

Web Technologies

Lecture 3












Web forms

HTML5 forms

- A **component** of a webpage that has **form controls**

- Text fields
- Buttons
- Checkboxes
- Range controls
- Color pickers

- User **interacts** with the form providing data which is **sent** to the server for further processing
 - E.g.: returning results of a search or calculation

Support:		
	Chrome (limited)	4+
	Chrome for Android (limited)	45+
	IE (limited)	10+
	UC Browser for Android	9.9+
	Firefox (limited)	4+
	iOS Safari (limited)	4.0-4.1+
	Opera Mini	None
	Android Browser (limited)	4.4+
	Safari (limited)	4+
	Opera (limited)	15+
	IE Mobile (limited)	10+
	Edge (limited)	12+

Web Forms 2.0

- If a pattern is popular enough **migrate** it from a scripted solution to a more **declarative form**
- Example
 - The *:hover* pseudo-class in CSS
- CSS has limitations
- HTML5 introduces many **new form enhancements**
- Features were part of the WHATWG specification called Web Forms 2.0 now part of HTML5

The <form> tag

<form

method= "get"

enctype="application/x-www-form-urlencoded"

action="https://www.random.org/integers/">

</form>

- Placed **inside** the <body> tag
- **Required** attributes
 - **method**
 - Get (for querying data) or post (for sending data, e.g., a file)
 - **action**
 - URL of the service handling the submitted data
- **Optional enctype**
 - **application/x-www-form-urlencoded** (default)
 - All characters encoded before being sent (e.g., spaces are converted to + characters and special characters to ASCII HEX)
 - » & → & (&);, “ → " (")
 - **multipart/form-data**
 - No characters are encoded
 - Used for file uploads
 - **text/plain**
 - Spaces are encoded, special characters not

Form content

- Contains **controls**
 - Many are represented by `<input>` elements
- Controls are **labeled** with the `<label>` tag
- Each part of a form is considered a **paragraph** and is separated by the rest by using `<p>` elements

```
<form ...>
```

```
  <p><label>Number of requested integers: <input></label></p>
```

```
</form>
```

Input types

- Specified via the **type** attribute

hidden (HTML 4)

text (HTML 4)

search

tel

url

email

password (HTML 4)

datetime

date

month

week

time

datetime-local

number

range

color

checkbox (HTML 4)

radio (HTML 4)

file

submit (HTML 4)

image






reset (HTML 4)

button (HTML 4)

Browser compatibility

- *Not all browsers support all input types*
- Check compatibility at <https://html.spec.whatwg.org/multipage/forms.html>
- The browser will **only** retain the type value you set if it supports that input type
- **Otherwise**, it will **ignore** the value you set and leave the type property as "text"
 - In this case, javascript handling of the value is required

Input restrictions

Attribute	Description
disabled	Specifies that an input field should be disabled
max	 Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	 Specifies the minimum value for an input field
pattern	 Specifies a regular expression to check the input value against
readonly	Specifies that an input field is read only (cannot be changed)
required	 Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	 Specifies the legal number intervals for an input field
value	Specifies the default value for an input field

Other form controls

- **<textarea>** tag
 - Specifies a multi-row text field

```
<textarea>At w3schools.com you will learn how to make a website. We  
offer free tutorials in all web development technologies.  
</textarea>
```
- **<option>**
 - Allows to setup a dropdown list with options

```
<select>  
  <option value="volvo" selected>Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="opel">Opel</option>  
  <option value="audi">Audi</option>  
</select>
```

New HTML5 attributes

- **Placeholder**
 - When no input is entered display default text
 - Grayed
 - Disappears when clicked on the input
- **Autofocus**
 - Focus on a particular form field
- **Required**
 - Marks a field as required
 - Requires javascript to check for its existence
- **Autocomplete**
 - Lets forms forget about auto-filling a field
 - Good in cases where you do not want other users to see some of your input such as bank accounts, etc.

Listing predefined options

- **<datalist>** tag
 - crossbreed between <input >and <select>

```
<input type="range" name="a" list="a-values">
```

```
<datalist id="a-values">
```

```
  <option value="10" label="Low">
```

```
  value="90" label="High">
```

```
</datalist>
```



Going into the future

- Browser vendors competing on the prettiness and usability of their HTML5 form controls
 - Should web developers style them instead?
 - Already some controls such as calendars and sliders cannot be styled using CSS

Data requests

- **Get**

- Requests data from a specified resource
- The query string is **sent in the URL of the request**

`https://www.random.org/integers/?`

`num=10&min=1&max=6&col=1&base=10&format=plain&rnd=new`

- Used by the REST (REpresentational State Transfer) architecture

- **Post**

- Submits data to be processed to a specified resource
- The query string is sent in the HTTP body of the request

`POST /test/demo_form.asp HTTP/1.1`

`Host: w3schools.com`

`name1=value1&name2=value2`

Get

- GET requests **can** be cached
- GET requests **remain** in the browser history
- GET requests **can** be bookmarked
- GET requests **should never** be used when dealing with sensitive data
- GET requests **have** length restrictions
 - Depends on implementation
- GET requests **should** be used only to retrieve data

Post

- POST requests **are never** cached
- POST requests **do not remain** in the browser history
- POST requests **cannot** be bookmarked
- POST requests **have no** restrictions on data length

Get vs. post

	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL

What's next?

- XML & XHTML
- Javascript
 - State vs. stateless
- Dynamic HTML manipulation
- AJAX
 - Synchronous vs. asynchronous
- JQUERY
- Server side programming