Exercise

1. Create a global variable representing a bank account balance, initialized to zero
2. Write the function `deposit(amount)`
   - Should check that amount being deposited is positive before adding it to the bank account balance
   - If the amount being deposited is not positive, then you should produce an error message
   - print the balance to the console
3. Write the function `withdrawal(amount)`
   - Should check that the amount being withdrawn is positive, and that there is enough money in the bank account balance for the withdrawal to occur
   - If there isn't enough money for the withdrawal, produce an insufficient funds error
   - If the amount being deposited is not positive, then you should produce an error message
   - print the balance to the console

Events and Event Driven Programming

Question:

- What does a webpage do when the page finish loading, but the user hasn't done anything yet?

- It waits for input, or other events
- Once an event happens then JavaScript code is executed.

So Far…

- Our programs have run as soon as the page loaded, and finished running before the user could interact the page
- So we have used prompt and alert to interact with the user
- This isn't typical website behavior
- Typical website behavior waits for the user to do something (to take an action), and then responds in some way
- This type of action/response behavior is an Event Driven Programming model

The Event Loop

- JavaScript has a built-in event loop

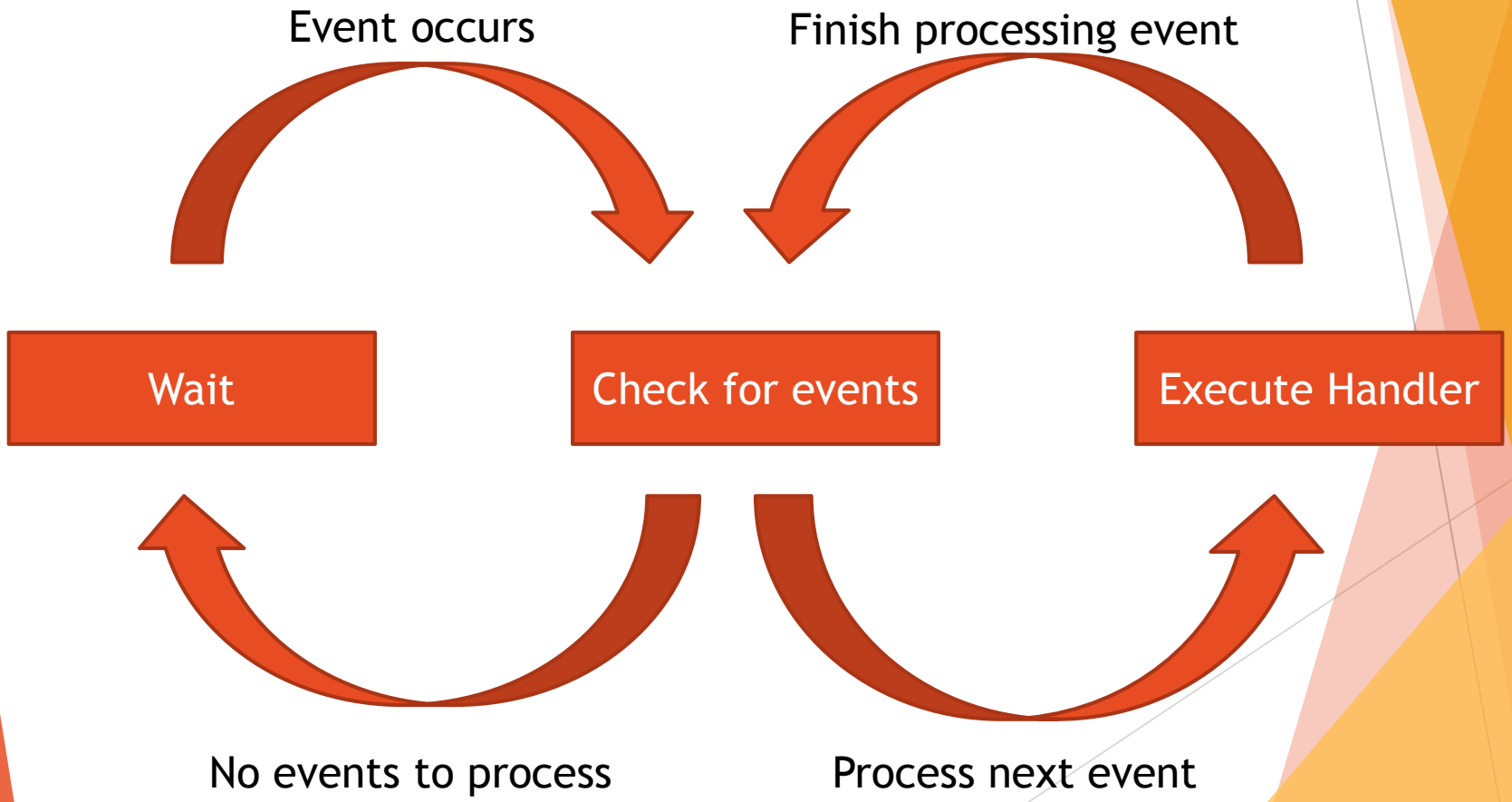
1. Once the page has loaded, any code that is outside of a function is executed
2. Then JavaScript then waits for events (action taken by user)

How Events are Handled

When an event occurs

1. The event is added to the list of events to be handled
2. If JS is not currently handling an event
    1. JS looks for an event handler for the earliest event on the list
    2. The event handler is executed
    3. The event is removed from the list
    4. JS checks for the next event on the list
        - If there are no more events on the list, it goes back to waiting

Event Loop


Event occurs
Finish processing event
Wait
Check for events
Execute Handler
No events to process
Process next event

Events

- An event is anything that happens on a web page
  - Significant moments generated by the web browser
  - Includes any user action
  - Usually associated with an element
- We will examine two types of events
  - onload – event occurs once the HTML file has finished loading
  - onclick – event occurs when an element has been clicked

onload

- This event is associated with the document
- Once the HTML has loaded is great time to set up our JS
- How do we handle onload?
  - We add an onload attribute to the body of the HTML page

```
<body onload="setup()">
…
</body>
```

- This code calls the setup() function once the page has loaded
- Notice the JS is in quotations marks

onclick

- This event is associated with the elements
- Usually used for buttons, but can work with any element
- How do we handle onclick?
  - We add an onclick attribute to the target element

```
<button id="button1" type="button"
onclick="deposit()">
Make a deposit</button>
```

- We can give the button a unique ID, and set type="button"
- Then the onclick attribute determines what function runs when the button is clicked

onclick

```
<button id="button1" type="button"
onclick="deposit()">
   Make a deposit
</button>
```

- We can have any function run when the button is clicked
- We should ALWAYS limit this code to a single function
- Technically you could place a full JS program in the quotes
- But using only one function call keeps the code tidy

Alternate way of Handling Events

- There is another way we can add the event handler to our code
- First we get the element from the HTML page
- Then we set the element's event property to refer to our function

```
var element =
document.getElementById("button1");
element.onclick = deposit;
```

- NOTE: There are no brackets after deposit
- This is because we want to refer to the function, not invoke it at this time

<input>

- We can add inputs to the webpage
  - Like buttons and textfields
  - Both use <input> tag
- We can use the textfield rather than the prompt to accept user input!
- Much less irritating
- But in order to be able to use the contents of the textfield, we have to let the user type things, hence why we needed events

HTML Inputs


```
<input type="text" id="input1">
```

- This HTML generates a text field
- We can give it any unique ID name that we want
- Type **must** be text
- Then in our JS
  - We get the text that user typed into the field by getting the "value" property from the input element

```
var inputField =
document.getElementById("input1");

var userData = inputField.value;
```

The Other Way to Add Buttons

```
<input type="button" id="button1"
onclick="example()">
```

- We give the button a unique ID
- Set type equal to button
- Then have any function run when the button is clicked

Other Input Types

- color
- password
- date
- file
- radio
- Range
- You’re free to investigate them
- The basic pattern is the same, you get the value from the input element for most input types, except buttons

Persisting Information

- If everything we code is inside functions
- And local variables go away once the function has executed
- Then information we want to keep around must exist inside global variables
- If everything is inside functions the our JavaScript program is now a collection of functions
- These functions can run in any order, which is usually determined by the user input
- Then these functions will change global variables to keep track of information

New Bank Account Example

- Let's make our bank account example more interactive!
- Let's add two text fields and some buttons!
- The user will enter the amount to deposit/withdraw in the text field
- Then click the button to perform the deposit/withdrawal function

Creating the HTML

```
<h3>Make a Deposit</h3>

<input type="text“ id=“depositAmount”>
<input type="button" value="Deposit Funds” onclick= “???">

<h3>Make a Withdrawal</h3>

<input type="text“ id=“withdrawalAmount”>
<input type="button" value="Withdraw Funds" onclick=“???”)">
```

- Can we just use our deposit and withdrawal functions here?

Creating the HTML

```
<h3>Make a Deposit</h3>

<input type="text“ id=“depositAmount”>
<input type="button" value="Deposit Funds” onclick= "makeDeposit()">

<h3>Make a Withdrawal</h3>

<input type="text“id=“withdrawalAmount”>
<input type="button" value="Withdraw Funds" onclick="makeWithdrawal()">
```

Creating makeDeposit()

- makeDeposit
  - Should read the deposit amount from the text field
  - Should perform the deposit

```
function makeDeposit(){
   var amt =
document.getElementById("depositAmount");
   amt = parseInt(amt.value);
   deposit(amt);
}
```

Creating makeWithdrawal()

- What does makeWithdrawal have to do?
  - Should read the withdrawal amount from the text field
  - Should perform the withdrawal

```
function makeWithdrawal(){
    var amt =
document.getElementById("withdrawalAmount");
    amt = parseInt(amt.value);
    withdrawal(amt);
}
```

Let’s See How It looks

- Let’s make one other change
  - Make deposit and withdrawal print to console instead of return
  - This way we can see what’s happening
- account.html

Wouldn’t It Be Nice…

- To be able to see our current account balance right on the page?
- Let’s add a place to display the balance to the HTML page
- `<h2 id="yourBalance"></h2>`
- And create a new function to update the balance

Creating displayBalance()

- What does this function have to do?
- It has to get the element we want to display the balance in
- Then it has to display the balance

```
function displayBalance(){
    var display =
       document.getElementById("yourBalance");
    display.innerHTML = accountBalance;
}
```

One Last Change

- This displayBalance function is never called in our code
- Where should it be called so that the correct and current account balance information is displayed?

- At the end of each function that updates the account balance
  - So at the end of makeWithdrawal and makeDeposit
- Let's see how it looks~!
  - account2.html

Other Events

- A list of events that you can respond to can be found here:
- http://www.w3schools.com/jsref/dom_obj_event.asp
- This will be useful for your projects.
- Mouse Events
  - ondblclick, onmousedown, onmouseenter, onmouseleave, onmouseover, etc.
- Keyboard events
  - Onkeydown, onkeypress, onkeyup
- The list goes on and on and on

Questions

1. What is a JavaScript event?
2. What is the keycode for the letter A?
3. What is a handler used for?
4. Name four mouse movement events.
5. What is the load event used for?
6. What event do you write a handler for to listen for mouse clicks?
7. What is an event object?
8. Name the three key events mentioned thus far.
9. What is a bug?
10. Describe why a scroll event may be useful in a site/app/page