

EdX and its Members use cookies and other tracking technologies for performance, analytics, and marketing purposes. By using this website, you accept this use. Learn more about these technologies in the [Privacy Policy](#).



[Course](#) > [Expres...](#) > [URL Pa...](#) > [URL pa...](#)

## URL parameters

### Video: Accessing URL Parameters

The video player displays a code editor with the following JavaScript code for an Express.js application:

```
12
13 app.get('/profile', (req, res) => {
14   res.send(profile)
15 })
16
17 app.post('/profile', (req, res) => {
18   profile.push(req.body)
19   console.log('created', profile)
20   res.sendStatus(201)
21 })
22
23 app.put('/profile/:id', (req, res) => {
24   Object.assign(profile[req.params.id], req.body)
25   console.log('updated', profile[req.params.id])
26   res.sendStatus(204)
27 })
28
29 app.delete('/profile/:id', (req, res) => {
30   profile[req.params.id]
31   console.log('deleted', profile)
32   res.sendStatus(204)
33 })
34
35 app.listen(3000)
36
```

The file explorer on the right shows the project structure:

- Lesson\_06
  - 1-URL parameters.md
  - 2-Query string data.md
  - 3-Input validation.md
- Lesson\_07
- Module02\_Assessment
- .DS\_Store
- Module\_03
- Module\_04
- Module\_XX
- Project\_Files
  - Module\_01
  - Module\_02
    - Lab\_01
    - Lab\_02
  - Lesson\_03
  - Lesson\_04
  - Lesson\_05
  - Lesson\_06
    - node\_modules
    - package-lock.json
    - package.json
    - server.js
    - test.sh
  - .DS\_Store
- Module\_03
- Module\_04
- .DS\_Store
- README.txt
- Verification\_Files
  - .DS\_Store
  - README.txt
- Video\_CC
  - .DS\_Store
- .gitignore

The video player controls at the bottom show a play button, a progress bar at 0:00 / 0:00, a speed control set to 1.50x, and icons for volume, full screen, and subtitles.

## Video

[Download video file](#)

## Transcripts

[Download SubRip \(.srt\) file](#)

[Download Text \(.txt\) file](#)

## Accessing URL Parameters

To access URL parameters such as for IDs, names or other information, define the parameter in the URL pattern string with a colon `:` and then access the parameter with `req.params`.

For example, for requests like this one which has an user ID after the string `/users/`:

```
GET /users/572611d856b11dcec61651bb
```

Use a URI segment parameterized by prefixing it with a semi-colon:

```
app.get('/users/:id', (request, response) => {  
  const userId = request.params.id  
  fetchUser(userId, (error, user) => {  
    if (error) return response.status(500).send(error)  
    response.send(user)  
  })  
})
```

## Multiple URL Parameters

Express supports multiple URL parameters in a route. Simply define all of them in the URL pattern. For example for the request `GET /users/:id/transactions/:transactionId/:filter`, the route will look like:

```
app.get('/users/:id/transactions/:transactionId/:filter', (req, res)  
  const usersId = request.params.id,  
    transactionId = request.params.transactionId,  
    filter = request.params.filter  
  res.status(200).send()  
})
```