HTML5. FORMS

Introduction

With HTML5, forms, evolved considerably. To achieve this, Web developers relied on many popular JavaScript frameworks for validating input formats, providing various input GUIs, such as calendars for dates, sliders, etc. Frameworks such as jQueryUI, Dojo, and Sencha, all provide a widget set for improving forms. Furthermore, it was time to take into account the specifics of mobile web applications, where the GUI of a date chooser cannot be the same as a 400x400 pixel wide calendar on a desktop. Contextual virtual keyboards provided the way forward on smartphones and tablets thanks to Apple, Google and others.

HTML5 took all this into account and thus provides:

- A set of input fields that include a validation API and visual feedback, contextualized keyboards, etc. Of course the look and feel depends on the web browser's implementations, but the HTML5 forms specification introduced **13 new <input type**=.../> fields: email, tel, color, url, date, datetime, datetime-local, month, week, time, range, number and search.
- Built-in validation system: JavaScript API for custom validation, CSS pseudo classes that are useful for changing an input field style depending on the validity of the input.
- Other goodies, such as the option to set an input field out of a <form>, new elements such as <datalist> for autocompletion, <output> for feedback, etc.

Ejemplo: https://codepen.io/paqui-molina/pen/xxLVMeZ

Forms. Client-side part.

Forms are a way to get user input which is sent to a remote server. This section focuses on the HTML5 additions to forms, and as such will only cover the client-side part.

On the server side, you may have PHP, Java, C#, Ruby, Python, etc. components. There are several ways to collect server-side data from a form in a Web page: REST Web services, servlets, Microsoft ASP pages, etc.

On the client side, the forms indicate **to which server** and **how the data should be sent**, using the **action and method attributes** respectively. A <**button type="submit"**> or an <input type=submit> field is used to submit the form content.

For example:

<form action="myServerCode.php" method="POST">...</form>. Here, we set the URL of the server side code (myServerCode.php), and the HTTP method that will be used by the browser for sending the form content (POST).

Another approach is to use JavaScript for sending the form content with Ajax.

Ejemplo: https://codepen.io/paqui-molina/pen/MWvyLMM

How to structure a web form?

https://developer.mozilla.org/en-US/docs/Learn/Forms/How_to_structure_a_web_form

Accesible forms

Forms are commonly used to enable user interaction in Web sites and Web applications. For example, they are used for login, registering, commenting, and purchasing.

Since HTML5 provides functionalities to assist with accessibility, developers should make a concerted effort to mark up Web based forms. The following two guidelines are to give you a good start to make your forms accessible:

- For every form field, ensure that a descriptive label is provided and use the <label> element to identify each form control.
- For larger or complex forms, use the <fieldset> and <legend> elements to respectively group and associate related form controls.

Web accessibility tutorials: https://www.w3.org/WAI/tutorials/forms/

Why is this important?

Accessible forms are easier to use for everyone, including people with disabilities.

- People with cognitive disabilities can better understand the form and how to complete it, as making forms accessible improves the layout structure, instructions, and feedback.
- People using speech input can use the labels via voice commands to activate controls and move the focus to the fields that they need to complete.
- People with limited dexterity benefit from large clickable areas that include the labels, especially for smaller controls, such as radio buttons and checkboxes.
- People using screen readers can identify and understand form controls more easily because they are associated with labels, field sets, and other structural elements.

Labeling controls

Labels need to describe the purpose of the form control.

When these labels are marked up correctly, people can interact with them using only the keyboard, using voice input, and using screen readers. Also, the label itself becomes clickable, which enables a person who has difficulty clicking on small radio buttons or checkboxes to click anywhere on the label text.

Whenever possible, use the label element to explicitly associate text with form elements.
The for attribute of the label must exactly match the id of the form control.

```
<label for="first_name">Your First Name</label>
<input id="first_name" type="text" name="fname"/>
```

https://codepen.io/paqui-molina/pen/eYEzJMy

Grouping controls

Groupings of form controls, typically groups of related checkboxes and radio buttons, sometimes require a higher level description. Grouping related form controls makes forms more understandable for all users, as related controls are easier to identify.

La agrupación debe realizarse visualmente y en el código, por ejemplo, utilizando los elementos <fieldset> y <legend> para asociar controles de formulario relacionados. El <fieldset> identifica la agrupación completa y la <legend> identifica el texto descriptivo de la agrupación.

Ejemplo: https://codepen.io/paqui-molina/pen/OJjXNgg

Input types

Color

https://codepen.io/paqui-molina/pen/gOxMrQG

Date

https://codepen.io/paqui-molina/pen/rNzLMVm?editors=1010

Number

This input field is useful for entering numerical values (integer or float), but not for entering zip codes.

For zip codes, a <input type="text" pattern="....."> is preferable.

https://codepen.io/paqui-molina/pen/xxLOERW

Attributes accepted: max, min, step, value

This input type is very interesting as it provides default validation behaviors:

- **If** the value entered using a keyboard is not a valid number, or is not in the range defined by the min and max attributes**, the field is invalid and gets the pseudo CSS class :invalid**.
- **If the difference between the value you enter and min is a multiple of step, then it gets the CSS pseudo class :valid** , otherwise it will be invalid. Example: if min=1 and step=5, the field will be valid with value=1, 6, 11, 16 etc. if min=0, with value=0, 5, 10, 15 etc.

Range

This input type renders as a slider. It accepts the same attributes as the <input type="number"> : min, max, step and value.

The basic use is to specify at least the value, min and max attributes, and eventually the step attribute, too:

But most of the time, you will need a visual feedback that shows the current value selected by the slider.

Example:

https://codepen.io/paqui-molina/pen/bGrgOMz

email, tel, URL and search

<input type=”email”>

This input type is easy to use. In mobile applications, this new input type pops up a keyboard layout adapted to email input. Note the "@" key, the "." key, etc.

This input type is very interesting as it provides **default validation behaviors**:

- If the value entered looks like an email address (contains a "@"...), the field is valid, and gets the pseudo CSS class :valid
- If the value entered does not contain an "@", and does not look like an email address, the field is invalid and gets the pseudo CSS class :invalid

https://codepen.io/paqui-molina/pen/ExvZraX

<input type=”tel”>

This input field is useful on smartphones and tablets, as it makes the browser pop up a keyboard layout suitable for entering phone numbers.

<input> elements of type tel are used to let the user enter and edit a telephone number. Unlike <input type="email"> and <input type="url"> , the input value is not automatically validated to a particular format before the form can be submitted, because formats for telephone numbers vary so much around the world.This input type is often used with the new **placeholder** and **pattern attributes**. It is supported by all recent major Web browsers, on mobile devices and desktops.

https://codepen.io/paqui-molina/pen/MWvJLJb

<input type=”URL”>

This input field is useful on smartphones and tablets, as it makes the browser pop up a keyboard layout suitable for entering URLs

Form attributes

Form

This attribute is useful for putting input fields outside the form itself. The form attribute of an external input field must share the same value as the id of the form the field belongs to. This is useful when using <fieldset> elements for making the page/form layout easier.

https://codepen.io/pen/?editors=1010

Pattern

The pattern attribute enables the validation of the user's input on the fly (also at submission time), based on regular expressions. It applies to the **text, search, url, tel, email, and password input types.**

https://regexone.com/

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions

más atributos: https://www.w3schools.com/html/html_form_attributes.asp

HTML5 forms elements

<output>

The output element represents the result of a computation or user action. You can see it as a "specialized or " for displaying interactive results.

attributes specific to the <output> element:

- **for:** a space-separated list containing the elements' ids whose values went into the calculation.
- **name:** the name of the element.
- **form:** associates the <output> element with its form owner. The value must be the id of a form in the same document. This allows you to place an <output> element outside of the <form> with which it is associated.

https://codepen.io/paqui-molina/pen/wvqdQrG

<meter>

The <meter> element displays colored bars to represent numeric values.

It can be useful to display a colored gauge to show disk usage, to highlight the relevance of a query result, or the fraction of a voting population that favours a particular candidate, etc. This element is often used with the <input type="range"> field as an instant feedback indicator.

https://codepen.io/paqui-molina/pen/OJjmazM

<progress>

The <progress> element is similar to <meter> but it is used for progress bars (i.e., the percentage of a file being uploaded, etc.).

The browser calculates the percentage corresponding to the value, min and max attributes and adjusts the length of the progress bar accordingly.

If no value attribute is set, the progress bar will display an "indeterminate look", that may slightly vary among different browser implementations.

https://codepen.io/paqui-molina/pen/zYdwMRa

<datalist>

https://codepen.io/paqui-molina/pen/zYdwMjL