ESTI019 – QS2020 – Prof. Minami

LAB2 – IMAGEM E VIDEO NO COLAB – PYTHON – GRAVAR DO BROWSER

1. Selecionar cores HTML:

https://imagecolorpicker.com/?fbclid=IwAR2A6BVv1hLysZrfZFuQ_v2S4TbebpeBLj9kWfIP65MCwMa_KZItiYCr2j0

2. Gravar Arquivo do browser:

https://developer.mozilla.org/en-US/docs/Web/API/MediaStream_Recording_API/Recording_a_media_element

3. Fazer o download e Converter para o formato mp4:

https://convertio.co/pt/download/d7eb0266a724e277f8adc4397b56ba98acf727/

download e usar no colab

Os códigos estão no ANEXO.

ANEXO: Códigos

HTML:

```
<!-- Learn about this code on MDN: https://developer.mozilla.org/en-US/docs/Web/API/MediaStream_Recording_API/Recording_a_media_element -->

<p>Click the "Start" button to begin video recording for a few seconds. You can stop the video by clicking the creatively-named "Stop" button. The "Download" button will download the received data (although it's in a raw, unwrapped form that isn't very useful).
</p>
<br>
<div class="left">
  <div id="startButton" class="button">
    Start
  </div>
  <h2>Preview</h2>
  <video id="preview" width="320" height="240" autoplay muted></video>
</div>
<div class="right">
  <div id="stopButton" class="button">
    Stop
  </div>
  <h2>Recording</h2>
  <video id="recording" width="320" height="240 " controls></video>
  <a id="downloadButton" class="button">
    Download
  </a>
</div>
<div class="bottom">
  <pre id="log"></pre>
</div>
```

CSS:

```
body {
  font: 14px "Open Sans", "Arial", sans-serif;
}

video {
  margin-top: 2px;
  border: 1px solid black;
}

.button {
  cursor: pointer;
  display: block;
  width: 320px;
  border: 1px solid black;
  font-size: 16px;
  text-align: center;
  padding-top: 2px;
  padding-bottom: 4px;
  color: white;
  background-color: darkgreen;
  text-decoration: none;
}

h2 {
  margin-bottom: 4px;
}

.left {
  margin-right: 10px;
  float: left;
  width: 320px;
  padding: 0px;
}

.right {
  margin-left: 10px;
  float: left;
  width: 320px;
  padding: 0px;
}

.bottom {
  clear: both;
  padding-top: 10px;
}
```

JS

```
let preview = document.getElementById("preview");
let recording = document.getElementById("recording");
let startButton = document.getElementById("startButton");
let stopButton = document.getElementById("stopButton");
let downloadButton = document.getElementById("downloadButton");
let logElement = document.getElementById("log");
let recordingTimeMS = 10000;
function log(msg) {
  logElement.innerHTML += msg + "\n";
}
function wait(delayInMS) {
  return new Promise(resolve => setTimeout(resolve, delayInMS));
}
function startRecording(stream, lengthInMS) {
  let recorder = new MediaRecorder(stream);
  let data = [];
  recorder.ondataavailable = event => data.push(event.data);
  recorder.start();
  log(recorder.state + " for " + (lengthInMS/1000) + " seconds...");
  let stopped = new Promise((resolve, reject) => {
    recorder.onstop = resolve;
    recorder.onerror = event => reject(event.name);
  });
  let recorded = wait(lengthInMS).then(
    () => recorder.state == "recording" && recorder.stop()
  );
  return Promise.all([
    stopped,
    recorded
  ])
  .then(() => data);
}
function stop(stream) {
  stream.getTracks().forEach(track => track.stop());
}
startButton.addEventListener("click", function() {
  navigator.mediaDevices.getUserMedia({
    video: true,
    audio: true
  }).then(stream => {
    preview.srcObject = stream;
    downloadButton.href = stream;
    preview.captureStream = preview.captureStream || preview.mozCaptureStream;
    return new Promise(resolve => preview.onplaying = resolve);
  }).then(() => startRecording(preview.captureStream(), recordingTimeMS))
  .then (recordedChunks => {
    let recordedBlob = new Blob(recordedChunks, { type: "video/webm" });
    recording.src = URL.createObjectURL(recordedBlob);
    downloadButton.href = recording.src;
    downloadButton.download = "RecordedVideo.webm";

    log("Successfully recorded " + recordedBlob.size + " bytes of " +
        recordedBlob.type + " media.");
  })
  .catch(log);
}, false);stopButton.addEventListener("click", function() {
  stop(preview.srcObject);
}, false);
```