transfer state forward and back

transfer state forward and back

mongo ds129053.mlab.com:29053/web322\_a71 -u <dbuser> -p <dbpassword>

mongorestore -h ds129053.mlab.com:29053 -d web322\_a71 -u jlee465 -p Vkvkdi0^

mongodb://user:password123@ds121014.mlab.com:21014/team-api-test

npm install

git init

heroku login

create heroku

git push master heroku

data-service.js --> closer

populate(join?????)

app.use(cors()); \*\*\*\*\*\*

jquery

$(function() { // this is the same as: $(document).ready(function(){

jquery-ajax

let tbody = $("#employees-table tbody");

let row = $('<tr>').append(fNameTD).append(lNameTD).append(positionTD); // create the row and append all of the TD elements.

## Moment.js

[Moment.js](https://momentjs.com/) is a JavaScript utility library designed to make the handling of date / datetime objects much simpler. It encapsulates a native Date object in order to provide extra manipulation, querying, display, validation and localization options.

### Downloading Moment.js

Moment.js exists as a single, minified .js file that can either be [downloaded](https://momentjs.com/downloads/moment-with-locales.min.js) (with or without all locales) and included in your local solution, or referenced using [the CDN](https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.18.1/moment-with-locales.min.js) (with or without all locales) in your pages/views. It is also available [via NPM](https://www.npmjs.com/package/moment) and can be used in your Node.js server applications as well.

### The Full Documentation

We will not be covering every faset of the Moment.js library during this lecture, however it is an extremely valuable resource and we recommend [**Bookmarking the Documentation**](https://momentjs.com/docs/) for future reference.

### Working with Moment.js

The following core pieces of functionality will provide a solid foundation for working with Moment.js in the future. For the below examples, we will be using the string representation of Zsa zsa Mannering’s start date, ie:

let hireDate = "2010-11-07T04:00:00.000Z"; // Zsa zsa Mannering's hire date in our "Employees" Collection

If you’re following along with the “Code Samples” the following functionality is located in the “week2/Moment” folder 

### Setting the “Locale”

*// moment.locale("fr-CA"); // Try out Quebec French*

### Creating a “Moment”

**let** mDate **=** moment(hireDate); *// create a new "moment" object*

### Setting UTC Mode & Displaying Data

mDate.utc(); *// switch to "UTC" mode*

*// display the UTC date*

**let** mDate1 **=** mDate.format('LLLL'); *// Sunday, November 7, 2010 4:00 AM*

### Setting “Local” Mode & Displaying Data

mDate.local(); *// switch to "local" mode*

*// display a localized date (now offset to Local Time)*

**let** mDate2 **=** mDate.format('LLLL'); *// Sunday, November 7, 2010 12:00 AM*

## Display Format Options

**let** mDate3 **=** mDate.format('MMMM Do YYYY, h:mm:ss a'); *// November 7th 2010, 12:00:00 am*

**let** mDate4 **=** mDate.format('dddd'); *// Sunday*

**let** mDate5 **=** mDate.format("MMM Do YYYY"); *// Nov 7th 2010*

**let** mDate6 **=** mDate.format('[date:] MM/DD/YYYY'); *// date: 11/07/2010*

**let** mDate7 **=** mDate.format(); *// 2010-11-07T00:00:00-04:00*

For a full list of formatting options, see: <https://momentjs.com/docs/#/displaying/format>

### Display Format Using Locale Options

**let** mDate8 **=** moment.locale(); *// en (or fr if testing Quebec french)*

**let** mDate9 **=** mDate.format('LT'); *// 12:00 AM*

**let** mDate10 **=** mDate.format('LTS'); *// 12:00:00 AM*

**let** mDate11 **=** mDate.format('L'); *// 11/07/2010*

**let** mDate12 **=** mDate.format('l'); *// 11/7/2010*

**let** mDate13 **=** mDate.format('LL'); *// November 7, 2010*

**let** mDate14 **=** mDate.format('ll'); *// Nov 7, 2010*

**let** mDate15 **=** mDate.format('LLL'); *// November 7, 2010 12:00 AM*

**let** mDate16 **=** mDate.format('lll'); *// Nov 7, 2010 12:00 AM*

**let** mDate17 **=** mDate.format('LLLL'); *// Sunday, November 7, 2010 12:00 AM*

**let** mDate18 **=** mDate.format('llll'); *// Sun, Nov 7, 2010 12:00 AM*

## Additional Display Options:

Note: the display options for the below methods are:

* ‘seconds’
* ‘minutes’
* ‘hours’
* ‘days’
* ‘weeks’
* ‘months’
* ‘years’

### .fromNow()

A common way of displaying time is handled by moment#fromNow. This is sometimes called timeago or relative time.

See: <https://momentjs.com/docs/#/displaying/fromnow/>

**let** mDate19 **=** mDate.fromNow(); *// 7 years ago*

### .diff()

To get the difference in milliseconds, use moment#diff like you would use moment#from. To get the difference in another unit of measurement, pass that measurement as the second argument.

See: <https://momentjs.com/docs/#/displaying/difference/>

**let** mDate21 **=** mDate.diff(moment([2010,0,1]), 'days'); *// 310*

**let** mDate22 **=** mDate.diff(moment([2010,0,1]), 'months'); *// 10*

**let** mDate23 **=** mDate.diff(moment([2010,0,1]), 'years'); *// 0*

### .toISOString()

Formats a string to the ISO8601 standard.

See: <https://momentjs.com/docs/#/displaying/as-iso-string/>

**let** mDate24 **=** mDate.toISOString(); *// 2010-11-07T04:00:00.000Z (the same as what we started with)*

### .daysInMonth()

Get the number of days in the current month.

See: <https://momentjs.com/docs/#/displaying/days-in-month/>

**let** mDate25 **=** mDate.daysInMonth(); *// 30*

## Manipulate Options:

The following methods are used to manipulate the working date.

### .add()

Mutates the original moment by adding time.

See: <https://momentjs.com/docs/#/manipulating/add/>

mDate.add(5, 'days'); *// Using: .format("LLLL") => Friday, November 12, 2010 12:00 AM*

### .subtract()

Mutates the original duration by subtracting time.

See: <https://momentjs.com/docs/#/durations/subtract/>

mDate.subtract(5, 'days'); *// Using: .format("LLLL") => Sunday, November 7, 2010 12:00 AM*

### .utcOffset()

Get the UTC offset in minutes.

See: <https://momentjs.com/docs/#/manipulating/utc-offset/>

**let** mDate26 **=** mDate.utcOffset(); *// 240 (minutes)*

## Query Options:

The following methods are used to compare and examime dates

### .isBefore()

Check if a moment is before another moment.

See: <https://momentjs.com/docs/#/query/is-before/> and also: <https://momentjs.com/docs/#/query/is-after/>

**let** mDate27 **=** mDate.isBefore(moment([2009,11,7])); *// false*

### .isSame()

Check if a moment is the same as another moment.

See: <https://momentjs.com/docs/#/query/is-same/>

**let** mDate28 **=** mDate.isSame(moment([2010,10,8])); *// false*

**let** mDate29 **=** mDate.isSame(moment([2010,10,8]), "month"); *// true*

### .isBetween()

Check if a moment is between two other moments, optionally looking at unit scale (minutes, hours, days, etc). The match is exclusive.

See: <https://momentjs.com/docs/#/query/is-between/>

**let** mDate30 **=** mDate.isBetween(moment([2010,0,1]), moment()); *// true*

### .isDST()

Checks if the current moment is in daylight saving time.

See: <https://momentjs.com/docs/#/query/is-daylight-saving-time/>

**let** mDate31 **=** mDate.isDST(); *// true*

### .isLeapYear()

Returns **true** if that year is a leap year, and **false** if it is not.

See: <https://momentjs.com/docs/#/query/is-leap-year/>

**let** mDate32 **=** mDate.isLeapYear(); *// false*

## Updating jquery-AJAX to show Dates

To see how we can use Moment to show dates for employees within our Teams API, open the **Moment-AJAX**Example located in the **week2** folder. We will walk through the solution together in class.

© 2017 - Seneca School of ICT