A 𝐝𝐞𝐚𝐝𝐥𝐨𝐜𝐤 occurs when two or more threads wait "forever" for a lock or resource held by another of the threads.  
  
There are three ways on how we can avoid deadlocks in our code:  
1) avoid the need for acquiring multiple locks for a thread  
2) if you really need multiple locks in the same thread, we should make sure that each thread acquires the locks in the same order, to avoid any cyclic dependency in lock acquisition.  
3) use timed lock attempts, like the 𝐭𝐫𝐲𝐋𝐨𝐜𝐤() method in the 𝐣𝐚𝐯𝐚.𝐮𝐭𝐢𝐥.𝐜𝐨𝐧𝐜𝐮𝐫𝐫𝐞𝐧𝐭.𝐥𝐨𝐜𝐤𝐬.𝐋𝐨𝐜𝐤 interface, to make sure that a thread does not block infinitely if it is unable to acquire a lock.  
  
The best example of deadlock in real life occurs when you have to negotiate your salary with a new company, they asks for your salary expectations but you are interested about position budget.

* When 2 or more threads are waiting for each other to release the resource they need (lock) and get stuck for infinite time, the situation is called deadlock.
* Can be avaoided Avoid nested locks, Avoid Unnecessary Locks , use Joins