HTML Forms

Form: ....Web Page

Input:.............Get Data

Submit: ....Send Data

   Action.............Target server file

   Method...........Public or Private

Server:

HTML Form Example

First name:

Mickey

Last name:

Mouse

Submit

Try it Yourself »

The <form> Element

The HTML `<form>` element defines a form that is used to collect user input:

```
<form>
.
form elements
.
</form>
```

An HTML form contains **form elements**.

Form elements are different types of input elements, like text fields, checkboxes, radio buttons, submit buttons, and more.

The <input> Element

The `<input>` element is the most important form element.

The `<input>` element can be displayed in several ways, depending on the **type** attribute.

Here are some examples:

Type	Description
<input type="text">	Defines a one-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

You will learn a lot more about input types later in this tutorial.

Text Input

`<input type="text">` defines a one-line input field for **text input**:

Example

```
<form>
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```
Try it Yourself »

This is how it will look like in a browser:

First name:

Last name:

Radio Button Input

`<input type="radio">` defines a **radio button**.

Radio buttons let a user select ONE of a limited number of choices:

Example

```
<form>
  <input type="radio" name="gender" value="male" checked> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
  <input type="radio" name="gender" value="other"> Other
</form>
```
Try it Yourself »

This is how the HTML code above will be displayed in a browser:

○ Male
○ Female
○ Other

The Submit Button

`<input type="submit">` defines a button for **submitting** the form data to a **form-handler**.

The form-handler is typically a server page with a script for processing input data.

The form-handler is specified in the form's **action** attribute:

Example

```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><br>
  <input type="submit" value="Submit">
</form>
```

Try it Yourself »

This is how the HTML code above will be displayed in a browser:

First name:

Mickey

Last name:

Mouse

Submit

The Action Attribute

The `action` attribute defines the action to be performed when the form is submitted.

Normally, the form data is sent to a web page on the server when the user clicks on the submit button.

In the example above, the form data is sent to a page on the server called "/action_page.php". This page contains a server-side script that handles the form data:

```
<form action="/action_page.php">
```

If the `action` attribute is omitted, the action is set to the current page.

The Method Attribute

The `method` attribute specifies the HTTP method (**GET** or **POST**) to be used when submitting the form data:

Example

```
<form action="/action_page.php" method="get">
```
Try it Yourself »

or:

Example

```
<form action="/action_page.php" method="post">
```

When to Use GET?

The default method when submitting form data is GET.

However, when GET is used, the submitted form data will be **visible in the page address field**:

/action_page.php?firstname=Mickey&lastname=Mouse

Notes on GET:

- Appends form-data into the URL in name/value pairs
- The length of a URL is limited (2048 characters)
- Never use GET to send sensitive data! (will be visible in the URL)
- Useful for form submissions where a user wants to bookmark the result
- GET is better for non-secure data, like query strings in Google

When to Use POST?

Always use POST if the form data contains sensitive or personal information.
The POST method does not display the submitted form data in the page address

---

Reference Only!

===

HTML Input Types

Here are the different input types you can use in HTML:

- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">`
- `<input type="time">`
- `<input type="url">`

- `<input type="week">`

When to Use GET?

The default method when submitting form data is GET.

However, when GET is used, the submitted form data will be **visible in the page address field**:

/action_page.php?firstname=Mickey&lastname=Mouse

Notes on GET:

- Appends form-data into the URL in name/value pairs
- The length of a URL is limited (2048 characters)
- Never use GET to send sensitive data! (will be visible in the URL)
- Useful for form submissions where a user wants to bookmark the result
- GET is better for non-secure data, like query strings in Google

When to Use POST?

Always use POST if the form data contains sensitive or personal information. The POST method does not display the submitted form data in the page address field.

Notes on POST:

- POST has no size limitations, and can be used to send large amounts of data.
- Form submissions with POST cannot be bookmarked

The Name Attribute

Each input field **must** have a `name` attribute to be submitted.

If the `name` attribute is omitted, the data of that input field will not be sent at all.

This example will only submit the "Last name" input field:

Example

```
<form action="/action_page.php">
  First name:<br>
  <input type="text" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><br>
  <input type="submit" value="Submit">
</form>
```
Try it Yourself »