How to set the webcam resolution

Note that as of June 2015, this only works with Google Chrome, but 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. Check also 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.

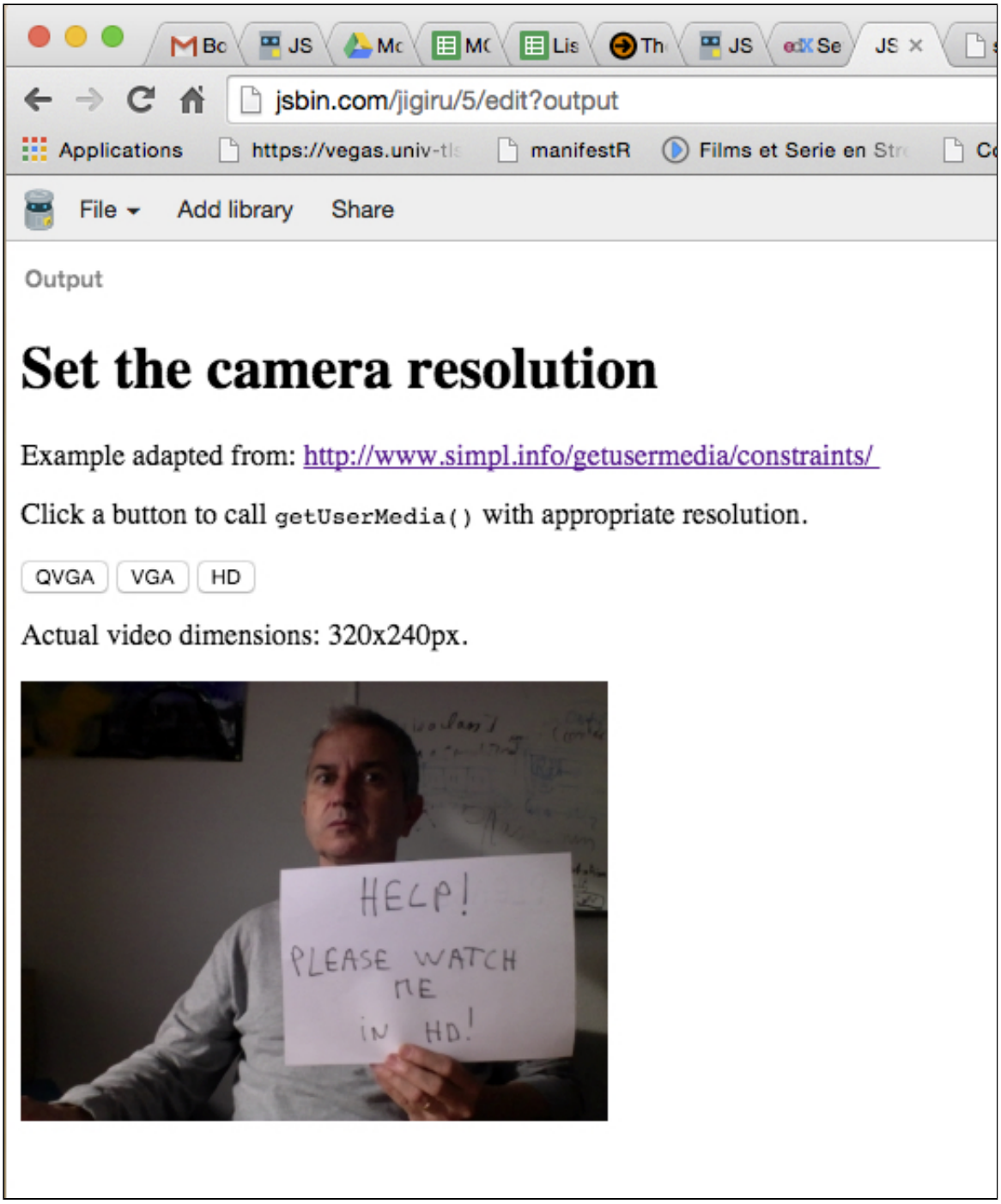
Typical use:

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

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



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:
  <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>

  <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');
  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);
  };
  hdButton.onclick = function() {
    getMedia(hdConstraints);
  };

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


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

var vgaConstraints = {
  video: {
    mandatory: {
      maxWidth: 640,
```

```
      maxHeight: 360
    }
  }
};

var hdConstraints = {
  video: {
    mandatory: {
      minWidth: 1280,
      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;
    stream.stop();
  }
  navigator.getUserMedia(constraints, successCallback, errorCallback);
}

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

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

// util 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.';
}
```