15-Short-lnk-app - video

Creating and calling methods

- to define our methods we are going to define something at bottom of the page
- it is not getting added to the Meteor.isServer
- they have to be defined in both the server and the client
- Meteor.methods({}) it takes one argument
- we are going to be making a very simple method to start
- so in links.js lets do this method

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    greetUser: function(){

    }
});
```

- if you notice above we are going to be using ES5 functions because we are going to be using the this binding
- but since we are difining it on an object we can take advantange of that ES6 object method syntax
- so it's going to look like this

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
```

```
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    greetUser(){

    }
});
```

- lets do an example

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    greetUser(){
        console.log('greetUser is running');

        return 'Hello user!'
    }
});
```

- now lets figure out how to call this from the server and client
- we are going to be using Meteor.call to call our methods
- so lets go to server/main.js

```
import { Meteor } from 'meteor/meteor';

import '../imports/api/users';
import '../imports/api/links';
```

```
Meteor.startup(() => {
  Meteor.call();
});
```

- Meteor.call() takes a few arguments, the first one is the method name as a string, from there you can also provide a call back function, this is going to let us know wether the method success or fails

```
import { Meteor } from 'meteor/meteor';

import '../imports/api/users';
import '../imports/api/links';

Meteor.startup(() => {
  Meteor.call('greetUser', ()=>{

  });
});
```

- and our function takes two argument error and the result

```
import { Meteor } from 'meteor/meteor';

import '../imports/api/users';
import '../imports/api/links';

Meteor.startup(() => {
  Meteor.call('greetUser', (err, res)=>{

  });
});
```

- we are going to console.log the arguments

```
import { Meteor } from 'meteor/meteor';

import '../imports/api/users';
import '../imports/api/links';
```

```
Meteor.startup(() => {
  Meteor.call('greetUser', (err, res)=>{
    console.log('Greet User Arguments', err, res);
  });
});
```

- lets see the results in the terminal

```
I20171018-09:23:02.089(-7)? greetUser is running
I20171018-09:23:02.098(-7)? Greet User Arguments undefined Hello user!
=> Meteor server restarted
```

- so as you can see this Meteor call was a success because err was undefined, and Hello user was shown
- which was the return value from our method from links.js
- now we are going to take Meteor.call and cut them and paste them on client/main.js

```
import { Meteor } from 'meteor/meteor';
import ReactDom from 'react-dom';
import { Tracker } from 'meteor/tracker';

import { routes, onAuthChange } from '../imports/routes/routes';



Tracker.autorun(()=>{
  const isAuthenticated = !!Meteor.userId();
  onAuthChange(isAuthenticated);
});

Meteor.startup(()=>{
  Meteor.call('greetUser', (err, res)=>{
    console.log('Greet User Arguments', err, res);
  });
  ReactDom.render(routes,  document.getElementById('app'));
});
```

- over in our terminal we see...

```
I20171018-09:27:52.144(-7)? greetUser is running
```

- **our client side method calls are essentially remote calls which basically means it tells some other computer to run some sort of method, in our case the client is telling the server to go ahead and run greetUser, that is why we are seeing the above log, this is how methods can be secure**
- **now over in the browser you will see this same log**

```
greetUser is running

Greet User Arguments undefined Hello user!
```

- Meteor methods can take other arguments right before the (err, res)
- so lets try it in the client/main.js

```
import { Meteor } from 'meteor/meteor';
import ReactDom from 'react-dom';
import { Tracker } from 'meteor/tracker';

import { routes, onAuthChange } from '../imports/routes/routes';



Tracker.autorun(()=>{
  const isAuthenticated = !!Meteor.userId();
  onAuthChange(isAuthenticated);
});

Meteor.startup(()=>{
  Meteor.call('greetUser', 'Mike' ,(err, res)=>{
    console.log('Greet User Arguments', err, res);
  });
  ReactDom.render(routes,  document.getElementById('app'));
});
```

- now over to links.js where our Method is we are going to be adding the name parameter
- and we can give it a default value, in our case we have Mike, so Mike will show, but if no arguments are provided, User will appear
- so finallly in the return statement we use template string and inject our name variable

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    greetUser(name="User"){
        console.log('greetUser is running');

        return `Hello ${name}!`
    }
});
```

- and now if we go to the browser we will see our result

```
Greet User Arguments undefined Hello Mike!
```

- now lets throw an error if there is no name provided- we are going to be throwing a Meteor error
- Meteor.Error takes two arguments, the first is the name of the error, and the second is the reason of the error
- lets also remove the default value for name

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
```

```
    greetUser(name){
        console.log('greetUser is running');
        if(!name){
            throw new Meteor.Error('invalid-arguments', 'Name is required');
        }
        return `Hello ${name}!`
    }
});
```

- now over to the Meteor.call in the client/main.js - remove Mike from the arguments

```
import { Meteor } from 'meteor/meteor';
import ReactDom from 'react-dom';
import { Tracker } from 'meteor/tracker';

import { routes, onAuthChange } from '../imports/routes/routes';



Tracker.autorun(()=>{
  const isAuthenticated = !!Meteor.userId();
  onAuthChange(isAuthenticated);
});

Meteor.startup(()=>{
  Meteor.call('greetUser',(err, res)=>{
    console.log('Greet User Arguments', err, res);
  });
  ReactDom.render(routes,  document.getElementById('app'));
});
```

- now over in the browser you should see two same errors, one is coming from the client and the other one from the server
- this was my challenge:
- which was in links.js

```
Meteor.methods({
    addNumber(a, b){
        if(typeof a === "number" && typeof b === "number"){
```

```
            return a + b;
        }else{
            throw new Meteor.Error('Invalid Type', 'Use numbers only');
        }
    }
});
```

- and over in the client/main.js

```
Meteor.startup(()=>{
  // Meteor.call('greetUser',(err, res)=>{
  //   console.log('Greet User Arguments', err, res);
  // });
  Meteor.call('addNumber', 2, "4", (err, res)=>{
    console.log('this is my result: ', res, 'this si my error: ', err)
  })
  ReactDom.render(routes,  document.getElementById('app'));
});
```

- the above will cause an error because I passed a string as argument, but when switch it to number is works
- and this is how the teacher solved it

```
Meteor.methods({
    addNumber(a, b){
        if(typeof a !== "number" || typeof b !== "number"){
            throw new Meteor.Error('Invalid Type', 'Use numbers only');
        }
        return a + b;

    }
});
```