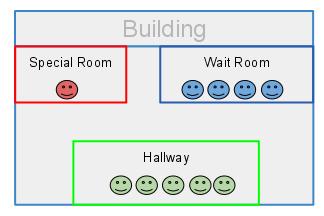
转载自：<http://ifeve.com/monitors-java-synchronization-mechanism/>

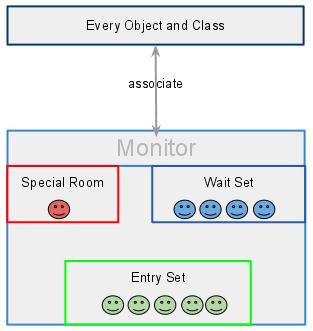
大学有一门课程叫操作系统，学习过的同学应该都记得，监视器是操作系统实现同步的重要基础概念，同样它也用在JAVA的线程同步中，这篇文章用一种类推的思想解释监视器”monitor”。

**1.什么是监视器**

监视器可以看做是经过特殊布置的建筑，这个建筑有一个特殊的房间，该房间通常包含一些数据和代码，但是一次只能一个消费者(thread)使用此房间，



当一个消费者(线程)使用了这个房间，首先他必须到一个大厅(Entry Set)等待，调度程序将基于某些标准(e.g. FIFO)将从大厅中选择一个消费者(线程)，进入特殊房间，如果这个线程因为某些原因被“挂起”，它将被调度程序安排到“等待房间”，并且一段时间之后会被重新分配到特殊房间，按照上面的线路，这个建筑物包含三个房间，分别是“特殊房间”、“大厅”以及“等待房间”。



简单来说，监视器用来监视线程进入这个特别房间，他确保同一时间只能有一个线程可以访问特殊房间中的数据和代码。

**2.JAVA中监视器的实现**

在JAVA虚拟机中，每个对象(Object和class)通过某种逻辑关联监视器，为了实现监视器的互斥功能，每个对象(Object和class)都关联着一个锁(有时也叫“互斥量”)，这个锁在操作系统书籍中称为“信号量”，互斥(“mutex “)是一个二进制的信号量。

如果一个线程拥有了某些数据的锁，其他的线程则无法获得锁，直到这个线程释放了这个锁。在多线程中，如果任何时候都是我们自己来写这个信号量，显然不是很方便，幸运的是，JVM为我们自动实现了这些。

为了使数据不被多个线程访问，java 提供了同步块以及同步方法两种实现，一旦一段代码被嵌入到一个synchronized关键字中，意味着放入了监视区域，JVM在后台会自动为这段代码实现锁的功能。

**3.JAVA的同步代码中，哪一部分是监视器？**

我们知道JAVA每个对象(Object/class) 都关联一个监视器，更好的说法应该是每个对象(Object/class)都有一个监视器，对象可以有它自己的临界区，并且能够监视线程序列为了使线程协作，JAVA为提供了wait()和notifyAll以及notify()实现挂起线程，并且唤醒另外一个等待的线程，此外这些方法有三种不同版本:

wait(long timeout, int nanos)

wait(long timeout) notified by other threads or notified by timeout.

notify(all)

这些方法只能在一个同步块或同步方法中被调用，原因是，如果一个方法不需要相互排斥，不需要监测或线程之间协作，每一个线程可以自由访问此方法，那就不需要协作。