JS ITERATION

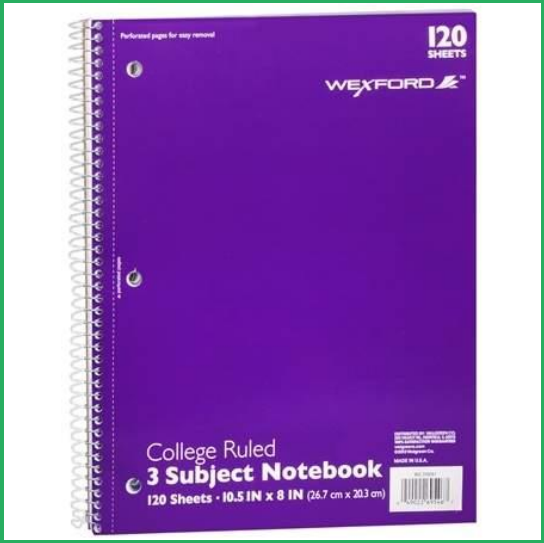
Array Iteration

Objectives

- Review and synthesize objects and functions to perform small routines in a programs.
- Review iteration and utilize common array methods: forEach, map, filter, and reduce.
- Define and demonstrate how to use arrays and objects to model our data.

Objects: Review

Write an object describing the properties of the following image:



Objects: Review

Write an object describing the properties of the following image:

Objects: Review

Write an object describing the properties of the following image:

Objects: Review

Write an object describing the properties of the following image:

Objects: Review

Use arrays and objects to describe the following image

Functions: Review

Write a function satisfying the following:

- Write a function to print the fullName of an object
  - ```
    fullName({first: "jane", last: "doe"});

    // => "jane doe"
    ```
    ```

Functions: Review

Write a function satisfying the following:

- Name it sum
- It takes an array and adds up the values
  - `sum([1, 2, 3, 4, 5, 6, 7]); // => 28`
  - `sum([]) // => 0`

Functions: Review

Write a function satisfying the following:

- Write a function to take a list of objects and return their fullNames
  - ```
    fullNames(
      [
        { first: "jane", last: "doe" },
        { first: "john", last: "doe" }
      ]
    );

    // => ["jane doe", "john doe"];
    ```
    ```

ITERATE!

- **Take a moment to pair and discuss.**
  - **What does iteration mean?**
  - **Where have we seen iteration?**

ITERATE!

What does iteration mean to me?

- Iteration requires the notion of a **sequence**.
  - **Sequence**:
    - A continuous or connected series
      - Ex: A series of items ordered and labeled by positive integers
- Iteration is proceeding in a **sequence** through some collection

ITERATE!

Where does it come up?

- When we go through every value in an array to do something.
  - Go through every value and print it to the console
  - Go through every value and capitalize it.
  - Go through every value and print only numbers that are odd
  - Go through every value and add them up

Each

Imagine we wanted to iterate through an array and run a function called normalize.

```
for(var i = 0; i < arr.length; i += 1 ) {
    normalize(arr[i]);
}
```

Each

Take a moment and think about at least three things that can go wrong here.

Then pair and discuss your top concern with the person next to you.

```
for(var i = 0; i < arr.length; i += 1 ) {
    normalize(arr[i]);
}
```

Each

Imagine I had the following code example

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.innerHTML.length > 20 ) {
      text = text.slice(0, 20);
      text += "...";
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

Each

There's already quite a lot going on here...

We have to access and set the current item we are dealing with.

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.innerHTML.length > 20 ) {
      text = text.slice(0, 20);
      text += "...";
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

Each

There's already quite a lot going on here...

We have to access and set the current item we are dealing with.

We have to retrieve the innerHTML of our item.

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.innerHTML.length > 20 ) {
      text = text.slice(0, 20);
      text += "...";
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

Each

There's already quite a lot going on here...

We have to access and set the current item we are dealing with.

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.i                    20 ) {
        Truncate it to twenty characters
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

We have to retrieve the innerHTML of our item.

Each

There's already quite a lot going on here...

We have to access and set the current item we are dealing with.

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.i                    20 ) {
        Truncate it to twenty characters
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

We have to retrieve the innerHTML of our item.

Capitalize text

Each

There's already quite a lot going on here...

We have to access and set the current item we are dealing with.

```
var items = document.querySelectorAll(".item");
var text;

for(var i = 0; i < items.length; i += 1 ) {
    var item = items[i];
    text = item.innerHTML;

    if (item.i                    20 ) {
        Truncate it to twenty characters
    }

    text = text.capitalize();

    item.innerHTML = text;
}
```

We have to retrieve the innerHTML of our item.

Capitalize text

Each

So with all the potential code that can be in a for loop…

- **Why worry about this?**

```
for(var i = 0; i < arr.length; i += 1 ) {
  // code
}
```

Each

So with all the potential code that can be in a for loop…

- **When we just care about this...**

```
// code
```

Each

This where functions come to the rescue!

- **Enter the `forEach`**

```
myArr.forEach(function (item) {
  // code
});
```

Each

Let’s use this! Try

```
var myArr = [1, 2, 3, 4, 5];

myArr.forEach(function (item) {
  console.log(item);
});
```

Each

Let’s use this! Try the following:

```
var friends = ["jane", "john", "joe"];

friends.forEach(function (item) {
  console.log(item);
});
```

Each

Let's use this! Try the following:

- **You can optionally use the index.**

```
var friends = ["jane", "john", "joe"];

friends.forEach(function (item, index) {
  console.log(index, item);
});
```

Each

Your turn: Let's re-write our `sum` function from earlier using forEach.

```
function sum(numbers) {
 var sum = 0;

 for (var i = 0; i < arr.length; i += 1 ) {
   sum += arr[i];
 }

 return sum;
}
```

Each

- Does the forEach method do?
- What problems does it help solve over a conventional for loop?

Map

We just discussed each. Our big take away was that a function could help us abstract out something we do every day: looping.

Map

Imagine we wanted to go through a list of names and capitalize them all.

```
var friends = ["jane", "john", "joe"];
var capNames = [];

for (var i = 0; i < friends.length; i += 1 ) {
  capNames.push(
    friends[i].toUpperCase()
  );
}
```

Map

Well as a first step we could just use each.

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.toUpperCase()
  );
}
```

Map

This is still a lot of work and we have remember to do a few things correctly.

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.toUpperCase()
  );
}
```

Map

What could go wrong here?

I could forget to create my array for results

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.toUpperCase()
  );
})
```

Map

What could go wrong here?

I could forget to create my array for results

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.toUpperCase();
  );
})
```

I could forget to add the result to array of results

Map

It's not readable enough

I could forget to create my array for results

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.toUpperCase());
  );
})
```

Obscures what's happening!

I could forget to add the result to array of results

Map

It’s not readable enough

```
var friends = ["jane", "john", "joe"];

var capNames = friends.map(function (friend) {
    friend.toUpperCase();
});
```

Map

Imagine we had accidentally used the wrong method name to capitalize

```
var friends = ["jane", "john", "joe"];

var capNames = friends.map(function (friend) {
    friend.capitalize();
});
```

Map

Now you're even less likely to spot it quickly.

```
var friends = ["jane", "john", "joe"];
var capNames = [];

friends.forEach(function (friend) {
  capNames.push(
    friend.capitalize()
  );
})
```

Map

- Utilize map to multiply all the numbers in an array by two.
  - [1, 2, 3, 4, 5]; => [2, 4, 6, 8, 10]
- Utilize map to subtract 1 from every number
  - [1, 2, 3, 4, 5]; => [0, 1, 2, 3, 4]
- Utilize map to multiply all the numbers in an array and then subtract one
  - [1, 2, 3, 4, 5]; => [1, 3, 5, 7, 9]

Map

- **Recall or re-write your fullName function**
  - **Use fullName and map to iterate over list of name objects.**
    -

```
var names = [
  { first: "jane", last: "doe" },
  { first: "john", last: "doe" }
];
```

Filtering

Imagine we had to go through a list of items and remove certain ones not containing a prefix

```
var names = [
  "bob",
  "jane",
  "john",
  "jack",
  "joe",
  "janet"
];
```

Filtering

Imagine we had to go through a list of items and remove certain ones not containing a prefix

```
var matchingNames = [];
var prefix = "ja";

names.forEach(function (name) {
  if (name.startsWith(prefix) ) {
    matchingNames.push(name);
  }
});
```

Filtering

I think we already see the problem… the only that matters here:

```
name.startsWith(prefix)
```

Filtering

Using filter we can remove items that don't pass our test by just returning true or false.

```
var prefix = "ja";

names.filter(function (currentName) {
  return currentName.startsWith(prefix);
});

// =>  ["jane", "jack", "janet"]
```

Filtering

Your turn..

Modify the previous filtering of names so it doesn’t care about upper or lower case letters.

Assessment

- In a new js bin use the following names to complete this exercise
  - Console log each name
  - Document.write each name

```
var names = [
  "bob",
  "jane",
  "john",
  "jack",
  "joe",
  "janet"
];
```

Assessment

- In a new js bin use the following names to complete this exercise
  - Console log each name
  - Document.write each name

Assessment

- In a new js bin use the following names to complete this exercise
  - Use for each to count the number of names that start with “ja”

```
var names = [
  "bob",
  "jane",
  "john",
  "jack",
  "joe",
  "janet"
];
```

Assessment

In a new JS bin or Code Pen

- **Write a function called average that takes in a collection of numbers and returns their average.**

Assessment

- In a new js bin use the following names to complete this exercise
  - Write a function called `find`
    - `It takes a searchName and a list of names`
    - Use forEach to iterate through an array of names and return the index of the first match.

Assessment

- In a new js bin ...
  - Write a function called `findProp`
    - That takes a property name and a list of objects
    - Return the first object that has the property.

Assessment

- In a new js bin...
  - Write a function called `averageAge`
    - `It takes a list of objects with age properties`
    - `Return the average of their ages.`

Assessment

- In a new js bin...
  - Write a function called upcaseAll
    - It should take a list of names
    - Return all the upper cased values of the strings
      - [“jane”, “joe”] => [“JANE”, “JOE”]

Assessment

- In a new js bin...
  - Write a function called squareAll
    - It should take an array of numbers
    - Returns the array of numbers with all the values squared

Assessment

- In a new js bin...
  - Write a function called createItems
    - It should take an array of strings
    - Returns each string wrapped in a “
” and “”
      - [“jane”, “joe”, “john”] => [“
jane”, “
joe”, “
john”]