

Reading the content of a `TEXTTRACK`

A `TEXTTRACK` OBJECT HAS DIFFERENT PROPERTIES AND METHODS:

- `kind`: equivalent to the `kind` attribute of HTML track elements. Its value is either "subtitles", "caption", "descriptions", "chapters", or "metadata". We will see examples of chapters, descriptions and metadata tracks in subsequent lessons.
- `label`: the label of the track, equivalent of the `label` attribute of HTML track elements.
- `language`: the language of the text track, equivalent to the `srclang` attribute of HTML track elements (be careful: it's not the same spelling!)
- `mode`: explained earlier. Can have values equal to: "disabled" | "hidden" | "showing". Can force a track to be loaded (by setting the mode to "hidden" or "showing").
- `cues`: get a list of cues as a `TextTrackCueList` object. This is the complete content of the WebVTT file!
- `activeCues`: used in event listeners while the video is playing. Corresponds to the cues located in the current time segment. The start and end times of cues can overlap. In reality this may rarely happen, but this property exists in case it does, returning a `TextTrackCueList` object that contains all active tracks at a given time.
- `addCue(cue)`: add a cue to the list of cues.
- `removeCue(cue)`: remove a cue from the list of cues.
- `getCueById(id)`: returns the cue with a given id (not implemented by all browsers - a polyfill is given in the examples from the next lessons).

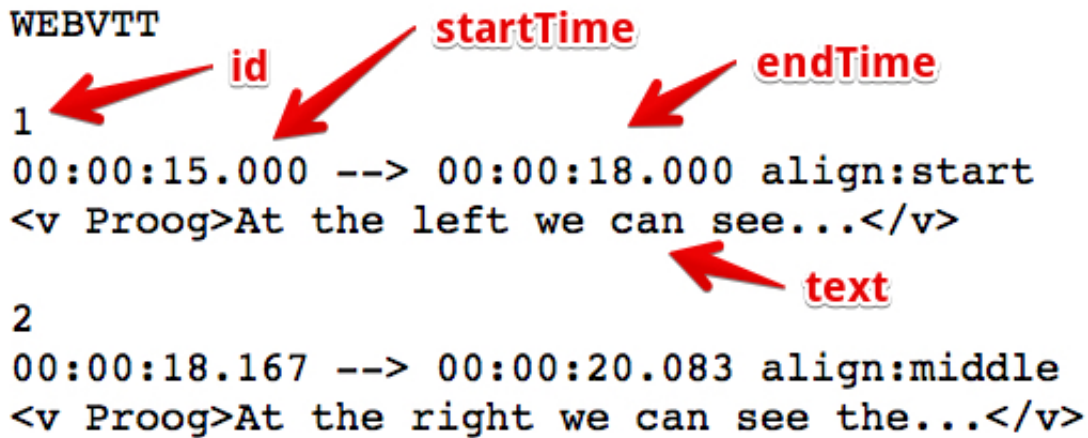
A `TEXTTRACKCUELIST` IS A COLLECTION OF CUES, EACH OF WHICH HAS DIFFERENT PROPERTIES AND METHODS

- `id`: the cue id as written in the line that starts cues in the WebVTT file.
- `startTime` and `endTime`: define the time segment for the cue, in seconds, as a floating point value. It is not the formatted String we have in the WebVTT file (see

screenshot below),

- `text`: the cue content.
- `getCueAsHTML()`: a method that returns an HTML version of the cue content, not as plain text.
- Others such as `align`, `line`, `position`, `size`, `snapToLines`, etc., that correspond to the position of the cue, as specified in the WebVTT file. See the HTML5 course Part 1 about cue positioning.

WEBVTT



```
1
00:00:15.000 --> 00:00:18.000 align:start
<v Proog>At the left we can see...</v>

2
00:00:18.167 --> 00:00:20.083 align:middle
<v Proog>At the right we can see the...</v>
```

The diagram illustrates the structure of WebVTT cues. Red arrows point from labels to specific parts of the cue lines: 'id' points to the cue number '1', 'startTime' points to the start time '00:00:15.000', 'endTime' points to the end time '00:00:18.000', and 'text' points to the cue content 'At the left we can see...'. The second cue follows a similar pattern with '2', '00:00:18.167', '00:00:20.083', and 'At the right we can see the...'.

EXAMPLE THAT DISPLAYS THE CONTENT OF A TRACK

[Here is an example at JSBin that displays the content of a track:](#)



HTML track descriptions

Force load track 0

Force load track 2

1
15 => 18
At the left we can see...

**Click on a button to force load
the track and display its
content (list of cues)**

2
18.167 => 20.083
At the right we can see the...

3
20.083 => 22
...the head-snarlers

We just changed the content of the `readContent(track)` method from the example from the previous lesson:

```
function readContent(track) {  
  console.log("reading content of loaded track...");  
  //displayTrackStatuses(htmlTracks);  
  // instead of displaying the track statuses, we display  
  // in the same div, the track content//  
  // first, empty the div
```

```

        trackStatusesDiv.innerHTML = "";
        // get the list of cues for that track
        var cues = track.cues;
        // iterate on them
        for(var i=0; i < cues.length; i++) {
            // current cue
            var cue = cues[i];
            var id = cue.id + "<br>";
            var timeSegment = cue.startTime + " =>
17. " + cue.endTime + "<br>";
            var text = cue.text + "<p>"

        trackStatusesDiv.innerHTML += id + timeSegment + text;
    }
}

```

As you can see, the code is simple: you first get the cues for the given `TextTrack` (it must be loaded; this is the case since we took care of it earlier), then iterate on the list of cues, and use the `id`, `startTime`, `endTime` and `text` properties of each cue.

This technique will be used in one of the next lessons, and we will show you how to make a clickable transcript on the side of the video - something quite similar to what the edX video player does.

KNOWLEDGE CHECK 1.2.4

What is the name of the `TextTrack` property that returns the list of all its cues?

☐ `activeCues`

☐ `cues`

☐ `cueList`

