PROGETTAZIONE E PRODUZIONE MULTIMEDIALE

HTML5

Prof. Alberto Del Bimbo

HTML5

- HTML5 introduces a new content model with new tags that have direct relationship with their content. It provides a new set of *semantically-meaningful elements* for describing a web page layout.
- This makes it easier to read and organise code, and search engines and screen readers to read and organise the content of a site.

- HTML5 has several goals:
  - *Support all existing web pages*
  - *Reduce the need for external plugins and scripts to show website content*
  - *Improve the semantic definition of page elements*
  - *Make the rendering of web content universal and independent of the device being used*
  - *Handle web documents errors in a better and more consistent fashion*

- It is still a work in progress. No browsers have full HTML5 support. It will be many years – perhaps not until 2018 - before being fully defined and supported

HTML5

- With HTML5 the new doctype must be used on the first line of every HTML page. The doctype tells the browser which type and version of document to expect. In HTML5 there is only one doctype:

```
<!DOCTYPE html>
   <!-- Everything else goes in here -->
    <!-- Ending with the closing tag: -->
</html>
```

- HTML5 simplifies the html line:

```
<html lang="en">
```

The lang attribute in the <html> element declares which language the page content is in. Though not strictly required, it should always be specified, as it can assist search engines and screen readers.

- HTML5 simplifies the <head> tag:

```
<head>
  <meta charset="utf-8">
  <title>My First HTML5 Page</title>
  <link rel="stylesheet" href="style.css">
</head>
```

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>My First HTML5 Page</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <p>HTML5 is fun!</p>
</body>
</html>
```

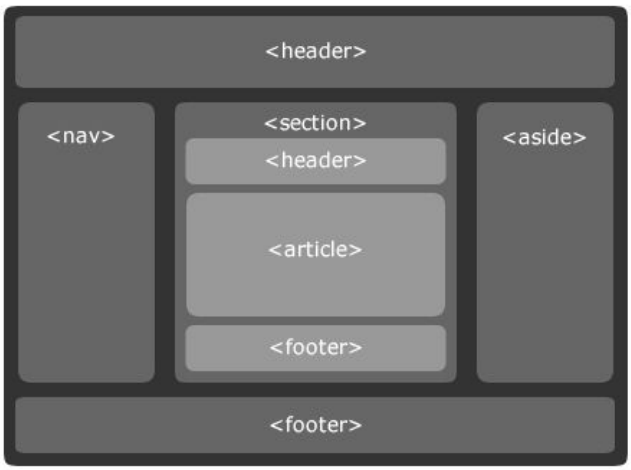
Recall HTML4 :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">

<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-88591">
      <title>HTML.it</title>
      <link rel="stylesheet"  type="text/css"  href="style.css">
   </head>

   <body
   .....
   </body>
</html>
```

type attribute is purely advisory and explains in detail how browsers should act if it's omitted. It doesn't explicitly say that an omitted type attribute is either valid or invalid, but you can safely omit it knowing that browsers will still react as you expect.

HTML5 sectioning elements

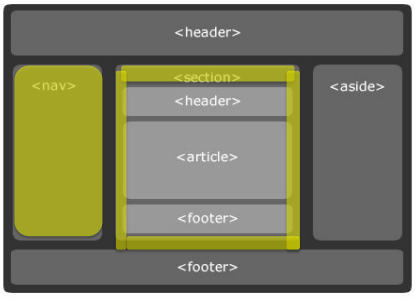
- HTML5 offers new sectioning elements:
  - **<article>**
  - **<section>**
  - **<nav>**
  - **<aside>**
  - **<hgroup>**
  - **<header>**
  - **<footer>**



- **<section>** – the `section` element represents **a generic document or application section**. A section, in this context, is a thematic grouping of content, typically with a heading.

Examples of sections would be chapters, the numbered sections of a thesis..... A Web site's home page could be split into sections for an introduction, news items, contact information.

- **<nav>** – the `nav` element represents **a section of a page that links to other pages or to parts** within the page: a section with navigation links.
  Not all groups of links on a page need to be in a nav element — only sections that consist of **major navigation blocks** are appropriate

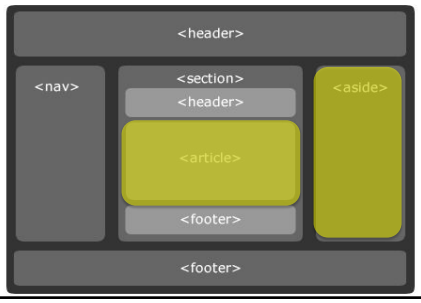


- **<article>** – the article element represents **a component of a page that consists of a self-contained composition** in a document, page, application, or site and that is intended to be independently distributable or reusable, e.g. in syndication.

This could be a forum post, a magazine or newspaper article, a Web log entry, a user-submitted comment, an interactive widget or gadget, or any other independent item of content.

- **<aside>** – the aside element represents **a section of a page that consists of content that is tangentially related to the content around the aside element**, and which could be considered separate from that content.

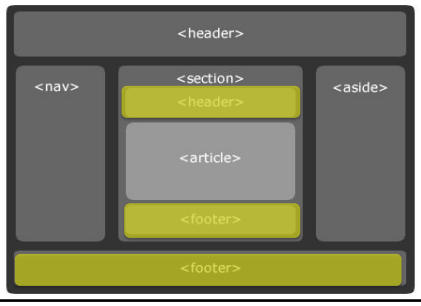
Such sections are often represented as sidebars in printed typography. The element can be used for advertising, for groups of nav elements, and for other content that is considered separate from the main content of the page.



- **<hgroup>** – the hgroup element represents **the heading of a section**. The element is used to group a set of h1–h6 elements when the heading has multiple levels, such as subheadings, alternative titles….
- **<header>** – the header element represents **a group of introductory or navigational aids**. A header element is intended to contain **the section's heading to wrap a section's table of content**s, a search form, or any relevant logos.
- **<footer>** – the footer element represents **a footer for its nearest ancestor sectioning content or sectioning root element**. A footer typically contains information about its section such as who wrote it, links to related documents, copyright data, and the like.

Footers don't necessarily have to appear at the end of a section, though they usually do. It can contain entire sections, indexes, ….. and other such content.

It is common for footers to have a short list of links to common pages of a site, such as the terms of service, the home page, and a copyright page. The `footer` element alone is sufficient for such cases, without a `nav` element.



- Sectioning elements provide semantic alternatives to ** ** in HTML4 documents. This doesn't mean is deprecated in HTML5 — you just won't need them anymore to markup these common document portions.
- In Internet Explorer until version 8 you just have to add some CSS to make them work as intended:

```
CSS
 article, aside, footer, header, section {
 display: block;
 }
```

- The sectioning element is not a generic container element. When an element is needed for styling purposes, authors are encouraged to use the ** ** element instead.
- A general rule is that the **sectioning element is appropriate only if the element's contents would be listed explicitly in the document's outline**

Elements' popularity

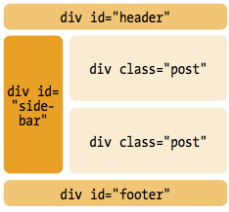
TABLE 1.1 Class Names

POPULARITY	VALUE	FREQUENCY
1	footer	179,528
2	menu	146,673
3	style1	138,308
4	msonormal	123,374
5	text	122,911
6	content	113,951
7	title	91,957
8	style2	89,851
9	header	89,274
10	copyright	86,979
11	button	81,503
12	main	69,620
13	style3	69,349
14	small	68,995
15	nav	68,634
16	clear	68,571
17	search	59,802
18	style4	56,032
19	logo	48,831
20	body	48,052

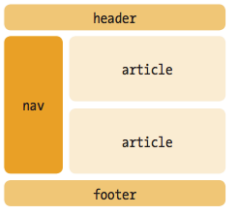
TABLE 1.2 ID Names

POPULARITY	VALUE	FREQUENCY
1	footer	288,061
2	content	228,661
3	header	223,726
4	logo	121,352
5	container	119,877
6	main	106,327
7	table1	101,677
8	menu	96,161
9	layer1	93,920
10	autonumber1	77,350
11	search	74,887
12	nav	72,057
13	wrapper	66,730
14	top	66,615
15	table2	57,934
16	layer2	56,823
17	sidebar	52,416
18	image1	48,922
19	banner	44,592
20	navigation	43,664

HTML 4



HTML 5

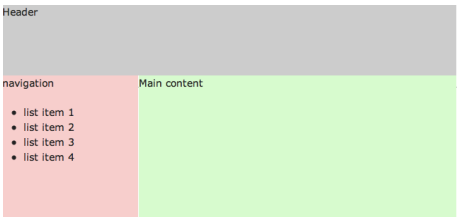


2009 Opera MAMA crawler statistics on class names and ids used in in 2,148,723 randomly chosen URLs

HTML5 Semantics example 2 columns layout

HTML

```
<!DOCTYPE html>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
 <html>
   <head>
     <title><!-- Your Title --></title>
   </head>

   <body>
     <header>
       <!-- ... -->
     </header>

     <nav>
       <!-- ... -->
     </nav>

     <div id="main">
       <!-- ... -->
     </div>

     <footer>
       <!-- ... -->
     </footer>
   </body>
</html>
```

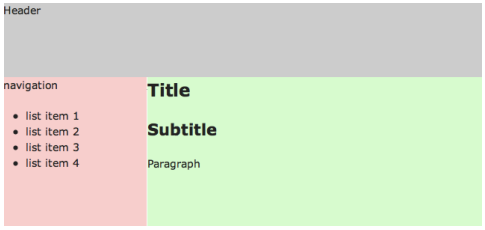


CSS

```
section, header, footer, aside, nav, article{
  display: block;
}

header {width:100%; height: 100px; background:#ccc;}
nav {float:left; width: 30%; background:#fcc; min-height:200px;}
#main {width:70%; background:#cfc; float:right; min-height:200px;}
```

HTML continue

```
<!DOCTYPE html>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">

<html>
    <body>
        <div id="main">
            <article>
              <hgroup>
                  <h2>Title</h2>
                  <h3>Subtitle</h3>
              </hgroup>
              <p>
                  <!-- ---->
              </p>
            </article>
        </div>
    </body>
</html>
```



CSS

```
section, header, footer, aside, nav, article{
  display: block;
}

header {width:100%; height: 100px; background:#ccc;}
nav {float:left; width: 30%; background:#fcc; min-height:200px;}
#main {width:70%; background:#cfc; float:right; min-height:200px;}
```

Semantics / article vs. section

An <article> is an independent, stand-alone piece of discrete content. (blog post, news item, individual emails within an email application, stories in a web-based feed reader)

```
<article>
<h1>Bruce Lawson is World's Sexiest Man</h1>

<p>Legions of lovely ladies voted luscious lothario Lawson ¬as the World's Sexiest Man today.</p>

<h2>Second-sexiest man concedes defeat</h2>

<p>Remington Sharp, jQuery glamourpuss and Brighton roister-doister, was gracious in defeat. "It's cool being the second sexiest man when number one is Awesome Lawson" he said from his swimming pool-sized jacuzzi full of supermodels.
</p>
</article>
```

Comments on blog posts are <article> inside a parent <article>. There are other uses for this nesting beside comments, for example a transcript to a video.

```
<article>
  <h1>Stars celebrate Bruce Lawson</h1> <video>...</video>
  <article class=transcript>
    <h1>Transcript</h1>
    <p>Supermodel #1: "He's so hunky!"</p> <p>Supermodel #2: "He's a snogtabulous bundle of gorgeous ¬manhood! And I saw him first, so hands off!"</p>
  </article>
</article>
```

Semantics / article vs. section

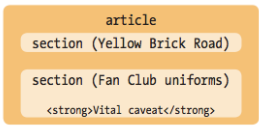
<section> isn't a self-contained composition in a document, page, application, or site and that is intended to be independently distributable or reusable." It's either a way of sectioning a page into different subject areas, or sectioning an article into sections. It can contain articles or can represent part of an article

```
<article>
  <h1>Rules for Munchkins</h1>
  <section>
    <h2>Yellow Brick Road</h2>
    <p>It is vital that Dorothy follows it so no selling bricks as "souvenirs"</p>
  </section>

  <section>
    <h2>Fan Club uniforms</h2>
    <p>All Munchkins are obliged to wear their "I'm a friend of Dorothy!" t-shirt when representing the club</p>
  </section>

  <p><strong>Vital caveat about the information above: ¬does not apply on the first Thursday of the month. ¬
</strong></p>
</article>
```

Semantics / article vs. section section

HTML5

```
<article>
  <h1>Rules for Munchkins</h1>
  <section>
    <h2>Yellow Brick Road</h2>
    <p>It is vital that Dorothy follows
it so no selling bricks as
“souvenirs”</p>
  </section>

  <section>
    <h2>Fan Club uniforms</h2>
    <p>All Munchkins are obliged to wear
their “I’m a friend of Dorothy!” t-shirt
when representing the club</p>
 </section>

  <p><strong>Vital caveat about the
information above: ¬does not apply on the
first Thursday of the month. ¬
</strong></p>
</article>
```



In **HTML4** this semantic distinction would be difficult. Which sentence the caveat is referred to?

<h1>Rules for Munchkins**</h1>**

<h2>Yellow Brick Road**</h2>**
** **It is vital that Dorothy follows it so no selling bricks ¬as “souvenirs”** **

<h2>Fan Club uniforms**</h2>**
** **All Munchkins are obliged to wear their “I’m a friend of Dorothy!” t-shirt when representing the club** **

** **Vital caveat about the information above: does not apply on the first Thursday of the month. ** **

HTML5 Semantics inline elements

<time> – the time element represents either a time on a 24 hour clock, or a precise date in the Gregorian calendar, optionally with a time and a time-zone offset.

```
<time datetime="2009-10-22" pubdate>October 22, 2009</time>
```

If the **<time>** element is in an **<article>** element, timestamp is the publication date of the article.
If the **<time>** element is not in an **<article>** element, timestamp is the publication date of the entire document.

<mark> – the mark element represents a run of text in one document marked or highlighted for reference purposes

```
Basically, it is used to bring the reader's attention to <mark>a part of the text</mark
```

<figure>: The **<figure>** tag specifies self-contained content, like illustrations, diagrams, photos, code listings, etc. While the content of the **<figure>** element is related to the main flow, its position is independent of the main flow, and if removed it should not affect the flow of the document

```
<figure>
  <img src="img_pulpit.jpg" alt="The Pulpit Rock"
  width="304" height="228" />
</figure>
```

<meter>: "Meter" is a new element in HTML5 which represenet value of a known range as a gauge. You are only allowed to use it when you are clearly aware of its minimum value and maximum value.

One example is score of rating. I would rate this movie <meter min="0" max="10" value="8">8 of 10</meter>.

Science :
Math :
Geography :
History :

```
Science : <meter min="0" max="100" value="95">95 of 100</meter> <br />
Math : <meter min="0" max="100" value="60">60 of 100</meter><br />
Geography : <meter min="0" max="100" value="20">20 of 100</meter> <br />
History : <meter min="0" max="100" value="50">50 of 100</meter>
```

HTML5 Semantics outlining

- HTML5 has an *outlining algorithm* that allows user agents to produce a outline from a web page. This could be used to give the user a quick overview of the web page
- One major departure from HTML4, and an important concept to grasp is that certain HTML5 elements **<article>, <section>, <nav>,** and **<aside>** are sectioning content, which begin new sections in the outline.
-

1. Hello
2. World

```
<h1>Hello</h1>
<div>
  <h1>World</h1>
</div>
```

1. Hello
   1. World

```
<h1>Hello</h1>
<article>
  <h1>World</h1>
</article>
```

The **<h1>** inside the article is a logical **<h2>** because **<article>** has started a new section. Using **<section>, <nav>,** or **<aside>** instead of **<article>** does the same thing, as they are all sectioning content.

Sectioning level

- It doesn't matter what level of heading is used; the outlining algorithm cares about nesting and relative levels, so this code:

```
<h3>Hello</h3>
  <article>
    <h6>World</h6>
   </article>
```

It means you're not restricted to six levels of headings, as in HTML 4. A heading element nested inside seven levels of **<section>, <article>, <nav>,** or **<aside>** becomes a logical **<h7>** element .

Rule: Sections may contain headings of any rank, but authors are strongly encouraged to either **use only h1 elements**, or to use elements of the appropriate rank for the section's nesting level.

Graphics and Multimedia Elements

- HTML5 offers tags for new **graphics** and **media** elements:

Graphics elements:

<canvas> Defines graphic drawing using JavaScript
<svg> Defines graphic drawing using SVG

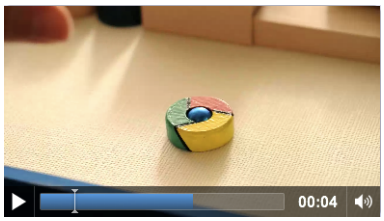
Media elements:

<audio> Defines sound or music content
<embed> Defines containers for external applications (like plug-ins)
<source> Defines sources for <video> and <audio>
<track> Defines tracks for <video> and <audio>
<video> Defines video or movie content

Graphics and Multimedia: Video and Audio

- Examples of video and audio tags:

```
<video id="video" src="movie.webm" autoplay controls></video>
document.getElementById("video").play();
```




```
<audio id="audio" src="sound.mp3" controls></audio>
document.getElementById("audio").muted = false;
```

Graphics and Multimedia: Video and Audio

- You may think of video files as "AVI files" or "MP4 files." In reality, "AVI" and "MP4" are just **container formats**. Just like a ZIP file can contain any sort of file within it, video container formats only define *how* to store things within them.

There are *lots* of video container formats. Some of the most popular include:

- **MPEG 4**, usually with an .mp4 or .m4v extension. The MPEG 4 container is based on Apple's older QuickTime container (.mov). Movies that you rent from iTunes are delivered in an MPEG 4 container.
- **Flash Video**, usually with an .flv extension. Flash Video is, unsurprisingly, used by Adobe Flash. Prior to Flash 9.0.60.184 (a.k.a. Flash Player 9 Update 3), this was the only container format that Flash supported. More recent versions of Flash also support the MPEG 4 container.
- **Ogg**, usually with an .ogv extension. Ogg is an open standard, open source–friendly, and unencumbered by any known patents. Firefox 3.5, Chrome 4, and Opera 10.5 support the Ogg container format, Ogg video (called "Theora"), and Ogg audio (called "Vorbis").
- **WebM** is a new container format (2010). It is supported natively, without platform-specific plugins, in the latest versions of Chromium, Google Chrome, Mozilla Firefox, and Opera. Adobe has also announced that a future version of Flash will support WebM video.
- **Audio Video Interleave**, usually with an .avi extension. The AVI container format was invented by Microsoft in a simpler time, when the fact that computers could play video at all was considered pretty amazing. It does not officially support features of more recent container formats like embedded metadata. It does not even officially support most of the modern video and audio codecs in use today.

Graphics and Multimedia / Video and Audio

When you "watch a video," your video player is doing at least three things at once:

- Interpreting the container format to find out which video and audio tracks are available, and how they are stored within the file so that it can find the data it needs to decode next
- Decoding the video stream and displaying a series of images on the screen
- Decoding the audio stream and sending the sound to your speakers

A *video codec* is an algorithm by which a video stream is encoded. There are *tons* of video codecs. The three most relevant codecs are H.264, Theora, and VP8.

- **H.264** is also known as "MPEG-4 part 10," a.k.a. "MPEG-4 AVC," a.k.a. "MPEG-4 Advanced Video Coding." H.264 was also developed by the MPEG group and standardized in 2003. It aims to provide a single codec for low-bandwidth, low-CPU devices (cell phones); high-bandwidth, high-CPU devices (modern desktop computers); and everything in between.
- To accomplish this, the H.264 standard is split into "**profiles**," which each define a set of optional features that trade complexity for file size.
- Apple's iPhone supports Baseline profile,
- the AppleTV set-top box supports Baseline and Main profiles,
- Adobe Flash on a desktop PC supports Baseline, Main, and High profiles.
- YouTube now uses H.264 to encode high-definition videos, playable through Adobe Flash;
- YouTube also provides H.264-encoded video to mobile devices, including Apple's iPhone and phones running Google's Android mobile operating system

- **Theora** evolved from the VP3 codec and has subsequently been developed by the Xiph.org Foundation.
- **Theora is a royalty-free codec and is not encumbered by any known patents** other than the original VP3 patents, which have been licensed royalty-free. Theora video can be embedded in any container format, although it is most often seen in an Ogg container. All major Linux distributions support Theora out-of-the-box, and Mozilla Firefox 3.5 includes native support for Theora video in an Ogg container.

- **VP8** is another video codec from On2, the same company that originally developed VP3 (later Theora). Technically, it produces output on par with H.264 High Profile, while maintaining a low decoding complexity on par with H.264 Baseline. In 2010, Google acquired On2 and published the video codec specification and a sample encoder and decoder as open source.

Like video codecs, ***audio codecs*** are algorithms by which an audio stream is encoded. There are *lots* of audio codecs but on the web, there are really **only three you need to know** about:

- **MP3**. The MP3 format (standardized in 1991) is patent-encumbered, which explains why Linux can't play MP3 files out of the box. Pretty much every portable music player supports standalone MP3 files, and MP3 audio streams can be embedded in any video container. Adobe Flash can play both standalone MP3 files and MP3 audio streams within an MP4 video container.
- **AAC**. Advanced Audio Coding is affectionately known as "AAC." Standardized in 1997, it lurched into prominence when Apple chose it as their default format for the iTunes Store. The AAC format is patent-encumbered. All current Apple products, including iPods, AppleTV, and QuickTime support certain profiles of AAC in standalone audio files and in audio streams in an MP4 video container. Adobe Flash supports all profiles of AAC in MP4, as do the open source MPlayer and VLC video players.
- **Vorbis**. Vorbis is often called "Ogg Vorbis," although this is technically incorrect. ("Ogg" is just a container format, and Vorbis audio streams can be embedded in other containers.) Vorbis is not encumbered by any known patents and is therefore supported out-of-the-box by all major Linux distributions and by portable devices running the open source Rockbox firmware. Mozilla Firefox 3.5 supports Vorbis audio files in an Ogg container, or Ogg videos with a Vorbis audio track. Android mobile phones can also play standalone Vorbis audio files. Vorbis audio streams are usually embedded in an Ogg or WebM container, but they can also be embedded in an MP4 or MKV container (or, with some hacking, in AVI).

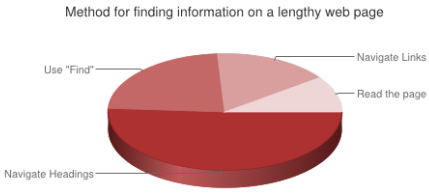
VIDEO CODEC SUPPORT IN SHIPPING BROWSERS

CODECS/CONTAINER	IE	FIREFOX	SAFARI	CHROME	OPERA	IPHONE	ANDROID
Theora+Vorbis+Ogg	·	3.5+	†	5.0+	10.5+	·	·
H.264+AAC+MP4	9.0+	·	3.0+	5.0–?‡	·	3.0+	2.0+
WebM	9.0+*	·	†	6.0+	10.6+	·	·

- Mozilla Firefox (3.5 and later) supports Theora video and Vorbis audio in an Ogg container.
- Firefox 4 also supports WebM.
- Opera (10.5 and later) supports Theora video and Vorbis audio in an Ogg container. Opera 10.60 also supports WebM.
- Google Chrome (3.0 and later) supports Theora video and Vorbis audio in an Ogg container. Google Chrome 6.0 also supports WebM.
- Safari on Macs and Windows PCs (3.0 and later) will support anything that QuickTime supports. It *does* ship with support for H.264 video (main profile) and AAC audio in an MP4 container.
- Mobile phones like Apple's iPhone and Google Android phones support H.264 video (baseline profile) and AAC audio ("low complexity" profile) in an MP4 container.
- Adobe Flash (9.0.60.184 and later) supports H.264 video (all profiles) and AAC audio (all profiles) in an MP4 container.
- Internet Explorer 9 supports all profiles of H.264 video and either AAC or MP3 audio in an MP4 container. It will also play WebM video if you install a third-party codec, which is not installed by default on any version of Windows. IE9 does not support other third-party codecs (unlike Safari, which will play anything QuickTime can play).
- Internet Explorer 8 has no HTML5 video support at all, but virtually all Internet Explorer users will have the Adobe Flash plugin.

Semantics / accessibility of outlining algorythm

50% of screen reader users often navigate by headings



Response	# of Respondents	% of Respondents
Navigate through the headings on the page	321	50.8%
Use the "Find" feature	145	22.9%
Navigate through the links of the page	102	16.1%
Read through the page	64	10.1%

http://webaim.org/projects/screenreadersurvey2/