Express part 03

Lesson Plan

PW SKILLS

Topics

- What is the path param in restful API?
- How does express read the path param in the form of the requested request?
- What are query params in restful API?
- How does express read the query param in the form of the same type?
- What if the body param in restful URL ?

What is the path param in restful API ?

In web development, when we make requests to a server, we often need to provide additional information to tell the server what we're looking for or what we want to do. One way to do this is by using parameters in the request, and one specific type of parameter is called a **"path parameter."**

Attached to a URL:
Request parameters are extra details in the URL that enhance communication with the server.

Points to a Specific Resource:
Path parameters are valuable for referencing specific resources, such as a student in a database.

Separated by a Slash (/):
Path parameters in the URL are set apart by slashes, aiding organization and clarity.

Used with Various HTTP Verbs (GET, POST, PUT, DELETE):
Path parameters work seamlessly with different HTTP verbs to perform actions like retrieving, adding, updating, or deleting specific resources.

Example:

```javascript
HTTP://localhost:8081/app/v1/student/{id}
```

In this example:

- **HTTP://localhost:8081** is the base URL of the server.
- /app/v1/student/ is the path to the resource (students in this case).
- **{id}** is the path parameter representing a specific student's ID.
- For example, if we want details about the student with ID 123, we would replace {id} with 123.

So, using path parameters helps us communicate with the server more effectively, especially when dealing with specific resources like individual students in this case.

Lets see on code example :

How does express read the path param in the form of the requested request?

Request Object and req.params:

- In Express, when a client makes a request, the req object represents the incoming HTTP request.
- The req.params field is an object that holds all the path parameters extracted from the URL.

Example URL:

- Consider the example URL: HTTP://localhost:8080/app/v1/classes/{classid}/students/{id}
- This URL contains path parameters (classid and id) enclosed in curly braces {}.

Using req.params:

- To extract values from path parameters, Express provides req.params.
- In the example URL, you can retrieve the values for classid and id using req.params.classid and req.params.id, respectively.

Example Code:

```javascript
app.get('/app/v1/classes/:classid/students/:id', (req, res) => {
  const classid = req.params.classid;
  const studentid = req.params.id;
  // Use classid and studentid in your logic
});
```

- In this example, req.params is used to access the values of the classid and id path parameters.

How to define the Rest URL with path param in Express

To represent the path parameter, we prefix any path parameter with ":"

Rest URL : **http://localhost:8080/app/v1/students/{id}**
Express way to represent it : **/app/v1/students/:id**

Rest URL : **http://localhost:8080/app/v1/classes/{classId}/students/{id}**
Express way to represent it : **/app/v1/classes/:classes/students/:id**

What is query params in restful api?

Query Criterion Regarding Resources:

- Query parameters are used to specify criteria for filtering or customizing data related to resources in a RESTful API.
- They allow clients to request a specific subset of data based on certain conditions.

Application at the End of the Resource Path:

- Query parameters are added at the end of the resource path in a URL.
- They follow the base URL and resource path and are separated from them by a question mark (?).

Two Parts: ? is Added at the End:

- Query parameters consist of two parts. The first part is the question mark (?), signifying the beginning of the query parameters.

Query String in Key-Value Format:

- The second part of query parameters is the query string, formatted as key-value pairs.
- Multiple key-value pairs are separated by an ampersand (&), with each pair having a parameter key and its corresponding value, separated by an equal sign (=).

Reserved Keywords: & and =:

- Reserved keywords, such as ampersand (&) and equal sign (=), play specific roles in constructing the query string.
- Ampersand is used to separate multiple key-value pairs, while the equal sign assigns values to keys.

Example URL:

Consider the following example URL: **http://localhost:8080/app/v1/students?class=5&school=dps**

```javascript
app.get('/app/v1/students', (req, res) => {
  // Accessing query parameters using req.query
  const studentClass = req.query.class;
  const school = req.query.school;
//  your logic
});
```

When a client makes a request like http://localhost:8080/app/v1/students?class=5&school=dps, the server extracts the values of class and school from the query parameters and uses them in the response logic.

How does express read multiple query params of the same type

For example:
Rest url : **http://localhost:8080/app/v1/students?class=5&class=6**

Express will store multiple class values in the rorm of an array classes = req.body.class (this will ve an array of students)
Classes -> [“5”,”6”]

Code example :

```javascript
app.get('/app/v1/students', (req, res) => {
  // Accessing query parameters using req.query
  const studentClass = req.query.class;
consol.log(studentClass) // ["5", "6"]
//  your logic
});
```

What if the body param in restful URL ?

In RESTful APIs, "body parameters" typically refer to the data that is sent in the body of an HTTP request. These parameters are used to send complex data, such as JSON or XML payloads, when making requests to create or update resources on the server. Body parameters are commonly used with the POST, PUT, and PATCH HTTP methods.

Data in Request Body:
Body parameters consist of data sent in the body of an HTTP request, typically in the form of JSON or XML.

Used for Creating or Updating Resources:
Body parameters are commonly used with POST requests to create new resources, PUT requests to update existing resources, and PATCH requests to partially update resources.

Content-Type Header:
The Content-Type header in the HTTP request specifies the type of data in the body. Common values include application/json for JSON data and application/xml for XML data.

Expressing Complex Data:
Body parameters are useful for sending complex data structures, allowing clients to express more than simple key-value pairs typically found in query parameters.

Example with Express.js:
In an Express.js route, body parameters are often accessed using middleware like body-parser or the built-in express.json() middleware.

Example **http://localhost:8080/app/vi/students**

Request body:

```JavaScript
{
  "id": 1434234,
   "name":"Vishwa",
   "age":99
}
```

Code example

```JavaScript
// Middleware to parse JSON in the request body
app.use(bodyParser.json());

// Route to handle a POST request with body parameters
app.post('/api/v1/resources', (req, res) => {
  const bodyParams = req.body;
  // Process and use the body parameters in your logic
});
```