Intro - General

* Meteor is a full stack development framework / platform for developing real-time / reactive web and mobile applications.
* It uses an isomorphic model i.e. allows you to develop in one language, JavaScript, in all environments (server & client)
* Meteor is written using Node.js and integrates with Mongo DB. Its model allows for data to be sent to the client directly for a seamless and fast UI experience.
* To install Meteor on mac type the command below into your terminal. Which performs a download and install at once: *curl https://install.meteor.com/ | sh*
* To create a new app from destination folder run: *meteor create app\_name*
* To launch the app, from within folder run: *meteor*
* You can then access the app from <http://localhost:3000> a virtual web server running on your machine via port 3000
* While running the app will auto refresh the browser automatically as it monitors all edits
* After creating an app you should create the: ./public (static assets), ./lib (collections) and client/templates folders to keep all your code organized.
* The files in project folders are run in a specific order. First the lib folder, then main files the remaining files are run alphabetically.
* Code in the server folder runs only on server, client folder runs only in the client. All other folder code is accessible to both.
* To run a code block only in client or server you can also use the *Meteor.isCleint* or *Meteor.isServer* Boolean test
* To get direct access to your server side code run: meteor shell
* When you need code run at start up of app use: *Meteor.startup (function (){actions});*
* To stop the app use the Ctrl + C shortcut

Packages

* You can add helpful packages to your meteor app by running: *meteor add fullPackageName*
* To get a list of packages run: *meteor search* or navigate to atmosphere website and search
* To find out what packages and versions you have installed you can check the .meteor/packages and /releases files
* Meteor comes with base packages that are compulsory, 1st party packages that are key but can be removed and 3rd party packages (from Atmosphere) that can be added or removed at any time for additional functionality
* To add a meteor package use: *meteor add fullPackageName*, this will download and install the package
* To remove a meteor package use: *meteor remove fullPackageName*, this will uninstall the package
* Meteor can also use NPM packages which are in node.js, they don’t need to be installed but added to the packages.json file
* Twbs: bootstrap is a common package that is used as a front end framework

Templates

* Meteor template system is called Space Bars and consists of HTML, inclusions, expressions and block helpers
* Inclusions e.g. *{{>templateName}}* tell meteor to replace the inclusion with the named template and render it
* Expressions e.g. {{title}} call a property of a current object or return the value of a template helper
* Block helpers e.g. {{#each}} are special tags that control the flow of the template
* Template helpers are functions used to aid templates in rendering data / complex items. To pass data into a template using helpers use the format: *Template.templateName.helpers (data object);* you must create the data object first usually defined in a JavaScript file.
* Each template will need its own helper function and set of event handlers
* Alternatively *Template.templateName.helpers ({objectName: arrayName});* can be used for array cases

Collections

* Collections are special data structures that take care of storing data in Mongo DB and synch the data with each connected client in real-time.
* Collections can exist on the server where it reads and writes changes to the DB and on the client as a local subset copy of the data that is kept up to date in real-time
* To get direct access to your database from terminal run: meteor mongo
* Meteors client side implementation of Mongo DB is called Mini mongo
* You can create a new collection by: *new Mongo.Collection (“collectionName”); var keyword not used so that collections remain global and accessible by the whole app*
* How many items are in the collection: *collectionName.find (). count ();*
* To insert data into your DB use: *collectionName.insert (data object)*
* To wipe all the data from the DBs run: *meteor reset*
* To feed DB data into a template use: *Template.templateName.helpers ({objectName: collectionName.find ()});*
* To remove items from the DB using ID filter try: *collectionName.remove ({“\_id”: theId});*
* To update values in the DB try: *collectionName.update ({“\_id”: yourId}, {$set: {propertyName: yrValue}});*
* To sort results of DB query try: *collectionName.Find ({}, {sort: {propertyName: -1}});*
* The reactive object result of running a query on a mongo collection is called a cursor.

Pubs & Subs

* Publications allow you to tell Meteor which data subset can be sent to the client. This helps avoid sending whole DB to the client and protect any sensitive data.
* Subscriptions allows clients to specify which data subset they need in particular
* Distributed Data Protocol (DDP) main function is to query and update a server-side database and for it to synchronize such updates to all the clients.
* Meteor.publish on the server side makes data available to the client side but to receive that data the client needs to have Meteor.subscribe.