Setup speed test on your website as API

1. Take the code and insert it on your website.

<script> sc_api_mode='1'; var sc_script = document.createElement("script"); sc_script.setAttribute("src", ("https:" == document.location.protocol ? "https" : "http") + "://www.speedcheckercdn.com/speedchecker.api.js"), sc_script.addEventListener("load", function () { window.SCAPI.init() }),

2. To start speed test, you need to call startTest() method:

```
window.SCApplication.startTest()
```

3. You can add event listeners in your code and get information in real time about speed test progress. More detailed info about events and methods you can check below.

The embedded web version of the Speedchecker Speed Test supports its own API which allows you to access various methods and events easily from the javascript code on your hosted page. Also, it’s possible to call the API from the browser’s console which is used typically for development purposes.

Interfaces

SavedTakenTest

This interface represents the JSON data of a unique completed speed test.

SavedTakenTest.TakenTest.Ping.time
Is a number containing the time in milliseconds of a visitor’s latency (ping).

SavedTakenTest.TakenTest.Download.speedInKbps
Is a number containing the speed in Kilobits per second of a visitor’s download.

SavedTakenTest.TakenTest.Upload.speedInKbps
Is a number containing the speed in Kilobits per second of a visitor’s upload.

SavedTakenTest.TakenTest.User.CountryCode
Is a string containing the code in ISO 3166-1 alpha-2 of a visitor's country.

SavedTakenTest.Provider.Title
Is a string containing the title of a visitor's internet service provider.

Methods

startTest()

This function starts the speed test from your code, simulating the visitor’s click on the Start Speed Test button. It has no argument. To implement please copy&paste the following script where you want the speed test to start.

Syntax

```
window.SCApplication.startTest()
```

Events

speedcheckerReady ()

This function is called once the speed test has been prepared and is ready to use.

Syntax

```
window.speedcheckerReady= function(data) {
    // Your code is here, e.g. you can start test

    window.SCApplication.startTest();
}
```

speedcheckerTakenTestSaved()

This function is called once the speed test has been completed. It has just one argument which takes a **SavedTakenTest** data object. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerTakenTestSaved = function(data) {
    // Your code here
}
```

Example

```
window.speedcheckerTakenTestSaved = function(data) {
    console.log("Test has just finished, following some details.");
    console.log("Visitor's ISP:", data.Provider.Title);
    console.log("Visitor's country:", data.TakenTest.User.CountryCode);

    console.log("Ping:", data.TakenTest.Ping.time, "ms");
    console.log("Download:", data.TakenTest.Download.speedInKbps);
    console.log("Upload:", data.TakenTest.Upload.speedInKbps);
}
```

speedcheckerPingStarted()

This function is called once the Ping test has been started. It has no argument. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerPingStarted = function() {
    // Your code here
}
```

speedcheckerPingFinished()

This function is called once the Ping test has been completed. It has three arguments which takes a server name, server country and speed in milliseconds. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerPingFinished = function(name, country, pingInMs) {
    // Your code here
}
```

Example

```
window.speedcheckerPingFinished = function(name, country, pingInMs) {
    console.log("Ping test has just finished, following some details.");
    console.log("Server name:", name);
    console.log("Server country:", country);
    console.log("Ping:", pingInMs, "ms");
}
```

speedcheckerDownloadStarted()

This function is called once the Download test has been started and will be preparing. It has no argument. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerDownloadStarted = function() { … }
```

speedcheckerDownloadPrepared()

This function is called once the Download test has been prepared and will be processing. It has no argument. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerDownloadPrepared = function() { … }
```

speedcheckerDownloadProgress()

This function is called once the Download test is being processing. It has just one argument which takes a speed data in Kilobits per second. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerDownloadProgress = function(speedInKbps) { … }
```

speedcheckerDownloadFinished()

This function is called once the Download test has been completed. It has just one argument which takes a speed data in Kilobits per second. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerDownloadFinished = function(speedInKbps) { … }
```

speedcheckerUploadStarted()

This function is called once the Upload test has been started and will be preparing. It has no argument. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerUploadStarted = function() { … }
```

speedcheckerUploadPrepared()

This function is called once the Upload test has been prepared and will be processing. It has no argument. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerUploadPrepared = function() { … }
```

speedcheckerUploadProgress()

This function is called once the Upload test is being processing. It has just one argument which takes a speed data in Kilobits per second. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerUploadProgress = function(speedInKbps) { … }
```

speedcheckerUploadFinished()

This function is called once the Upload test has been completed. It has just one argument which takes a speed data in Kilobits per second. To implement please copy&paste the following script.

Syntax

```
window.speedcheckerUploadFinished = function(speedInKbps) { … }
```

Errors

speedcheckerError()

This function is called when any error occurs. The function takes exactly one argument and this must be an error object. Error object has the following format: { type: String } To implement please copy & paste the following script.

Syntax

```
window.speedcheckerError = function(error) { ... }
```

Error.type = LoadError
This error occurs if initialisation part (loading UI, parsing arguments, etc.) fails.

Error.type = PingError
This error occurs if ping test fails.

Error.type = DownloadError
This error occurs if download test fails.

Error.type = DownloadErrorAfterRetry
This error occurs if download test fails after retry.

Error.type = UploadError
This error occurs if upload test fails.

Error.type = UploadServerError
This error occurs if upload test fails after retry.

Example

```
window.speedcheckerError = function (error) {
    if (error.type == "LoadError") {
          console.log("Loading did fail…");
    }
}
```

Optional – Measuring High Latency connections

For scenarios where the majority of users are connecting using a high latency network (e.g. satellite connection) you might get better results using the following parameters to enable the "high latency" version of the algorithm:

window.sc_highLatency = true;

window.sc_maxDownloadDuration = 30000;

window.sc_maxUploadDuration = 30000;

Other parameters

window.serverUrl // Add the serverUrl parameter to match your serverURL e.g.

```
window.serverUrl = "https://www.myserverdomain.com";
```

(if you did not install SSL on your server, change to http://....)

window.sc_forceSecureWebsockets // If you did not install SSL certificates in your server, you will need to add this parameter to force websocket connections on unsecured protocol:

```
window. sc_forceSecureWebsockets = 0;
```

window.serverDownloadFolder

window.serverUploadFolder // Add correct location of files for HTTP fallback:

```
window.serverDownloadFolder = "/";
window.serverUploadFolder = "/";
```

window.sc_enableWebsockets // Possible values: 1, 0. By default is 1.

Websockets are enabled by default. If server supports it, speed test will run via websockets, but if you want to disable websockets, you need to set this parameter to 0, e.g.

```
window.sc_enableWebsockets = 0;
```

window.serverTitle // Add the serverTitle parameter which will indicate what will be to the user:

```
window.serverTitle = "London Server";
```

window.onlyDownload // Possible values: true/false. By default is false.

If value is true, upload phase will be ignored.

window.onlyUpload // Possible values: true/false. By default is false.

If value is true, download phase will be ignored.