Bees - get ready to learn some science, and ES6

Bees (the most noble of insects) progress through specific developmental stages over the course of their lifetimes. Each phase has specific traits and behaviors associated with it that are unique to that lifestage. In this repo, you will create bees of many lifestages while practicing the new class syntax that ES6 instatiation pattern, inheritance, and subclassing.

Class Structure

├── Grub

│ └── Bee

│ ├── HoneyMakerBee

│ └── ForagerBee

│ └── RetiredForagerBee

As you can see in the diagram above, all bees start out as a Grub, grow into a Bee and are then assigned its specific jobs and tasks among its other charactersitics. This means that characteristics are inherited from the classes above (although they can be modified on the current class level).

Resources

* [ES6 Classes in Depth](https://ponyfoo.com/articles/es6-classes-in-depth)

**Bare minimum requirements**

Bower

* Don't worry for now about how Bower works. Just know that when bower install is run from the command line, the packages listed in bower.json will be automatically installed to client/bower\_components.
  + If you don't have npm installed on your system, install it by running brew install node
  + Install Bower by running the following command npm install -g bower.
  + Use bower to install the client-side dependancies by running the following command from the root directory of the repo: bower install.

ES6

The ECMAScript specification is a scripting language specification upon which JavaScript implementations (such as those found in web browsers like Chrome) are based. In June 2015, the 6th edition of the ECMAScript standard was finalized, and is commonly referred to as ES6.

ES6 introduces a wealth of new features to JavaScript while being entirely reverse-compatible with older JavaScript. Even the most popular of web browsers like Chrome have a ton of work to do before all ES6 features are available, however, a lot of developers are using ES6 features and you should look forward to seeing more and more of ES6 in the next several years.

One exciting feature is the inclusion of a class keyword! This is a big change for engineers who write object-oriented code because it makes JavaScript look and feel more like a traditional class based language even though it's really just syntactic sugar on top of the pseudoclassical instantiation pattern you are familiar with. ES6 makes subclassing much easier as well with the introduction of the extendskeyword. This sprint is designed to get you comfortable with this new instantiation pattern.

Subclassing

Build your first class in Grub.js. Grub will act as the superclass for all other types of bees. Work through the repo in the order specified below, making sure to pass all of the specs in test/ - which you can run by typing npm test in your terminal.

Things To Note:

* This assignment must be written with ES6 classes by using the new class, extends, and super keywords
* Although there are multiple tests for each class, you will only be able to see one test at a time. As soon as one test fails the spec runner stops running and you must pass the current test to continue.

Grub

* Create a Grub class, in ES6 style, with:
  + an age property that is set to 0
  + a color property that is set to pink
  + a food property that is set to jelly
  + an eat method that returns 'Mmmmmmmmm jelly'

Bee

* Create a Bee class, in ES6 style, with:
  + the Grub superclass
  + an age property that is set to 5
  + a color property that is set to yellow
  + a food property that is inherited from grub
  + an eat method that is inherited from grub
  + a job property that is set to Keep on growing

HoneyMakerBee

* Create a HoneyMakerBee class, in ES6 style, with:
  + the Bee superclass
  + an age property that is set to 10
  + a job property that is set to make honey
  + a color property inherited from bee that is set to yellow
  + a food property that is inherited from grub
  + an eat method that is inherited from grub
  + a honeyPot property that is set to 0
  + a makeHoney method that adds 1 to that honeyBee's honeyPot
  + a giveHoney method that subtracts 1 from that honeyBee's honeyPot

ForagerBee

* Create a ForagerBee class, in ES6 style, with:
  + the Bee superclass
  + an age property that is set to 10
  + a job property that is set to find pollen
  + a color property inherited from bee that is set to yellow
  + a food property that is inherited from grub
  + an eat method that is inherited from grub
  + a canFly property that is set true
  + a treasureChest property that is set to an empty array []
  + a forage method that allows the bee to add a treasure to the treasureChest

RetiredForagerBee

* Create a RetiredForagerBee class, in ES6 style, with:
  + the ForagerBee superclass
  + an age property that is set to 40
  + a job property that is set to gamble
  + a canFly property that is set to false
  + a color property that is set to grey
  + a forage method that returns I am too old, let me play cards instead
  + a food property that is inherited from grub
  + an eat method that is inherited from grub
  + a treasureChest property inherited from ForagerBee that is set to an empty array []
  + an always winning gamble method that allows the bee to add a treasureto the treasureChest