Synchronized on Method:

Giới hạn chỉ 1 thread được truy câjp vào method đó trong 1 thời điểm.

Nếu đang có 1 thread sd thì các thread khác phải đợi cho tới khi method đó được chạy xong bởi thread X đang sử dụng.

public class LockHolder {  
 // all threads that see this method can access it at the same time  
 public static void callBack(String name){  
 for(int i = 1; i < 10; i++){  
 System.*out*.println("thread: " + name + " - " + i);  
 }  
 }  
  
 // SYNCHRONIZED: prevent multi thread access this method at the same time, when one thread is holding the lock(using method)  
 // then NO others thread can access it. They have to wait until the lock get release and so on.  
 public static synchronized void synchronizedCallBack(String name){  
 for(int i = 1; i < 10; i++){  
 System.*out*.println("thread: " + name + " - " + i);  
 }  
 }  
}

. Synchronized block in method:

Thường dúng để chỉ định **1 tài nguyên nào đó chỉ được 1 thread** truy cập 1 lúc, phạm vi nhỏ hơn **Synchronized on Method**. Có hiệu quả hơn so với **Synchronized on Method:**

public class LockHolder {  
 final List<Integer> integers = Arrays.*asList*(1,2,3,4); // only 1 thread use this LIST at onece time

// like Synchronized block in method because synchronized hold THIS  
 public void showSynchronized(String name){  
 synchronized (this){  
 for(int i = 1; i < 6 ; i++){  
 System.*out*.println(name + " " + i);  
 }  
 }  
 }

// main point  
 public void printSynchronized(String name){  
// final List<Integer> integers = Arrays.asList(1,2,3,4); // local: mean nothing, it must be global variable then it takes effects  
  
 synchronized (integers){  
 for(int i : integers){  
 System.*out*.println(name + " " + i);  
 }  
 }  
 }