Package 'video'

February 2, 2023

Type Package

Title 'Shiny' Extension of 'video.js'

Version 0.1.1

Description Video interactivity within 'shiny' applications using 'video.js'. Enables the status of the video to be sent from the UI to the server, and allows events such as playing and pausing the video to be triggered from the server.

License Apache License (>= 2)

URL https://github.com/ashbaldry/video,
       https://github.com/videojs/video.js

BugReports https://github.com/ashbaldry/video/issues

Encoding UTF-8

Depends R (>= 2.10)

Imports shiny, htmlwidgets, jsonlite

Suggests rmarkdown, knitr, shinytest2, testthat (>= 3.0.0)

Language en-GB

RoxygenNote 7.2.3

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Ashley Baldry [aut, cre],
       Steve Heffernan [aut] (Creator of video.js)

Maintainer Ashley Baldry <arbaldry91@gmail.com>

Repository CRAN

Date/Publication 2023-02-02 19:20:02 UTC
addVideoLanguages

Description
Enabling languages (other than English) to appear as tooltips and other buttons in video.js widgets.

Usage
addVideoLanguages(video, languages)

availableLanguages()

Arguments
video
A video
languages
A character vector of languages to support in the video. See availableVideoLanguages() for a full list

Details
If any languages are missing, you can add a separate script in the head of the application that will apply the language to all videos. See https://videojs.com/guides/languages/ for more details

Value
An updated video with extra language support

Examples
video <- video("https://vjs.zencdn.net/v/oceans.mp4")
video <- addVideoLanguages(video, c("es", "fr", "de"))

if (interactive()) {
  library(shiny)

  ui <- fluidPage(lang = "fr", video)
**guessVideoFormat**

server <- function(input, output) {
  shinyApp(ui, server)
}

### Description

If no type is provided when generating a video.js video, then the format needs to be guessed. Included in the package is a dataset of the default type of each video. This will give the default type of each file provided.

### Usage

```
guessVideoFormat(files)
```

### Arguments

- **files**  
  A vector of URL paths (relative or absolute) to videos

### Value

A vector the same length as files of the video types.

### Examples

```
guessVideoFormat("video.mp4")
```

---

**includeTextTracks**

### Description

video.js contains the ability to include tracks with the video, including subtitles, captions and descriptions. includeTextTracks will make sure that they are included on load, and find the defaults to embed with the video.
Usage

includeTextTracks(
  video,
  files,
  language = "en",
  label = "English",
  kind = "subtitles",
  default = FALSE
)

Arguments

video A video()
files A vector of WebVTT files that contain "cues" of when text should appear, hide and what text to display
language The valid BCP 47 code for the language of the text track, e.g. "en" for English or "es" for Spanish.
label Short descriptive text for the track that will used in the user interface. For example, in a menu for selecting a captions language.
kind An optional vector to match the type of text tracks in files:
  subtitles (default): Translations of the dialogue in the video for when audio is available but not understood. Subtitles are shown over the video.
  captions Transcription of the dialogue, sound effects, musical cues, and other audio information for viewer who are deaf/hard of hearing, or the video is muted. Captions are also shown over the video.
  chapters Chapter titles that are used to create navigation within the video. Typically, these are in the form of a list of chapters that the viewer can use to navigate the video.
  descriptions Text descriptions of the action in the content for when the video portion isn’t available or because the viewer is blind or not using a screen. Descriptions are read by a screen reader or turned into a separate audio track.
  metadata Tracks that have data meant for JavaScript to parse and do something with. These aren’t shown to the user.

default The boolean default attribute can be used to indicate that a track’s mode should start as "showing". Otherwise, the viewer would need to select their language from a captions or subtitles menu.

Details

All vectors must either be the same length as files or of length 1. In the latter, they will be applied to all files supplied.

Value

An updated video with text tracks included
runVideoExample

Examples

```r
vid <- video("https://vjs.zencdn.net/v/oceans.mp4")
includeTextTracks(vid, "url/to/subtitles.vtt")
```

Description

Run {video} Example Applications

Usage

```r
runVideoExample(example = "basic", display.mode = "showcase", ...)

availableVideoExamples()
```

Arguments

- **example** Name of the example to load. Current examples include:
  - basic Basic example of video in use
  - full Basic example of using all buttons available in video
  - server Example showing server-side functionality

- **display.mode** The mode in which to display the application. By default set to "showcase" to show code behind the example.

- **...** Optional arguments to send to shiny::runApp

Value

This function does not return a value; interrupt R to stop the application (usually by pressing Ctrl+C or Esc).

Examples

```r
availableVideoExamples()

if (interactive()) {
  library(shiny)
  library(video)

  runVideoExample("server")
}
```
Description

A video player that can be embedded in HTML pages.

Usage

```r
video(
  files,
  format = NULL,
  options = list(),
  seek_ping_rate = 1000,
  width = NULL,
  height = NULL,
  elementId = NULL
)
```

Arguments

- **files** A vector of file paths or URLs pointing
- **format** An optional list of formats of video
- **options** A named list of options to apply to the video. List of available options available in Details
- **seek_ping_rate** Number of milliseconds between each update of ‘input{id}_seek’ while playing. Default is set to 1000. If set to 0, then ‘input{id}_seek’ will not exist.
- **width, height** Must be a valid CSS unit (like ‘100%’, ‘400px’, ‘auto’) or a number, which will be coerced to a string and have ‘px’ appended. Use NA for it to use the original video width/height.
- **elementId** HTML id tag to be given to the video player element

Details

Here are some more common options to implement:

- **autoplay** Whether or not the video will autoplay on load. NOTE: There is not a guarantee autoplay will work in the browser.
  - FALSE Default: Video won’t autoplay
  - TRUE Video will use browser’s autoplay
  - "muted" Will mute the video and then manually call `play()` on `loadstart()`. Likely to work on browsers
  - "play" Will call `play()` on `loadstart()`, similar to browser autoplay

- **controls** Determines whether or not the player has controls that the user can interact with. By default `video` will include controls even if not specified in the options.
poster  A URL to an image that displays before the video begins playing. This is often a frame of
the video or a custom title screen.

For a full list of available options check out https://videojs.com/guides/options/

Value

A shiny.tag containing all of the required options for a videojs JS object to be initialised in a shiny
application.

On the server side there will be up to four additional objects available as inputs:

{id}_playing  A logical value as to whether or not the video is playing audio
{id}_seek (If seek_ping_rate > 0) the current time of the track loaded
{id}_duration  The duration of the track loaded

Examples

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    title = "howler.js Player",
    video("https://vjs.zencdn.net/v/oceans.mp4")
  )

  server <- function(input, output) {
  }

  runShiny(ui, server)
}

-------------------------------------
video-server  Update video.js Server-Side

Description

Change the state of the video player from the server.

playVideo, pauseVideo and stopVideo will all be applied to the current video.
changeVideo will update the track to the URL or file specified.
updatePlaybackRate will change how fast the video is playing.
Usage

playVideo(id, session = getDefaultReactiveDomain())

pauseVideo(id, session = getDefaultReactiveDomain())

stopVideo(id, session = getDefaultReactiveDomain())

seekVideo(id, seek, session = getDefaultReactiveDomain())

changeVideo(id, files, format = NULL, session = getDefaultReactiveDomain())

updatePlaybackRate(id, playrate = 1, session = getDefaultReactiveDomain())

Arguments

id ID of the video to update
session Shiny session
seek Time (in seconds) to set the position of the track
files A vector of file paths or URLs pointing
format An optional list of formats of video
playrate Speed of playback of the video. Default is set to 1 (normal speed)

Value

Updates the the state of the specified video in the shiny application.

Examples

if (interactive()) {
  library(shiny)

  ui <- fluidPage(
    title = "howler.js Player",
    video("https://vjs.zencdn.net/v/oceans.mp4",
      elementId = "video"
    ),
    actionButton("pause", "Pause Video")
  )

  server <- function(input, output) {
    observeEvent(input$pause, pauseVideo("video"))
  }

  runShiny(ui, server)
}
video-shiny

Shiny bindings for video

Description

Output and render functions for using video within Shiny applications and interactive Rmd documents.

Usage

```r
videoOutput(outputId, width = "100\%", height = "400px")
```

```r
renderVideo(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

- `outputId` output variable to read from
- `width`, `height` Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
- `expr` An expression that generates a video
- `env` The environment in which to evaluate `expr`.
- `quoted` Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable.

Value

An output or render function that enables the use of the widget within Shiny applications.
Index

addVideoLanguages, 2
availableLanguages (addVideoLanguages), 2
availableVideoExamples
  (runVideoExample), 5
changeVideo (video-server), 7
guessVideoFormat, 3
includeTextTracks, 3
pauseVideo (video-server), 7
playVideo (video-server), 7
renderVideo (video-shiny), 9
runVideoExample, 5
seekVideo (video-server), 7
stopVideo (video-server), 7
updatePlaybackRate (video-server), 7
video, 2, 4, 6
video-server, 7
video-shiny, 9
videoOutput (video-shiny), 9