

# Advanced Node.js

Claudia Hauff  
[cse1500-ewi@tudelft.nl](mailto:cse1500-ewi@tudelft.nl)

# Web technology overview

1. HTTP: the language of web communication
2. HTML & web app design
3. JavaScript: interactions in the browser
4. Node.js: JavaScript on the server
5. CSS: adding style
6. Node.js: advanced topics
7. Cookies & sessions
8. Web security

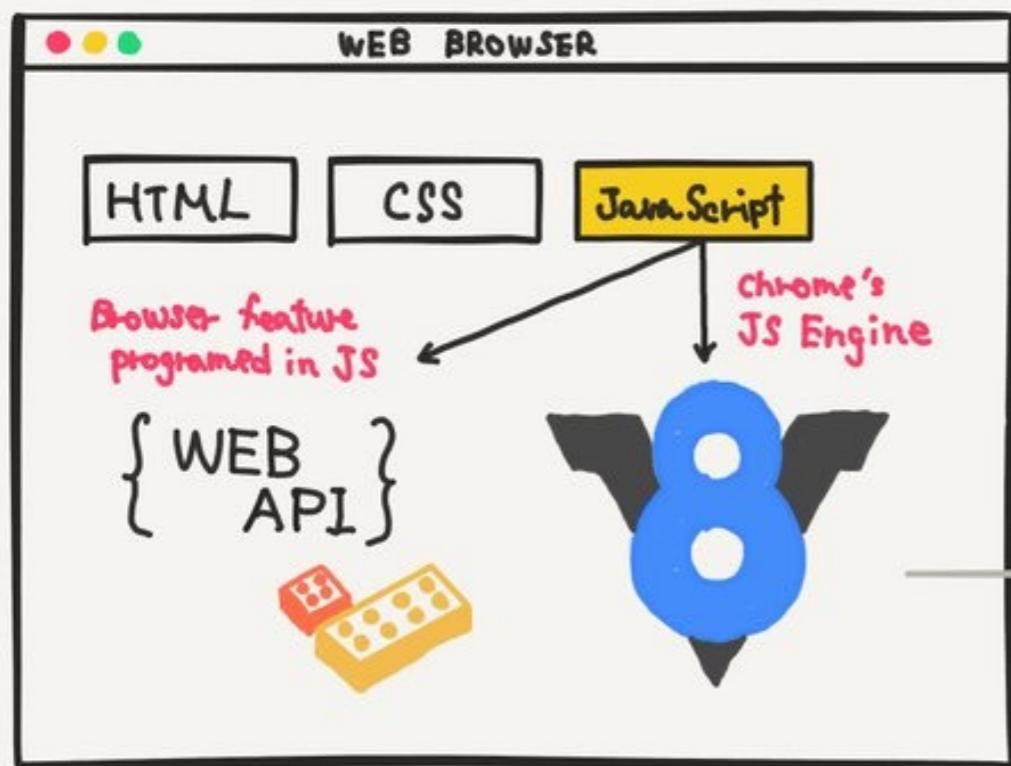
# Learning goals

- **Organize** Node.js code into modules
- **Understand and employ** the concept of middleware
- **Employ** routing
- **Employ** templating

# Organization and reusability of Node.js code

## What is Node.js ??

What's browser JS ??

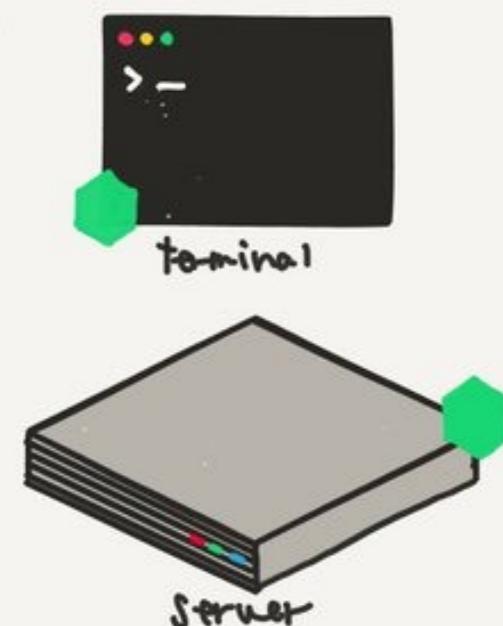
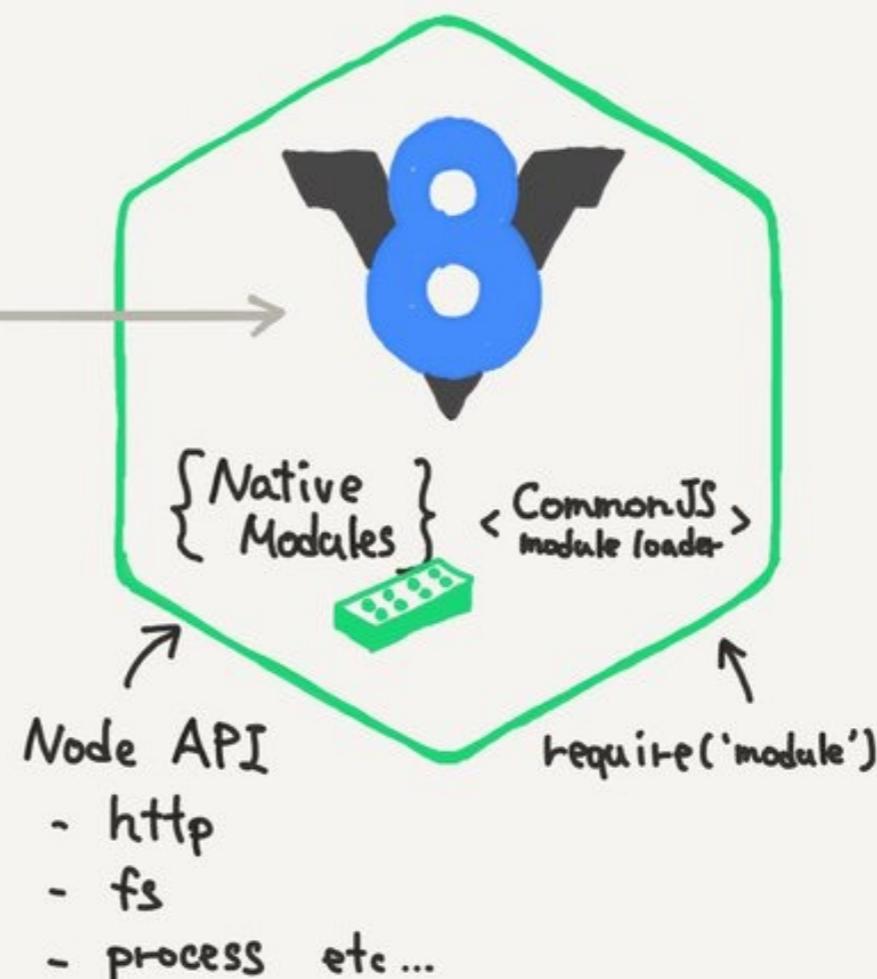


Web Platform  
- Window  
- Fetch  
- Web Audio etc...

Language features  
- ES6  
- promise  
- Typed Array etc ...

Web API & JavaScript Language are often described as the JavaScript (in intro books)

Node is a JavaScript ENVIRONMENT with Special API (like http) and default module loader.



Node runs on  
EVERYWHERE.

# So far...

- All server-side code maintained **within a single file**
  - Often makes sense for client-side JS (**minification**)
- Possible for small projects
- Larger projects suffer in this setting
  - **Debugging** is cumbersome
  - **Team-work** is cumbersome
  - **Programming** is cumbersome

```

1 /* ESLint global variables information */
2 /* global Setup, Status, Messages, englishDict*/
3
4 /* basic constructor of game state */
5 function GameState(visibleWordBoard, sb, socket){
6
7     this.playerType = null;
8     this.MAX_ALLOWED = Setup.MAX_ALLOWED_GUESSES;
9     this.wrongGuesses = 0;
10    this.visibleWordArray = null;
11    this.alphabet = new Alphabet();
12    this.alphabet.initialize();
13    this.visibleWordBoard = visibleWordBoard;
14    this.targetWord = null;
15    this.statusBar = sb;
16
17    this.initializeVisibleWordArray = function(){
18        this.visibleWordArray = new Array(this.targetWord.length);
19        this.visibleWordArray.fill(Setup.HIDDEN_CHAR);
20    };
21
22    this.getPlayerType = function () {
23        return this.playerType;
24    };
25
26    this.setPlayerType = function (p) {
27        console.assert(typeof p == "string", "%s: Expecting a string", p);
28        this.playerType = p;
29    };
30
31    this.setTargetWord = function (w) {
32        console.assert(typeof w == "string", "%s: Expecting a string", w);
33        this.targetWord = w;
34    };
35
36    this.getVisibleWordArray = function(){
37        return this.visibleWordArray;
38    };
39
40    this.incrWrongGuess = function(){
41        this.wrongGuesses++;
42
43        if(this.whoWon() == null){
44            //kill a balloon
45            let id = "b"+this.wrongGuesses;
46            document.getElementById(id).className += " balloonGone";
47            setTimeout(function () {
48                new Audio("../data/pop.wav").play();
49            }, 500);
50        }
51    };

```

# minification

```

1 function GameState(t,e,a){this.playerType=null,this.MAX_ALLOWED=Setup.MAX_ALLOWED_GUESSES,
2 this.wrongGuesses=0,this.visibleWordArray=null,this.alphabet=new Alphabet,this.
3 alphabet.initialize(),this.visibleWordBoard=t,this.targetWord=null,this.statusBar=e,
4 this.initializeVisibleWordArray=function(){this.visibleWordArray=new Array(this.
5 targetWord.length),this.visibleWordArray.fill(Setup.HIDDEN_CHAR)},this.getPlayerType=
6 function(){return this.playerType},this.setPlayerType=function(t){console.assert(
7 "string"==typeof t,"%s: Expecting a string, got a %s",arguments.callee.name,typeof t),
8 this.playerType=t},this.setTargetWord=function(t){console.assert("string"==typeof t,"
9 %s: Expecting a string, got a %s",arguments.callee.name,typeof t),this.targetWord=t},
10 this.getVisibleWordArray=function(){return this.visibleWordArray},this.incrWrongGuess=
11 function(){if(this.wrongGuesses++,null==this.whoWon()){let t="b"+this.wrongGuesses;
12 document.getElementById(t).className+=" balloonGone",setTimeout(function(){new Audio(
13 "../data/pop.wav").play()},500)}},this.whoWon=function(){return this.wrongGuesses>Setup.
14 MAX_ALLOWED_GUESSES?"A":this.visibleWordArray.indexOf("#")<0?"B":null},this.
15 revealLetters=function(t,e){console.assert("string"==typeof t,"%s: Expecting a
16 string, got a %s",arguments.callee.name,typeof t),console.assert(e instanceof Array,
17 "%s: Expecting an array",arguments.callee.name);for(let a=0;a<e.length;a++)this.
18 visibleWordArray[e[a]]=t},this.revealAll=function(){this.visibleWordBoard.setWord(this.
19 targetWord)},this.updateGame=function(t){console.assert("string"==typeof t,"%s:
20 Expecting a string, got a %s",arguments.callee.name,typeof t);var s=this.alphabet.
21 getLetterInWordIndices(t,this.targetWord);0==s.length?this.incrWrongGuess():this.
22 revealLetters(t,s),this.alphabet.makeLetterUnAvailable(t),this.visibleWordBoard.
23 setWord(this.visibleWordArray);var i=Messages.O_MAKE_A_GUESS;i.data=t,a.send(JSON.
24 stringify(i));let r=this.whoWon();if(null!=r){this.revealAll();var n=document.
25querySelectorAll(".alphabet");let t;if(Array.from(n).forEach(function(t){var e=t.clo-
26 neNode(!0);t.parentNode.replaceChild(e,t)}),t==this.playerType?Status.gameWon:Status.
27 gameLost,t+=Status.playAgain,e.setStatus(t),"B"==this.playerType){let t=Messages.
28 O_GAME WON_BY;t.data=r,a.send(JSON.stringify(t))}a.close()}}}function AlphabetBoard(t)
29 {this.initialize=function(){var e=document.querySelectorAll(".alphabet");Array.from(e).
30 .forEach(function(e){e.addEventListener("click",function a(s){var i=s.target.id;new
31 Audio("../data/click.wav").play(),t.updateGame(i),e.removeEventListener("click",a,!1)}
32 }))}}function disableAlphabetButtons(){var t=document.getElementById("alphabet").get-
33 ElementsByTagName("div");for(i=0;i<t.length;i++)t.item(i).className+="";
34 alphabetDisabled"}!function(){var t=new WebSocket(Setup.WEB_SOCKET_URL),e=new
35 VisibleWordBoard,a=new StatusBar;createBalloons();var s=new GameState(e,a,t),i=new
36 AlphabetBoard(s);t.onmessage=function(r){let n=JSON.parse(r.data);if(n.type==Messages.
37 T_PLAYER_TYPE)if(s.setPlayerType(n.data),"A"==s.getPlayerType()){
38 disableAlphabetButtons(),a.setStatus(Status.player1Intro);let i=-1,r=Status.prompt,n=
39 null;for(;i<0;)null==(n=Prompt(r))?r=Status.prompt:(n=n.toUpperCase()).length<Setup.
40 MIN_WORD_LENGTH||n.length>Setup.MAX_WORD_LENGTH?r=Status.promptAgainLength:0==/^([a-zA-
41 Z]+$/).test(n)?r=Status.promptChars:0==englishDict.hasOwnProperty(n.toLocaleLowerCase())
42 ?r=Status.promptEnglish:i=1;a.setStatus(Status.chosen+n),s.setTargetWord(n),s.
43 initializeVisibleWordArray(),e.setWord(s.getVisibleWordArray());let o=Messages.
44 O_TARGET WORD;o.data=n,t.send(JSON.stringify(o))}else a.setStatus(Status.
45 player2IntroNoTargetYet);n.type==Messages.T_TARGET WORD&&"B"==s.getPlayerType()&&(s.
46 setTargetWord(n.data),a.setStatus(Status.player2Intro),s.initializeVisibleWordArray(),
47 i.initialize(),e.setWord(s.getVisibleWordArray()),n.type==Messages.T_MAKE_A_GUESS&&"A"
48 ==s.getPlayerType()&&(a.setStatus(Status.guessed+n.data),s.updateGame(n.data)),t.
49 onopen=function(){t.send("{}"),t.onreadystatechange=null==s.whoWon()&&a.setStatus(
50 Status.aborted),t.onerror=function(){}})()|}

```

all of interaction.js; 50% bytes saved  
 interaction.js  
 first ~50 lines (266 in total)

```
function GameState(t,e,a){  
    this.wrongGuesses=0,th  
    alphabet.initialize(),  
    this.initializeVisible  
    targetWord.length),thi  
    function(){return thi  
    string"==typeof t,"%s:  
    this.playerType=t},thi  
    %s: Expecting a string  
    this.getVisibleWordArr  
    function(){if(this.wro  
    document.getElementById  
    "../data/pop.wav").play  
    .MAX_ALLOWED_GUESSES?"  
    revealLetters=function  
    string, got a %s",argu  
    %s: Expecting an array  
    visibleWordArray[e[a]]  
    .targetWord},this.upd  
    Expecting a string, go  
    getLetterInWordIndices  
    revealLetters(t,s),thi  
    setWord(this.visibleWo  
    stringify(i));let r=th  
    querySelectorAll(".alp  
    neNode(!0);t.parentNod  
    .gameLost,t+=Status.pl  
    O_GAME WON_BY;t.data=r  
    {this.initialize=funct  
    .forEach(function(e){e  
    Audio("../data/click.w  
    })})}function disableA  
    ElementsByTagName("div  
    alphabetDisabled"}!fun  
    VisibleWordBoard,a=new  
    AlphabetBoard(s);t.onm  
    T_PLAYER_TYPE)if(s.set  
    disableAlphabetButtons  
    null;for(;i<0;)null==(  
    MIN_WORD_LENGTH||n.len  
    Z]+$/test(n)?r=Status  
    )?r=Status.promptEngli  
    initializeVisibleWordA  
    O_TARGET_WORD;o.data=n  
    player2IntroNoTargetYe  
    setTargetWord(n.data),  
    i.initialize(),e.setWo  
    "=="s.getPlayerType()&&  
    onopen=function(){t.se  
    Status.aborted}),t.one
```

```
1 function GameState(t, e, a) {  
2     this.playerType = null, this.MAX_ALLOWED = Setup.MAX_ALLOWED_GUESSES, this.wrongGuesses =  
3         0, this.visibleWordArray = null, this.alphabet = new Alphabet, this.alphabet.  
4         initialize(), this.visibleWordBoard = t, this.targetWord = null, this.statusBar = e,  
5         this.initializeVisibleWordArray = function() {  
6             this.visibleWordArray = new Array(this.targetWord.length), this.visibleWordArray.fill(  
7                 Setup.HIDDEN_CHAR)  
8         }, this.getPlayerType = function() {  
9             return this.playerType  
10        }, this.setPlayerType = function(t) {  
11            console.assert("string" == typeof t, "%s: Expecting a string, got a %s", arguments.  
12                callee.name, typeof t), this.playerType = t  
13        }, this.setTargetWord = function(t) {  
14            console.assert("string" == typeof t, "%s: Expecting a string, got a %s", arguments.  
15                callee.name, typeof t), this.targetWord = t  
16        }, this.getVisibleWordArray = function() {  
17            return this.visibleWordArray  
18        }, this.incrWrongGuess = function() {  
19            if (this.wrongGuesses++, null == this.whoWon()) {  
20                let t = "b" + this.wrongGuesses;  
21                document.getElementById(t).className += " balloonGone", setTimeout(function() {  
22                    new Audio("../data/pop.wav").play()  
23                }, 500)  
24            }  
25        }, this.whoWon = function() {  
26            return this.wrongGuesses > Setup.MAX_ALLOWED_GUESSES ? "A" : this.visibleWordArray.  
27                indexOf("#") < 0 ? "B" : null  
28        }, this.revealLetters = function(t, e) {  
29            console.assert("string" == typeof t, "%s: Expecting a string, got a %s", arguments.  
30                callee.name, typeof t), console.assert(e instanceof Array, "%s: Expecting an array",  
31                arguments.callee.name);  
32            for (let a = 0; a < e.length; a++) this.visibleWordArray[e[a]] = t  
33        }, this.revealAll = function() {  
34            this.visibleWordBoard.setWord(this.targetWord)  
35        }, this.updateGame = function(t) {  
36            console.assert("string" == typeof t, "%s: Expecting a string, got a %s", arguments.  
37                callee.name, typeof t);  
38            var s = this.alphabet.getLetterInWordIndices(t, this.targetWord);  
39            0 == s.length ? this.incrWrongGuess() : this.revealLetters(t, s), this.alphabet.  
40                makeLetterUnAvailable(t), this.visibleWordBoard.setWord(this.visibleWordArray);  
41            var i = Messages.O_MAKE_A_GUESS;  
42            i.data = t, a.send(JSON.stringify(i));  
43            let r = this.whoWon();  
44            if (null != r) {  
45                this.revealAll();  
46                var n = document.querySelectorAll(".alphabet");  
47                let t;
```

# Code obfuscation

```
/* ESLint global variables information */
/* global Setup, Status, Messages, englishD
/* basic constructor of game state */
function GameState(visibleWordBoard, sb, so

    this.playerType = null;
    this.MAX_ALLOWED = Setup.MAX_ALLOWED_GU
    this.wrongGuesses = 0;
    this.visibleWordArray = null;
    this.alphabet = new Alphabet();
    this.alphabet.initialize();
    this.visibleWordBoard = visibleWordBoar
    this.targetWord = null;
    this.statusBar = sb;

    this.initializeVisibleWordArray = funct
        this.visibleWordArray = new Array(t
            this.visibleWordArray.fill(Setup.HI
};

    this.getPlayerType = function () {
        return this.playerType;
    };

    this.setPlayerType = function (p) {
        console.assert(typeof p == "string"
        this.playerType = p;
    };

    this.setTargetWord = function (w) {
        console.assert(typeof w == "string"
        this.targetWord = w;
    };

    this.getVisibleWordArray = function(){
        return this.visibleWordArray;
    };

    this.incrWrongGuess = function(){
        this.wrongGuesses++;

        if(this.whowon() == null){
            //kill a balloon
            let id = "b"+this.wrongGuesses;
            document.getElementById(id).cla
            setTimeout(function () {
                new Audio("../data/pop.wav"
            }, 500);
        }
    };
}
```

```
var _0x13fe=['name','setTargetWord','assert','getVisibleWordArra
getElementById','\x20balloonGone','../data/pop.wav','play','w
revealLetters','%s:\x20Expecting\x20an\x20array','length','re
getLetterInWordIndices','makeLetterUnAvailable','0_MAKE_A_GUESS','send',' stringify',
querySelectorAll','.alphabet','from','forEach','cloneNode','parentNode','replaceChild',
gameLost','setStatus','0_GAME WON_BY','close','addEventListene
click','target','
../data/click.wav','div','item','\x20alphabetDisabled','WEB_SOCKET_URL','onmessage',
parse,'data','T_PLAYER_TYPE','getPlayerType','player1Intro','prompt','toUpperCase',
MIN_WORD_LENGTH','MAX_WORD_LENGTH','promptAgainLength','test','promptChars',
'hasOwnProperty','toLocaleLowerCase','promptEnglish','0_TARGET_WORD','type',
T_TARGET_WORD','T_MAKE_A_GUESS','updateGame','aborted','MAX_ALLOWED',
'MAX_ALLOWED_GUESSES','wrongGuesses','visibleWordArray','alphabet','initialize',
visibleWordBoard','targetWord','statusBar','initializeVisibleWordArray','playerType',
'setPlayerType','string','%s:\x20Expecting\x20a\x20string,\x20got\x20a\x20%s','callee'];
(function(_0xb67384,_0x27295a){var _0x156d7c=function(_0x52a1f8){while(--_0x52a1f8){
_0xb67384['push'](_0xb67384['shift']());}};_0x156d7c(++_0x27295a);}_0x13fe,_0x172);var
_0x14ff=function(_0x594e55,_0x3e24bd){_0x594e55=_0x594e55-0x0;var _0x5cea58=_0x13fe[
_0x594e55];return _0x5cea58;};function GameState(_0x583931,_0x5cdcd4,_0x2f0ad4){this['
playerType']=null;this[_0x14ff('0x0')]=Setup[_0x14ff('0x1')];this[_0x14ff('0x2')]=0x0;
this[_0x14ff('0x3')]=null;this['alphabet']=new Alphabet();this[_0x14ff('0x4')][_0x14ff(
'0x5')]();this[_0x14ff('0x6')]=_0x583931;this[_0x14ff('0x7')]=null;this[_0x14ff('0x8')]=
_0x5cdcd4;this[_0x14ff('0x9')]=function(){this[_0x14ff('0x3')]=new Array(this[_0x14ff(
'0x7')]['length']);this[_0x14ff('0x3')]['fill'](Setup['HIDDEN_CHAR']);};this['
getPlayerType']=function(){return this[_0x14ff('0xa')];};this[_0x14ff('0xb')]=function(
_0x442164){console['assert'](typeof _0x442164==_0x14ff('0xc'),_0x14ff('0xd'),arguments[
_0x14ff('0xe')][_0x14ff('0xf')],typeof _0x442164);this['playerType']=_0x442164;};this[
_0x14ff('0x10')]=function(_0x4694eb){console[_0x14ff('0x11')](typeof _0x4694eb==_0x14ff(
'0xc'),'%s:\x20Expecting\x20a\x20string,\x20got\x20a\x20%s',arguments[_0x14ff('0xe')][
_0x14ff('0xf')],typeof _0x4694eb);this[_0x14ff('0x7')]=_0x4694eb;};this[_0x14ff('0x12')]=
function(){return this[_0x14ff('0x3')];};this[_0x14ff('0x13')]=function(){this[_0x14ff(
'0x2')]++;if(this['whowon']()==null){let _0xb7d465='b'+this[_0x14ff('0x2')];document[
_0x14ff('0x14')](_0xb7d465)['className']+=_0x14ff('0x15');setTimeout(function(){new
Audio(_0x14ff('0x16'))[_0x14ff('0x17')]();},0x1f4);}};this[_0x14ff('0x18')]=function(){
if(this[_0x14ff('0x2')]>Setup[_0x14ff('0x1')]){return'A';}if(this[_0x14ff('0x3')][
_0x14ff('0x19')]( '#')<0x0){return'B';}return null;};this[_0x14ff('0x1a')]=function(
_0x11dabc,_0x591157){console[_0x14ff('0x11')](typeof _0x11dabc=='string',_0x14ff('0xd'),
arguments[_0x14ff('0xe')][_0x14ff('0xf')],typeof _0x11dabc);console['assert'](_0x591157
instanceof Array,_0x14ff('0x1b'),arguments[_0x14ff('0xe')][_0x14ff('0xf')]);for(let
_0x2694ba=0x0;_0x2694ba<_0x591157[_0x14ff('0x1c')];_0x2694ba++){this[_0x14ff('0x3')][
_0x591157[_0x2694ba]]=_0x11dabc;}};this[_0x14ff('0x1d')]=function(){this[_0x14ff('0x6')][
_0x14ff('0x1e')](this[_0x14ff('0x7')]);};this['updateGame']=function(_0x5c88a1){console[
_0x14ff('0x11')](typeof _0x5c88a1==_0x14ff('0xc'),_0x14ff('0xd'),arguments['callee'][
_0x14ff('0xf')],typeof _0x5c88a1);var _0x3a7220=this[_0x14ff('0x4')][_0x14ff('0x1f')][
_0x5c88a1,this[_0x14ff('0x7')]];if(_0x3a7220[_0x14ff('0x1c')]==0x0){this[_0x14ff('0x13')][
()];}else{this[_0x14ff('0x1a')](_0x5c88a1,_0x3a7220);}}this[_0x14ff('0x4')][_0x14ff('0x20
')]( _0x5c88a1);this[_0x14ff('0x6')][_0x14ff('0x1e')](this[_0x14ff('0x3')]);var _0x438e50
=Messages[_0x14ff('0x21')];_0x438e50['data']=_0x5c88a1;_0x2f0ad4[_0x14ff('0x22')](JSON[
_0x14ff('0x23')]( _0x438e50));let _0x3a6e16=this[_0x14ff('0x18')]();if(_0x3a6e16!=null){
this[_0x14ff('0x1d')]();var _0x5cdfe1=document[_0x14ff('0x24')][_0x14ff('0x25')];Array[
_0x14ff('0x26')](_0x5cdfe1)[_0x14ff('0x27')](function(_0x35f612){var _0x169fe3=_0x35f612
[_0x14ff('0x28')](!![]);_0x35f612[_0x14ff('0x29')][_0x14ff('0x2a')](_0x169fe3,_0x35f612
);});let _0x4d88fc;if(_0x3a6e16==this[_0x14ff('0xa')]){_0x4d88fc=Status['gameWon'];}else{
_0x4d88fc=Status[_0x14ff('0x2b')];}_0x4d88fc+=Status['playAgain'];_0x5cdcd4[_0x14ff(
'0x2c')]( _0x4d88fc);if(this[_0x14ff('0xa')]==='B'){let _0x464e12=Messages[_0x14ff('0x2d')]
```

# Node.js modules

- Code can be organised in **modules**
- Not all functions and variables in a module are exposed
  - Exposed elements have to be made known explicitly
- Modules can be **published** to [npmjs.com](https://npmjs.com)
  - Makes distribution of modules to other developers easy, e.g.

```
npm install --save alexa-sdk
```



Search packages

Search

log in or sign up

Share your code. npm Orgs help your team discover, share, and reuse code. [Create a free org »](#)

## express

4.16.4 • Public • Published a month ago

[Readme](#)[30 Dependencies](#)[28,739 Dependents](#)[261 Versions](#)

# express

Fast, unopinionated, minimalist web framework for [node](#).[npm v4.16.4](#) [downloads 19M/m](#) [linux passing](#) [windows passing](#) [coverage 100%](#)

```
var express = require('express')
var app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})

app.listen(3000)
```

install

`> npm i express`

↓ weekly downloads

6,167,803



version

4.16.4

license

MIT

open issues

105

pull requests

51

homepage

expressjs.com

repository

[github](#)

last publish

a month ago

```
claudiahauff@wlan-145-94-186-167:~/local/pienapple-front-master $ npm list  
pienapple-front@1.0.0 /local/pienapple-front-master  
+-- autoprefixer@6.4.0  
+-- browserslist@1.3.5  
+-- caniuse-db@1.0.30000517  
+-- normalize-range@0.1.2  
+-- num2fraction@1.2.2  
+-- postcss@5.1.1  
| +-- js-base64@2.1.9  
| +-- postcss-value-parser@3.3.0  
+-- babel-core@6.11.4  
| +-- babel-code-frame@6.11.0  
| | +-- chalk@1.1.3  
| | +-- ansi-styles@2.2.1  
| | +-- escape-string-regexp@1.0.5  
| | +-- has-ansi@2.0.0  
| | | +-- ansi-regex@2.0.0  
| | +-- strip-ansi@3.0.1  
| | +-- supports-color@2.0.0  
| +-- esutils@2.0.2  
| +-- js-tokens@2.0.0  
+-- babel-generator@6.11.4  
| +-- detect-indent@3.0.1  
| | +-- get-stdin@4.0.1  
| | +-- repeating@1.1.3  
| | | +-- is-finite@1.0.1  
| | | | +-- number-is-nan@1.0.0  
+-- babel-helpers@6.8.0  
+-- babel-messages@6.8.0  
+-- babel-register@6.11.6  
+-- core-js@2.4.1
```

# npm list

`==` (same value) vs. `==` (same type and value)

5 lines (4 sloc) | 82 Bytes

```
1 'use strict';  
2 module.exports = Number.isNaN || function (x) {  
3     return x !== x;  
4 };
```

Strict equality treats NaN as unequal to every other value -- including itself.

```
» var x = 3; typeof(x);  
← "number"  
» var y = NaN; typeof(y);  
← "number"  
» x == 3  
← true  
» x !== 3  
← false  
» y == NaN  
← false  
» y !== NaN  
← true
```

TECHNOLOGY LAB —

# Rage-quit: Coder unpublished 17 lines of JavaScript and “broke the Internet”

Dispute over module name in npm registry became giant headache for developers.

# npm search

2018-11-28 00:45:50 🏃 Cladias-MacBook-Air in ~/GitHub/Web-Teaching/demo-code/balloons-game

± |master → origin +11 {5} ✓| → npm search express

NAME	DESCRIPTION	AUTHOR	DATE	VERSION	KEYWORDS
express	Fast,...	=dougwilson...	2018-10-11	4.16.4	express framework sinatra web rest restful router app api
path-to-regexp	Express style path...	=blakeembrey...	2018-08-26	2.4.0	express regexp route routing
cors	Node.js CORS...	=dougwilson...	2018-11-04	2.8.5	cors express connect middleware
morgan	HTTP request logger...	=dougwilson	2018-09-11	1.9.1	express http logger middleware
apollo-server-express	Production-ready...	=abernix...	2018-11-26	2.2.3	GraphQL Apollo Server Express Connect Javascript
express-validator	Express middleware...	=ctavan...	2018-07-23	5.3.0	express validator validation validate sanitize sanitization xss
express-graphql	Production ready...	=asiandrummer...	2018-10-29	0.7.1	express restify connect http graphql middleware api
helmet	help secure...	=adam_baldwin...	2018-11-07	3.15.0	security headers express connect x-frame-options x-powered-by cs
serve-favicon	favicon serving...	=dougwilson	2018-03-29	2.5.0	express favicon middleware
@babel/helper-optimise-ca ll-expression	Helper function to...	=nicolo-ribaudo...	2018-08-27	7.0.0	
@feathersjs/express	Feathers Express...	=corymsmith...	2018-09-21	1.2.7	feathers feathers-plugin
@babel/helper-member-exp ession-to-functions	Helper function to...	=nicolo-ribaudo...	2018-08-27	7.0.0	
babel-helper-optimise-cal l-expression	Helper function to...	=hzoo...	2017-04-07	6.24.1	
babel-helper-explode-assi gnable-expression	Helper function to...	=hzoo...	2017-04-07	6.24.1	
micromatch	Glob matching for...	=doowb =es128...	2018-03-22	3.1.10	bash expand expansion expression file files filter find glob glo
aws-serverless-express	This library...	=amzn-oss...	2018-08-20	3.3.5	aws serverless api gateway lambda express
@nguniversal/express-engi ne	Express Engine for...	=nguniversal	2018-10-19	7.0.2	express ssr universal
ignore	Ignore is a manager...	=kael	2018-11-03	5.0.4	ignore .gitignore gitignore npmignore rules manager filter regex
express-rate-limit	Basic IP...	=nfriedly	2018-11-12	3.3.2	express-rate-limit express rate limit ratelimit rate-limit middl
escape-string-regexp	Escape RegExp...	=sindresorhus	2016-02-21	1.0.5	escape regex regexp re regular expression string str special cha

## npmjs documentation

**Semantic versioning:**  
**Major.Minor.Patch**  
(breaking backwards compatibility requires Major increment)

Code status	Stage	Rule	Example version
First release	New product	Start with 1.0.0	1.0.0
Backward compatible bug fixes	Patch release	Increment the third digit	1.0.1
Backward compatible new features	Minor release	Increment the middle digit and reset last digit to zero	1.1.0
Changes that break backward compatibility	Major release	Increment the first digit and reset middle and last digits to zero	2.0.0

```
2018-11-28 09:47:53 wlan-145-94-167-249 in ~/GitHub/Web-Teaching/demo-code/balloons-game
[± |master {5} ✓| → more package.json
{
  "name": "balloons",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node app.js 3000",
    "test": "jest --coverage"
  },
  "dependencies": {
    "cookie-parser": "~1.4.3",
    "debug": "~2.6.9",
    "ejs": "~2.5.7",
    "express": "^4.16.3",
    "http-errors": "~1.6.2",
    "morgan": "~1.9.0",
    "ws": "^6.0.0"
  },
  "devDependencies": {
    "eslint": "^5.4.0",
    "jest": "^23.5.0"
  }
}
```

**tilde range:  $\geq 1.4.3 < 1.5.0$**

**caret range:  $\geq 4.16.3 < 5.0.0$**

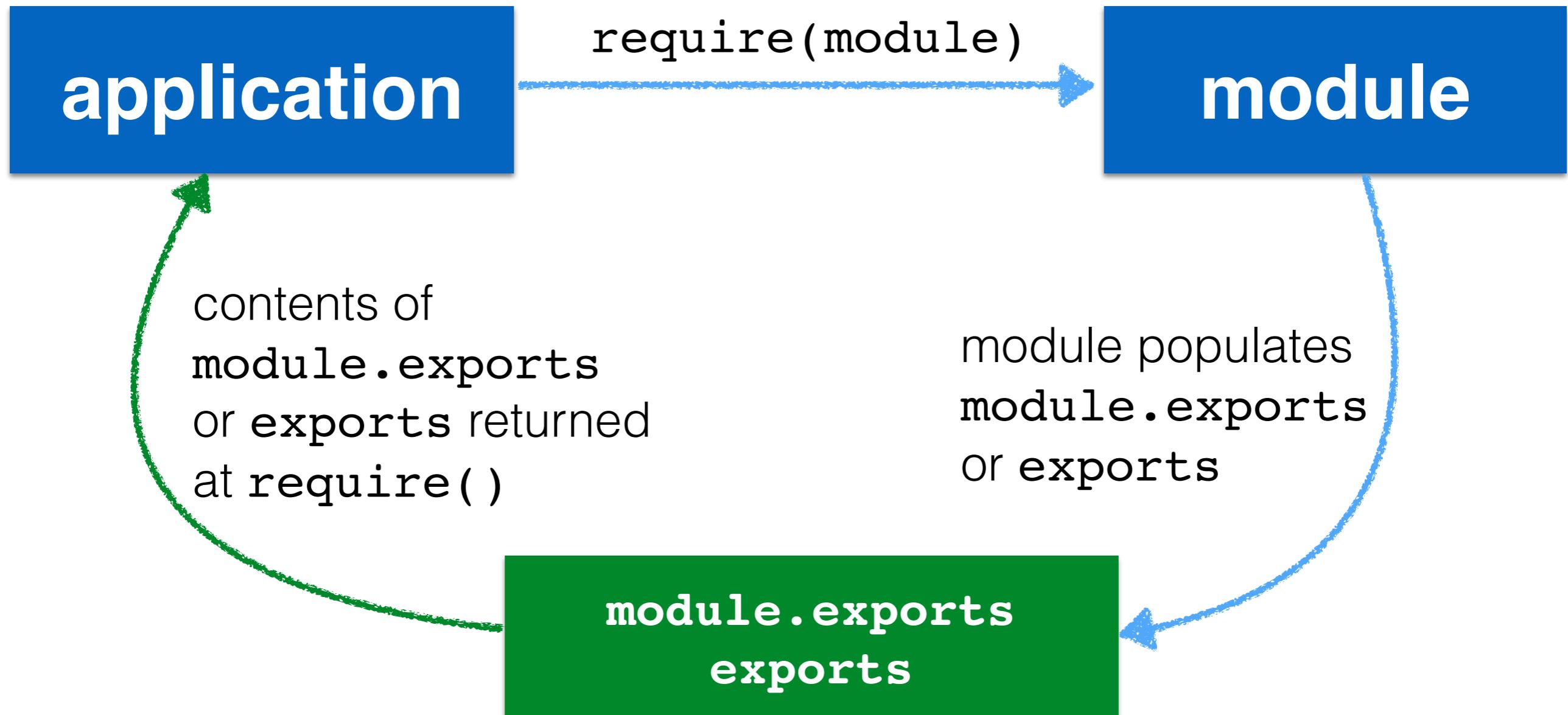
# A file-based module system

Do you remember how much effort we put into the module design pattern?

- A **file is its own module**; no pollution of the global namespace
- A file accesses its **module definition** through the **module** object
- **module.exports** (or its alias **exports**) makes code externally available
- To import a module, use the globally available **require** function

```
module {           console.log(module)
  id: '.',          id: 'nodejs-file',
  exports: {},      exports: {},
  parent: null,     parent: null,
  filename: '/path/to/nodejs-file.js', filename: '/path/to/nodejs-file.js',
  loaded: false,    loaded: false,
  children: [],     children: [],
  paths:            paths: [
    '/Users/path/path2/node_modules', '/Users/path/node_modules',
    '/Users/path/node_modules',     '/Users/node_modules' ]
}
```

# App-module cycle



# A first example

foo.js

```
var fooA = 1;
module.exports = "Hello!";
module.exports = function() {
  console.log("Hi from foo!");
}
```

bar.js

```
var foo = require('./foo');
foo();
require('./foo')();
console.log(foo);
console.log(foo.toString());
console.log(fooA);
console.log(module.exports);
```

Node.js **runs** the referenced JavaScript file in **a new scope** and returns the **final value** of **module.exports**

Hi from foo!

[Function]

{ }

```
function () {
  console.log("Hi from foo!");
}
```

ReferenceError

# require()

- `require()` is blocking
- `module.exports` is **cached**, i.e. the first time `require(a_file)` is called, `a_file` is read from disk, and subsequently the **in-memory object is returned**
- Useful to share states between modules (configuration, etc.)

```
var t1 = new Date().getTime();
var foo1 = require('./foo');
console.log(new Date().getTime() - t1);
// > 0 (unless the machine is very fast)
```

```
var t2 = new Date().getTime();
var foo2 = require('./foo');
console.log(new Date().getTime() - t2); // approx 0
```

# module.exports

- `module.exports={}` is by default present in every Node.js file (a new empty object)
- Node.js provides an alias: `exports = module.exports`

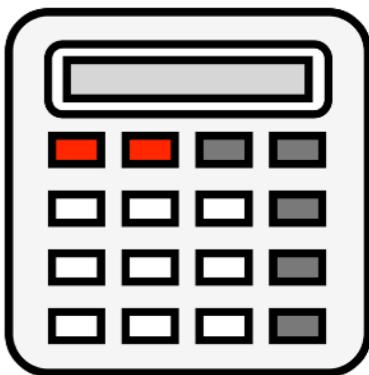
```
module.exports.foo = function () {  
    console.log('foo called');  
};
```

```
module.exports.bar = function () {  
    console.log('bar called');  
};
```

```
exports.foo = function () {  
    console.log('foo called');  
};  
  
exports.bar = function () {  
    console.log('bar called');  
};
```

equivalent

- Important: do **not assign** to `exports` directly, **attach** to it instead (otherwise the reference to `module.exports` is broken); assign only to `module.exports`



# Creating a rounding module

A module can be

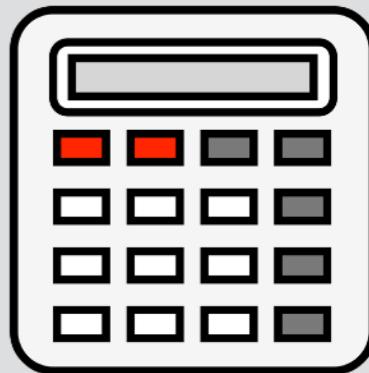
- a single file, or
- a directory of files (which includes a file `index.js`)

```
1 function roundGrade(grade) {  
2     return Math.round(grade);  
3 }  
4  
5 function roundGradeUp(grade) {  
6     return Math.round(0.5+parseFloat(grade));  
7 }  
8 exports.maxGrade = 10;  
9 exports.roundGradeUp = roundGradeUp;  
10 exports.roundGradeDown = function(grade) {  
11     return Math.round(grade-0.5);  
12 }
```

not exposed in this module;  
application cannot use it

determines what exactly is  
exposed to the outer world

# Using a module



require returns the contents  
of the exports object

```
1 var express = require("express");
2 var url = require("url");
3 var http = require("http");
4 var grading = require("./grades");
5 var app;
6
7 var port = process.argv[2];
8 app = express();
9 http.createServer(app).listen(port);
10
11 app.get("/round", function (req, res) {
12   var query = url.parse(req.url, true).query;
13   var grade = ( query["grade"] !=undefined ) ?
14     query["grade"] : "0";
15   res.send("Rounding up: " +
16           grading.roundGradeUp(grade) + ", and down: " +
17           grading.roundGradeDown(grade));
18 });


```

adding our module  
(current directory)

accessing module functions

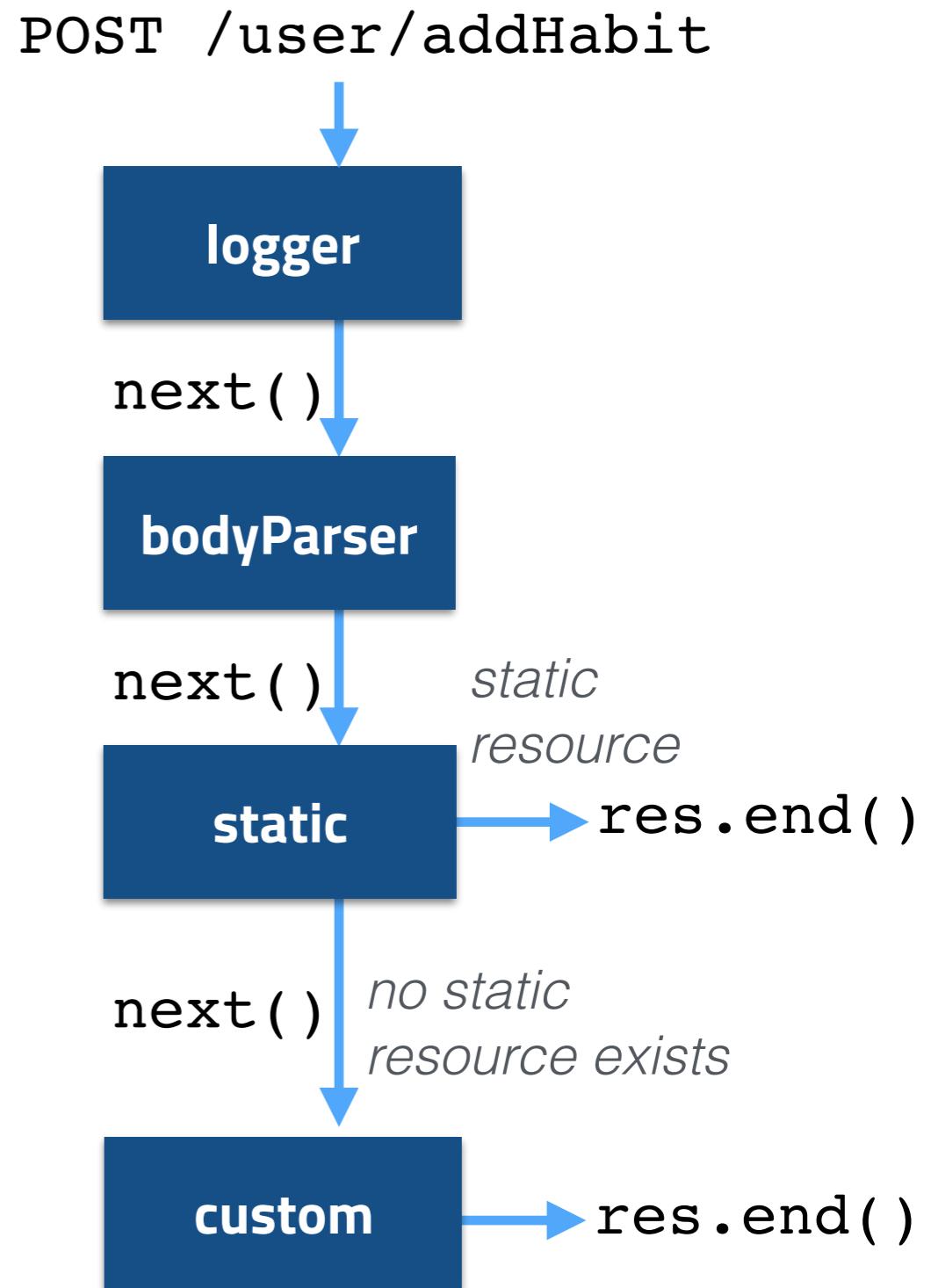
# Middleware in Express

# Middleware components

- **Small, self-contained** and **reusable** across applications
- Middleware components take **three arguments**:
  - **HTTP request** object
  - **HTTP response** object
  - Callback function (**next()**) to indicate that the component is finished and the *dispatcher* (orchestration component) can move to the next component

# Middleware abilities

- **Execute** code
- **Alter** the request and response objects
- **End** the request-response cycle
- **Call** the next middleware function in the middleware stack



# A simple logger component

- **Goal:** create a log file which records the request method and the URL of the request coming into the server
- **Required:** JavaScript function which accepts the request and response objects and the `next()` callback function

```
1 var express = require('express');
2
3 //a middleware logger component
4 function logger(request, response, next) {
5   console.log('%s %t %s %t %s', new Date(),
6             request.method, request.url,
7             next());
8 }
9 var app = express();
10 app.use(logger);
11 app.listen(3001);
```

**string substitutions**

**control shifts to next middleware function**

**register middleware component**

# A simple logger and “Hello World” response

```
1 var express = require('express');
2
3 function logger(request, response, next) { .... }
4
5 function helloWorld(request, response, next) {
6   response.setHeader('Content-Type',
7                     'text/plain');
8   response.end('Hello World!');
9 }
10
11 var app = express();
12 app.use(logger);
13 app.use(helloWorld);
14 app.listen(3001);
```

No call to next! Response has been sent.

any number of components can be registered

their order matters

# An authorisation component

```
1 app.use(function (req, res, next) {  
2     var auth = req.headers.authorization;  
3     if (!auth) {  
4         return next(new Error("Unauthorized access!"));  
5     }  
6  
7     //extract username and password  
8     var parts = auth.split(' ');  
9     var buf = new Buffer(parts[1], 'base64');  
10    var login = buf.toString().split(':');  
11    var user = login[0];  
12    var password = login[1];  
13  
14    //compare to 'correct' username/password combination  
15    //hardcoded for demonstration purposes  
16    if (user === "user" && password === "password") {  
17        next();  
18    }  
19    else {  
20        return next(new Error("Wrong username/password combination!"));  
21    }  
22});
```

We can pass errors through next.

Only a correct login/password combination leads to next()

Demo time 3

# Components are configurable

- **Goal:** middleware components are reusable across applications
- **Approach:** wrap original middleware function in a setup function

```
1 function setup(options) {  
2     // setup logic  
3     return function(req, res, next) {  
4         // middleware logic  
5     }  
6 }  
7  
8 app.use( setup({ param1 : 'value1' }) );
```

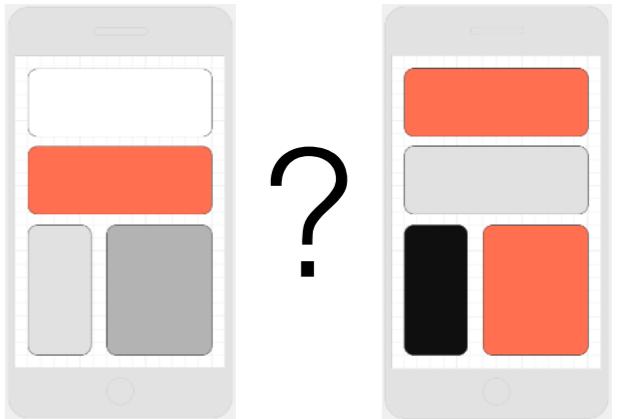
important: function call is made!

# Routing

# Overview

- Mechanism by which requests (as specified by a URL and HTTP method) **are routed to the code** that handles them
  - Request types: GET /user vs. POST /user
  - Request routes: GET /user vs. GET /admin
- In the past: **file-based routing**
  - File contact.html accessed through `http://my_site/contact.html`

# Routes and Express



- We have used routes already
- **Route handlers are middleware**

```
//clients requests todos
app.get("/todos", function (req, res, next) {
  //hardcoded "A-B" testing
  if (Math.random() < 0.5) {
    return next();
  }
  console.log("Todos in schema A returned");
  res.json(todosA);
};

app.get("/todos", function (req, res, next) {
  console.log("Todos in schema B returned");
  res.json(todosB);
});
```

Two route handlers defined for a route

half the requests will move on

# Routes and Express

```
//A-B-C testing
app.get('/todos',
  function(req,res, next){
    if (Math.random() < 0.33) {
      return next();
    }
    console.log("Todos in schema A returned");
    res.json(todosA);
  },
  function(req,res, next){
    if (Math.random() < 0.5) {
      return next();
    }
    console.log("Todos in schema B returned");
    res.json(todosB);
  },
  function(req,res){
    console.log("Todos in schema C returned");
    res.json(todosC);
  }
);
```

a single `app.get()` can contain multiple handlers

Idea: create **generic functions** that can be dropped into **any route**

# Routing paths and string patterns

- Routes are converted into **regular expressions** (*patterns to match character combinations in strings*) by Express
- **Route path types**: strings (*you have seen those*), **string patterns** (*we cover those now*), regular expressions (*we ignore those*)
- Express supports only a **subset** of the standard regex meta-characters
- Available regex meta-characters: **+ ? \* ( ) [ ]**

# Routing paths and string patterns

- Routes are converted into **regular expressions** (*patterns to match character combinations in strings*) by Express
- **Route path types**: strings (*you have seen those*), **string patterns** (*we cover those now*), regular expressions (*we ignore those*)
- Expressions can contain the following standard regex meta-characters:
- Available **preceding element (regex)!** ( ) [ ]

In other languages/frameworks:  
zero or more occurrences of the preceding element (regex)!

Character	Description	Regex	Matched expressions
+	one or more occurrences	ab+cd	abcd, abbcd, ...
?	zero or one occurrence	ab?cd	acd, abcd
*	zero or more occurrences of any char (wildcard)	ab*cd	abcd, ab1234cd, ...
[...]	match anything inside for one character position	ab[cd]?e	abe, abce, abde
(...)	boundaries	ab(cd)?e	abe, abcde

## string pattern

```
app.get('/user(name)?s+', function(req,res){  
  res.send(...)  
});
```

FYI only! We ignore those in this course.

regular expression, delimited by / /

```
app.get('/.*users$/',
  function(req, res) {
  res.send(...)
}) ;
```

```
(?:(a-z0-9!#$%&'*+/=?^_`{|}|~-]+(?:(\.[a-z0-9!#$%&'*+/=?^_`{|}|~-]+)*|"(?:[\\x01-\x08\x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]|\\\\[\\x01-\x09\x0b\x0c\x0e-\x7f])*")@(?:(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?)?|[(?:(?:2(5[0-5]|[0-4][0-9])|1[0-9][0-9]|1-9?[0-9]))\.\.){3}(?:2(5[0-5]|0-4|[0-9])|1[0-9][0-9]|1-9?[0-9])|[a-z0-9-]*[a-z0-9]:(?:(?:[\\x01-\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]|\\\\[\\x01-\x09\x0b\x0c\x0e-\x7f])+)\])
```

# Routing parameters

Enable **variable input** as part of the route

```
var todoTypes = {  
    important: ["TI1506", "OOP", "Calculus"],  
    urgent: ["Dentist", "Hotel booking"],  
    unimportant: ["Groceries"],  
};  
  
app.get('/todos/:type', function (req, res, next) {  
    var todos = todoTypes[req.params.type];  
    if (!todos) {  
        return next(); // will eventually fall through to 404  
    }  
    res.send(todos);  
});
```

```
localhost:3000/todos/important  
localhost:3000/todos/urgent  
localhost:3000/todos/unimportant
```

# Routing parameters

Enable **variable input** as part of the route

```
var todoTypes = {  
    important: ["T11506", "ODP", "G111111"],  
    urgent: ["Dentist", "Meeting"],  
    unimportant: ["Groceries", "Housework"]};  
  
app.get('/todos/:type', function (req, res, next) {  
    var todos = todoTypes[req.params.type];  
    if (!todos) {  
        return next(); // will eventually fall through to 404  
    }  
    res.send(todos);  
});
```

localhost:3000/todos/important  
localhost:3000/todos/urgent  
localhost:3000/todos/unimportant

Will match any string that does not contain /. It is available with key type in the req.params object.

# Routing parameters

localhost:3000/todos/important/tomorrow

```
var todoTypes = {  
    important: {  
        today: ["TI1506"],  
        tomorrow: ["OOP", "Calculus"]  
    },  
    urgent: {  
        today: ["Dentist", "Hotel booking"],  
        tomorrow: []  
    },  
    unimportant: {  
        today: ["Groceries"],  
        tomorrow: []  
    }  
};  
app.get('/todos/:type/:level', function (req, res, next) {  
    var todos = todoTypes[req.params.type][req.params.level];  
    if (!todos) {return next();}  
    res.send(todos);  
});
```

No restrictions on the number of variable input

# Organizing routes

- Keeping routes in the main application file becomes unwieldy as the codebase grows
- Move routes into a module and pass `app` instance into the module

routes.js

```
module.exports = function(app){  
  app.get('/', function(req,res){  
    res.send(...);  
  })  
  //...  
};
```

my\_app.js

```
require('./routes.js')(app);
```

# Templating with EJS

# Recall ... Express & HTML

```
1 var express = require("express");
2 var url = require("url");
3 var http = require("http");
4 var app;
5
6 var port = process.argv[2];
7 app = express();
8 http.createServer(app).listen(port);
9
10 app.get("/greetme", function (req, res) {
11   var query = url.parse(req.url, true).query;
12   var name = ( query["name"] !=undefined) ? query[
13     "name" ] : "Anonymous";
14   res.send("<html><head></head><body><h1>
15           Greetings "+name+"</h1></body></html>
16           ");
17 });
18
19 app.get("/goodbye", function (req, res) {
20   res.send("Goodbye you!");
21 });
```

**error-prone, not maintainable**, fails at anything larger than a toy project.



one extra request/response cycle

# Recall ... Ajax

The screenshot shows a Twitter profile page for Claudia Hauff (@CharlotteHase) on the left, and a feed of tweets from various users like Oxford Comp Sci Retweeted, Open Data Institute, and Dura Vermeer. On the right, there's a sidebar for "Who to follow". Below the main content, the browser's developer tools Network tab is open, displaying a list of requests. A large blue callout bubble with the text "XMLHttpRequest" is overlaid on the table, pointing to a specific request row.

Status	Method	F...			Size	0 ms	20.48 s	40.96 s
304	GET	Kkhapl...	🔒		7.37 KB	→ 6 ms		
304	GET	BJ5Cm...	🔒	vid... xhr	mp2t	cached	27.72 KB	→ 4 ms
200	GET	timeline...	🔒	twi... xhr	js	1.42 KB	368 B	→ 178 ms
200	GET	timeline...	🔒	twi... xhr	js	1.42 KB	368 B	→ 168 ms
200	GET	toast_p...	🔒	twi... xhr	js	1.36 KB	236 B	→ 192 ms
200	GET	timeline...	🔒	twi... xhr	js	1.42 KB	368 B	→ 185 ms
200	GET	timeline...	🔒	twi... xhr	js	1.42 KB	368 B	→ 184 ms
200	GET	timeline...	🔒	twi... xhr	js	1.42 KB	368 B	→ 184 ms

163 requests | 7.98 MB / 6.89 MB transferred | Finish: 52.75 s | DOMContentLoaded: 1.46 s | load: 1.61 s

# Instead ... templates

- **Goal:** write as little HTML “by hand” as possible



- Separates logic from presentation markup
- Different **template engines** exist for Node.js
- We use **EJS** (**E**mbedded **J**a**S**cript), a popular **template engine** and language
- Different **versions** of EJS exist, we use v2 (latest)  
<https://www.npmjs.com/package/ejs>

# Model-View-Controller (MVC)

- Standard **design pattern** to keep **logic**, **data** and **presentation** separate
- User request of a resource from the server triggers a cascade of events:
  1. **controller** requests application data from the **model**
  2. **controller** passes the data to the **view**
  3. **view** formats the data for the user (template engines)

# EJS in examples

```
1 var ejs = require('ejs');  
2 var template = '<%= message %>';  
3 var context = {message: 'Hello template!'};  
4 console.log(ejs.render(template, context));
```

<%= outputs the value into the template (HTML escaped)

- **Node REPL:** Read–Eval–Print–Loop implementation to quickly test Node snippets (an alternative to VSC)

- Start the REPL via

```
node -e "require('repl').start({ignoreUndefined:true})"
```

evaluate args as JS

# EJS in examples

<%- outputs the value into the template  
(unespaced); enables cross-site scripting attacks

```
1 var ejs = require('ejs');  
2 var template = '<%- message %>'; //UNESCAPED  
3 //var template = '<%= message %>'; //ESCAPED  
4 var context = {message: "<script>alert('hi!');</script>"};  
5 console.log(ejs.render(template, context));
```

- **Node REPL:** Read–Eval–Print–Loop implementation to quickly test Node snippets (an alternative to VSC)
- Start the REPL via  
`node -e "require('repl').start({ignoreUndefined:true})"`

# EJS: user-defined functions

- Often you want to make slight changes: **transform** or **filter**

```
1 var ejs = require('ejs');  
2  
3 var people = ['wolverine', 'paul', 'picard'];  
4 var transformUpper = function (inputString) {  
5     return inputString.toUpperCase();  
6 }  
7 // define your own function  
8 var template = '<%= helperFunc(input.join(", "))+  
9 ; %>';  
10 var context = {  
11     input: people,  
12     helperFunc: transformUpper  
13 };  
14  
15 console.log(ejs.render(template, context));
```

define your own function

function

array

define the context object

# JavaScript within EJS templates

use <% for control-flow purposes; no output

```
1 var ejs = require('ejs');
2
3 var template = "<% if(movies.length>2){movies.forEach(function(m){console.log(
4   m.title)})} %>";
5
6 var context = {'movies': [
7   {title:'The Hobbit', release:2014},
8   {title:'X-Men', release:2016},
9   {title:'Superman V', release:2014}
10  ]};
11 ejs.render(template, context);
```

Array.prototype.forEach()

applied to each element

# Configuring views with Express

- **Setting** the **views** directory (the directory containing the templates)

```
app.set('views', __dirname + '/views');
```

- **Setting** the **template engine**

```
app.set('view engine', 'ejs');
```

- Creating **template files**

Demo time 6

# Exposing data to views

```
1 var express = require("express");
2 var url = require("url");
3 var http = require("http");
4 var app;
5
6 var port = process.argv[2];
7 app = express();
8 http.createServer(app).listen(port);
9
10 var todos = [];
11 todos.push({ message: 'Midterm exam tomorrow',
12               dueDate: '01/12/2015' });
13 todos.push({ message: 'Prepare for assignment
14               5', dueDate: '05/01/2016' });
15 todos.push({ message: 'Sign up for final exam',
16               dueDate: '06/01/2016' });
17
18
19 app.set('views', __dirname + '/views');
20 app.set('view engine', 'ejs');
21
22 app.get("/todos", function (req, res) {
23   res.render('todos', { title: 'My list of
24                       TODOS', todo_array: todos });
25 })
```

the list of todos we want  
to serve to the clients

# Exposing data to views

```
1 var express = require("express");
2 var url = require("url");
3 var http = require("http");
4 var app;
5
6 var port = process.argv[2];
7 app = express();
8 http.createServer(app).listen(port);
9
10 var todos = [];
11 todos.push({ message: 'Midterm exam tomorrow',
12               dueDate: '01/12/2015' });
13 todos.push({ message: 'Prepare for assignment
14               5', dueDate: '05/01/2016' });
15 todos.push({ message: 'Sign up to
16               dueDate: '06/01/2016' });
17
18
19 app
20 app render() indicates
21 the use of a template
22 app.get('/todos', function (req, res) {
23   res.render('todos', { title: 'My list of
24   -odos', todo_array: todos });
25 }) template to use
```

the list of todos we want to serve to the clients

informing express about the view templates

me + '/views');

ejs'

variables of the template

variables of the template

variables of the template

variables of the template

# EJS template file

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title><%= title %></title>
5 </head>
6 <body>
7   <h1>TODOs</h1>
8   <div>
9     <% todo_array.forEach(function(todo) { %>
10    <div>
11      <h3><%= todo.dueDate %></h3>
12      <p><%= todo.message %></p>
13    </div>
14    <% } ) %>
15  </div>
16 </body>
17 </html>
```

JavaScript between `<% %>` is **executed**.

JavaScript between `<%= %>` **adds output** to the result file.

- Work on **Assignment 3**
- Book yourself an assessment timeslot (one per group)
- Start preparing for the midterm (~40 multiple choice questions on all web tech lectures)