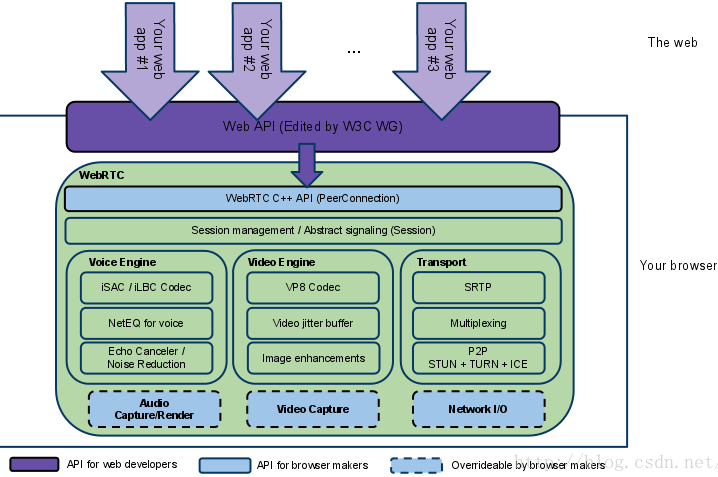
WebRTC实现了实时、无插件的音频、视频和数据通信，WebRTC项目的宗旨：API开源、免费、标准、可内建于浏览器且比其他现存的技术更加高效。

一、WebRTC的架构图



WebRTC对编解码器和协议做了大量的工作，使实时通信成为可能，甚至在一些不可靠的网络中：

1.包补偿

2.回声消除

3.自适应带宽

4.视频抖动缓冲

5.自动增益控制

6.噪声抑制

7.图像清除

1. 快速开始

开发WebRTC应用需要做好下列准备：

1.获取音视频流或者其他数据

2.得到网络信息，如IP地址和端口，通过网络和其它WebRTC客户端交换数据，解决NATs/防火墙穿透问题。

3.协调信令通信来报告错误、启动或关闭会话。

4.交换媒体和客户端信息，比如分辨率和编解码参数。

5.传输音视频流或者其他数据。

为了实现数据流通信，WebRTC实现了下列API：

MediaStream(别名getUserMedia)：用户提供访问硬件设备媒体（摄像头、视频、音频、地理位置等）的接口，从设备获取数据流

RTCPeerConnection：在浏览器（别名peer）之间互通数据流，音视频通话，包括设备加密和带宽管理， 代表一个由本地计算机到远端的WebRTC连接。该接口提供了创建，保持，监控，关闭连接的方法的实现。

RTCDataChannel：接口代表在两者之间建立了一个双向数据通道的连接，RTCDataChannel 的语法与 WebSocket 类似

1、getUserMedia

<https://simpl.info/getusermedia/>

应用举例：

（1）、Webcam Toy是一个photobooth

<https://webcamtoy.com/app/>

（2）、ASCII Camera使用Canvas API来生成ASCII码的图片

<https://idevelop.ro/ascii-camera/>

2、RTCPeerConnection

<https://simpl.info/rtcpeerconnection/>

本地demo位置：

webRTC分享\samples-gh-pages\src\content\capture\video-pc

接口参数参考（都有明确的描述）：

<https://developer.mozilla.org/zh-CN/docs/Web/API/RTCPeerConnection>

3、RTCDataChannel

接口参数参考（都有明确的描述）：

<https://developer.mozilla.org/zh-CN/docs/Web/API/RTCDataChannel>

4、demos练习WebRTC的JavaScript API

[*https://webrtc.github.io/samples/*](https://webrtc.github.io/samples/)

**补充知识：**

信令：会话控制，网络和媒体信息

WebRTC使用RTCPeerConnection在浏览器之间互通数据流，但是需要一种机制去协调通信或者发送控制消息，这个过程被称为**信令**。WebRTC**没有指定信令方法和协议**，信令不是RTCPeerConnection API的一部分。

       因此，WebRTC应用的**开发者可用选择其擅长的消息协议**，比如SIP或XMPP，或者其他合适的双工通信协议。

例如：apprtc.appspot.com使用XHR和Channel API作为信令机制；codelab是我们通过Socket.io构建，运行在Node server上的应用。

      信令通常用于交互三类信息：

1.会话控制消息；初始化或者关闭通信，报告错误。

2.网络信息：对于外部而言，我的IP地址和端口是什么？

3.媒体信息：什么编码和分辨率浏览器可以处理，我的浏览器要和谁通信。

       在p2p的流传输之前，必须通过信令成功的交换信息。

三、无服务器的RTCPeerConnection

<https://webrtc.github.io/samples/src/content/peerconnection/pc1/>

代码来自https://webrtc.github.io/samples/src/content/peerconnection/pc1，包含基于网页的本地和远程RTCPeerConnection。这个例子中caller和callee在同一个网页中，能更加清晰的展示RTCPeerConnection API的工作流程，因为RTCPeerConnection对象之间可以直接交换数据和消息，**不需要通过中继信道机制**。

四、有服务器的RTCPeerConnection

实际应用中，WebRTC需要服务器，无论多简单，下面四步是必须的：  
1.用户通过交换名字之类的信息发现对方。  
2.WebRTC客户端应用交换网络信息。  
3.客户端交换媒体信息包括视频格式和分辨率。  
4.WebRTC客户端穿透NAT网关和服务器。

换句话说，WebRTC需要四种类型的服务端功能。

1.用户发现和通信

2.信令

3.NAT/防火墙穿透

4.中继服务器，防止端到端的通信失败