

How to set the webcam resolution

Note that as of June 2015, this only works with Google Chrome, but it should come soon on the other browsers that support the getUserMedia API.

It is possible to set "hints" for the preferred resolution during video capture. This is done by using a ["constraint" object](#) that is passed as a parameter to the `getUserMedia(...)` method. It's just the same object we passed in the basic [example](#): `navigator.getUserMedia({video:true}, success, error)` except that this time this object is a little more complex.

For more information, [this article on HTML5rocks.com about the getUserMedia API](#) gives extra examples on how to set the camera resolution. Also check [this good article that tested systematically a set of "preferred resolutions"](#) and compared them to the actual resolutions returned by the browser. Remember that the requested resolution is a hint, and there is no real guarantee that your configuration will allow it.

Typical use:

```
var constraint = {
  video: {
    mandatory: {
      maxWidth: 320,
      maxHeight: 200
    }
  }
};

10. navigator.getUserMedia(constraint, success, error);

function sucess(stream) {
```

```
        video.src =window.URL.createObjectURL(stream);  
    }  
  
    function error(error) {  
        console.log('navigator.getUserMedia error: ', error);  
    }
```

COMPLETE EXAMPLE: CHOOSE BETWEEN 3 DIFFERENT RESOLUTIONS

[Online example at JS Bin](#)

jsbin.com/jigiru/5/edit?output

Applications https://vegas.univ-tls manifestR Films et Serie en Str

File Add library Share

Output

Set the camera resolution

Example adapted from: <http://www.simpl.info/getusermedia/constraints/>

Click a button to call `getUserMedia()` with appropriate resolution.

Actual video dimensions: 320x240px.

A man in a grey shirt is holding a white sign that reads "HELP! PLEASE WATCH ME IN HD!". He is standing in front of a whiteboard with some writing on it.

HTML code:

```
<!DOCTYPE html>
<html>
<head>
  <title>getUserMedia constraints for choosing
resolution</title>
</head>
<body onload="init();">
  <h1>Set the camera resolution</h1>
```

Example adapted from:

```
10. <a href="http://www.simpl.info/getusermedia/constraints/">
    http://www.simpl.info/getusermedia/constraints/
</a>
<br>
<p>Click a button to call<code>getUserMedia()</code> with
appropriate
    resolution.</p>

    <div id="buttons">
        <button id="qvga">QVGA</button>
        <button id="vga">VGA</button>
        <button id="hd">HD</button>
    </div>
21. <p id="dimensions"></p>

    <video autoplay></video>
</body>
</html>
```

JavaScript code:

```
var vgaButton, qvgaButton, hdButton, dimensions, video, stream;

function init() {
    vgaButton =document.querySelector('button#vga');
    qvgaButton =document.querySelector('button#qvga');
    hdButton =document.querySelector('button#hd');
    dimensions =document.querySelector('p#dimensions');
    video =document.querySelector('video');
10. navigator.getUserMedia =navigator.getUserMedia ||

    navigator.webkitGetUserMedia ||navigator.mozGetUserMedia;
    // Defines event listeners for the buttons that set the
    resolution

    qvgaButton.onclick = function() {
        getMedia(qvgaConstraints);
    };
};
```

```

    vgaButton.onclick = function() {
        getMedia(vgaConstraints);
20.    };
    hdButton.onclick = function() {
        getMedia(hdConstraints);
    };

    // Trick: regularly check the size of the video element and
    // display it
    // When getUserMedia is called the video element changes its
    // size but for
    // a while its size is zero pixels... so we check every
    // half a second
    video.addEventListener('play',function() {
30.        setTimeout(function() {
            displayVideoDimensions();
        }, 500);
    });
}

// The different values for the constraints on resolution
var qvgaConstraints = {
    video: {
40.        mandatory: {
            maxWidth: 320,
            maxHeight: 180
        }
    }
};

var vgaConstraints = {
    video: {
        mandatory: {
50.            maxWidth: 640,
            maxHeight: 360
        }
    }
};

var hdConstraints = {

```

```

        video: {
            mandatory: {
                minWidth: 1280,
60.         minHeight: 720
            }
        }
    };

    // The function that is called when a button has been clicked:
    starts the video
    // with the preferred resolution
    function getMedia(constraints) {
        if (!!stream) {
            video.src = null;
70.         stream.stop();
        }

        navigator.getUserMedia(constraints, successCallback, errorCallback);
    }

    // callback if the capture is a success or an error
    function successCallback(stream) {
        window.stream = stream; // For resetting it later if we
        change the resolution
        video.src = window.URL.createObjectURL(stream);
    }

81. function errorCallback(error) {
        console.log('navigator.getUserMedia error: ', error);
    }

    // until function that is called by the setInterval(...) every
    0.5s, for
    // displaying the video dimensions
    function displayVideoDimensions() {
        dimensions.innerHTML = 'Actual video dimensions:
        ' + video.videoWidth +
        'x' + video.videoHeight + 'px.';
    }

```