COMP6080 WK5 Tutorial 🗺️

Joanna He

Agenda

- Week 4 overview
- Setting up assignment 3 locally
- Promises (but there's more 👀)
- Demo

Assignment 3 (slackr)

is released 🎊🎉

Helpful resources for Ass03

- COMP6080 Style Guide
- MDN Web Docs
- Resources from last week's slides

Any questions?

Last week we covered...

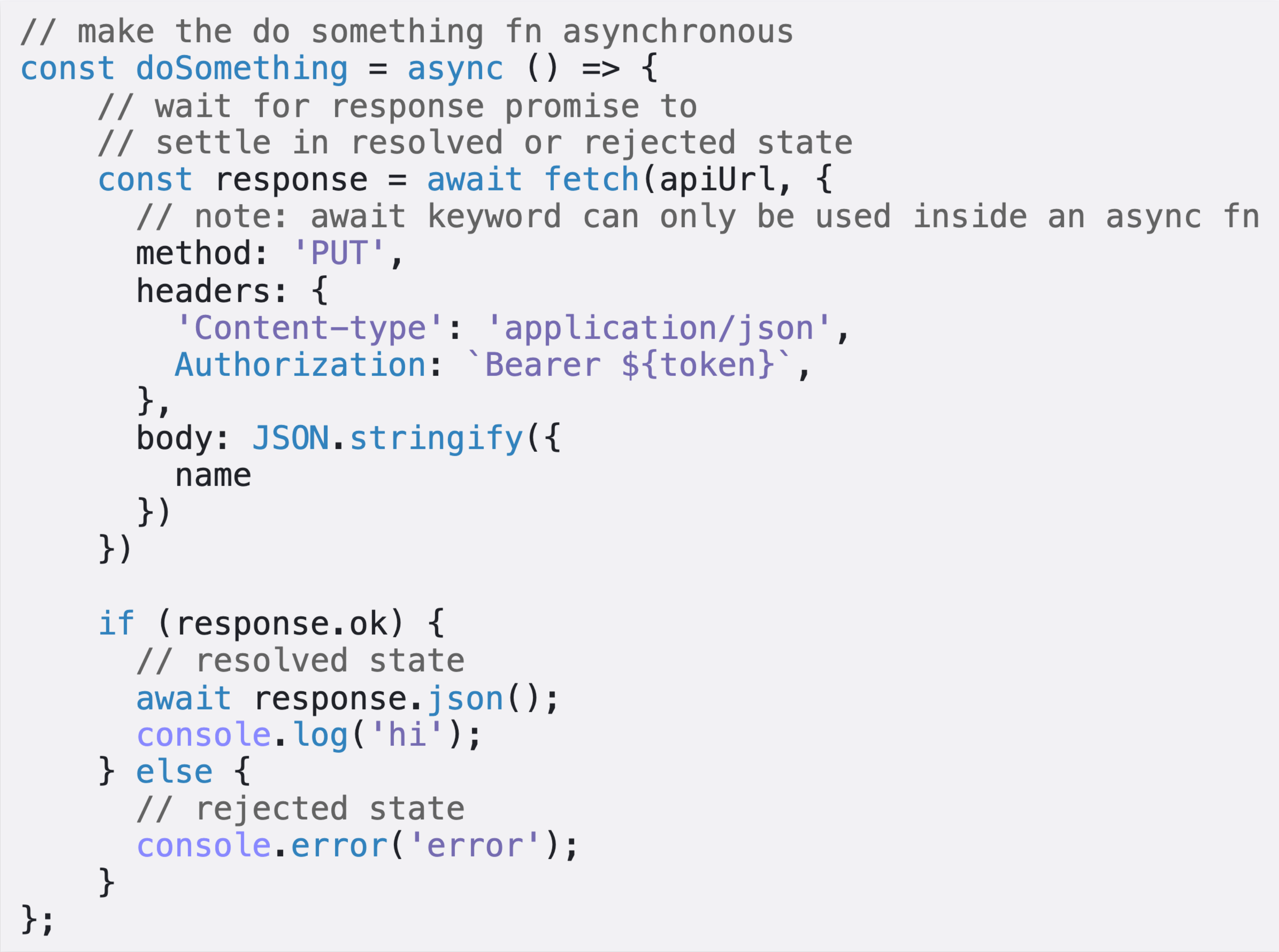
- JavaScript
- Data fetching
- Promises

Onto promises...

```
const doSomething = () => {
  fetch(apiUrl, {
      method: "PUT",
      headers: {
          "Content-type": "application/json",
          "Authorization": `Bearer ${userToken}`
      },
      body: JSON.stringify({
          name
      }),
  })
  .then((res) => res.json());
  .then((data) => {
      if (!data.ok) console.log("hi");
  })
  .catch((error) => console.error("hi"));
};
```

I don't want to `.then()`?

- You can use `async` and `await` keywords
    - `async` - creates binding of async fn to a given name
    - `await` - wait for promise to settle in either a resolved or rejected state
- Note: `async`/`await` is **banned** in assignment 3, use `.then()`/`.catch()` instead



```
// make the do something fn asynchronous
const doSomething = async () => {
    // wait for response promise to
    // settle in resolved or rejected state
    const response = await fetch(apiUrl, {
      // note: await keyword can only be used inside an async fn
      method: 'PUT',
      headers: {
        'Content-type': 'application/json',
        Authorization: `Bearer ${token}`,
      },
      body: JSON.stringify({
        name
      })
    })

    if (response.ok) {
      // resolved state
      await response.json();
      console.log('hi');
    } else {
      // rejected state
      console.error('error');
    }
};
```

Demo

Let's refactor our pokemon fetcher to use `async` / `await` and our favourite CSS library 🔥

Tutorial code can be found at

https://github.com/joanna209/tutoring/tree/main/comp6080/23T3