

NodeJS

Part 2

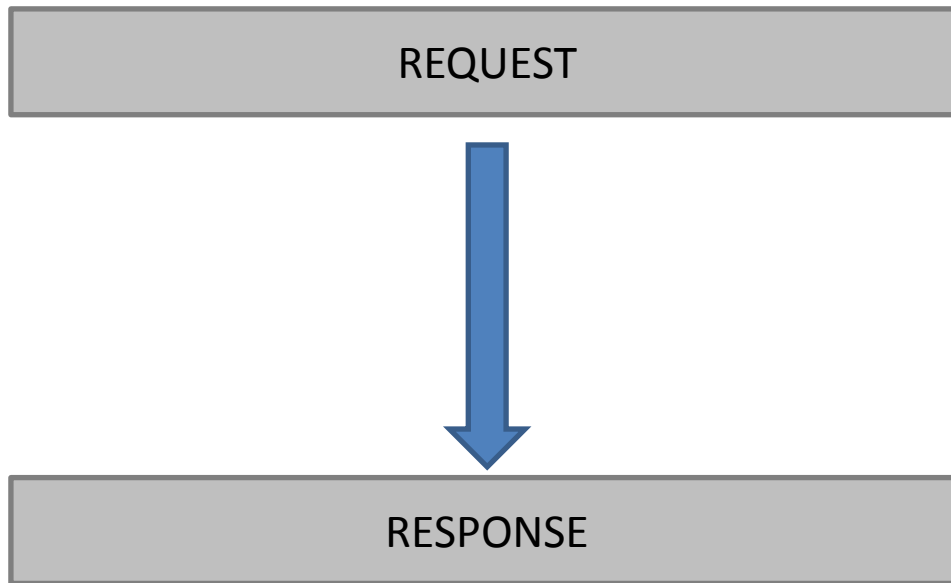
jan.schulz@devugees.org

Agenda

1. Middleware
2. Templates
3. Relational Databases: SQL

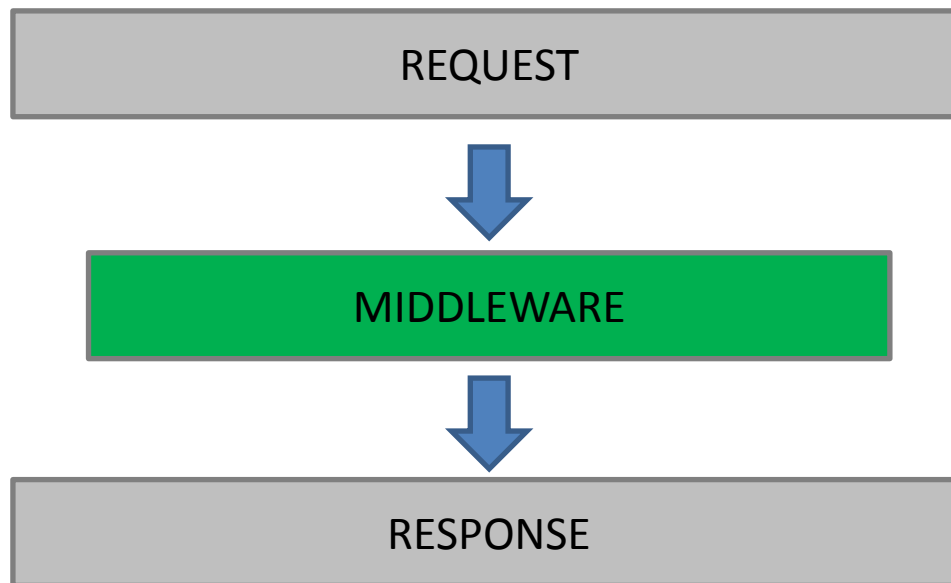
1. Middleware

- Middleware: Code that sits between two layers of software



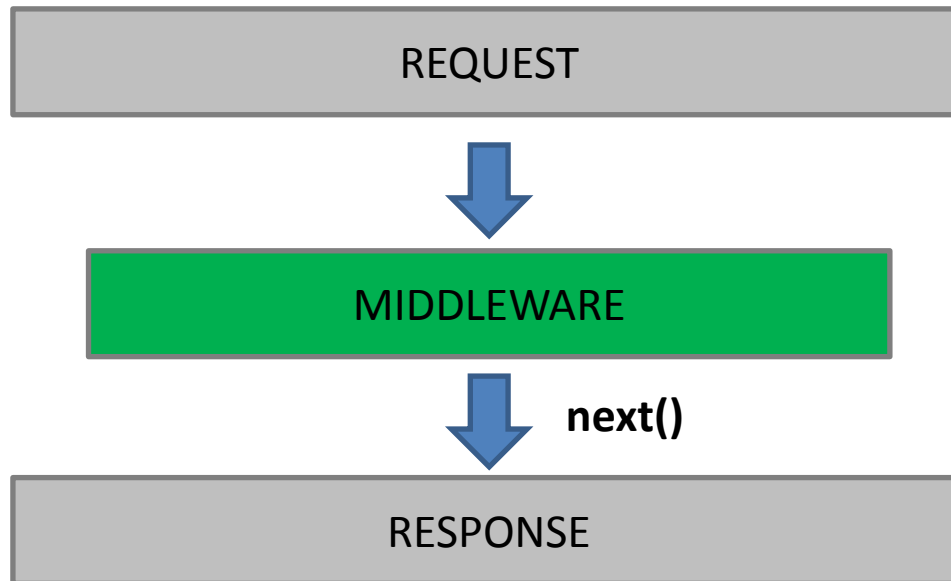
1. Middleware

- Middleware: Code that sits between two layers of software



1. Middleware

- Middleware: Code that sits between two layers of software



1. Middleware

- Middleware: creates re-usable ways of dealing with HTTP-requests

2. Templates

- A blueprint for HTML-files
- Placeholders will be replaced by JavaScript-variables when a template is rendered

```
<html>
  <head></head>
  <% if (helloworld) %>
    <h2><%=helloworld%></h2>
  <% } %>
  <body></body>
</html>
```

2. Templates

Task:

Convert your personal blog to your own NodeJS server.

- 1.) Create a static asset /public
- 2.) Implement a route GET /
- 3.) Convert your HTML file to an EJS template.
- 4.) Implement two routes in your app.js
 - GET /contacts
 - POST /contacts
- for GET /contacts
 1. Open a file contacts.json which is in the root folder of the server.
 2. If it does not exist, create it with the initial content "[]". Otherwise, read its contents and return them as JSON.
- for POST /contacts
 1. Open a file contacts.json which is in the root folder of the server.
 2. If it does not exist, create it with the initial content "[]". Otherwise, read its content and parse it as a JavaScript object (an array of objects).
 3. Read the POST body: you should receive 3 variables: name, email and text.
 4. Construct an object in which you store name, email and text.
 5. Push .4) into the array of 2.)
 6. Stringify 5.) as JSON and then save it as contacts.json. (overwrite contacts.json)
- 5.) Change your IP-address in your \$.ajax – Request in your main.js to <http://localhost:3000/contacts>.
- 6.) Test it.

3. Relational Databases - SQL

- Relational Database:

ID	Firstname	Lastname
17	John	Doe

3. Relational Databases - SQL

- Relational Database:

ID	Firstname	Lastname
17	John	Doe

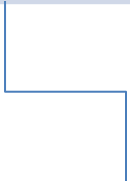


Person	Street	Number
17	Main Street	2

3. Relational Databases - SQL

- Relational Database:

ID	Firstname	Lastname
17	John	Doe



Person	Street	Number
17	Main Street	2

Task: How would this look as JavaScript Object?

3. Relational Databases - SQL

```
{  
  firstname: 'John',  
  lastname: 'Doe',  
  address: {  
    street: 'Main Street',  
    number: 2  
  }  
}
```

3. Relational Databases - SQL

Task:

Implement two routes for your personal blog NodeJS server.

GET /contact

Lists all of your contact requests

POST /contact

Creates a new contact request.

Note: Create a MySQL database and a table for this.