Announcements

- Check class announcements daily
- You must implement programming projects by yourself
**Events**

- **Event** → Notification that something has occurred
- Example of situations that make the web browser generate an event
  - Browser finishes loading a document
  - When the user clicks on a button
  - When the user moves the mouse
  - Others
- **Event handler** (also known as event listener)
  - JavaScript function or code fragment that is executed when a particular event occurs
- **Event handler registration**
  - Associating an event handler with a particular event
- **Example**: EventEx.html
Event-driven Programming

- Normal (control flow-based) programming
  - Approach
    - Start at main()
    - Continue until end of program or exit()

- Event-driven programming
  - Start at main()
  - Register event handlers.
  - Await events & perform associated computation

- GUIs (Graphical User Interfaces)
  - Example of event-driven software
Event Handler Attributes for most HTML

**Mouse Related**
- `onclick` → mouse button is pressed and released
- `ondblclick` → mouse button is double-click over element
- `onmouseover` → mouse moves over element
- `onmouseout` → mouse moves off element
- `onmousemove` → mouse pointer is moved
- `onmousedown` → mouse is pressed down while cursor is over the element
- `onmouseup` → mouse is released while the cursor is over the element

**Keyboard Related**
- `onkeypress` → key pressed and released
- `onkeydown` → key is pressed
- `onkeyup` → key is released

**Other**
- Keep in mind that there additional handlers that are specific to certain tags. We will address those later on
HTML Forms

- **Forms** → means by which information passes from the user to a server
- For now we will use forms to read values to be processed by our JavaScript programs
- `<form>` tag
  - Defines the form
  - It has two attributes: action and method
  - `action` → indicates where the form contents will be sent when the form is submitted
  - `method` → defines how the contents will be sent (post/get)
- `<input>` tag
  - Appears inside of the `<form>` tag
  - Defines several input data alternatives.
  - The general format is: `<input type="ALTERNATIVE" />
  - **ALTERNATIVE** can be text, password, checkbox, radio, file, submit, image, button, reset, hidden
  - We have can several forms in our document
- **Example**: AssociateButtonWithFunctionV1.html
Accessing Data Associated with HTML

- As we saw we can access data in forms by using
  ```javascript
  document.getElementById("elementId");
  ```
- `getElementById` returns a reference to an element that we can use to:
  - Retrieve the value of the element (e.g., text field in a form)
    ```javascript
    var login = document.getElementById("loginId").value;
    ```
  - Set the function to call when an element is clicked on (e.g., button)
    ```javascript
    document.getElementById("processButton").onclick = functionDoesProcessing;
    ```
  - Get/Set Attributes
    ```javascript
    var imageElement = document.getElementById("myImage");
    var imageName = imageElement.getAttribute("src");
    imageElement.setAttribute("src", "imageFile.jpg");
    ```
- **Example:** AssociateButtonWithFunctionV2.html
- **Example:** GetValueInTextField.html
- **Example:** UpdateValueInTextField.html
- **Example:** GetSetAttribute.html
Reset

- The functionality of the Reset button is already provided by HTML. You don't need to add any JavaScript or define a button.
- You can change the text associated with the Reset button by using the value attribute.
- **Example:** Reset.html
Examples

- **Example:** PhotoViewer.html
- Will it work with http://www.cs.umd.edu/assets/images/frontpage/vm.jpg?
- **Example:** Animation.html