**Deadlock**

Deadlock case happens in concurrent or multi-threaded environment. It is **kind of a situation in which two or more competing threads or tasks wait for the other task to finish** and **they never finish**.

 It is a vary famous problem in multi processing systems, parallel computing and distributed systems. This case basically happens with the shared resource. This resource locks in this case. Deadlock has very vast explanation.

**Conditions of Deadlock**

 A condition is called Deadlock if the following condition matches:

1. **Mutual Exclusion**: Only process can use the resource at the given time.
2. **Hold and Wait**
3. No Preemption
4. Circular Wait

These four conditions are know as the **Coffman Conditions. Edward G. Coffman, Jr.**described this in his article published in 1971.

### Ways to Make an Object Thread-Safe

The following are the three approaches to make an object thread-safe:

1. **Synchronize the critical section**.
2. Make it **immutable**
3. **Use a thread-safe wrapper**

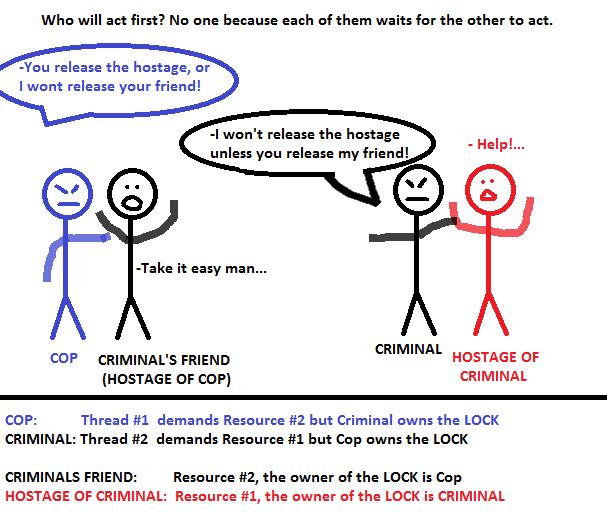
-

-

-

Let me explain a real world (not actually real) example for a deadlock situation from the crime movies. Imagine a criminal holds an hostage and against that, a cop also holds an hostage who is a friend of the criminal. In this case, criminal is not going to let the hostage go if cop won't let his friend to let go. Also the cop is not going to let the friend of criminal let go, unless the criminal releases the hostage. This is an endless untrustworthy situation, because both sides are insisting the first step from each other.

Criminal & Cop Scene

[](https://i.stack.imgur.com/3XVzK.png)

So simply, when two threads needs two different resources and each of them has the lock of the resource that the other need, it is a deadlock.

Another High Level Explanation of Deadlock : Broken Hearts

You are dating with a girl and one day after an argument, both sides are heart-broken to each other and waiting for an **I-am-sorry-and-I-missed-you** call. In this situation, both sides want to communicate each other if and only if one of them receives an **I-am-sorry** call from the other. Because that neither of each is going to start communication and waiting in a passive state, both will wait for the other to start communication which ends up in a deadlock situation.