FAQ

How should I structure my application?

There is no definitive answer to this question. The answer depends on the scale of your application and the team that is involved. To be as flexible as possible, Express makes no assumptions in terms of structure.

Routes and other application-specific logic can live in as many files as you wish, in any directory structure you prefer. View the following examples for inspiration:

- Route listings
- Route map
- MVC style controllers

Also, there are third-party extensions for Express, which simplify some of these patterns:

- Resourceful routing

How do I define models?

Express has no notion of a database. This concept is left up to third-party Node modules, allowing you to interface with nearly any database.

See LoopBack for an Express-based framework that is centered around models.

How can I authenticate users?

Authentication is another opinionated area that Express does not venture into. You may use any authentication scheme you wish. For a simple username / password scheme, see this example.

Which template engines does Express support?

Express supports any template engine that conforms with the `(path, locals, callback)` signature.

To normalize template engine interfaces and caching, see the consolidate.js project for support. Unlisted template engines might still support the Express signature.

For more information, see Using template engines with Express.

How do I handle 404 responses?

In Express, 404 responses are not the result of an error, so the error-handler middleware will not capture them. This behavior is because a 404 response simply indicates the absence of additional work to do; in other words, Express has executed all middleware functions and routes, and found that none of them responded. All you need to do is add a middleware function at the very bottom of the stack (below all other functions) to handle a 404 response:

```
app.use((req, res, next) => {
  res.status(404).send("Sorry can't find that!")
```

```
})
```

Add routes dynamically at runtime on an instance of `express.Router()` so the routes are not superseded by a middleware function.

How do I setup an error handler?

You define error-handling middleware in the same way as other middleware, except with four arguments instead of three; specifically with the signature `(err, req, res, next)`:

```
app.use((err, req, res, next) => {
  console.error(err.stack)
  res.status(500).send('Something broke!')
})
```

For more information, see Error handling.

How do I render plain HTML?

You don't! There's no need to "render" HTML with the `res.render()` function. If you have a specific file, use the `res.sendFile()` function. If you are serving many assets from a directory, use the `express.static()` middleware function.

What version of Node.js does Express require?

- Express 4.x requires Node.js 0.10 or higher.
- Express 5.x requires Node.js 18 or higher.

Previous: More examples