

OSC原创会

年终盛典 2016

WebRTC与视频直播如何相互成就？

- 1、WebRTC介绍
- 2、视频直播介绍
- 3、如何相互成就

- 视频相关应用场景

视频通话

视频直播

在线教育

智能硬件

视频监控

企业通讯

等等等等



- 技术实现上都有哪些难度？

跨平台实现(android,ios,windows,mac)

视频编解码(VP8,VP9,H264,H265)

语音编解码(opus,silk,ilbc)

信号处理(AEC,AGC,NS,NetEQ)

网络传输与抖动缓冲

丢包重传(FEC,NACK)

安全性等等



- WebRTC是什么?

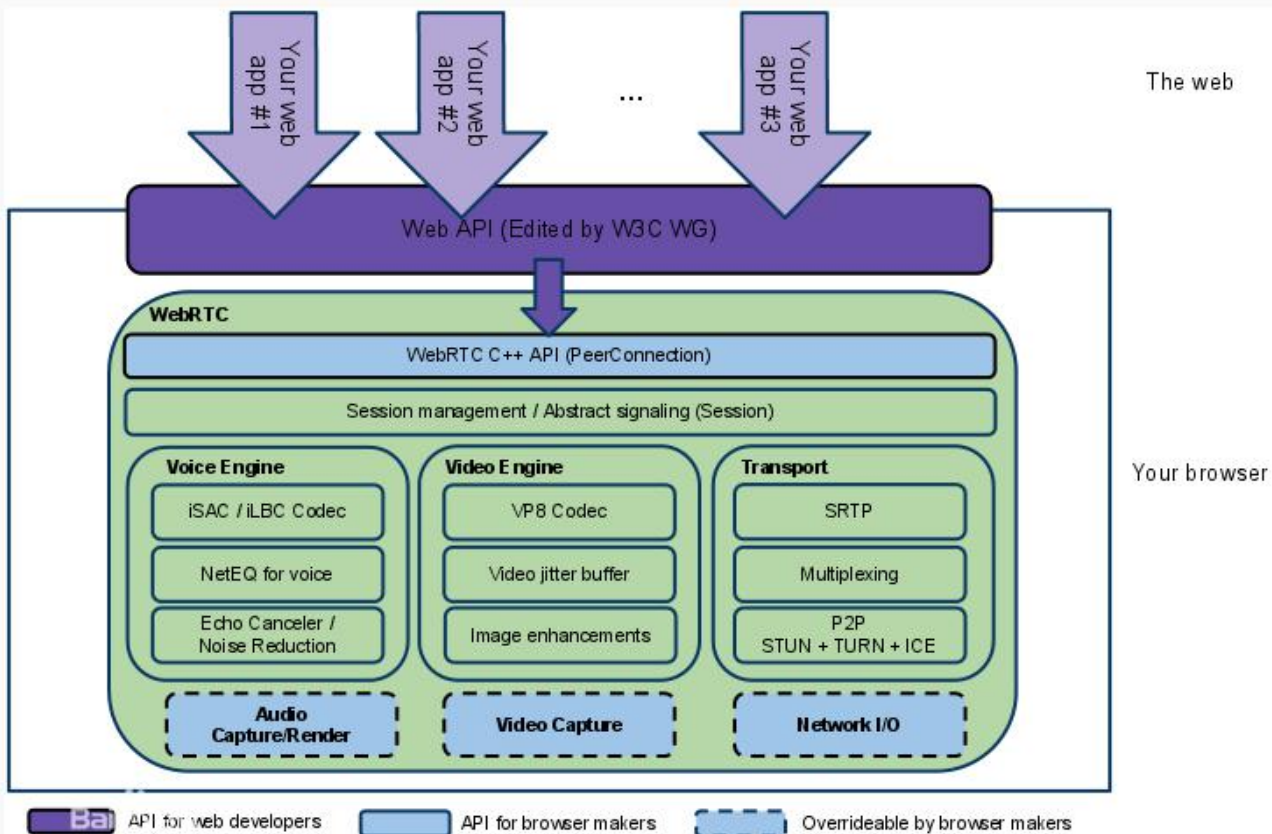
WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs.

The WebRTC components have been optimized to best serve this purpose.



WebRTC介绍

OSC源创会
年终盛典 2016











WebRTC架构图

该图片由zengxijin2012贡献

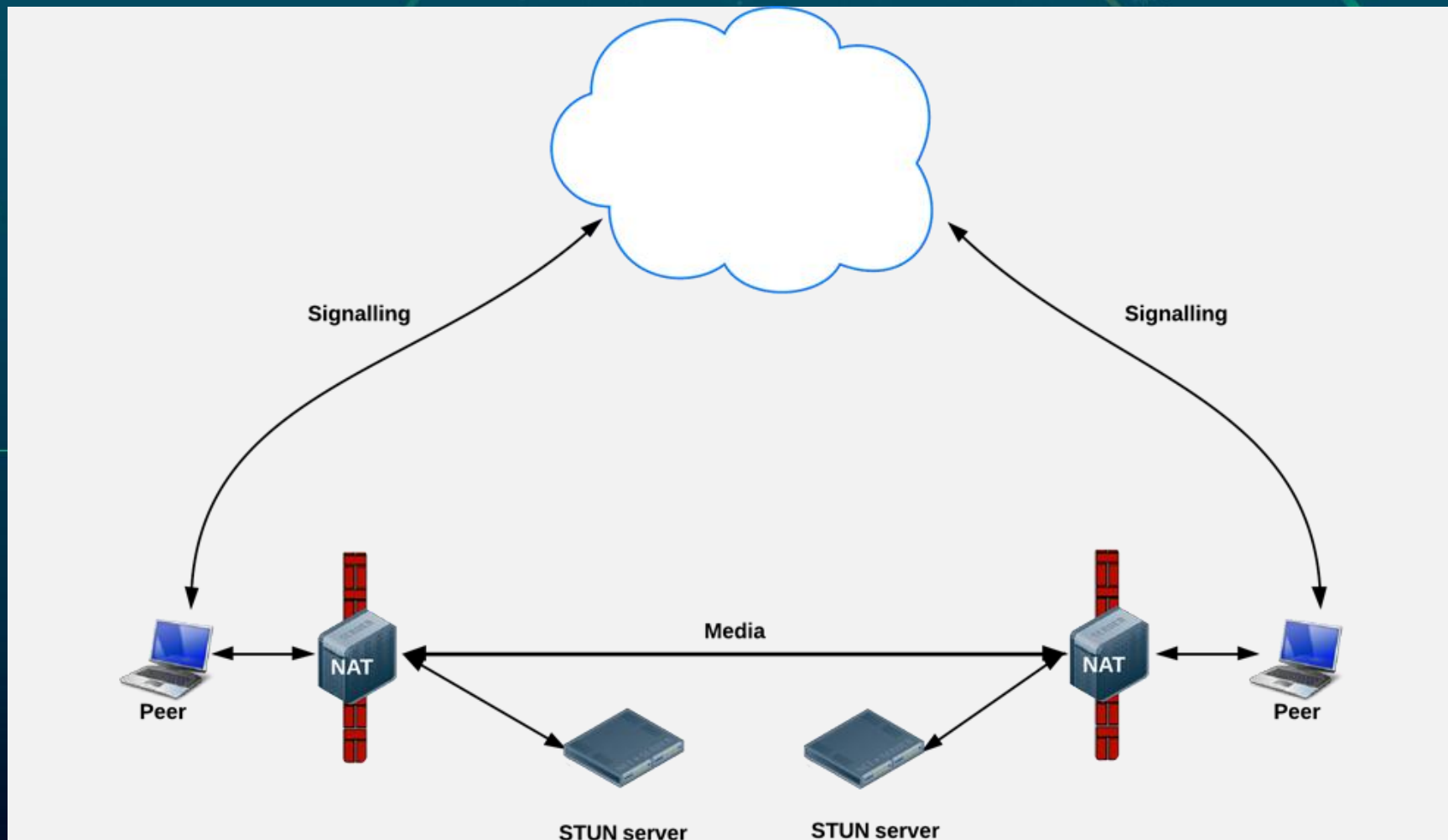
WebRTC介绍

OSC 原创会
年终盛典 2016

								
	Canary	Chrome	Opera	Nightly	Firefox	Bowser	Edge	Safari
PeerConnection API	Green	Green	Green	Green	Green	Green	Yellow	Red
getUserMedia	Green	Green	Green	Green	Green	Green	Green	Red
dataChannels	Green	Green	Green	Green	Green	Green	Red	Red
TURN support	Green	Green	Green	Green	Green	Green	Green	Red
Echo cancellation	Green	Green	Green	Green	Green	Green	Green	Red
MediaStream API	Green	Green	Green	Green	Green	Green	Green	Red
mediaConstraints	Yellow	Yellow	Yellow	Green	Green	Yellow	Yellow	Red
Multiple Streams	Yellow	Yellow	Yellow	Green	Green	Green	Green	Red
Simulcast	Yellow	Yellow	Yellow	Green	Yellow	Red	Yellow	Red
Screen Sharing	Yellow	Yellow	Yellow	Yellow	Yellow	Red	Red	Red
Stream re-broadcasting	Green	Yellow	Yellow	Green	Green	Red	Red	Red
getStats API	Yellow	Yellow	Yellow	Green	Green	Red	Green	Red
ORTC API	Red	Red	Red	Red	Red	Red	Green	Red
H.264 video	Green	Green	Green	Green	Green	Green	Green	Red
VP8 video	Green	Green	Green	Green	Green	Green	Red	Red
Solid interoperability	Green	Green	Green	Green	Green	Green	Yellow	Red
srcObject in media element	Green	Yellow	Yellow	Green	Green	Red	Green	Red
Promise based getUserMedia	Green	Green	Green	Green	Green	Green	Green	Red
Promise based PeerConnection API	Green	Green	Green	Green	Green	Green	Yellow	Red
WebAudio Integration	Green	Yellow	Yellow	Green	Green	Red	Yellow	Red
MediaRecorder Integration	Green	Green	Green	Green	Green	Red	Red	Red
Canvas Integration	Green	Green	Green	Green	Green	Green	Green	Red
Test support	Green	Green	Green	Green	Green	Red	Green	Red

WebRTC介绍

OSC 原创会
年终盛典 2016



```
10
11 var errorElement = document.querySelector('#errorMsg');
12 var video = document.querySelector('video');
13
14 // Put variables in global scope to make them available to the browser console.
15 var constraints = window.constraints = {
16   audio: false,
17   video: true
18 };
19
20 function handleSuccess(stream) {
21   var videoTracks = stream.getVideoTracks();
22   console.log('Got stream with constraints:', constraints);
23   console.log('Using video device: ' + videoTracks[0].label);
24   stream.oninactive = function() {
25     console.log('Stream inactive');
26   };
27   window.stream = stream; // make variable available to browser console
28   video.srcObject = stream;
29 }
30
31 function handleError(error) {
32   if (error.name === 'ConstraintNotSatisfiedError') {
33     errorMsg('The resolution ' + constraints.video.width.exact + 'x' +
34       constraints.video.width.exact + ' px is not supported by your device.');
```

```
35   } else if (error.name === 'PermissionDeniedError') {
36     errorMsg('Permissions have not been granted to use your camera and ' +
37       'microphone, you need to allow the page access to your devices in ' +
38       'order for the demo to work.');
```

```
39   }
40   errorMsg('getUserMedia error: ' + error.name, error);
41 }
42
43 function errorMsg(msg, error) {
44   errorElement.innerHTML += '<p>' + msg + '</p>';
45   if (typeof error !== 'undefined') {
46     console.error(error);
47   }
48 }
49
50 navigator.mediaDevices.getUserMedia(constraints).
51   then(handleSuccess).catch(handleError);
```

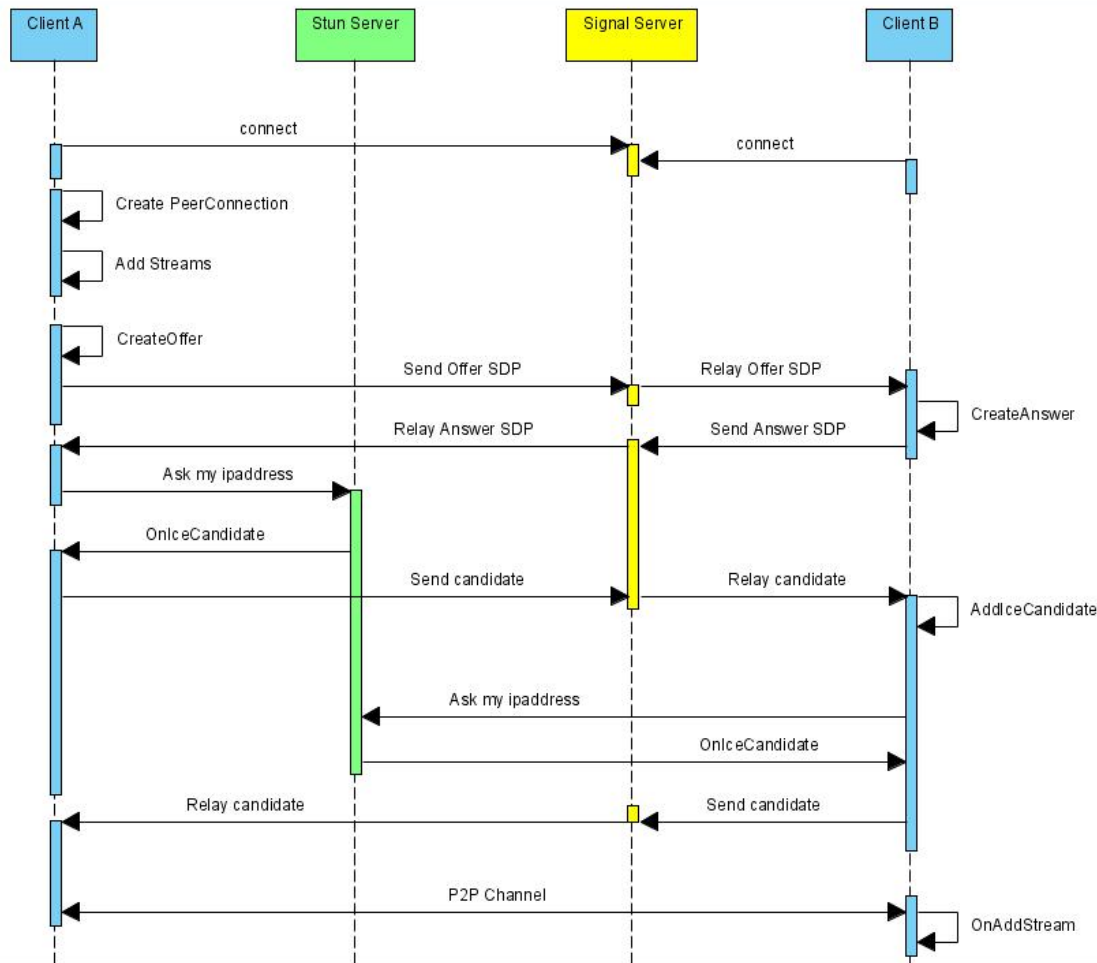


```
82 function call() {
83   callButton.disabled = true;
84   hangupButton.disabled = false;
85   trace('Starting call');
86   startTime = window.performance.now();
87   var videoTracks = localStream.getVideoTracks();
88   var audioTracks = localStream.getAudioTracks();
89   if (videoTracks.length > 0) {
90     trace('Using video device: ' + videoTracks[0].label);
91   }
92   if (audioTracks.length > 0) {
93     trace('Using audio device: ' + audioTracks[0].label);
94   }
95   var servers = null;
96   pc1 = new RTCPeerConnection(servers);
97   trace('Created local peer connection object pc1');
98   pc1.onicecandidate = function(e) {
99     onIceCandidate(pc1, e);
100  };
101  pc2 = new RTCPeerConnection(servers);
102  trace('Created remote peer connection object pc2');
103  pc2.onicecandidate = function(e) {
104    onIceCandidate(pc2, e);
105  };
106  pc1.oniceconnectionstatechange = function(e) {
107    onIceStateChange(pc1, e);
108  };
109  pc2.oniceconnectionstatechange = function(e) {
110    onIceStateChange(pc2, e);
111  };
112  pc2.onaddstream = gotRemoteStream;
113
114  pc1.addStream(localStream);
115  trace('Added local stream to pc1');
116
117  trace('pc1 createOffer start');
118  pc1.createOffer(
119    offerOptions
120  ).then(
121    onCreateOfferSuccess,
122    onCreateSessionDescriptionError
123  );
124 }
```

```
35 function createConnection() {
36     dataChannelSend.placeholder = '';
37     var servers = null;
38     pcConstraint = null;
39     dataConstraint = null;
40     trace('Using SCTP based data channels');
41     // SCTP is supported from Chrome 31 and is supported in FF.
42     // No need to pass DTLS constraint as it is on by default in Chrome 31.
43     // For SCTP, reliable and ordered is true by default.
44     // Add localConnection to global scope to make it visible
45     // from the browser console.
46     window.localConnection = localConnection =
47         new RTCPeerConnection(servers, pcConstraint);
48     trace('Created local peer connection object localConnection');
49
50     sendChannel = localConnection.createDataChannel('sendDataChannel',
51         dataConstraint);
52     trace('Created send data channel');
53
54     localConnection.onicecandidate = iceCallback1;
55     sendChannel.onopen = onSendChannelStateChange;
56     sendChannel.onclose = onSendChannelStateChange;
57
58     // Add remoteConnection to global scope to make it visible
59     // from the browser console.
60     window.remoteConnection = remoteConnection =
61         new RTCPeerConnection(servers, pcConstraint);
62     trace('Created remote peer connection object remoteConnection');
63
64     remoteConnection.onicecandidate = iceCallback2;
65     remoteConnection.ondatachannel = receiveChannelCallback;
66
67     localConnection.createOffer().then(
68         gotDescription1,
69         onCreateSessionDescriptionError
70     );
71     startButton.disabled = true;
72     closeButton.disabled = false;
73 }
74
75 function onCreateSessionDescriptionError(error) {
76     trace('Failed to create session description: ' + error.toString());
77 }
78
79 function sendData() {
80     var data = dataChannelSend.value;
81     sendChannel.send(data);
82     trace('Sent Data: ' + data);
83 }
```

WebRTC介绍

OSC 原创会
年终盛典 2016



WebRTC介绍

OSC 原创会
年终盛典 2016

- 测试地址:

<https://github.com/webRTC/samples>

getUserMedia()

RTCPeerConnection

RTCDataChannel

ICE(Stun,Turn)



• 主要技术难点

推流播放
电子白板
桌面共享
视频美颜
系统并发
卡顿延迟
用户互动
其他.....



WebRTC & 视频直播

OSC 原创会
年终盛典 2016

指数趋势 webrtc 2011-01-01 至 2016-11-13 全国

整体趋势

PC趋势

移动趋势

最近

7天

30天

90天

半年

全部

webrtc

平均值

搜索指数

@index.baidu.com

2011年 2012年 2013年 2014年 2015年 2016年

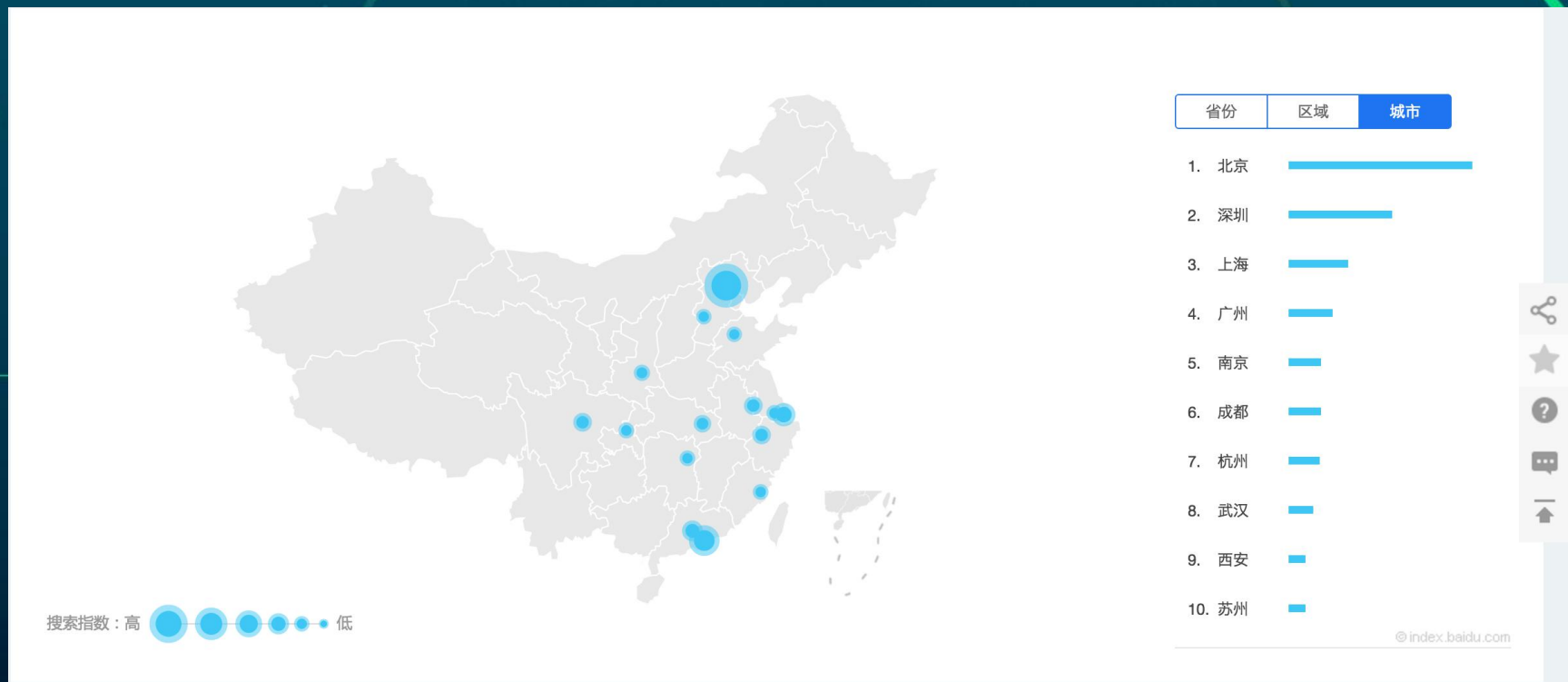
媒体指数

2011 2012 2013 2014 2015 2016



WebRTC & 视频直播

OSC 原创会
年终盛典 2016



WebRTC & 视频直播

OSC 原创会
年终盛典 2016

指数趋势 ? 视频直播 2011-01-01 至 2016-11-13 全国

整体趋势

PC趋势

移动趋势

最近

7天

30天

90天

半年

全部

视频直播

平均值

搜索指数

14,000

2011年

2012年

2013年

2014年

2015年

2016年

媒体指数

72

2011

2012

2013

2014

2015

2016

36

分享

收藏

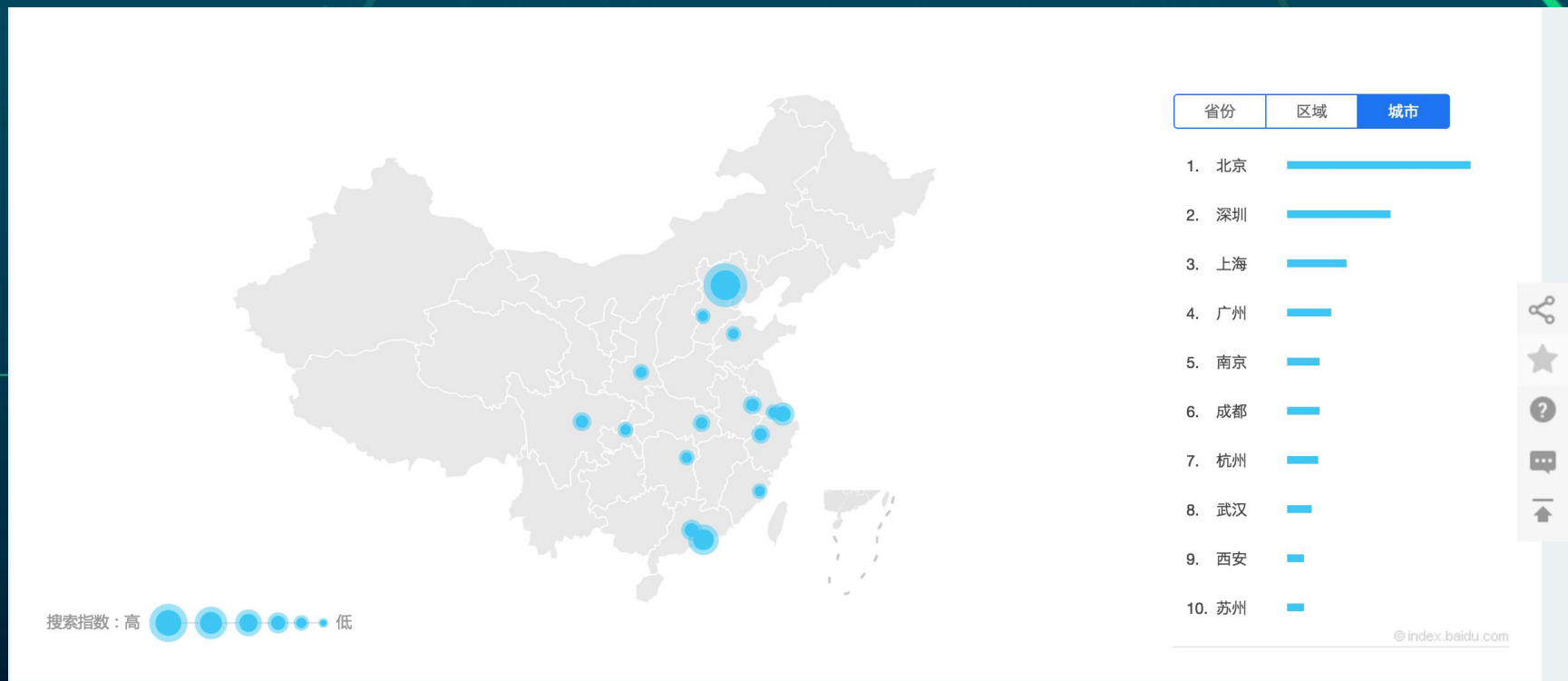
帮助

评论

返回

WebRTC & 视频直播

OSC 原创会
年终盛典 2016



WebRTC & 视频直播

OSC 原创会
年终盛典 2016

- WebRTC提供技术支撑
- 直播创造更多应用场景



编风网

专注WebRTC和音视频领域



欢迎关注我们



谢谢!

