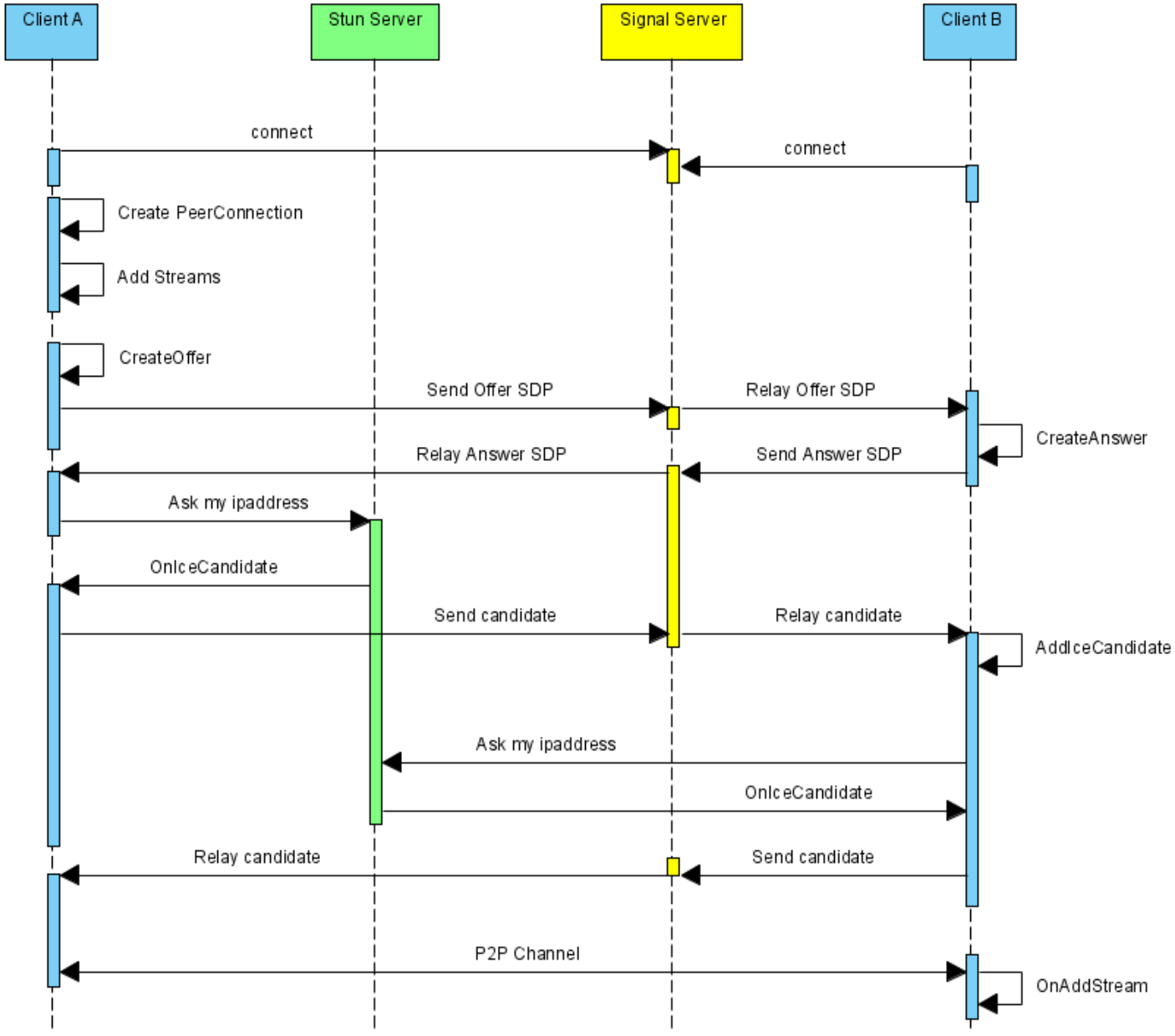
WebRTC手记之初探 - 孤竹君 - 博客园

WebRTC是HTML5支持的重要特性之一，有了它，不再需要借助音视频相关的客户端，直接通过浏览器的Web页面就可以实现音视频对聊功能。而且WebRTC项目是开源的，我们可以借助WebRTC源码快速构建自己的音视频对聊功能。无论是使用前端JS的WebRTC API接口，还是在WebRTC源码上构建自己的对聊框架，都需要遵循以下执行流程：



上述序列中，WebRTC并不提供Stun服务器和Signal服务器，服务器端需要自己实现。Stun服务器可以用google提供的实现stun协议的测试服务器（stun:stun.l.google.com:19302），Signal服务器则完全需要自己实现了，它需要在ClientA和ClientB之间传送彼此的SDP信息和candidate信息，ClientA和ClientB通过这些信息建立P2P连接来传送音视频数据。由于网络环境的复杂性，并不是所有的客户端之间都能够建立P2P连接，这种情况下就需要有个relay服务器做音视频数据的中转，本文本着源码剖析的态度，这种情况就不考虑了。这里说明一下，stun/turn、relay服务器的实现在WebRTC源码中都有示例，真是个名副其实的大宝库。

上述序列中，标注的场景是ClientA向ClientB发起对聊请求，调用描述如下：

- ClientA首先创建PeerConnection对象，然后打开本地音视频设备，将音视频数据封装成MediaStream添加到PeerConnection中。
- ClientA调用PeerConnection的CreateOffer方法创建一个用于offer的SDP对象，SDP对象中保存当前音视频的相关参数。ClientA通过PeerConnection的SetLocalDescription方法将该SDP对象保存起来，并通过Signal服务器发送给ClientB。
- ClientB接收到ClientA发送过的offer SDP对象，通过PeerConnection的SetRemoteDescription方法将其保存起来，并调用PeerConnection的CreateAnswer方法创建一个应答的SDP对象，通过PeerConnection的SetLocalDescription的方法保存该应答SDP对象并将它通过Signal服务器发送给ClientA。
- ClientA接收到ClientB发送过来的应答SDP对象，将其通过PeerConnection的SetRemoteDescription方法保存起来。
- 在SDP信息的offer/answer流程中，ClientA和ClientB已经根据SDP信息创建好相应的音频Channel和视频Channel并开启Candidate数据的收集，Candidate数据可以简单地理解成Client端的IP地址信息（本地IP地址、公网IP地址、Relay服务端分配的地址）。
- 当ClientA收集到Candidate信息后，PeerConnection会通过OnIceCandidate接口给ClientA发送通知，ClientA将收到的Candidate信息通过Signal服务器发送给ClientB，ClientB通过PeerConnection的AddIceCandidate方法保存起来。同样的操作ClientB对ClientA再来一次。
- 这样ClientA和ClientB就已经建立了音视频传输的P2P通道，ClientB接收到ClientA传送过来的音视频流，会通过PeerConnection的OnAddStream回调接口返回一个标识ClientA端音视频流的MediaStream对象，在ClientB端渲染出来即可。同样操作也适应ClientB到ClientA的音视频流的传输。

这里的流程仅仅是从使用层面上描述了一下，具体内部都做了什么、怎么做的，以后的文章中会慢慢细扒，万事开头难，自我鼓励一下。