说，不要将某些文本居中，而另外一些文本右对齐）。

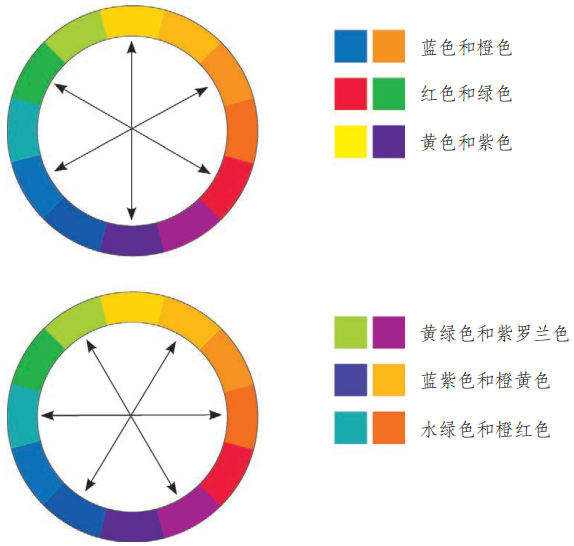
要着力避免居中对齐

* 重复
* 对比

要想实现有效的对比，对比就必须强烈。千万不要畏畏缩缩

页面上的不同元素之间要有对比效果，达到吸引读者的对比效果。

颜色：



 需要印刷的项目应当使用 CMYK；

 需要在屏幕上看的内容则应使用 RGB。

## 0228-工厂模式和抽象工厂模式

今天看了网上代码，照葫芦画瓢搬出来两个设计模式的代码。先谈理解：

**工厂模式：**对有公共性质的类进行抽象，让它们继承自同一个抽象类或者实现同一个接口。抽象类或者接口中就是每个子类应该有的方法，这些方法在不同子类中的实现是有区别的。所谓工厂类就是一个特殊的类，它可以根据传入的参数返回不同的子类，但是函数返回值是父类，这里用到了多态。因此调用工厂类的地方就能够通过父类名称获得某一具体子类的引用，从而调用该子类的方法；

**抽象工厂模式：**比工厂模式更深一层，是工厂类的抽象。路径大概是：实例-->父类-->工厂抽象类-->工厂类的工厂抽象类；层级关系比较简单，但是代码逻辑不太好理解，这里还需要沉淀。

# Studylist

Socket编程