
HTML에서 웹앱까지

11주차_02

한 동 대 학 교
김군오 교수

학습 목표:

RESTful API 알아보기

- RESTful API 소개
- Express.js 로 RESTful API 만들기

RESTful API 란

- REST란?
 - Representational State Transfer 의 약자
 - api 에 대한 상세 스펙을 보지 않아도 HTTP 요청의 여러 정보를 통해 API가 하는 일을 알 수 있도록 설계된 api
- REST API가 표현하는 것
 - 자원과 자원에 대한 행위
 - 자원 : 서버에 저장되어 있는 데이터들
 - 쇼핑몰 서비스에서의 자원 : 상품 목록, 주문 목록, 회원 목록 등
 - 도서관리 서비스에서의 자원 : 도서 자체
- REST API 를 통해 사용하려는 자원을 HTTP uri 로 명시
- HTTP 메서드로 자원에 대한 행위를 명시

CRUD

- **Create** : 생성(POST)
- **Read** : 조회(GET)
- **Update** : 수정(PUT)
- **Delete** : 삭제(DELETE)

REST API 작성 규칙

- 자원은 URI 에 영문 소문자, 명사 복수형으로 표현
- 파일 확장자는 URI 에 포함하지 않음
- 자원의 계층구조는 / (슬래쉬) 로 표현

REST API 예시

CRUD	HTTP 메서드	URI
책들의 목록을 표시	GET	/books
책 한권의 정보를 표시	GET	/books/{id}
새 책을 생성	POST	/books
책을 수정	PUT	/books/{id}
책을 삭제	DELETE	/books/{id}

RESTful API를 사용하는 이유

- HTTP 프로토콜 상에서 동작
- 특정 언어나 플랫폼에 종속되지 않음
- API 의 의도를 API 사용자가 쉽게 파악할 수 있음

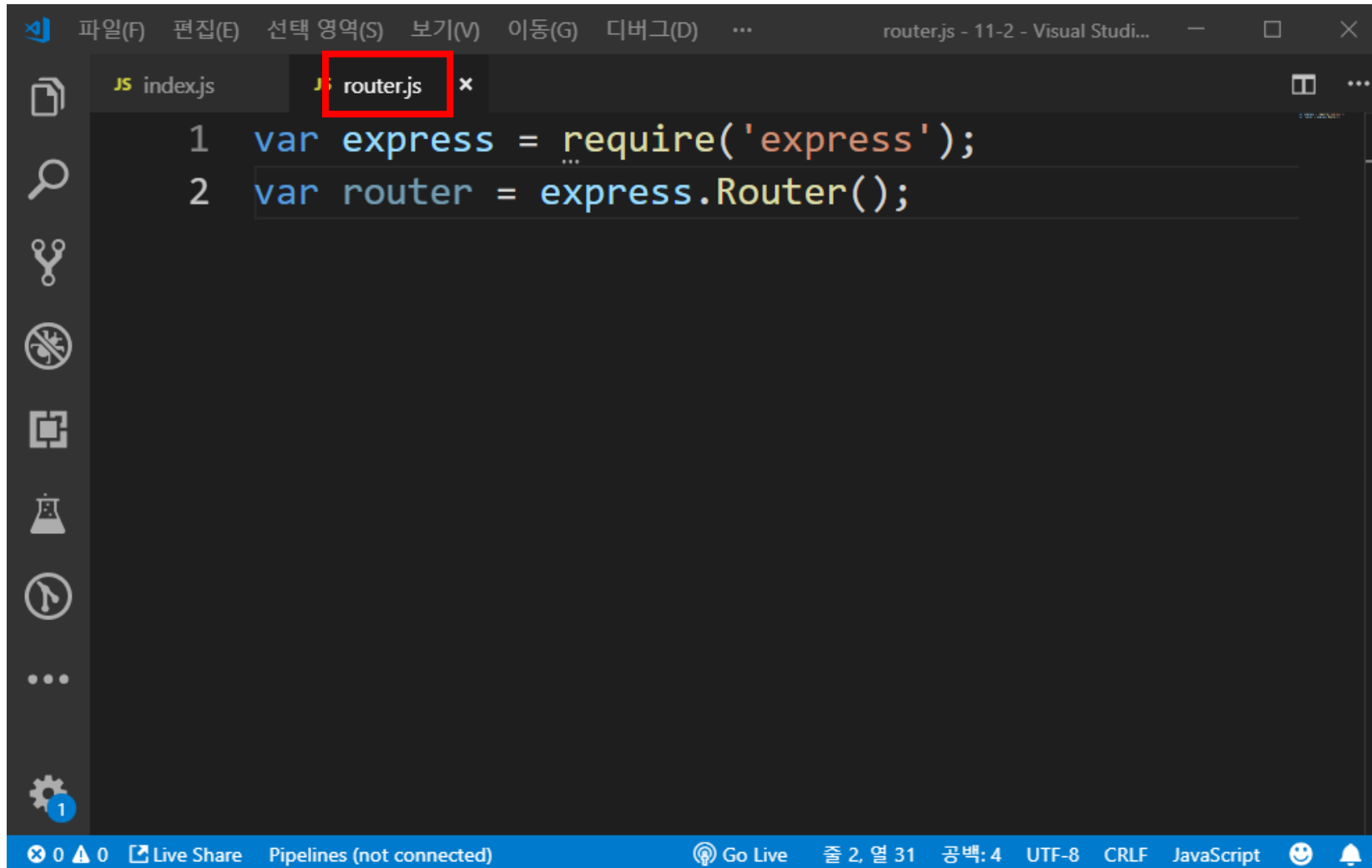
ExpressJS 로 REST API 서버 만들기

- Express 로 HTTP 메서드 처리
- Express 로 URI 처리
- Express 로 HTTP 바디 처리

Express Router

- 미들웨어의 한 종류
- HTTP 메서드, uri 에 따라 미들웨어 실행 여부 결정
- Router 생성
 - `express.Router()` 로 Express Router 생성
- 생성한 라우터는 다른 미들웨어와 마찬가지로 `app.use` 를 통해 등록할 수 있음
- Express router 를 사용하면 Express 앱의 모듈화가 가능

Express Router 사용하기

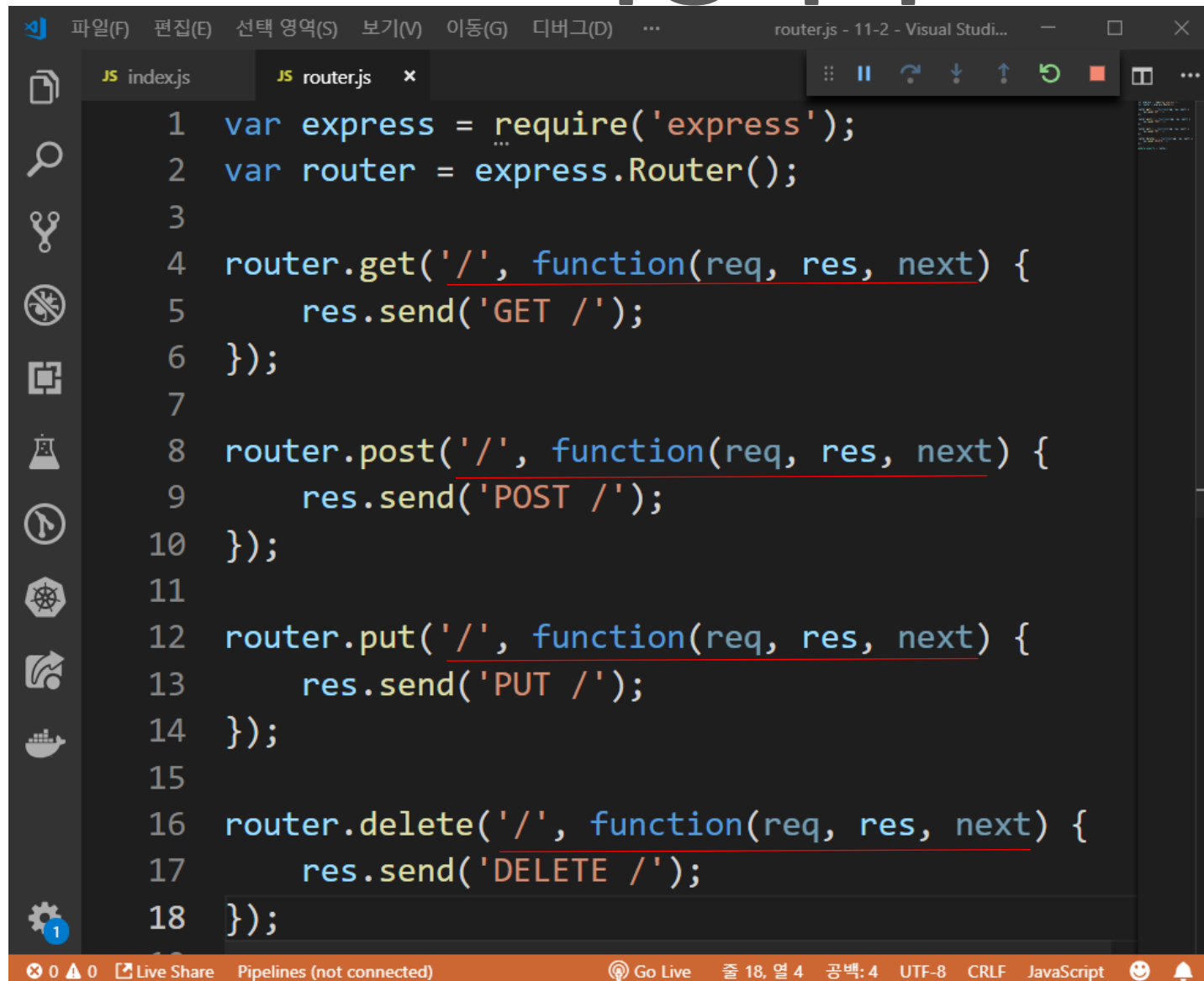


The screenshot shows the Visual Studio Code editor interface. The top menu bar includes options like '파일(F)', '편집(E)', '선택 영역(S)', '보기(V)', '이동(G)', and '디버그(D)'. The title bar indicates the file is 'router.js - 11-2 - Visual Studi...'. The file explorer on the left shows two files: 'index.js' and 'router.js', with 'router.js' highlighted by a red rectangle. The editor window displays the following JavaScript code:

```
1 var express = require('express');  
2 var router = express.Router();
```

The status bar at the bottom shows '0 0 0', 'Live Share', 'Pipelines (not connected)', 'Go Live', '줄 2, 열 31', '공백: 4', 'UTF-8', 'CRLF', 'JavaScript', and a notification bell icon.

Express Router 사용하기

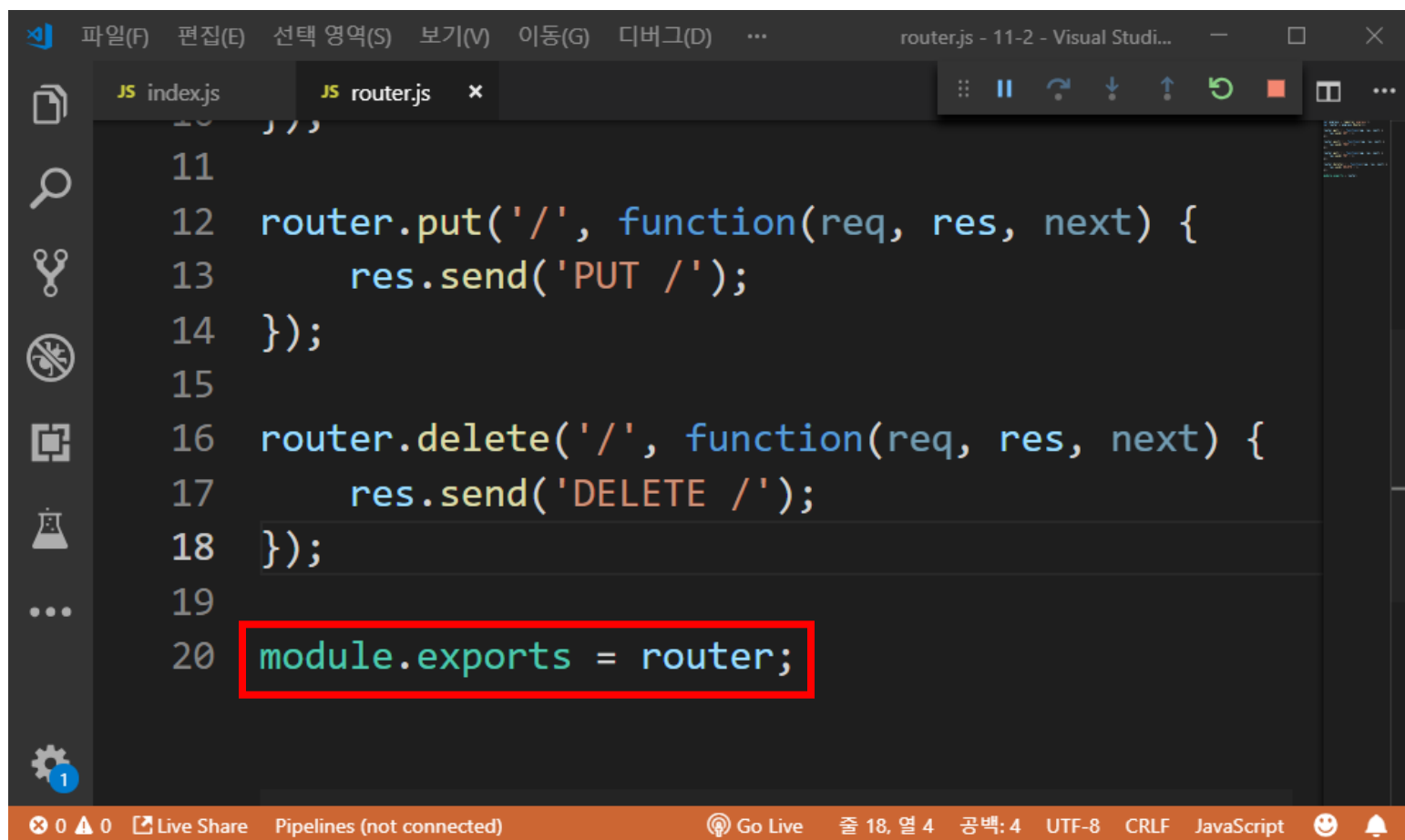


The image shows a screenshot of the Visual Studio Code editor interface. The title bar at the top reads "router.js - 11-2 - Visual Studi...". The editor has two tabs open: "JS index.js" and "JS router.js". The "router.js" tab is active, displaying the following JavaScript code:

```
1 var express = require('express');
2 var router = express.Router();
3
4 router.get('/', function(req, res, next) {
5     res.send('GET /');
6 });
7
8 router.post('/', function(req, res, next) {
9     res.send('POST /');
10 });
11
12 router.put('/', function(req, res, next) {
13     res.send('PUT /');
14 });
15
16 router.delete('/', function(req, res, next) {
17     res.send('DELETE /');
18 });
```

The left sidebar contains the Explorer, Search, Source Control, Run and Debug, Extensions, Testing, and Docker views. The bottom status bar shows "0 0 Live Share Pipelines (not connected)", "Go Live", "줄 18, 열 4", "공백: 4", "UTF-8", "CRLF", "JavaScript", and a smiley face icon.

Express Router 사용하기

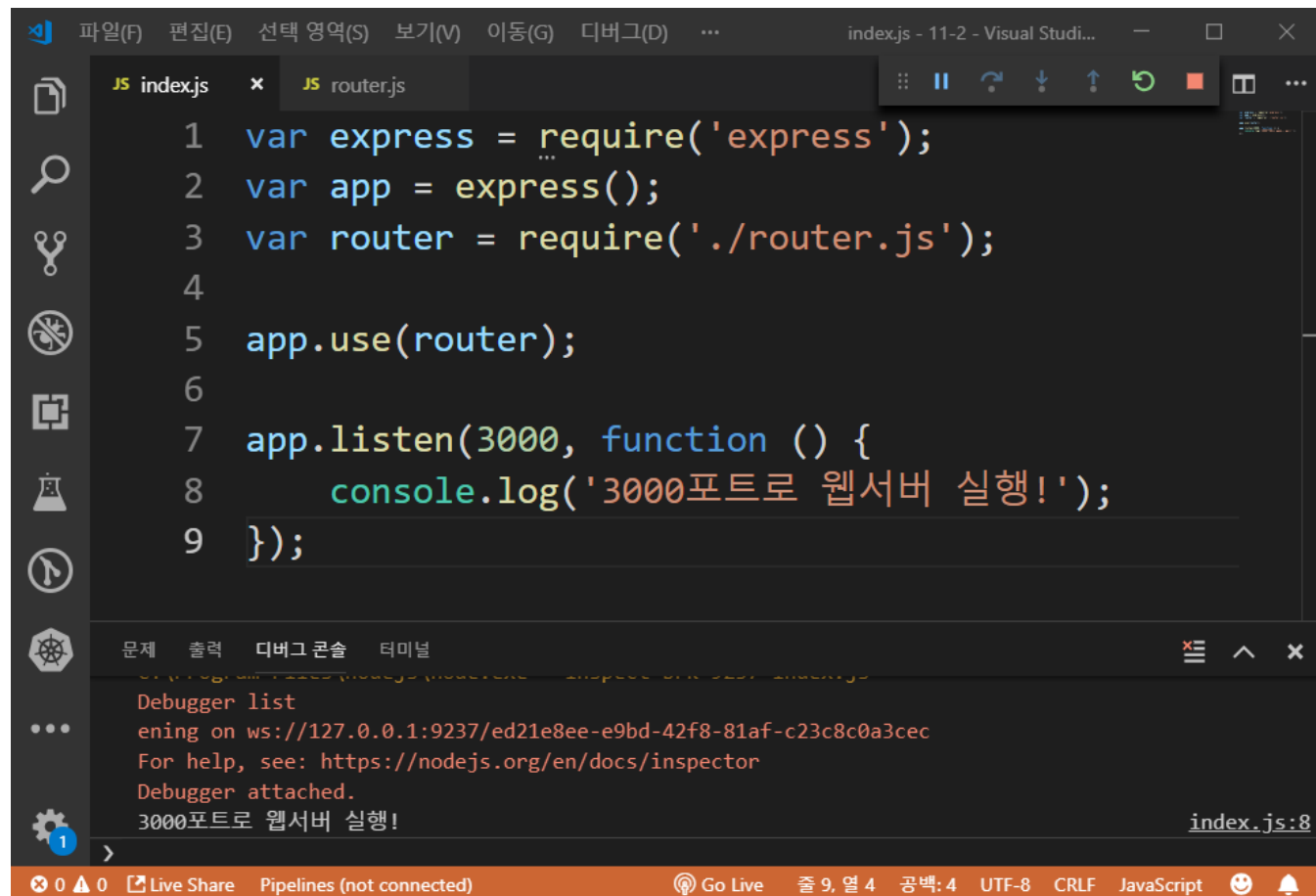


The image shows a screenshot of the Visual Studio Code editor interface. The title bar at the top indicates the file is 'router.js - 11-2 - Visual Studi...'. The editor has two tabs open: 'index.js' and 'router.js'. The 'router.js' tab is active, displaying the following JavaScript code:

```
11
12 router.put('/', function(req, res, next) {
13     res.send('PUT /');
14 });
15
16 router.delete('/', function(req, res, next) {
17     res.send('DELETE /');
18 });
19
20 module.exports = router;
```

The line `module.exports = router;` on line 20 is highlighted with a red rectangular box. The status bar at the bottom shows various icons and information, including 'Live Share', 'Pipelines (not connected)', 'Go Live', '줄 18, 열 4', '공백: 4', 'UTF-8', 'CRLF', 'JavaScript', and a smiley face icon.

Express Router 사용하기



The image shows a Visual Studio Code editor window with two files open: `index.js` and `router.js`. The `index.js` file contains the following code:

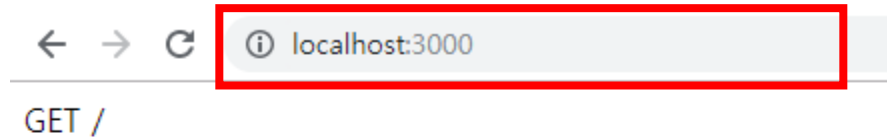
```
1 var express = require('express');
2 var app = express();
3 var router = require('./router.js');
4
5 app.use(router);
6
7 app.listen(3000, function () {
8     console.log('3000포트로 웹서버 실행!');
9 });
```

The `router.js` file is also open but its content is not visible. The terminal at the bottom shows the following output:

```
Debugger list
ening on ws://127.0.0.1:9237/ed21e8ee-e9bd-42f8-81af-c23c8c0a3cec
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.
3000포트로 웹서버 실행! index.js:8
```

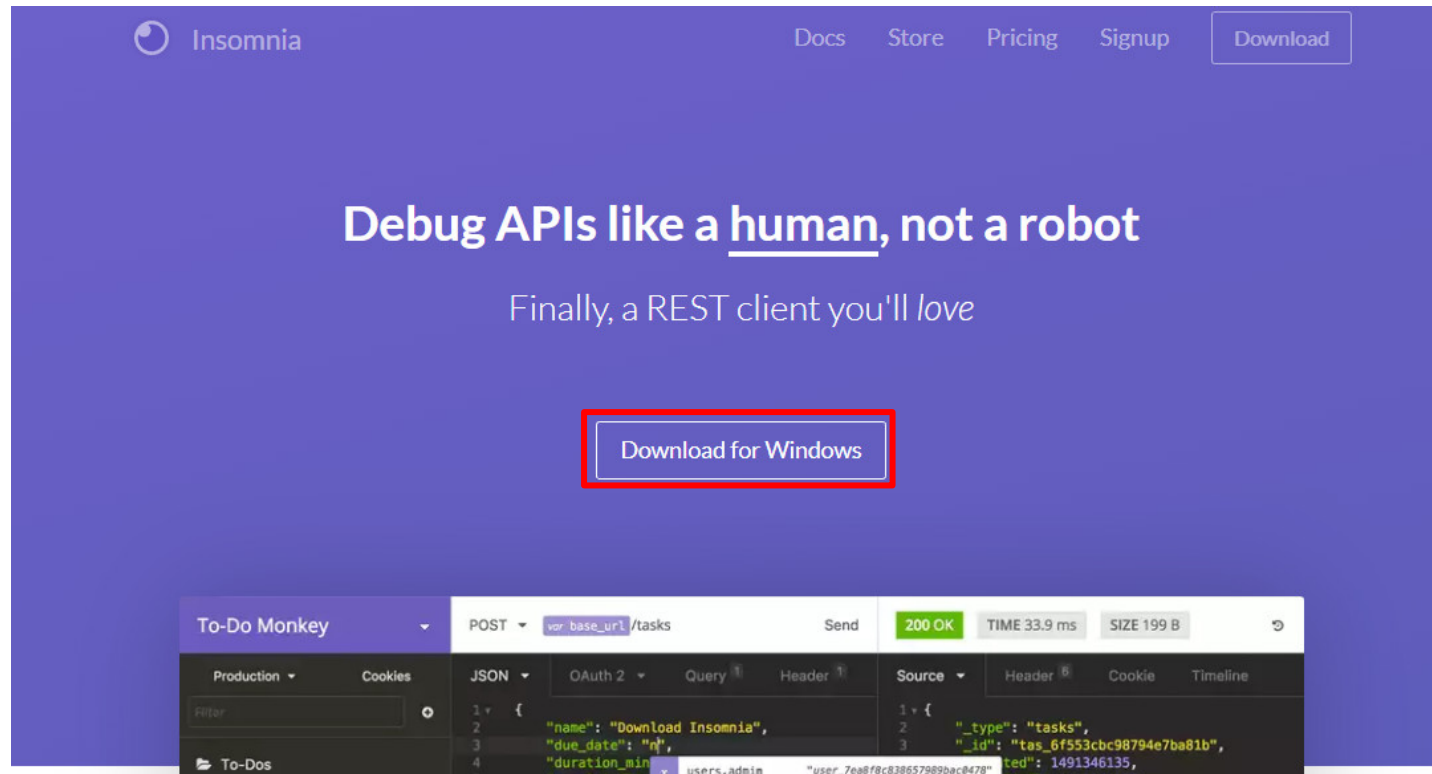
The status bar at the bottom indicates the file is `index.js:8`, the encoding is `UTF-8`, the line ending is `CRLF`, and the language is `JavaScript`.

Express Router 사용하기



Insomnia 설치하기

- <https://insomnia.rest/>



Insomnia 설치하기

Download Insomnia

So you can finally GET some REST 😴



📄 OSX 10.9+

OR `brew cask install insomnia`



📄 Windows 7+

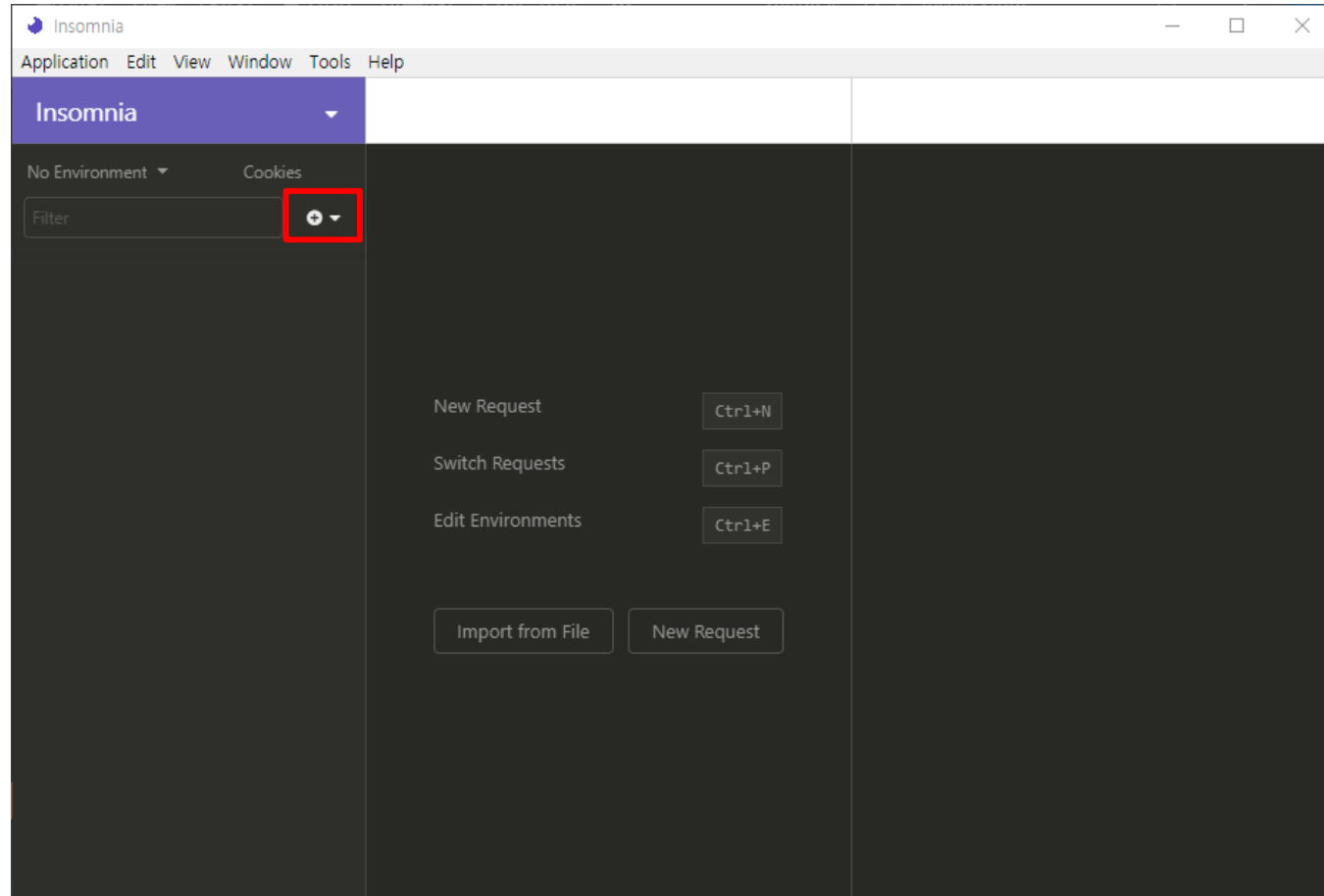
(64-bit only)



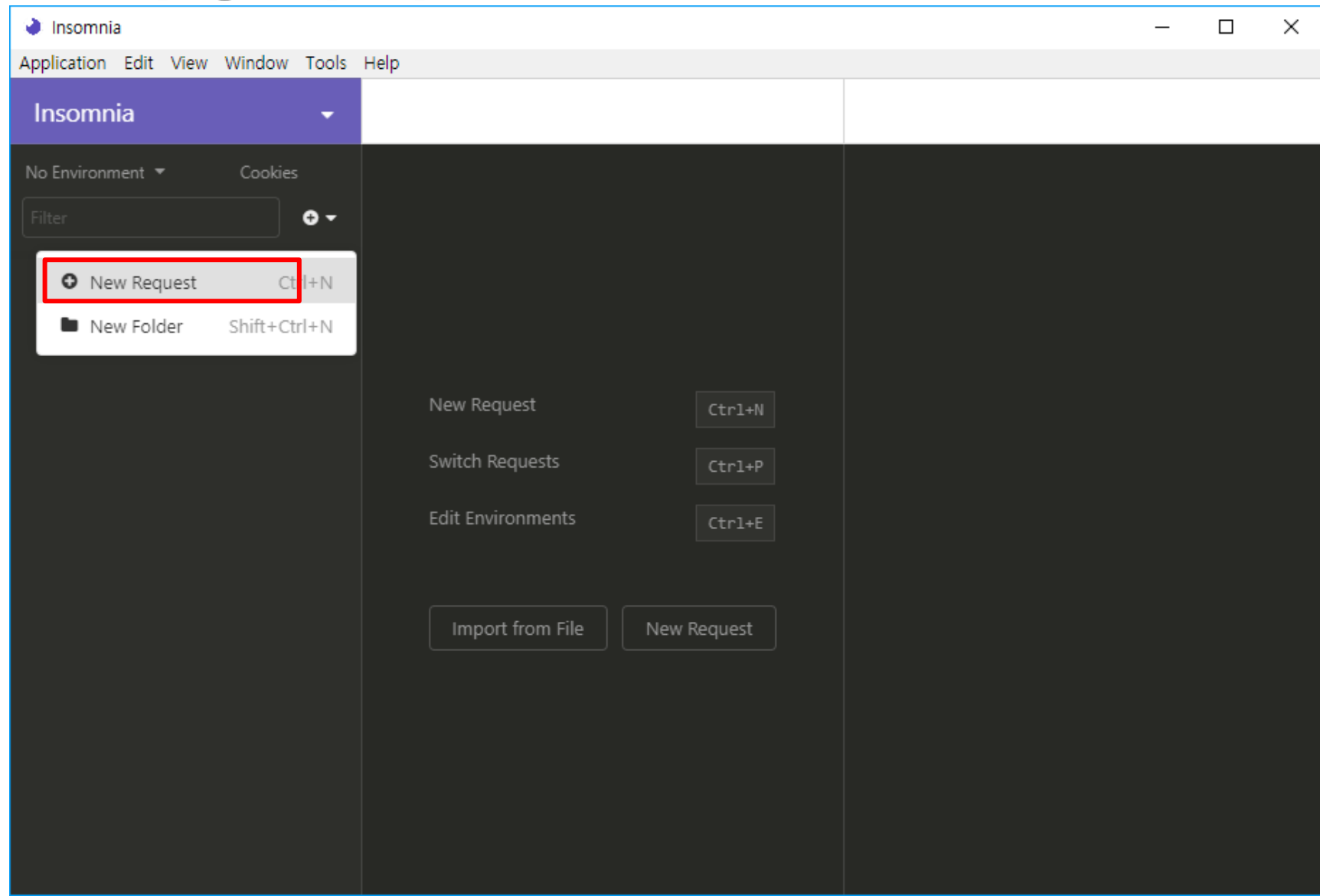
📄 Ubuntu 14.04+

OR `sudo snap install insomnia`
or [view other methods](#)

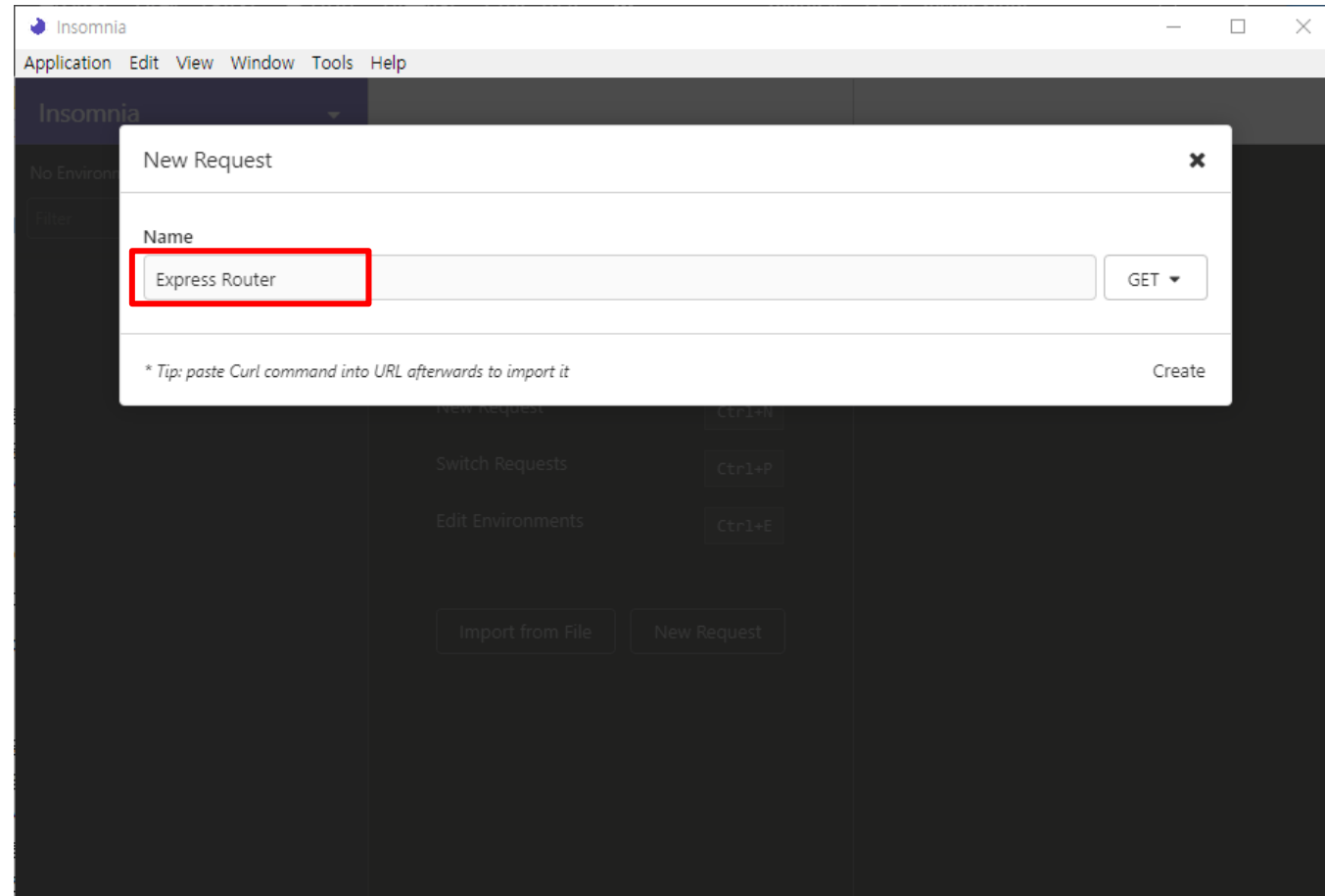
Insomnia 사용하기



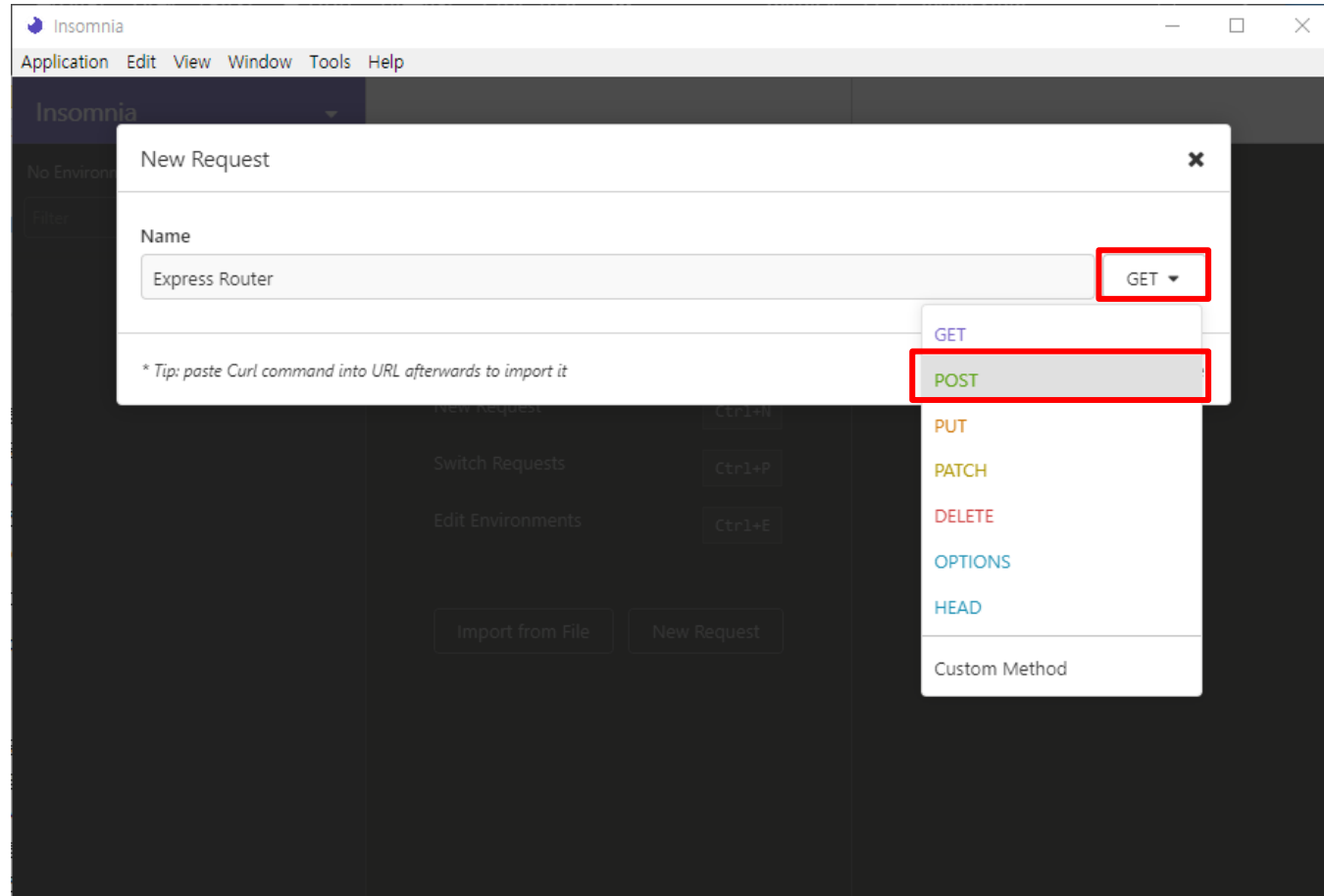
Insomnia 사용하기



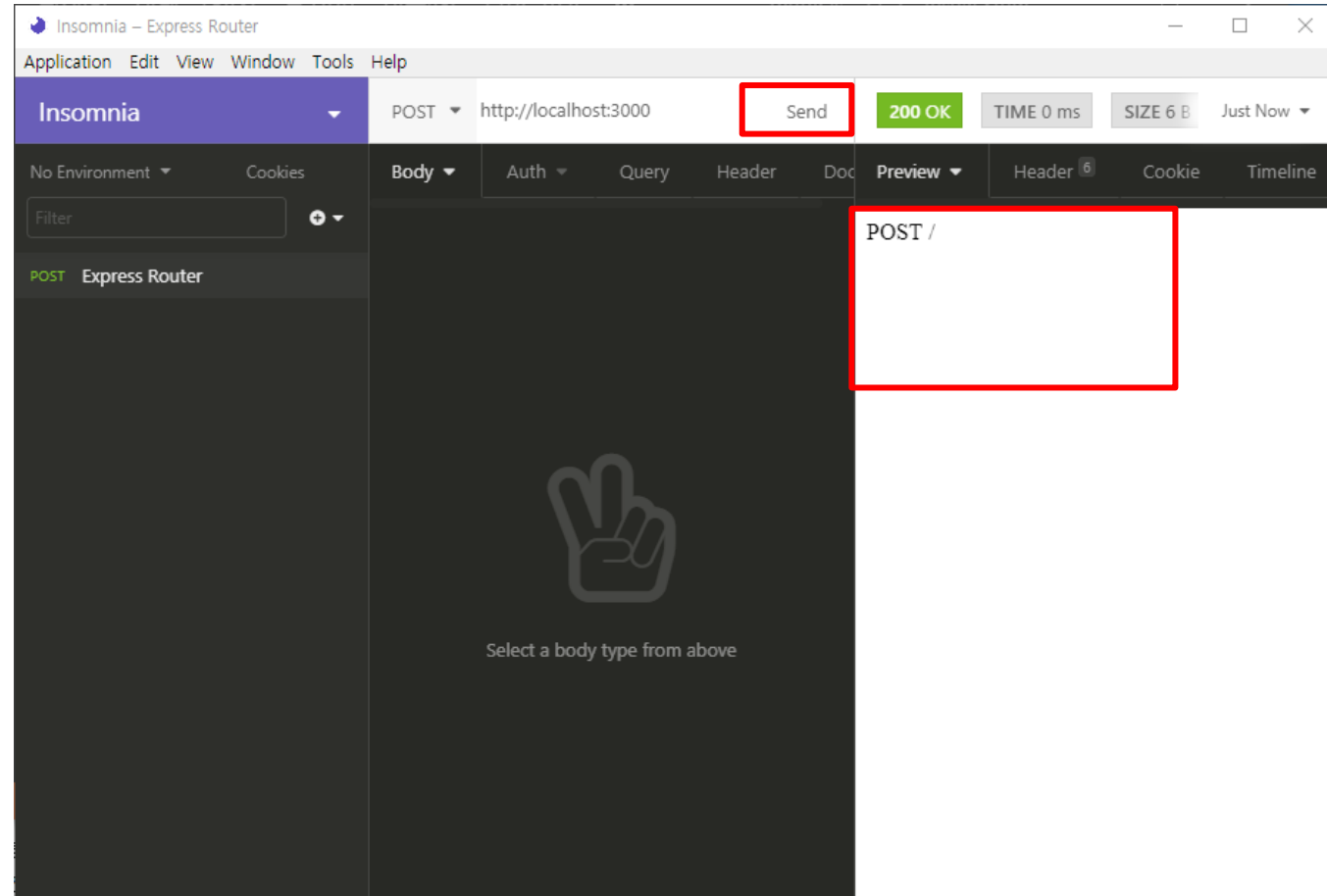
Insomnia 사용하기



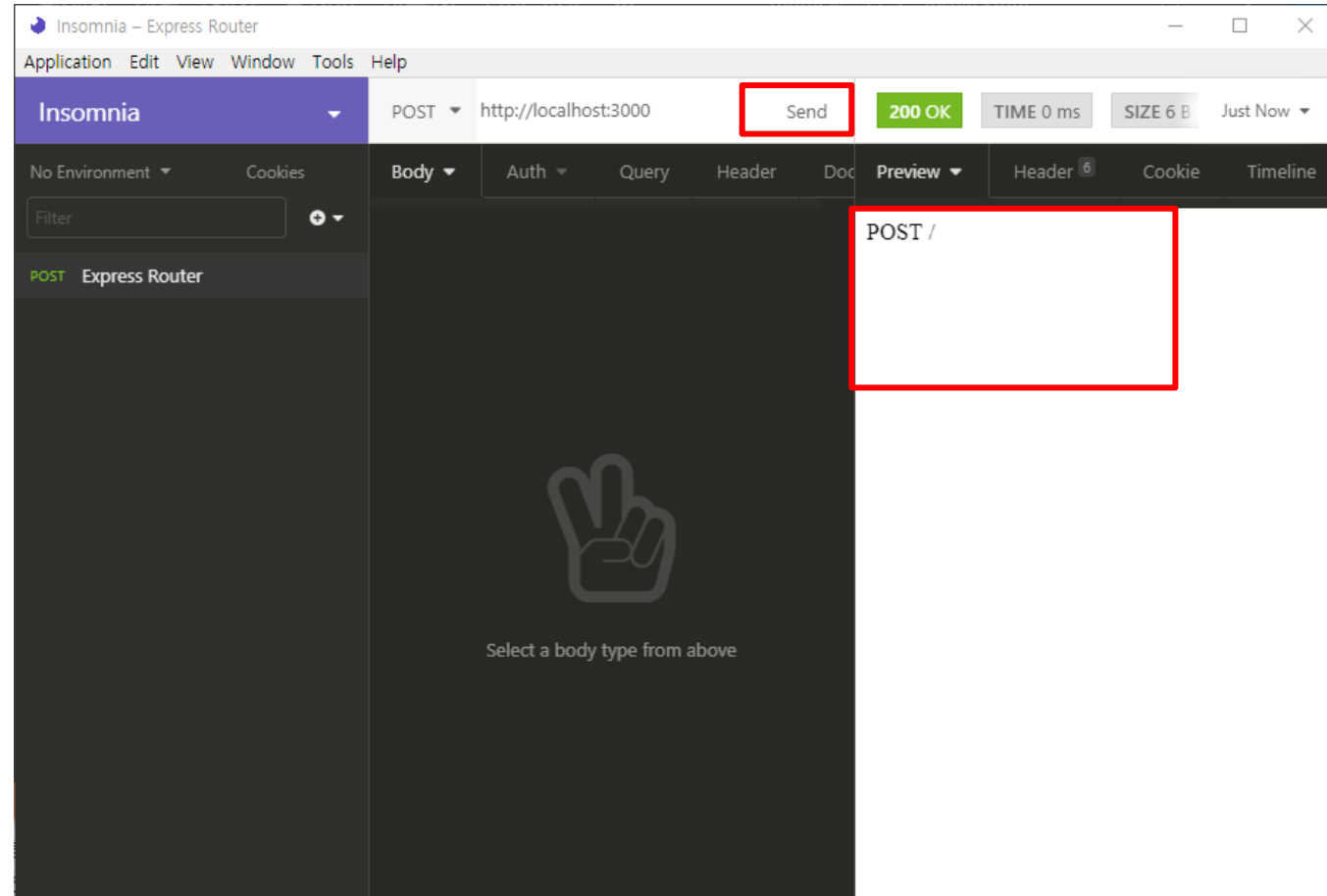
Insomnia 사용하기



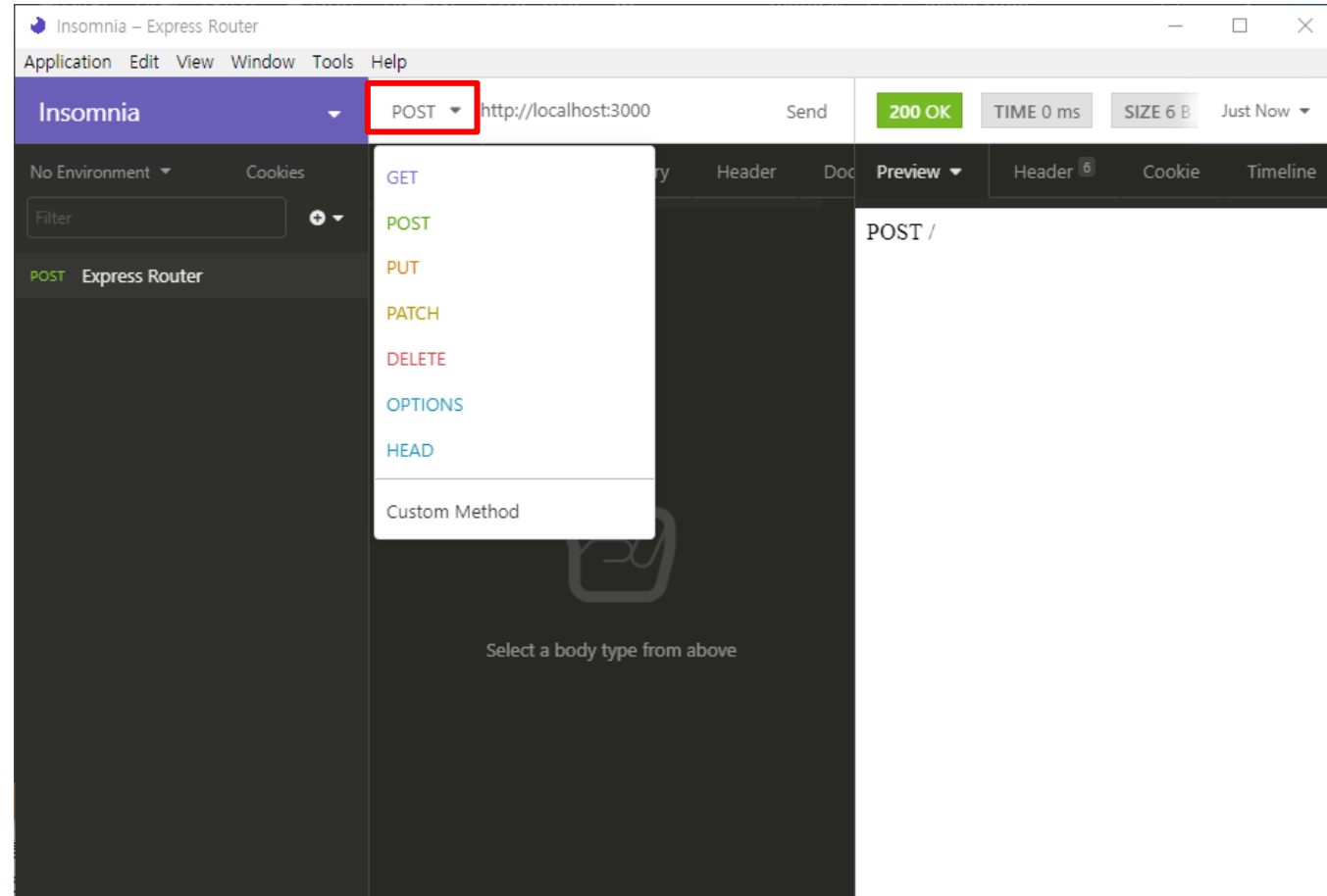
Insomnia 사용하기



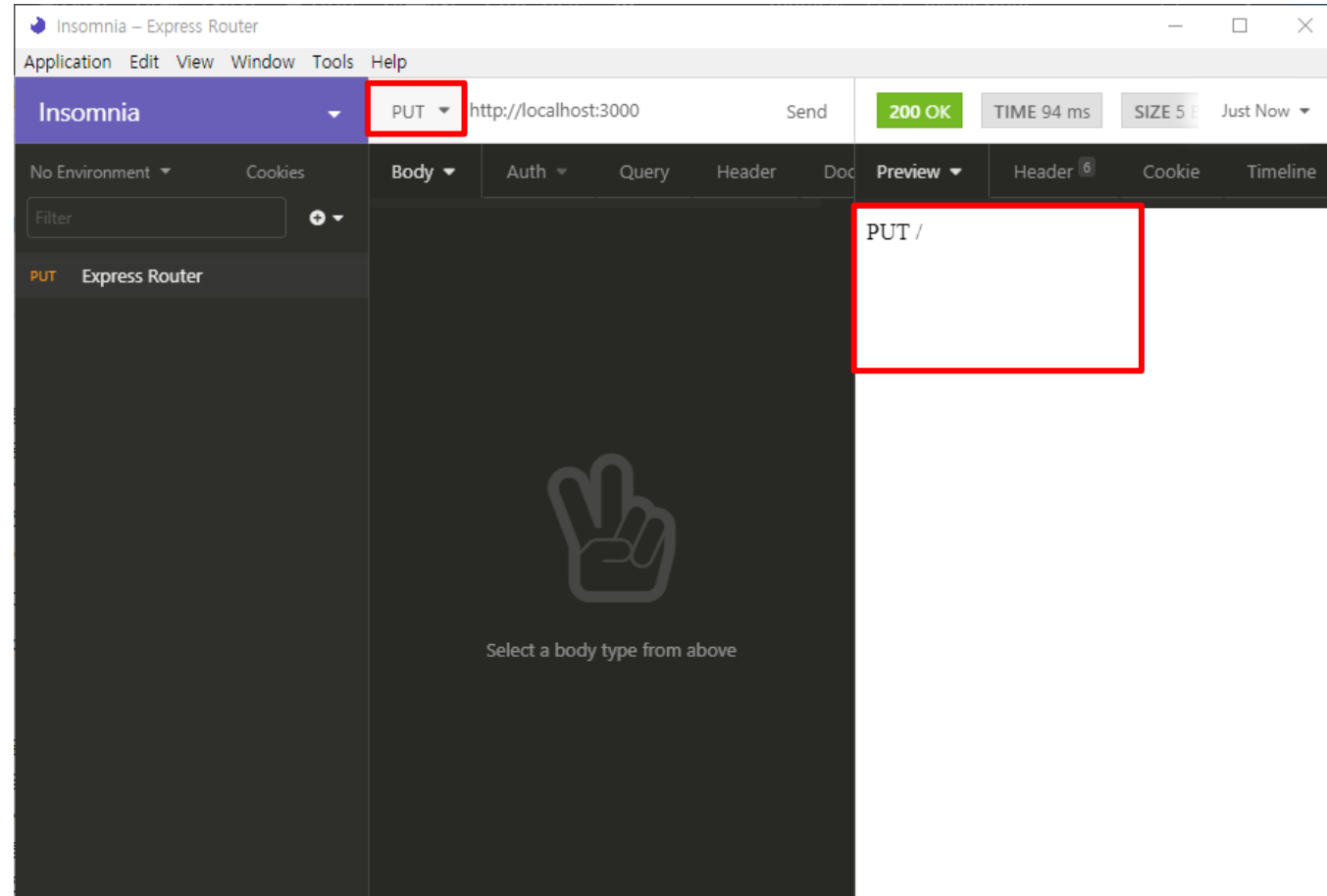
Insomnia 사용하기



Insomnia 사용하기



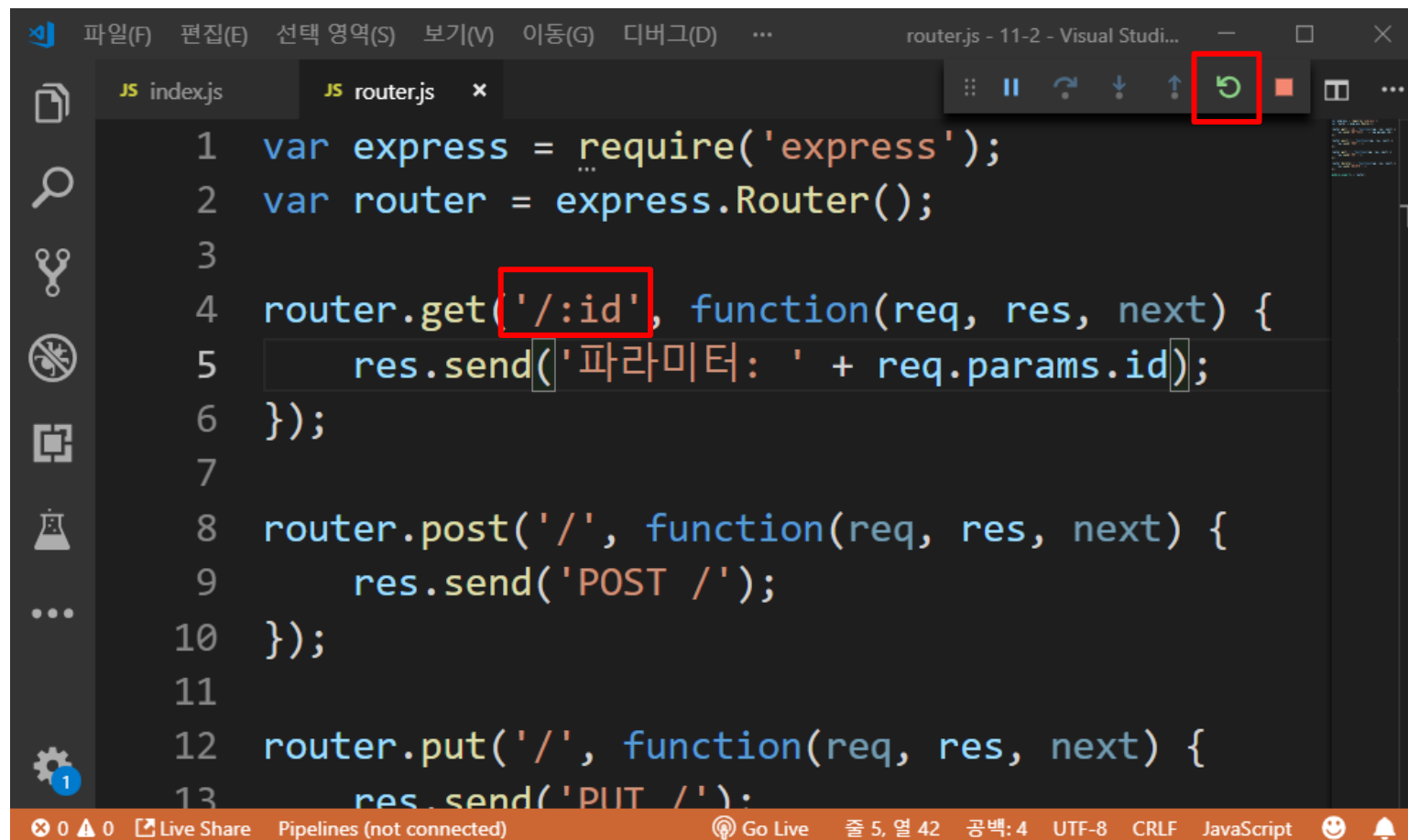
Insomnia 사용하기



Express Route Parameter

- REST Api 에서 자원의 아이디를 uri 를 통해 전달
- 아이디와 같이 Uri 의 변하는 요소를 Route parameter 라고 함
- Express Router 에서는 : (콜론) 과 파라미터 이름으로 Route parameter 를 설정
- Request 객체의 params 프로퍼티로 Route parameter 에 접근

Express Route Parameter

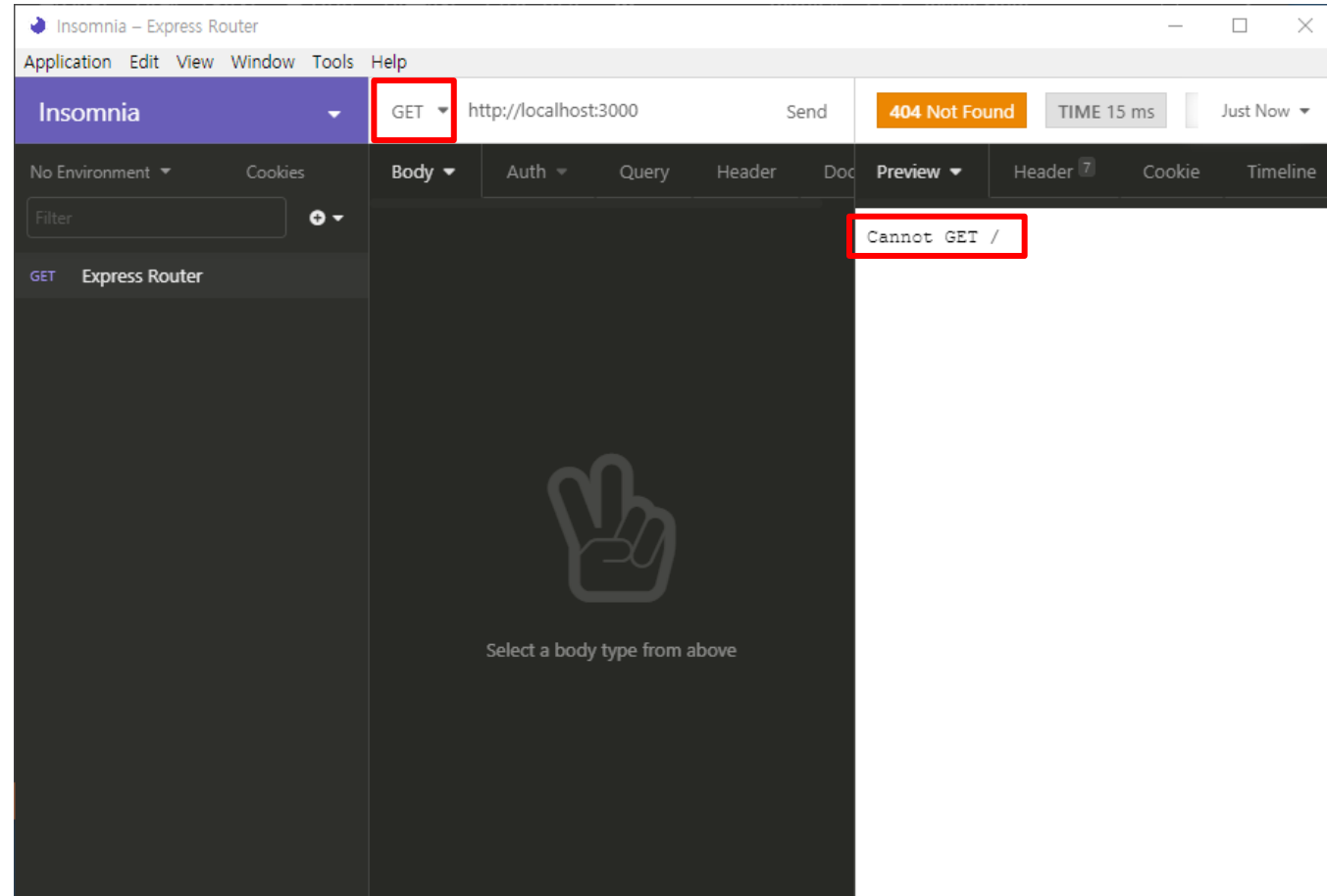


The image shows a screenshot of the Visual Studio Code editor interface. The top menu bar includes options like '파일(F)', '편집(E)', '선택 영역(S)', '보기(V)', '이동(G)', and '디버그(D)'. The title bar indicates the file is 'router.js - 11-2 - Visual Studi...'. The editor has two tabs open: 'JS index.js' and 'JS router.js'. The 'router.js' tab is active, displaying the following JavaScript code:

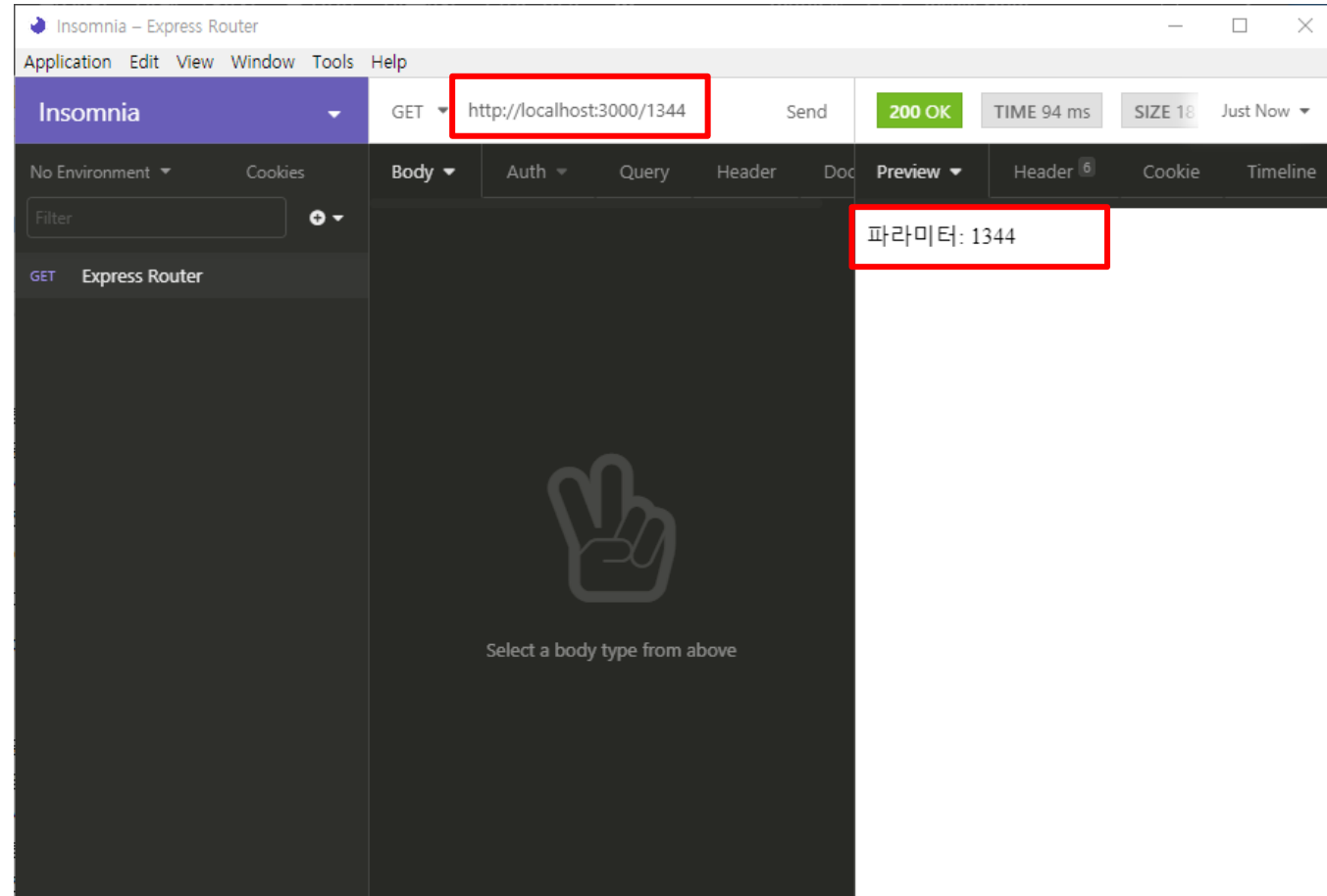
```
1 var express = require('express');
2 var router = express.Router();
3
4 router.get('/:id', function(req, res, next) {
5   res.send('파라미터: ' + req.params.id);
6 });
7
8 router.post('/', function(req, res, next) {
9   res.send('POST /');
10 });
11
12 router.put('/', function(req, res, next) {
13   res.send('PUT /');
```

In the code, the route parameter `/:id` in the `router.get` method is highlighted with a red box. Additionally, the run button (a green circular arrow icon) in the top right toolbar is also highlighted with a red box. The status bar at the bottom shows '0 0', 'Live Share', 'Pipelines (not connected)', 'Go Live', '줄 5, 열 42', '공백: 4', 'UTF-8', 'CRLF', 'JavaScript', and some system icons.

Express Route Parameter



Express Route Parameter



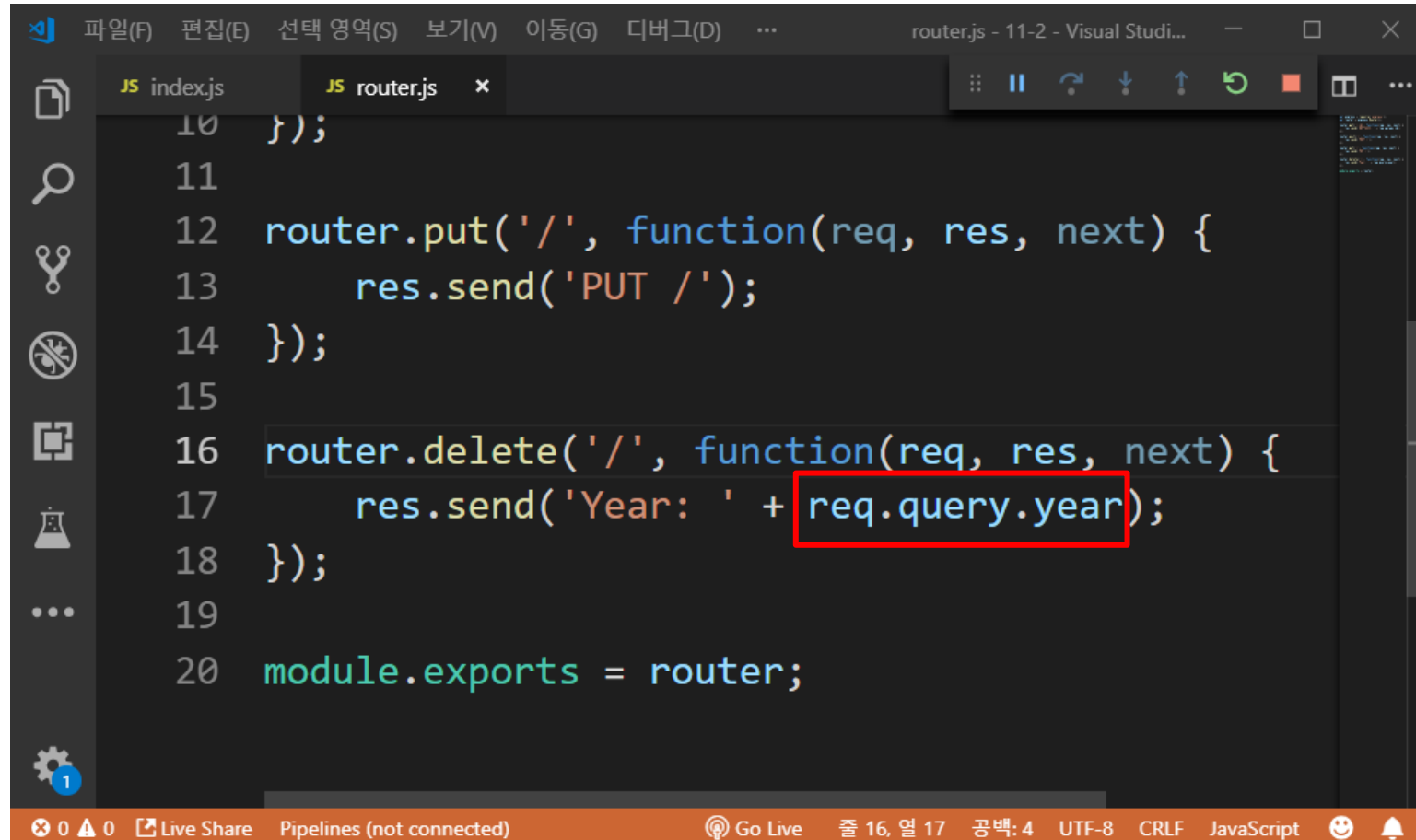
Query String

- URI 를 통해 값을 전달하는 또 하나의 방법
- REST API 에서는 주로 자원을 검색, 필터링 할 때 사용
ex) /books?author=김군오&language=kr

Express 에서 Query String 사용

- Request 객체의 query 프로퍼티로 접근
- Route parameter 와 다르게 어떤 query string 을 받을지 미리 설정하지 않음
- 라우트에 영향을 주지 않음
 - 쿼리스트링의 유무와 관계없이 http 메서드와 uri 를 통해서만 라우트가 결정
- 전달하지 않은 query string 에 접근하면 값이 설정되어 있지 않음 -
> undefined 변수로 취급

Express 에서 Query String 사용

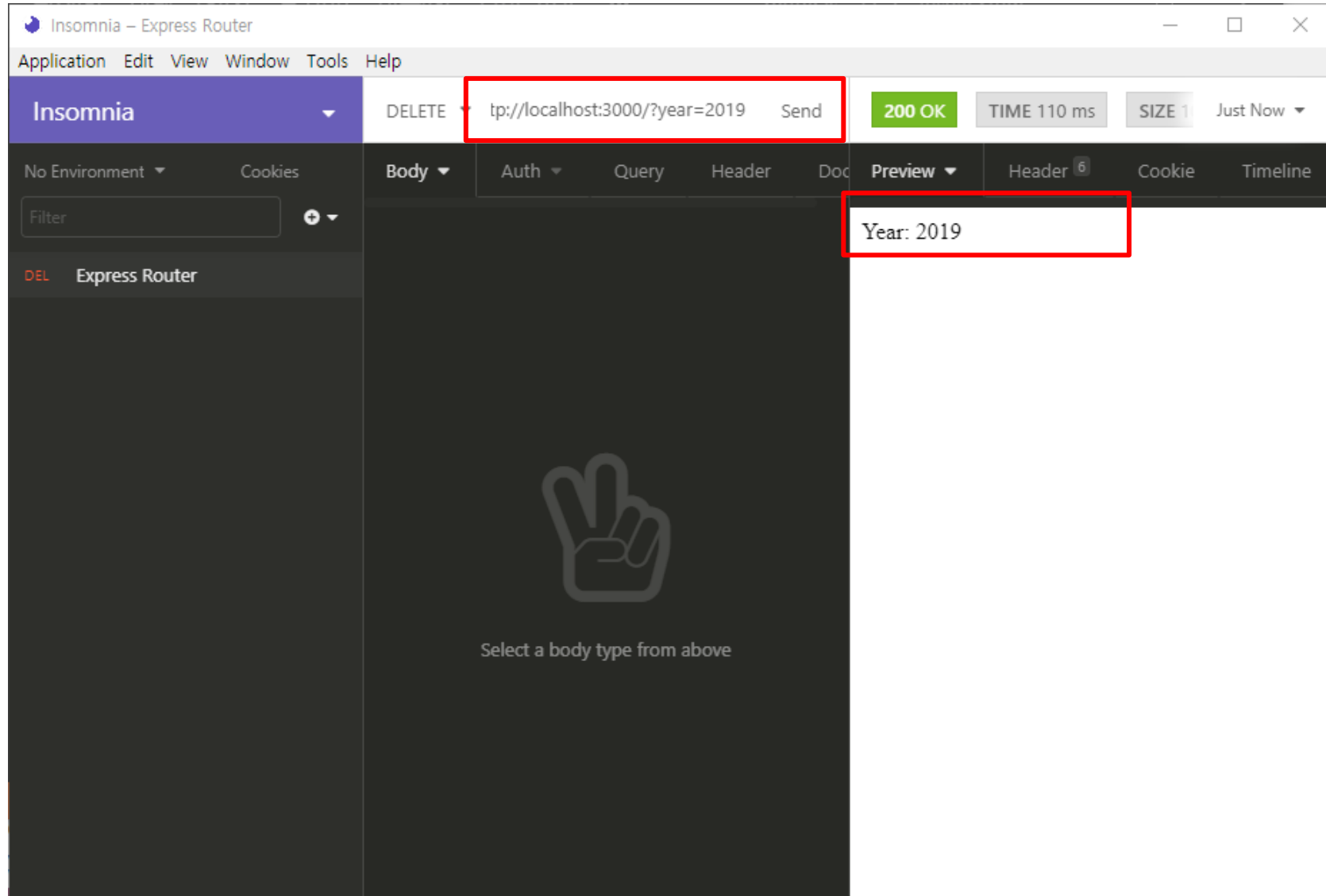


The image shows a screenshot of the Visual Studio Code editor interface. The top menu bar includes options like '파일(F)', '편집(E)', '선택 영역(S)', '보기(V)', '이동(G)', and '디버그(D)'. The title bar indicates the file is 'router.js - 11-2 - Visual Studi...'. The editor has two tabs open: 'JS index.js' and 'JS router.js'. The 'router.js' tab is active, showing the following code:

```
10 });  
11  
12 router.put('/', function(req, res, next) {  
13     res.send('PUT /');  
14 });  
15  
16 router.delete('/', function(req, res, next) {  
17     res.send('Year: ' + req.query.year);  
18 });  
19  
20 module.exports = router;
```

The code uses syntax highlighting: keywords like 'router.put', 'router.delete', 'function', and 'module.exports' are in blue; strings like 'PUT /' and 'Year: ' are in orange; and variables like 'req', 'res', 'next', and 'year' are in green. A red rectangular box highlights the expression 'req.query.year' in line 17. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The bottom status bar shows '0 0', 'Live Share', 'Pipelines (not connected)', 'Go Live', '줄 16, 열 17', '공백: 4', 'UTF-8', 'CRLF', 'JavaScript', and a notification bell.

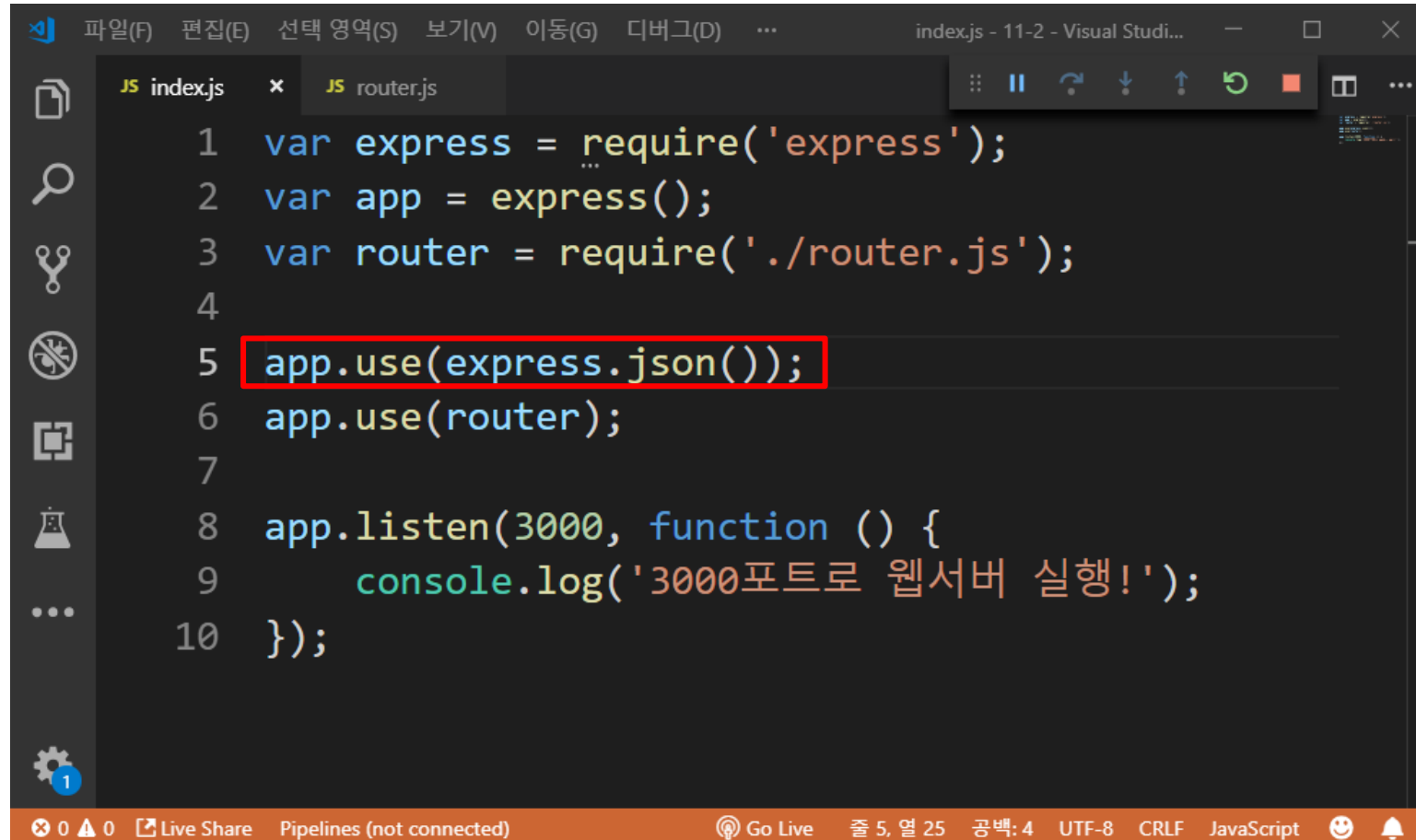
Express 에서 Query String 사용



HTTP Body

- POST, PUT 메서드는 HTTP 요청에 Body 전달 가능
- Body 에는 text, json, xml, html, 파일 등 어느 것이라도 전송될 수 있음
- Body 는 Content-type 헤더에 따라 적절히 해석되어야 함

Express JSON 미들웨어

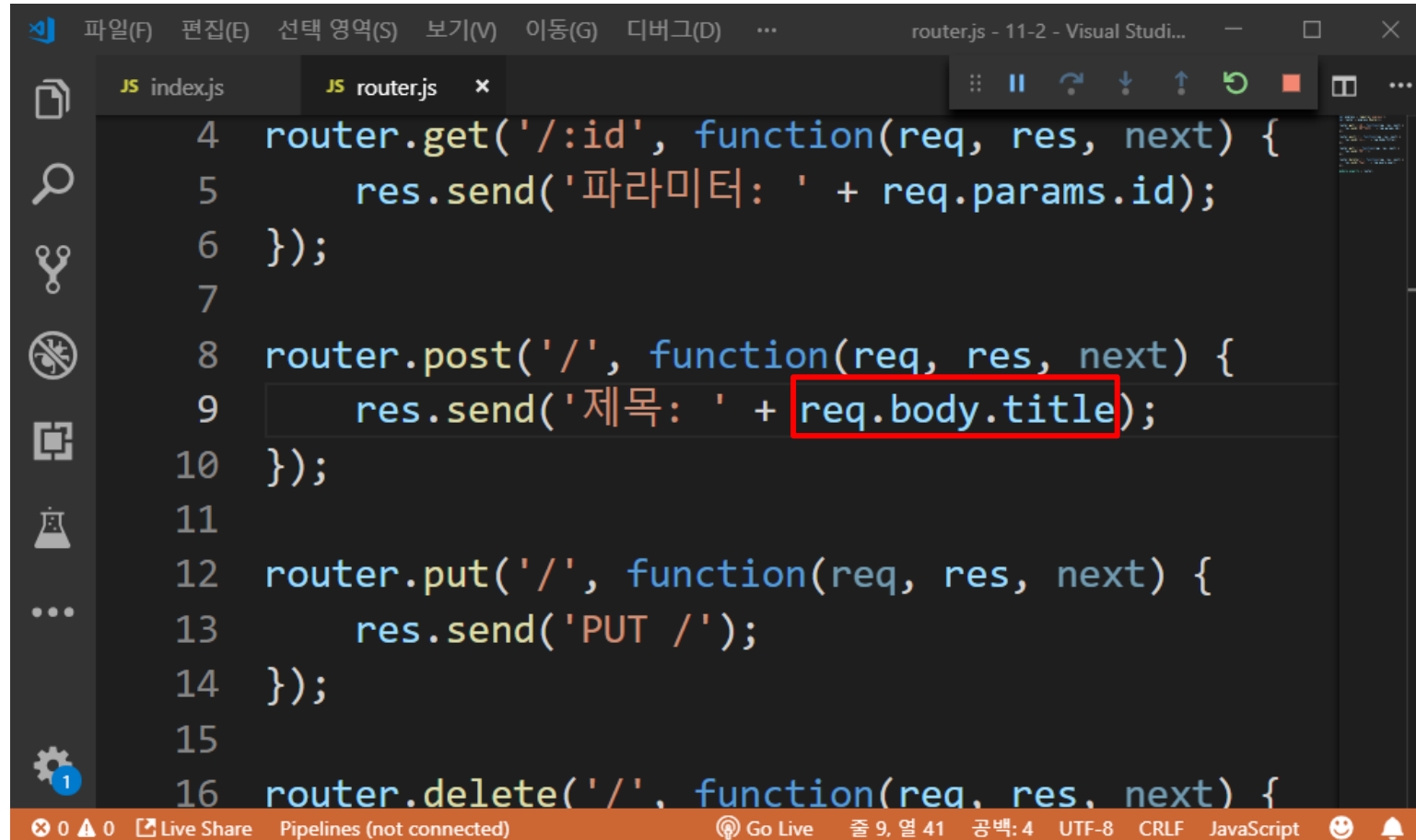


The image shows a screenshot of the Visual Studio Code editor interface. The top menu bar includes options like '파일(F)', '편집(E)', '선택 영역(S)', '보기(V)', '이동(G)', and '디버그(D)'. The title bar indicates the file is 'index.js - 11-2 - Visual Studi...'. The editor has two tabs open: 'JS index.js' and 'JS router.js'. The code in 'index.js' is as follows:

```
1 var express = require('express');
2 var app = express();
3 var router = require('./router.js');
4
5 app.use(express.json());
6 app.use(router);
7
8 app.listen(3000, function () {
9     console.log('3000포트로 웹서버 실행!');
10 });
```

The line `app.use(express.json());` on line 5 is highlighted with a red rectangular box. The status bar at the bottom shows '0 0', 'Live Share', 'Pipelines (not connected)', 'Go Live', '줄 5, 열 25', '공백: 4', 'UTF-8', 'CRLF', 'JavaScript', and a smiley face icon.

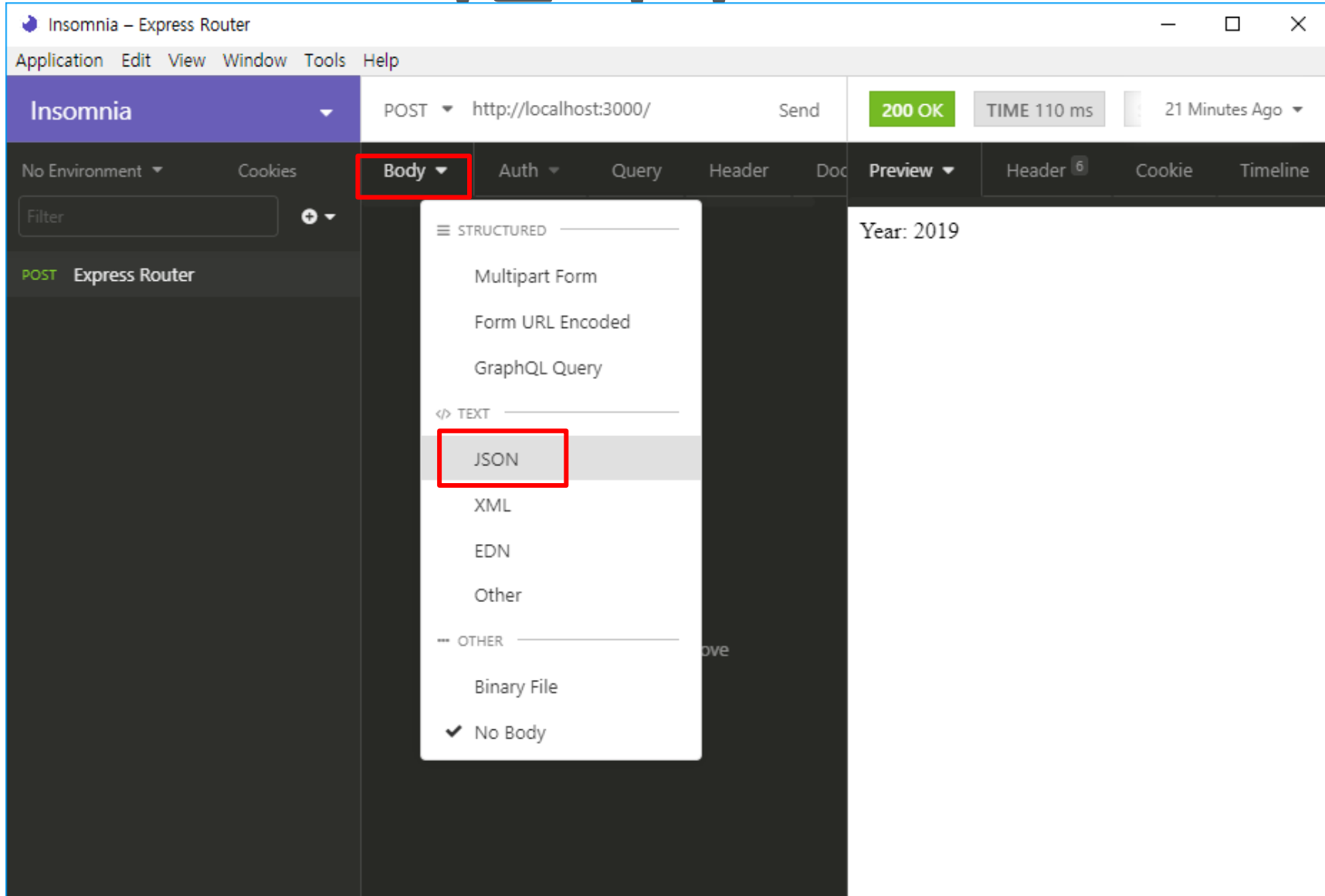
Express JSON 미들웨어



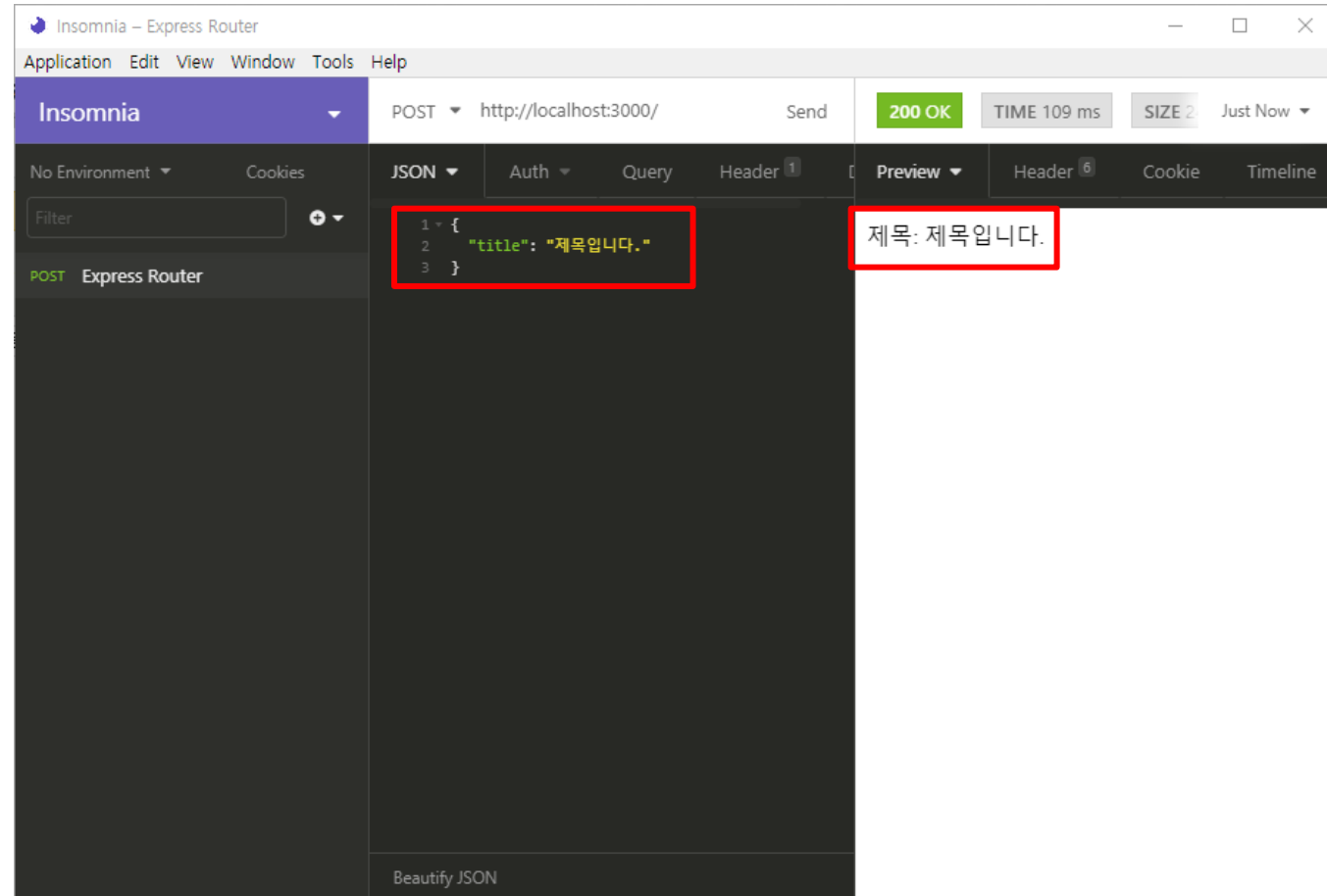
```
router.js - 11-2 - Visual Studi...
JS index.js JS router.js x
4 router.get('/:id', function(req, res, next) {
5     res.send('파라미터: ' + req.params.id);
6 });
7
8 router.post('/', function(req, res, next) {
9     res.send('제목: ' + req.body.title);
10 });
11
12 router.put('/', function(req, res, next) {
13     res.send('PUT /');
14 });
15
16 router.delete('/', function(req, res, next) {
```

0 0 Live Share Pipelines (not connected) Go Live 줄 9, 열 41 공백: 4 UTF-8 CRLF JavaScript

Express JSON 미들웨어



Express JSON 미들웨어



요약

- RESTful API
- Express.js로 RESTful API 만들기

차시 예고

- 11-3 : **nodejs를 ec2에 설치 및 PM2로 관리하기**
 - EC2 서버에 nodejs 설치
 - 도서 관리 프로그램 만들기
 - PM2 로 NodeJS 앱 관리하기

강의를 마치겠습니다
수고하셨습니다

11주차_02 RESTful API 알아보기