JavaScript + Angular

Part 7

Components, $resource, node-js

Author: Andrey Kucherenko

node.js

globals

```
Global Objects
    Class: Buffer
    __dirname
    __filename
    clearImmediate(immediateObject)
    clearInterval(intervalObject)
    clearTimeout(timeoutObject)
    console
    exports
    global
    module
    process
    require()
    require.cache
    require.extensions
    require.resolve()
    setImmediate(callback[, arg][, ...])
    setInterval(callback, delay[, arg][, ...])
    setTimeout(callback, delay[, arg][, ...])
```

fs

```
fs.open(path, flags[, mode], callback)
fs.openSync(path, flags[, mode])
fs.read(fd, buffer, offset, length, position, callback)
fs.readdir(path[, options], callback)
fs.readdirSync(path[, options])
fs.readFile(file[, options], callback)
fs.readFileSync(file[, options])
fs.readlink(path[, options], callback)
fs.readlinkSync(path[, options])
fs.readSync(fd, buffer, offset, length, position)
fs.realpath(path[, options], callback)
fs.realpathSync(path[, options])
fs.rename(oldPath, newPath, callback)
fs.renameSync(oldPath, newPath)
fs.rmdir(path, callback)
fs.rmdirSync(path)
fs.stat(path, callback)
fs.statSync(path)
fs.symlink(target, path[, type], callback)
fs.symlinkSync(target, path[, type])
fs.truncate(path, len, callback)
...
```

events

```
const EventEmitter = require('events');

class MyEmitter extends EventEmitter {}

const myEmitter = new MyEmitter();
myEmitter.on('event', () => {
  console.log('an event occurred!');
});
myEmitter.emit('event');
```

path

```
path.basename(path[, ext])
path.delimiter
path.dirname(path)
path.extname(path)
path.format(pathObject)
path.isAbsolute(path)
path.join([path[, ...]])
path.normalize(path)
path.parse(path)
path.posix
path.relative(from, to)
path.resolve([path[, ...]])
path.sep
path.win32
```

express.js

```
npm install express --save
```

express.js

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

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

app.listen(3000, function () {
  console.log('Example app listening on port 3000!'
});
```

express.js

```
app.use(express.static('public'));

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

app.post('/', function (req, res) {
  res.send('Got a POST request');
});
```

Tasks

- Make service for your flower pot application(use server example from lesson 7)

components

```
module.directive(name, fn);

module.directive('directiveName', () => {
  return {
    scope: {},
    bindToController: {
      count: '='
    },
    controller: function () {...},
    controllerAs: 'ctrl',
    template: `...`
  };
});
```

```
module.component(name, options);

module.component('componentName', {
    bindings: {
      count: '='
    },
    controller: 'SomeCtrl as something',
    template: `...`
});
```

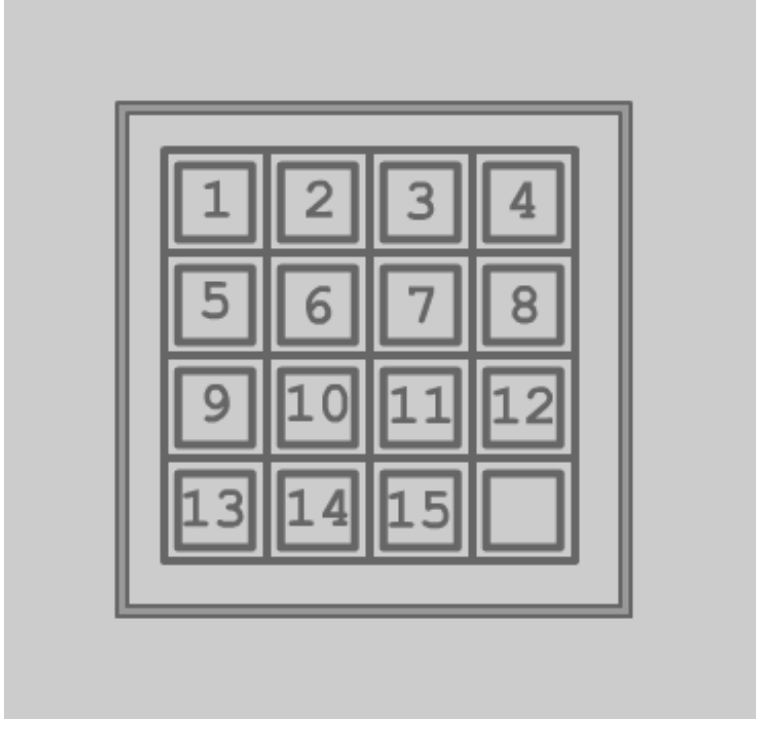
```
module.component('componentName', {
    bindings: {
      count: '='
    },
    require: {
      parent: '^^parentComponent'
    },
    controller: function () {
      // use this.parent to access required Objects
      this.parent.foo();
    }
    ...
});
```

```
module.component('componentName', {
    bindings: {
      count: '=',
      countONE: '<one'

    }
    ...
});
```

```
var myComponent = {
  bindings: {},
  controller: function () {
    this.$onInit = function() {

    };
    this.$postLink = function () {
      // fire away...
    };
    this.$onDestroy = function () {
      // component scope is destroyed
    };
  }
};
```

Tasks

- Make tag game



$resources

```
npm install angular-resource --save
```

```
import "angular-resource";

let app = angular.module('myApp.services', ['ngResource']);

app.factory('Entry', function($resource) {
    // Note the full endpoint address
    return $resource('/api/entries/:id');
})
/*
get()
query()
save()
remove()
delete()
*/
```

```
var User = $resource('/user/:userId', {userId:'@id'});
User.get({userId:123}, function(user) {
  user.abc = true;
  user.$save();
});

var CreditCard = $resource('/user/:userId/card/:cardId',
 {userId:123, cardId:'@id'}, {
  charge: {method:'POST', params:{charge:true}}
 });

// We can retrieve a collection from the server
var cards = CreditCard.query(function() {
  // GET: /user/123/card
  // server returns: [ {id:456, number:'1234', name:'Smith'} ];

  var card = cards[0];
  // each item is an instance of CreditCard
  expect(card instanceof CreditCard).toEqual(true);
  card.name = "J. Smith";
  // non GET methods are mapped onto the instances
  card.$save();
  // POST: /user/123/card/456 {id:456, number:'1234', name:'J. Smith'}
  // server returns: {id:456, number:'1234', name: 'J. Smith'};

  // our custom method is mapped as well.
  card.$charge({amount:9.99});
  // POST: /user/123/card/456?amount=9.99&charge=true {id:456, number:'1234', name:'J. Smith'}
});

// we can create an instance as well
var newCard = new CreditCard({number:'0123'});
newCard.name = "Mike Smith";
newCard.$save();
```

Tasks

- Add counter to game
- Save top chart of players

Pet Project