Methods

Example method use:

```
player.play(); // Start playback
player.fullscreen.enter(); // Enter fullscreen
```

Method	Parameters	Description
`play()` [1]	-	Start playback.
`pause()`	-	Pause playback.
`togglePlay(toggle)` [1]	Boolean	Toggle playback, if no parameters are passed, it will toggle based on current status.
`stop()`	-	Stop playback and reset to start.
`restart()`	-	Restart playback.
`rewind(seekTime)`	Number	Rewind playback by the specified seek time. If no parameter is passed, the default seek time will be used.
`forward(seekTime)`	Number	Fast forward by the specified seek time. If no parameter is passed, the default seek time will be used.
`increaseVolume(step)`	Number	Increase volume by the specified step. If no parameter is passed, the default step will be used.
`decreaseVolume(step)`	Number	Increase volume by the specified step. If

Method	Parameters	Description
		no parameter is passed, the default step will be used.
`toggleCaptions(toggle)`	Boolean	Toggle captions display. If no parameter is passed, it will toggle based on current status.
`fullscreen.enter()`	-	Enter fullscreen. If fullscreen is not supported, a fallback "full window/viewport" is used instead.
`fullscreen.exit()`	-	Exit fullscreen.
`fullscreen.toggle()`	-	Toggle fullscreen.
`airplay()`	-	Trigger the airplay dialog on supported devices.
`toggleControls(toggle)`	Boolean	Toggle the controls (video only). Takes optional truthy value to force it on/off.
`on(event, function)`	String, Function	Add an event listener for the specified event.
`once(event, function)`	String, Function	Add an event listener for the specified event once.
`off(event, function)`	String, Function	Remove an event listener for the specified event.
`supports(type)`	String	Check support for a mime type.
`destroy()`	-	Destroy the instance and garbage collect any elements.

1. For HTML5 players, `play()` will return a `Promise` for most browsers - e.g. Chrome, Firefox, Opera, Safari and Edge according to MDN at time of writing.

Getters and Setters

Example setters:

```
player.volume = 0.5; // Sets volume at 50%
player.currentTime = 10; // Seeks to 10 seconds
```

Example getters:

```
player.volume; // 0.5;
player.currentTime; // 10
player.fullscreen.active; // false;
```

Property	Getter	Setter	Description
`isHTML5`	✓	-	Returns a boolean indicating if the current player is HTML5.
`isEmbed`	✓	-	Returns a boolean indicating if the current player is an embedded player.
`playing`	✓	-	Returns a boolean indicating if the current player is playing.
`paused`	✓	-	Returns a boolean indicating if the current player is paused.
`stopped`	✓	-	Returns a boolean indicating if the current player is stopped.

Property	Getter	Setter	Description
`ended`	✓	-	Returns a boolean indicating if the current player has finished playback.
`buffered`	✓	-	Returns a float between 0 and 1 indicating how much of the media is buffered
`currentTime`	✓	✓	Gets or sets the currentTime for the player. The setter accepts a float in seconds.
`seeking`	✓	-	Returns a boolean indicating if the current player is seeking.
`duration`	✓	-	Returns the duration for the current media.
`volume`	✓	✓	Gets or sets the volume for the player. The setter accepts a float between 0 and 1.
`muted`	✓	✓	Gets or sets the muted state of the player. The setter accepts a boolean.
`hasAudio`	✓	-	Returns a boolean indicating if the current media has an audio track.
`speed`	✓	✓	Gets or sets the speed for the player. The setter accepts a value in the options specified in your config. Generally the minimum should be 0.5.
`quality`[1]	✓	✓	Gets or sets the quality for the player. The setter accepts a value from the options

Property	Getter	Setter	Description
			specified in your config.
`loop`	✓	✓	Gets or sets the current loop state of the player. The setter accepts a boolean.
`source`	✓	✓	Gets or sets the current source for the player. The setter accepts an object. See source setter below for examples.
`poster`	✓	✓	Gets or sets the current poster image for the player. The setter accepts a string; the URL for the updated poster image.
`autoplay`	✓	✓	Gets or sets the autoplay state of the player. The setter accepts a boolean.
`currentTrack`	✓	✓	Gets or sets the caption track by index. `-1` means the track is missing or captions is not active
`language`	✓	✓	Gets or sets the preferred captions language for the player. The setter accepts an ISO two-letter language code. Support for the languages is dependent on the captions you include. If your captions don't have any language data, or if you have multiple tracks with the same language, you may want to use `currentTrack` instead.
`fullscreen.active`	✓	-	Returns a boolean indicating if the current player is in fullscreen mode.

Property	Getter	Setter	Description
`fullscreen.enabled`	✓	-	Returns a boolean indicating if the current player has fullscreen enabled.
`pip` [1]	✓	✓	Gets or sets the picture-in-picture state of the player. The setter accepts a boolean. This currently only supported on Safari 10+ (on MacOS Sierra+ and iOS 10+) and Chrome 70+.
`ratio`	✓	✓	Gets or sets the video aspect ratio. The setter accepts a string in the same format as the `ratio` option.
`download`	✓	✓	Gets or sets the URL for the download button. The setter accepts a string containing a valid absolute URL.

1. HTML5 only

The `.source` setter

This allows changing the player source and type on the fly.

Video example:

```
player.source = {
  type: 'video',
  title: 'Example title',
  sources: [
    {
      src: '/path/to/movie.mp4',
      type: 'video/mp4',
      size: 720,
```

```
    },
    {
      src: '/path/to/movie.webm',
      type: 'video/webm',
      size: 1080,
    },
  ],
  poster: '/path/to/poster.jpg',
  previewThumbnails: {
    src: '/path/to/thumbnails.vtt',
  },
  tracks: [
    {
      kind: 'captions',
      label: 'English',
      srclang: 'en',
      src: '/path/to/captions.en.vtt',
      default: true,
    },
    {
      kind: 'captions',
      label: 'French',
      srclang: 'fr',
      src: '/path/to/captions.fr.vtt',
    },
  ],
};
```

Audio example:

```
player.source = {
  type: 'audio',
  title: 'Example title',
  sources: [
    {
      src: '/path/to/audio.mp3',
      type: 'audio/mp3',
```

```
    },
    {
      src: '/path/to/audio.ogg',
      type: 'audio/ogg',
    },
  ],
};
```

YouTube example:

```
player.source = {
  type: 'video',
  sources: [
    {
      src: 'bTqVqk7FSmY',
      provider: 'youtube',
    },
  ],
};
```

Vimeo example

```
player.source = {
  type: 'video',
  sources: [
    {
      src: '143418951',
      provider: 'vimeo',
    },
  ],
};
```

Note: `src` property for YouTube and Vimeo can either be the video ID or the whole URL.

Property	Type	Description
`type`	String	Either `video` or `audio`. *Note:* YouTube and Vimeo are currently not supported as audio sources.
`title`	String	*Optional*. Title of the new media. Used for the `aria-label` attribute on the play button, and outer container. YouTube and Vimeo are populated automatically.
`sources`	Array	This is an array of sources. For HTML5 media, the properties of this object are mapped directly to HTML attributes so more can be added to the object if required.
`poster` [1]	String	The URL for the poster image (HTML5 video only).
`tracks` [1]	String	An array of track objects. Each element in the array is mapped directly to a track element and any keys mapped directly to HTML attributes so as in the example above, it will render as `<track kind="captions" label="English" srclang="en" src="https://cdn.selz.com/plyr/1.0 /example_captions_en.vtt" default>` and similar for the French version. Booleans are converted to HTML5 value-less attributes.
`previewThumbnails` [1]	Object	The same object like in the `previewThumbnails` constructor option. This means you can either change the thumbnails vtt via the `src` key or disable the thumbnails plugin for the next video by passing `{ enabled: false }`.

1. HTML5 only

Events

You can listen for events on the target element you setup Plyr on (see example under the table). Some events only apply to HTML5 audio and video. Using your reference to the instance, you can use the `on()` API method or `addEventListener()`. Access to the API can be obtained this way through the `event.detail.plyr` property. Here's an example:

```
player.on('ready', event => {
  const instance = event.detail.plyr;
});
```