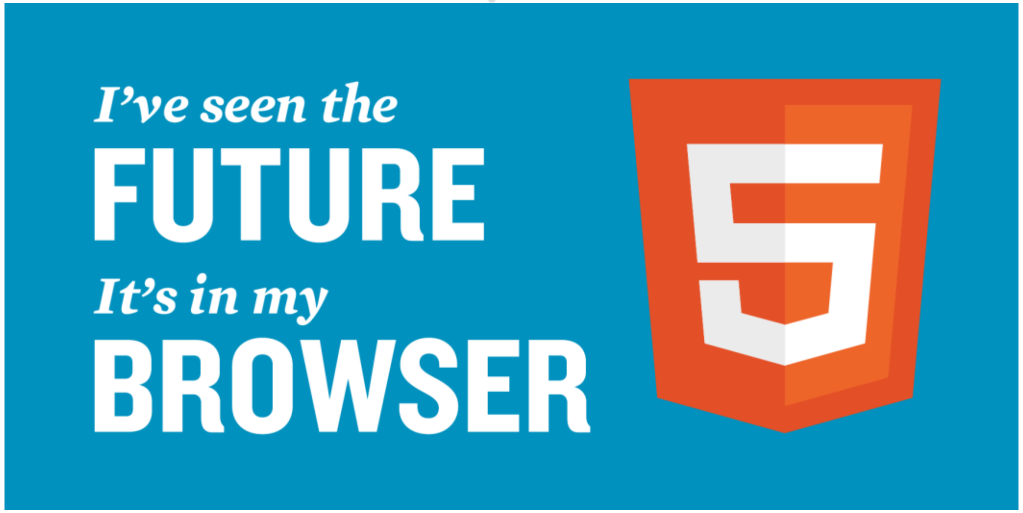
Sign of success

HTML 5

New markups
and attributes

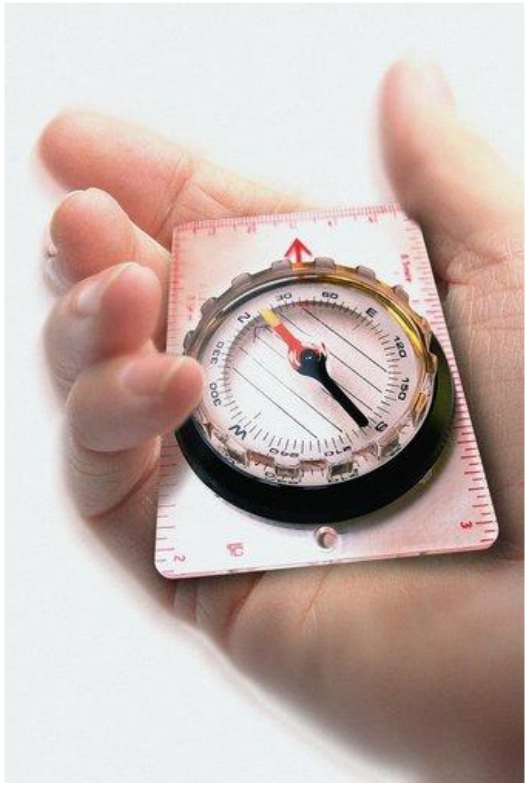


Course objectives

By completing this course, you will be able to:

- Use new semantic markups
- Validate form fields without JavaScript
- Use new form input types
- Play media resources without plugins
- Make accessible widgets thanks to ARIA
- Structure your data markup with Microdatas

Course topics



Course's plan:

– Semantic Markups
– Web Form
– Media Markups
– ARIA
– Microdata

HTML5 - New markups

SEMANTIC MARKUPS

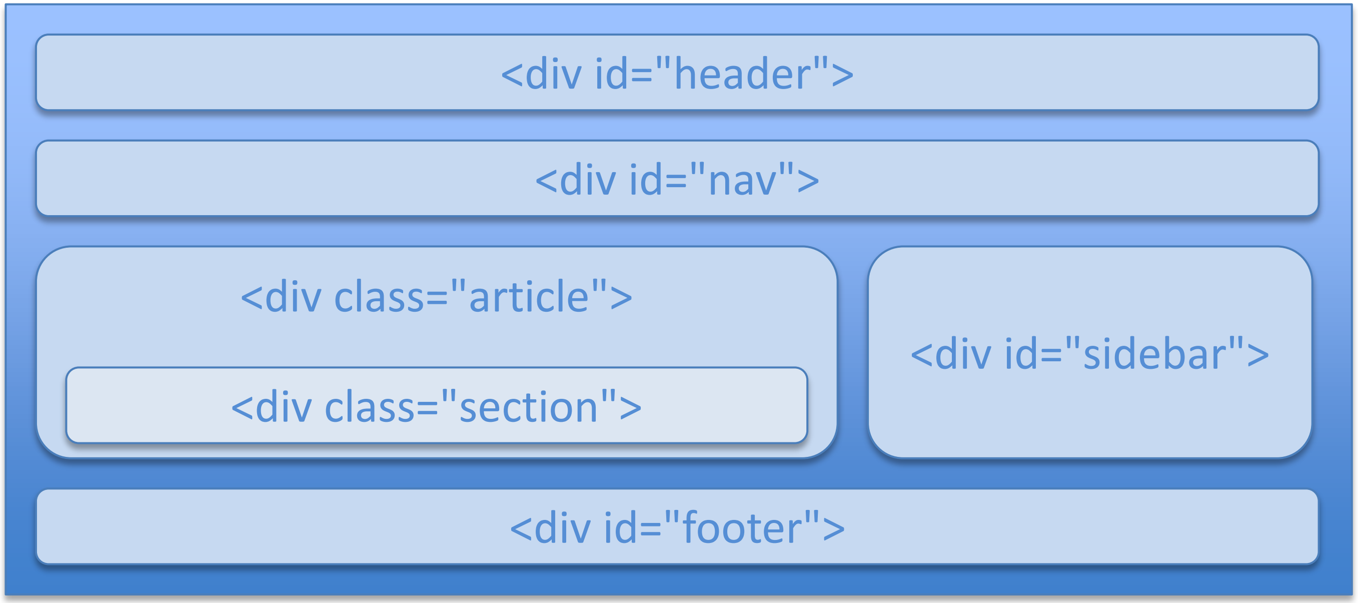
Presentation

- Semantic elements describe their meaning or purpose clearly to the browsers

- With HTML5, no more overuse of the *div* tag to define a division or a section
  - Doesn’t tell us anything about its content
  - Doesn’t convey any clear meaning

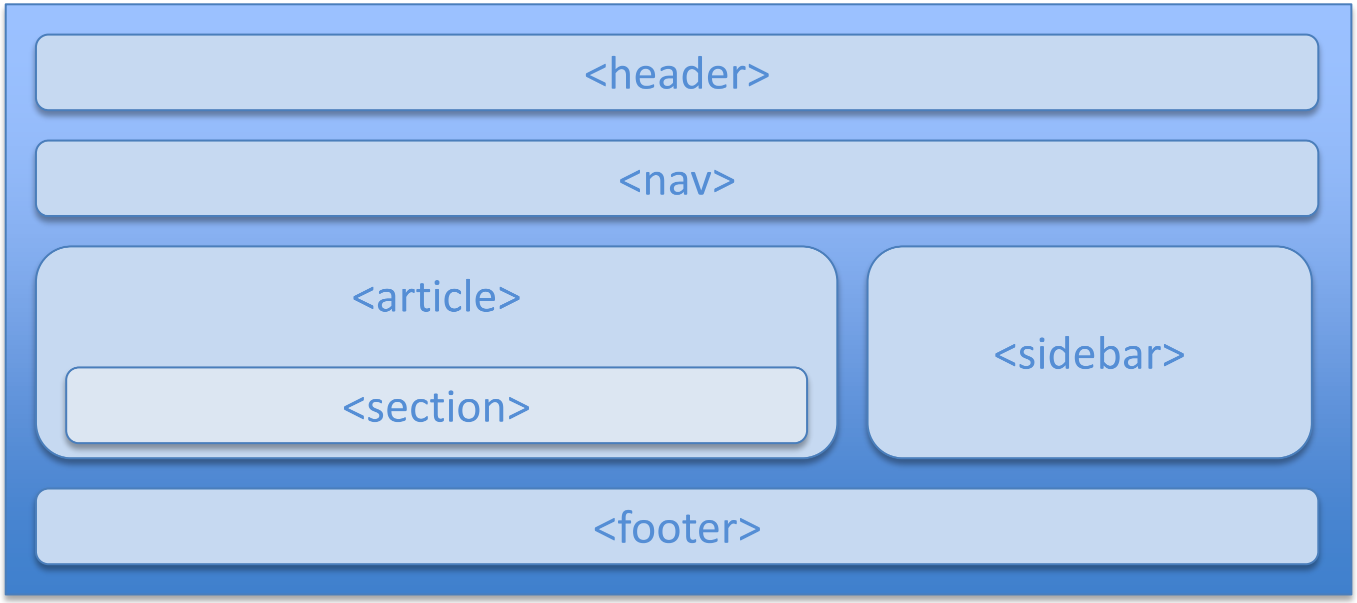
Semantic

- Most websites have some common uses of generic blocks:



Semantic

- HTML5 introduces the following new markups:



Semantic Example

```html
<body>
  <header><!-- Header content --></header>
  <article>
     <section>
           <!-- Section content -->
     </section>
     <section>
           <!-- Other section content -->
     </section>
  </article>
  <aside>
     <!-- Aside content -->
  </aside>
  <footer><!-- Footer content --></footer>
</body>
```

Section

- The **section** element represents a generic section of a document
  - A chapter for example

```
<section>
   <h1>Chapter 2 : Basic HTML tags</h1>
   <p>
       HTML markup consists of several key components,
       including elements […], character-based data types,
       character references and entity references
   </p>
</section>
```

Section

- The **section** element is not a generic container element
  - Use *div* when an element is needed only for styling purposes or as a convenience for scripting
  - Use **article** when it would make sense to syndicate the contents of the element

Article

- The article element represents a self-contained composition independently distributable or reusable
  - News
  - Blog posts
  - …

```
<article id="comment-6">
   <header>
      <h4>John Doe</h4>
   </header>
   <p>Very good course ;-)</p>
</article>
```

Nav

- The **nav** element represents a section of navigation links

```
<nav>
   <h1>Navigation</h1>
   <ul>
       <li><a href="/">Home</a></li>
       <li><a href="/about.html">About</a></li>
   </ul>
</nav>
```

Header

- The **header** element represents the header of a section

```
<header>
  <h1>Introduction to HTML5</h1>
  <p class="author-line">By Brice Argenson</p>
</header>
```

Semantic markups

Footer

- The **footer** element represents a footer for its nearest ancestor sectioning content

```
<footer>
   <p>© 2012 SUPINFO International University</p>
</footer>
```

Address

- The **address** element represents the contact information for its nearest article or body element ancestor

```
<footer>
   <address>
       For more details, contact
       <a href="mailto:js@example.com">John Smith</a>.
   </address>
   <p><small>© Copyright 2042 Plop Corp.</small></p>
</footer>
```

Hgroup

- The **hgroup** element represents the heading of a section

- Used to group a set of **h1–h6** elements when the heading has multiple levels
  - Such as subheadings, alternative titles, or taglines

Hgroup

- Example:

```
<hgroup>
   <h1>HTML5</h1>
   <h2>Or: How to design modern Web Applications</h2>
</hgroup>
```

Headings and sections

- The **h1–h6** elements and the **hgroup** element are headings

- The first in a section represents the heading for that section

Headings and sections

- Subsequent headings
  - of equal or higher rank:
    - start new (implied) sections
  - of lower rank:
    - start implied subsections that are part of the latter

Semantically equivalent

```
<body>
 <h1>HTML5</h1>
 <h2>Offline Webapps</h2>
 <h3>LocalStorage</h3>
 <h2>New Markups</h2>
</body>
```

```
<body>
  <h1>HTML5</h1>
  <section>
    <h1>Offline Webapps</h1>
    <section>
     <h1>LocalStorage</h1>
    </section>
  </section>
  <section>
    <h1>New Markups</h1>
  </section>
</body>
```

Questions ?

HTML5 - New markups

WEB FORMS

Presentation

- HTML5 introduce an update to the forms features found in HTML4
- Add support for common needs like
  - Basic data typing
  - Simpler validation on the client side
  - XML submission
  - …

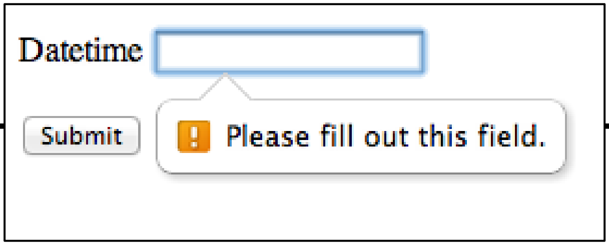
Client-side validation

- Forms can be annotated to check the user's input before the form is submitted
- Allows the user to avoid the wait of a validation by the server
- Before HTML5, we used JavaScript to do that

Client-side validation

- HTML5 introduce the *required* annotation
  - With it, an input can't be submitted until a value is given

```
<p>
  <label>
     Datetime: <input type="datetime" required />
  </label>
</p>
```



Client-side validation

- It is also possible to limit the length of the input, using the *maxlength* attribute

```
<p>
   <label>
       Comments: <textarea name="comments" maxlength="160"/>
   </label>
</p>
```

Client-side validation

- The *pattern* attribute allows you to specifies a RegEx that the control’s value has to match

```
<label>
  Course Code: <input type="text" pattern="[1-5][A-Z]{3}"/>
</label>
```

Client-side validation

- The *min* and *max* attributes indicate the allowed range of values for the element

```
<label>
  Birthday: <input type="date" min="1900-12-31" />
</label>

<label>
  Quantity: <input type="number" min="1" max="20" />
</label>
```

New input types

- New input types are available :
  - search
  - tel
  - url
  - email
  - datetime
  - date
  - month
  - week
  - time
  - number
  - range
  - color
  - image

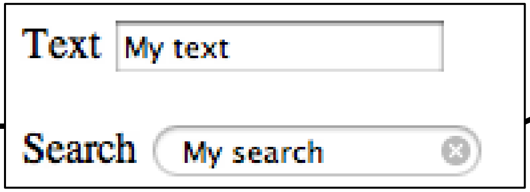
New input types

- Each one of these new types, bring one or more of the following advantages:
  - – A better semantic (*tel*, *search*, …)
  - – User input validation (*email*, *url*, *number*, …)
  - – New controls provided by the browser (*range*, …)

Search field

- The *search* type is very similar to the *text* one
  - The difference is primarily stylistic

```
<p>
  <label>
    Search <input type="search" />
  </label>
</p>
```


Text My text

Search My search

Telephone field

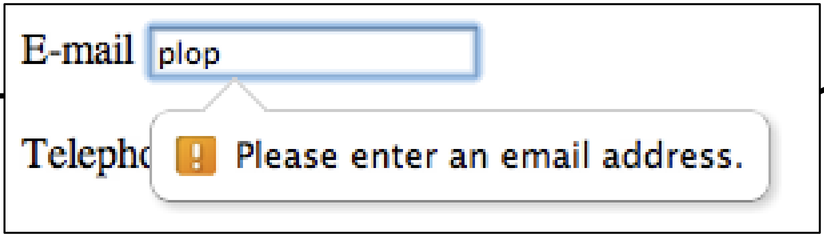
- The *tel* type represents a control for editing a telephone number
  - The type doesn't enforce a particular syntax to support all the format variety around the world

```
<p>
  <label>
    Telephone <input type="tel" />
  </label>
</p>
```

Email field

- The *email* type represents a control for editing an email address

```
<p>
  <label>
    E-mail <input type="email" />
  </label>
</p>
```


E-mail plop

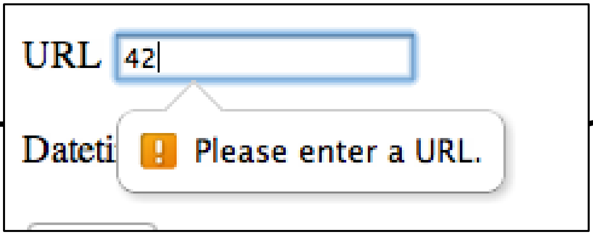
Teleph

Please enter an email address.

URL field

- The *url* type represents a control for editing a single absolute URL

```
<p>
  <label>
    URL <input type="url" />
  </label>
</p>
```


URL 42

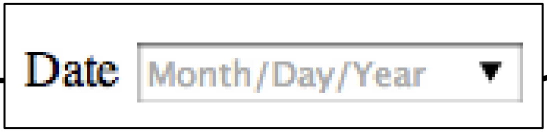
Dateti

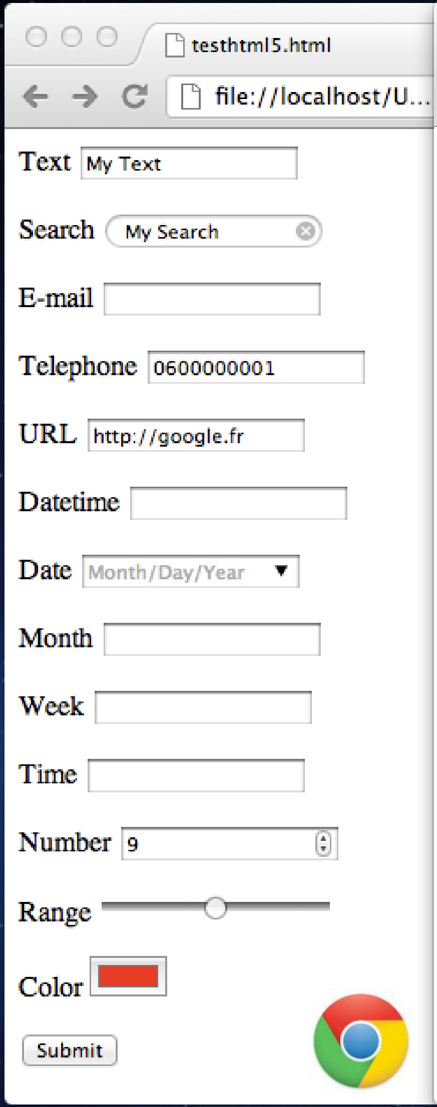
Please enter a URL.

Date field

- The *date* type represents a control for setting the element's value to a specific date

```
<p>
  <label>
    Date <input type="date" />
  </label>
</p>
```




testhtml5.html

file://localhost/U...

Text: My Text

Search: My Search

E-mail:

Telephone: 0600000001

URL: http://google.fr

Datetime:

Date: Month/Day/Year

Month:

Week:

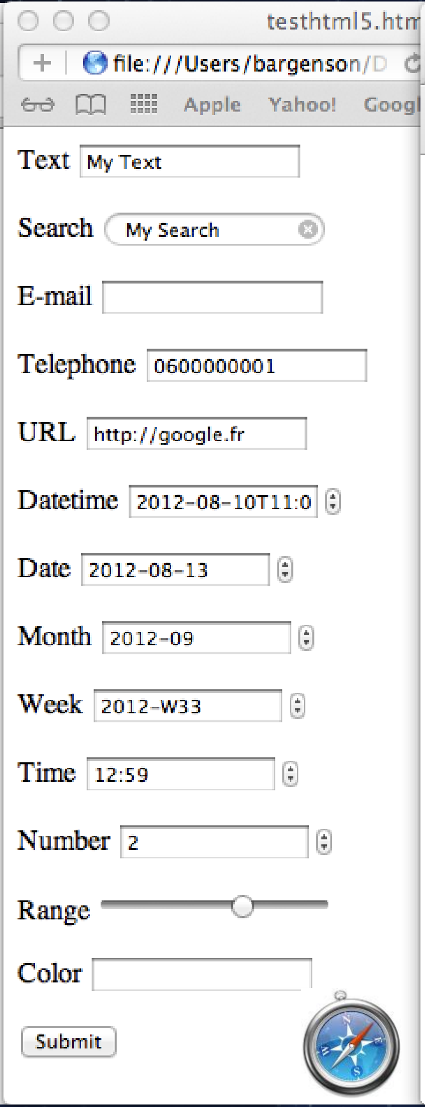
Time:

Number: 9

Range:

Color:

Submit


testhtml5.htm

file:///Users/bargenson/D

Apple Yahoo! Googl

Text: My Text

Search: My Search

E-mail:

Telephone: 0600000001

URL: http://google.fr

Datetime: 2012-08-10T11:0

Date: 2012-08-13

Month: 2012-09

Week: 2012-W33

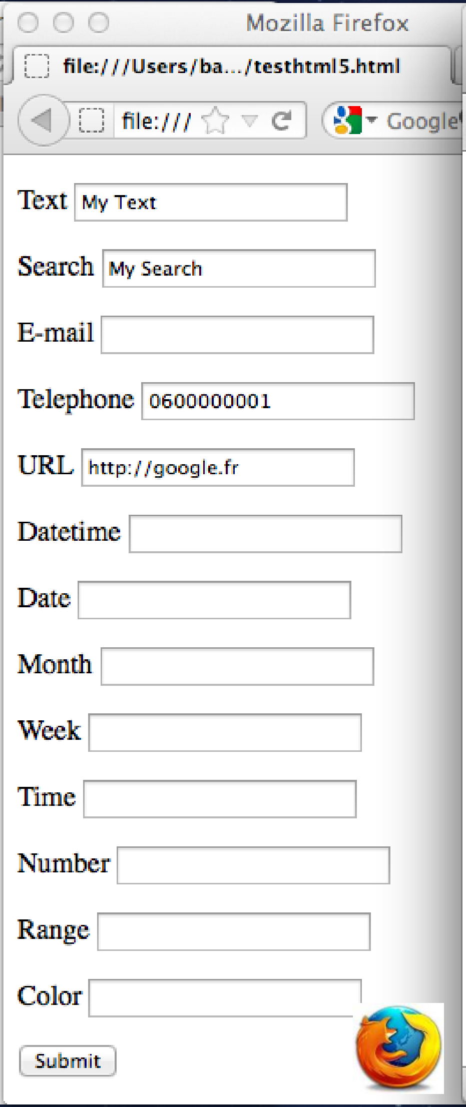
Time: 12:59

Number: 2

Range:

Color:

Submit


Mozilla Firefox

file:///Users/ba.../testhtml5.html

file:///

Google

Text: My Text

Search: My Search

E-mail:

Telephone: 0600000001

URL: http://google.fr

Datetime:

Date:

Month:

Week:

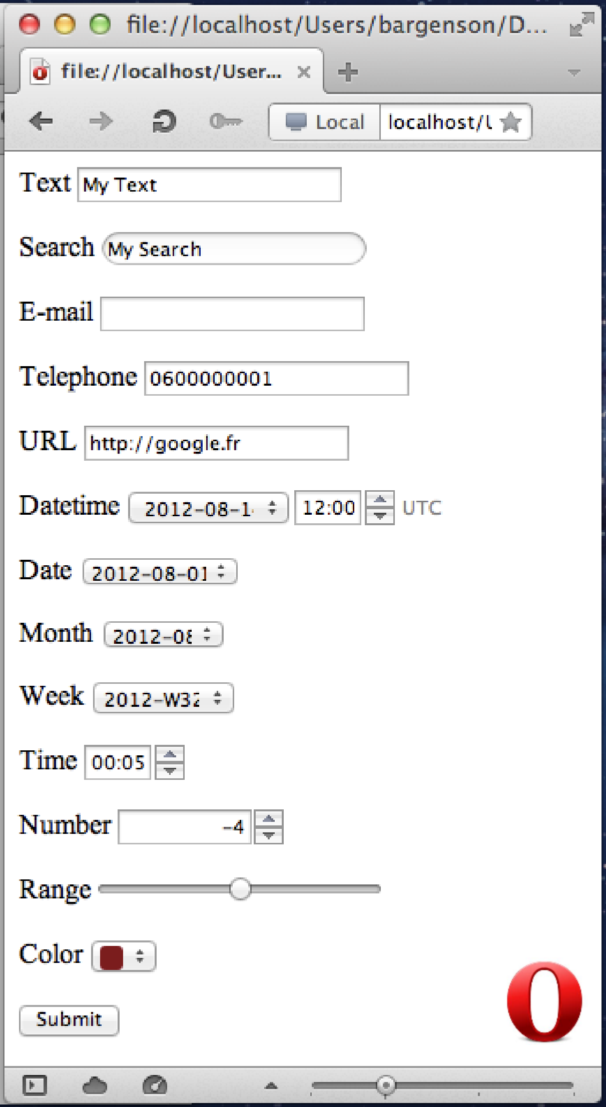
Time:

Number:

Range:

Color:

Submit


file://localhost/Users/bargenson/D...

file://localhost/User...

Local localhost/l

Text: My Text

Search: My Search

E-mail:

Telephone: 0600000001

URL: http://google.fr

Datetime: 2012-08-1 12:00 UTC

Date: 2012-08-01

Month: 2012-0

Week: 2012-W3

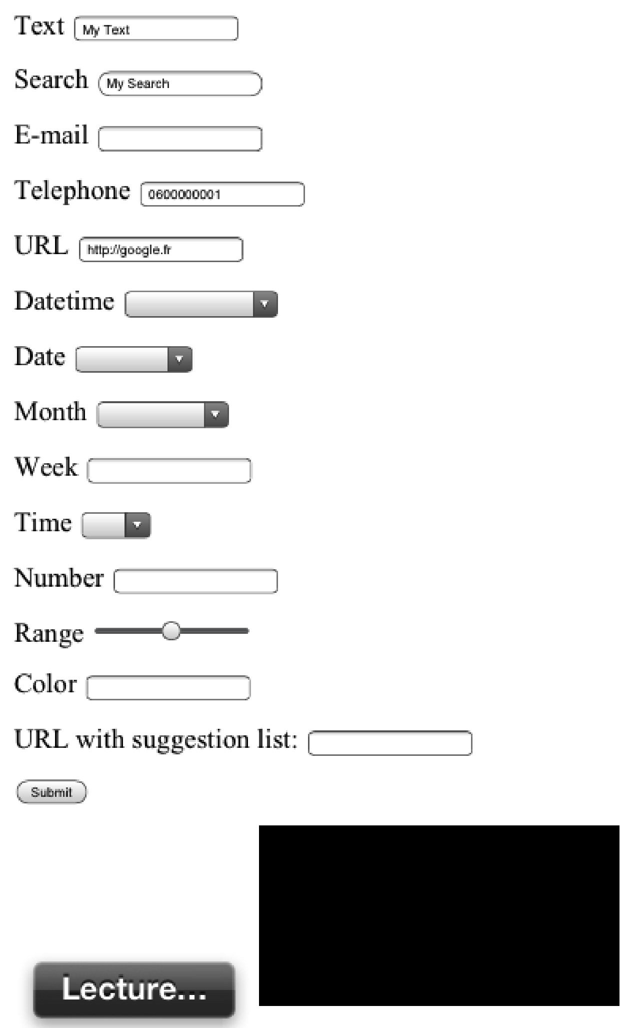
Time: 00:05

Number: -4

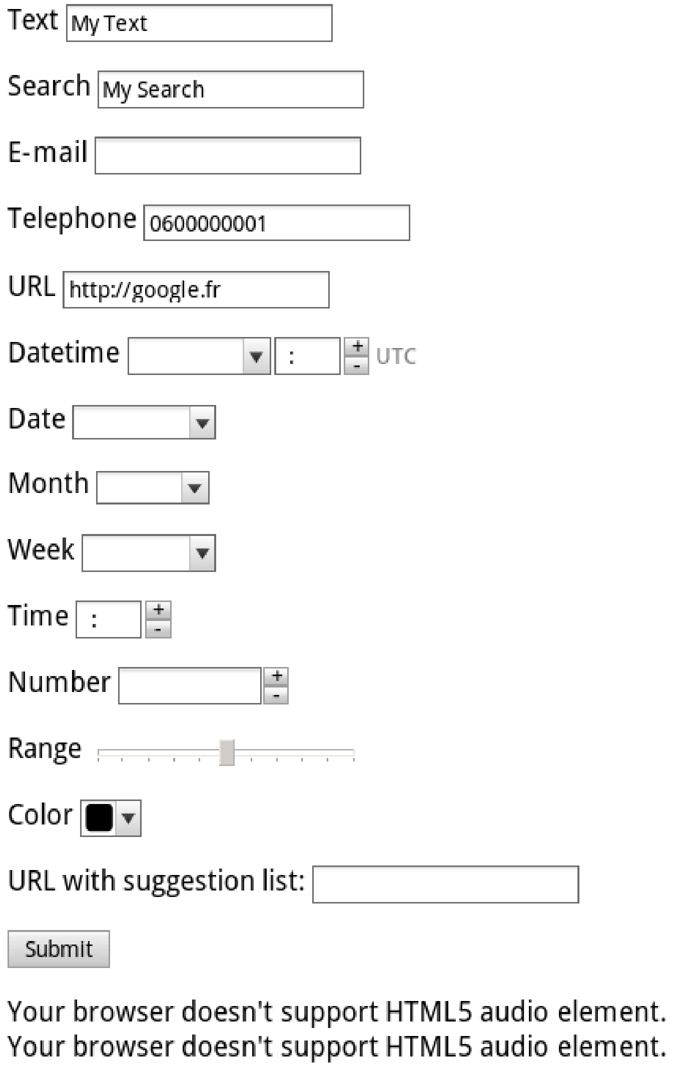
Range:

Color:

Submit



Text	My Text
Search	My Search
E-mail	
Telephone	0600000001
URL	http://google.fr
Datetime	
Date	
Month	
Week	
Time	
Number	
Range	
Color	
URL with suggestion list:	



Text	My Text
Search	My Search
E-mail	
Telephone	0600000001
URL	http://google.fr
Datetime	: UTC
Date	
Month	
Week	
Time	:
Number	
Range	
Color	
URL with suggestion list:	

Your browser doesn't support HTML5 audio element.
Your browser doesn't support HTML5 audio element.

Auto-completion

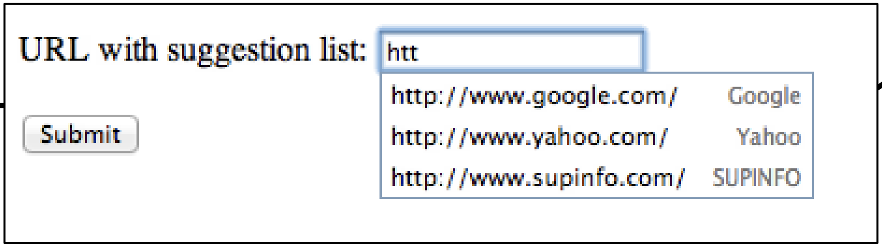
- The new *list* attribute allows you to defined suggestions to the user
  - The attribute must refer to a *datalist* id containing the suggestions

- You can use *Ajax* to update the *datalist* content in function of the user input if you need to

Web Forms

Auto-completion

```
<label>
   URL with suggestion list:
   <input type="url" list="hpurls">
</label>
<datalist id="hpurls">
   <option value="http://www.google.com/" label="Google">
   <option value="http://www.yahoo.com/" label="Yahoo">
   <option value="http://www.supinfo.com/" label="SUPINFO">
</datalist>
```



URL with suggestion list: htt

http://www.google.com/	Google
http://www.yahoo.com/	Yahoo
http://www.supinfo.com/	SUPINFO

Questions ?

HTML5 - New markups

MEDIA MARKUPS

Audio & Video

- There are more and more Audio and Video on the web

- Until now, you had to use non standardize technologies like Java Applets or Flash
  - – Need specific plug-in

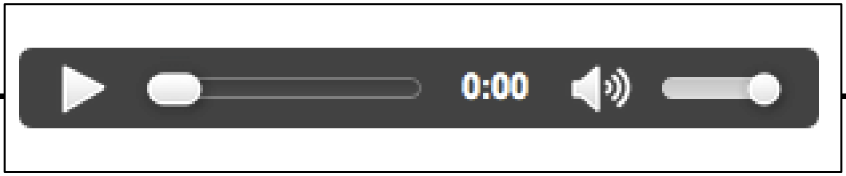
Audio

- The *audio* new element represents a sound or audio stream
- Content may be provided inside the audio element
  - User agents should show this content to the user only if they don't support the *audio* element

Audio

- Example :

```
<audio
   src="http://images.wikia.com/starwars/images/3/38/UlicFal
   ls.ogg" controls>
   Your browser doesn't support the HTML5 audio element.
</audio>
```



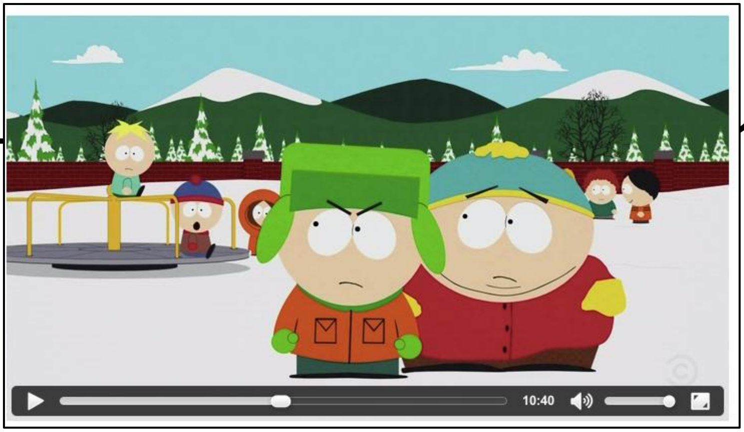
Video

- The new *video* element is used for playing videos and audio files with captions
- Again, content may be provided inside the *video* element
  - User agents should show this content to the user only if they don't support the *video* element

Video

- Example :

```
<video src="myVideo.mp4" poster="movie.jpg" controls>
  Your browser doesn't support the HTML5 video element.
</video>
```



Common attributes

- *audio* and *video* have some common attributes
  - *preload*: to pre-download the media resource
  - *autoplay*: to automatically begin playback of the media resource

Common attributes

- *audio* and *video* have some common attributes
  - *controls*: to ask to the user agent to provide its own set of controls
  - *loop*: to seek back to the start of the media resource upon reaching the end

Alternative sources

- The *source* element allows to specify multiple alternative of sources to a media element

- Allows the browser to choose the better source based on its media type or codec support

Media Markups

Alternative sources

- Example:

```
<audio controls autoplay loop>
  <source src="song.ogg" type="audio/ogg" />
  <source src="song.mp3" type="audio/mpeg" />

  Your browser doesn't support the HTML5 audio element.

</video>
```

Media Markups

Support for audio codecs

Browser	Version	Codec Support
Internet Explorer	9.0+	MP3, AAC
Firefox	3.6+	Ogg Vorbis, WAV
Google Chrome	6.0+	Ogg Vorbis, MP3, WAV (since Chrome 9)
Apple Safari	5.0+	MP3, AAC, WAV
Opera	10.0+	Ogg Vorbis, WAV
Android	2.3+	MP3, AAC (device dependent)
iOS	3.0+	MP3, AAC
Blackberry	6.0+	MP3, AAC

Support for video codecs

Browser	Version	Codec Support
Internet Explorer	9.0+	MP4
Firefox	4.0+	WebM, Ogg
Google Chrome	6.0+	MP4, WebM, Ogg
Apple Safari	5.0+	MP4
Opera	10.6+	WebM, Ogg
Android	2.3+	MP4, WebM (since 4.0)
iOS	3.0+	MP4

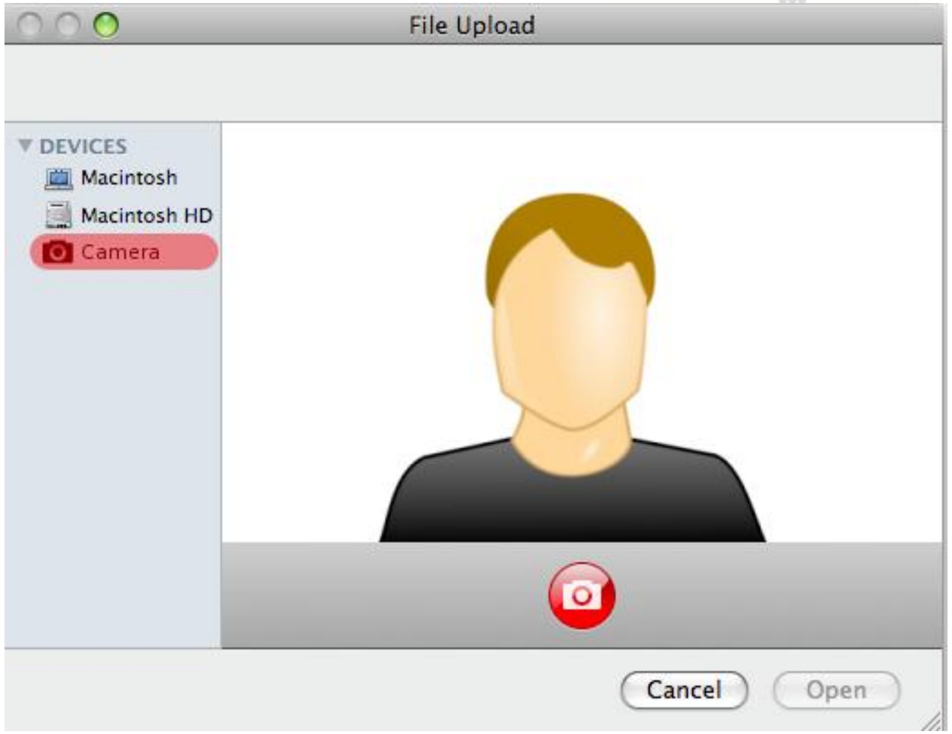
Capture attribute

- Input tags now have the capture attribute
  - Works with attribute type set to “file”
  - Allows to take picture with a camera

```
<input type="file" accept="image/*" capture="camera">
```

Capture attribute

- Rendering example



Questions ?

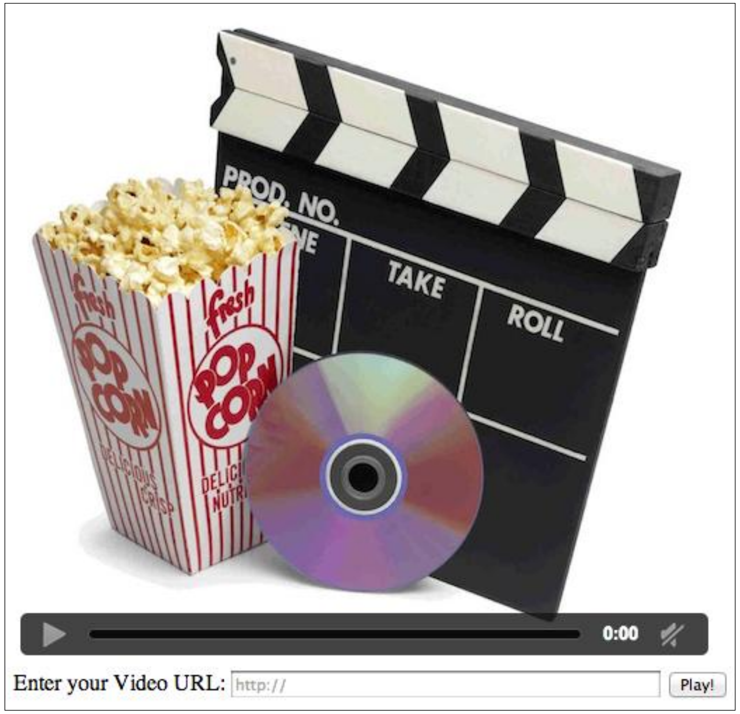
Exercise (1/2)

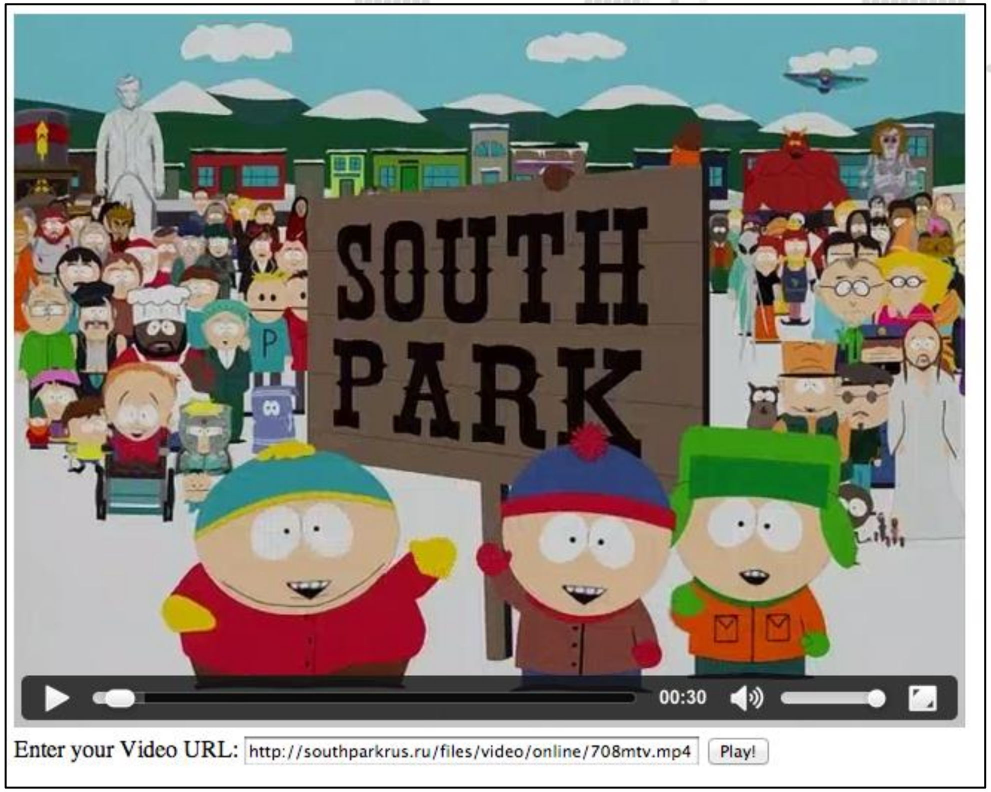
- Let’s design a simple HTML5 Media Player !

- Create a simple HTML5 page with a video element and a simple form to enter a media URL
- When the form is submit, the video element must play the media

Media Markups

Exercise (2/2)





HTML5 - New markups

ARIA

Introduction to widgets

- HTML has native support for only a few user interface controls
  - Links and form elements
- More and more webapps use widgets
  - More complex controls
  - Combination of HTML elements and script

Introduction to widgets

- A control have:
  - – A role (purpose)
  - – Properties (state)

- A script can change them on native HTML elements in order to create custom widgets

Introduction to widgets

- For example, look at the following HTML code :

```html
<div id="tb-1" class="toolbar">
   <button type="button">Print</button>
   <button type="button">Move</button>
   <button type="button">Delete</button>
</div>
```

- With some styles and scripts, we can transform those elements into a widget

Introduction to widgets

- Other example:

```
<ul class="tree">
   <li class="treeitem">Item 1</li>
   <li class="presentation">
      <a href="#group-1" class="treeitem">Item 2</a>
      <ul class="group">
         <li class="treeitem">Item 2-1</li>
         <li class="treeitem">Item 2-2</li>
      </ul>
   </li>
</ul>
```

Widget & Accessibility

- We saw how to write HTML by using the proper semantic elements and attributes
  - Necessary in order to expose element semantics to assistive technology

- But how to exposed widgets to these technology ?

Widget & Accessibility

- W3C addressed this issue in **W**eb **C**ontent **A**ccessibility **G**uidelines (WCAG) 2.0
  - – With the ARIA specification !

Accessibility with ARIA

- **ARIA** for **A**ccessible **R**ich **I**nternet **A**pplications

- W3C specification to improve the accessibility of custom widgets

- Focus on dynamic content and UI components
  – Ajax, HTML, JavaScript and related technologies

Accessibility with ARIA

- Provides additional semantics beyond what is available in current implementations of HTML

- Bridging technology filling gaps between versions of the HTML specs
  - Neither HTML 4 or 5 provide a toolbar or dialog widget, but both can be declared using ARIA

Role attribute

- For example, our previous toolbar can be declared like that:

```
<div id="tb-1" role="toolbar">
  <button type="button">Print</button>
  <button type="button">Move</button>
  <button type="button">Delete</button>
</div>
```

Role attribute

- The *role* attribute document the purpose of the control
  - Can only be set to one of several predefined values

- When building custom widgets always start with the closest native semantics and supplement with ARIA

Role attribute: Widget Roles

- The following roles act as standalone UI widgets or as part of composite widgets :

- alert
- alertdialog
- button
- checkbox
- dialog
- gridcell
- link
- marquee
- menuitem
- menuitemcheckbox
- menuitemradio
- option
- progressbar
- radio
- scrollbar
- slider
- spinbutton
- status
- tab
- tabpanel
- textbox
- timer
- tooltip
- treeitem

Role attribute: Document Structure

- The following roles describe structures that organize content in a page
  - Document structures are not usually interactive

- article
- columnheader
- definition
- directory
- document
- group
- heading
- img
- list
- listitem
- math
- note
- presentation
- region
- row
- rowheader
- separator
- toolbar

Role attribute

- Other example:

```
<ul role="tree">
   <li role="treeitem">Item 1</li>
   <li role="presentation">
      <a href="#group-1" role="treeitem">Item 2</a>
      <ul role="group">
         <li role="treeitem">Item 2-1</li>
         <li role="treeitem">Item 2-2</li>
      </ul>
   </li>
</ul>
```

Other example

```
<figure role="img" aria-labelledby="fish-caption">
   <pre>
        o           .'`/
            '      /  (
          o    .-'` ` `'-._      .')
             _/ (o)        '.  .' /
             )       )))     ><  <
             `\  |_\      _.'  '. \
               '-._  _ .-'       '.)
           jgs     `\__\
    </pre>
    <figcaption id="fish-caption">
        Joan G. Stark, "<cite>fish</cite>".
   </figcaption>
 </figure>
```

ARIA attributes

- As you can see in the last example, ARIA widgets can have attributes
- ARIA attributes are always prefixed by *"aria-"*

```
<ul role="tree" aria-multiselectable="true">...</ul>
```

ARIA attributes

- ARIA attributes can be :
  - States
  - Properties
- Some available attributes:

- aria-autocomplete
- aria-checked
- aria-describedby
- aria-disabled
- aria-haspopup
- aria-label
- aria-labelledby
- aria-selected
- …

ARIA attributes

- Three ways to label a widget :
  - <label> element

```
<label for="male">Male</label>
<input type="radio" name="sex" id="male" />
```

ARIA attributes

- Three ways to label a widget :
  - *aria-label* attribute

```
<input type="checkbox" aria-label="Message 1"
             title="Click to select this message" />
```

ARIA attributes

- Three ways to label a widget :
  - *aria-labelledby* attribute

```
<div role="alertdialog" aria-labelledby="hd">
  <form>
    <fieldset>
      <legend id="hd">Confirm Action</legend>
      <p>Are you sure you want to submit this form?</p>
      <input type="button" value="OK"> ...
    </fieldset>
  </form>
</div>
```

Styling States and Properties

- ARIA widgets states and properties are unmanaged
  – Developers are responsible for defining the visual style associated with widget states and properties

- In order to do that, you can use CSS Attribute Selectors using ARIA attributes

Styling States and Properties

- Example:

```
a[aria-expanded=true] + ul[role=group] {
   display: block;
}

a[aria-expanded=false] + ul[role=group] {
   display: none;
}
```

Questions ?

HTML5 - New markups

MICRODATA

Presentation

- Specification used to nest semantics within existing content on web pages

- Used to provide a richer browsing experience for users
  - Can be extracted and processed by:
    - Search engines
    - Web crawlers
    - Browsers
    - …

Presentation

- You can label your content to describe a specific type of information
  - Articles
  - Events
  - Person information
  - …

Presentation

- Each information type describes a specific type of item
  - For example, an event has the properties :
    - Venue
    - Starting time
    - Name
    - …

Presentation

- The specification introduces simple attributes in HTML tags:

```
<div itemscope itemtype="http://data-vocabulary.org/Person">
   My name is <span itemprop="name">Bob Smith</span> but
   people call me <span itemprop="nickname">Smithy</span>.
   Here is my home page:
   <a href="http://www.example.com" itemprop="url">
       www.example.com
   </a>
</div>
```

Attributes

- *itemscope* :
  - An element with this attribute specified creates a new item, a group of name-value pairs
  - Elements creating new items may have an *itemtype* attribute

Attributes

- *itemtype* :
  - A valid URL of a vocabulary that describes the item and its properties context
  - A collection of commonly used (and Google Supported) vocabularies are located at:
    - http://www.schema.org

Attributes

- *itemid* :
  - Optional, indicates a unique identifier of the item

```
<dl itemscope itemtype="http://vocab.example.net/book"
        itemid="urn:isbn:0-330-34032-8">
   ...
</dl>
```

Attributes

- *itemref* :

  – Properties that are not descendants of the element with the itemscope attribute can be associated with the item using this attribute

  – Provides a list of element *itemids* with additional properties elsewhere in the document

Attributes

- The following snippets are equivalent :

```
<div itemscope>
   <p itemprop="a">1</p>
   <p itemprop="a">2</p>
   <p itemprop="b">test</p>
</dl>
```

```
<div id="x">
   <p itemprop="a">1</p>
</div>
<div itemscope itemref="x">
   <p itemprop="b">test</p>
   <p itemprop="a">2</p>
</div>
```

Microdata

Attributes

- *itemprop* :
  - Indicates that its containing tag holds the value of the specified item property
  - The properties name and value context are described by the item vocabulary

Attributes

- Example:

```
<dl itemscope itemtype="http://vocab.example.net/book"
      itemid="urn:isbn:0-330-34032-8">
  <dt>Title</dt>
  <dd itemprop="title">The Reality Dysfunction</dd>
  <dt>Author</dt>
  <dd itemprop="author">Peter F. Hamilton</dd>
</dl>
```

Rich Snippets Testing Tool

- Go to :

  http://www.google.com/webmasters/tools/richsnippets

- Test some of the provided examples

Questions ?

Exercise (1/2)

- Let’s design a simple widget: an HTML5 event manager!
  - Create a form asking some information:
    - Event:
      - Name, start date, end date
    - Location:
      - City, postal code, box number and street
  - Use HTML5 new input types & validation attributes

Exercise (2/2)

- Let's design a simple widget: an HTML5 event manager!
  - Display data inside an HTML dialog on submit
    - Be creative for the design!
    - Use microdata to each property
    - Use ARIA role and label on dialog box
  - Validate your HTML code with W3C validator and Rich Snippet Testing Tool

That's all Folks!