Delivering Audio and Video Using Web Standards

Part 1

Vicki Murley
Safari Technologies Evangelist


HTML5


iPad
9:41 AM
Wednesday
27
Calendar
Contacts
Notes
280
Maps
Videos
YouTube
iTunes
App Store
Settings
Safari
Mail
7
Photos
iPod

Apple – iPad-ready websites

http://www.apple.com/ipad/ready-for-ipad/

Google

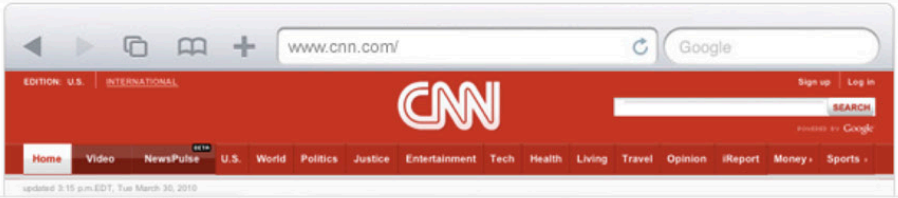
Store | Mac | iPod | iPhone | iPad | iTunes | Support | Search

iPad ready.

iPad features Safari, a mobile web browser that supports the latest web standards — including HTML5, CSS3, and JavaScript. Here are just a few of the sites that take advantage of these web standards to deliver content that looks and functions beautifully on iPad.

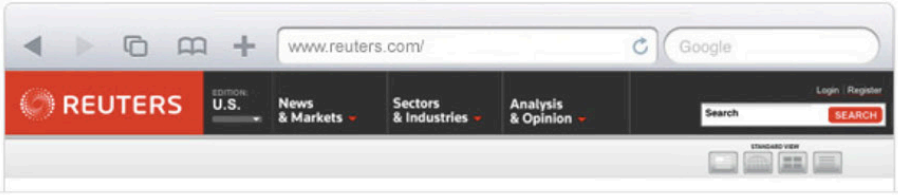
CNN

When you're browsing CNN.com on iPad, the site automatically displays an HTML5 video player, providing you with the best possible viewing experience.



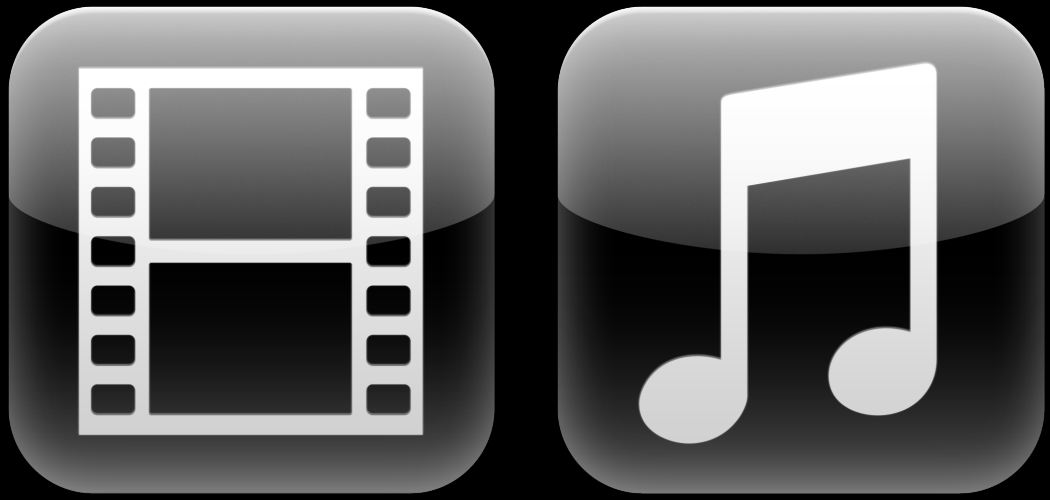
Reuters

An HTML5 video player on Reuters.com lets you view most of the site's video content on iPad.



What You'll Learn

- Adding media to your webpages with HTML5
- Optimizing your media experience with standard web technologies
- Using built-in functions to control media



How can I disable automatic buffering on the desktop?

How can I trigger fullscreen mode?

Can I make my own buttons to play and pause?

What codecs will the video tag play?

Are there platform-specific considerations for iOS?

What can I do to optimize for different screen sizes?

Can I detect whether the browser can play my video?

How do I detect whether a browser supports the video tag?

Adding Media to Webpages with HTML5

Adding Media with HTML5

The basics

- Specifying fallback content
- Writing cross-browser compatible code
- Platform-specific considerations

poster, height, width, videoHeight, videoWidth

```
<video src="myMovie.mp4"></video>
```


HTML5 video support is required to play this media. Go download Safari!
<video src="myMovie.m4v"></video>

Supported Media

- Safari on the desktop
  - Any linear codec supported by QuickTime, including user-installed codecs
- Safari on iOS
  - H.264 video (baseline profile)
  - Uncompressed WAV and AIF audio
  - MP3, AAC-LC and HE-AAC audio
- QuickTime reference movies
- HTTP Streaming

```
<video src="myMovie.m4v">
  This text is only shown if the browser
  doesn't implement the video tag.
  HTML5 video support is required to
  play this media. Go download Safari!
</video>
```

Working With Multiple Media Resources

Using the <source> tag

- User agent iterates through the list
  - Attempts to play first supported encoding
  - Does not continue after a failure

```
<video>
    <source src="safariDemo.mp4">
    <source src="safariDemo.ogg">
    <source src="safariDemo.mov">
</video>
```

Working With Multiple Media Resources

Tips for using the <source> tag

- Use the `type` attribute
  - Avoid unnecessary network requests

```
<video>
    <source src="safariDemo.mp4">type="video/mp4">
    <source src="safariDemo.ogg">type="video/ogg">
    <source src="safariDemo.mov">type="video/quicktime">
</video>
```

Working With Multiple Media Resources

Tips for using the <source> tag

- Use the `type` attribute
  - Avoid unnecessary network requests
  - Include `codecs` for the best results

```
<video>
  <source src='baseline.mp4'
          type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>

  <source src='high.mp4'
          type='video/mp4; codecs="avc1.64001E, mp4a.40.2"'>
</video>
```

Working With Multiple Media Resources

Tips for using the <source> tag

- Use the `media` attribute
  - Choose an optimized resource by querying device characteristics

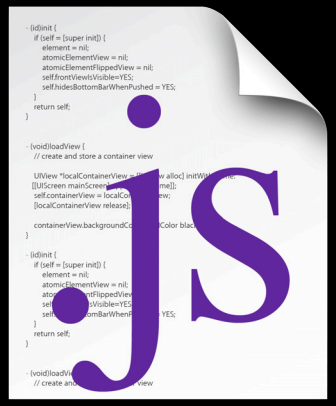
```
<video>
  <source src='small.mp4'
          type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'
          media="screen and (max-device-width:320px)">

  <source src='large.mp4'
          type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'
          media="screen and (min-device-width:321px)">
</video>
```

Dynamic Detection

Using JavaScript to query browser capabilities

- Detecting HTML5 `<video>` support

```
if ('HTMLVideoElement' in window) {
  // your browser supports the HTML5 video element
}
```



Dynamic Detection

Using JavaScript to query browser capabilities

- Detecting HTML5 <video> support
- Detecting playability

```
if ('HTMLVideoElement' in window) {
    var myVideo = document.createElement("video");
    var support = myVideo.canPlayType('video/mp4;
                                       codecs="avc1.42E01E, mp4a.40.2"');

    if (support == "maybe" || support == "probably") {
      // set attributes for myVideo, etc.
      document.getElementsByTagName('body')[0].appendChild('myVideo');
    }
}
```

Dynamic Detection

Using JavaScript to query browser capabilities

- Detecting HTML5 <video> support
- Detecting playability

```
if ('HTMLVideoElement' in window) {
    var myVideo = document.createElement("video");
    var support = myVideo.canPlayType('video/mp4;
                                       codecs="avc1.42E01E, mp4a.40.2"');

    if (support == "maybe" || support == "probably") {
      // set attributes for myVideo, etc.
      document.getElementsByTagName('body')[0].appendChild('myVideo');
    }
}
```

Dynamic Detection

Using JavaScript to query browser capabilities

- Detecting HTML5 <video> support
- Detecting playability

```
if ('HTMLVideoElement' in window) {
   var myVideo = document.createElement("video");
   var support = myVideo.canPlayType('video/mp4;
                                      codecs="avc1.42E01E, mp4a.40.2"');

   if (support == "maybe" || support == "probably") {
     // set attributes for myVideo, etc.
     document.getElementsByTagName('body')[0].appendChild('myVideo');
   }
}
```

Dynamic Detection

Using JavaScript to query browser capabilities

- Detecting HTML5 <video> support
- Detecting playability

```
if ('HTMLVideoElement' in window) {
   var myVideo = document.createElement("video");
   var support = myVideo.canPlayType('video/mp4;
                                      codecs="avc1.42E01E, mp4a.40.2"');

   if (support == "maybe" || support == "probably") {
     // set attributes for myVideo, etc.
     document.getElementsByTagName('body')[0].appendChild('myVideo');
   }
}
```

Dynamic Detection

Using JavaScript to query browser capabilities

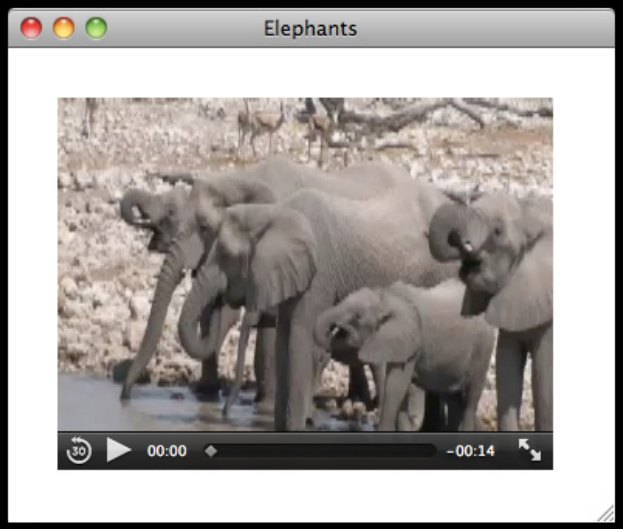
- Detecting HTML5 <video> support
- Detecting playability

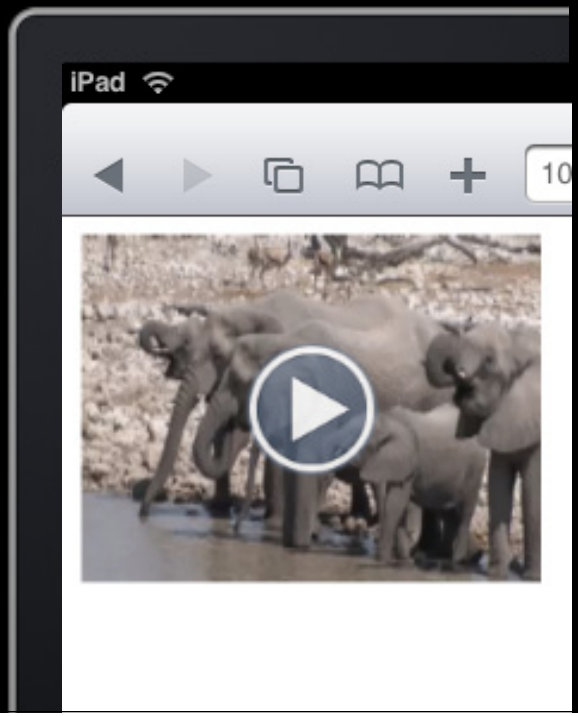
```
if ('HTMLVideoElement' in window) {
   var myVideo = document.createElement("video");
   var support = myVideo.canPlayType('video/mp4;
                                      codecs="avc1.42E01E, mp4a.40.2"');

   if (support == "maybe" || support == "probably") {
     // set attributes for myVideo, etc.
     document.getElementsByTagName('body')[0].appendChild('myVideo');
   }
}
```

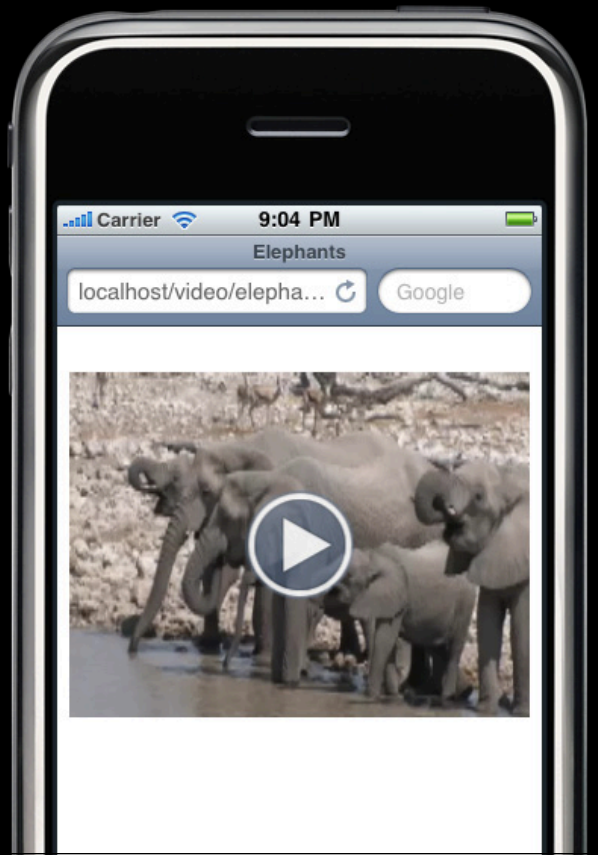
iPhone and iPad use default dimensions

no automatic buffering

```
<video src="myMovie.m4v">height="240" width="320" controls >
  HTML5 video support is required to play this media.
  Go download Safari!
</video>
```



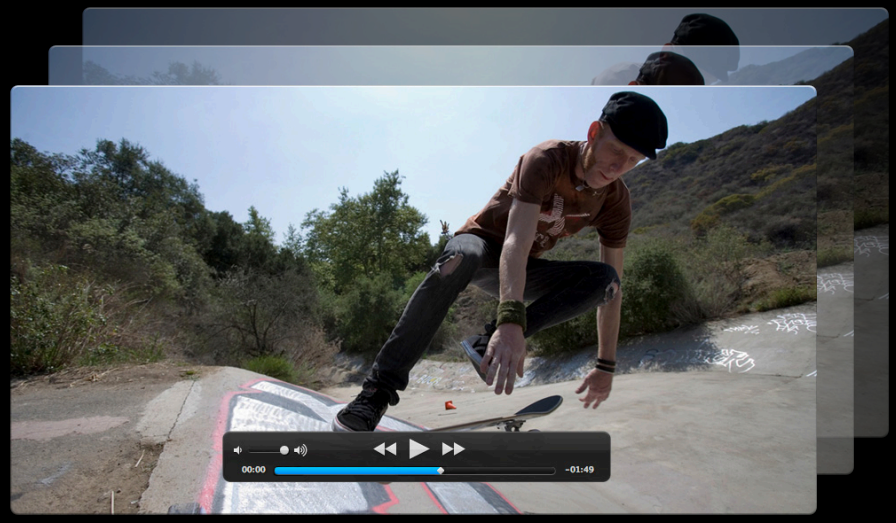




Other Platform-specific Details

Safari on Mac OS X and Windows

- Optimized for concurrent playback from multiple media resources
- Programmatic volume control via `volume` and `muted` properties
- Automatic buffering and playback
  - `autoplay` and inline `play()` are on



Other Platform-specific Details

Safari on iOS

- Optimized for playback from a single media resource
- Volume is under user control
- No automatic buffering or playback
  - Playback requires a a user action
  - `autoplay` and inline `play()` are off

What You'll Learn

- Adding media to your webpages with HTML5
- Optimizing your media experience with standard web technologies
- Using built-in functions to control media

Optimizing with Standard Web Technologies

Attributes and events

`autoplay` or inline `play()`

Examining Event Flow

Default behaviors on iOS and the desktop

Understanding Event Flow

- Critical when designing a custom experience
- Try this example out with different
  - devices
  - media sources
  - software versions

Disabling Automatic Buffering

Using the preload attribute


```
<video id="iPad" controls src="iPad.m4v" preload="none">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

Disabling Automatic Buffering

Setting src on the fly

- All versions of Safari

```
<script>
  function setSource() {
     var myVideo = document.querySelector("video");
     myVideo.src = "iPad.m4v";
     myVideo.play();
  }
</script>
```

```
<video id="iPad" controls src="" onclick="setSource()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

Disabling Automatic Buffering

Setting src on the fly

- All versions of Safari

```
<script>
  function setSource() {
     var myVideo = document.querySelector("video");
     myVideo.src = "iPad.m4v";
     myVideo.play();
  }
</script>
```

```
<video id="iPad" controls src="" onclick="setSource()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `loadedmetadata`
  - set `height` and `width` on the fly

```
<script>
  function setHeightAndWidth() {
    var myVideo = document.querySelector("video");
    myVideo.height = myVideo.videoHeight;
    myVideo.width = myVideo.videoWidth;
  }
</script>
```

```
<video src="myMovie.m4v" controls onloadedmetadata="setHeightAndWidth()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `loadedmetadata`
  - set `height` and `width` on the fly

```
<script>
  function setHeightAndWidth() {
    var myVideo = document.querySelector("video");
    myVideo.height = myVideo.videoHeight;
    myVideo.width = myVideo.videoWidth;
  }
</script>
```

```
<video src="myMovie.m4v" controls onloadedmetadata="setHeightAndWidth()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `loadedmetadata`
  - set `height` and `width` on the fly
  - check to see if fullscreen mode is supported in Safari 5

```
<script>
  function checkFullscreenSupport() {
    if (!myVideo.webkitSupportsFullscreen) {
      document.getElementById('fullscreen').style.display = 'none';    // hide this button
    }
  }
</script>

<button id="fullscreen" onclick="document.getElementById('iPad').webkitEnterFullscreen()">
  Go Fullscreen
</button>
```

```
<video id="iPad" src="iPad.m4v" onloadedmetadata="checkFullscreenSupport()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `loadedmetadata`
  - set `height` and `width` on the fly
  - check to see if fullscreen mode is supported in Safari 5

```
<script>
  function checkFullscreenSupport() {
    if (!myVideo.webkitSupportsFullscreen) {
      document.getElementById('fullscreen').style.display = 'none';    // hide this button
    }
  }
</script>
```

```
<button id="fullscreen" onclick="document.getElementById('iPad').webkitEnterFullscreen()">
  Go Fullscreen
</button>
```

```
<video id="iPad" src="iPad.m4v" onloadedmetadata="checkFullscreenSupport()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `loadedmetadata`
  - set `height` and `width` on the fly
  - check to see if fullscreen mode is supported in Safari 5

```
<script>
  function checkFullscreenSupport() {
    if (!myVideo.webkitSupportsFullscreen) {
      document.getElementById('fullscreen').style.display = 'none';    // hide this button
    }
  }
</script>
```

```
<button id="fullscreen" onclick="document.getElementById('iPad').webkitEnterFullscreen()">
  Go Fullscreen
</button>

<video id="iPad" src="iPad.m4v" onloadedmetadata="checkFullscreenSupport()">
  HTML5 video support is required to play this media. Go download Safari!
</video>
```

HTML5 Media Element Events

When to do what

- `canplay, canplaythrough, waiting`
  - show a progress indicator until media can be played
  - then programmatically `play()`

Demo

Optimizing the user experience with events

Beth Dakin
Safari and WebKit Engineer

HTML5 <video>

Just another webpage element

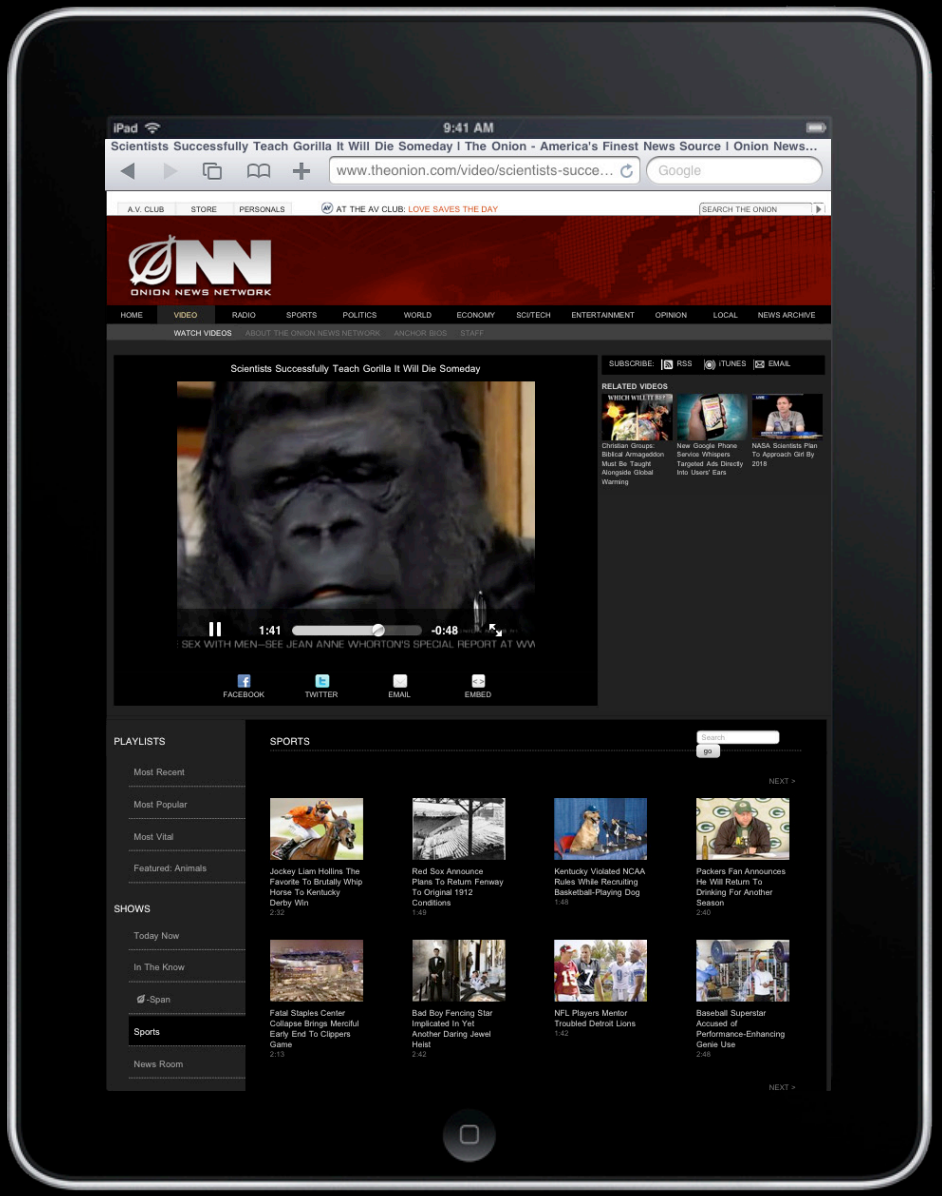
- Layered with other elements
- Styled with CSS
- Manipulated with JavaScript

What You'll Learn

- Adding media to your webpages with HTML5
- Optimizing your media experience with standard web technologies
- Using built-in functions to control media

Using Built-in Functions to Control Media

Creating simple controls


iPad
9:41 AM
Scientists Successfully Teach Gorilla It Will Die Someday ! The Onion - America's Finest News Source ! Onion News...
www.theonion.com/video/scientists-succe...
Google
A.V. CLUB
STORE
PERSONALS
AT THE AV CLUB: LOVE SAVES THE DAY
SEARCH THE ONION
ONION NEWS NETWORK
HOME
VIDEO
RADIO
SPORTS
POLITICS
WORLD
ECONOMY
SCI/TECH
ENTERTAINMENT
OPINION
LOCAL
NEWS ARCHIVE
WATCH VIDEOS
ABOUT THE ONION NEWS NETWORK
ANCHOR BIOS
STAFF
Scientists Successfully Teach Gorilla It Will Die Someday
1:41
-0:48
SEX WITH MEN--SEE JEAN ANNE WHORTON'S SPECIAL REPORT AT WW
FACEBOOK
TWITTER
EMAIL
EMBED
SUBSCRIBE:
RSS
iTUNES
EMAIL
RELATED VIDEOS
Christian Groups: Biblical Armageddon Must Be Taught Alongside Global Warming
New Google Phone Service Whispers Targeted Ads Directly Into Users' Ears
NASA Scientists Plan To Approach Girl By 2015
PLAYLISTS
Most Recent
Most Popular
Most Vital
Featured: Animals
SHOWS
Today Now
In The Know
C-Span
Sports
News Room
SPORTS
Search
go
NEXT >
Jockey Liam Hollins The Favorite To Brutally Whip Horse To Kentucky Derby Win
2:52
Red Sox Announce Plans To Return Fenway To Original 1912 Conditions
1:49
Kentucky Violated NCAA Rules While Recruiting Basketball-Playing Dog
1:48
Packers Fan Announces He Will Return To Drinking For Another Season
2:40
Fatal Staples Center Collapse Brings Merciful Early End To Clippers Game
2:13
Bad Boy Fencing Star Implicated In Yet Another Daring Jewel Heist
2:42
NFL Players Mentor Troubled Detroit Lions
1:42
Baseball Superstar Accused of Performance-Enhancing Genie Use
2:48
NEXT >

Playing and Pausing Programmatically

Creating a simple play/pause button

```
<script>
function playPause() {
    var myVideo = document.querySelector("video");

    if (myVideo.paused)
        myVideo.play();
    else
        myVideo.pause();
}
</script>

<input type=button onclick="playPause()" value="Play/Pause">

<video src="iPad.m4v">
   HTML5 video support is required to play this media. Go download Safari!
</video>
```

Playing and Pausing Programmatically

Creating a simple play/pause button

```
<script>
function playPause() {
    var myVideo = document.querySelector("video");

    if (myVideo.paused)
        myVideo.play();
    else
        myVideo.pause();
}
</script>

<input type=button onclick="playPause()" value="Play/Pause">
```

```
<video src="iPad.m4v">
   HTML5 video support is required to play this media. Go download Safari!
</video>
```

Playing and Pausing Programmatically

Creating a simple play/pause button

```
<script>
function playPause() {
    var myVideo = document.querySelector("video");

    if (myVideo.paused)
        myVideo.play();
    else
        myVideo.pause();
}
</script>
```

```
<input type=button onclick="playPause()" value="Play/Pause">
```

```
<video src="iPad.m4v">
   HTML5 video support is required to play this media. Go download Safari!
</video>
```

Playing and Pausing Programmatically

Creating a simple play/pause button

```
<script>
function playPause() {
    var myVideo = document.querySelector("video");

    if (myVideo.paused)
        myVideo.play();
    else
        myVideo.pause();
}
</script>
```

```
<input type=button onclick="playPause()" value="Play/Pause">

<video src="iPad.m4v">
   HTML5 video support is required to play this media. Go download Safari!
</video>
```

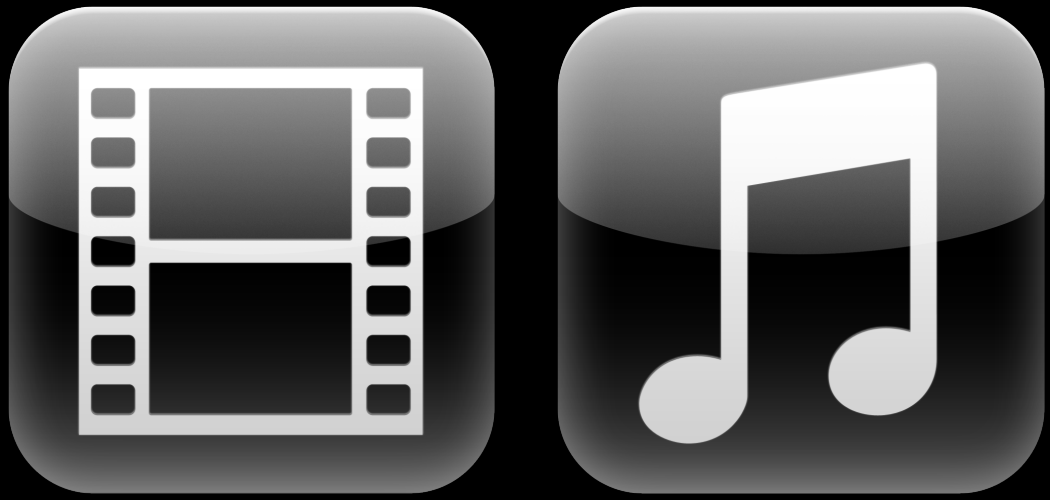
Demo

Adding Play and Pause buttons

Beth Dakin
Safari and WebKit Engineer

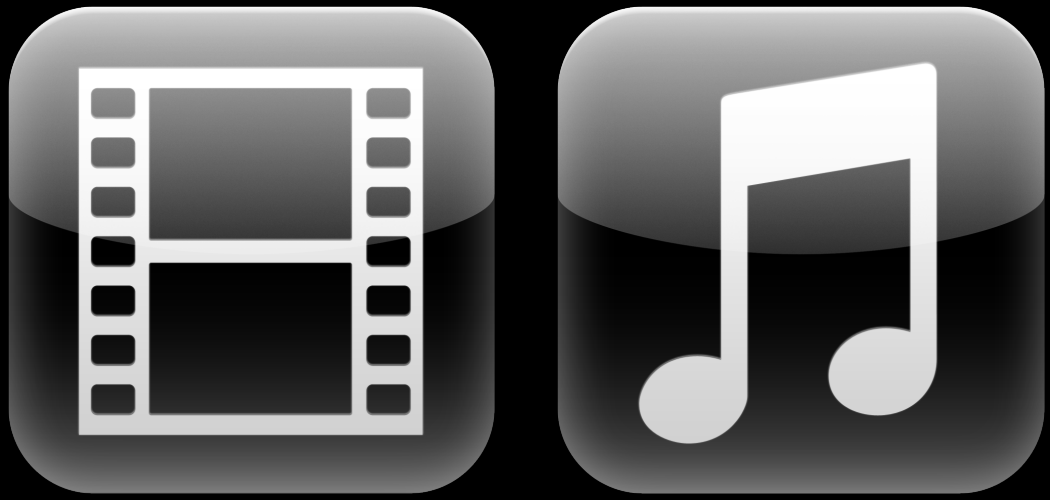
What You'll Learn

- Adding media to your webpages with HTML5
- Optimizing your media delivery experience with standard web technologies
- Using built-in functions to create custom media controls



Summary

- It's easy to add media to your webpages with HTML5
- You can make it as basic or sophisticated as you like
- If you're a web programmer, you're already an expert



More Information

Vicki Murley
Safari Technologies Evangelist
vicki@apple.com

Documentation
Safari Dev Center
http://developer.apple.com/safari

Safari HTML5 Audio and Video Guide
http://developer.apple.com/safari

HTML5: W3C Working Draft
http://www.w3.org/TR/html5/

Apple Developer Forums
http://devforums.apple.com

Related Sessions

Delivering Audio and Video Using Web Standards, Part Two	Marina Tuesday 10:15AM
Advances in HTTP Streaming	Marina Tuesday 11:30AM
CSS Effects Part 1: UI Elements and Navigation	Marina Tuesday 3:15PM
CSS Effects Part 2: Galleries and 3D Effects	Marina Tuesday 4:30PM
Adding Touch and Gesture Detection to Web Pages on iPhone OS	Nob Hill Wednesday 2:00PM

Labs

HTML5 Audio and Video Lab	Internet & Web Lab A Tuesday 2:00-6:30
Safari Open Lab	Internet & Web Lab B Tuesday 2:00-6:30

Q&A


WWDC10