



1. Introduction

This section is non-normative.

Audio, video, or data packets transmitted over a peer-connection can be lost, and experience varying amounts of network delay. A web application implementing WebRTC expects to monitor the performance of the underlying network and media pipeline.

This document defines the statistic identifiers used by the web application to extract metrics from the user agent.

2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words may, must, and must not are to be interpreted as described in [RFC2119].

This specification defines the conformance criteria that applies to a single product: the user agent.

Implementations that use ECMAScript to implement the objects defined in this specification must implement them in a manner consistent with the ECMAScript Bindings defined in the Web IDL specification [WEBIDL], as this document uses that specification and terminology.

This specification does not define what objects a conforming implementation should generate. Specifications that refer to this specification have the need to specify conformance. They should put in their document text like this:

- An implementation must support generating statistics for the type `RTCIInboundRtpStreamStats`, with attributes `packetsReceived`, `bytesReceived`, `packetsLost`, `jitter`, and `fractionLost`.
- It must support generating statistics for the type `RTCOutboundRtpStreamStats`, with attributes `packetsSent`, `bytesSent`.For all subclasses of `RTCRtpStreamStats`, it must include `ssrc` and `kind`.
- When stats exist for both sides of a connection, in the form of an `inbound-rtp` / `remote-outbound-rtp` pair or an `outbound-rtp` / `remote-inbound-rtp` pair, the members `remoteId` and `localId` must also be present.It may support generating other stats.

3. Terminology

The concepts `queue a task`, and `fires a simple event` are defined in [HTML5].

The terms `event`, `event handlers`, and `event handler event types` are defined in [HTML5].

The terms `MediaStream`, `MediaStreamTrack`, and `Consumer` are defined in [GETUSERMEDIA].

The terms `RTCPeerConnection`, `RTCDataChannel`, `RTCDtlsTransport`, `RTCDtlsTransportState`, `RTCIceTransport`, `RTCIceRole` and `RTCPriorityType` are defined in [WEBRTC].

The term `RTP stream` is defined in [RFC7656] section 2.1.10.

The terms `RTCStats`, `RTCStats.timestamp`, `RTCStats.type`, `RTCStats.id`, `RTCCertificate`, and `statsended` are defined in [WEBRTC].

The terms `performance.timeOrigin` and `performance.now()` are defined in [HIGHRES-TIME].

4. Basic concepts

This section is non-normative.

The basic object of the stats model is the stats object. The following terms are defined to describe it:

Monitored object

An internal object that keeps a set of data values. Most monitored objects are object defined in the WebRTC API; they may be thought of as being internal properties of those objects.

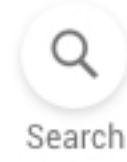
Stats object

This is a set of values, copied out from a monitored object at a specific moment in time. It is returned as a WebIDL dictionary through the `getStats` API call.

Stats object reference

A monitored object has a stable identifier "id", which is reflected in all stats objects produced from the monitored object. Stats objects may contain references to other stats objects using this "id" value. In a stats object, these references are represented by a DOMString containing "id" value of the referenced stats object.

All stats object references have type DOMString and attribute names ending in 'Id', or they have type sequence<DOMString> and attribute names ending in 'Ids'.



Search



File a bug



Github



Pull Request



Commit history

and media pipeline.

This document defines the statistic identifiers used by the web application to extract metrics from the user agent.

2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words may, must, and must not are to be interpreted as described in [RFC2119].

This specification defines the conformance criteria that applies to a single product: the user agent.

Implementations that use ECMAScript to implement the objects defined in this specification must implement them in a manner consistent with the ECMAScript Bindings defined in the Web IDL specification [WEBIDL], as this document uses that specification and terminology.

This specification does not define what objects a conforming implementation should generate. Specifications that refer to this specification have the need to specify conformance. They should put in their document text like this:

- An implementation must support generating statistics for the type `RTCIInboundRtpStreamStats`, with attributes `packetsReceived`, `bytesReceived`, `packetsLost`, `jitter`, and `fractionLost`.
- It must support generating statistics for the type `RTCOOutboundRtpStreamStats`, with attributes `packetsSent`, `bytesSent`.For all subclasses of `RTCRtpStreamStats`, it must include `ssrc` and `kind`.
- When stats exist for both sides of a connection, in the form of an `inbound-rtp` / `remote-outbound-rtp` pair or an `outbound-rtp` / `remote-inbound-rtp` pair, the members `remoteld` and `localId` must also be present.It may support generating other stats.

3. Terminology

The concepts queue a task, and fires a simple event are defined in [HTML5].

The terms event, event handlers, and event handler event types are defined in [HTML5].

The terms `MediaStream`, `MediaStreamTrack`, and `Consumer` are defined in [GETUSERMEDIA].

The terms `RTCPeerConnection`, `RTCDDataChannel`, `RTCDtlsTransport`, `RTCDtlsTransportState`, `RTCIceTransport`, `RTCIceRole` and `RTCPriorityType` are defined in [WEBRTC].

The term RTP stream is defined in [RFC7656] section 2.1.10.

The terms `RTCStats`, `RTCStats.timestamp`, `RTCStats.type`, `RTCStats.id`, `RTCCertificate`, and `statsended` are defined in [WEBRTC].

The terms `performance.timeOrigin` and `performance.now()` are defined in [HIGHRES-TIME].

4. Basic concepts

This section is non-normative.

The basic object of the stats model is the stats object. The following terms are defined to describe it:

Monitored object

An internal object that keeps a set of data values. Most monitored objects are object defined in the WebRTC API; they may be thought of as being internal properties of those objects.

Stats object

This is a set of values, copied out from a monitored object at a specific moment in time. It is returned as a WebIDL dictionary through the `getStats` API call.

Stats object reference

A monitored object has a stable identifier "id", which is reflected in all stats objects produced from the monitored object. Stats objects may contain references to other stats objects using this "id" value. In a stats object, these references are represented by a `DOMString` containing "id" value of the referenced stats object.

All stats object references have type `DOMString` and attribute names ending in 'Id', or they have type `sequence<DOMString>` and attribute names ending in 'Ids'.

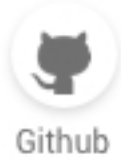
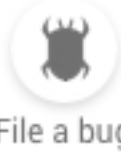
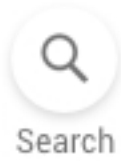


TABLE OF CONTENTS

Contributors	+
Meta Data	+
Reference	+
<hr/>	
1. Introduction	
2. Conformance	
3. Terminology	
4. Basic concepts	
4.1 Guidelines for design of stats objects	
4.2 Guidelines for implementing stats objects	
4.3 Lifetime considerations for monitored objects	
4.4 Guidelines for getStats() results caching/throttling	
5. Maintenance procedures for stats object types	
5.1 Adding new stats objects	
5.2 Retiring stats objects	
6. RTCStatsType	
6.1 RTCStatsType enum	
7. Stats dictionaries	
7.1 The RTP statistics hierarchy	
7.2 RTCRtpStreamStats dictionary	
7.3 RTCCodecStats dictionary	
7.3.1 RTCCodecType enum	
7.4 RTCReceivedRtpStream Stats dictionary	
7.5 RTCInboundRtpStream Stas dictionary	
7.6 RTCRemoteInboundRtp StreamStats dictionary	
7.7 RTCSentRtpStreamStats dictionary	
7.8 RTCOutboundRtpStream Stats dictionary	
7.9 RTCQualityLimitation Reason enum	

1. Introduction

This section is non-normative.

Audio, video, or data packets transmitted over a peer-connection can be lost, and experience varying amounts of network delay. A web application implementing WebRTC expects to monitor the performance of the underlying network and media pipeline.

This document defines the statistic identifiers used by the web application to extract metrics from the user agent.

2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words may, must, and must not are to be interpreted as described in [RFC2119].

This specification defines the conformance criteria that applies to a single product: the user agent.

Implementations that use ECMAScript to implement the objects defined in this specification must implement them in a manner consistent with the ECMAScript Bindings defined in the Web IDL specification [WEBIDL], as this document uses that specification and terminology.

This specification does not define what objects a conforming implementation should generate. Specifications that refer to this specification have the need to specify conformance. They should put in their document text like this:

- An implementation must support generating statistics for the type RTCInboundRtpStreamStats, with attributes packetsReceived, bytesReceived, packetsLost, jitter, and fractionLost.
- It must support generating statistics for the type RTCOutboundRtpStreamStats, with attributes packetsSent, bytesSent.For all subclasses of RTCRtpStreamStats, it must include ssrc and kind.
- When stats exist for both sides of a connection, in the form of an inbound-rtp / remote-outbound-rtp pair or an outbound-rtp / remote-inbound-rtp pair, the members remoteld and localld must also be present.It may support generating other stats.

3. Terminology

The concepts queue a task, and fires a simple event are defined in [HTML5].

The terms event, event handlers, and event handler event types are defined in [HTML5].

The terms MediaStream, MediaStreamTrack, and Consumer are defined in [GETUSERMEDIA].

The terms RTCPeerConnection, RTCDataChannel, RTCDtlsTransport, RTCDtlsTransportState, RTCIceTransport, RTCIceRole and RTPriorityType are defined in [WEBRTC].

The term RTP stream is defined in [RFC7656] section 2.1.10.

The terms RTCStats, RTCStats.timestamp, RTCStats.type, RTCStats.id, RTCCertificate, and statsended are defined in [WEBRTC].

The terms performance.timeOrigin and performance.now() are defined in [HIGHRES-TIME].

4. Basic concepts

This section is non-normative.

The basic object of the stats model is the stats object. The following terms are defined to describe it:

Monitored object

An internal object that keeps a set of data values. Most monitored objects are object defined in the WebRTC API; they may be thought of as being internal properties of those objects.

Stats object

This is a set of values, copied out from a monitored object at a specific moment in time. It is returned as a WebIDL dictionary through the getStats API call.

Stats object reference

A monitored object has a stable identifier "id", which is reflected in all stats objects produced from the monitored object. Stats objects may contain references to other stats objects using this "id" value. In a stats object, these references are represented by a DOMString containing "id" value of the referenced stats object.

All stats object references have type DOMString and attribute names ending in 'Id', or they have type sequence<DOMString> and attribute names ending in 'Ids'.

Within specification



Search



File a bug



Github



Pull Request



Commit history

1. Introduction

This section is non-normative.

Audio, video, or data packets transmitted over a peer-connection can be lost, and experience varying amounts of network delay. A web application implementing WebRTC expects to monitor the performance of the underlying network and media pipeline.

This document defines the statistic identifiers used by the web application to extract metrics from the user agent.

2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words may, must, and must not are to be interpreted as described in [RFC2119].

This specification defines the conformance criteria that applies to a single product: the user agent.

Implementations that use ECMAScript to implement the objects defined in this specification must implement them in a manner consistent with the ECMAScript Bindings defined in the Web IDL specification [WEBIDL], as this document uses that specification and terminology.

This specification does not define what objects a conforming implementation should generate. Specifications that refer to this specification have the need to specify conformance. They should put in their document text like this:

- An implementation must support generating statistics for the type `RTCIInboundRtpStreamStats`, with attributes `packetsReceived`, `bytesReceived`, `packetsLost`, `jitter`, and `fractionLost`.
- It must support generating statistics for the type `RTCOOutboundRtpStreamStats`, with attributes `packetsSent`, `bytesSent`.For all subclasses of `RTCRtpStreamStats`, it must include `ssrc` and `kind`.
- When stats exist for both sides of a connection, in the form of an `inbound-rtp` / `remote-outbound-rtp` pair or an `outbound-rtp` / `remote-inbound-rtp` pair, the members `remoteId` and `localId` must also be present.It may support generating other stats.

3. Terminology

The concepts `queue a task`, and `fires a simple event` are defined in [HTML5].

The terms `event`, `event handlers`, and `event handler event types` are defined in [HTML5].

The terms `MediaStream`, `MediaStreamTrack`, and `Consumer` are defined in [GETUSERMEDIA].

The terms `RTCPeerConnection`, `RTCDataChannel`, `RTCDtlsTransport`, `RTCDtlsTransportState`, `RTCIceTransport`, `RTCIceRole` and `RTCPriorityType` are defined in [WEBRTC].

The term `RTP stream` is defined in [RFC7656] section 2.1.10.

The terms `RTCStats`, `RTCStats.timestamp`, `RTCStats.type`, `RTCStats.id`, `RTCCertificate`, and `statsended` are defined in [WEBRTC].

The terms `performance.timeOrigin` and `performance.now()` are defined in [HIGHRES-TIME].

4. Basic concepts

This section is non-normative.

The basic object of the stats model is the stats object. The following terms are defined to describe it:

Monitored object

An internal object that keeps a set of data values. Most monitored objects are object defined in the WebRTC API; they may be thought of as being internal properties of those objects.

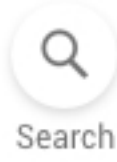
Stats object

This is a set of values, copied out from a monitored object at a specific moment in time. It is returned as a WebIDL dictionary through the `getStats` API call.

Stats object reference

A monitored object has a stable identifier `"id"`, which is reflected in all stats objects produced from the monitored object. Stats objects may contain references to other stats objects using this `"id"` value. In a stats object, these references are represented by a `DOMString` containing `"id"` value of the referenced stats object.

All stats object references have type `DOMString` and attribute names ending in `'Id'`, or they have type `sequence<DOMString>` and attribute names ending in `'Ids'`.



Search



File a bug



Github



Pull Request



Commit history

Identifiers for WebRTC's Statistics API

TABLE OF CONTENTS

- Contributors
- Meta Data
- Reference

1. Introduction
2. Conformance
3. Terminology
4. Basic concepts

4.1 Guidelines for design of stats objects

4.2 Guidelines for implementing stats objects

4.3 Lifetime considerations for monitored objects

4.4 Guidelines for getStats() results caching/throttling
5. Maintenance procedures for stats object types

5.1 Adding new stats objects

5.2 Retiring stats objects
6. RTCStatsType

6.1 RTCStatsType enum
7. Stats dictionaries

7.1 The RTP statistics hierarchy

7.2 RTCRtpStreamStats dictionary

7.3 RTCCodecStats dictionary

7.3.1 RTCCodecType enum

7.4 RTCReceivedRtpStream Stats dictionary

7.5 RTCInboundRtpStream Stas dictionary

7.6 RTCRemoteInboundRtp StreamStats dictionary

7.7 RTCSentRtpStreamStats dictionary

7.8 RTCOutboundRtpStream Stats dictionary

7.9 RTCQualityLimitation Reason enum

64 Specs (of 1143)

HTML Accessibility API Mappings 1.0

WD

Web Platform Working Group

2018-05-03 - [History](#) - [Editor's Draft](#)
Steve Faulkner , Jason Kiss , Alexander Surkov , Bogdan Brinza , Cynthia Shelly

Accessibility HTML

HTML 5.3

WD

Web Platform Working Group

2018-04-26 - [History](#) - [Editor's Draft](#)
Patricia Aas , Shwetank Dixit , Terence Eden , Bruce Lawson , Sangwhan Moon , Xiaoqian Wu , Scott O'Hara

HTML

HTML Microdata

WD

Web Platform Working Group

2018-04-26 - [History](#) - [Editor's Draft](#)
Charles McCathie Nevile , Dan Brickley , Ian Hickson

HTML

ARIA in HTML

WD

Web Platform Working Group

2018-04-23 - [History](#) - [Editor's Draft](#)
Steve Faulkner

Accessibility

XHTML™ 1.1 - Module-based XHTML - Second Edition

ret

XHTML2 Working Group

2018-03-27 - [History](#)
Masayasu Ishikawa , Shane McCarron

HTML

XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)

ret

XHTML2 Working Group

2018-03-27 - [History](#)
Steven Pemberton

HTML

1 2 3 4 5 >

This specification does not define what objects a conforming implementation should generate. Specifications that refer to this specification have the need to specify conformance. They should put in their document text like this:

- An implementation must support generating statistics for the type `RTCIInboundRtpStreamStats`, with attributes `packetsReceived`, `bytesReceived`, `packetsLost`, `jitter`, and `fractionLost`.
- It must support generating statistics for the type `RTCOutboundRtpStreamStats`, with attributes `packetsSent`, `bytesSent`.For all subclasses of `RTCRtpStreamStats`, it must include `ssrc` and `kind`.
- When stats exist for both sides of a connection, in the form of an inbound-rtp / remote-outbound-rtp pair or an outbound-rtp / remote-inbound-rtp pair, the members `remoteId` and `localId` must also be present.It may support generating other stats.

3. Terminology

The concepts `queue` a task, and fires a simple event are defined in [HTML5].

The terms `event`, `event handlers`, and `event handler event types` are defined in [HTML5].

The terms `MediaStream`, `MediaStreamTrack`, and `Consumer` are defined in [GETUSERMEDIA].

The terms `RTCPeerConnection`, `RTCDataChannel`, `RTCDtlsTransport`, `RTCDtlsTransportState`, `RTCIceTransport`, `RTCIceRole` and `RTCPriorityType` are defined in [WEBRTC].

The term `RTP stream` is defined in [RFC7656] section 2.1.10.

The terms `RTCStats`, `RTCStats.timestamp`, `RTCStats.type`, `RTCStats.id`, `RTCCertificate`, and `statsended` are defined in [WEBRTC].

The terms `performance.timeOrigin` and `performance.now()` are defined in [HIGHRES-TIME].

4. Basic concepts

This section is non-normative.

The basic object of the stats model is the stats object. The following terms are defined to describe it:

Monitored object

An internal object that keeps a set of data values. Most monitored objects are object defined in the WebRTC API; they may be thought of as being internal properties of those objects.

Stats object

This is a set of values, copied out from a monitored object at a specific moment in time. It is returned as a WebIDL dictionary through the `getStats` API call.

Stats object reference

A monitored object has a stable identifier "id", which is reflected in all stats objects produced from the monitored object. Stats objects may contain references to other stats objects using this "id" value. In a stats object, these references are represented by a DOMString containing "id" value of the referenced stats object.

All stats object references have type DOMString and attribute names ending in 'Id', or they have type `sequence<DOMString>` and attribute names ending in 'Ids'.

Commit history