CS1116/CS5018

Web Development 2

Dr Derek Bridge

School of Computer Science & Information Technology
University College Cork

Server-side programs that expect user input

- Web pages can contain **forms** to gather data from the user
- When the user presses *Submit*, the browser sends the user's data to the server, where it can be processed by a server-side program

A web page that contains a form

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <title>Challenge</title>
    </head>
    <body>
        <p>
            Halt! Who goes there?
        </p>
        <form action="response.py" method="get">
            <input type="text" name="firstname" />
            <input type="text" name="surname" />
            <input type="submit" />
        </form>
    </body>
</html>
```

The `form` element

- The `form` start tag has two attributes:
    - `action`: the URL of the program that will handle the data
    - `method`: an HTTP command, e.g. GET or POST
- The `form` element contains '**controls**' such as textfields, radio buttons, menus, etc.

The `input` element and its `type` and `name` attributes

- Most controls are specified using the `input` element
  Question: What kind of element is it?
- The `type` attribute specifies what kind of control we want:
  - E.g. `text`, `password`, ...
  - http://www.w3schools.com/tags/att_input_type.asp
- The `name` attribute is required for all types except `reset` and `submit`.
  - Its value will later be used in the server-side program.

Text entry fields

- If you want a text entry field (for a single line of text), use an `input` element with `type="text"`:

```
<input type="text" />
```

- But it's useless unless you give it a name:

```
<input type="text" name="surname" />
```

- By default, it's 20 chars wide, but optionally use the `size` attribute to change its width:

```
<input type="text" name="surname" size="25" />
```

- By default, the user can type as many chars as s/he wishes (it scrolls horizontally!), but optionally use the `maxlength` attribute to restrict this:

```
<input type="text" name="surname" size="25" maxlength="25" />
```

- By default, the field is initially empty, but optionally use the `value` attribute to supply your own initial value:

```
<input type="text" name="num_children" size="25" maxlength="25" value="0" />
```

- By default, the field is enabled, which is what we want. If you want to use a field only for output, then include `disabled`:

```
<input type="text" name="total_cost" size="10" value="34.99" disabled />
```

Reset buttons

- A form might contain a reset button, which the user can press to clear the data s/he has typed
- Use an `input` element with attribute `type="reset"`:

```
<input type="reset" />
```

- By default in Chrome, the button has *Reset* written on it but you can supply your own label using the value attribute:

```
<input type="reset" value="CLEAR" />
```

Submit buttons

- To be useful, a form will contain a a submit button, which the user can press to send the data to the server
- Use an `input` element with attribute `type="submit"`:

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

- By default in Chrome, the button has *Submit* written on it but you can supply your own label using the value attribute:

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

How can we tell the user what to type?

- You usually want to explain to a user what s/he must enter into a text field
- There are at least three ways of doing this
  - the `title` attribute
  - the `placeholder` attribute
  - the `label` element

The `title` attribute

- The `title` attribute causes a tooltip to appear when the user hovers over the field
- E.g.:

```
<input type="text" name="zip"
    title="A valid US zip code or UK postcode. For other countries, leave blank." />
```

- You probably shouldn't use this method on its own. Why?

The `placeholder` attribute

- New to HTML5 is the `placeholder` attribute
- E.g.

```
<input type="text" name="firstname" placeholder="Your first name" />
```

- Its value appears in the text field, disappears when the user focuses on the field, and is restored when focus leaves the field if the user hasn't typed anything
- **Question**: How does `placeholder` differ from `value`?

The `label` element

- The most obvious way is to put some text next to the text field
- You could do this:

```
Surname: <input type="text" name="surname" />
```

- But it is better to use the `label` tag:

```
<label>Surname:</label>
<input type="text" name="surname" />
```

- **Question**: When might `placeholder` be better than `label`?

The for attribute

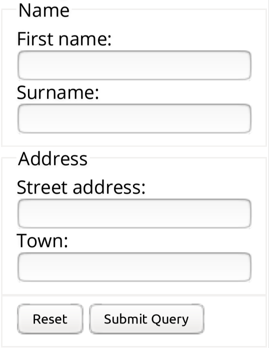
- Better still is to 'tie together' the `label` and the text field
    - put an `id` attribute on the text field (we'll use the same value as the name but they're different things)
    - put a `for` attribute on the `label`

```
<label for="surname">Surname:</label>
<input type="text" name="surname" id="surname" />
```

- Why?
    - Screen readers/voice browsers can use the `labels`
    - (BTW the labels are also clickable)

How can we group the controls in a form?

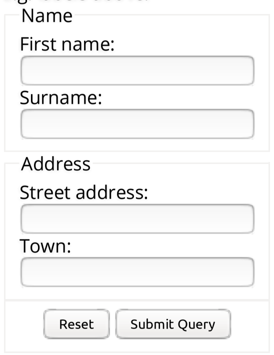
- Within the `form` element, group together related controls using a `fieldset` element
- Include a `legend` element nested in the `fieldset` to describe the group of controls
- E.g.



```
<form>
    <fieldset>
        <legend>Name</legend>
        <label for="firstname">First name: </label>
        <input type="text" name="firstname" id="firstname" />
        <label for="surname">Surname: </label>
        <input type="text" name="surname" id="surname" />
    </fieldset>
    <fieldset>
        <legend>Address</legend>
        <label for="street_address">Street address: </label>
        <input name="street_address" id="street_address" />
        <label for="town">Town: </label>
        <input name="town" id="town" />
    </fieldset>
    <fieldset>
        <input type="reset" />
        <input type="submit" />
    </fieldset>
</form>
```

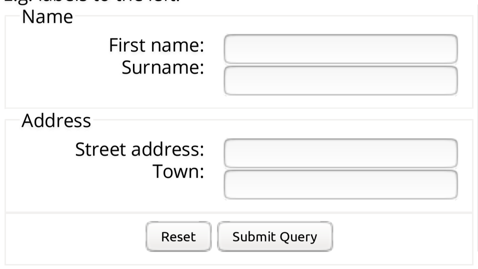
CSS for forms

- Many people use HTML tables for form layout, but CSS can be used instead
- E.g. labels above:



```
label {
    display: block;
}
```

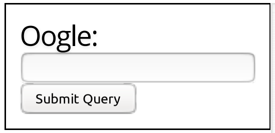
- E.g. labels to the left:



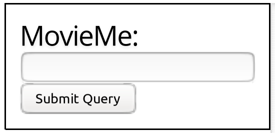
```
label {
    float: left;
    clear: left;
    width: 10em;
    margin-right: 1em;
    text-align: right;
}

input {
    float: right;
    clear: right;
}
```

Two working examples



```
<form action="https://www.google.com/search" method="get">
    <label for="search">Oogle: </label>
    <input type="text" name="q" id="search" />
    <input type="submit" />
</form>
```



```
<form action="https://www.imdb.com/find" method="get">
    <label for="search">MovieMe: </label>
    <input type="text" name="q" id="search" />
    <input type="submit" />
</form>
```