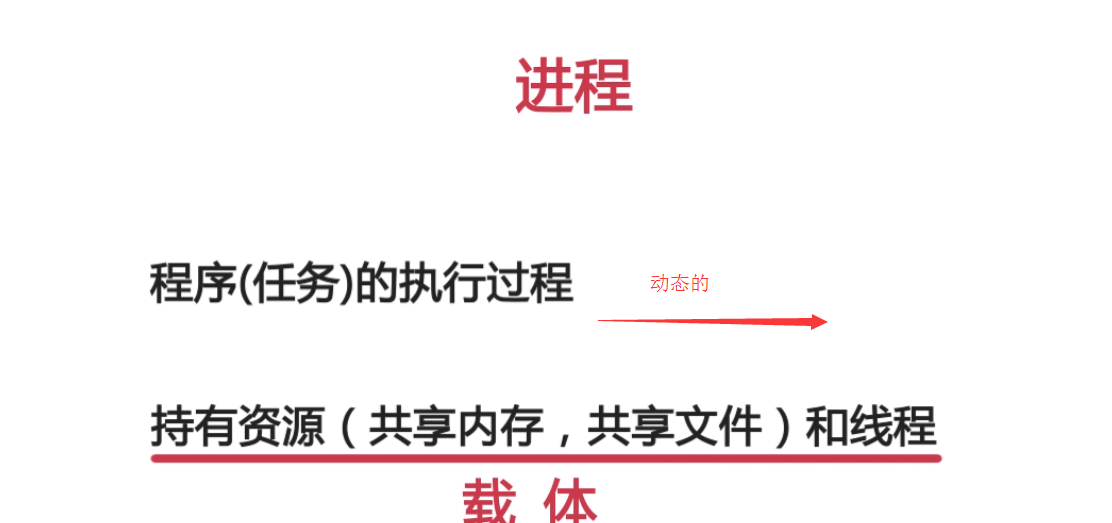
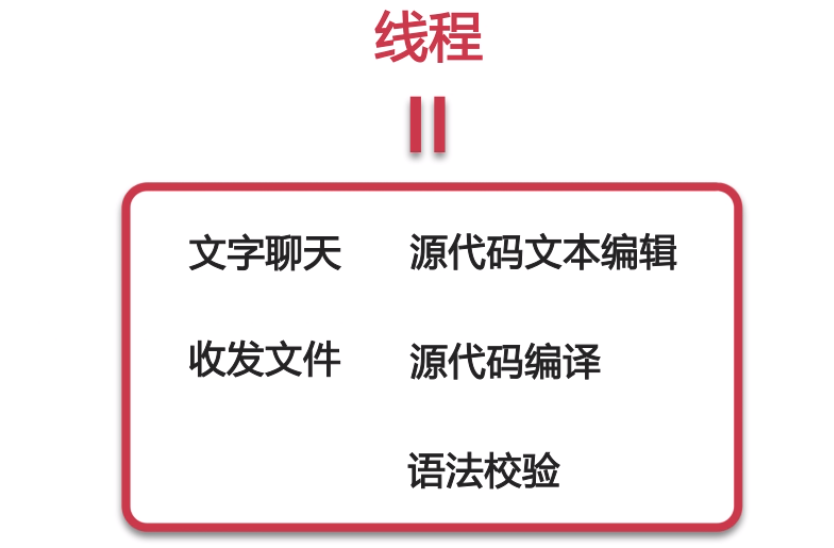
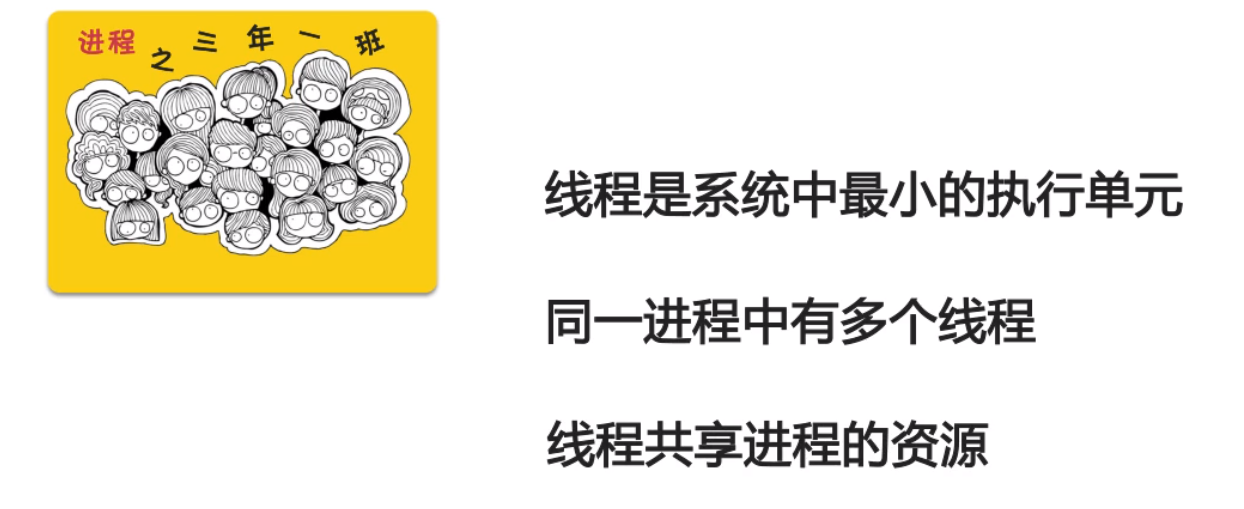
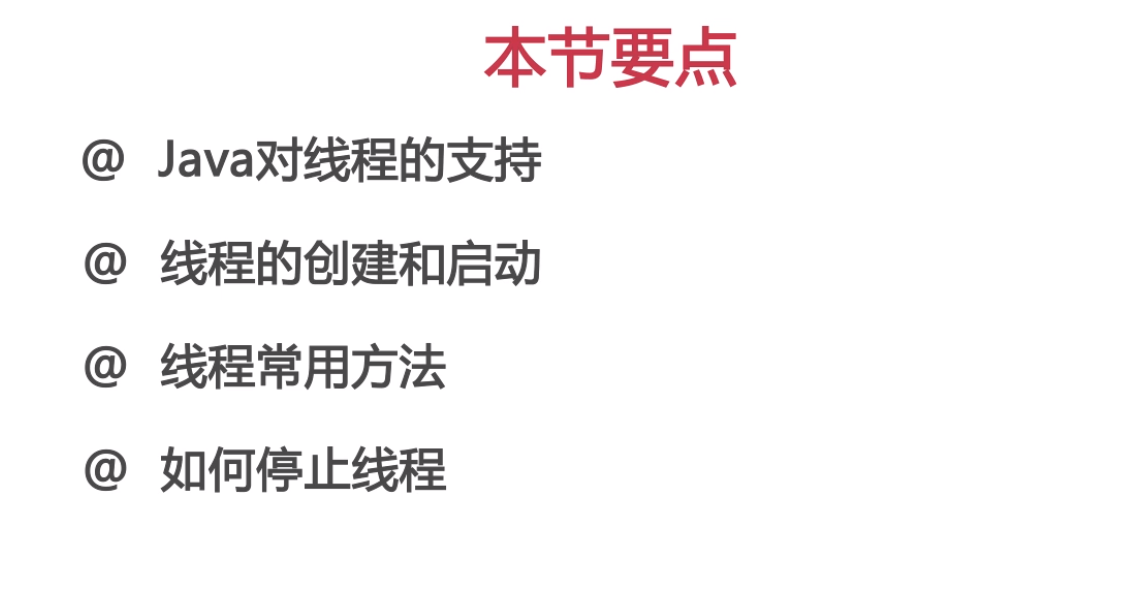
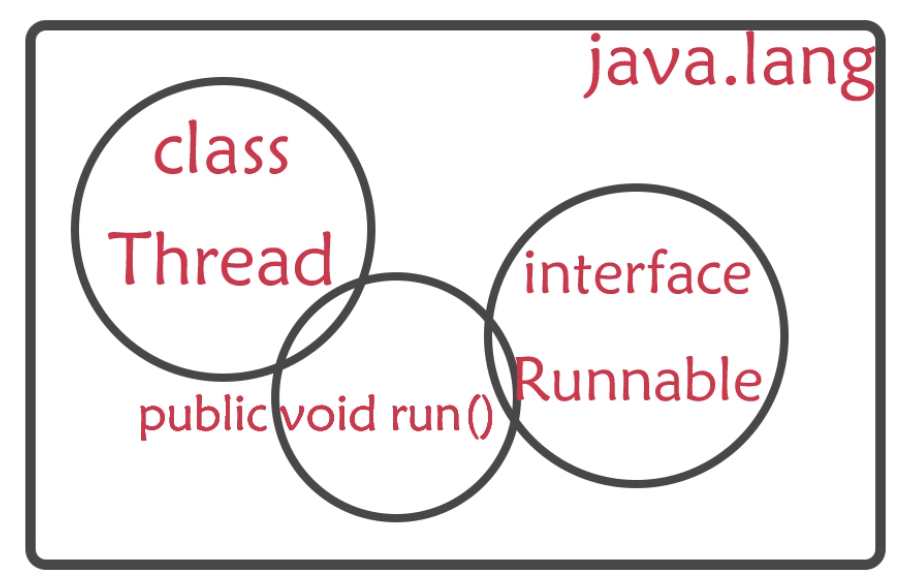
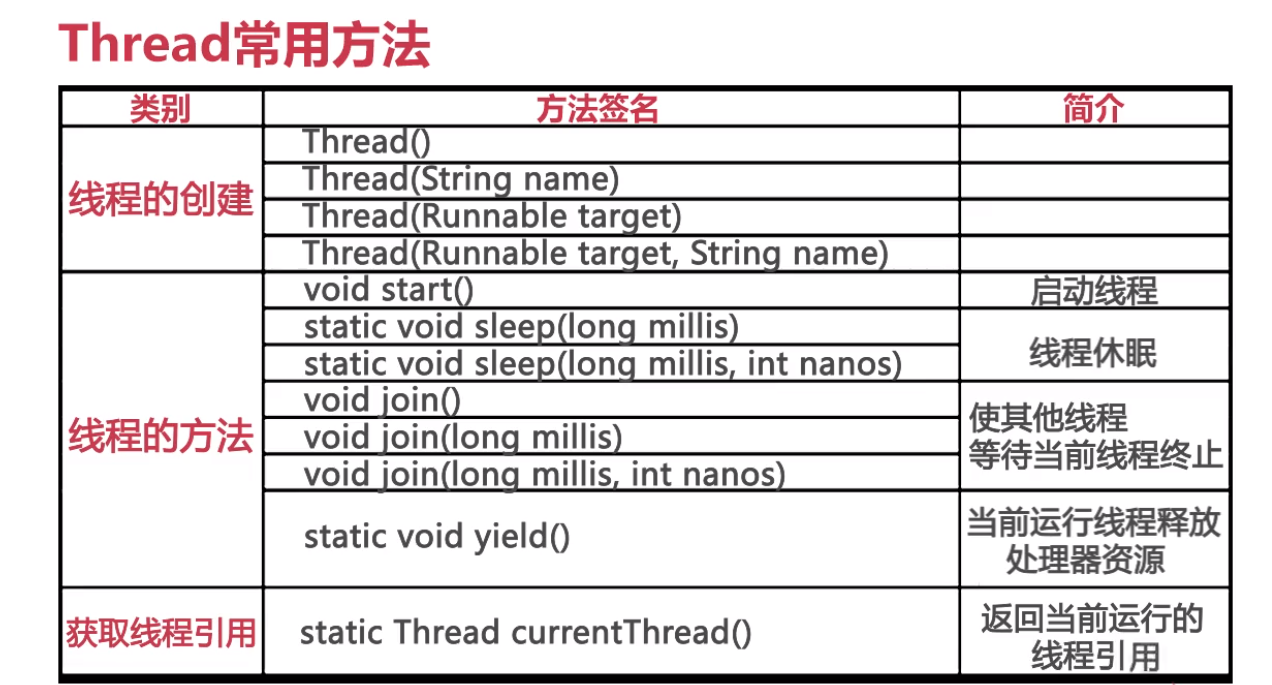
1-1:Java多线程的基础概念介绍：

        **package** com.concurrent;  
  
*/\*\*  
 \* 线程的基本行为：  
 \*/***public class** Actor **extends** Thread {  
  
 @Override  
 **public void** run() {  
 System.***out***.println(getName()+**"是一个演员！"**);  
  
 **int** count = 0;  
 **boolean** keepRunning = **true**;  
 **while** (keepRunning) {  
 System.***out***.println(getName() + **"登台演出了"** + (++count));  
  
 **if**(count == 100){  
 keepRunning = **false**;  
 }  
*// 通过sleep休眠* **if**(count%10 == 0){  
 **try** {  
 System.***out***.println(**"--10场休息下-3秒-"**);  
 Thread.*sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
  
 }  
 System.***out***.println(getName()+**"演出结束！"**);  
 }  
  
 **public static void** main(String[] args) {  
  
 Thread actor = **new** Actor();  
 actor.setName(**"Mr.Thread"**);  
 actor.start();  
 *//2个演员登台演出* Thread actressThread = **new** Thread(**new** Actress(),**"Ms.Runnable"**);  
 actressThread.start();  
*// --------交替登台的情况-----* }  
}  
  
**class** Actress **implements** Runnable{  
  
 @Override  
 **public void** run() {  
 System.***out***.println(Thread.*currentThread*().getName()+**"是一个演员！"**);  
  
 **int** count = 0;  
 **boolean** keepRunning = **true**;  
 **while** (keepRunning) {  
 System.***out***.println(Thread.*currentThread*().getName() + **"登台演出了"** + (++count));  
  
 **if**(count == 100){  
 keepRunning = **false**;  
 }  
*// 通过sleep休眠* **if**(count%10 == 0){  
 **try** {  
 System.***out***.println(**"--10场休息下-3秒-"**);  
 Thread.*sleep*(1000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
  
 }  
 System.***out***.println(Thread.*currentThread*().getName()+**"演出结束！"**);  
 }  
}

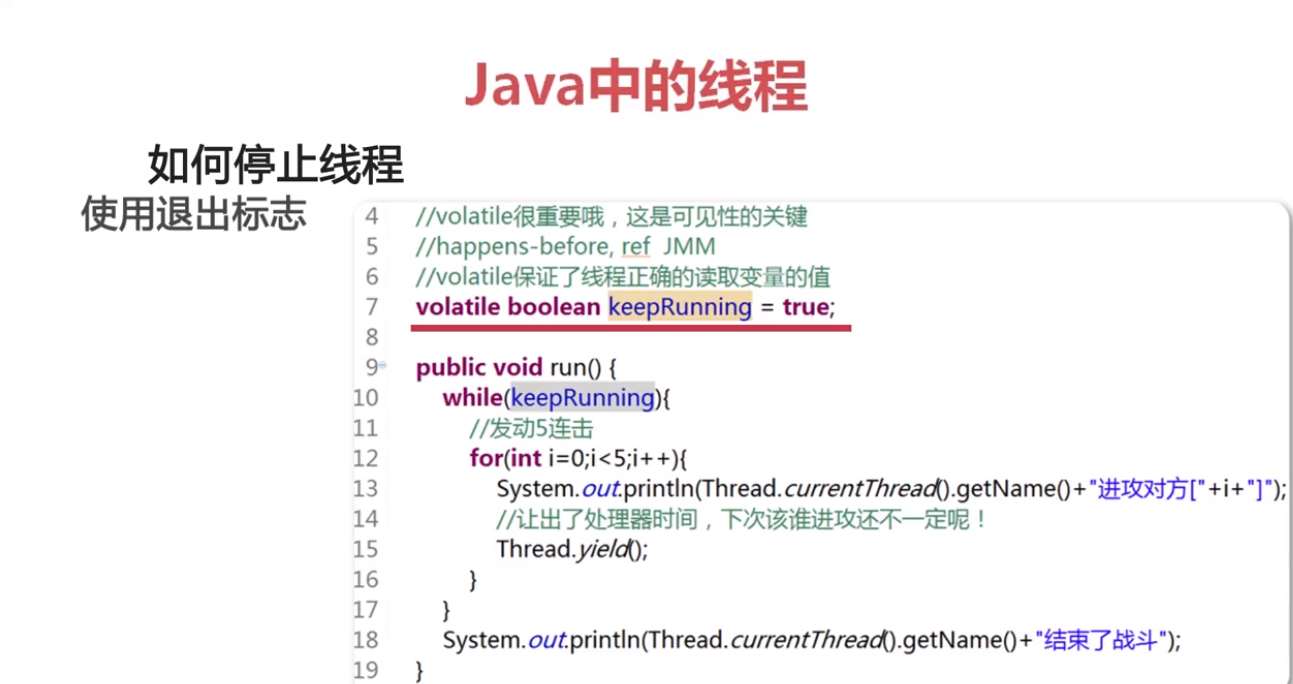
隋唐演义的案例：

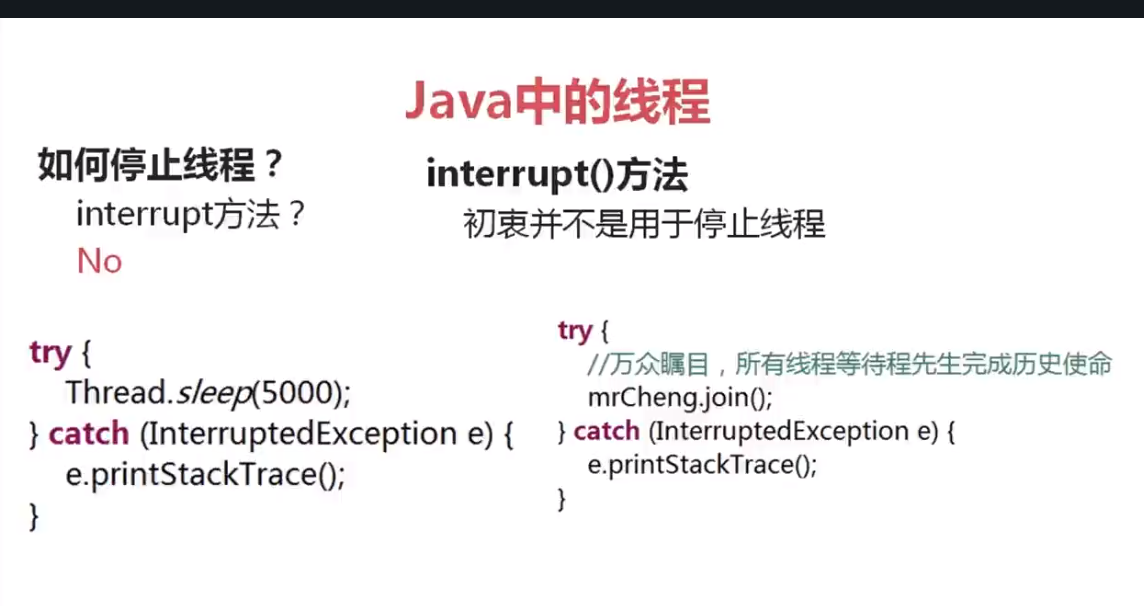
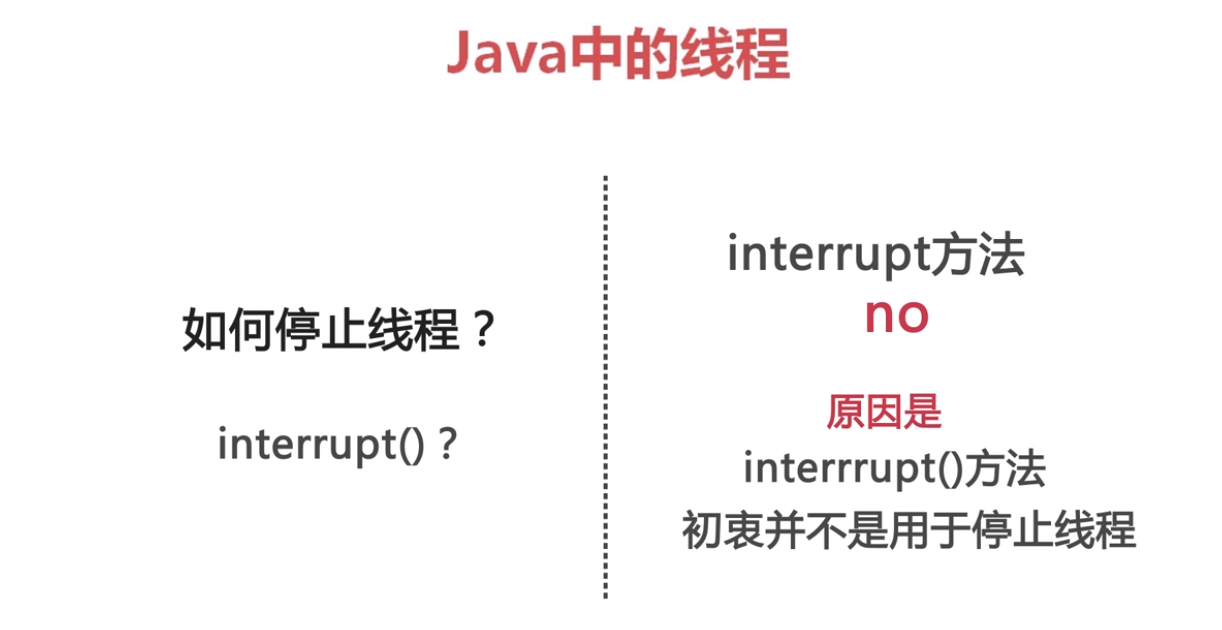
**package** com.concurrent.base;  
  
*/\*\*  
 \* 军队线程，模拟作战双发的行为  
 \*/***public class** ArmyRunnable **implements** Runnable {  
*// volatile 保证了线程可以正确的读取 其他线程写入的值  
// 其他线程写入 keep  
// 可见性： ref JMM ,happens-before原则* **volatile boolean keepRunning** = **true**;  
 @Override  
 **public void** run() {  
  
 **while** (**keepRunning**){  
  
 **for** (**int** i=0;i<5;i++) {  
 System.***out***.println(Thread.*currentThread*().getName() + **" -进攻对方-["** + i + **" ]"**);  
*// 让出了处理器时间，下次谁进攻不一定  
// 所以作战结果 有事交替- 有事连续* Thread.*yield*();  
 }  
 }  
 System.***out***.println(**"--结束战斗咯-----"**);  
 }  
}

------------------------------------------------------------------------------

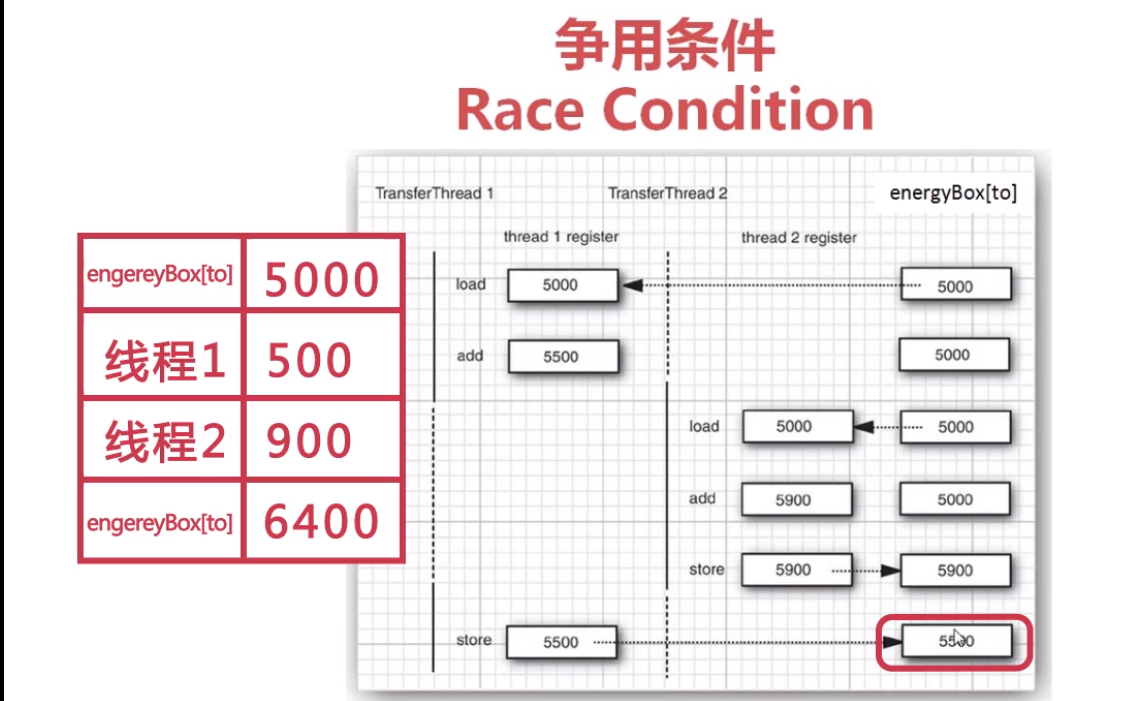
**package** com.concurrent.base;  
  
**public class** KeyPersonThread **extends** Thread {  
  
 @Override  
 **public void** run() {  
 System.***out***.println(Thread.*currentThread*().getName()+**" 开始了战斗！"**);  
*// 关键人物就可以10连击* **for** (**int** i=0;i<10;i++){  
 System.***out***.println(Thread.*currentThread*().getName()+ **"左突右杀，攻击随军.[ "**+i+**" ]"**);  
 }  
  
 System.***out***.println(Thread.*currentThread*().getName()+**" 结束了战斗！"**);  
 }  
}

**package** com.concurrent.base;  
  
*/\*\*\*  
 \* 隋唐演义大戏的舞台  
 \*/***public class** Stage **extends** Thread {  
  
 @Override  
 **public void** run() {  
  
 System.***out***.println(**"欢迎观看隋唐演义"**);  
*// 让关注们安静片刻，等待大戏上演* **try** {  
 Thread.*sleep*(5000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 System.***out***.println(**"--大幕徐徐拉开--"**);  
 **try** {  
 Thread.*sleep*(5000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 System.***out***.println(**"话说隋朝末年，随军农民起义杀的昏头暗地..."**);  
  
 ArmyRunnable armTaskOfSuiDynasty = **new** ArmyRunnable();  
 ArmyRunnable armyTaskOfRevolt = **new** ArmyRunnable();  
  
*//使用Runnable接口创建线程* Thread armyOfSuiDynasty = **new** Thread(armTaskOfSuiDynasty,**"隋军"**);  
 Thread armyOfRevolt = **new** Thread(armyTaskOfRevolt,**"农民起义军"**);  
*//启动线程，让军队开始作战* armyOfSuiDynasty.start();  
 armyOfRevolt.start();  
*// 让舞台线程暂时休眠，大家专心观看军队作战* **try** {  
 Thread.*sleep*(100);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
*//  
// armTaskOfSuiDynasty.keepRunning=false;  
// armyTaskOfRevolt.keepRunning=false;  
  
//  
  
// try {  
// armyOfRevolt.join();  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }* System.***out***.println(**"正当双方激战正酣，半路杀出个程咬金---"**);  
  
 Thread mrCheng = **new** KeyPersonThread();  
 mrCheng.setName(**"程咬金"**);  
 System.***out***.println(**"程咬金的理想就是结束战争，使得安居乐业"**);  
  
*// 停止军队作战* armTaskOfSuiDynasty.**keepRunning**= **false**;  
 armyTaskOfRevolt.**keepRunning**=**false**;  
  
 **try** {  
 Thread.*sleep*(2000);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 */\*\*  
 \* 历史大戏留给关键人物  
 \*/* mrCheng.start();  
  
*// 万众瞩目，所有线程等待程咬金完成历史使命* **try** {  
 mrCheng.join();  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 System.***out***.println(**"战争结束，人们安居乐业---程咬金完成了自己的梦想---------"**);  
 System.***out***.println(**"谢谢观看演绎！再见！"**);  
 }  
  
 **public static void** main(String[] args) {  
 **new** Stage().start();  
*//* }  
}





Java中的线程交互：



互斥：关键数据同一时间只能被一个线程访问：

Synchronized

同步：wait(),notify,notifyall(),方法：

