30-short-lnk-app video

Working with Time

- lets go server/main.js to play around with the Date object

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

import '../imports/api/users';
import { Links } from '../imports/api/links';
import '../imports/startup/simple-schema-configuration.js';

Meteor.startup(() => {

    const now = new Date();
    console.log(now);
```

```
I20171026-10:47:36.838(-7)? Thu Oct 26 2017 10:47:36 GMT-0700 (PDT)
```

- to simplify Date we are going to be using a library called moment that will make things easier for us
- to explore this go to https://www.npmjs.com/package/moment
- to check the documentation go to http://momentjs.com/
- lets go ahead and install it in the terminal

```
meteor npm install moment@2.17.1 --save
```

- now lets import it in server/main.js
- now lets create a new instance of moment
- and we then are going to be using the format()

```
import { Meteor } from 'meteor/meteor';
import { WebApp }from 'meteor/webapp';
import moment from 'moment';

import '../imports/api/users';
import { Links } from '../imports/api/links';
import '../imports/startup/simple-schema-configuration.js';


Meteor.startup(() => {


    const now = new Date();

    console.log(now);


    const momentNow = moment();

    console.log(momentNow.format());
```

- **and this** is what shows up in the terminal

```
I20171026-10:58:09.387(-7)? 2017-10-26T10:58:09-07:00
```

- **format** does take an argument string
- an example how to use this is in the doc:

```
moment().format();                                  // "2014-09-08T08:02:17-05:00" (ISO 8601)

moment().format("dddd, MMMM Do YYYY, h:mm:ss a"); // "Sunday, February 14th 2010, 3:25:50 pm"

moment().format("ddd, hA");                         // "Sun, 3PM"

moment('gibberish').format('YYYY MM DD');          // "Invalid date"
```

- we are going to be using the following format for this project

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

import { WebApp }from 'meteor/webapp';

import moment from 'moment';


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

import { Links } from '../imports/api/links';

import '../imports/startup/simple-schema-configuration.js';


Meteor.startup(() => {


    const now = new Date();

    console.log(now);
    const momentNow = moment();
    console.log(momentNow.format('MMM Do, YYYY'));
```

- **and this** is the result in the terminal:

```
I20171026-11:03:02.551(-7)? Oct 26th, 2017
```

- **and now we are** going to be doing the time

```
import { Meteor } from 'meteor/meteor';
import { WebApp }from 'meteor/webapp';
import moment from 'moment';

import '../imports/api/users';
import { Links } from '../imports/api/links';
import '../imports/startup/simple-schema-configuration.js';

Meteor.startup(() => {

    const now = new Date();
    console.log(now);

    const momentNow = moment();
    console.log(momentNow.format('MMM Do, YYYY'));
    console.log(momentNow.format('h:mma'));
```

- **and** the result over in the terminal...

```
I20171026-11:07:17.091(-7)? 11:07am
```

- **for** our project we are going to be using the Time for now from the moment library

```
moment([2007, 0, 29]).fromNow();     // 4 years ago
moment([2007, 0, 29]).fromNow(true); // 4 years
```

- in our app we are going to be passing 0 to moment(0) so it starts from 1969 and then add the fromNow() method

```
import { Meteor } from 'meteor/meteor';
import { WebApp }from 'meteor/webapp';
import moment from 'moment';

import '../imports/api/users';
import { Links } from '../imports/api/links';
import '../imports/startup/simple-schema-configuration.js';

Meteor.startup(() => {

    const now = new Date();
    console.log(now);

    const momentNow = moment(0);
    console.log(momentNow.fromNow());
```

- and this is the result

```
I20171026-11:15:23.249(-7)? 48 years ago
```

- **now l**ets go to LinksListItem
- and we are going to be modifying the line below

```
    render(){
        return(
            <div>
                <p>{this.props.url}</p>
                <p>{this.props.shortUrl}</p>
                <p>{this.props.visible.toString()}</p>
                <p>{this.props.visitedCount} - {this.props.lastVisitedAt}</p>
                <button ref="copy" data-clipboard-text={this.props.shortUrl}>
                {this.state.justCopied ? 'Copied' : 'Copy'}
                </button>
                <button onClick={()=>{
                    Meteor.call('links.setVisibility', this.props._id, !this.props.visible);
                }}>{this.props.visible ? 'Hide' : 'Unhide'}</button>
            </div>
        )
    }
}
```

-

```
    render(){
        return(
            <div>
                <p>{this.props.url}</p>
                <p>{this.props.shortUrl}</p>
                <p>{this.props.visible.toString()}</p>
                {this.renderStats()}
                <button ref="copy" data-clipboard-text={this.props.shortUrl}>
                {this.state.justCopied ? 'Copied' : 'Copy'}
                </button>
                <button onClick={()=>{
                    Meteor.call('links.setVisibility', this.props._id, !this.props.visible);
                }}>{this.props.visible ? 'Hide' : 'Unhide'}</button>
            </div>
        )
    }
}
```

- and now lets define that renderStats() method up above

```
   renderStats(){
       return <p>{this.props.visitedCount} - {this.props.lastVisitedAt}</p>
   }
```

- lets import moment in our file first
- now lets start modifying renderStats
- the first thing we need to do is add the words visit or visit
- we are going to be using a ternary operator to determine wheater is plural or singular

```
   renderStats(){
       const visitMessage = this.props.visitedCount === 1 ? 'visit' : 'visits';
       return <p>{this.props.visitedCount} {visitMessage}- {this.props.lastVisitedAt}</p>
```

```
    }
```

- now lets create another variable but this time use let, because its value will change

```
    renderStats(){
        const visitMessage = this.props.visitedCount === 1 ? 'visit' : 'visits';
        let visitedMessage = null;
        return <p>{this.props.visitedCount} {visitMessage}- {visitedMessage}</p>
    }
```

- and now we are going to be updating the visitedMessage variable with an if statement
- we are going to be checking if the lastVisitedAt is a number

```
    renderStats(){
        const visitMessage = this.props.visitedCount === 1 ? 'visit' : 'visits';
        let visitedMessage = null;

        if(typeof this.props.lastVisitedAt === 'number'){

        }
```

- and now we are going to incorporate the moment library like we did before

```
  renderStats(){
        const visitMessage = this.props.visitedCount === 1 ? 'visit' : 'visits';
        let visitedMessage = null;
        const momentNow = moment(this.props.lastVisitedAt);

        if(typeof this.props.lastVisitedAt === 'number'){
             visitedMessage = `(visited ${momentNow.fromNow()})`;
        }

        return <p>{this.props.visitedCount} {visitMessage} {visitedMessage}</p>
    }
```

-