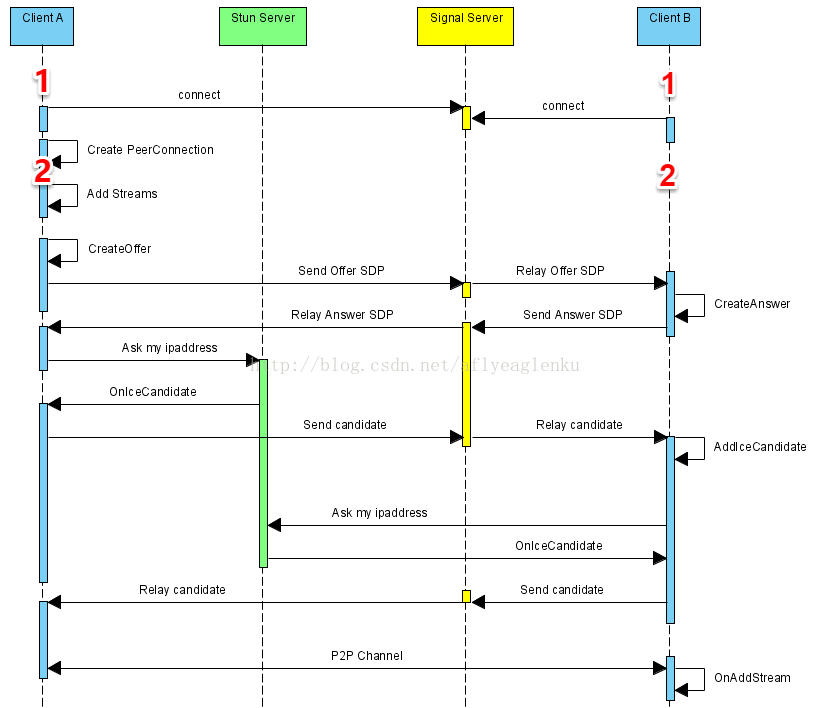
WebRTC Peerconnection通信过程中的四种角色：

* Signaling Server
* ICE/TURN/STUN Server
* Remote Peer
* Local Peer

通信过程中的基本概念：

* Room：Signaling Server使用Room的概念对同一组通信的peers进行配对管理，一个room中包含有1个或者多个peers。当没有peers存在时room销毁；当第一个peer连接到signaling server时执行create room动作，此时因为没有其他peers，建立room的这个peer不与其他节点建立P2P通信；其他peers随后加入room，加入room后会主动与room已经存在的peer建立连接；
* Offer：主动与其他peer建立P2P链接的peer把自己的SDP信息整理好，通过signaling server转发给room里面的其他peer，这个SDP信息包就是Offer；
* Answer：被动连接的Peer在收到signaling server转发的其他peer的offer信息以后，也把自己的SDP信息整理好，同样通过signaling server转发给主动连接它的peer，他自己的SDP信息包就是Answer；
* IceCandidate：Peer与ICE/TURN/STUN Server直接建立连接，获取自己的NAT类型以及外网IP和端口，这些ICE/TURN/STUN Server返回的消息就是IceCandidate或者直接简称Candidate；

PeerConnection连接建立流程图：



对于上图中描述的PeerConnection建立的完整流程进行以下说明（上图是以ClientA主动向ClientB发起连接为例）：

* 首先ClientA和ClientB均通过双向通信方式如WebSocket连接到Signaling Server上；
* ClientA在本地首先通GetMedia访问本地的media接口和数据，并创建PeerConnection对象，调用其AddStream方法把本地的Media添加到PeerConnection对象中。****对于ClientA而言，既可以在与Signaling Server建立连接之前就创建并初始化PeerConnection如阶段1，也可以在建立Signaling Server连接之后创建并初始化PeerConnection如阶段2；ClientB既可以在上图的1阶段也可以在2阶段做同样的事情，访问自己的本地接口并创建自己的PeerConnection对象****。
* 通信由ClientA发起，所以ClientA调用PeerConnection的CreateOffer接口创建自己的SDP offer，然后把这个SDP Offer信息通过Signaling Server通道中转发给ClientB；
* ClientB收到Signaling Server中转过来的ClientA的SDP信息也就是offer后，调用CreateAnswer创建自己的SDP信息也就是answer，然后把这个answer同样通过Signaling server转发给ClientA；
* ClientA收到转发的answer消息以后，两个peers就做好了建立连接并获取对方media streaming的准备；
* ClientA通过自己PeerConnection创建时传递的参数等待来自于ICE server的通信，获取自己的candidate，当candidate available的时候会自动回掉PeerConnection的OnIceCandidate；
* ClientA通过Signling Server发送自己的Candidate给ClientB，ClientB依据同样的逻辑把自己的Candidate通过Signaling Server中转发给ClientA；
* 至此ClientA和ClientB均已经接收到对方的Candidate，通过PeerConnection建立连接。至此P2P通道建立。