

Project

angularJS -js framework, create client/user interfaces

do a tutorial on top 10 concepts
slides/description on the technology

testing packages

what are the big components of given framework

nodeJS - js for serverside

reactJS
vueJS
angularJS
emberJS
meteorJS
nodal abs

python/flask

a little app, tutorial for it
slides that cover big concepts
test packages

git repo Storybooks storybook
<https://github.com/storybooks/storybook>

METEOR

platform for building single page applications SPA
-single page load
make spa without having to worry too much about infrastructure
package that combines all tools already used to make rich
javascript based spa in a simple and easy to use way

- dont need to write your own ajax code/manipulate dom
- access database directly from the browser

KEY CHARACTERISTICS

Same JS APIs on client and server
automatic updating of pages
eliminates need for REST APIs
Latency handling
Self-contained application bundles

TECHNOLOGIES IN METEOR

javascript

-used in client AND server, simplifies web development

node.js

- uses event driven nonblocking IO model
- usually, node apps handle requests on a single thread
- do those operations asynchronously
- meteor has different instances of node.js to handle different requests, single thread per request
- nodes default is asynchronous callbackstyle

mongoDB

-when writing to db on client, its not exactly the same as writing to db on server

handlebars

- library for creating template
- automatically updates DOM in your page

you can switch these out, for example, use a different DB or a different templating engine than handlebars

UNDERSTANDING METEOR

->HOW METEOR WORKS

usually we use jquery to manipulate the DOM to reflect any data changes, manually, Meteor does it for you.

in meteor we create html templates (using something like handlebars), if these templates use any data from the database, meteor automatically wraps the data that this template uses in some code that watches for changes on the data and creates new HTML markups that represents how the data should be displayed, this is done through a rendering function, that rendering function generates a document fragment that is part of the loaded html page in the DOM. meteor recalls the rendering function when data that the rendering function uses

changes, and the rendering function regenerates a doc fragment which is swapped out in the page. this happens automatically when using templates with meteor.

Meteor uses a protocol called DDP to communicate b/w client and server. publish and subscribe and remote procedure calls.

establish ddp connection->client subscribe to data on the server by sending subscribe command and the server can publish changes to the client as well.

ddp connection can also be used by client to make a remote procedure call to the server and get back a response from it.

ddp sends data via EJSON which is just an implementation of JSON.

Three important meteor concepts:
Collections, templating, and Publishing

model, view, controller

in meteor, a collection is a data structure that meteor knows how to synchronize b/w the client and server

```
db.tasks.insert({ text: "Hello world!", createdAt: new Date() });
```

Meteor is a js framework for building applications using node.js using whatever frontend framework you choose (such as angular or react, meteor's own default is something called Blaze). database is mongoDB by default and difficult to change. Platform for building node apps. It is a development platform, a collection of libraries and packages, that makes web development easier.

Realtime features, building realtime app couldn't be any faster. Big advantage with meteor is how nicely integrated its frontend is with its backend. Instead of having to write an API or interact with an API and have -ta separate frontend and backend, you can use things like Meteor methods where you define your serverside functionality on the server and then call those methods directly from client side, and not have to interact with an API. Client and server start to merge together.

Accounts and user authentication system is super easy, import packages, and now you have access to nice secure logins.

Its super fast to get up and running. Great for testing, you can directly add things to the database and see it show up in the frontend in real time immediately. during the testing and experimenting phase this is useful and fast. to be able to interact with the backend so readily with the front end before you cement the authentication and blocks.

no need to code a separate backend, its all part of one codebase.

How does meteor write to its database, using Meteor Methods.

You can think of them like a POST request to the server. A method is an API endpoint for your server, you can define a method on the server and its counterpoint on the client then call to it with some data, write to the db, and get a return value with a callback.