Some additional HTML form widgets

Radio buttons

- **Radio buttons** are another type of **`input`** element;
- This can be a good choice when the user is to select **EXACTLY ONE** of a "SMALLISH" number of specific options.
  - Selecting one of these causes the previous selection to be UN-selected.

IMPORTANT TO KNOW about radio buttons

- An **`input`** element with **`type="radio"`** is a radio button.
- Browsers typically format a radio button as a small circle -- clicking it fills in that circle and "un-fills" the circle for the previously-selected radio button in its logical group.
- Each should have a **`name`** attribute. To logically group those radio buttons representing the options for a single choice, give each the SAME value for each of their **`name`** attributes.
  - [One way to think about this: all together, the radio buttons with the same value for their **`name`** attribute represent a "single" logical element.]
- Each should have a **`value`** attribute.
  - When the form is submitted, one name=value pair will be sent for the whole logical group of radio buttons;
  - the name will be the value of that logical group's **`name`** attribute,
  - and the value will be the **`value`** attribute's value for the currently-selected radio button in that logical group of radio buttons.
- Add attribute **`checked="checked"`** to the SINGLE radio button you'd like to be INITIALLY selected when the form is created.
  - What if you don't? Then, when the form is submitted, if the user never clicked one of the radio buttons, NO name=value pair will be submitted for that logical set of radio buttons, which may be confusing on the application tier when handling this form.
- Add a logically-related **`label`** element along with each radio button instance to logically associate that radio button's label text with that radio button.

Checkboxes

- **Checkboxes** are another type of **`input`** element;
- This can be a good choice when the user is to select **ZERO OR MORE** from a set of specific options.

IMPORTANT TO KNOW about checkboxes

- An **`input`** element with **`type="checkbox"`** is a checkbox.
- Browsers typically format a checkbox as a small square -- clicking it checks or unchecks this checkbox.
- Each should have a **`name`** attribute.
- If a checkbox **is** checked when its form is submitted, a name=value pair will be submitted for that checkbox;

  if a checkbox is **NOT** checked when a form is submitted, **NO** name=value pair will be submitted for it.

  - For each checked checkbox, this **`name`** attribute's value will be sent as the name in a name=value pair.
  - By default, the value for the name=value pair submitted for a checked checkbox will be **`on`** (although if there is a **`value`** attribute, its value will be the value instead.)
- Add attribute **`checked="checked"`** for EACH checkbox you'd like to be INITIALLY selected when the form is created
  - USE THIS FOR GOOD, NOT EVIL! ...consider: what is a REASONABLE default? considering user convenience...!
- Add a logically-related **`label`** element along with each checkbox instance to logically associate that checkbox's label text with that checkbox.

Drop-down box - **`select`** element

- This one actually IS a separate element, and **NOT** a type of **`input`** element!
- This can work better than radio buttons when user is selecting **EXACTLY ONE** from a ***larger*** set of specific options.
  - (Although you can add an attribute to allow the user to select more than one from the set of options, how this works on the user's end varies enough amongst different browsers that users

may find it frustrating.)

- BASIC syntax:

```
<select name="desired-name">
    <option value="whatever1"> shows in drop-down </option>
    <option value="whatever2"> shows in drop-down </option>
    <option value="whatever3" selected="selected"> shows initially
        </option>
</select>
```

IMPORTANT TO KNOW about drop-down boxes/`select` elements

- **`select`** is an inline element, and it CAN have content
- Use attribute **`selected="selected"`** for the SINGLE **`option`** element whose content you'd like to be INITIALLY selected/showing when the form is created.
- Be sure to give the **`select`** element a **`name`** attribute,

  AND to give each **`option`** element nested within it a **`value`** attribute.
- Note that the **`option`** element's content is what the user will **see**,

  and the value of the **`option`** element's **`value`** attribute is what will be sent as the value for this element's name=value pair if that option is selected when the form is submitted.

  (and the name in the name=value pair will be the **`name`** attribute's value in the **`select`** element's start tag)
- Add a logically-related **`label`** element along with the **`select`** element to logically associate that select/drop-down element's label text with that select/drop-down element.

`textarea` element

- This one also is a separate element, and **NOT** a type of **`input`** element!
- This can be a good choice when the user may be entering **multiple** lines of freeform content/text.

IMPORTANT TO KNOW about `textarea` elements

- **`textarea`** is an inline element, and it CAN have content
- Be sure to give the **`textarea`** element a **`name`** attribute.
  - The value of its **`name`** attribute will be the name for the name=value pair sent for this **`textarea`** when its form is submitted.
- Its optional **`rows`** attribute specifies how many rows are displayed (height!).

- Its optional **`cols`** attribute specifies roughly how many "characters" are displayed (width!), but note that this is approximate, especially if the display font is not monospaced (a W is wider than an i !)
- If this element happens to include content, that content will be the *initial* text showing within the **`textarea`** element when the form is displayed.
  - Fun fact: if there are blanks or newline in this content, it will be included within the textarea when the form is displayed!
- Using a **`placeholder`** attribute along with empty content can be convenient, but note that:

  1. This may not be read by screen reader software, so do **not** use this *instead* of a logically-associated **`label`** element!

  2. This **`placeholder`** attribute's value will **not** be sent as the value for the textarea if the user has not entered anything in this textarea when the form is submitted.

  3. If the **`textarea`** element has ANY content -- even a single blank! -- it **OVERRIDES** the **`placeholder`** attribute and you won't see the placeholder attribute's value in your textarea. So, when you have a **`placeholder`** attribute, put the **`textarea`**'s end tag **RIGHT** after its start tag.
- Whatever (non-`placeholder`) content is appearing within the **`textarea`** when the form is submitted will become the value for the name=value pair sent for this textarea.
  - And, if there are blanks or newline in this content, it will be converted to URL-friendly character equivalents and included in the value submitted!
- Add a logically-related **`label`** element along with the **`textarea`** element to logically associate that **`textarea`** element's label text with that **`textarea`**.