

</talentlabs>

Express Lecture 3

Overview of Different Response Types





</talentlabs>

Agenda

- Router & routes
- JSON Response
- Files Response
- HTML Response
- Server-side vs Client-side Rendering

Router & Routes

</talentlabs>



Starter Template

/routes/index.js

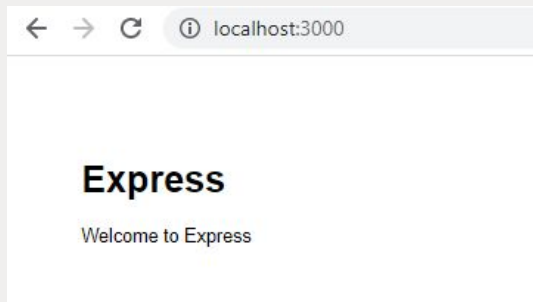
```
var express = require("express");
var router = express.Router();

/* GET home page. */
router.get("/", function (req, res, next) {
  res.render("index", { title: "Express" });
});

module.exports = router;
```

Each file in the **“/routes”** folder actually contains an **Express Router**.

Then we are adding routes to this router.



Adding more routes to the router

/routes/index.js

```
var express = require("express");
var router = express.Router();

router.get("/", function (req, res, next) {
  res.render("index", { title: "Express" });
});

router.get("/1", function (req, res, next) {
  res.render("index", { title: "Express" });
});

router.get("/another", function (req, res,
next) {
  res.render("index", { title: "Express" });
});

module.exports = router;
```

Each file in the **“/routes”** folder actually contains an **Express Router**.

Then we are adding routes to this router.
3 Routes doing the same things
GET “/”
GET “/1”
GET “/another”

Adding more routes to the router

/routes/index.js

```
var express = require("express");
var router = express.Router();

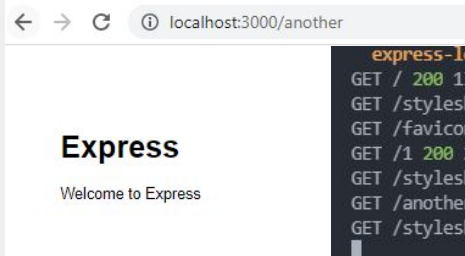
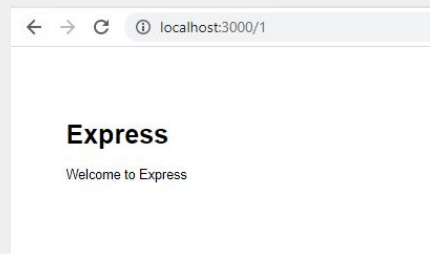
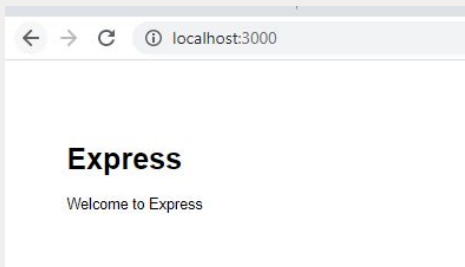
router.get("/", function (req, res, next) {
  res.render("index", { title: "Express" });
});

router.get("/1", function (req, res, next) {
  res.render("index", { title: "Express" });
});

router.get("/another", function (req, res,
next) {
  res.render("index", { title: "Express" });
});

module.exports = router;
```

Each file in the **“/routes”** folder actually contains an **Express Router**.



```
express-lecture-3-demo:server Listening on port 3000 +0ms
GET / 200 13.803 ms - 207
GET /stylesheets/style.css 200 3.482 ms - 111
GET /favicon.ico 404 2.079 ms - 1243
GET /1 200 2.205 ms - 207
GET /stylesheets/style.css 304 1.143 ms - -
GET /another 200 1.667 ms - 207
GET /stylesheets/style.css 304 0.682 ms - -
```

Duplicate routes

/routes/index.js

```
var express = require("express");
var router = express.Router();

router.get("/", function (req, res, next) {
  res.render("index", { title: "Express" });
});

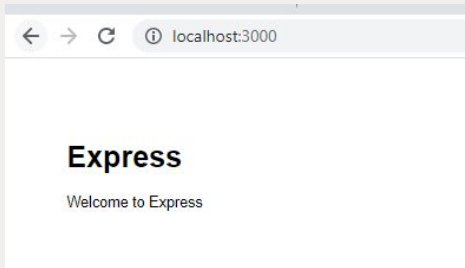
....

router.get("/", function (req, res, next) {
  res.render("index", { title: "HI!!!" });
});

module.exports = router;
```

Each file in the **“/routes”** folder actually contains an **Express Router**.

If the **HTTP method** and the **HTTP Path** of a route of 2 routes are the same, the **first one** will be used.



JSON Response

</talentlabs>



JSON

- JSON response is the most common kind of response for an API server.
- JSON is not a HTML page. JSON is a data structure containing the data the user requested.
- The requester usually is a Frontend Application.

JSON Response

```
router.get("/json1", function (req, res,
next) {
  res.json({
    name: "Peter",
    gender: "Male",
  });
});
```

```
router.get("/json2", function (req, res,
next) {
  res.json({
    name: "Amy",
    gender: "Female",
  });
});
```



More Complicated JSON Response

GET

https://api.thecatapi.com/v1/images/search?breed_id=beng
ng

```
api.thecatapi.com/v1/images/search?breed_id=beng

{
  "breeds": [
    {
      "weight": {
        "imperial": "6 - 12",
        "metric": "3 - 7"
      },
      "id": "beng",
      "name": "Bengal",
      "cfa_url": "http://cfa.org/Breeds/BreedsAB/Bengal.aspx",
      "vetstreet_url": "http://www.vetstreet.com/cats/bengal",
      "vcahospitals_url": "https://vcahospitals.com/know-your-pet/cat-breeds/bengal",
      "temperament": "Alert, Agile, Energetic, Demanding, Intelligent",
      "origin": "United States",
      "country_codes": "US",
      "country_code": "US",
      "description": "Bengals are a lot of fun to live with, but they're definitely no",
      "life_span": "12 - 15",
      "indoor": 0,
      "lap": 0,
      "adaptability": 5,
      "affection_level": 5,
      "child_friendly": 4,
      "cat_friendly": 4,
      "dog_friendly": 5,
      "energy_level": 5,
      "grooming": 1,
      "health_issues": 3,
      "intelligence": 5,
      "shedding_level": 3,
      "social_needs": 5,
      "stranger_friendly": 3,
      "vocalisation": 5,
      "bidability": 3,
      "experimental": 0,
      "hairless": 0,
      "natural": 0,
      "rare": 0,
      "rex": 0,
      "suppressed_tail": 0,
      "short_legs": 0,
      "wikipedia_url": "https://en.wikipedia.org/wiki/Bengal_(cat)",
      "hypoallergenic": 1,
      "reference_image_id": "03btzlls0"
    },
    {
      "id": "4-55zDNIL",
      "url": "https://cdn2.thecatapi.com/images/4-55zDNIL.jpg",
      "width": 880,
      "height": 1100
    }
  ]
}
```

JSON Response Summary

- JSON is not a HTML page. JSON is a data structure containing the data the user requested.
- It is about the data, not about the UI.
- The **frontend applications can have different UI designs to display the data.**
 - Mobile application...
 - Website application...

Files Response

</talentlabs>



Static Files

In Express, we will organize files into a static resource folder.
You can find this configuration in the /app.js file.

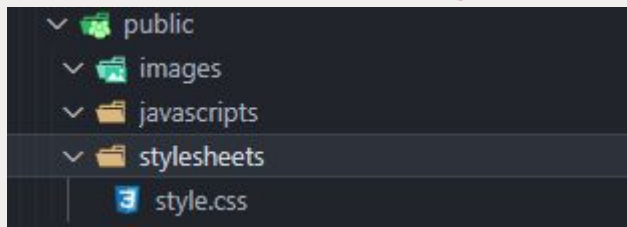
```
app.use(express.static(path.join(__dirname, 'public')));
```

Static Files

In Express, we will organize files into a static resource folder. You can find this configuration in the `/app.js` file.

```
app.use(express.static(path.join(__dirname, 'public')));
```

We don't need to define routes for them, it is handled by Express internally.

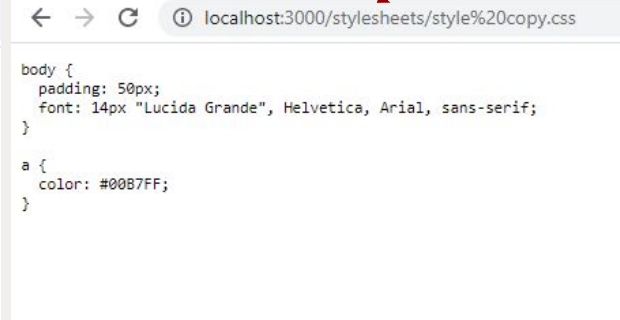
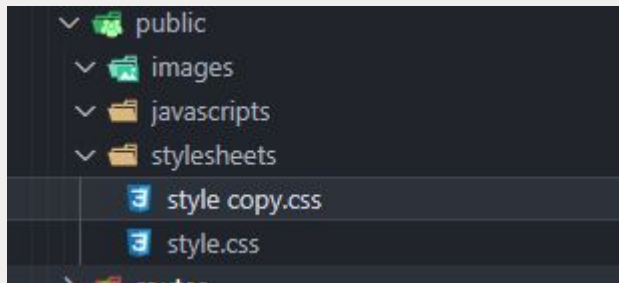


Static Files

In Express, we will organize files into a static resource folder.
You can find this configuration in the /app.js file.

```
app.use(express.static(path.join(__dirname, 'public')));
```

We don't need to define routes for them, it is handled by Express internally.



HTML Response

</talentlabs>



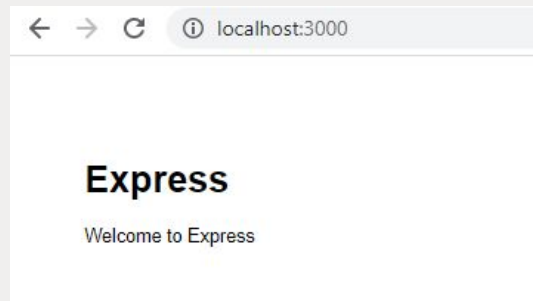
HTML Templates

```
router.get("/", function (req, res, next) {  
  res.render("index",  
    { title: "Express" });  
});
```

Template name,
index -> **/views/index.ejs**

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title><%= title %></title>  
    <link rel='stylesheet'  
href='/stylesheets/style.css' />  
  </head>  
  <body>  
    <h1><%= title %></h1>  
    <p>Welcome to <%= title %></p>  
  </body>  
</html>
```

Rendering Context,
We will replace the “title”
placeholder in the template with
the value “Express”



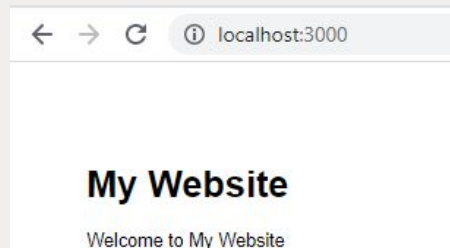
HTML Templates

```
router.get("/", function (req, res, next) {  
  res.render("index",  
    { title: "My Website" });  
});
```

Template name,
index -> **/views/index.ejs**

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title><%= title %></title>  
    <link rel='stylesheet'  
href='/stylesheets/style.css' />  
  </head>  
  <body>  
    <h1><%= title %></h1>  
    <p>Welcome to <%= title %></p>  
  </body>  
</html>
```

Rendering Context,
We will replace the “title”
placeholder in the template with
the value “Express”



Server-side vs Client-side Rendering

</talentlabs>



Rendering (Rendering the UI)



Client-side Rendering

In the JSON Response section, we talked about a JSON Response is only returning the data. **We have some other frontend code to handle the rendering.**

Server-side Rendering

In the HTML Template section, you can also **define the UI with Express.**

Rendering (Rendering the UI)



Server-side rendering is easier, you only need backend developers to build a web application.

However, Client-side rendering is the trend nowadays. Because the clients are not just browser (reading HTML files). Many clients don't understand HTML file such as Mobile Apps.

```
/* GET home page. */  
router.get('/', function(req, res, next) {  
  res.render('index', {  
    title: 'Express App',  
    message: "I am here!!!!!!",  
  });  
});
```

```
ReactDOM.render(  
  <React.StrictMode>  
    <App />  
  </React.StrictMode>,  
  document.getElementById("root")  
);
```

Server-side Rendering Overview

We will talk about this in the coming lectures

