Ajax Lab

At this point you have a page that displays a list of persons. But this list is hardcoded, being included and read from a .json file. Let's read those persons from an API by making an Ajax call.

Examining the API

1. First, open a browser and navigate to https://randomuser.me/api/?results=5
2. Take a look at what is being returned. It may be helpful to open the browser's developer tools and study the network tab. If you notice, the value coming back is pure JSON. Remember that URL because we'll come back and use it in just a minute.

Crafting a call to the API

3. Open your index.js file and find getPersons(). Right now it is running a setTimeout() to simulate a long-running process. Well, guess what? We're going to replace it with a real-life long-running proces -- no simulations needed.
4. Comment out or just remove the whole promise including the setTimeout and everything.
5. Instead, do a fetch from the URL we visited above.
6. Remember from the lecture that fetch returns a promise. Go ahead and return that same promise from getPersons().
7. Since your main function handles a promise already, you can just recompile and run and it will process the call. Unfortunately, the data being returned needs to be processed before it can show up.

Processing the API response

8. After your fetch call, do this:

```
.then(res => res.json())
```

9. That will convert the request string to a true JSON object.
10. Run and test again.
11. Still not working? Take a look in the developer tools again and examine the data being returned. You should have an object that has our list of people inside it. What is the name of that key? ____
12. Chain another .then() function that will take the object you saw in the step above and return only the property you discovered above. (Hint: you'll add one line of code and it will be another .then())
13. Run and test. If you've done it right, every time you refresh, you'll see a new list of random people.

14. Bonus! Change your getPersons method to receive in a "numberOfPeople" parameter with a default of 5. Use it in getPersons to fetch that number of people instead of the hardcoded number you have in there already.

Bonus!! Doing it server-side

Got extra time? Want a challenge? Let's do it in node.

15. npm install node-fetch.
16. Open (or create) a node script in the IDE.
17. require('node-fetch') at the top.
18. Use it to fetch a set number of people from randomuser.me like above.
19. Process the promise returned and display the results in the console.