Initializing Instances

- Due No Due Date
- Points 1
- Submitting a website url

[(https://github.com/learn-co-curriculum/phase-1-initializing-instances)](https://github.com/learn-co-curriculum/phase-1-initializing-instances) [(https://github.com/learn-co-curriculum/phase-1-initializing-instances/issues/new)](https://github.com/learn-co-curriculum/phase-1-initializing-instances/issues/new)

Learning Goals

- Create a class in JavaScript using the class keyword
- Provide a JavaScript class's constructor instantiation data for an instance

Introduction

In this lab, we are going to practice creating Object-Oriented classes and instances using JavaScript's `class` keyword. We are also going to use JavaScript's class `constructor` to instantiate data into a class. In other words, we're going to create a class, and then put some data in it.

Create a class in JavaScript using the class keyword

Remember, when we want to create a class in JavaScript, we want to create it using the `class` keyword:

```
class Dog {}
```

Provide a JavaScript class's `constructor` instantiation data for an instance

Within our class, JavaScript class's `constructor` allows us to pass data to our new class.

```
class Dog {
  constructor(name, breed) {
    this.name = name;
```

```
      this.breed = breed;
    }
  }
```

You could then create data for the class above by doing something like the following:

```
let bigFluffyDog1 = new Dog("Buzz", "greatPyrenees");
let bigFluffyDog2 = new Dog("Woody", "labrador");

bigFluffyDog1; // => Dog { name: 'Buzz', breed: 'greatPyrenees' }
bigFluffyDog2; // => Dog { name: 'Woody', breed: 'labrador' }
```

Instructions

To practice OOP concepts, let's create 3 classes that use constructor methods. These constructors will assign properties based on initial parameters.

1. Create a class for `Breakfast` . `Breakfast` will have a constructor with a food and a drink.
2. Create a class for `Lunch` . `Lunch` will have a constructor with a salad, a soup, and a drink.
3. Create a class for `Dinner` . Since dinner is a little bit fancier, `Dinner` will have a constructor with salad, soup, entree, and dessert. Initialize dessert as a private property by prefixing its name with the hash symbol (`#`).

Note: Recall from the last lesson that you need to declare private fields before assigning them a value in the constructor, e.g.:

```
class Transaction {
  #amount;
  constructor(amount, date, memo) {
    this.#amount = amount;
    this.date = date;
    this.memo = memo;
  }
}
```

If you don't declare the private fields correctly, you'll get a syntax error with a long message like this when running the tests:

```
/phase-1-initializing-instances/node_modules/@babel/core/lib/parser/index.js:93
    throw err;
    ^

SyntaxError: /phase-1-initializing-instances/index.js: Private name #dessert is not defined. (4:9)
```

Conclusion

By effectively creating classes and instances, you have mastered the beginning of Object-Oriented JavaScript. Feel free to play around with your newly created classes to create more instances!

Resources

- **Classes (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Classes)**