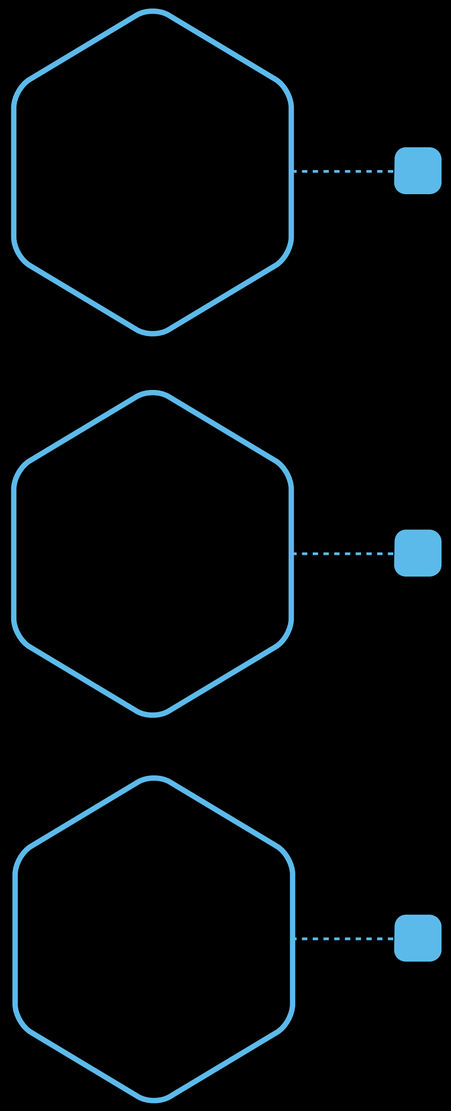
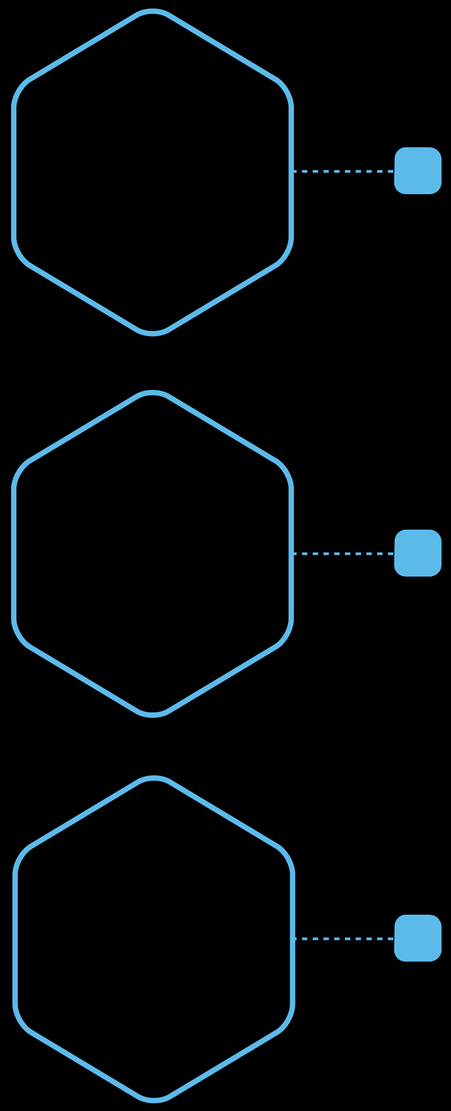
MODERN JAVASCRIPT WEB ARCHITECTURE

Pratik Patel @prpatel



- WHY?
- ECOSYSTEM
- TOOLS

@prpatel



- SERVER TO BROWSER ARCH
- FOCUS ON CLI TOOLS
- EXERCISES

@prpatel

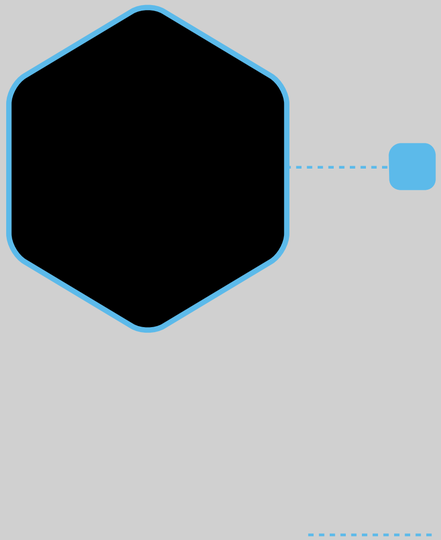
YESTERDAY'S JAVASCRIPT

@prpatel

TODAY'S JAVASCRIPT

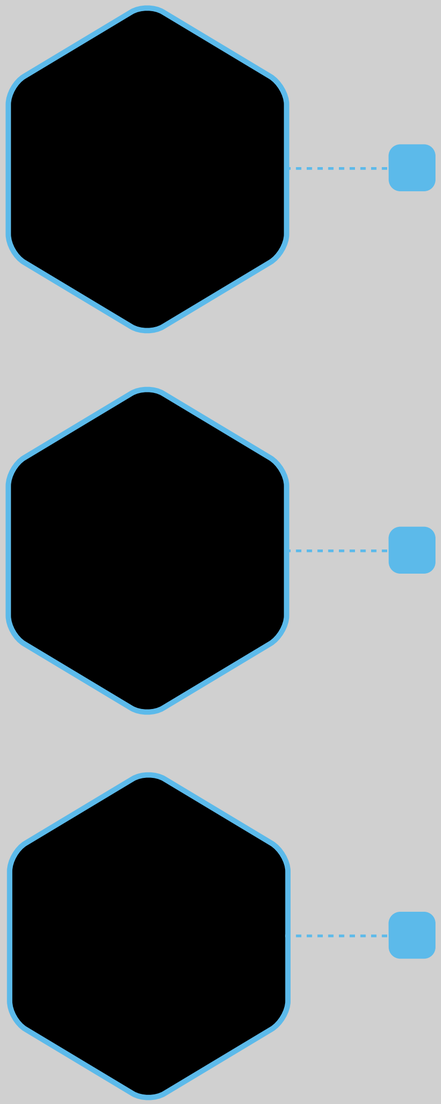
@prpatel

WHAT'S CHANGED?



JS IS THE SAME LANGUAGE IT WAS 20 YEARS AGO

NO CHANGE!

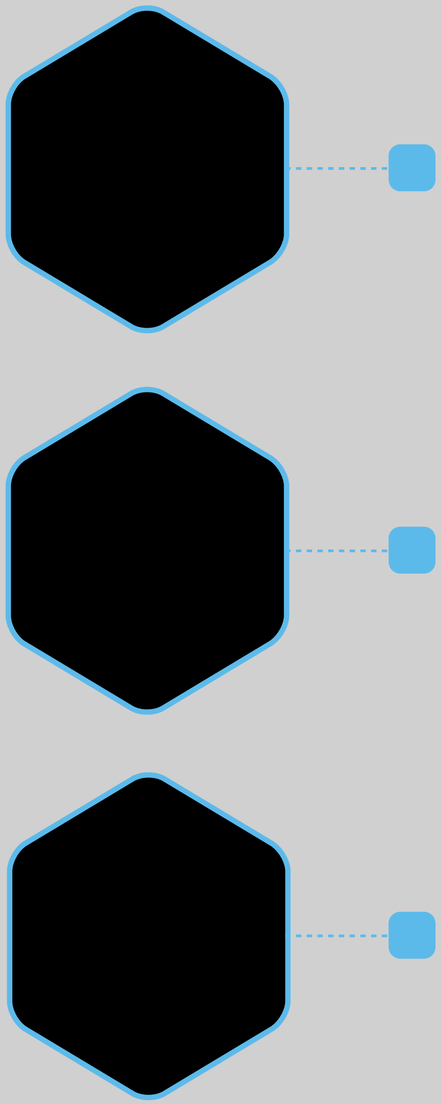


BEST PRACTICES

TOOLING

MUCH BETTER RUNTIMES

WE'VE CHANGED



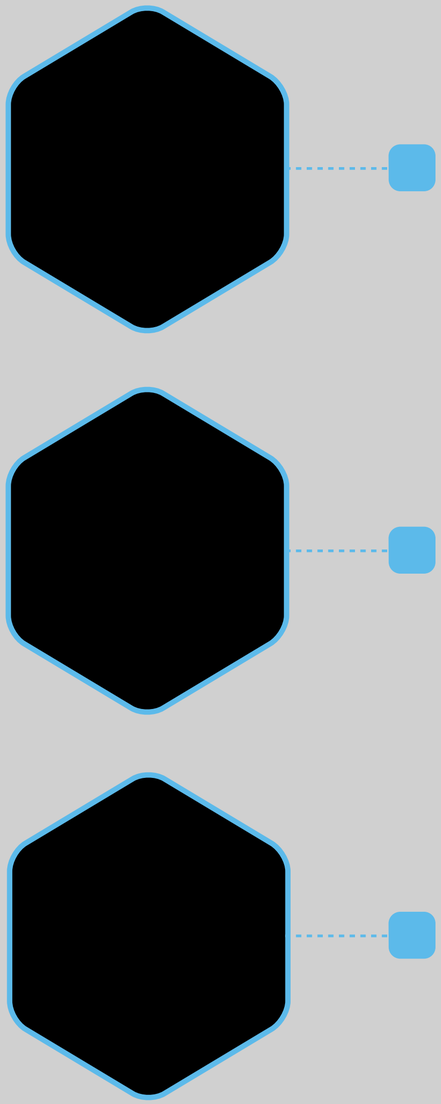
JS STILL HAS BAD PARTS

ASYNC, EVENT-DRIVEN

EVERY BROWSER SUPPORT

WHAT HASN'T CHANGED

JS OUTSIDE THE BROWSER?



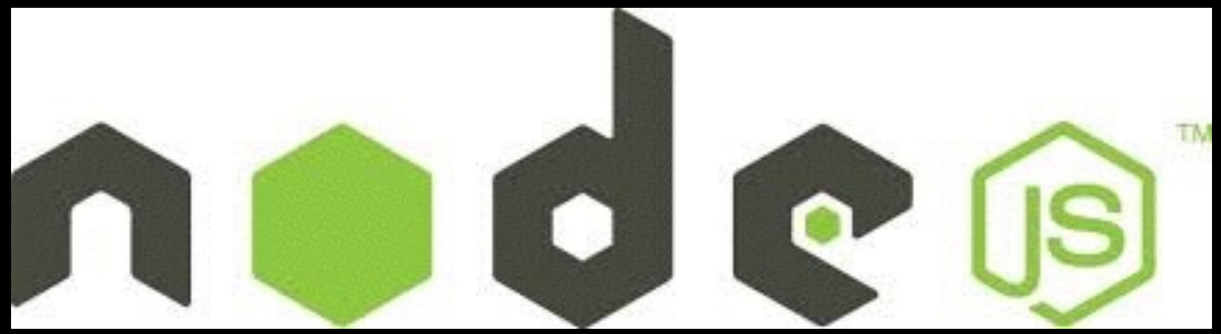
SUPER FAST JS RUNTIME

INCLUDED IN CHROME

RUNS ON CLI/SERVER

V8

NODE.JS



@prpatel

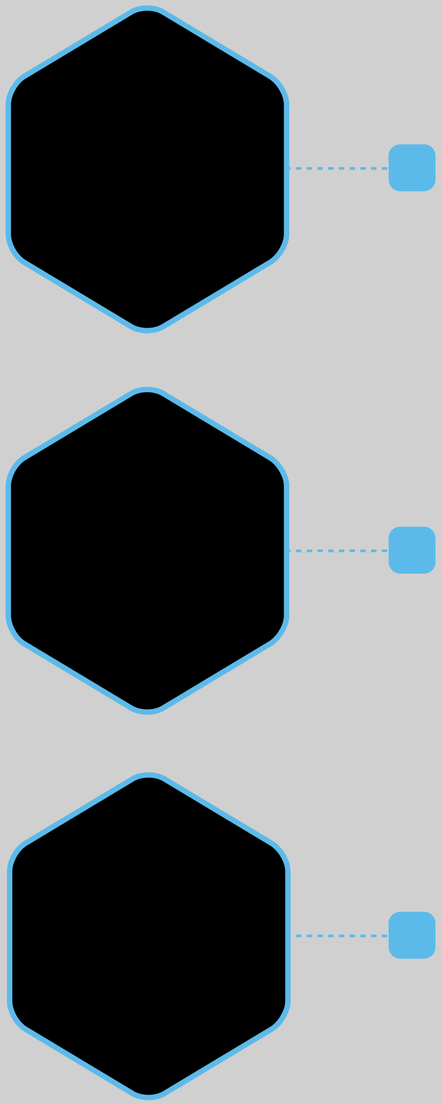
BUILT INTO JVM FOR YEARS

RHINO (CURRENT)

NASHORN (NEXT GEN)

WHAT ABOUT THE JVM?

WHY JAVASCRIPT ON THE SERVER?



SINGLE LANG FOR BROWSER & SERVER

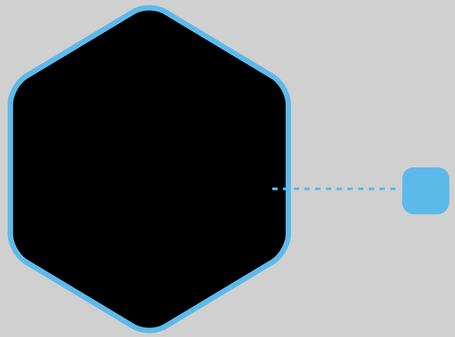
FAST

SCALABLE

WHY JS?

ISOMORPHIC APPS

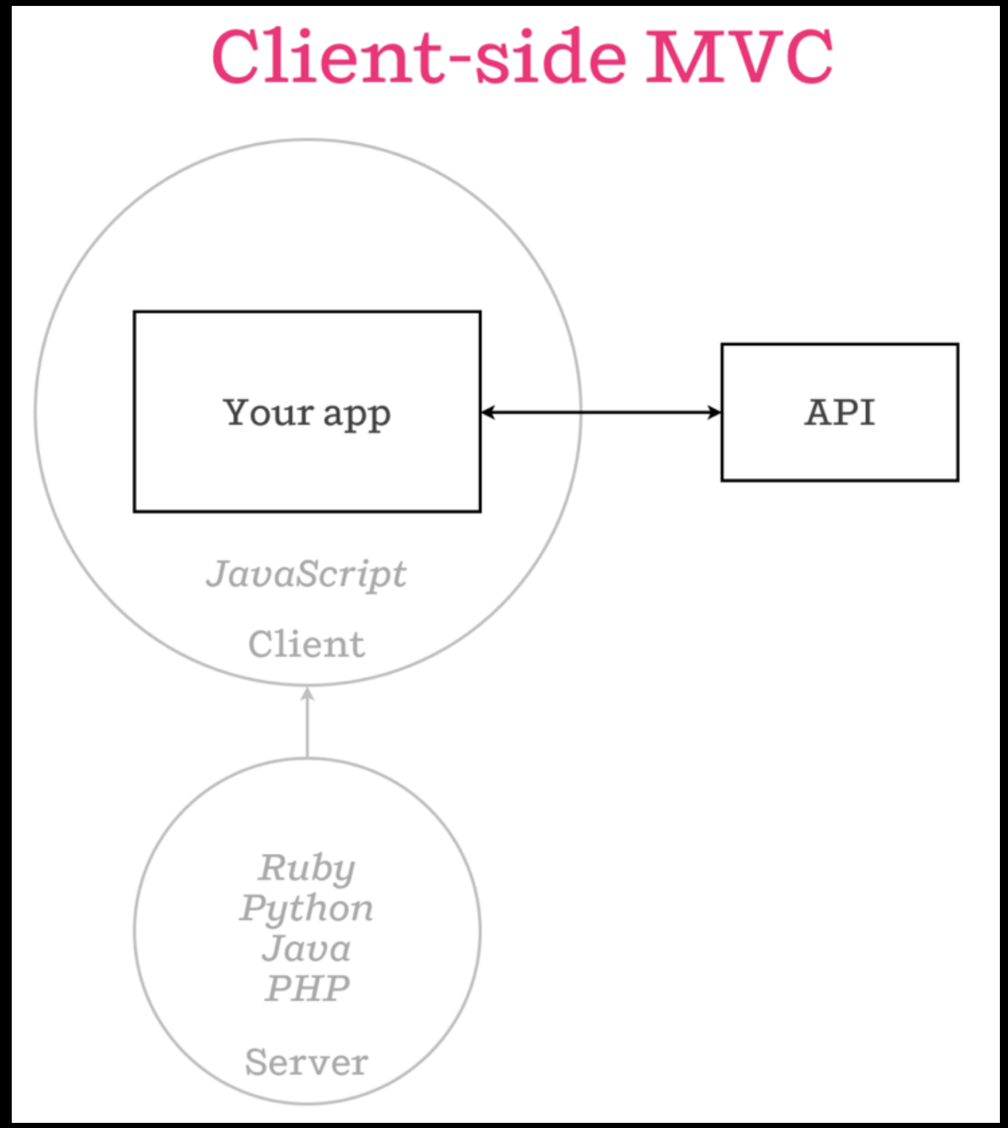
@prpatel



CODE CAN RUN EITHER ON BROWSER OR SERVER!

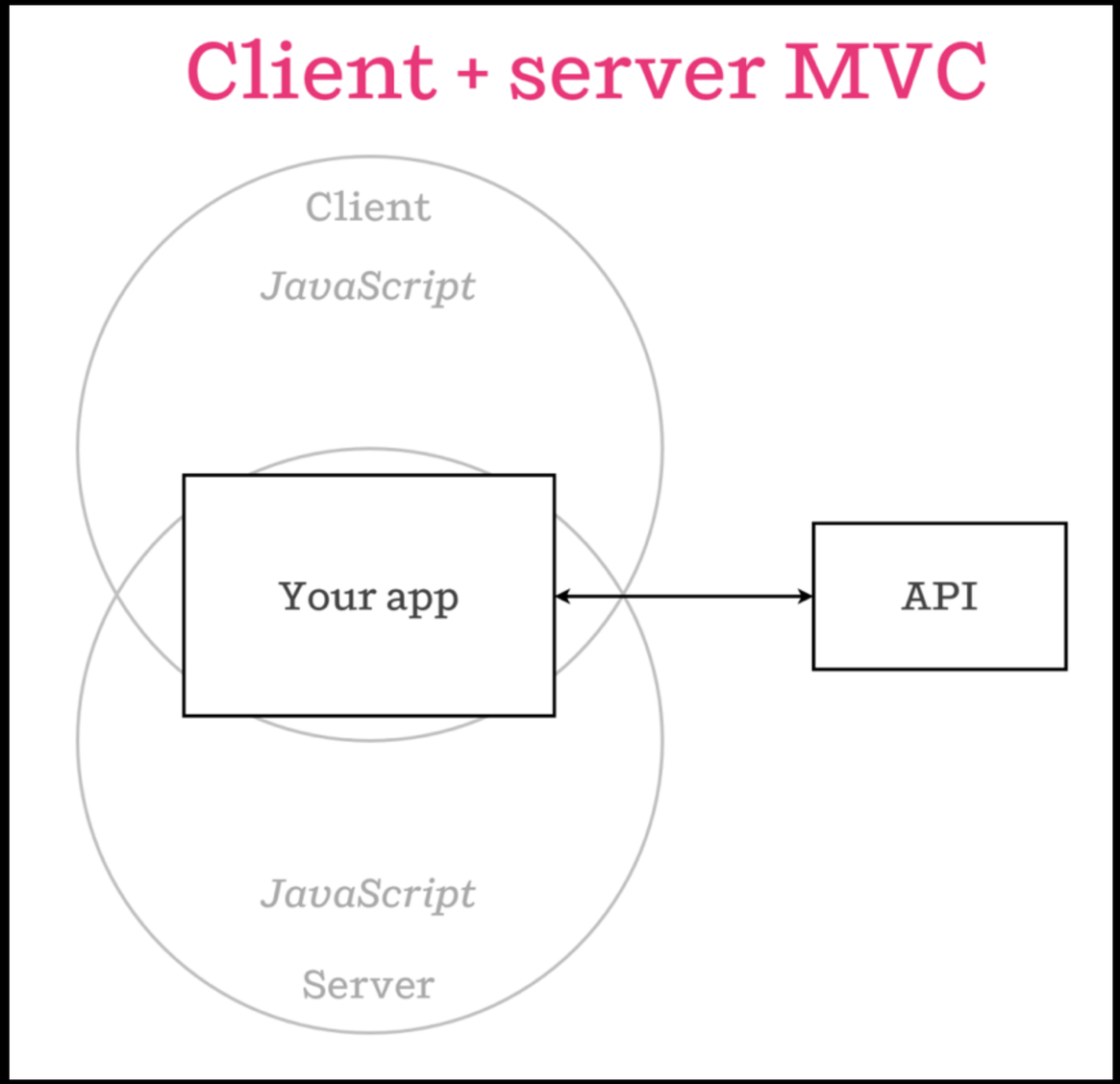
ISOMORPHIC APPS

“TRADITIONAL” WEBAPP

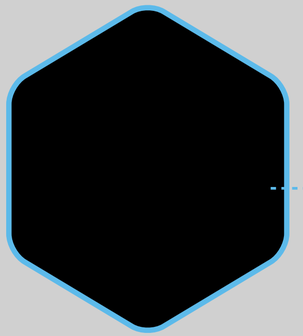


@prpatel

ISOMORPHIC



@prpatel



FLEXIBILITY

PERFORMANCE

WHY ISOMORPHIC?

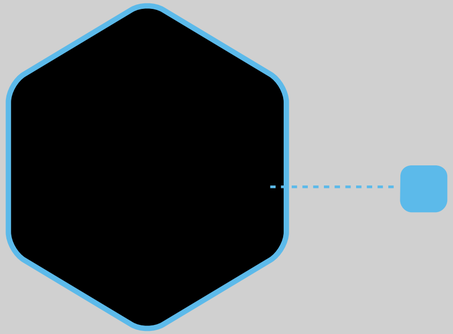
ECOSYSTEM

NODE.JS

NPM

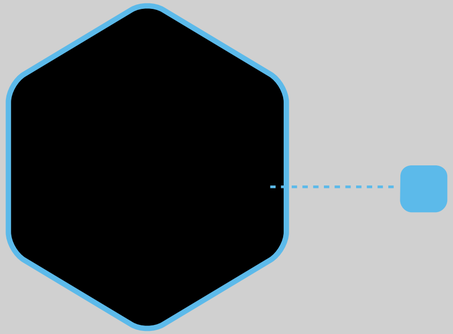
LOTS OF MODULES

ECOSYSTEM



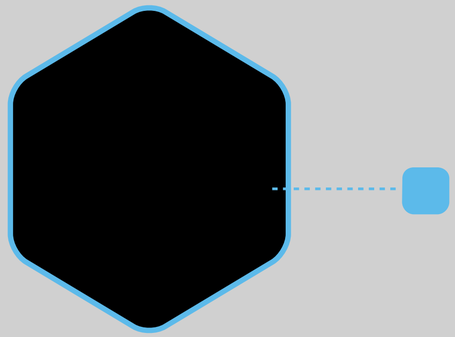
JAVASCRIPT RUNTIME BASED ON V8

NODE.JS



NODE PACKAGE MANAGER

NPM

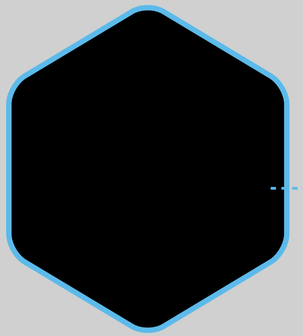


EXPRESS, GRUNT, GULP, WEBPACK, JASMINE, ETC

MODULES

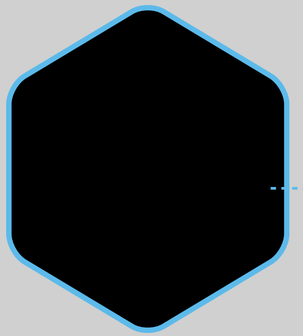
LAB0: NODE.JS

@prpatel



- INSTALL FROM NODEJS.ORG

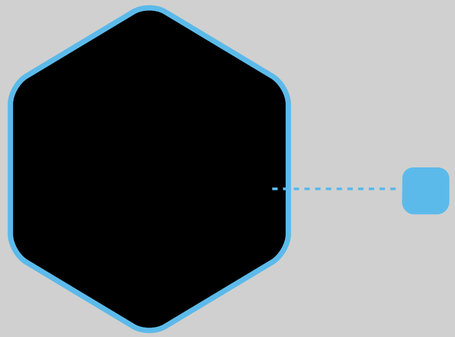
- INSTALL USING BREW (MAC)



CREATE DIRECTORY

CREATE TEST.JS AND RUN

NODE.JS



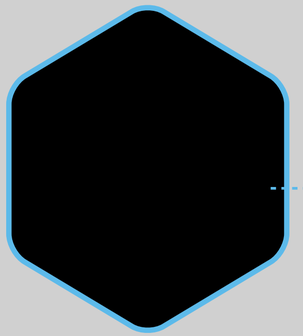
RUN FROM COMMAND-LINE

NODE.JS

NODE.JS

```
$ mkdir lab0; cd lab0
// create and edit test.js
$ node test.js
Hello BOS!


————- test.js:
console.log('Hello BOS!')
```

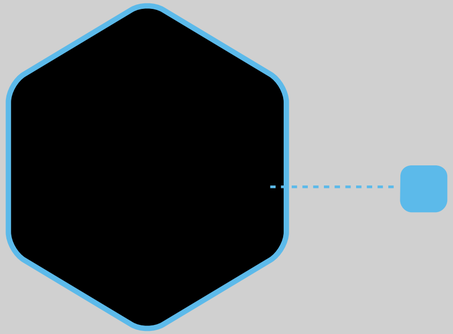
NPM



- BASIC PKG MANAGEMENT

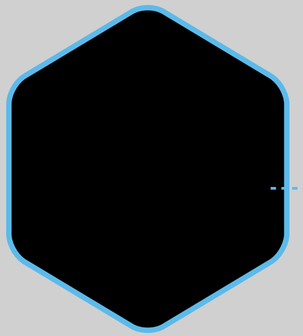
- COMES WITH NODE.JS

NPM



127,000+ PACKAGES IN THE NPM REPO

NPM



CREATE PACKAGE.JSON

NPM INSTALL

NODE.JS

LAB 1 & 2: NPM & PACKAGE.JSON

NPM

```
$ npm ls
$ npm install grunt -g
$ npm install underscore -save
$ npm install express -save
```

PACKAGE.JSON

```
// do this and run “npm install”
{
  "name": "hello-world",
  "description": "hello world test app",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "express": "3.x"
  }
}
```

WHY DO WE CARE ABOUT NODE.JS AGAIN?

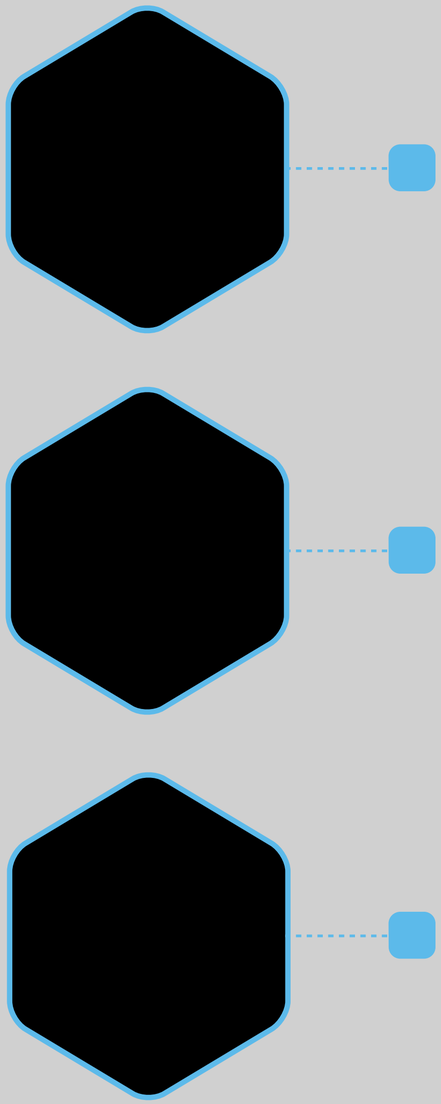
@prpatel

BEVY OF TOOLS FOR WEBDEV

ISOMORPHIC APPS

SCALABLE SOLUTIONS

WHY NODE?

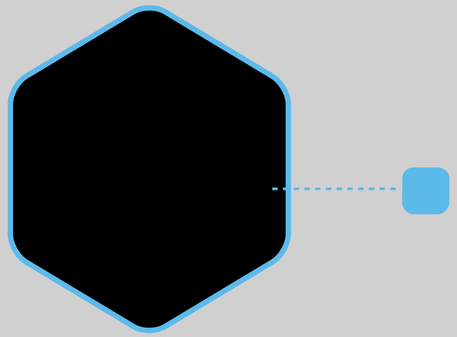


SOURCE CODE QUALITY

BUNDLING / DEVELOPMENT

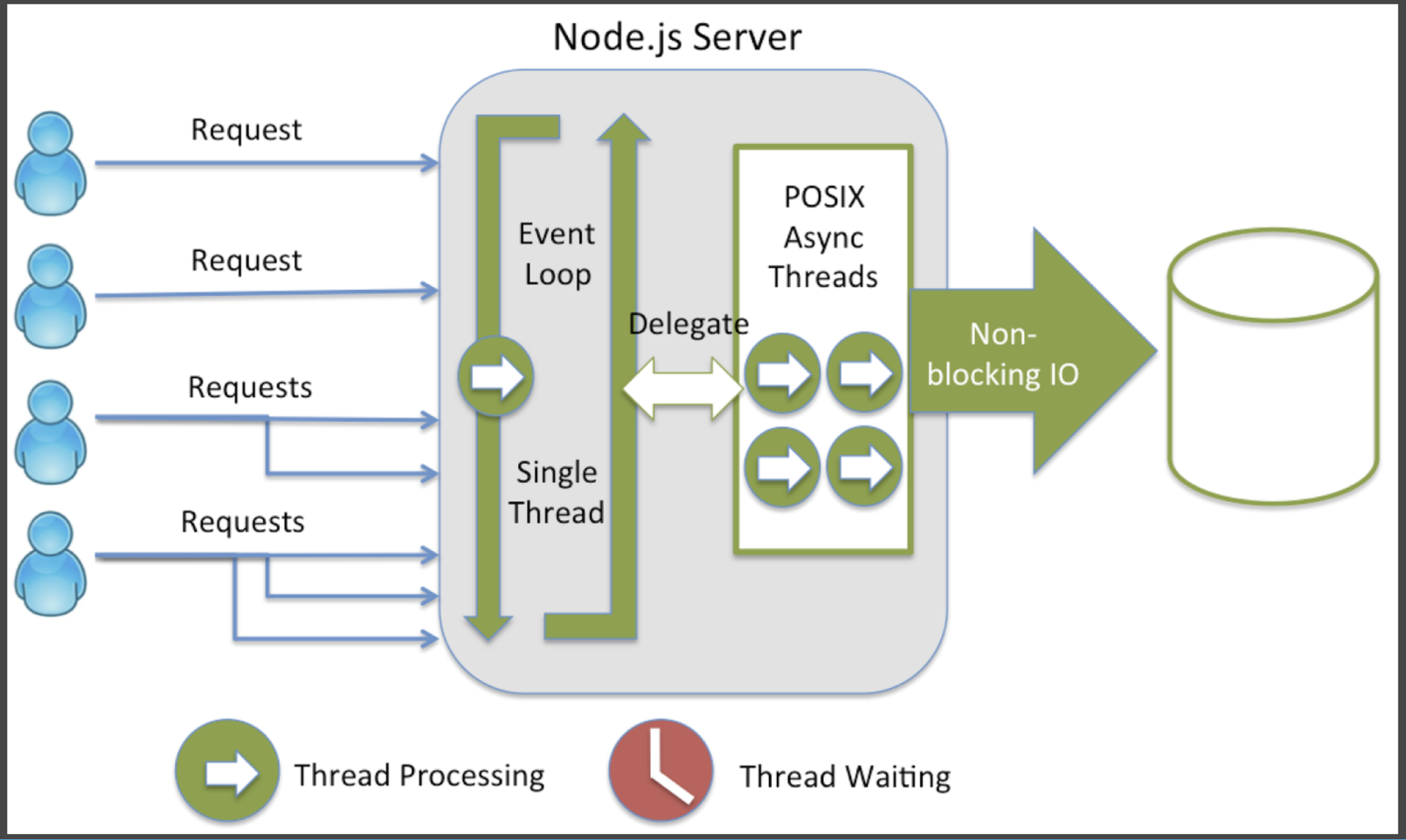
SERVER-SIDE JS

WHY NODE?

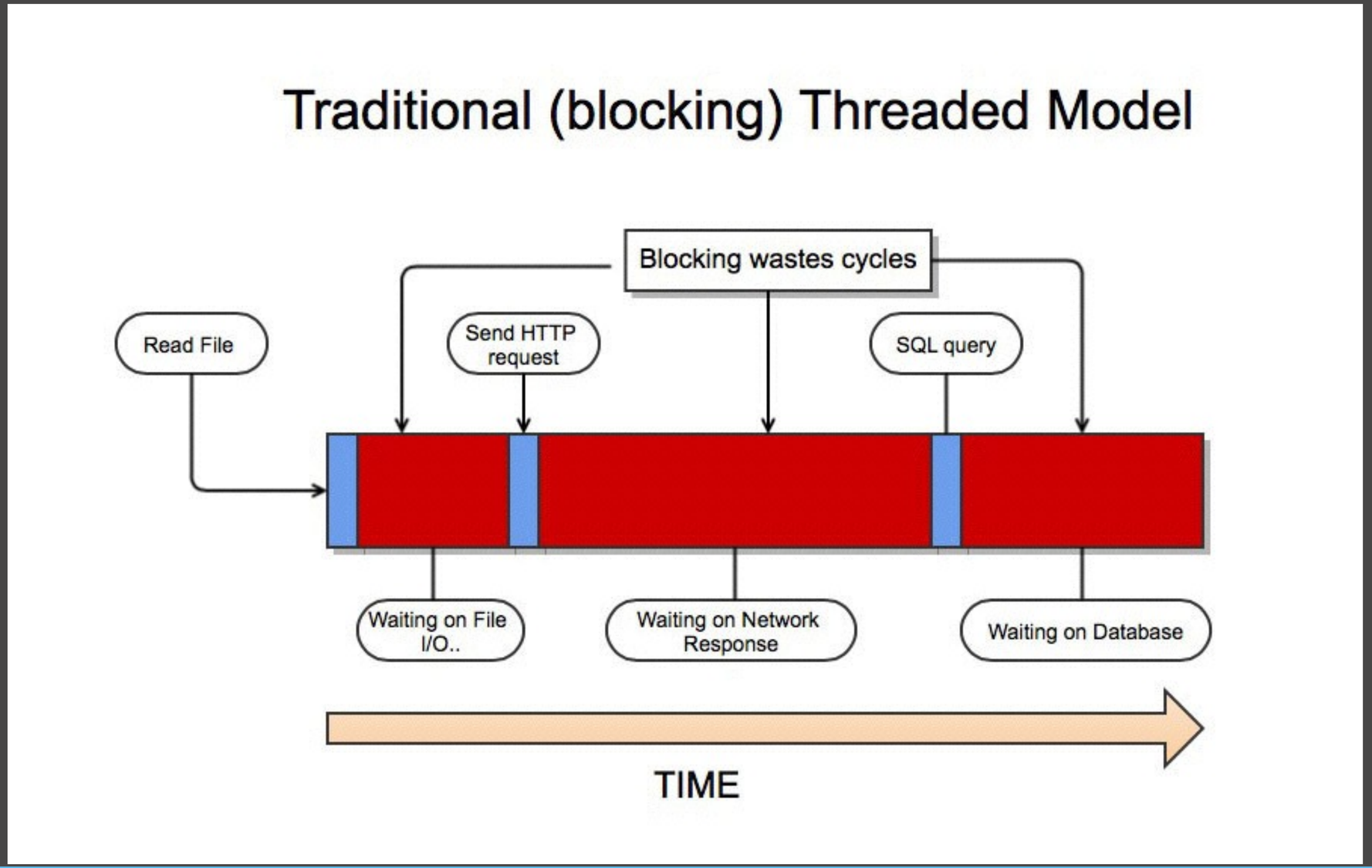


EXPRESS, GRUNT, GULP,
WEBPACK, JASMINE,
BROWSERSYNC, ETC

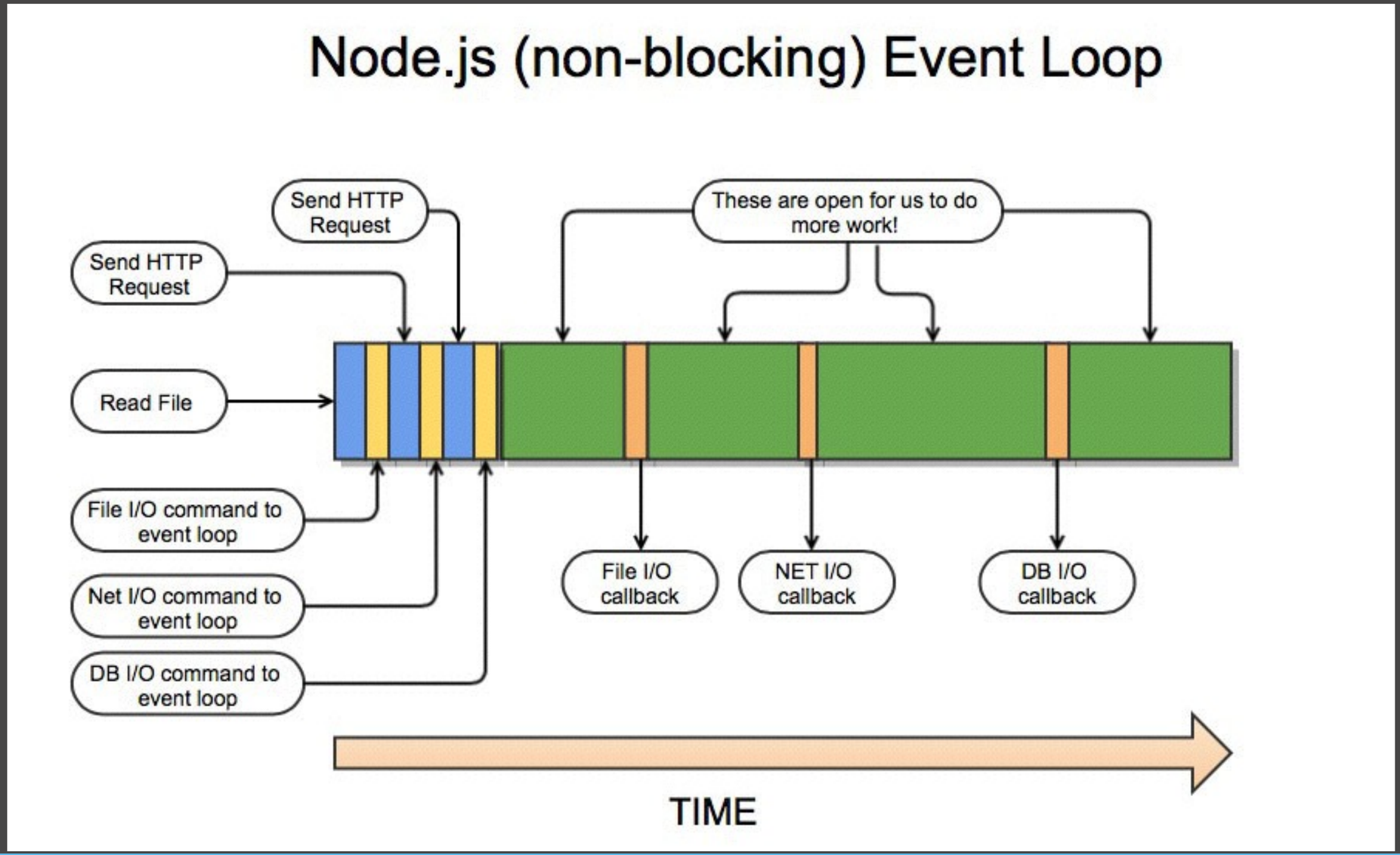
MODULES


Node.js Server
Request
Request
Requests
Requests
Event Loop
Single Thread
Delegate
POSIX Async Threads
Non-blocking IO
Thread Processing
Thread Waiting

SCALABLE



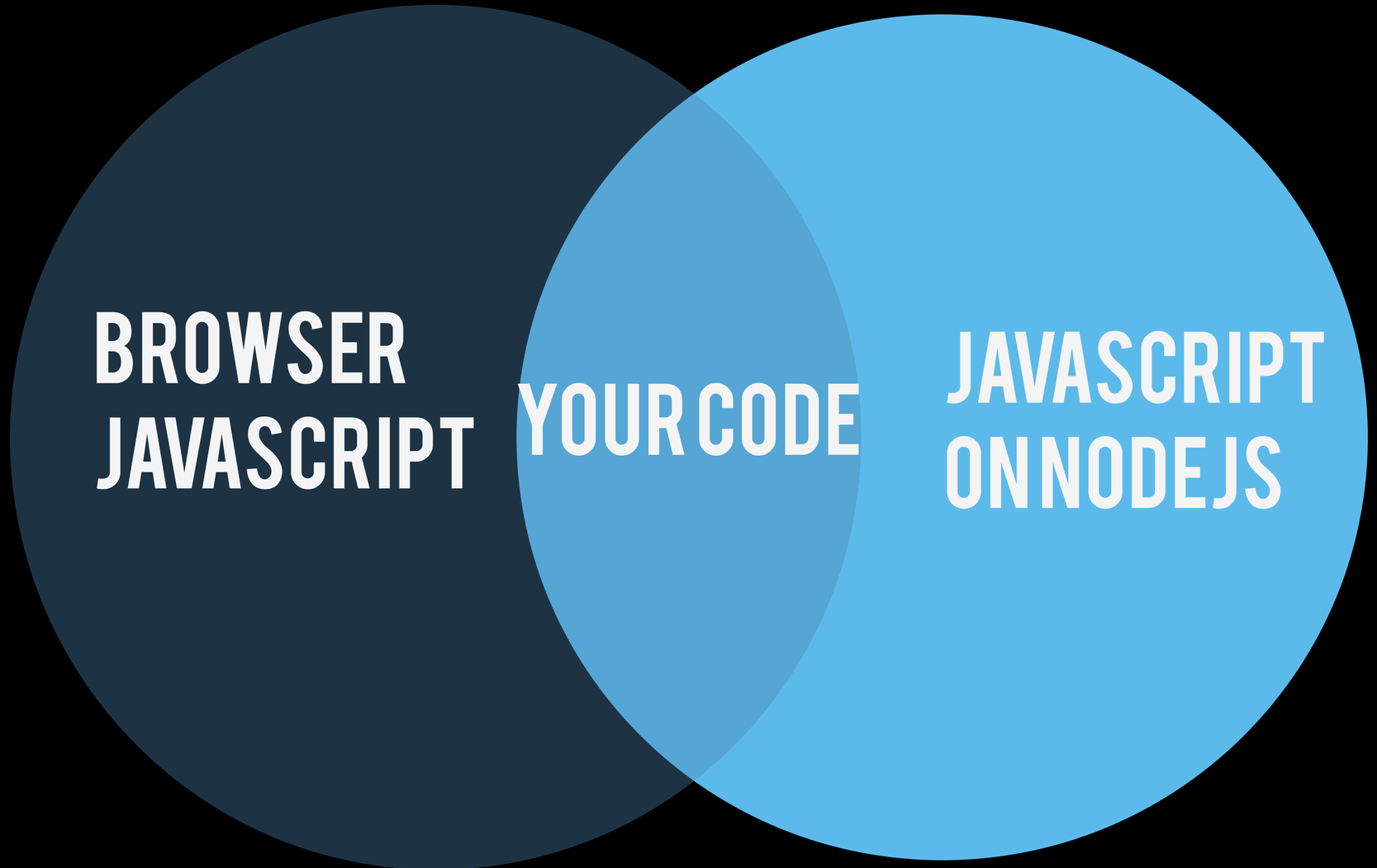
NOT SCALABLE


Node.js (non-blocking) Event Loop
Send HTTP Request
Send HTTP Request
These are open for us to do more work!
Read File
File I/O command to event loop
Net I/O command to event loop
DB I/O command to event loop
File I/O callback
NET I/O callback
DB I/O callback
TIME

SCALABLE

CODE RUNS ON THE BROWSER
_OR_
THE SERVER

ISOMORPHIC



@prpatel

LAB3: Standard Libs

@prpatel

Lab 3: Standard Libs

* open lab 3
* read a CSV file
* return a JSON result of the contents
* try to grok the code
* start in start/ folder
* solution in finished/ folder
* node app.js
* view result in browser at http://localhost:3000/
* final result:
  {“someKey":"someValue","anotherKey":"anotherValue"...

Lab4: CommonJS Modules

Lab 4: CommonJS

* Create 2 CommonJS modules
* Install underscore via npm (no package.json required!)
* in proj dir: npm install underscore
* Exposing something from a module
* module.exports = Person;
* exports.SOMETHING

Lab 5: Async Programming

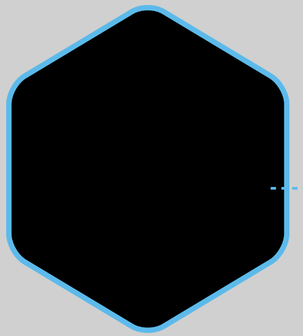
@prpatel

Lab 5: Async

* Examine and run the async-* examples
* Run csvParse.js
* Note the out of order log messages
* Observe the standard async style in csvParse.js:
* fs.readFile(arg1, arg2, **function(err, data)** {

Lab6: EXPRESS

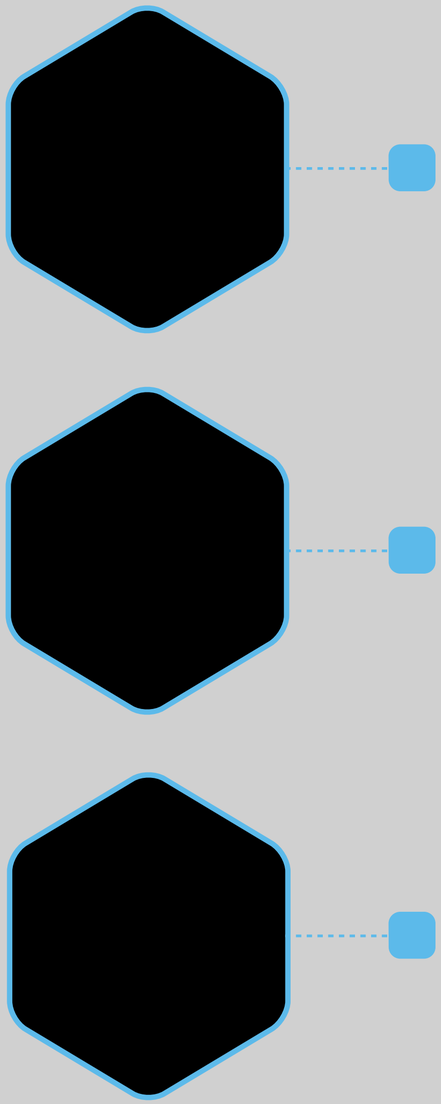
@prpatel



BASIC SERVER-SIDE WEB FRAMEWORK

LOTS OF OTHER OPTIONS: SAILS, KOA, GEDDY, ETC

EXPRESS



EVENT-LOOP MODEL

ASYNC, EVENT-DRIVEN

SUPER SCALABLE

EXPRESS

EXPRESS

```
$ cat hello-world/package.json
{
  "name": "hello-world",
  "description": "hello world test app",
  "version": "0.0.1",
  "private": true,
  "dependencies": {
    "express": "3.x"
  }
}
$ npm install
$ npm app
Express server listening on port 3000
```

Lab 6: Express

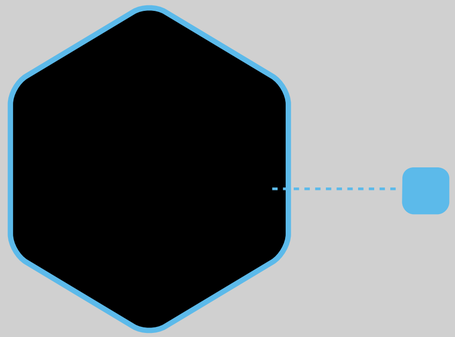
* Render HTML page using EJS template
* Have a simple form, POST back and save username
* Store username somehow
* If user exists in memory, show the username
* node app.js —> starts web server on port 3000
* note how the ‘start’ folder will hang until you write the code

TOOLS

@prpatel

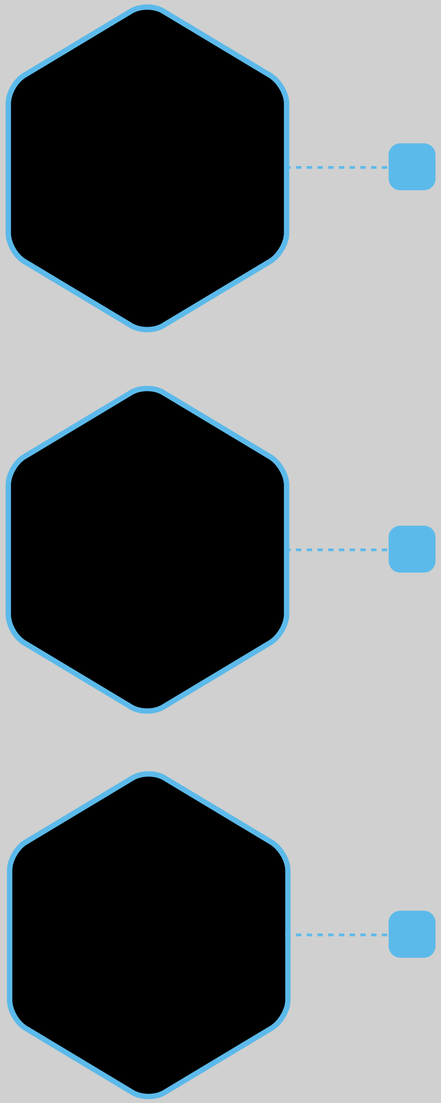
Lab 7: GRUNT

@prpatel



TASK BASED COMMAND-LINE TOOL

GRUNT



JAVASCRIPT LINTING

UNIT TEST RUNNING

MINIFICATION

USES

PACKAGE.JSON

```
"devDependencies": {
  "grunt": "~0.4.2",
  "grunt-contrib-jshint": "~0.6.3",
  "grunt-contrib-nodeunit": "~0.2.0",
  "grunt-contrib-uglify": "~0.2.2"
}
```

GRUNTFILE

```
module.exports = function (grunt) {
    grunt.initConfig({
        jshint: {
            all: ['Gruntfile.js', 'lib/**/*.js', 'test/**/*.js',
'public/**/*.js',
            'routes/*.js', 'routes/**/*.js', 'app.js']
        }
    });
    grunt.loadNpmTasks('grunt-contrib-jshint');
    grunt.registerTask('default', ['jshint']);
};
```

@prpatel

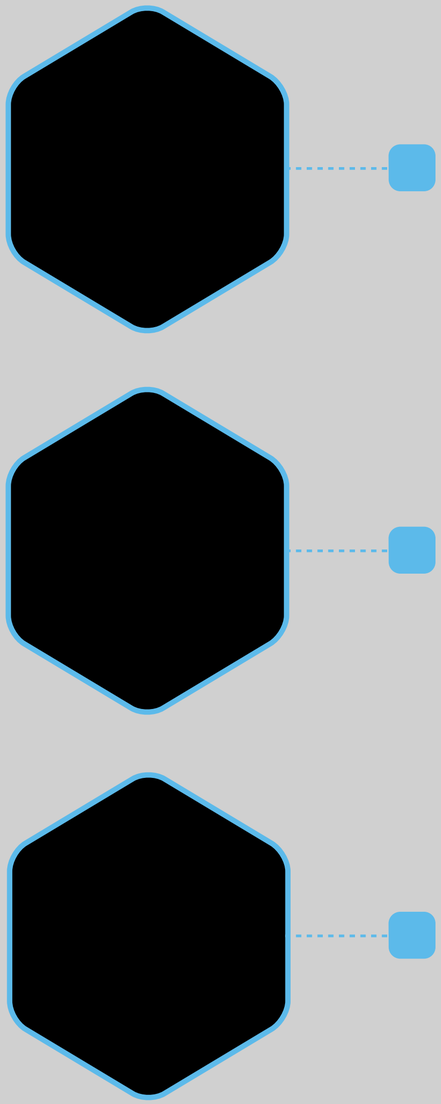
RUNNING

```
$ grunt
Running "jshint:all" (jshint) task
Linting app.js ...ERROR
[L12:C20] W033: Missing semicolon.
var app = express()

Warning: Task "jshint:all" failed. Use --force
to continue.

Aborted due to warnings.
```

Lab 7: Grunt

* npm install grunt-cli -g
* npm install grunt
* Examine the Gruntfile.js
* run: grunt
* fix the source file that's broken
* run: grunt

Lab 8: Testing & Test Automation



DIRECT TEST OF JAVASCRIPT

HEADLESS TESTING

AUTOMATION

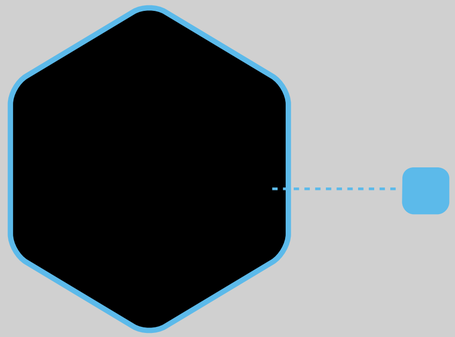
TESTING YOUR PROJECT

Lab 8: Testing

* Jasmine & JQuery
* grunt jasmine
* Examine the project: test/ scripts/ folders
* grunt jasmine
* fix the broken test
* grunt jasmine
* What is that PhantomJS thing?

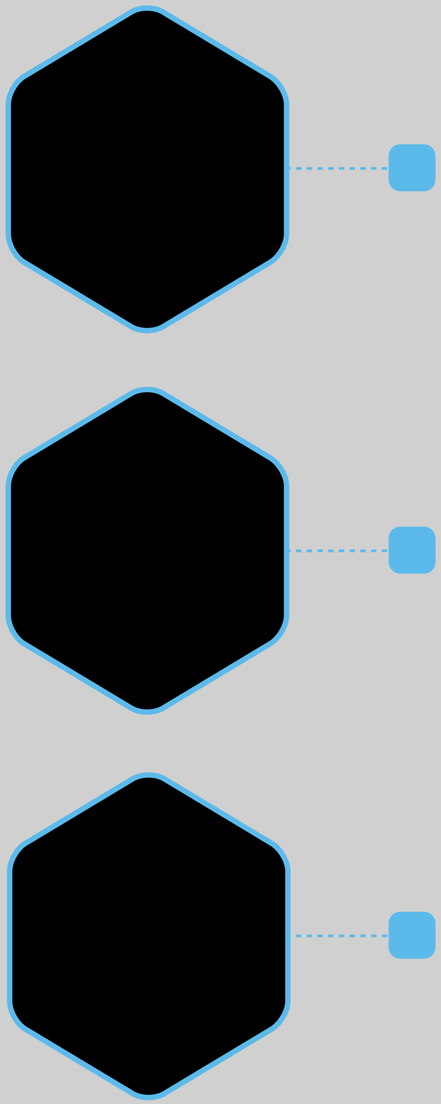
Socket IO

@prpatel



WEBSOCKET

SOCKET.IO



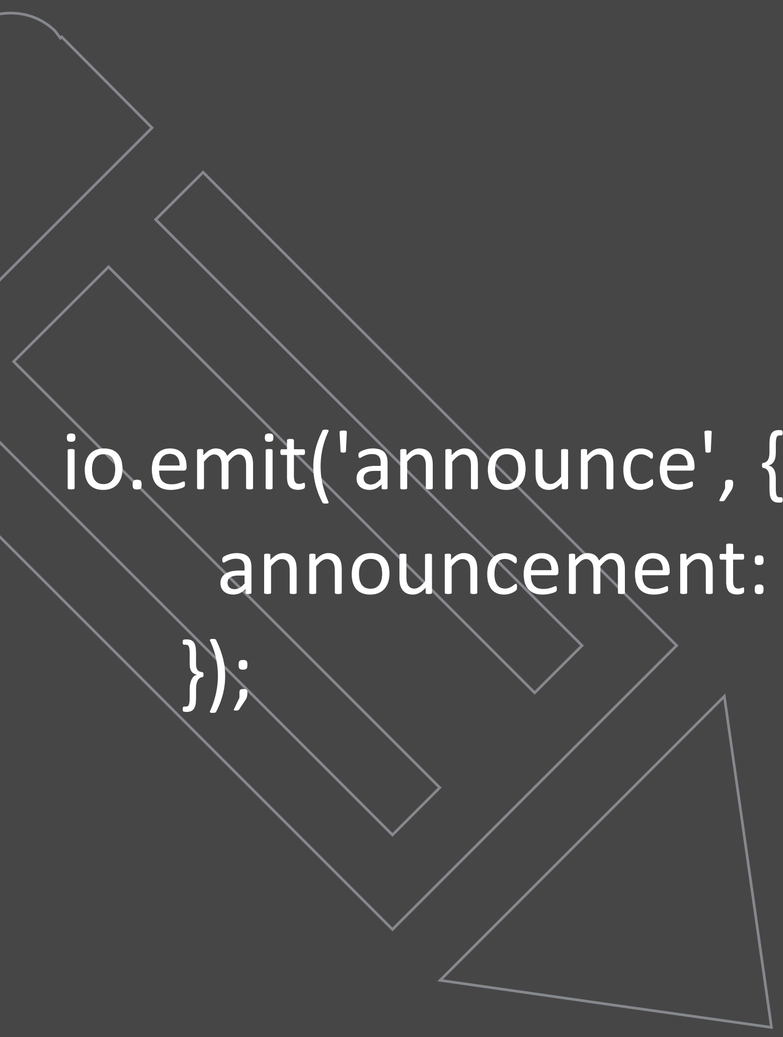
PERSISTENT CONNECTION

NODE HANDLES IT WELL

SIMPLE ON FRONT & BACK

SOCKET.IO

SERVER



```
io.emit('announce', {
        announcement: announcement
    });
```

BROWSER

```
socket.on('announce', function (data) {
    self.data.announcement = data;
    self.displayNewData();
  });
```

Lab 9: Websocket

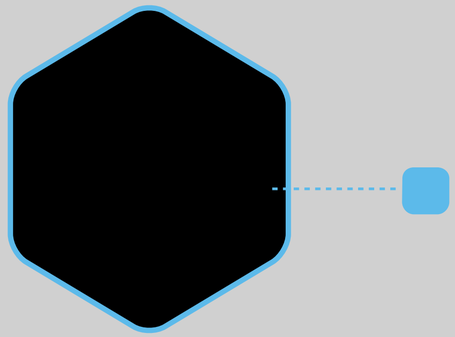
@prpatel

Lab 9: Socket.io

* Create a chat application
* Start in start/ , solution in finished/
* Store session info
* Include socket.io in HTML page
* send/broadcast chat messages

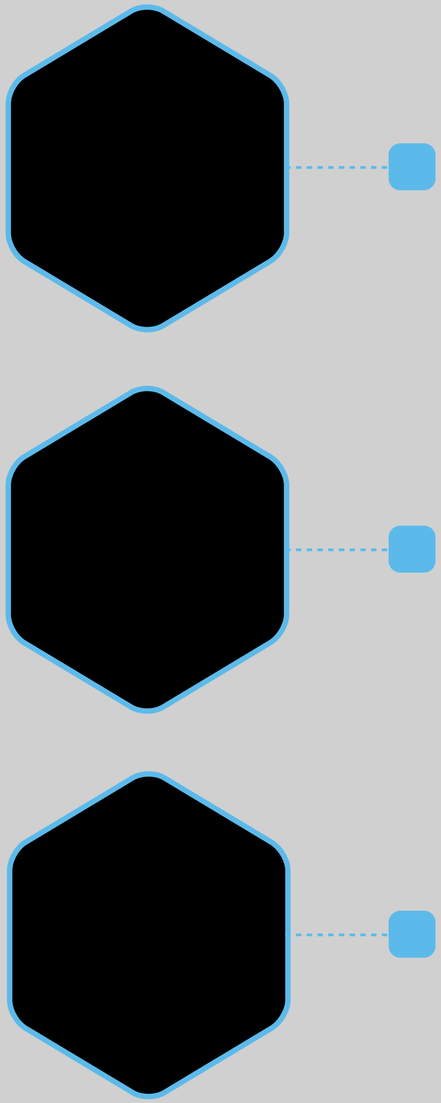
Webpack

@prpatel



MODULE BUILDER

WEBPACK

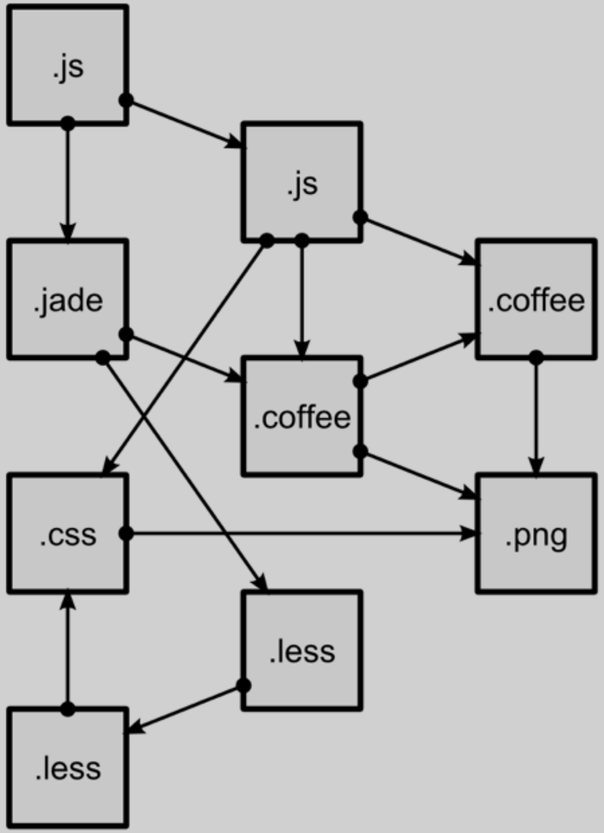


SUITED FOR LARGE PROJECTS

CODE SPLITTING

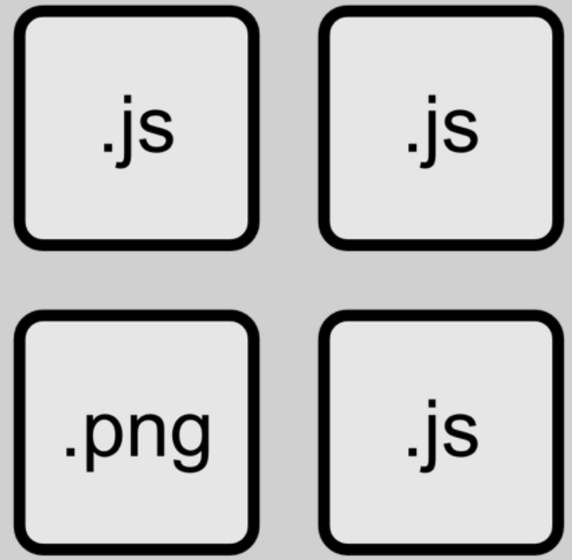
STATIC ASSETS

WEBPACK



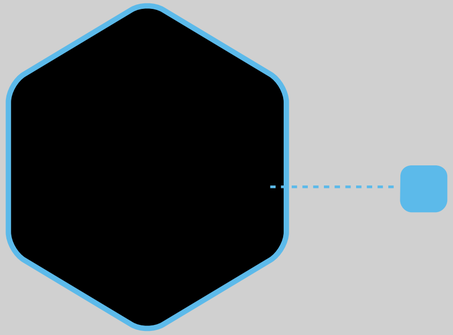
modules
with dependencies

webpack
MODULE BUNDLER



static
assets

WEBPACK



BROWSER SYNC & LIVE

HTTP://WEBPACK.GITHUB.IO/DOCS/COMPARISON.HTML

Lab 10: ReactJS

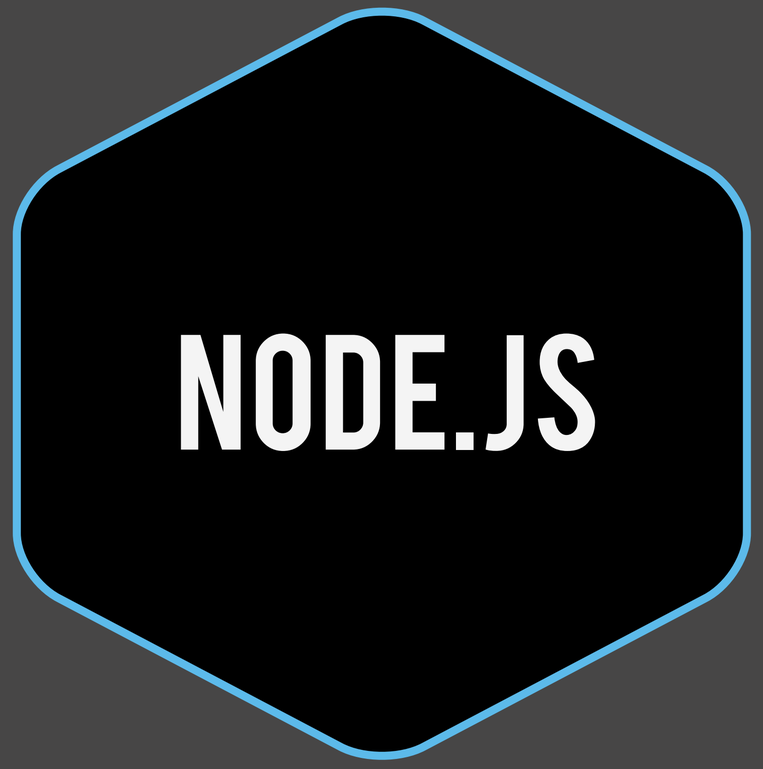
@prpatel

Lab 10: REACTJS

* npm run dev
* Point browser to localhost:8080
* Change Home.js
* Create a new component, include it in Home.js

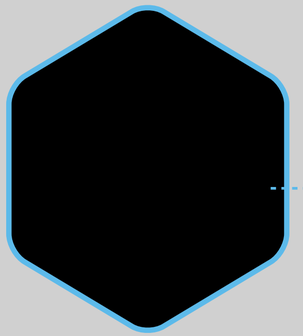
Lab 11: Browser debugging

* Load lab11/index.html in Chrome
* Click sources tab
* Click in gutter to set break point
* Reload page



DIFFERENT UNIVERSE THAN WHAT YOU ARE USED TO

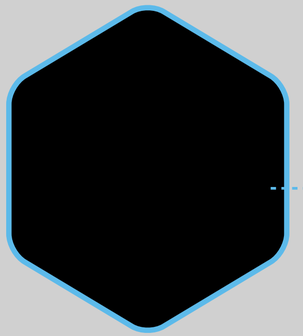
@prpatel



ASSEMBLE YOUR OWN FRAMEWORK

SMALLER MODULES

SERVER-SIDE JS



- NO “ONE-WAY” TO DO THINGS

- OVERLAPPING TOOLS

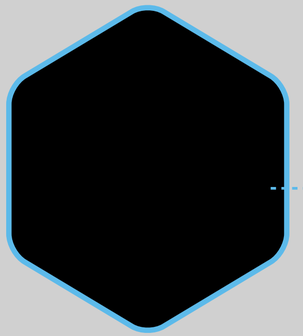
SERVER-SIDE JS

NOT FOR EVERY KIND OF APP

ASYNC, EVENT-DRIVEN

STILL MATURING

CLOSING THOUGHTS



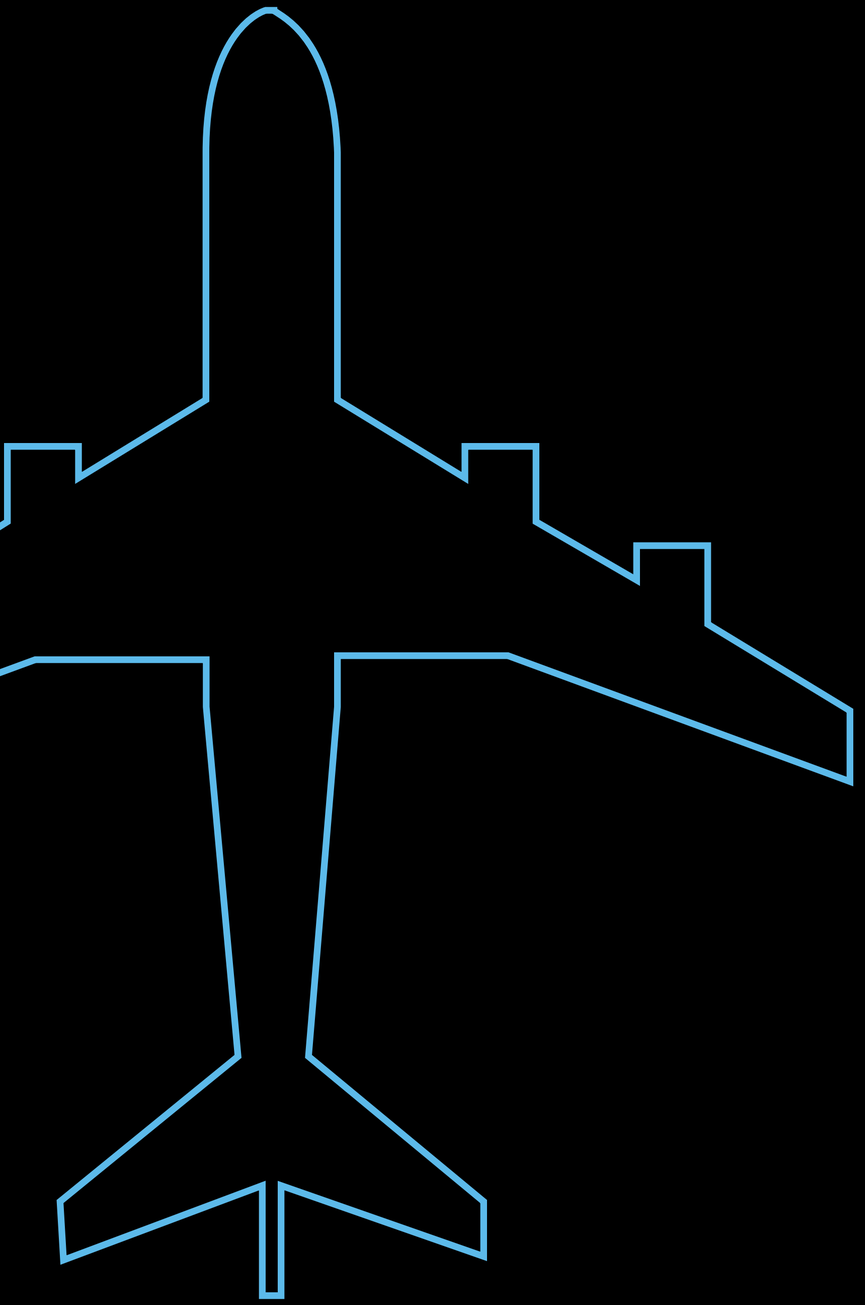
CAN BE TIED INTO EXISTING BUILD PROCESS

CAN USE EXISTING DB & OTHER INFRA

SERVER-SIDE JS

References:
https://strongloop.com/strongblog/node-js-performance-event-loop-monitoring/
http://charwangles.com/2014/08/15/what-is-node-js/
https://github.com/justinklemm/nodejs-async-tutorial
https://github.com/kwhinnery/node-workshop

@prpatel

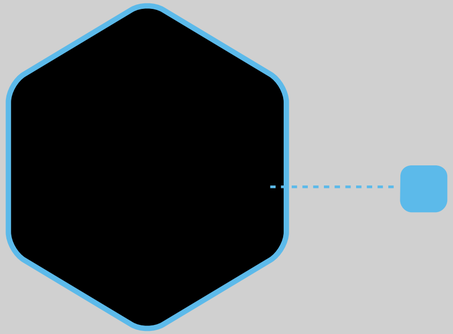


THANK YOU

@prpatel

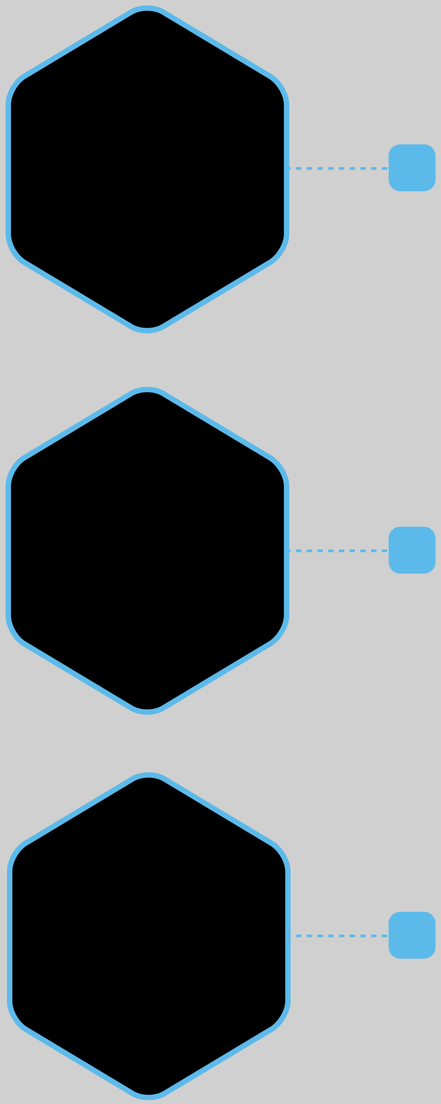
GULP

@prpatel



STREAMING BUILD TOOL

GULP



- PLUGIN BASED
- STREAMING
- SUCCESSOR TO GRUNT (?)

USES

GULPFILE

```
gulp.task('build', ['clean'], function(cb) {
  runSequence(['vendor', 'assets', 'styles', 'bundle'], cb);
});
```

GULPFILE

```
gulp.task('assets', function() {
  src.assets = [
    'src/assets/**',
    'src/content*/**/*.*',
    'src/templates*/**/*.*'
  ];
  return gulp.src(src.assets)
    .pipe($.changed('build'))
    .pipe(gulp.dest('build'))
    .pipe($.size({title: 'assets'}));
});
```

@prpatel

GULPFILE

```
gulp.task('bundle', function(cb) {
  var started = false;
  var config = require('./webpack.config.js');
  var bundler = webpack(config);
...
}
```

References:

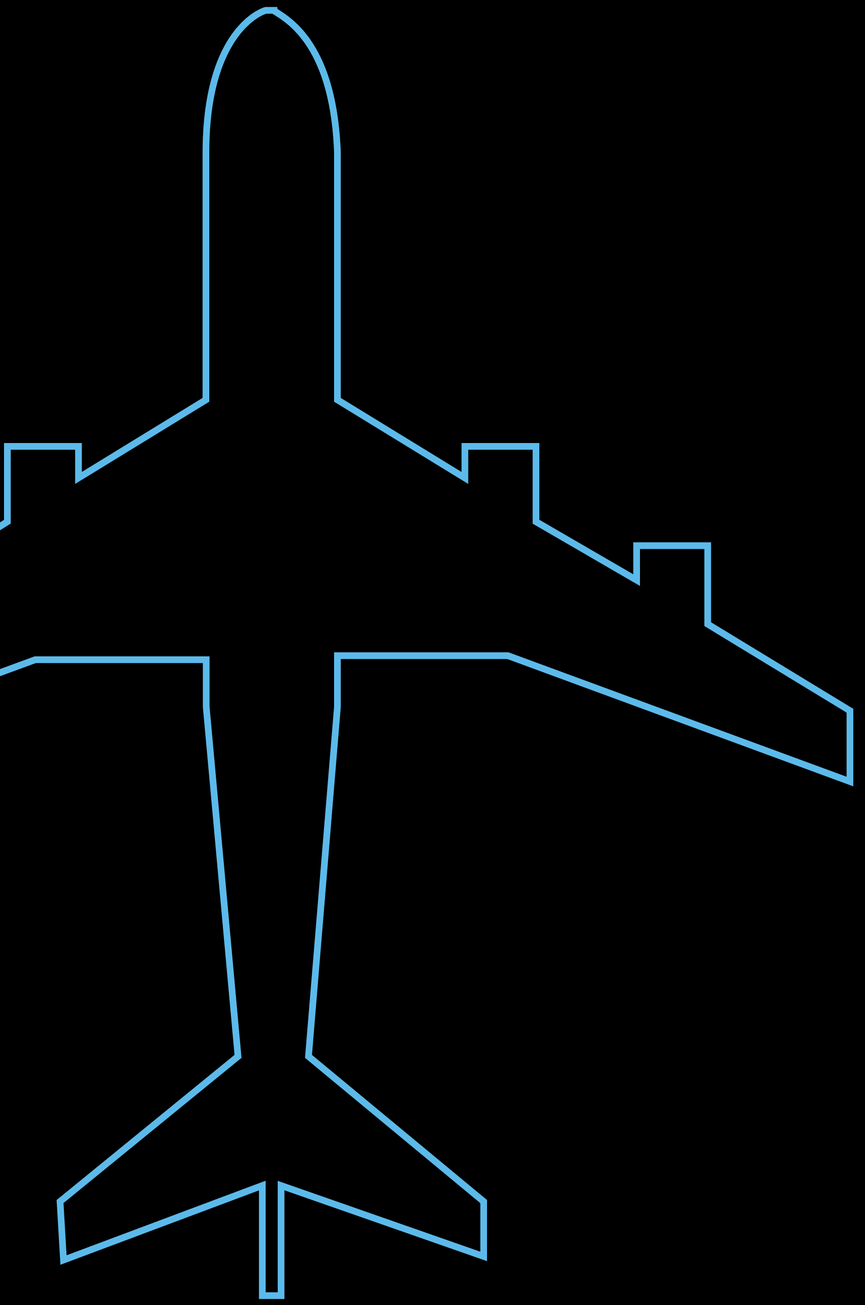
http://gulpjs.com/

https://github.com/kriasoft/react-starter-kit

PAGESPEED

```
$ gulp pagespeed
[15:48:49] Using gulpfile ~/dev/react/react-starter-kit/gulpfile.js
[15:48:49] Starting ‘pagespeed'...

CSS size                    | 148.13 kB
HTML size                    | 20.84 kB
Image size                  | 54.24 kB
JavaScript size              | 1.72 MB
```

@prpatel



THANK YOU

@prpatel