

A large blue graduation cap icon serves as a background for the text. It has a wide, flat top and a tassel hanging from the right side.

JS Akademie

Základy Backendu

<https://goo.gl/FuOLtc>

An orange square logo with two thin vertical white lines, one on the left and one on the right, framing the text.

ARTIN

Dnešní program

Uděláme “opravdový” server, který posílá kontakty na frontend

Dozvíme se o

- node.js, express
- HTTP, REST a JSON

Naučíme se z frontendu volat služby backendu

- a to asynchronně



ARTIN

Postup

- trocha teorie
- instalace express serveru
- nahrazení static za express
- služba Hello World
- služba pro kontakty - na backendu
- použít backend službu z frontendu

IN PROGRESS

Node JS

Co to je?

Nemá **<script>** tagy, má **require** a **export**

```
soubor - worker.js:  
module.export = {  
  dolt: function() {  
    jduNaPivo();  
  }  
};
```

```
var worker = require('./worker.js');  
worker.dolt();
```



Express

- Server (jako static)
- Umí odpovídat na dynamické požadavky (narozdíl od static)
 - může spustit libovolný kód, který vygeneruje odpověď, případně něco udělá
 - např. pošli kontakt s id = 1, nebo “smaž” kontakt

```
app.get('/pocasi', function(req, res) {  
  var day = req.params.day;  
  if (isWeekend(day)) {  
    res.send("prsi");  
  } else {  
    res.send("krasne");  
  }  
});
```

HTTP, REST, JSON

HTTP, REST

- metody - GET, PUT, POST, DELETE, ...
- kódy odpovědí - 200 OK, 404 Not Found, 500 Server Error, ...

JSON

- aneb jak poslat objekt po síti

```
{  
  "name": "Pepa",  
  "nohy": ["levá", "pravá", "jiná"],  
  "ruce": 2  
}
```



Angular \$http service

- Posílá z angularu (frontendu) HTTP požadavky na server
- Převádí data z a do JSONu
- Až dojde odpověď ze serveru, nechá váš kód zpracovat odpověď
 - z pohledu počítače se na odpověď **čeká staletí**, takže se zpracovává **asynchronně**

```
$http.get('/pocasi?day=nedele').then(handlePocasi);
```

```
function handlePocasi(res) {  
  if (res.data === 'prsi') {  
    toHellWithIt();  
  }  
}
```

Angular \$http service

Nebo taky stručněji:

```
$http.get('/pocasi?day=nedele')  
  .then(function handlePocasi(res) {  
    if (res.data === 'prsi') {  
      toHellWithIt();  
    }  
  });
```


Otázka za mikinu



ARTIN

Postup

- **trocha teorie - done**
- instalace express serveru
- nahrazení static za express
- služba Hello World
- služba pro kontakty - na backendu
- použít backend službu z frontendu

IN PROGRESS

Instalace Express Serveru

`npm install --save express`

V package.json se objeví:

```
"dependencies": {  
  "express": "^4.13.4"  
}
```

Nahrazení static za express

Proč?

- chceme “chytřejší” server
- potřebujeme servírovat i statické soubory pro frontend
 - html, obrázky, frontendové skripty

Jak?

- založíme serverovou aplikaci “server.js”
 - tam vytvoříme server
 - a řekneme mu, kde najde soubory pro frontend
- upravíme “npm run dev” aby použil nový server

Nahrazení static za express

/server.js:

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

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

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

```
app.listen(8080);
```

/package.json změnit

```
"dev": "static public -c-1",
```

na

```
"dev": "node server.js",
```

Služba Hello World

```
do /server.js:  
app.get('/helloworld', function(req, res) {  
  res.send('Hello World!');  
});
```

restartujte server

<http://localhost:8080/helloworld>

Kdo se nudí, může si napsat službu pro počasí...

Služba pro kontakty

Co chceme:

- kontakty uložené na serveru (všichni vidí stejné kontakty)
- služba, která dodá aktuální seznam na požádání
- stejné kontakty jako jsme měli do teď

Jak:

- soubor `/api/contacts.js`
- podobně jako `contacts-service.js`
- jenom na serveru

Služba pro kontakty

/api/contacts.js

```
var contacts = [  
  {id: 1, name: 'Donald Black', company: 'Topicware', phone: '6-(880)062-6935', email:  
    'dblack0@mashable.com', note: 'Lorem ipsum'},  
  {id: 2, name: 'Frank Little', company: 'Browseblab', phone: '9-(804)406-9373', email:  
    'flittle2@tumblr.com', note: 'Lorem ipsum'}  
];  
  
module.exports = {  
  findAll: function() {  
    return contacts;  
  }  
};
```


Služba pro kontakty

do server.js přidat:

```
var contacts = require('./api/contacts');
```

a:

```
app.get('/api/contacts', function(req, res) {  
  res.json(contacts.findAll());  
});
```

a test:

<http://localhost:8080/api/contacts>

Použití na frontendu

Co chceme:

- nemít kontakty uložené v prohlížeči, ale získávat je z nové služby
- necháme si angulaří službu `contactsService`, ale
 - místo aby si pamatovala kontakty
 - bude vyptávat kontakty ze serveru
 - angulaří služba je styčný důstojník pro frontend

Jak:

- upravíme `contact-service.js` a `home.js`
- použijeme `$http` service pro spojení s backendem

contacts-service:

```
var app = angular.module('sample-app');
app.factory('contactsService', function($http) {

  return {
    create: function(contact) {
      console.warn('contactService.create not implemented!');
      return contact;
    },

    update: function(contact) {
      console.warn('contactService.update not implemented!');
      return contact;
    },

    find: function(id) {
      console.warn('contactService.find not implemented!');
      return {};
    },

    findAll: function() {
      return $http.get('/api/contacts')
        .then(function(res) {
          return res.data;
        });
    }
  };
});
```

home.js upravit:

```
controller: function($scope, contactsService)
{
  contactsService.findAll()
    .then(function(contacts) {
      $scope.contacts = contacts;
    });
}
```

Backend pro find

server.js přidat:

```
app.get('/api/contacts/:id', function(req, res) {  
  res.json(contacts.find(req.params.id));  
});
```

contacts.js přidat (na správné místo):

```
var _ = require('lodash');  
a  
find: function(id) {  
  return _.find(contacts, {id: +id});  
},
```

Co chybí?

Frontend pro find - domácí úkol ;-)

Edit - backend i frontend

Trvalé uložení kontaktů - po restartu serveru se úpravy ztratí



Dotazy? Výkřiky?



ARTIN