SGCWEBSOCKETS · TECHNICAL DOCUMENT

RTCPeerConnection

TsgcRTCPeerConnection — Delphi-side RTCPeerConnection implementation paired with the W3C WebRTC API for browser interop.

eSeGeCe

WWW.ESEGECE.COM

Overview

The TsgcRTCPeerConnection is a client component that allows you to connect peers using P2P through UDP.

At a glance

COMPONENT CLASS
`TsgcRTCPeerConnection`

STANDARDS / SPEC
WebRTC 1.0 — W3C

TRANSPORTS
TCP, TLS

PLATFORMS
Windows, macOS, Linux, iOS, Android

FRAMEWORKS
VCL, FireMonkey, Lazarus / FPC

EDITION
Standard / Professional / Enterprise

Features

- Native Delphi implementation with full ANSI/Unicode support.

Technical specification

Standards & specs	WebRTC 1.0 — W3C · SDP — RFC 8866
Component class	`TsgcRTCPeerConnection` (unit `sgcRTCPeerConnection`)
Frameworks	VCL, FireMonkey, Lazarus / FPC
Platforms	Windows, macOS, Linux, iOS, Android

Main properties

The principal published / public properties used to configure and drive the component. Consult the online help for the full list.

`RTCOptions`	Connection options: STUN/TURN server, signalling WebSocket endpoint and DTLS encryption settings.
`Version`	Read-only component version string.

Main methods

The principal public methods exposed by the component.

`WriteData()`	Sends data to the remote peer through the negotiated candidate pair.
`GatherCandidates()`	Starts the ICE candidate gathering process.
`Clear()`	Resets the peer-connection state and releases the active channel binding.

Public events

The component exposes the following published events; consult the online help for full event-handler signatures.

`OnRTCCandidatePairFailed`	TsgcRTCPeerConnection › Events › OnRTCCandidatePairFailed
`OnRTCCandidatePairNominated`	TsgcRTCPeerConnection › Events › OnRTCCandidatePairNominated

`OnRTCConnect`	Fired when the data channel is open and ready to send and receive data.
`OnRTCException`	Fired when an exception is raised inside the peer-connection pipeline.
`OnRTCLocalCandidate`	Fired when a local ICE candidate has been gathered; set Accept to False to drop it.
`OnRTCLocalDescription`	Fired when the local SDP offer/answer has been generated and can be edited.
`OnRTCMessage`	Fired when data is received from the remote peer through the data channel.
`OnRTCRemoteCandidate`	Fired when a remote ICE candidate is received; set Accept to False to drop it.
`OnRTCRemoteDescription`	Fired when the remote SDP offer/answer is received through signalling.
`OnRTCWebSocketBeforeConnect`	TsgcRTCPeerConnection › Events › OnRTCWebSocketBeforeConnect
`OnRTCWebSocketConnect`	Fired when the signalling WebSocket has connected to the server.
`OnRTCWebSocketDisconnect`	TsgcRTCPeerConnection › Events › OnRTCWebSocketDisconnect
`OnRTCWebSocketMessage`	Fired when a raw signalling text message is received on the WebSocket.
`OnRTCWebSocketRemoteDisconnect`	TsgcRTCPeerConnection › Events › OnRTCWebSocketRemoteDisconnect

Quick Start

Drop the component on a form, configure the properties below and activate it. The snippet that follows shows the typical **Send Data** configuration sourced from the online help.

> **About this scenario.** Use the method WriteData to send any data to the other peer. You can send a string or an array of bytes.

Delphi (VCL / FireMonkey)

```
oRTCPeerConnection := TsgcRTCPeerConnection.Create(nil);
...
oRTCPeerConnection.WriteData('Hello from sgcWebSockets!!!');
```

C++ Builder

```
TsgcRTCPeerConnection *oRTCPeerConnection = new TsgcRTCPeerConnection();
...
oRTCPeerConnection->WriteData("Hello from sgcWebSockets!!!");
```

.NET (C#)

```
TsgcRTCPeerConnection oRTCPeerConnection = new TsgcRTCPeerConnection();
...
oRTCPeerConnection.WriteData("Hello from sgcWebSockets!!!");
```

Common scenarios

The following scenarios are lifted verbatim from the online help. Each shows the configuration and method calls needed to drive the component through a specific real-world flow.

1 · RTCPeerConnection | WebSocket Server

The TsgcRTCPeerConnection client requires a WebSocket Server for signaling. The client makes use of the WebSocket protocol to exchange the SDP of the peers and the candidates (IPs and ports), which will allow peers to communicate.

Delphi (VCL / FireMonkey)

```
oServer := TsgcWebSocketServer.Create(nil);
oProtocol := TsgcWSPServer_RTCPeerConnection.Create(nil);
oProtocol.Server := oServer;
oServer.Port := 8080;
oServer.Active := True;
```

C++ Builder

```
TsgcWebSocketServer *oServer = new TsgcWebSocketServer();
TsgcWSPServer_RTCPeerConnection *oProtocol = new TsgcWSPServer_RTCPeerConnection();
oProtocol->Server = oServer;
oServer->Port = 8080;
oServer->Active = true;
```

.NET (C#)

```
TsgcWebSocketServer oServer = new TsgcWebSocketServer();
TsgcWSPServer_RTCPeerConnection oProtocol = new TsgcWSPServer_RTCPeerConnection();
oProtocol.Server = oServer;
oServer.Port = 8080;
oServer.Active = true;
```

2 · Receive Data

Every time the TsgcRTCPeerConnection receives any data from the other peer, the event OnRTCMessage will be called.

Delphi (VCL / FireMonkey)

```
procedure OnRTCMessage(Sender: TObject; const aBytes: TBytes);
begin
  ShowMessage(TEncoding.UTF8.GetString(aBytes));
end;
```

C++ Builder

```
void OnRTCMessage(TObject *Sender, const TBytes aBytes)
{
  ShowMessage(TEncoding→UTF8→GetString(aBytes));
}
```

.NET (C#)

```
void OnRTCMessage(TObject Sender, byte[] aBytes)
{
  MessageBox.Show(System.Text.Encoding.Default.GetString(aBytes));
}
```

Sources used to build this document

Every external claim links back to a primary source. The online-help references decode the canonical deep-link the company maintains for this component.

Primary standard / spec — WebRTC 1.0 — W3C	www.w3.org/TR/webrtc/
Primary standard / spec — SDP — RFC 8866	datatracker.ietf.org/doc/html/rfc8866
Online help — component page	www.esegece.com/help/sgcWebSockets/Components/P2P/RTCPeerConnection/TsgcRTCPeerConnection.htm
Delphi demo project (in the sgcWebSockets package)	`Demos\35.P2P\05.RTCPeerConnection`
Component page	www.esegece.com/products/websockets/p2p/rtcpeerconnection/
Product page	www.esegece.com/products/websockets/

> **Document scope.** This document covers the publicly-documented surface of the RTCPeerConnection component shipped with sgcWebSockets. For full property, method and event reference consult the online help linked above.