



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ

«Информатика и системы управления»

КАФЕДРА

«Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

По лабораторной работе №4

По курсу: «Архитектура ЭВМ»

Студент:

Ле Ни Куанг

Группа:

ИУ7и-56Б

Преподаватель:

Попов А. Ю.

Москва

2020

Цель работы

Изучить о взаимодействии между серверами, о передачи параметров скрипту, узнать о дочерних процессах. Работать с языком логического программирования prolog.

Задача 7

Задание 1

Создать сервер А. На стороне сервера хранится файл с содержимым в формате JSON. При получении запроса на `/insert/record` идёт добавление записи в файл. При получении запроса на `/select/record` идёт получение записи из файла. Каждая запись хранит информацию о машине (название и стоимость).

Создать сервер Б. На стороне сервера хранится файл с содержимым в формате JSON. Каждая запись в файле хранит информацию о складе и массиве машин, находящихся на данном складе. То есть каждая запись хранит в себе название склада (строку) и массив названий машин (массив строк). При получении запроса на `/insert/record` идёт добавление записи в файл. При получении запроса на `/select/record` идёт получение записи из файла.

Создать сервер С. Сервер выдаёт пользователю страницы с формами для ввода информации. При этом сервер взаимодействует с серверами А и Б. Реализовать для пользователя функции:

- создание нового типа машины
- получение информации о стоимости машины по её типу
- создание нового склада с находящимися в нём машинами
- получение информации о машинах на складе по названию склада

Реализовать удобный для пользователя интерфейс взаимодействия с системой (использовать поля ввода и кнопки).

Листинг 1: File index.js

```
1 const express = require('express')
2 const request = require('request')
3
4 const app = express()
5 const port = 5000
6
7 const serverA = 'http://localhost:5001/'
8 const serverB = 'http://localhost:5002/'
9
10 app.listen(port, () => {
11   console.log('http://localhost:${port}/')
12 })
13 app.get('/', (req, res) => {
14   res.sendFile(__dirname + '/static/car.html')
15 })
16
17
18 function loadBody(req, callback) {
19   let body = []
20   req.on('data', (chunk) => {
21     body.push(chunk)
22   }).on('end', () => {
23     body = Buffer.concat(body).toString()
24     callback(body)
25   })
26 }
27
28
29 app.get('/car', (req, res) => {
30   res.sendFile(__dirname + '/static/car.html')
31 })
32
33 app.get('/store', (req, res) => {
34   res.sendFile(__dirname + '/static/store.html')
35 })
36
37 app.get('/car/select', (req, res) => {
38   const url = serverA + 'select/record' +
39     req.originalUrl.substring('/car/select'.length)
40   request.get(url).pipe(res)
41 })
42
43 app.post('/car/insert', (req, res) => {
44   const url = serverA + 'insert/record'
45   loadBody(req, (body) => {
46     request.post({url: url, body: body}, (err, response, body) => {
47       res.end(body)
48     })
49   })
50 })
```

```

47     })
48   })
49 })
50
51 app.get('/store/select', (req, res) => {
52   const url = serverB + 'select/record' +
53     req.originalUrl.substring('/store/select'.length)
54   request.get(url).pipe(res)
55 })
56
57 app.post('/store/insert', (req, res) => {
58   const url = serverB + 'insert/record'
59   loadBody(req, (body) => {
60     request.post({url: url, body: body}, (err, response, body) => {
61       res.end(body)
62     })
63   })
64 })
65
66 // solve import problem css, js when use sendFile()
67 app.use('/', express.static(__dirname + '/static'))

```

Листинг 2: File a.js

```

1  const express = require('express')
2  const fs = require('fs')
3
4  const app = express()
5  const port = 5001
6
7  app.listen(port, () => {
8    console.log(`http://localhost:${port}/`)
9  })
10
11 app.use(function(req, res, next) {
12   res.header("Cache-Control", "no-cache, no-store, must-revalidate")
13   res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept")
14   res.header("Access-Control-Allow-Origin", "*")
15   next()
16 })
17
18
19 const dbfile = 'data/car.json'
20 let db = JSON.parse(fs.readFileSync(dbfile))
21
22 function loadBody(req, callback) {
23   let body = []

```

```

24     req.on('data', (chunk) => {
25         body.push(chunk)
26     }).on('end', () => {
27         body = Buffer.concat(body).toString()
28         callback(body)
29     })
30 }
31
32 app.get('/select/record', (req, res) => {
33     const key = req.query.name
34     if (key) {
35         res.end(db[key].toString())
36     } else {
37         res.sendFile(__dirname + '/' + dbfile)
38     }
39 })
40
41 app.post('/insert/record', (req, res) => {
42     loadBody(req, (body) => {
43         const entry = JSON.parse(body)
44         const key = Object.keys(entry)[0]
45         if (db[key]) {
46             res.end('{"result":␣false}')
47         } else {
48             db[key] = parseInt(entry[key])
49             fs.writeFileSync(dbfile, JSON.stringify(db, null, 2))
50             res.end('{"result":␣true}')
51         }
52     })
53 })

```

Листинг 3: File b.js

```

1  const express = require('express')
2  const fs = require('fs')
3
4  const app = express()
5  const port = 5002
6
7  app.listen(port, () => {
8      console.log('http://localhost:${port}/')
9  })
10
11 app.use(function(req, res, next) {
12     res.header("Cache-Control", "no-cache,␣no-store,␣must-revalidate")
13     res.header("Access-Control-Allow-Headers", "Origin,␣X-Requested-With,␣
14         Content-Type,␣Accept")
15     res.header("Access-Control-Allow-Origin", "*")
16     next()

```

```

16 })
17
18
19 const dbfile = 'data/store.json'
20 let db = JSON.parse(fs.readFileSync(dbfile))
21
22 function loadBody(req, callback) {
23     let body = []
24     req.on('data', (chunk) => {
25         body.push(chunk)
26     }).on('end', () => {
27         body = Buffer.concat(body).toString()
28         callback(body)
29     })
30 }
31
32 app.get('/select/record', (req, res) => {
33     const key = req.query.name
34     if (key) {
35         res.end(db[key].toString())
36     } else {
37         res.sendFile(__dirname + '/' + dbfile)
38     }
39 })
40
41 app.post('/insert/record', (req, res) => {
42     loadBody(req, (body) => {
43         const entry = JSON.parse(body)
44         const key = Object.keys(entry)[0]
45         if (db[key]) {
46             res.end('{"result":␣false}')
47         } else {
48             db[key] = entry[key]
49             fs.writeFileSync(dbfile, JSON.stringify(db, null, 2))
50             res.end('{"result":␣true}')
51         }
52     })
53 })

```

Листинг 4: File static/ajax.js

```

1 function ajaxGet(urlString, callback) {
2     let r = new XMLHttpRequest()
3     r.open('GET', urlString, true)
4     r.setRequestHeader('Content-Type', 'text/plain;charset=UTF-8')
5     r.send(null)
6     r.onload = function() {
7         callback(r.response)
8     }

```

```

9 }
10
11 function ajaxPost(urlString, bodyString, callback) {
12     let r = new XMLHttpRequest()
13     r.open('POST', urlString, true)
14     r.setRequestHeader('Content-Type', 'application/json;charset=UTF-8')
15     r.send(bodyString)
16     r.onload = function() {
17         callback(r.response)
18     }
19 }

```

Листинг 5: File static/car.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Lab 4-1</title>
6     <link rel="stylesheet" href="main.css">
7 </head>
8 <body>
9     <div class="card" style="width: 30%;>
10         <a href="/store">Go to Store</a>
11     </div>
12
13     <div id="form">
14         <table>
15             <tr>
16                 <th>Car</th>
17                 <td><input id="car" type="text" minlength=3 required></td>
18             </tr>
19             <tr>
20                 <th>Price</th>
21                 <td><input id="price" type="number" autocomplete="off"
22                     required></td>
23             </tr>
24         </table>
25
26         <button onClick="addCar()">Add Car</button>
27         <button onClick="getPrice()">Check Price</button>
28     </div>
29
30     <br>
31     <table id="TB" border="1px">
32         <tr>
33             <th>Car</th>
34             <th>Price</th>

```



```

35 </table>
36
37 <script src="ajax.js"></script>
38 <script language="javascript">
39     const table = document.getElementById('TB')
40     const tbody = table.querySelector('tbody')
41     const car = document.getElementById('car')
42     const price = document.getElementById('price')
43
44     function addRow(c, p) {
45         let tr = document.createElement('tr')
46         tr.innerHTML = '<td>${c}<td>${p}'
47         tbody.appendChild(tr)
48     }
49
50     function fetchData() {
51         ajaxGet('car/select', (listStr) => {
52             const list = JSON.parse(listStr)
53             for (let i in list) {
54                 addRow(i, list[i])
55             }
56         })
57     }
58
59     function getPrice() {
60         if (car.checkValidity()) {
61             const url = 'car/select/?name=${car.value}'
62             ajaxGet(url, (p) => {
63                 if (p)
64                     price.value = p
65             })
66         }
67     }
68
69     window.onload = () => {
70         fetchData()
71     }
72
73     function addCar() {
74         if (car.checkValidity() &&
75             price.checkValidity()) {
76             const record = {}
77             record[car.value] = price.value
78             ajaxPost('car/insert', JSON.stringify(
79                 record
80             ), (answer) => {
81                 let result = JSON.parse(answer).result
82                 if (result) {

```

```

83         addRow(car.value, price.value)
84         alert("Insert!")
85     } else {
86         alert("Already exist!")
87     }
88 })
89 }
90 }
91 </script>
92
93 </body>
94 </html>

```

Листинг 6: File static/store.html

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="UTF-8">
5     <title>Lab 4-1</title>
6     <link rel="stylesheet" href="main.css">
7 </head>
8 <body>
9     <div class="card" style="width: 30%;">
10         <a href="/car">Go to Cars</a>
11     </div>
12
13     <div id="form">
14         <table>
15             <tr>
16                 <th>Store</th>
17                 <td><input id="store" type="text" minlength=3 required></td>
18             </tr>
19             <tr>
20                 <th>Cars</th>
21                 <td><input id="cars" type="text" autocomplete="off"
22                     required></td>
23             </tr>
24         </table>
25
26         <button onClick="addStore()">Add Store</button>
27         <button onClick="getCars()">Check Cars</button>
28     </div>
29
30     <br>
31     <table id="TB" border="1px">
32         <tr>
33             <th>Store</th>
34             <th>Cars</th>

```

```

34         </tr>
35     </table>
36
37     <script src="ajax.js"></script>
38     <script language="javascript">
39         const table = document.getElementById('TB')
40         const tbody = table.querySelector('tbody')
41         const store = document.getElementById('store')
42         const cars = document.getElementById('cars')
43
44         function addRow(c, p) {
45             let tr = document.createElement('tr')
46             tr.innerHTML = '<td>${c}<td>${p}'
47             tbody.appendChild(tr)
48         }
49
50         function fetchData() {
51             ajaxGet('store/select', (listStr) => {
52                 const list = JSON.parse(listStr)
53                 for (let i in list) {
54                     addRow(i, list[i])
55                 }
56             })
57         }
58
59         function getCars() {
60             if (store.checkValidity()) {
61                 const url = 'store/select/?name=${store.value}'
62                 ajaxGet(url, (p) => {
63                     if (p)
64                         cars.value = p
65                 })
66             }
67         }
68
69         window.onload = () => {
70             fetchData()
71         }
72
73         function addStore() {
74             if (store.checkValidity() &&
75                 cars.checkValidity()) {
76                 const record = {}
77                 record[store.value] = cars.value.split(',')
78                 ajaxPost('store/insert', JSON.stringify(
79                     record
80                 ), (answer) => {
81                     let result = JSON.parse(answer).result

```

```

82         if (result) {
83             addRow(store.value, cars.value)
84             alert("Insert!")
85         } else {
86             alert("Already exist!")
87         }
88     })
89 }
90 }
91 </script>
92
93 </body>
94 </html>

```

Листинг 7: File static/main.css

```

1 body {
2     font-family: Avenir, Helvetica, Arial, sans-serif;
3     -webkit-font-smoothing: antialiased;
4     -moz-osx-font-smoothing: grayscale;
5     text-align: center;
6     color: #2c3e50;
7     padding: 8px;
8     background-color: whitesmoke;
9     text-shadow: #66666660 2px 2px 10px;
10 }
11
12 table {
13     margin: 50px auto 20px auto;
14     width: min(60%, 400px);
15 }
16
17 button {
18     padding: 5px 10px 5px 10px;
19     cursor: pointer;
20     border-radius: 5px;
21     border: none;
22     background-color: #c0f0ff;
23 }
24
25 button:hover {
26     filter: brightness(102%);
27     box-shadow: #66666660 2px 2px 10px;
28 }
29
30 input {
31     margin: 5px;
32 }
33

```

```
34 .card {
35     border-radius: 6px;
36     margin-bottom: min(20px, 2vw);
37 }
38
39 .card:hover {
40     filter: brightness(102%);
41     box-shadow: #66666660 2px 2px 12px;
42 }
43
44 a {
45     color: #00796B;
46     text-decoration: none;
47     letter-spacing: 1px;
48     word-spacing: 1px;
49 }
50
51 div {
52     margin: auto;
53 }
```

Тест 1

Go to Store

Car

Price

Add Car

Check Price

| Car | Price |
|--------------|--------|
| Elfin | 62838 |
| Mercedes-AMG | 83461 |
| Holden | 28236 |
| RUF | 40523 |
| Lada | 69116 |
| Ranz | 53934 |
| Hillman | 104858 |
| Nissan | 55166 |
| Nissan GT-R | 99163 |
| DeSoto | 48850 |

Car

Nissan

Price

55166

Add Car

Check Price

| Car | Price |
|--------------|--------|
| Elfin | 62838 |
| Mercedes-AMG | 83461 |
| Holden | 28236 |
| RUF | 40523 |
| Lada | 69116 |
| Ranz | 53934 |
| Hillman | 104858 |
| Nissan | 55166 |

Go to Cars

Store

Cars

| Store | Cars |
|-----------|--|
| Lzgwgb | Scion,Lada,Marlin,Hillman |
| Qioxde | Ranz,DeSoto,Hafei,Singer,Cole,SEAT |
| Ugklxuzpj | EDAG,Chery |
| Zlmytcfjl | Gilbern,Lloyd,Mercedes-AMG,Carlsson,Nissan GT-R |
| Qplyvshm | Peterbilt,Brilliance,SEAT,OSCA,Nissan GT-R,Gilbern |
| Mbvgnptdg | Hafei,SEAT,Opel,Bufori |
| Vepqkcu | Bufori,Panoz,Eicher,Mercedes-Benz |
| Wemzsp | Mack,Franklin,Berkeley,OSCA,Carlsson,Brabus,Mercedes-Benz |
| Nunkhfj | Bizzarrini,Cole |
| Zxuiksp | SEAT,Peterbilt |
| Bfiefwn | Nissan GT-R,Mercedes-Benz,Spyker,Škoda,Marlin,Lloyd |
| Vqdvitjff | Škoda,Saleen,Bizzarrini,Gilbern,Spyker,Pierce-Arrow,Nissan |
| Awpfa | Elfin,Opel,Nissan |
| Pwasgmye | Škoda,Peterbilt,EDAG |
| Lnhxti | Marlin,Cole,Berkeley,Lloyd,OSCA |
| Qiox | Ranz,DeSoto,Hafei,Singer,Cole,SEAT |
| New Store | Nissan,Mercedes-Benz |

Задание 2

Написать скрипт, который принимает на вход число и считает его факториал. Скрипт должен получать параметр через `process.argv`.

Написать скрипт, который принимает на вход массив чисел и выводит на экран факториал каждого числа из массива. Скрипт принимает параметры через `process.argv`.

При решении задачи вызывать скрипт вычисления факториала через `execSync`.


Листинг 8: File factorial_arr.js

```
1 const execSync = require('child_process').execSync
2
3 function useCmd(cmd) {
4     const options = { encoding: 'utf8' }
5     const answer = execSync(cmd, options)
6     return answer
7 }
8
9
10 for (let i = 2; i < process.argv.length; i++) {
11     const n = process.argv[i]
12     const fac = useCmd(`node factorial ${n}`)
13     const answer = `${n}! = ${fac}`
14     console.log(answer)
15 }
```

Листинг 9: File factorial.js

```
1 function factorial(n, res=1) {
2     if (n <= 1) return res
3     res *= n
4     return factorial(n-1, res)
5     // return n * factorial(n-1)
6 }
7
8 console.log(factorial(process.argv[2]))
```


Тест 2

```
EVM/7/2 via  v14.11.0  
→ node factorial_arr 3 5 8 6 0 1  
3! = 6  
  
5! = 120  
  
8! = 40320  
  
6! = 720  
  
0! = 1  
  
1! = 1
```

Задача 8

Задание 1

С клавиатуры считываются числа А и В. Необходимо вывести на экран все числа Фибоначчи, которые принадлежат отрезку от А до В.

Листинг 10: File 1.pl

```
1 writeEqualGreater(X, MIN) :- X >= MIN, write(X), write(' ').
2
3 fib(A, B, START, END) :- once(writeEqualGreater(A, START) ; true)
4                           , NEW is A + B
5                           , B <= END
6                           , fib(B, NEW, START, END).
7
8
9 f(START, END) :- write('fib('), write(START), write(','), write(END),
10                write('): '),
                  fib(0, 1, START, END).
```

Тест 1

```
?- f(0,50).
fib(0,50): 0 1 1 2 3 5 8 13 21 34
false.
```

```
?- f(34,34).
fib(34,34): 34
false.
```

```
?- f(3,100).
fib(3,100): 3 5 8 13 21 34 55 89
false.
```

Вывод

При выполнении лабораторной работы было изучено взаимодействие между серверами, узнал как передачи параметров скрипту, узнал немного о дочерних процессах, укрепил знания html, css, javascript. Немного узнал о языке логического программирования prolog.