

express-sessionで学ぶ
Cookieとセッション

セッションとは

”セッションはユーザーがWebサイトを表示して、離脱するまでの一連の流れを意味します。”

などと説明されています。

セッション (session) とは | SEO用語集

<https://www.seohacks.net/basic/terms/session/>

セッションとは

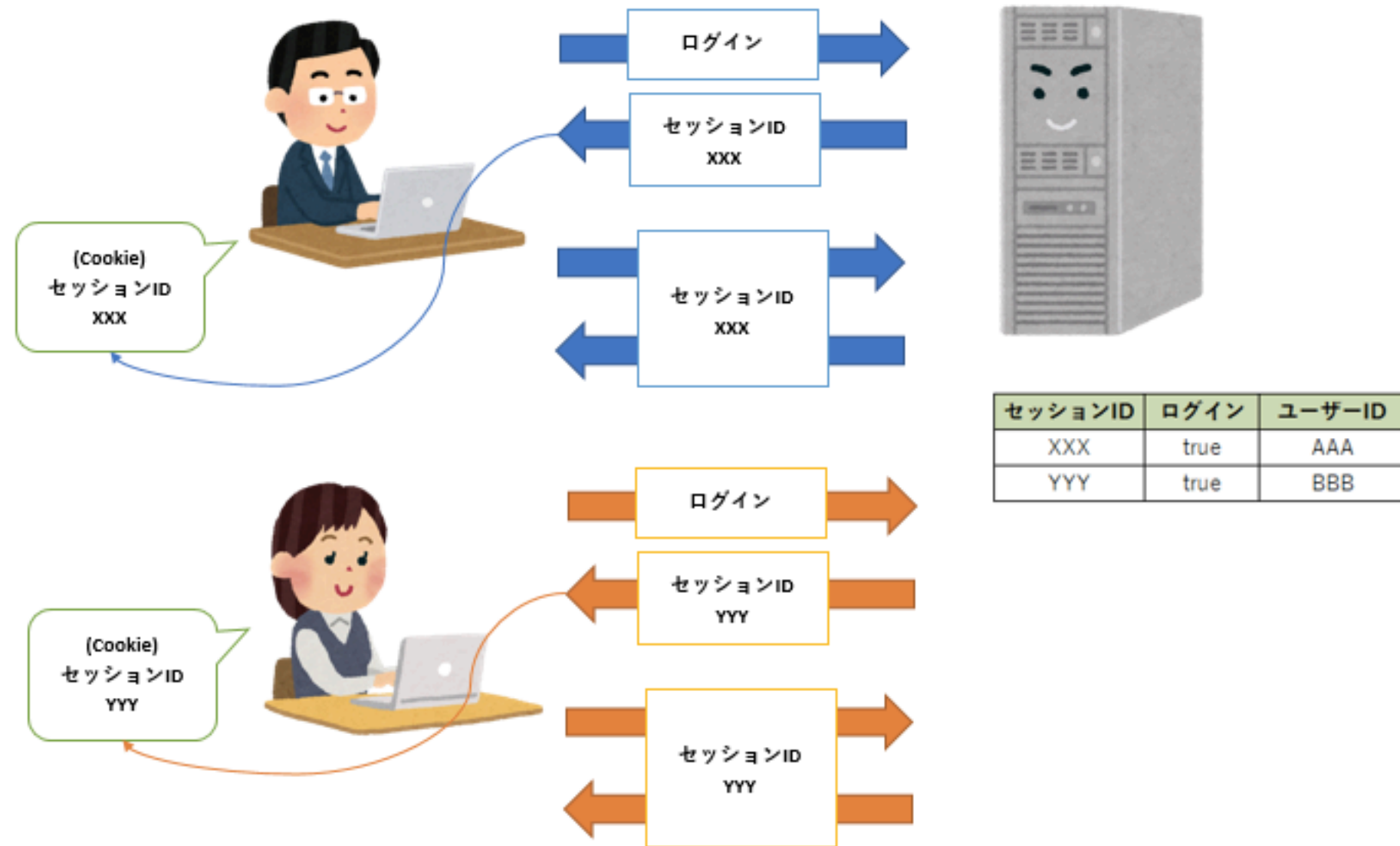
Webサイトのアカウント認証の文脈では

「セッションとは、ユーザーのログインからログアウトまでを1単位として管理するもの。サーバ側はセッションIDによりログイン状態を確認し、セッションIDに紐づけて付随する情報を保持する。」

くらいに理解してしまってもよいと思います。

※ログイン前からセッションを開始する場合もあるが、ログイン時には必ず新しくセッションを開始する。

セッションとは

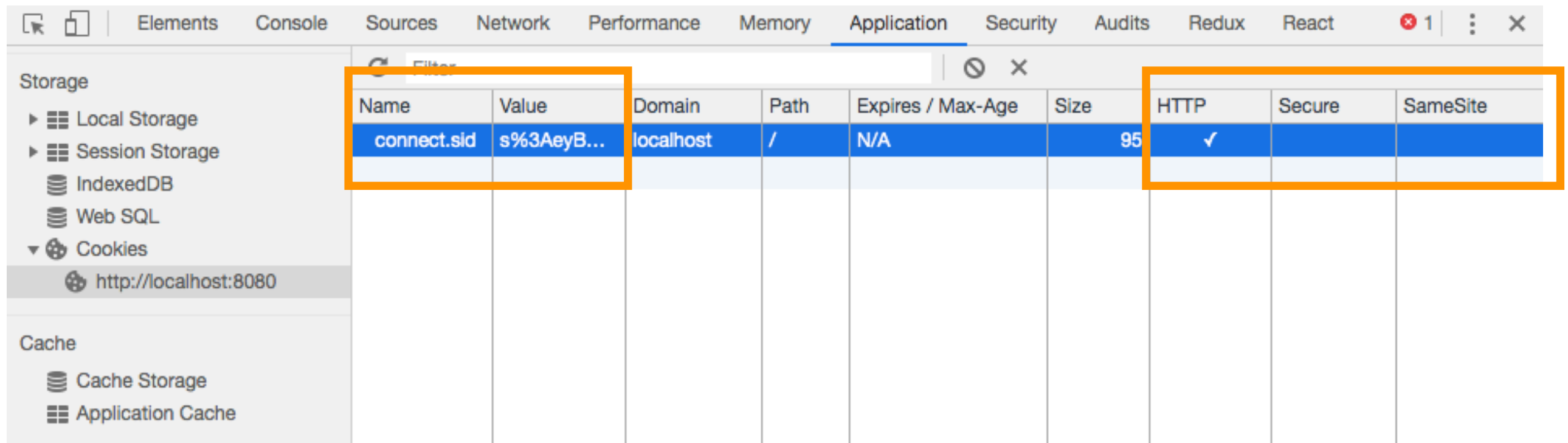


Cookieとは

ブラウザでデータを保存する方法。

Webサイトが発行したセッションidをブラウザ側はCookieで保存し、以降、そのサイトにアクセスするたびにその内容を送信する。

開発ツールでのCookieの見え方



The screenshot shows the Chrome DevTools Application tab with the 'Storage' section expanded. The 'Cookies' folder is selected, showing a single cookie for 'http://localhost:8080'. The cookie table has columns for Name, Value, Domain, Path, Expires / Max-Age, Size, HTTP, Secure, and SameSite. The first row is highlighted in blue and has orange boxes around the first six columns and the last three columns.

Name	Value	Domain	Path	Expires / Max-Age	Size	HTTP	Secure	SameSite
connect.sid	s%3AeyB...	localhost	/	N/A	95	✓		

Name : セッションIDはconnect.sidという名前で設定されている

Value : URLエンコーディングされている。%3A = : (コロン)

HTTP : ブラウザのJavascript (document.cookie) でアクセスさせない

Secure : HTTPSでのみクッキーを許可

SameSite : 別サイトからのCookieを制限。StrictまたはLax。

express-sessionでの実装 初期化

```
app.use(session({  
  secret: process.env.SESSION_SECRET,  
  resave: false,  
  saveUninitialized: false,  
  cookie: {  
    maxAge: 30 * 60 * 1000  
  },  
}));
```

セッション初期化のパラメータについては最後に触れます。

express-sessionでの実装

セッションの開始

```
req.session.hoge = 'hoge'
```

最初のセッション情報を設定すると、セッションが始まる＝セッションIDが振られる。

express-sessionでの実装

ログインチェック

```
if(req.session.hoge) {  
  // ログインしてる  
} else {  
  // ログインしてない  
}
```

任意のセッション情報を確認。

express-sessionでの実装

ログインチェック

```
if(req.session.hoge) {  
  // ログインしてる  
} else {  
  // ログインしてない  
}
```

任意のセッション情報を確認。

express-sessionでの実装

セッション終了

```
req.session.destroy();
```

セッションIDは削除されずに、紐づくセッション情報がすべて削除される。

次にセッション情報を設定すると新しいセッションIDが振られる。

express-sessionでの実装 セッションIDの確認

```
req.cookies['connect.sid']
```

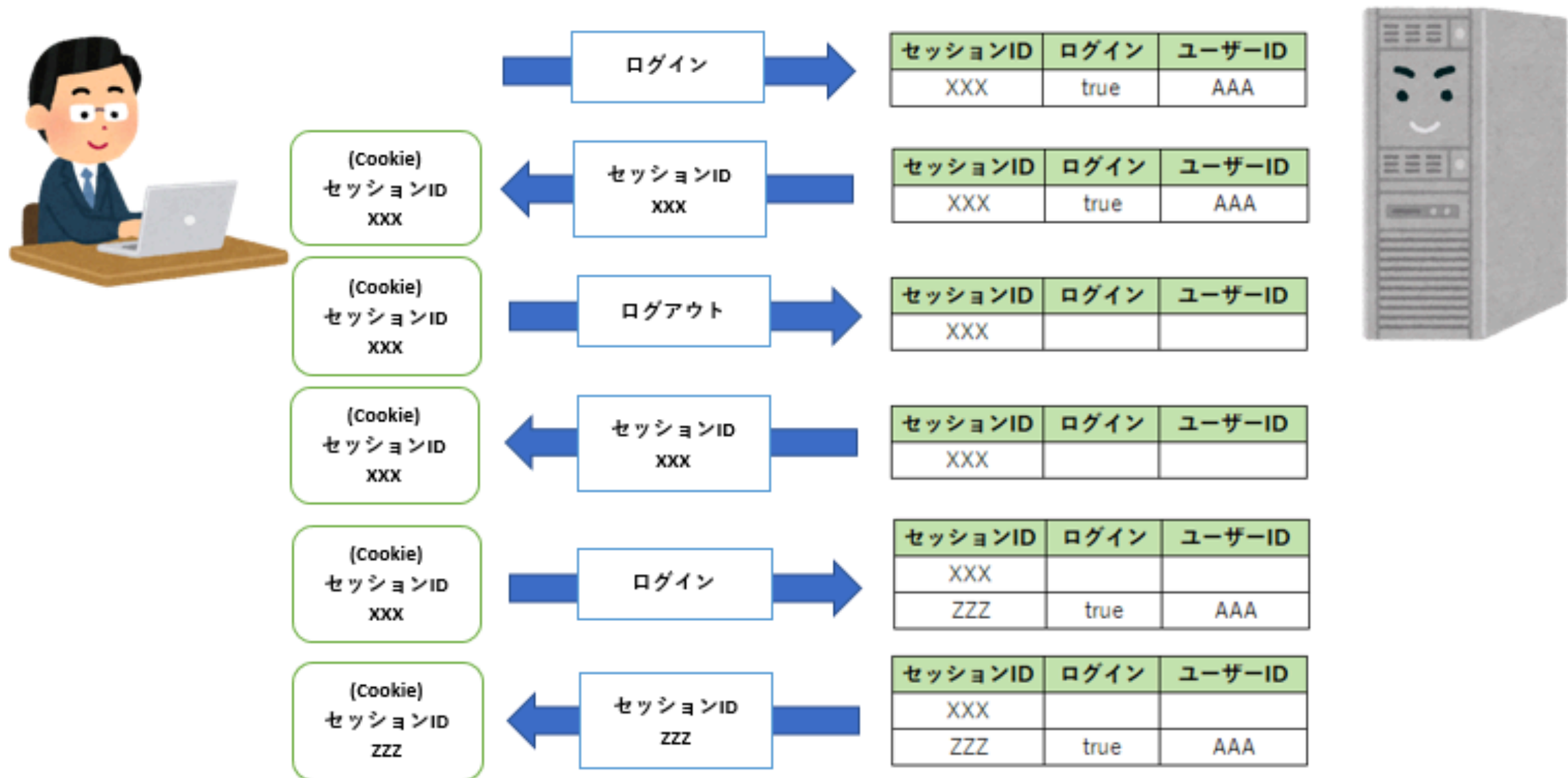
学習やデバッグ時以外ではあまり使わないかも？

express-sessionでの実装 セッションIDの確認

```
req.cookies['connect.sid']
```

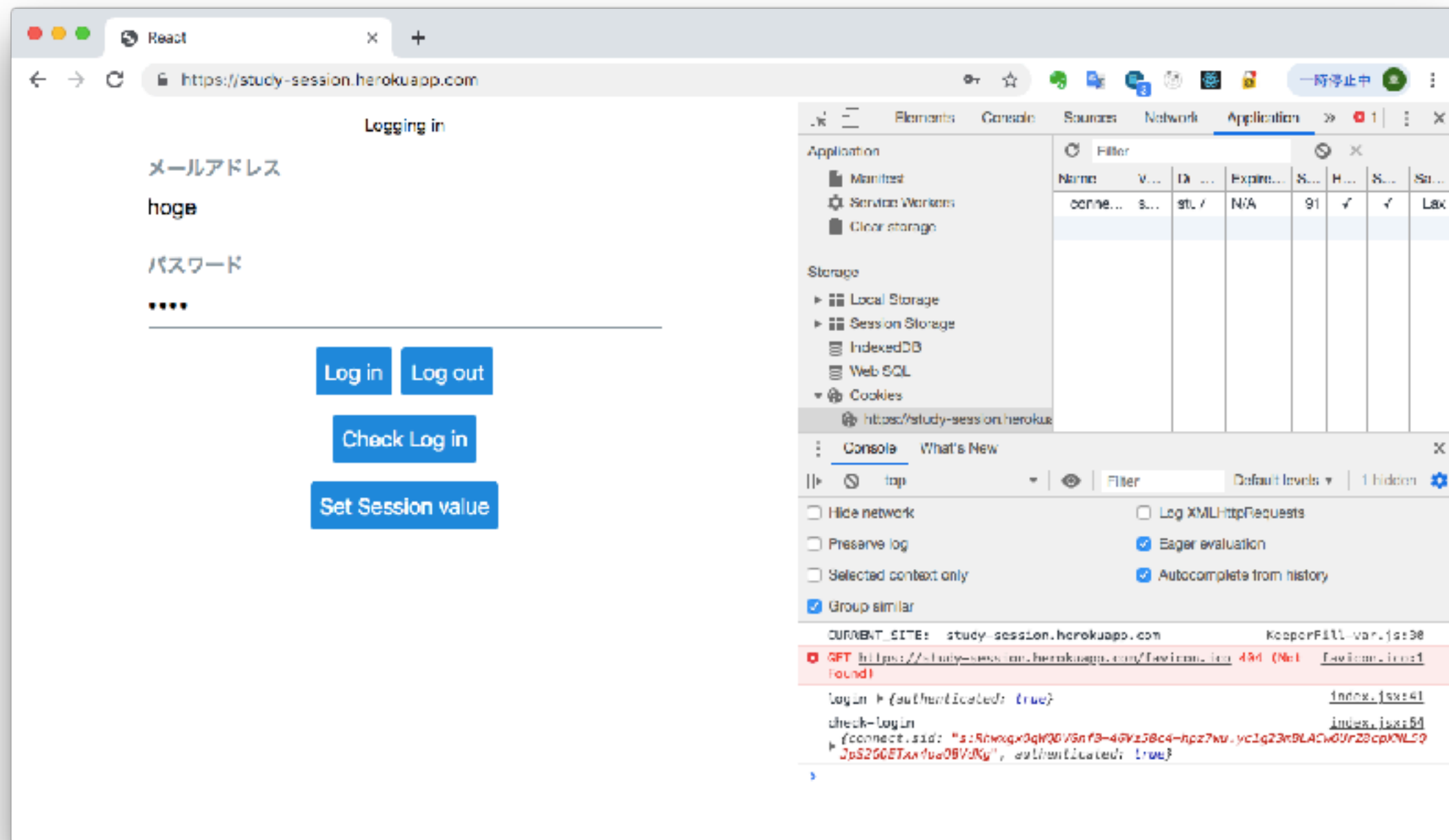
学習やデバッグ時以外ではあまり使わないかも？

express-sessionでのセッションの動作



※図中では省略していますが、ログアウトや2回目のログインのリクエストでもCookie（セッションID）が送信されています。

プログラムで確認



<https://github.com/ytkayamura/study-session>
<https://study-session.herokuapp.com/>

おまけ

express-session初期化時のパラメータ

`secret`

秘密の乱数の種みたいなもの。適当な文字列を設定する。

`resave`

セッションストアを強制的に上書きする。

defaultはtrueだが、通常はfalseを設定する。

`saveUninitialized`

通常のセッション開始(ブラウザ側ログインとサーバ側のセッション情報の設定)を行わずとも強制的にセッションを開始。

falseだとリソースを節約できる。

trueにすればセッションなしの並列リクエストによる競合発生を回避できる。

trueにする際にはEUのCookie法違反に注意。

デフォルトはtrueだが、明示的な設定を推奨。

おまけ

express-session初期化時のパラメータ

cookie.maxAge

ブラウザを閉じてでもセッション(ログイン状態)を保持する場合に設定

cookie.secure

true推奨だが、httpsでないとcookieが使えなくなる。

cookie.httpOnly

ブラウザ側のjavascriptでcookieをいじらせない
デフォルトはtrue

cookie.sameSite

別サイトからのCookieを制限。

StrictとLaxがあるがStrictでオッケー。

Laxは別サイトからのPOSTでCookieを受け取らない。GETでは受け取る。
デフォルトは設定なし。

Cookie Same Site検証ページ

<https://same-site-default.herokuapp.com>

<https://same-site-strict.herokuapp.com>

<https://same-site-lax.herokuapp.com/>

<https://attack-cookie.herokuapp.com/>