



31.03.21 08:56

CSCI 165

Introduction to the Internet and the World Wide Web

Lec 4: Backend Programming II



Jetic Gū

Overview

- Focus: Web Development
- Architecture: Internet
- Core Ideas:
 1. Protocols
 2. NodeJS cont'

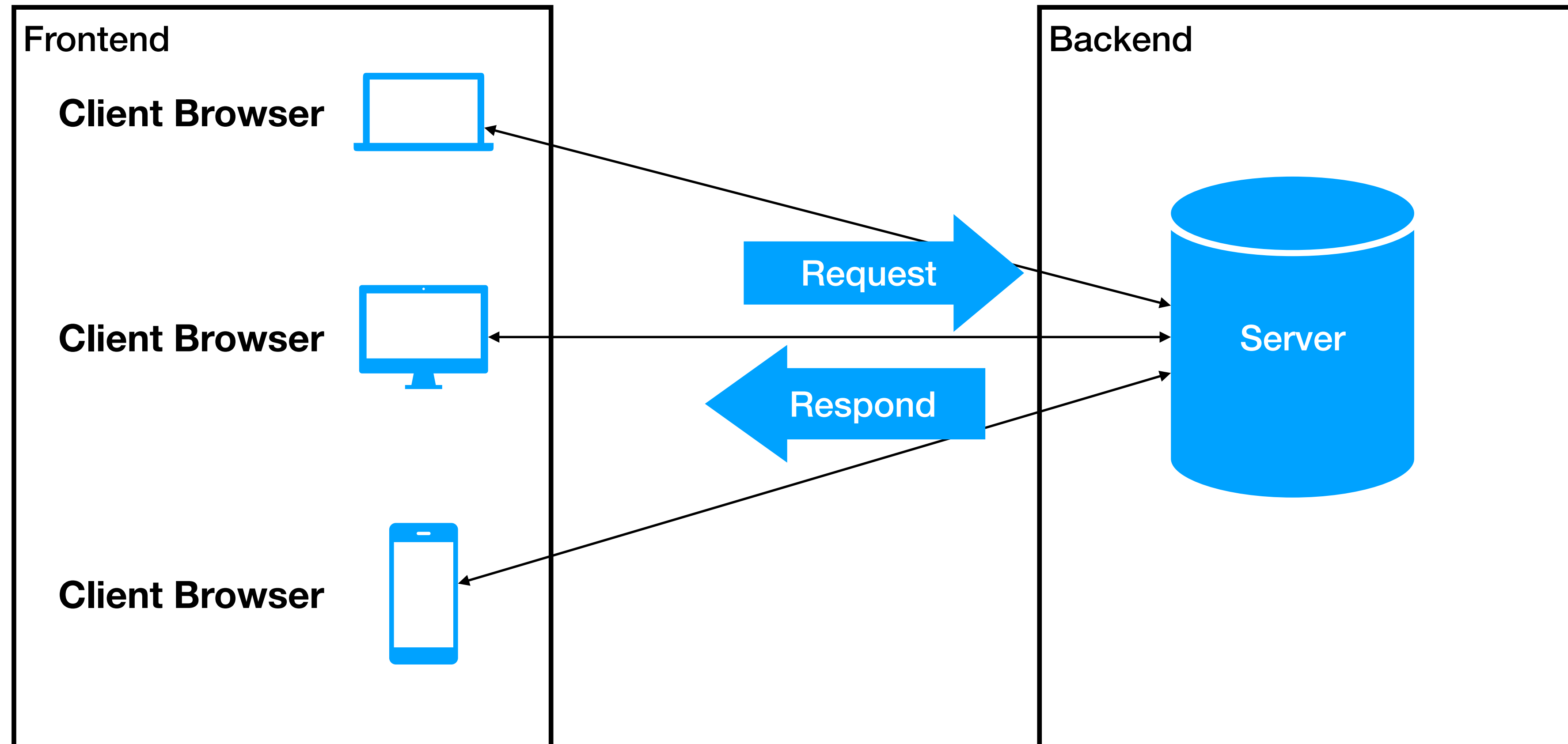
Protocols

HTTP

What we've done so far

- Frontend
 - HTML + CSS + Javascript
 - Executed on the browser
- Backend
 - NodeJS

What we've done so far



How do computers talk to each other?

- Through **protocols**
 - Established rules determine how data is transmitted between different devices in the same network.
 - Protocols are like **languages**, for two computers to talk to each other, they **must speak the same language**
 - Computer Networks have a lot of Protocols

How do computers talk to each other?

- Low-Level protocol: TCP/IP
- High-Level protocols:
 - HTTP: Hypertext (HTML) transfer protocol, default port 80
 - HTTPS: secure HTTP protocol, default port 443
 - FTP: file transfer protocol 21

So...

- I have my HTML file on my desktop, I double clicked to open it
 - This is using `file` protocol, which accesses files on your hard drive
Other computers on the network cannot do that.
 - NodeJS: creates an HTTP/HTTPS server, provide access to your HTML for all computers in the network

NodeJS

Continued

NodeJS Code

- Serving a static HTML page

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello World!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```

NodeJS Code

- create variable `http` as a new instance of NodeJS `http` type
- This is your webserver's protocol module

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello world!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```

NodeJS Code

- create variable `server` as a new instance of NodeJS `http server` type
- This is your webserver instance

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello world!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```

NodeJS Code

- Depending on the request, you can write different things
- Content-Type can be 'text/plain', 'text/html', 'text/css', etc.

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello world!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```

NodeJS Code

- Transmitting the body for the same type

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello world!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```


NodeJS Code

- Run the server on specified port

```
var http = require("http");  
var port = 8080;  
  
var server = http.createServer(function (request, response) {  
  response.writeHead(200, {'Content-Type': 'text/html'});  
  response.end('<h1>Hello world!</h1>');  
})  
  
server.listen(port, function() {  
  console.log('Server unning at http://localhost:8080');  
})
```