W04 - Forms

- Form Controls • Accessing form elements • Form properties/metho
- Form elements • Submitting a form • Retrieving and changing values
- Validation

<form> - contains form controls (input field, select menus, buttons)

- More common to pre-process information using javascript before sending to server.

Forms UX: usability and accessibility considerations
"Designing UX: Forms by Jessica Enders

A SEARCHING EXAMPLE

document.forms returns a collection that can be accessed using index notation [i]:

```
const form = document.forms[0];
```

or we can use getElementByTag name:

```
const form = document.getElementByTagName('form')[0];
```

or we can use the name attribute to identify the form:

```
const form = document.forms.search; ← not recommended
const form = document.forms['search'];
```

Form Object also has a method (elements) that returns an HTML Collection of all elements contained in the form:

```
const [input, button] = form.elements;
```

can also access form controls using name attributes

```
Const input = form.searchInput
```

FORM PROPERTIES AND METHODS

form.submit(); ← submits form
form.reset(); ← resets form controls back to original state
button with type attribute 'reset' will also do this without additional scripting. ← may not be best practice, though

W04 - Forms

form.action ← used to set the action attribute of a form:
form.action = '/an/other.url';

FORM EVENTS

focus ← when an element is focused on or selected.
blur ← when user moves focus away from the element.
change ← when user moves focus away from element after changing it.

SUBMITTING a FORM

submit ← occurs when the form is submitted
usually sends form data to server but can be intecepted

RETRIEVING AND CHANGING VALUES FROM A FORM

text input elements have a value property
- retrieve or change value

```
input.value = 'Search Here';
if(input.value === 'Search Here'){ input.value='';}
```

This can also be done using placeholder attribute in HTML

```
<input type='text' name='search-box' placeholder='Search Here
```

FORM CONTROLS

- <input> fields (text, passwords) check boxes, radio buttons, file uploads
- <select> menus, drop-down lists of options
- <textarea> longer text entry
- <button> submission and reset, etc.

autofocus attribute ← give focus on page load
document.forms.hero.heroname.focus(); // JavaScript equiv
maxlength attribute ← limits number of characters
form elements: sitepoint.com/html5-forms-markup

W04- Forms

makeHero function creates an object using the form information and also prevents the form from being sent to the server.

JSON.stringify( ) // used to convert the object into a JSON string

INPUT FIELDS - most common type of form control.

Text Input Fields - used for entering a short piece of information.

type='text' attribute is not required but it is good to use to signify intent.

value='10' attribute to pre-fill recommended value

Password Input Fields

type='password' ← conceals characters, everything else is treated just like 'text'

Checkbox Input Fields

type='checkbox' ← checked == true, unchecked == false
user can select more than one.

- use same 'name' property makes them accessed as an HTML collection (form.powers;)
- Iterate over collection using a for loop.
- can preset a checkbox to checked
  document.forms.hero.powers[0].checked = true;

Radio Button Input Fields

type='radio' ← allow user to check a single option as true

- use same 'name' attribute, used to group them
- allows us to assign a category ~~property~~.
- stored in form.category.value
- can have a preset
  <input type='radio' name='type' value='villain' checked>

Hidden Input Fields - type='hidden' - not displayed in browser but can contain information submitted with form
-do not use for sensitive data

W04 - Forms

File Input Fields - type='file' used to upload files

Other Input Types

many such as number, tel, color, etc.

'number' - has optional attribute - min/max/step

SELECT DROP-DOWN LIST - used to select one or more options from a list of values

- Selected attribute will set an initial value

  <option value='' selected>Choose a value </option>

- name attribute is used to access in JavaScript as a property of the form object
  - if only one item is selected, it will return a reference to that selection. Otherwise a collection containing each selection.
  - each selection object has a value property equal to the value attribute of the <option> tag
  - SelectedIndex property returns the index value of selected items.

Text Areas - <textarea> - enter long pieces of information over multiple lines - comments, blog, etc.

- Access using 'name' and 'value' properties

Buttons

<button type='reset'>Reset</button> ← not recommended

<button type='button'>Click Me</button> ← creates a clickable button

FORM VALIDATION - Process of checking information is entered correctly

- Required Fields
- Email Addresses
- Numbers
- Passwords

client-side and serverside validation

JavaScript Validation should be used to enhance user experience

Server-side validation for security

Some Validation capabilities in HTML5 (i.e. required attribute)

W04 - Forms

Instant feedback provides a better user experience than an alert after clicking / tapping submit.

→ set an event listener directly to the input field using the keyup event the feedback can be inserted inside the label element of that field with a class of error for more direct feedback.

Disabling Submit Button

```
<button type='submit' id='submit' disabled>Submit</button>
```

adding the disabled attribute will disable the button. This can be altered programmatically.

using a JavaScript function to enable/disable the button based on the field's content:

```
if (event.target.value === '') {
  document.getElementById('submit').disabled = true;
} else {
  document.getElementById('submit').disabled = false;
  }
}
```