# Tutorial

## Activity 1: HTML <body> Elements

HTML <body> child elements have a number of different properties, for example:

* some have **paired** tags, others have **single / empty** tags
* some are **block** elements, others are **inline** elements
* some can be placed anywhere in <body>, others must be **nested** inside other elements.

Classify the following HTML <body> elements and identify at least one possible parent tag:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Element / Tag | Paired | Empty | Inline | Block | Parent Tag(s) |
| <a> | ✓ |  | ✓ | ✓ | <div>, <p> etc |
| <br> | ✓ | ✓ | ✓ |  |  |
| <div> |  |  |  | ✓ |  |
| <em> |  |  | ✓ |  |  |
| <form> |  |  |  | ✓ |  |
| <header> | ✓ |  |  | ✓ | <body> |
| <h5> |  |  |  | ✓ |  |
| <hr> |  | ✓ |  |  |  |
| <img> |  | ✓ | ✓ |  |  |
| <input> |  | ✓ | ✓ |  |  |
| <li> |  | ✓ |  | ✓ |  |
| <mark> |  |  | ✓ |  |  |
| <ol> |  |  |  | ✓ |  |
| <p> |  |  |  | ✓ |  |
| <span> |  |  | ✓ |  |  |
| <table> |  |  |  | ✓ |  |
| <th> |  | ✓ |  |  |  |
| <ul> | ✓ |  | ✓ | ✓ |  |

How are <div> and <span> similar? How are they different?

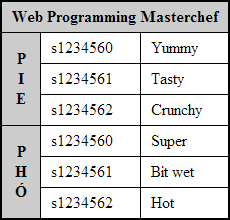
* The difference between span and div is that a span element is in-line and is used for a sentence or short paragraph whereas a div element is block-line (which is basically equivalent to having a line-break before and after it) and used to group larger chunks of code.

## Activity 2: Creating a Table.

A table element is a HTML structure made up of rows and cells that is used to lay out **tabular data** in a **grid** with neatly lined up rows and columns. Occasionally, you will want data or headings to expand to other cell spaces and you use rowspan and colspan attributes to make this happen.

The commonly used tags used to create a table are:

<table>, <tr>, <th> and <td>.



## 

## How many rows and how many columns does the table have before "cell spanning"?

* The table has 3 columns and 7 rows before cell spanning

To give a table more structure and "semantic" meaning, the following tags are often included to contain rows and cells:

<thead>, <tbody> and <tfoot>.

These tags are placed inside the <table> tag in any order, but they will always be rendered in the order above.

**Please note:** When designing a layout, it is preferable to use the **CSS Grid layout model** which has many of the benefits of a table but without the fixed structure. Please avoid using tables for layout in your website as they do not adapt well when viewed in small screens such as mobile phones etc.

Tables can still be used but only for tabular data information, but even then always be mindful of the lack of layout adaption on smaller screens.

## Activity 3: Forms and Form Submission

Forms are used to submit data to a processing script. In this course we will write processing scripts in PHP, but there are many server side languages that can process form data.

A form has the following attributes:

|  |
| --- |
| <form action="..." method="..." onsubmit="..." />  <!-- Input controls etc, will cover soon -->  </form> |

What goes in the action attribute?

* The location where the form data is to be sent to the server after submission of the form

The method attribute takes either post or get as a value. The post method puts the form data inside the body of the message, whereas get puts the form data inside the url. Discuss the benefits of each method option, think of examples where you would use one rather than the other.

* Advantage of “Get” method:

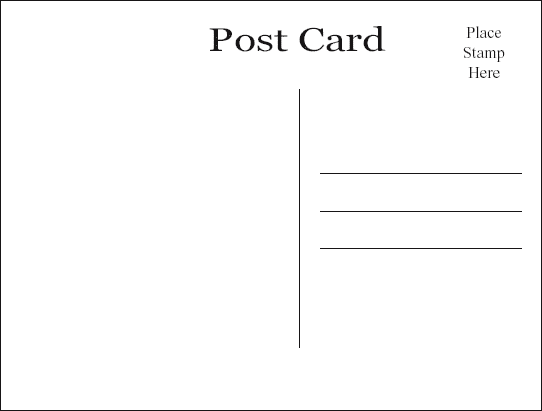
1. Since the data sent by the GET method are displayed in the URL, it is possible to bookmark the page with specific query string values.

* Advantage of “Post” method:

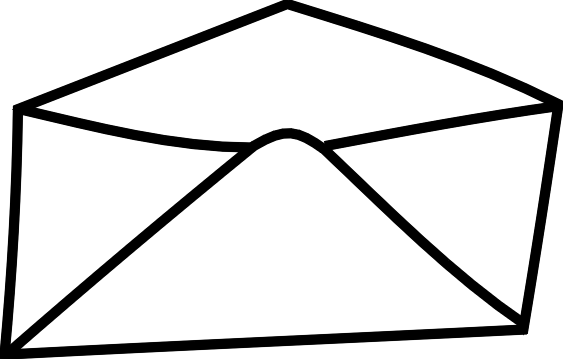
1. It is more secure than GET because user-entered information is never visible in the URL query string or in the server logs.
2. There is a much larger limit on the amount of data that can be passed and one can send text data as well as binary data (uploading a file) using POST.

* Example:

When I want to send large file with not only binary numbers but also text to explain my data, I’d use “Post” method.



GET



POST

Are there similarities with physical postcards and envelopes?

* Actually, The “Get” method is quite similar to the postcard as it has limited space to write on and also it is less discrete as anyone can read the content of the postcard while the “Post” method is quite similar to the envelope as it is more discrete as the content of the letter is hidden inside the envelop and you can have so much more space on the letter to write.

In the onsubmit attribute, if we write "return checkForm()" and this function returns false, what will happen when a user tries to submit the form?

* The form will not be submitted

## Activity 4: Form Elements

### Input elements:

What do each of the following 3 input attributes do? Why are they important:

|  |
| --- |
| <input type="..." name="..." value="..." /> |

1. **Type**: the type attribute specifies the type of <input> element to display. For embed, link, object, script, source, and style elements, the type attribute specifies the Internet media type (formerly known as MIME type).
2. **Name**: The name attribute specifies the name of an <input> element. The name attribute is used to reference elements in a JavaScript, or to reference form data after a form is submitted.
3. **Value**: The value attribute specifies the value of an <input> element. The value attribute is used differently for different input types

They are important because each of them has a specific task that overall contribute to

What is the difference between text**,** password and hidden inputs? For example:

|  |
| --- |
| Name: <input type="text" name="name" value="" /><br/>  Password: <input type="password" name="password" value="" /><br/>  Rated: <input type="hidden" name="client-rating" value="2 stars" /> |

* With the input type “text”, User can enter into it and the computer will define it as text field, while the input type “password” the characters being typed in will be masked and the computer will define it as a password field. Input type “hidden” will be define as a hidden field and will not be visible to the user. A hidden field will store database record that needs to be updated when the form is submitted.

What is the difference between radio and checkbox inputs AND why are there square brackets in the checkboxes' name attributes?

|  |
| --- |
| <!-- Radios -->  <input type="radio" name="gender" value="male" /> Male<br/>  <input type="radio" name="gender" value="female" /> Female<br/>  <input type="radio" name="gender" value="other" /> Other  <!-- Checkboxes -->  <input type="checkbox" name="seeks[]" value="male" /> Male<br/>  <input type="checkbox" name="seeks[]" value="female"/> Female<br/>  <input type="checkbox" name="seeks[]" value="other"/> Other |

* A radio input provides a single-selection user choice. It is often depicted as a empty circle (unselected) and filled circle (selected).
* A checkbox input provides a multi-selection user choice. It is often depicted as an empty square (unselected) and a checkmark within a square (selected).
* The reason for the square brackets is that it informs PHP that the value may be an array of information. Users can select multiple values, and PHP will place them all into an array of the value of the name attribute.

What is the difference between submit, reset and button inputs? For example:

|  |
| --- |
| <input type="submit" value="Buy Now"/>  <input type="reset" value="Clear Form"/>  <input type="button" value="Calculate Something"/> |

* The submit-button must always be the last value selected, as it initiates the form submission whereas the reset-button can be pressed at all times during the form fill out.
* Button won't submit form on its own. It is a simple button which is used to perform some operation by using javascript

### 

### Textarea, Select & Option elements:

These controls have more features, have a different format and are paired tags.

The textarea control is a multi-line text field and has extra attributes to control height and width:

|  |
| --- |
| <textarea name="..." rows="..." cols="..."> ... </textarea> |

The select and option elements are combined to create **Drop down** boxes and **Combo lists**:

|  |
| --- |
| <!-- This will be rendered as a drop down list -->  <select name="...">  <option value="mr">Mr.</option>  <option value="mrs">Mrs.</option>  <option value="miss">Miss</option>  <option value="ms">Ms.</option>  <option value="dr">Dr.</option>  </select>  <!-- This will be a combo box -->  <select name="..." size='...'>  <option value="mr">Mr.</option>  <option value="mrs">Mrs.</option>  <option value="miss">Miss</option>  <option value="ms">Ms.</option>  <option value="dr">Dr.</option>  </select> |

The multiple attribute allows the user to select more than one option. The list loses "radio-like" behaviour and gains "checkbox-like" behavior.

|  |
| --- |
| <select name="..." multiple > ... </select> |

Code up these inputs and draw what each looks like.

### 

### Labels:

For semantic reasons, it is useful to link a label with an input, for example:

|  |
| --- |
| <!-- Text, but no label element -->  <p>Name <input type='text' name='name' id= 'name' /></p>  <!-- Text inside a label element -->  <p><label>Name</label> <input type='text' name='name' id='name' /></p> |

In addition, we can make labels "clickable" by linking the label's for attribute to the id of an input:

|  |
| --- |
| <p><label for='name'>Name</label> <input id='name' type='text' name='name' /></p> |

Useful for text based inputs and very useful for small "hard to click" radio and checkbox inputs.

|  |
| --- |
| <!-- Gender is a "non-clickable" label -->  <p><label>Gender</label>  <!-- But labels below are clickable and increase the usability of the form -->  <input type='radio' name='gender' id='male' /> <label for='male' >Male</label>  <input type='radio' name='gender' id='female' /> <label for='female'>Female</label>  <input type='radio' name='gender' id='other' /> <label for='other' >Other</label>  </p> |

*If there is time, teaching staff will demonstrate some of the new HTML5 input elements and attributes.*