



HTML5 VIDEO TAG

What you need to know about HTML5 video tag behavior and compatibility

WHITEPAPER



CONTENTS

- 3** Introduction
- 4** Background
- 5** Video Tag Event Differences
- 5** Video Tag Attribute Differences
- 5** Device Behavior Differences
- 6** Playback In-Page vs. Full Screen
- 6** Bottom Line
- 7** Introducing Next-Generation Smart Players
- 8** Conclusion

WHAT YOU NEED TO KNOW ABOUT HTML5 VIDEO TAG BEHAVIOR AND COMPATIBILITY

This document presents the findings from a wide-ranging survey of HTML5 video tag usage and behavior across devices.



The study was part of a Brightcove initiative to develop a next-generation Video Cloud Smart Player, which resulted in a significant advancement in video player architecture. Today, the new Video Cloud Smart Player provides a unique “abstraction layer” that helps ensure consistent and reliable HTML5 video performance regardless of device, browser or operating system. The result is a more rapid development cycle for cross-platform video player innovations and more stable technology to support business-critical services for advertising, analytics, e-commerce and more.

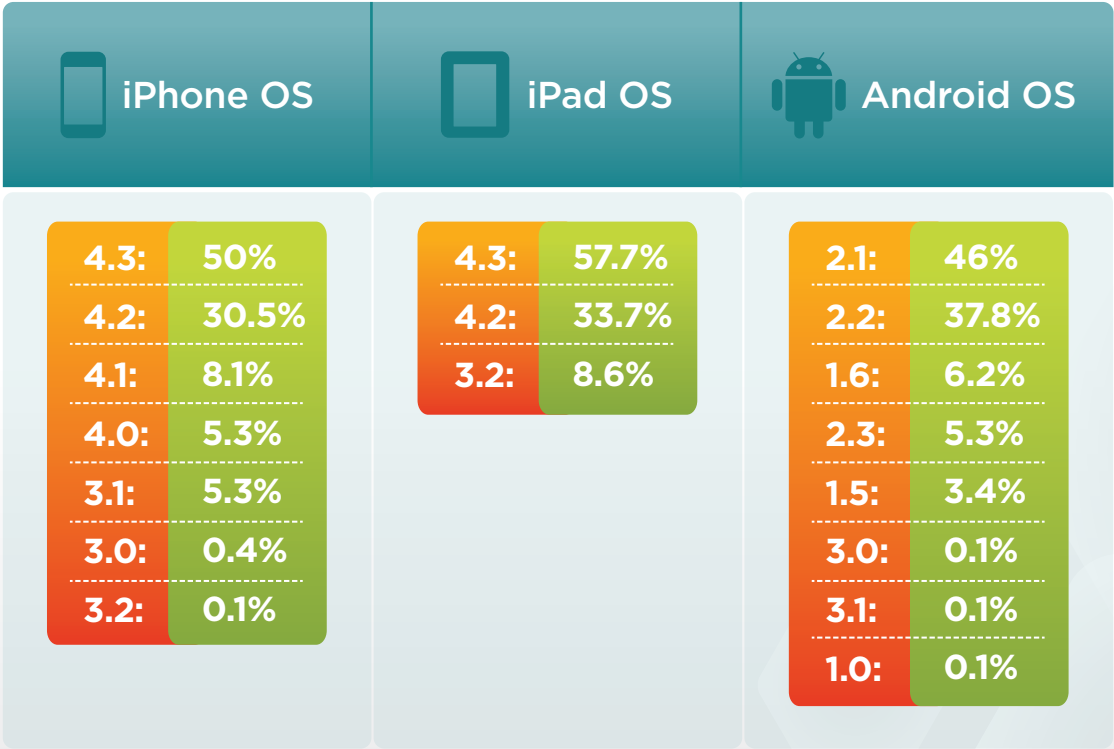
HTML5 video tag behavior

Background

HTML5 is an emerging standard for presenting content on the Web, including video. When HTML5 video burst on the scene in 2010 with the introduction of the iPad, media publishers scrambled to figure out how to make their video available on the hot new consumer device. It was no longer just a question of publishing video on a website, but rather how to provide consistent video experiences across an increasingly complex landscape of both Flash and HTML5-compatible devices.

Today, HTML5 video is the key to reaching a large part of the fast-growing mobile audience on iOS devices, as well as other connected and mobile devices. Currently, HTML5 mobile video usage across Brightcove's customer base is distributed between the iPad (52%), iPhone (38.2%) and Android devices (9.8%).

These smartphones and tablets access HTML5 video using a variety of different versions of the WebKit open source browser technology, but the way WebKit browsers render HTML5 video varies depending on the version of iOS or Android the device is running. This variation can produce inconsistent playback experiences, as well as differences in advertising and analytics behavior. The table below provides a snapshot of Brightcove Video Cloud platform data, which illustrates that the distribution of operating systems used by devices accessing HTML5 video experiences is becoming more fragmented over time.



HTML5 video tag behavior

Brightcove ran a series of tests to help identify specific differences in HTML5 1) video tag events, 2) video tag attributes, 3) device behavior and 4) playback in-screen vs. full screen. To establish a baseline, we started with a basic HTML page with a video tag. We added event listeners to the tag and captured all of the events that were fired on each mobile platform, when they fired, what order they fired in, and what the attributes were on the video tag at these times. This enabled us to get a feel for how these scenarios differed on each platform and, in turn, how our implementation may need to cater to special cases.

The results indicate that any video player that utilizes the HTML5 video tag must account for major variances in all of these areas to ensure consistent playback and behavior. These differences are described in detail below.

The impact of this fragmentation is that material numbers of the potential audience may not be able to see video playback at all, or may see inexplicable delays and unexpected behaviors that erode their confidence and distract attention away from the content and message. Additionally, valuable advertising opportunities and analytic insights may be sacrificed.

Video Tag Event Differences

One of the most difficult challenges of developing HTML5 video players is event inconsistency from the video tag. Even basic events like play/pause fire at different times and for different reasons across each device platform, including different versions of iOS and Android operating systems. The goal of developing from the HTML5 working specification event model and assuming things “just work” is sorely optimistic.

Attempts to account for these differences within a global architecture are problematic because of the need to handle multiple conditions within common functions dealing with the video playback for each device/OS/browser.

Tests also revealed that a single class for handling all events for the video tag across devices and OS versions does not suffice. The video tag needs to be wrapped individually to account for the differences and to normalize the events with limited consistent information obtained for that particular device/OS.

Video Tag Attribute Differences

In addition to devices firing different events at different times, we also examined the properties of the video object itself when these events fired, which produced similarly diverse results. There were a number of video attributes that were unreliable depending on what device was in use.

For instance, when the video tag is initially processed, browsers report network states ranging from empty, to idle to loading. When playback starts, browsers report network states of idle, loading and no source and ready states ranging from “have nothing”, “have enough data”, and “have future data.” Interestingly, some browsers report at completion that the video was paused and ended, while others report not paused and not ended.

Others video properties that fluctuate across devices included time, videoWidth, videoHeight, preload, bufferedLength, currentTime, duration, playedLength, seekableLength, paused and ended.

Device Behavior Differences

Aside from the consistency issues with the events firing and their corresponding attribute data, the unique behaviors of the devices also introduce complications.

For instance, some versions of Android tablets always play video files in full screen, similar to iPhone and Android phones. Yet, other Android tablets play in-page like the iPad (see playback in-page vs. full screen section for more on this difference).

The requirements placed on the software, hardware, and even the video files themselves, in order to successfully play back, are also different. Simply having an H.264 video in the tag is not sufficient to ensure you will have success trying to play it. There are restrictions on the specifics of H.264 encoding (both audio and video), as well as the overall size of the files.

Furthermore, the implementation of proprietary formats, such as webM and Apple HLS, require even more specific attention and development to recognize and support as appropriate.

HTML5 video tag behavior

A few other device behaviors are difficult to navigate. For instance, iOS devices do not allow autoplay. This behavior causes issues for both the standalone players (which can autoplay on desktop browsers if configured to do so) and Brightcove's API implementations of `loadVideo()` and `play()`, which must account for this restriction. Android devices not restarting from the same position after existing full screen also require special handling.

Finally, testing revealed numerous inconsistencies on how devices handled the sizing and orientation changes with the video tag. Nevertheless, the most substantial behavior difference observed is playback in-page versus full screen.

Playback In-Page vs. Full Screen

Note that "full screen" within this section refers to the default playback environment on a device, not the full screen mode that can be launched from the browser video controls on tablets which support both in-page and a full screen mode.

Depending on the device, playback of a video through the use of an HTML video tag will occur either within the context of the HTML page or in a full screen mode using the native video player of the device. The former case is most common on the iPad and newer tablets running Android 3.1 or higher. Full screen playback is a case for handheld Android devices and for the iPhone.

There are significant differences in playback in both types of scenarios. The more difficult behavior occurs in the full screen native player on devices. Most obviously, since playback within full screen is controlled completely using the native player on the device, there is no ability to configure graphically the controls used or the playback experience. For example, when paused there is no ability to display any information about the video, as can be done in-page in an overlay over the video. In addition, while in the full screen native player on Android devices, there are no pause or resume events that fire to enable any monitoring or reaction to user activity.

Upon completion of playback of a video, the behavior differs between Android and iOS devices. On iOS, upon video completion the last frame of the video is displayed within the native player, and to return to the

browser requires the user to tap the "DONE" button. On Android, a video completion results in the native video player closing and returning to the browser view.

Although this behavior can be seen as more desirable in single video players, when playing multiple videos in a playlist player with automatic advance, the user is forced back to the browser before the source of the video tag is switched, which re-launches the native player for the new video. It also introduces problems if for any reason the video needs to be switched during playback on Android. In that case there is no way to force the device out of its native player, yet switching the video source while the native player is playing back a video simply results in the video stopping. The user is required to exit the full screen, at which point the new video is displayed and can be played back by the user by once again launching the native player.

Exiting from the full screen mode on handhelds is handled in two ways. For Android, the user must use the device's back button outside of the video player, and often it requires multiple taps. For iPhone users, a "DONE" button returns the user to the browser from the native player. In both cases, there are no events within the browser or video tag that can be listened to in order to determine which view is currently displayed to the user.

Bottom Line

The video tag is an outstanding addition to the HTML5 spec but unfortunately also introduces significant variation in video behavior from device to device, and often based on the specific OS version a device is running. As such, any video player that utilizes the tag must account for these major variances to ensure consistent playback for the end-user across devices, as well as a stable environment for analytics and advertising.

Based on our experience, we believe that few Web development organizations are aware that these differences exist. Even fewer have a detailed understanding of how these differences affect the quality of the end user experience and advertising and analytics behaviors. And still fewer have the skills to develop effective workarounds for these inconsistencies.

HTML5 video tag behavior

Introducing Next-Generation Smart Players

Brightcove has developed a new framework for manipulating the HTML5 video tag that not only helps ensure this consistency, but also allows Brightcove to rapidly deploy new player innovations that work regardless of OS version and without the risk of breaking backwards compatibility with older versions.

We have supported our development efforts with mobile device test automation across all versions to guarantee this consistency and backwards compatibility in order to give us and our customers confidence as we move forward, pushing the boundaries of what an HTML5 video player can do.

The new Video Cloud Smart Player implements a series of innovations that overcome the limitations of the HTML5 video tag to provide a reliable and effective video experience across devices:

Consistency

- Brightcove now provides “video tag wrappers” that deliver a consistent and reliable set of standard events that allow us to rapidly build out new functionality against them.
- The Video Cloud Universal Player API presents a set of software interfaces that are consistent across video experiences implemented in HTML5 and Flash. This enables developers to create highly customized video experiences and plug-ins that will behave the same way across both Flash and HTML5 environments.
- A new HTML5 Abstraction Layer automatically adapts the video experience to account for operating system and browser inconsistencies, allowing reliable playback and execution of advertising and analytics logic across both current and future operating systems and browsers.
- In addition to iOS devices, Video Cloud Smart Players now also include support for the most recent Android and iOS operating system versions to ensure content is being delivered to these devices reliably and consistently.
- Video Cloud’s encoding and ingestion options are also able to accommodate the specific mobile requirements. And, working with strategic partners like Akamai and Limelight enable Brightcove to ensure the requirements for multi-part file delivery for mobile are also met.

Analytics

- HTML5 Analytics provide deep insight into how viewers are engaging with content across every device and playback environment. Now, Video Cloud analytics cover both Flash and HTML5 to give customers a comprehensive view into audience behaviors and device-specific engagement.

Advertising

- Video Cloud Smart Player now supports the VAST 2.0 advertising standard and provides turnkey integration with Google DoubleClick ad serving solutions. This enables Google to serve ads to customers for both HTML5 and Flash, regardless of the device or operating system version a consumer is using to access the content.

Customization

- Brightcove Experience Markup Language (BEML) Support for HTML5 ensures video experiences play appropriately and gracefully handle situations where unsupported custom components cannot successfully render in an HTML5 video player. Customers can also use the Universal Player API and existing Video Cloud Media APIs to create custom experiences and playlists around the player.

HTML5 video tag behavior

Conclusion

The complexity associated with publishing video across HTML5-compatible devices continues to grow, as the distribution of operating systems and browsers used by these devices has become increasingly fragmented. This has also had an adverse effect on the reliability of crucial business services and plug-ins associated with video content, including advertising and analytics. At Brightcove, we have invested considerable effort to hide the complexity of the HTML5 video from our customers and to take these compatibility issues off the table. As a result of the inconsistencies uncovered in this wide-ranging study, Brightcove Video Cloud Smart Players now help our customers ensure their content is accessible across all devices and operating systems.