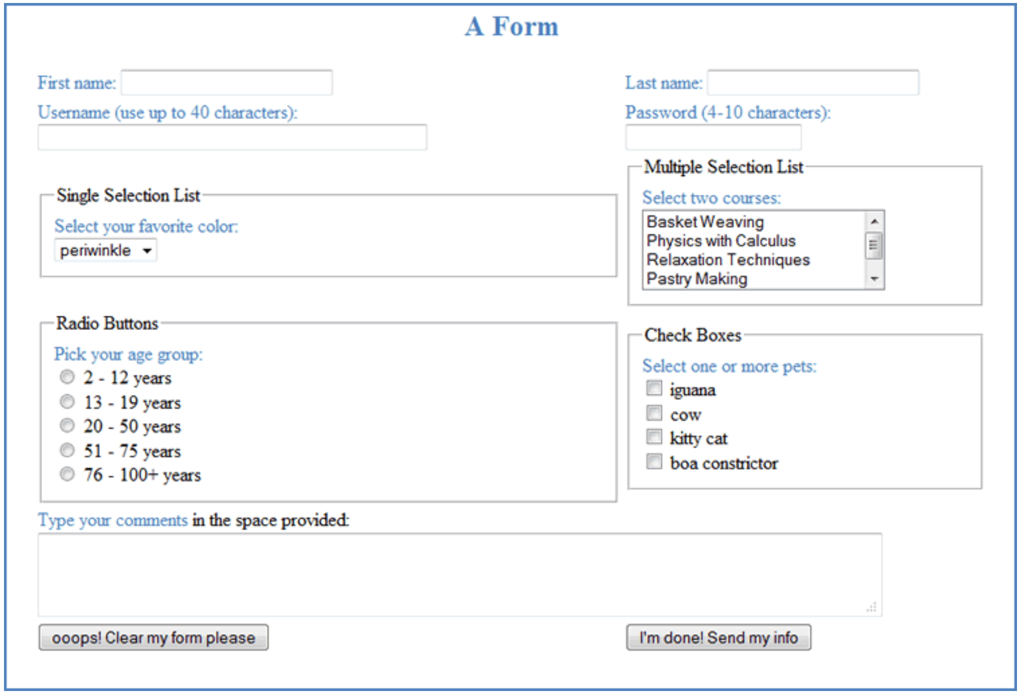
PROG8020

Week 3b

Forms and Form Controls

What is a Form?



A Form

First name: ______ Last name: ______

Username (use up to 40 characters): ______

Password (4-10 characters): ______

Single Selection List

Select your favorite color:
periwinkle ▾

Multiple Selection List

Select two courses:
Basket Weaving
Physics with Calculus
Relaxation Techniques
Pastry Making

Radio Buttons

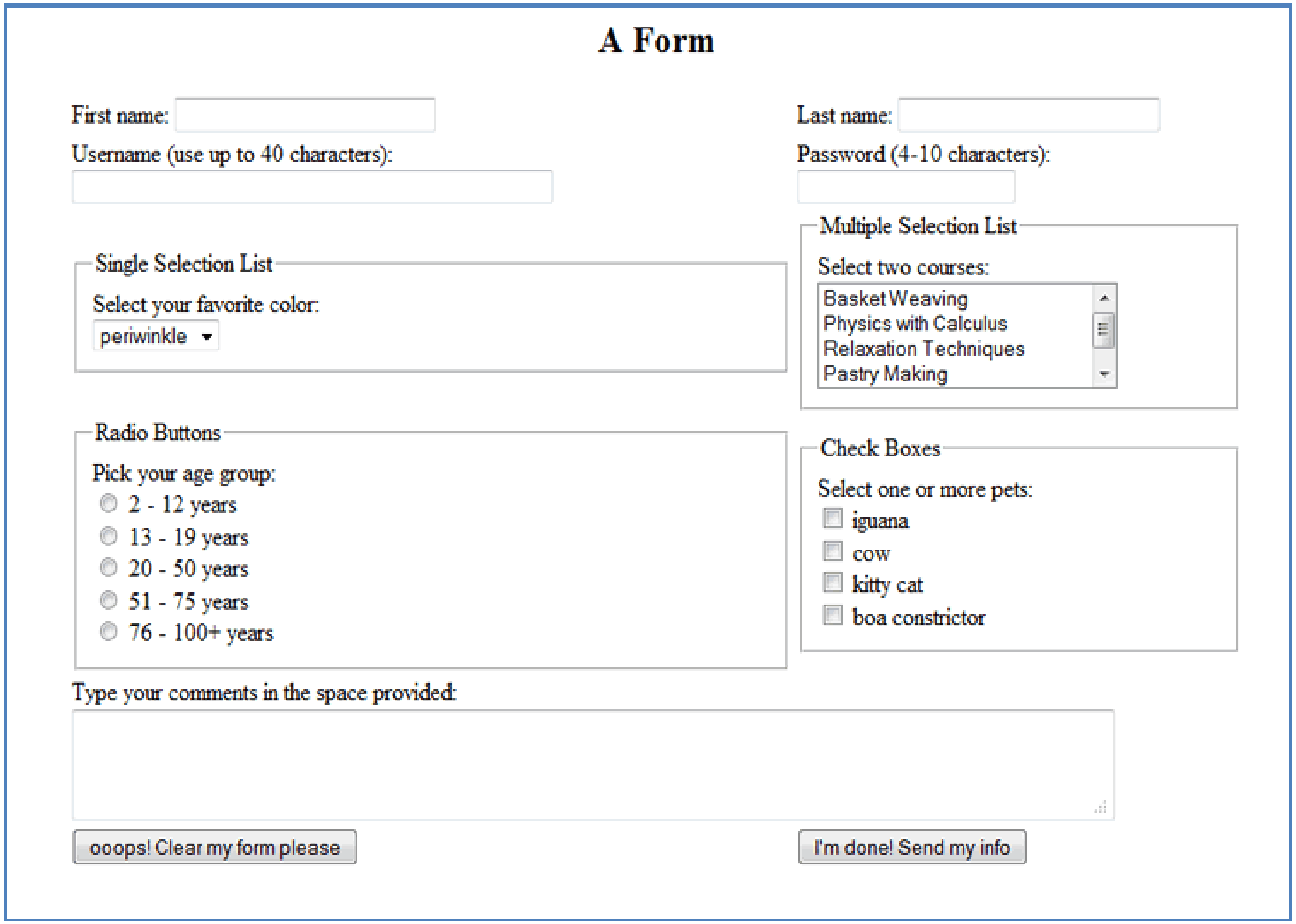
Pick your age group:
- ○ 2 - 12 years
- ○ 13 - 19 years
- ○ 20 - 50 years
- ○ 51 - 75 years
- ○ 76 - 100+ years

Check Boxes

Select one or more pets:
- □ iguana
- □ cow
- □ kitty cat
- □ boa constrictor

Type your comments in the space provided:

An HTML form is a way to enclose a section of a page with a name and use that name to access the form or the elements in the form, similar to creating a `<div></div>`. However, the elements in a form are treated differently from other HTML elements.



A Form

First name: Last name:

Username (use up to 40 characters): Password (4-10 characters):

Single Selection List

Select your favorite color:
periwinkle ▾

Multiple Selection List

Select two courses:
Basket Weaving
Physics with Calculus
Relaxation Techniques
Pastry Making

Radio Buttons

Pick your age group:
○ 2 - 12 years
○ 13 - 19 years
○ 20 - 50 years
○ 51 - 75 years
○ 76 - 100+ years

Check Boxes

Select one or more pets:
□ iguana
□ cow
□ kitty cat
□ boa constrictor

Type your comments in the space provided:

The `<form></form>` tag pair

- A form is an HTML object
- The object is created by using an opening `<form>` tag and a closing `</form>` tag
- Methods, events, attributes, and properties can be used by the form object
- The most important is the `name`
  - A form is used to collect user input
  - Without a `name`, there is no way to access the form and retrieve the input

Creating a Form

```
<body>
<form name="myfirstform" action="mailto:liz@forms.net"
                      method="post" enctype="text/plain">
       form elements go here
              .
              .
              .
       </form>
</body>
```

- **`name`** defines the name of this form and will be used to access the information on the form
- **`action`** returns the value of this attribute
  - In this form, the action will be to send an email to the following imaginary email address: `liz@forms.net`
- **`method`** specifies how to send the results
  - In this case, results will be sent as an HTTP post transaction
- **`enctype`** specifies how the data from the form should be encoded before sending it
  - In this case, the data will use plain text

The `submit` and `reset` Buttons

- To return data to a server or an email address, a `submit` button is required.
- To clear entries in case a user changes his/her mind, a `reset` button should be used.

```
<form name = “myfirstform“ action=“mailto:liz@forms.net”
                      method=“post“ enctype=“text/plain”>
       <h3>The contents of the form would go here</h3>
       <input type=“reset” value=“ooops! Clear my form please”>
       <input type="submit" value=“I'm done! Send my info”>
</form>
```

The `reset` type automatically clears all the user's entries on the form. The `submit` type automatically submits the user's information using the attributes defined in the `<form>` tag.

Returning Data by Email

- Each submission can also be returned to the developer through an email message.
- Simple method, can be used by anyone with an email account
- Not ideal – better ways to process large amounts of information
- But might work, for example, to process complaints or specific questions from a user to a website
- Syntax to send form data by email is, assuming a manager in charge of complaints is named `Liz Loverly` at `liz.loverly@jackiejewels.net`:

```
<form name = "complaints" method = "post" id =
        "complaints“ action =
        "mailto:liz.loverly@jackiejewels.net">
```

- This method generates an email message to `liz.loverly@jackiejewels.net` from whatever email program the user employs.

Form Controls

Form Controls: Radio Buttons

The radio button is an object in an HTML form with properties and events.

The `name` property defines a group of buttons and thus requires that only one of them can be selected at any time. This distinguishes the radio button from a checkbox.

Property	Description
checked	sets or returns the checked state of the button
defaultChecked	returns the default value of the checked attribute
disabled	sets or returns whether or not the button is disabled
form	returns a reference to the form where the button is
name	sets or returns the name of the button
type	returns the type of the form element
value	sets or returns the value assigned to the button

Form Controls: Checkboxes & Radio Buttons

The checkbox is also an object in an HTML form. It supports the same properties and events as the radio button. However, when the user sees a list of options that are checkboxes, any number of these checkboxes may be selected.

- The syntax for each radio button is as follows:

```
<input type = "radio" name = "radio_button_name" id =
       "radio_button_id" value = "radio_button_value">
```

- The syntax for each checkbox is as follows:

```
<input type="checkbox" name = "box_name" id = "box_id“
       value = "box_value">
```

Form Controls: Textboxes

- The textbox is an input element that allows the web developer to display a small area for a user to enter some information.
- It has several properties that are not available to radio buttons or checkboxes.
  - Can set the `size` of the box (i.e., its width)
  - Can set the `maxlength` which configures the maximum number of characters that will be accepted
  - Can place an initial value in the box, if desired.

The syntax for a textbox is as follows:

```
<input type="text" name = "box_name" id = "box_id“
            size = "20" maxlength = "25" value =
            "my box!">
```

Form Controls: Label, FieldSet, Legend

- The `<label></label>` tags allow you to enter a label (a description) for your textbox.
  - The opening `<label>` tag goes right before the desired label and the closing `</label>` tag goes after the label or after the `<input>` statement.
- If a group of form controls are enclosed in `<fieldset></fieldset>` tags, the browser will put a border around these elements.
- Adding the `<legend></legend>` tags will allow the browser to include a label for the fieldset grouping.

Form Controls: TextArea

- A textarea box designates a space for a user to enter text.
  - Both height and width can be specified in a textarea box.
  - `textarea` tags are `<textarea></textarea>`
  - The `cols` and `rows` properties determine the size of the box.
- These boxes are normally used to allow a web site visitor to include comments or questions when returning a form.

The syntax for a text is as follows:

```
<textarea name = "box_name" id = "box_id"
            cols = "20“ rows = "5">Default
            text if desired</textarea>
```

Form Actions: email

- The `email` action is placed in the opening `<form>` tag
- Can also add a subject line to the generated email
- Can add a copy to be sent to another recipient.

The syntax for these options are as follows:

- This will generate an email sent to `whoever@wherever.net` with the subject line `Whatever`:

```
      <form name = "myform" method = "post" enctype =
            "text/plain" action  =
            "mailto:whoever@wherever.net?Whatever">
```

- This will generate an email sent to `whoever@wherever.net` with the subject line `Whatever` and will send a copy to `whatshisname@whereisit.net`:

```
      <form name = "myform" method = "post" enctype =
            "text/plain" action =
            "mailto:whoever@wherever.net
            ?Whatever&cc=whatshisname@whereisit.net">
```

Hidden Fields and Passwords

The `Hidden` Form Element

- Imagine a business website where a customer signs in with his/her username which you want to use on every subsequent page. You can store that username in a hidden field and carry it from page to page.
- You can also use the information in a hidden field when you communicate with the server.
- Properties of a hidden object are: `name`, `type`, `id`, and `value`.

The general syntax for a hidden field is as follows:

```
<input type = "hidden" name = "field_name" id =
  "field_id" value = "field_value" />
```

The `Password` Form Element

- The password form element is a single-line input field in a form.
- The content of the field will be masked (replaced by a character such as an asterisk or small dot).
- A password field can be accessed by using `document.getElementById()`.
- The general syntax of a password field is as follows:

```
<input type = "password" then set desired properties />
```

- The password object uses the same properties as the other input fields as well as some others

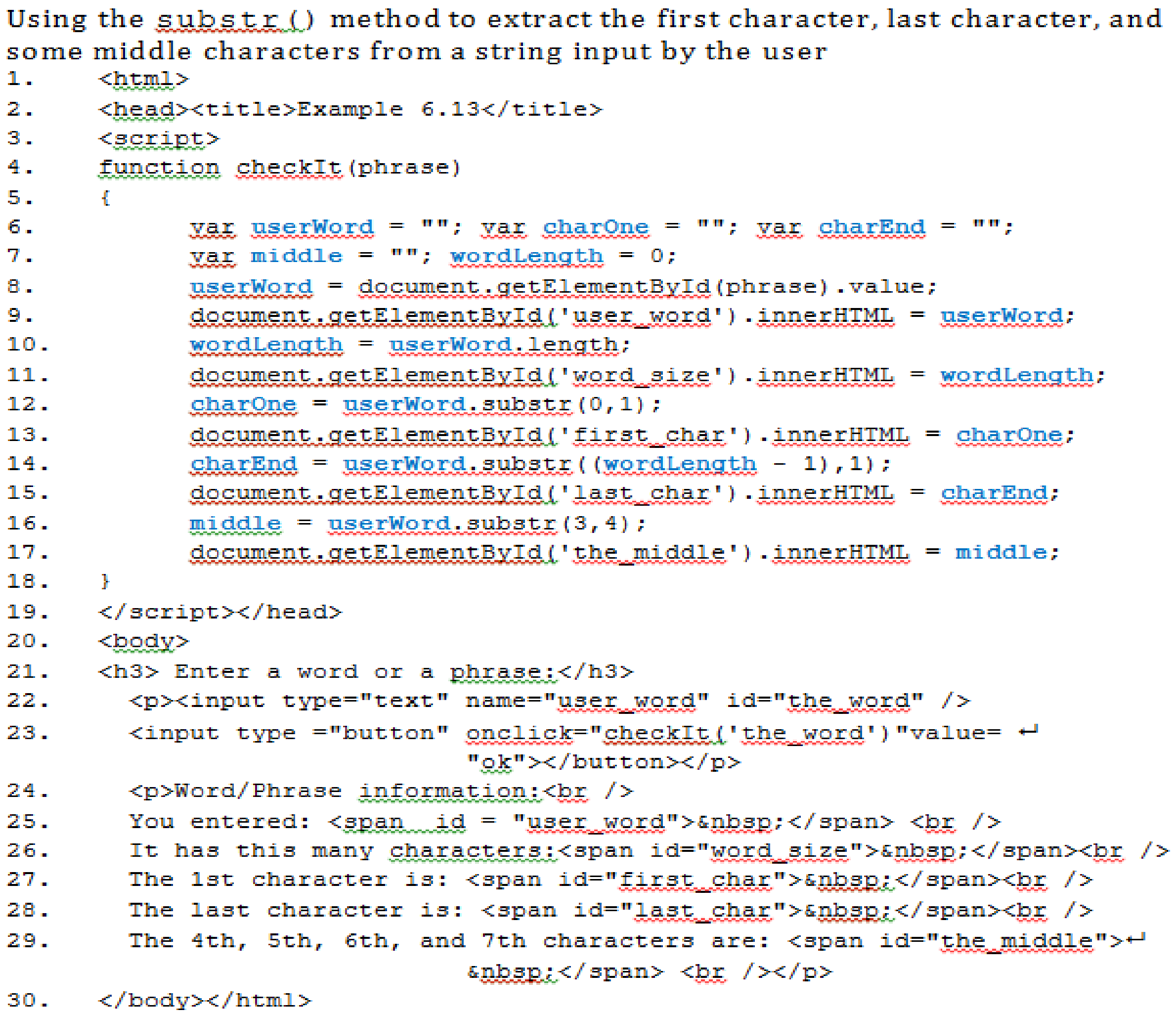
Property	Description
defaultValue	returns or sets the default value of a password field
disabled	sets or returns whether or not the field is disabled
form	returns a reference to the form where the field is
name	sets or returns the name of the password field
maxLength	sets or returns the maximum number of characters allowed
readOnly	sets or returns whether or not the field is read-only
type	returns the type of the form element
value	sets or returns which type of form element the field is
size	sets or returns the width of the field (i.e., number of characters)

The `substr()` Method

- The `substr()` method will extract the characters from a string, beginning at the character you specify and continuing through as many characters as you want. It returns the new substring.

String	Character Number							
	0	1	2	3	4	5	…	n
cat	c	a	t					
A table	A		t	a	b	l	e	
Jones-Smith	J	o	n	e	s	-	…	h

Example: substr()



Using the substr() method to extract the first character, last character, and some middle characters from a string input by the user

```
<html>
<head><title>Example 6.13</title>
<script>
function checkIt(phrase)
{
      var userWord = ""; var charOne = ""; var charEnd = "";
      var middle = ""; wordLength = 0;
      userWord = document.getElementById(phrase).value;
      document.getElementById('user_word').innerHTML = userWord;
      wordLength = userWord.length;
      document.getElementById('word_size').innerHTML = wordLength;
      charOne = userWord.substr(0,1);
      document.getElementById('first_char').innerHTML = charOne;
      charEnd = userWord.substr((wordLength - 1),1);
      document.getElementById('last_char').innerHTML = charEnd;
      middle = userWord.substr(3,4);
      document.getElementById('the_middle').innerHTML = middle;
}
</script></head>
<body>
<h3> Enter a word or a phrase:</h3>
  <p><input type="text" name="user_word" id="the_word" />
  <input type ="button" onclick="checkIt('the_word')"value= ↵
                        "ok"></button></p>
  <p>Word/Phrase information:<br />
  You entered: <span  id = "user_word">&nbsp;</span> <br />
  It has this many characters:<span id="word_size">&nbsp;</span><br />
  The 1st character is: <span id="first_char">&nbsp;</span><br />
  The last character is: <span id="last_char">&nbsp;</span><br />
  The 4th, 5th, 6th, and 7th characters are: <span id="the_middle">↵
                        &nbsp;</span> <br /></p>
</body></html>
```

Selection Lists and More

Selection Lists

- A selection list is created using the `<select></select>` container tags.
- Similar to `<ul></ul>` HTML tags; it defines a container which will house options.
- Like `<li></li>` tags, a selection list configures the items with `<option></option>` tags.

The general syntax for a selection list, where `N` is some number is as follows:

```
<select size = "N" name = "list_name" id = "list_id">
        <option value ="option1 value">some text </option>
        <option value ="option2 value">some text </option>
                          ......
                          ......
        <option value ="optionN value">some text </option>
</select>
```

- The `<option>` tag can contain the `selected` property which, when included and set to `"selected"` will display the value in that tag as highlighted.

The `size` and `multiple` Attributes

- `size` shows how many of the options will be visible.
- If `size` is set to 1 a drop-down list will automatically be created to show all the options.
- If the `size` is set to fewer than the number of options, a scroll bar is automatically added to allow the user to see all the options.
- When a selection list is created, by default the user is only allowed to select one item.
- The `multiple` attribute allows you to configure a selection box so the user is permitted to select more than one of the options.
  - Sometimes this attribute may be useful but the user must hold down a particular key to select multiple items so it may be more complicated than it’s worth.

Form Element Enhancements

- tabindex attribute: The default action for the tab (↹) key is to move to the next form control. This attribute allows you to change the tab order.

- accesskey attribute: Allows you to assign a keyboard character as a hot key that user can press to move the cursor immediately to a specific form control. General syntax:
  - `element.accesskey = key_you_choose;`

- onfocus event: when an element gets focus.
  - General syntax for use in an HTML document:

    ```
    <element onfocus = "JavaScript code" >
    ```

- this keyword: always refers to the function or element that you are referring to.
  - General syntax of the this keyword is as follows:

    ```
    <input type="text" name="box_name" id="box_id‶ onfocus = ↵
              "setFunction(this.id)" />
    ```

  - In this case, the this keyword, combined with .id, identifies the id of this textbox.

Thank You!