**Why does setting a variable to null, help with garbage collection?**

I briefly touched on this point during the lecture, but it's common practice among experienced developers to set their peer connections and data channels to **null** after closing them.

Setting a variable to **null** offers several benefits.

First and foremost, it removes references. When you assign **null** to a variable (like **pc**), you are explicitly eliminating the reference to the object it was previously pointing to. This means that if there are no other references to that object in your code, it becomes unreachable (and yes, therefore eligible for garbage collection).

Take this code for example:

let pc = new RTCPeerConnection();

// ... use pc for some operations ...

pc.close(); // Closes the peer connection

pc = null; // Removes the reference to the RTCPeerConnection object

In this above example, even after calling **pc.close()**, the **pc** variable still holds a reference to the **RTCPeerConnection** object. By setting **pc** to **null**, you indicate that you no longer require that object, allowing the garbage collector to reclaim its memory if there are no other references to it.

Next Reading Material:

**Closing a Data Channel**

When a peer closes a data channel using dataChannel.close(), it triggers the onclose event on both ends of the data channel.

This means that if one peer closes the data channel, the other peer will receive a close event on their corresponding data channel. This is part of the WebRTC specification, which ensures that both peers are aware of the state of the data channel.

To me, this makes sense!

Remember, data channels are meant to be independent and can be opened and closed as needed. This allows applications to manage multiple data streams without needing to renegotiate the entire peer connection again. For example, in your app, you might want to open a data channel for text chat and another for file transfer. Closing one should not affect the other, hence the design choice to notify both peers about the closure of one specific data channel.

*(as you'll see soon, this is not the case with calling the close() method on the peer connection itself)*

#### Understanding Garbage Collection



In JavaScript, garbage collection is the process by which the JavaScript engine automatically frees up memory that is no longer in use. The engine keeps track of objects in memory and determines whether they are still reachable or if they can be safely removed.

By explicitly nullifying the variable, you help the garbage collector identify that the object is no longer needed. This can be particularly useful in long-running applications where many objects are created and destroyed over time. It helps prevent memory leaks, which can occur if objects remain in memory because they are still referenced somewhere in the code, even if they are no longer needed.

**Got it?**

Great. Let's move on.

*(p.s. don't overlook the simple reason also. Setting a variable to* ***null*** *also serves as a clear signal to other developers (or your future self) that the variable is intentionally no longer in use. This can help prevent accidental usage of the variable after it has been closed, which could lead to errors)*