Express.js

Express.js is a fast, flexible and minimalist web framework for Node.js. It's effectively a tool that simplifies building web applications and APIs using JavaScript on the server side. Express is an open-source that is developed and maintained by the Node.js foundation.

Steps to Create an Express.js Application

Step 1: Write this command to create a NodeJS application in your terminal because our express server will work inside the node application.

```
npm init
```

This will ask you for a few configurations about your project you can fill them in accordingly, also you can change them later from the **package.json** file.
Note: Use '**npm init -y**' for default initialization

Step 2: Install the necessary dependencies for our application. In this, we will install express.js dependency.

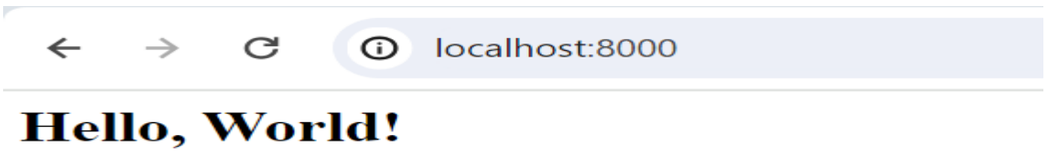
```
npm install express
```

Express Js Program to Print Hello world:

Example

```
// app.js

const express = require('express');

const app = express();


app.get('/', (req, res) => {

  res.send('Hello World!');

});


app.listen(3000, () => {

  console.log('Server is running on port 3000');

});
```

Output



Basic Routing

Example

```
// app.js

const express = require('express');

const app = express();
```

```
app.get('/', (req, res) => {
  res.send('Home Page');
});

app.get('/about', (req, res) => {
  res.send('About Page');
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Home Page
About Page
```

Route Parameters

Example

```
// app.js
const express = require('express');
const app = express();

app.get('/user/:id', (req, res) => {
  res.send(`User ID: ${req.params.id}`);
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
User ID: 123
```

Query Parameters

Example

```
// app.js
const express = require('express');
const app = express();

app.get('/search', (req, res) => {
  res.send(`Search Query: ${req.query.q}`);
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Search Query: express
```

Handling POST Requests

Example

```
// app.js
const express = require('express');
const app = express();
app.use(express.json());

app.post('/submit', (req, res) => {
  res.send(`Received: ${req.body.data}`);
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Received: test data
```

Middleware

Example

```
// app.js
const express = require('express');
const app = express();

const logger = (req, res, next) => {
  console.log(`Request: ${req.method} ${req.url}`);
  next();
};

app.use(logger);

app.get('/', (req, res) => {
  res.send('Hello World with Middleware!');
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Request: GET /
Hello World with Middleware!
```

Static Files

Example

```
// app.js
const express = require('express');
const app = express();

app.use(express.static('public'));

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Serves static files from the 'public' directory
```

Error Handling

Example

```
// app.js
const express = require('express');
const app = express();

app.get('/', (req, res) => {
  throw new Error('BROKEN');
});

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

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Something broke!
```

Using Template Engines

Example

```
// app.js
const express = require('express');
const app = express();
app.set('view engine', 'pug');

app.get('/', (req, res) => {
  res.render('index', { title: 'Express', message: 'Hello there!' });
```

```
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
Renders 'index.pug' with the given title and message
```

Connecting to a Database (MongoDB)

Example

```
// app.js
const express = require('express');
const mongoose = require('mongoose');
const app = express();

mongoose.connect('mongodb://localhost:27017/test', { useNewUrlParser: true, useUnifiedTopology: true
});

const User = mongoose.model('User', { name: String });

app.get('/', async (req, res) => {
  const user = new User({ name: 'John' });
  await user.save();
  res.send('User saved!');
});

app.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Output

```
User saved!
```