## EXPLORING THE API ATTRIBUTES

It's important for you to know about the attributes that each multimedia element can have as part of the JavaScript API. Each attribute along with its description is listed in **Table 5.1**.

Some attributes are specific to video only; these are listed in **Table 5.2**.

For those attributes that raise an event when their value is changed, the event name is given for easy reference. See the section "Harnessing the API Events" later in this chapter for a list of events.

**TABLE 5.1** Audio and Video Attributes

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **duration** | Contains the duration of the audio or video. This is a read-only attribute and may only be available if the video has been preloaded.<br>When this value becomes available for reading, the `durationchange` event is raised. |
| **volume** | Contains the volume setting for the audio or video.<br>0 is the lowest value; 1.0 is the highest.<br>When this value is changed, the `volumechanged` event is raised. |
| **paused** | Is a boolean attribute that indicates whether or not the audio or video resource is currently paused.<br>When this value changes to true, a `pause` event is raised.<br>When this value changes to false, a `play` event is raised. |
| **playbackRate** | Indicates the required speed at which the audio or video resource is to be played. Valid values range from –1.0 to 1.0.<br>A negative value plays the resource backwards faster; a positive value plays it forward faster.<br>A value of 0 stops playback.<br>When this value changes, a `ratechange` event is raised. |
| **ended** | Is a boolean attribute that indicates whether the audio or video resource has finished playing (and the play direction is forward! See `defaultPlaybackRate` later in this table and `playbackRate` earlier). |
| **autoplay** | Is a boolean attribute that indicates whether or not the audio or video is to play automatically. |

**TABLE 5.1** Audio and Video Attributes *(continued)*

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **loop** | Is a boolean attribute that indicates whether or not the audio or video resource is set to loop. |
| **controls** | Is a boolean attribute that indicates whether or not the browser is to display its default control set on the audio or video. |
| **muted** | Is a boolean attribute that indicates whether or not the audio or video is muted. |
| **src** | Contains the URL to the media source as set in the attribute of the `audio` or `video` element. |
| **currentSrc** | Initially empty, this contains the URL to the media source that's actually selected for play by the `audio` or `video` element, usually set via a child `source` element. |
| **startTime** | Contains the start-time value of the `audio` or `video` element. This is usually 0 but can occasionally be a positive value for various reasons, such as being part of a live stream or if it's a fragment from a larger audio or video resource. |
| **crossOrigin** | Contains the setting for the `audio` or `video` element's `crossOrigin` attribute. This is intended to allow or prohibit cross-origin media of the audio source using the CORS (Cross-Origin Resource Sharing) specification (see Chapter 2 for more details). |
| **networkState** | Contains the current network state of the `audio` or `video` element.<br>It can have one of four settings (the numeric value is provided in brackets after the setting name):<br>• **NETWORK_EMPTY (0)** Indicates that the element has not yet been initialised. When this value is set, the `emptied` event is raised. The `abort` and `error` events can also be raised when this value is set.<br>• **NETWORK_IDLE (1)** A resource to play has been selected but is not currently being played. When this value is set, the `suspend` event is raised. The `abort` and `error` events can also be raised.<br>• **NETWORK_LOADING (2)** The browser is currently attempting to download data. When this value is set, the following events are raised: `loadstart`, `progress`, and `stalled`.<br>• **NETWORK_NO_SOURCE (3)** The element is currently looking for a resource to play but hasn't yet found one. |

**TABLE 5.1** Audio and Video Attributes *(continued)*

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **preload** | Contains the value of the element's `preload` attribute, which is used to provide a hint to the browser on how the media source is to be preloaded (see Chapter 2 for more details). |
| **buffered** | Contains the time range of the `audio` or `video` element that the browser has currently buffered (if any).<br>This returns a `TimeRange` object. |
| **readyState** | Contains the current ready state of the `audio` or `video` element and can contain one of five possible values (the numeric value is provided in brackets after the value name):<br>• **HAVE_NOTHING (0)** — There is currently no information on the element available.<br>• **HAVE_METADATA (1)** — The metadata of the element in question is currently available. In the case of the `video` element, it also indicates that the video's dimensions are available. The resource's start position is currently unknown.<br>When this value is set, the `loadedmetadata` event is raised.<br>• **HAVE_CURRENT_DATA (2)** — Enough data has been retrieved to know the start position of the audio or video but not enough to actually play it.<br>When this value is set, the `loadeddata` event is raised.<br>• **HAVE_FUTURE_DATA (3)** — Enough data has been retrieved so that the resource can be played. Playing the resource now, however, might mean that playback may attempt to overtake the amount of the resource retrieved.<br>When this value is set, the `canplay` event is raised.<br>• **HAVE_ENOUGH_DATA (4)** — Enough of the data has been retrieved that the resource can be played without fear of playback reaching the end of the available data before the end of the resource has been loaded.<br>When this value is set, the `canplaythrough` event is raised.<br>You could check this attribute periodically to see when to start/stop displaying a waiting icon for example. |
| **seeking** | Is a boolean attribute that indicates whether the browser is currently looking for a different playback position within the audio or video resource. For example, the user may have dragged the default controls forward to play at a later point in the resource. |

**TABLE 5.1** Audio and Video Attributes *(continued)*

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **currentTime** | Contains the current time in seconds that the playback position within the audio or video resource has reached.<br>When this value changes, the `timeupdate` event is raised.<br>When this value equals the end of the media resource, the `ended` event is raised. |
| **initialTime** | Contains the initial playback position of the audio or video resource. |
| **startOffsetTime** | Contains a `Date` object representing the current timeline offset (the explicit date and time corresponding to the zero time in the resource) within the resource. |
| **defaultPlaybackRate** | Indicates the required speed at which the audio or video resource is to be played when it starts. Valid values range from -1.0 to 1.0.<br>A negative value plays the resource backwards faster; a positive value plays it forward faster.<br>A value of 0 stops playback. Normal play rate is 0.1.<br>When this value changes, a `ratechange` event is raised. |
| **played** | Returns a `TimeRanges` object indicating the time ranges, if any, that the browser has so far played of the resource. |
| **seekable** | Returns a `TimeRanges` object that indicates the time ranges, if any, of the media resource that the browser can "seek" to (i.e., look ahead), which depends on how much of the resource is actually loaded. |
| **mediaGroup** | Contains the resource's `mediaGroup` attribute value, which allows multiple media elements to be linked together via a common name/group (see Chapter 2 for more details), if set. |
| **controller** | Contains the resource's current media controller, if set; otherwise `null`. |
| **defaultMuted** | Is a boolean that reflects the value of the resource's `muted` attribute (see Chapter 2). |
| **audioTracks** | Contains the live audio track list that is available in the media element's resource, if any. |
| **videoTracks** | Contains the live video track list that is available in the media element's resource, if any. |
| **textTracks** | Is a read-only attribute that contains a list of the text tracks available in the media element's resource, if any. |

## THE **TIMERANGE** OBJECT

Some of the attributes mentioned in Table 5.1 return a `TimeRange` object, which essentially contains the following bits of data:

- **length.** The number of ranges actually in the object.
- **start(index).** The timestamp of the start time of the range for the given index.
- **end(index).** The timestamp of the end time of the range for the given index.

Thus, if a particular `TimeRange` object's length is 1, you can access the start and end timestamps of that particular object via `start(0)` and `end(0)`, respectively.

**TABLE 5.2** Video Attributes

| ATTRIBUTE | DESCRIPTION |
|---|---|
| **width** | Contains the width of the video element in pixels or percentage.<br>Altering this value will not affect the value of `videoWidth`. |
| **height** | Contains the height of the video element in pixels or percentage.<br>Altering this value will not affect the value of `videoHeight`. |
| **videoWidth** | Is a read-only attribute that contains the actual width of the video media file in pixels, or 0 if it's not known, at the time the video is loaded. |
| **videoHeight** | Is a read-only attribute that contains the actual height of the video media file in pixels, or 0 if it's not known, at the time the video is loaded. |
| **poster** | Contains the URL to the poster image for the video, if any. |

Phew, that's quite a list of attributes! But don't worry; many of them you might not ever use. I've made an attempt at putting them in order of those you might be most interested in.

Let's look at a couple of quick examples that show how you might use these attributes. If, for example, you wanted to grab the current played time position of the video, you'd use the `currentTime` attribute:

```
<script>
    var video = document.getElementsByTagName("video")[0];
    alert(video.currentTime);
</script>
```

You grab a handle to the video object using the standard `getElementsByTagName()` JavaScript function (there's only one video here, and you know that, hence the index of 0), and then read the video's `currentTime` attribute. This could of course be 0 if the video hasn't started yet and is constantly updated as the video plays.

Another example would be to check if the video is looping, and if so, stop it from doing so:

```
<script>
    var video = document.getElementsByTagName("video")[0];
    if (video.loop) video.loop = false;
</script>
```

As you can see, you can achieve this by checking if the `loop` attribute is true, and if so, setting it to false to stop the video from looping.

These are just two brief examples of how you can use the attributes. You'll see plenty more examples when you start putting together your own media controls later in this chapter.

Tables 5.1 and 5.2 contain the entire list of attributes just in case you want to try one or more in the future and are curious as to what does what and where. Also, you might have noticed that changing some of these attributes raises events that you can catch, allowing you to act on them.

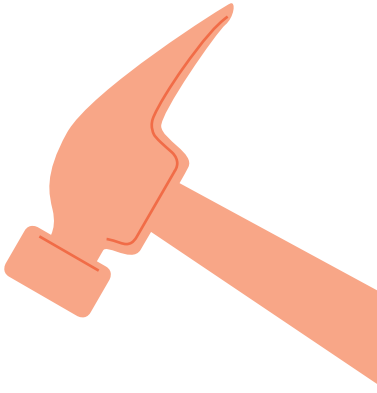
Let's take a closer look at these events.

## HARNESSING THE API EVENTS

Because JavaScript adds interactivity to web documents, when a user performs an action, there needs to be a way of detecting that the action occurred. This is the role of events.

A number of events in the Media JavaScript API can be raised based on value changes, method calls, and browser actions. Listening to certain events and responding to them can be key when rolling out your own control set for audio and video resources.

A full list of the events that can be raised is provided in **Table 5.3**.



**TIP:** If JavaScript events are new to you, check out www.quirksmode.org/js/introevents.html for a good solid introduction.

**TABLE 5.3** Events Raised in the Media JavaScript API

| EVENT | DESCRIPTION |
|---|---|
| **loadstart** | Is raised when the browser starts looking for media data to load. |
| **progress** | Is raised when the browser is retrieving media data. |
| **suspend** | Is raised when the browser was fetching media data but paused and has not yet fetched the entire media resource. |
| **abort** | Is raised when the browser stops fetching media data but not because of an error. |
| **error** | Is raised when an error occurred while the browser was fetching the media data. |
| **emptied** | Is raised when the network connection was lost while the browser was fetching media data or the `load()` method was called when one such method was already in progress. |
| **stalled** | Is raised when the browser is attempting to fetch media data, but for some reason the data isn't being transferred. |
| **loadedmetadata** | Is raised when the duration and dimensions of the media resource have been acquired by the browser. |
| **loadeddata** | Is raised when the browser knows the start position of the media resource. |
| **canplay** | Is raised when the browser can start playback for the first time but can't guarantee that if playback begins now that it won't have to pause to fetch more media data. |

**TABLE 5.3** Events Raised in the Media JavaScript API *(continued)*

| EVENT | DESCRIPTION |
|---|---|
| **canplaythrough** | Is raised when the browser is capable of playing the media resource from start to finish without having to pause to fetch more media data. |
| **playing** | Is raised when playback of a media resource is ready to start after having been previously paused or delayed due to lack of sufficient media data. |
| **waiting** | Is raised when the browser has stopped playback due to insufficient media data being available. It does, however, expect the required data to become available shortly. |
| **seeking** | Is raised when the media resource's `seeking` attribute is set to true. |
| **seeked** | Is raised when the media resource's `seeking` attribute is set to false. |
| **ended** | Is raised when the media resource stops playing because it has finished. |
| **durationchange** | Is raised when the media resource's `duration` attribute has been updated. |
| **timeupdate** | Is raised when the media resource's current playback position has been changed (e.g., as part of normal playback). |
| **play** | Is raised when the media resource that was previously paused no longer is paused, and normal playback has resumed. |
| **pause** | Is raised after the `pause()` method has returned and the media resource has been paused. |
| **ratechange** | Is raised when either the media resource's `defaultPlaybackRate` or `playbackRate` attributes are changed. |
| **volumechange** | Is raised when the media resource's `volume` or `muted` attribute has changed. |