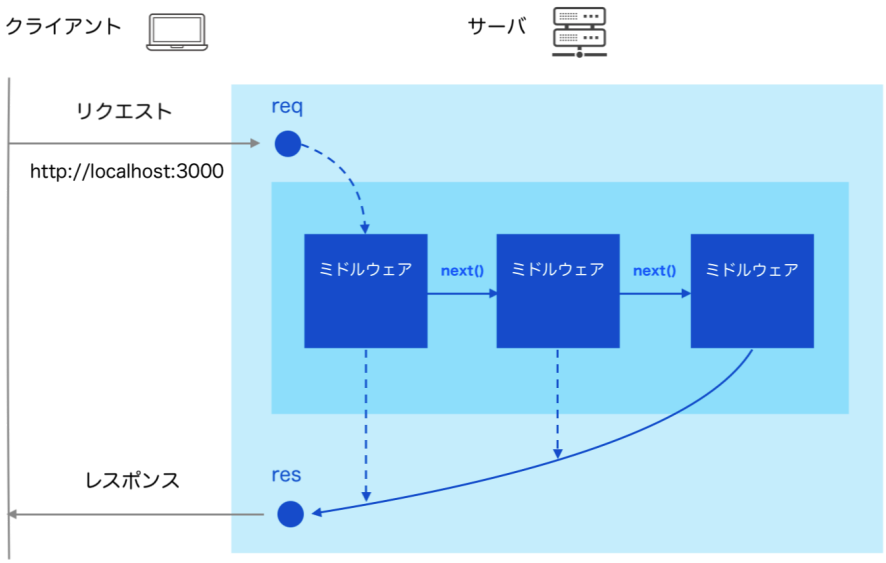
9. ミドルウェア

ミドルウェアとは

ミドルウェアは、**リクエストされたら事前処理して次の処理へ遷移**する仕組みです。 `Express` でもミドルウェア関数が用意されています。

ミドルウェアのサイクル

`Express` のミドルウェアは、次のミドルウェアへ処理を引き継いだり、リクエスト＆レスポンスを終了させるといったサイクルがあります。



- URL リクエスト
- リクエストオブジェクトの取得
- ミドルウェアの実行
- リクエストレスポンスサイクルを終了
- 次のミドルウェアを呼び出し

- レスポンスオブジェクトをレスポンス

ミドルウェアの設計はさまざま

ミドルウェアの実装は、アプリケーションの特徴や設計・デザインパターン、開発グループの考え方によってさまざまです。ミドルウェア自体のサイクルは把握しておくとよいでしょう。

参考

Express: Express アプリケーションで使用するミドルウェアの作成

Express のミドルウェア

ミドルウェアは use() で定義

ミドルウェアは `use()` で定義し、引数にコールバック関数を設定します。

```
const app = express()
app.use(コールバック関数)
```

すべてのリクエストの処理

すべてのリクエストに対して事前処理をするには、第1引数にコールバック関数を指定します。また次の処理に遷移するように、 `next()` を実行します。

```
app.use((req, res, next) => {
    next();
});
```

静的コンテンツのアクセス許可

`Express` サーバで画像、CSS、JavaScript などの静的コンテンツにアクセスできるようにするには、 `express.static()` で処理します。

```
express.static(静的コンテンツディレクトリパス)
```

ログイン処理

ファイル構成

```
├── .env
├── node_modules
├── package-lock.json
├── package.json
├── public
|      └── login.html
└── server.js
```

現在のディレクトリパス

`__dirname` は、実行中のプログラムのディレクトリパスを取得します。

```
__dirname
```

public/ アクセス許可

`server.js` で `public/` のアクセス許可をします。

```
app.use(express.static(__dirname + '/public'));
```

login.html 作成

`public/login.html` ファイルにログインフォームを作成します。

```
...
  <form action="/auth" method="post">
    <div>
      <label>ログイン名</label>
      <input type="text" name="login_name">
    </div>

    <div>
```

```
      <label>パスワード</label>
      <input type="password" name="password">
    </div>

    <div>
      <button>Loigin</button>
    </div>
  </form>
...
```

POSTのルーティング

`/auth` に対して `POST` でルーティングし、メッセージをレスポンスします。

```
app.post('/login', (req, res) => {
    let message = 'ログインできませんでした'
    res.send(message);
})
```

サーバ起動

node サーバ再起動

`server.js` を修正したら、ターミナルで 再起動します。

```
% node server
```

ブラウザで確認

http://localhost:3000/login.html にアクセスしてみましょう。また【Login】ボタンでPOSTリクエストできるかも確認してみましょう。

ログイン名
パスワード
Loigin

ソース

server.js

```
const express = require('express');

require('dotenv').config();
const host = process.env.HOST;
const port = process.env.PORT;

const app = express();

app.use(express.static(__dirname + '/public'));

app.get("/", (req, res) => {
    console.log(req.body);
    console.log(req.url);
    console.log(req.query);

    res.send("Hello Express!");
});

app.get("/profile", (req, res) => {
    res.send("This is Profile page.");
});

app.post('/auth', (req, res) => {
    let message = 'ログインできませんでした';
    res.send(message);
})

app.listen(port, host, () => {
    console.log(`Server listen: http://${host}:${port}`);
});
```

public/login.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>

<body>
  <form action="/auth" method="post">
    <div>
      <label>ログイン名</label>
```

```
      <input type="text" name="login_name">
    </div>

    <div>
      <label>パスワード</label>
      <input type="password" name="password">
    </div>

    <div>
      <button>Loigin</button>
    </div>
  </form>
</body>

</html>
```