



Forms

Introduction

Let's get this out of the way up front – forms have a bad reputation because they can be frustrating. Very few people actually *enjoy* filling in forms – most just want to get them over and done with! The moment a form appears, the user must gauge “Is this really worth my time?” “Does this one need an American State and valid zip code?”

With the amount of time and effort put into bringing users to our websites to sign up, it's surprising that we tend not to spend a proportional amount of time designing usable forms. Basically, we are gambling that the pay-off for the user will outweigh the pain of submitting the form.

You may ask yourself, why take that risk? The answer is that a well-designed, intuitive form can give a website a huge advantage in terms of user experience (UX) and, perhaps most importantly, conversion rates.

In this unit, we will learn how to build a form from the ground up. We will also take a look at many of the HTML5 features which enhance the UX, such as *autocomplete*, because the best form is one that completes itself.

Learning Outcomes

Topics include:

- What is a Form?
- Input Types
- Other Form Tags

- Exploring input Attributes
- Numbers and Patterns
- Making Choices

Topic 1: What is a Form?

HTML **Forms** collect data from visitors to a website. You will have seen all kinds of form examples, such as:

- Shopping cart
- User signup
- Inputting an e-mail in webmail
- Newsletter subscription
- User comments, and of course,
- Online application forms!

The Basics

The most basic form follows the following format:

1. An opening **form** tag.
2. A list of **input** types, followed by a **submit**.
3. A closing **form** tag.

The example below would be a typical **pre-HTML5** form layout:

```
1. <form id="UserRegForm" action="server_side_script.php"
   method="post">
2.   Firstname:<input type="text" name="fname"/>
3.   Surname:<input type="text" name="sname"/>
4.   Email:<input type="text" name="email"/>
5.   <input type="submit" value="submit"/>
6. </form>
```

Attributes

The attributes assigned to each tag determines how the form operates. Let's take a look at the important attributes:

Tag	Attribute	Description
form	action	The action attribute determines what happens to the form information when you hit the submit button. Usually, a back end script will be the target and the script will insert the data into a database.
	method	This attribute determines how the action is performed (get or post), we don't have to worry about these yet as we are all about the Front end for now.
input	type	The input tag requires a number of attributes, the most important of which is type . The example above uses two types, text and submit . There are many others that we will discuss below.
	id	The id attribute is not just for styling but also to access an element in a form i.e. labelling (again, we will discuss this a little later).
	value	The value attribute sets a default value for an input . The value attribute is required for some input types i.e. checkboxes, radio buttons.

Topic 2: Input Types

The most basic input types are **text** and **submit**, and they have been around as long as the **form** tag itself. The **text** type anticipates user text input and the **submit** type creates a submit button. Notice, that in the implementation example below, the **input** tag is self closing.

Text Type

```
Firstname:<input type="text" name="fname"/>
```

First Name:

Submit

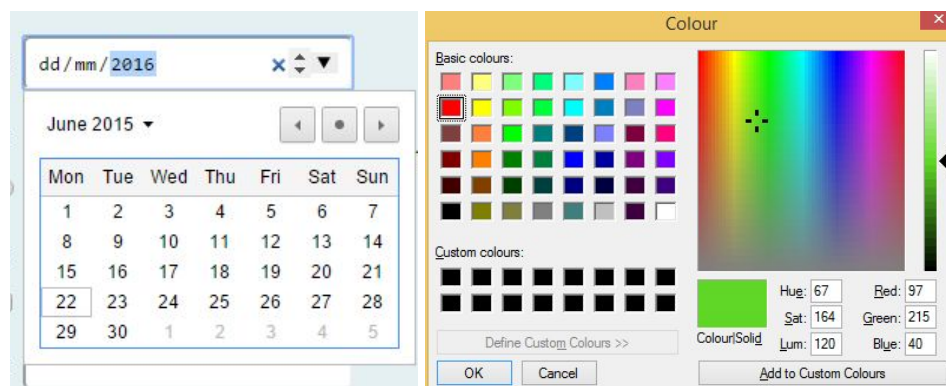
```
<input type="submit" value="submit"/>
```

submit

Other Input Types

One of the many gifts HTML5 has bestowed upon us is an abundance of shiny new input types to play with.

Choosing the right input type is important, especially for mobile devices. For example, if I choose a **number** type, the keyboard layout can change to a number pad on a touch screen. Similarly, an **email** type can display the @ symbol which enhances the user experience. There are even new fancy **date** and **color** pickers!



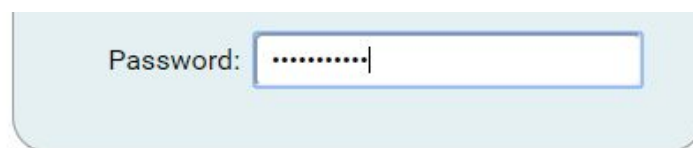
The below list demonstrates the most common input types:

- text
- number
- password
- url
- email
- search
- tel (telephone)
- date
- color

Now, let us look at some of these input types and their implementations:

Password Type

The **password** type is like a standard textbox but the characters are replaced with dots for security.



```
1. <input id="password" name="password" type="password" />
```

Email Type

The **email** type is also like a textbox. However, it anticipates an email input and will “complain” if the email format is incorrect.

```
1. <input id="email" name="email" type="email"/>
```

URL Type

The **url** type anticipates a URL input (similar to the **email** type).

Note:

You may be wondering why we included a **name** attribute above and how it is different to id attribute. The name attribute get passed with the form information to the server and is used as a reference to that particular field. The is used for CSS styling and from JavaScript to reference.

Note:

Some of the browsers are slow to implement some of the latest input types. For naming and shaming purposes, and as at the time of writing, Firefox and IE do not support the date input type. The color input type is also not supported in IE.

Challenge 1

Forms 1

Open **form.html** (located in the files folder), the file contains some CSS to help with the form layout.

Add the following elements and attributes:

- <form>

- FirstName
- Input type with a for email
- Input type with a for website (URL)
- Password
- Submit button



First Name:

Password:

Email:

Website:

Topic 3: Other Form Tags

By now, you should be able to create a basic form. So, let's kick it up a notch with some additional tags that will make your form look and work better.

Label

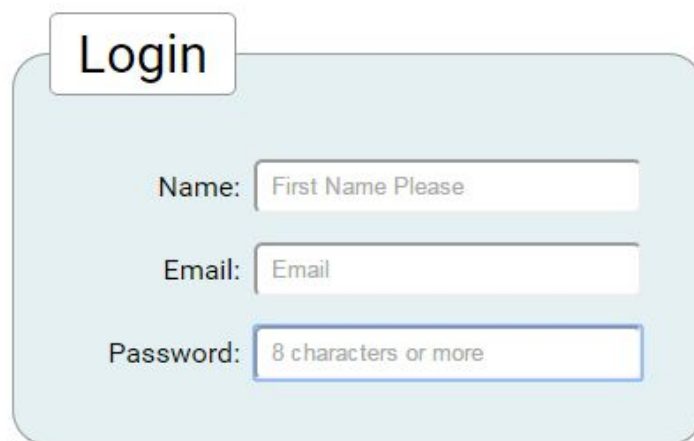
The **label** element associates a label with a particular **input** element; in our case an **input** field. It uses the **for** attribute to link the label to the **id** of an element. This is crucial for creating accessible sites as it is read by screen readers for visually

impaired users. **Labelling** is important for mobile devices as the user can touch the label to enter the associated text field.

```
<label for="userName"> Username:</label>
<input type="text" id="userName" name="userName"/>
```

Fieldset

The **fieldset** tags can group a number of inputs together in some logical fashion. It also places a border around the group by default.



```
<fieldset>
  <!-- set of related fields -->
</fieldset/>
```

Legend

The **legend** element gives a title to your fieldset. In our example, we have used the “Login” legend.

```
<fieldset>
  <legend> Login </legend>
  <!-- set of related fields -->
```



```
</fieldset/>
```

Challenge 2

In this challenge you will use some additional input types.

1. Continuing from **Challenge 1** add a new fieldset with the following elements and attributes:
 - ☐ Give labels to all input fields (for each label give a class of *field* for styling)
 - ☐ Add a Fieldset to the group
 - ☐ Add Legend (Registration)
 - ☐ Add Telephone field.
2. Test your page on the major browsers.

Personal Details

Email:

Telephone:

Website:

Topic 4: Exploring input Attributes

HTML5 also comes with a selection of new attributes. Let us look at a number of them now.

Placeholder

The **placeholder** attribute allows you to place temporary text within a *textbox* or *textarea*. This is useful for when you need to describe the information to be inputted into the text field.

```
<input id="email" name="email" type="email" placeholder="Enter your  
Email here"/>
```

Login Details

First Name:

Password:

Email:

Website:

Note:

The **placeholder** text should not be used as a replacement for **label** elements. When the user starts typing, the **placeholder** will disappear. Instead, the **placeholder** text should be used as a guide for the field's content.

Autocomplete

This is one of the most important HTML5 features in terms of usability. The **autocomplete** attribute remembers previously completed forms in the browser and magically pre-fills certain form fields, forever more.

```
<input id="name" name="name" type="text" autocomplete = "on"/>
```

Required

The **required** attribute does exactly what it says on the tin. The field must have a value before the form can be submitted. However, beware that required fields look exactly the same as ordinary fields, so it may be wise to supplement required fields with some visual cue (an asterisk * or **placeholder** text).

```
<input type="text" name="name" required>
```

Autofocus

The **autofocus** attribute allows you to assign which input should receive the focus when the page loads, i.e., the cursor will automatically appear in the textbox.

```
<input id="email" name="email" type="email" autofocus  
placeholder="Email"/>
```

Challenge 3

3. Continuing from **Challenge 2** add the following elements and attributes:
 - ☐ Required on firstname & email
 - ☐ autofocus on email
 - ☐ Add placeholder text to all appropriate fields
4. Test your page on the major browsers.

Topic 5: Numbers and Patterns

Until now, we have been able to create input restrictions using the **required** attribute and the **email** and **url** types. However, you can make your forms more user friendly by utilizing some more complex restrictions.

Max, Min

If we are expecting a number input and it is required to be within a certain range, we can define minimum and maximum values. This can prevent users inputting negative numbers or other invalid inputs.

```
1. <input type="number" min="0" max="120" name="age"/>
```

Pattern

With **pattern**, we can specify exactly what we expect the user to input in a field. For example a social security number could start with 5 numbers followed by a letter. If the user enters the number incorrectly, the field is considered invalid and they can be asked to try again.

Patterns work with the following inputs:

- text
- Search
- url
- tel
- email
- password

This is a big step forward for input validation. Previously Javascript would be needed to achieve this. The possibilities for **pattern** are wide-ranging (e.g. passport numbers, product codes, invoice numbers). Let us look at how **pattern** is implemented, followed by the syntax.

Example 1: Social Security Number

```
1. <input name="social security number" type="text"
   pattern="[0-9][A-Z]{3}" title="Single number followed by three
   Uppercase letters."/>
```

Valid input: **1ABC**

Example 2: Passport

```
1. <input name="passport" type="text" pattern="[0-9]{7}[A-Za-z]{1}"
   title="Seven numbers followed by a single upper or lower case
   letter."/>
```

Valid input: 1234567A

The Syntax

- [0-9] Number from 0-9
- [A-Z] Uppercase Letter
- [a-z] Lowercase Letter
- [A-Za-z] Uppercase or Lowercase
- {1} a single instance of
- {3} sequence of three

Use the **title** attribute to describe what pattern you expect, It will aid the user if do not follow the pattern . See example of title use below.

Personal Details

Email:

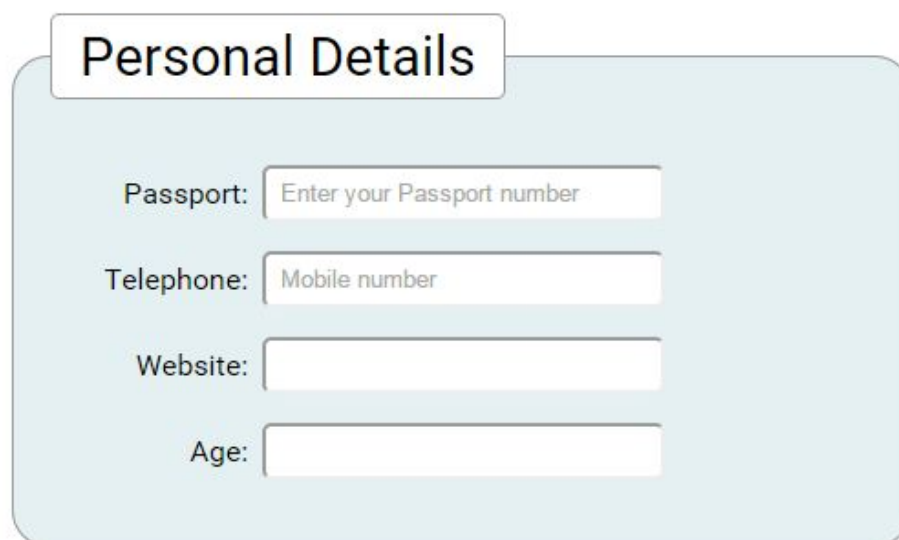
Telephone:

Website:

! Please match the format requested.
Mobile telephone number expected beginning 08*****

Challenge 4: Using Numbers and Patterns

1. Continuing from **Challenge 3**, add the following elements and attributes:
 - Passport number (with correct pattern for Passport)
 - Add pattern to Telephone number (correct pattern for Mobile phone)
 - Add input for an [eircode](#) field with pattern
 - Add input for credit card field.
 - Age (with Max and Min)
2. Add placeholder text to all appropriate fields.



Personal Details

Passport:

Telephone:

Website:

Age:

Range

There is now an input type for a **range** of values. This is ideal for volume control or seeking through video. The range takes two attributes, min and max.

```
<input type="range" min="0" max="100"/>
```

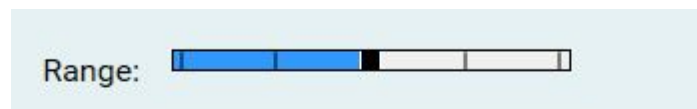
Depending on browser support, the **range** can be displayed as a slider control.



chrome



firefox



Internet Explorer

Topic 6: Making Choices

Forms are not limited to text boxes and submit buttons. There are various ways to get users to select from predetermined lists. Let's explore some of these.

Checkboxes

Using **checkboxes** is a good option when you want to give your visitors the choice of one or more options. The **type** attribute must have the value "**checkbox**" and then you have to make sure that every checkbox within the same group have the same name.

Gem Stones:

- ☒ Sapphire
- ☐ Jade
- ☒ Smoky Quartz
- ☐ Emerald
- ☐ Ruby

1. `<input type="checkbox" name='stone' value="Sapphire"/>Sapphire`
2. `<input type="checkbox" name='stone' value="Jade"/>Jade`
3. `<input type="checkbox" name='stone' value="Quartz"/>Quartz`
4. `<input type="checkbox" name='stone' value="Emerald"/>Emerald`
5. `<input type="checkbox" name='stone' value="Ruby"/>Ruby`

Radiobuttons

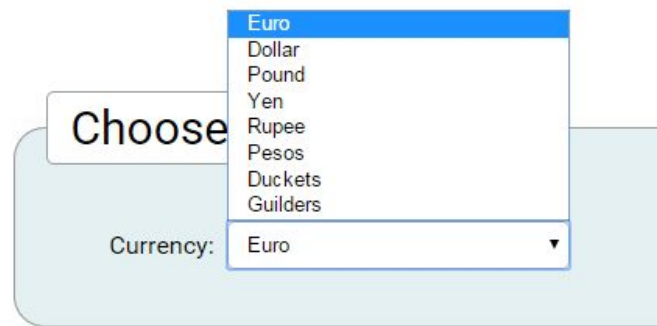
Radiobuttons are similar to checkboxes and are used when you want to present a lot of options to the user, but at the same time makes sure they choose just one.

☐ Tea
☒ Lemon Tea
☐ Nettle Tea

6. `<input type="radio" name='tea' value="Tea"/>Tea`
7. `<input type="radio" name='tea' value="Lemon Tea"/>Lemon Tea`
8. `<input type="radio" name='tea' value="Nettle Tea"/>Nettle Tea`

Select Box

Select boxes have been around almost forever and are still very popular.



```

1. <label for="currency">Currency:</label>
2. <select id="currency">
3.     <option value="euro">Euro</option>
4.     <option value="dollar">Dollar</option>
5.     <option value="pound">Pound</option>
6.     <option value="yen">Yen</option>
7.     <option value="rupee">Rupee</option>
8.     <option value="pesos">Pesos</option>
9.     <option value="duckets">Duckets</option>
10.    <option value="guilders">Guilders</option>
11. </select>

```

Option Groups

The `<optgroup>` tag allows grouping the options within the select box.



```

<optgroup label="Popular">
  <option value="euro">Euro</option>

```

```
<option value="dollar">Dollar</option>
<option value="pound">Pound</option>
<option value="yen">Yen</option>
</optgroup>

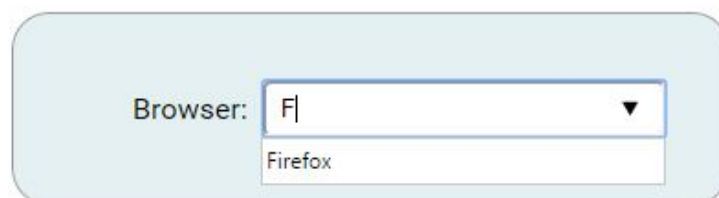
<optgroup label="more">
  <option value="rupee">Rupee</option>
  <option value="pesos">Pesos</option>
  <option value="duckets">Duckets</option>
  <option value="guilders">Guilders</option>
</optgroup>
```

List & DataList

So what is a **DataList** and how is it different from a **select box**?

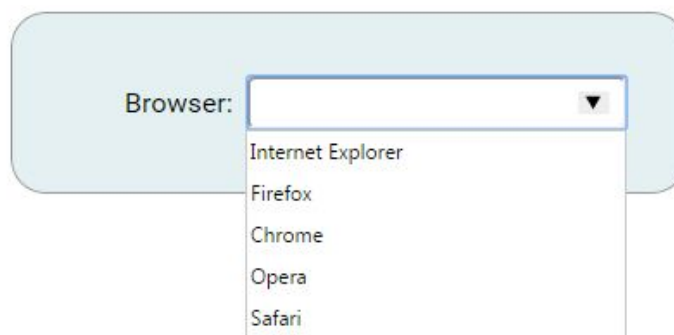
DataLists allow the user to type their own input but also have some predefined values to ease the heavy burden. When the user types, they will receive suggestions to complete the field.

The **list** attribute enables the user to associate a list of options with a particular field. The value of the list attribute must be the same as the **ID** of a **datalist** element that resides in the same document.



Browser: ▼

Firefox



Browser: ▼

Internet Explorer
Firefox
Chrome
Opera
Safari

```
1. <input list="browsers" name="browser">
2.   <datalist id="browsers">
3.     <option value="Internet Explorer">
4.     <option value="Firefox">
5.     <option value="Chrome">
6.     <option value="Opera">
7.     <option value="Safari">
8.   </datalist>
```

Challenge 5

- Continuing from Challenge 4, add the following elements and attributes:
 - ☐ Date
 - ☐ Checkbox (Would you like: Starter, Main , Dessert)- include labels
 - ☐ **note:** remove **class=field** from the label tags for checkbox
 - ☐ Radiobuttons (Gender: Male, Female, Other) - include labels
 - ☐ **note:** remove **class=field** from the label tags for radiobuttons
 - ☐ Select Box with Option Group (i.e. Signs of the Zodiac - sun, earth, fire, water)
 - ☐ DataList (i.e Music Genres)
 - ☐ Range for with minimum value 0 maximum 11.
- Add placeholder text to all appropriate fields.

Review your form in different browsers.

Validate your Code with the [Validator](#)

Summary

The importance of forms is often easily overlooked. In many cases, forms are the most interactive part of a website.

However, forms up until now, can be frustrating for the user. However, lately with the progression of HTML5, there has been a dramatic leap forward in form design.

In this unit, we have looked at form creation including some of the new HTML5 form attributes, which help improve user experience and save on development time.

See more on forms at <http://www.html5-tutorials.org/forms/>

AUTOMATTIC

1. Home2. About Us3. News4. Work With Us

Digital Millennium Copyright Act (DMCA) Notice

Form not submitted — please check the following errors:

- The value for Zip / Postal Code is too short.

First Name *	<input type="text" value="Robert"/>
Last Name *	<input type="text" value="Harte"/>
Company Name	<input type="text"/>
Address *	<input type="text" value="Dublin"/>
City *	<input type="text" value="Dublin"/>
State/Region/Province *	<input type="text" value="Dublin"/>
Country *	<input type="text" value="Ireland"/>