

# Node.js \_ 쿠키, 세션

OYR 2조 \_ 오다혜, 이정화

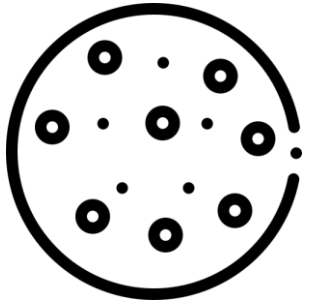
intro \_ about cookie and session

01 \_ cookie

쿠키 사용 준비, 사용 예제

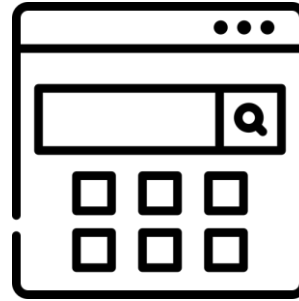
02 \_ session

세션 사용 준비, 사용 예제



Cookie \_ 쿠키

이름, 값, 만료일(저장 기간 설정), 경로 정보로 구성  
클라이언트에 총 300개의 쿠키까지 저장 가능  
하나의 도메인 당 20개의 쿠키를 가질 수 있음  
하나의 쿠키는 4KB(=4096byte)까지 저장 가능



Session \_ 세션

웹 서버에 웹 컨테이너의 상태를 유지하기 위한 정보를 저장  
브라우저를 닫거나 서버에서 세션을 삭제했을때만 삭제가  
되므로 쿠키보다 비교적 보안이 좋음  
서버 용량이 허용하는 한 저장 데이터에 제한이 없음  
각 클라이언트 고유 Session ID를 부여하고 Session ID로  
클라이언트를 구분하여 각 클라이언트 요구에 맞는 서비스  
제공

## OYR \_ Node.js \_ intro

	Cookie _ 쿠키	Session _ 세션
저장 위치	클라이언트(=접속자의 PC)	웹 서버
저장 형식	TEXT	OBJECT
만료 시점	쿠키 저장시 설정가능 브라우저가 종료되어도 만료시점이 지나지 않으면 자동 삭제되지 않음	브라우저 종료시 삭제 다만 따로 기간을 지정 가능

## Cookie 만들어보기

```
var http = require('http');
http.createServer(function(request, response){
  response.writeHead(200, {
    'Set-Cookie': ['cookie=white chocolate macadamia', 'cookie2=double chocolate']
  });
  response.end('subway cookie menu');
}).listen(3000);
```

subway cookie menu

The screenshot shows the Chrome DevTools Network tab. The 'Cookies' sub-tab is selected for the 'localhost' resource. The 'Response Cookies' table lists two cookies: 'cookie' with value 'white chocolate macadamia' and 'cookie2' with value 'double chocolate chips'. Both cookies are local, session-based, and have no expiration date.

Name	Value	Do...	Path	Exp...	Size	Http...	Sec...	Sa...	Pri...
cookie	white chocolate macadamia	loc...		Ses...	33				Me...
cookie2	double chocolate chips	loc...		Ses...	30				Me...

## Cookie-parser 사용하기

요청된 cookie를 쉽게 추출할 수 있도록 도와주는 미들웨어

npm install cookie-parser --save를 통해 install하여 사용 가능

SERVER IS RUNNING...

```
PS D:\2020_OYR\발표자료> npm install cookie-parser --save
```

```
var http = require('http');  
var express = require('express');  
var cookieParser = require('cookie-parser');  
var app = express();  
app.use(cookieParser());
```

## 입력한 값을 쿠키로 집어넣기

### 1. 폼 만들기

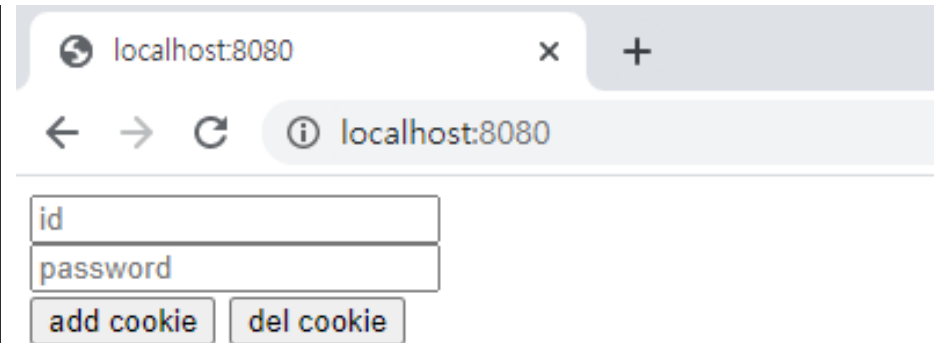
```
var http = require('http');
var url = require('url');
var querystring = require('querystring');

var server = http.createServer(function(request, response){
    response.writeHead(200, {'Content-Type': 'text/html'});
    response.end(text);
});

var text = `
<html>
  <head>

  </head>
  <body>
    <form action='/login' method='post'>
      <input type="text" id="name" value="" placeholder="id"><br>
      <input type="password" id="value" value="" placeholder="password"><br>
      <input type="button" value="add cookie" onclick="AddCookie()">
      <input type="button" value="del cookie" onclick="DelCookie()">
    </form>
  </body>
</html>
`;

server.listen(8080, function(){
    console.log('Server is running...');
});
```



localhost:8080

id

password

add cookie del cookie

```
PS D:\2020_OYR\발표자료> node 00.js
Server is running...
```

## 입력한 값을 쿠키로 집어넣기

### 2. 입력받은 값을 쿠키로 넣는 함수 만들기

```
function SetCookie( strName, strValue, iSecond ){
    var strCookie = strName + "=" + encodeURIComponent(strValue);
    if( typeof iSecond === "number" ){
        strCookie += "; max-age=" + iSecond;
    }
    document.cookie = strCookie;
}

function AddCookie(){
    var strName = document.getElementById("name" ).value;
    var strValue = document.getElementById("value" ).value;
    SetCookie( strName, strValue, null );
}

function DelCookie(){
    var strName = document.getElementById("name" ).value;
    var strValue = document.getElementById("value" ).value;
    SetCookie( strName, strValue, 0 );
}
```



입력한 값을 쿠키로 집어넣기

## 3. 값을 입력해보기

information

.....

add cookie

del cookie

Elements Console Sources Network Performance Memory Application Security Lighthouse

Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies

Blocked Requests

10 ms 20 ms 30 ms 40 ms 50 ms 60 ms 70 ms 80 ms 90 ms 100 ms 110 ms

localhost favicon.ico

Request Cookies

Name	Value	Do...	Path	Exp...	Size	Http...	Sec...	Sa...	Pri...
information	helloworld	loc...	/	Ses...	21				Me...

### Session 사용하기

Cmd창에서 `npm install -s express-session` 으로 모듈 설치

```
PS C:\Users\ohda\Desktop\세송내학교\openyearround\node> npm install -s express-session
+ express-session@1.17.1
updated 1 package and audited 212 packages in 2.001s

9 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

### 모듈 사용

Session 실행하면서 객체 전달

```
var express = require('express')
var parseurl = require('parseurl')
var session = require('express-session')
```

```
var app = express()
```

```
app.use(session({
  secret: 'keyboard cat',
  resave: false,
  saveUninitialized: true
}))
```

### Session 사용하기

콘솔 로그(console.log(session)) 를 찍은 결과

```
Session {  
  cookie: { path: '/', _expires: null, originalMaxAge: null, httpOnly: true }  
}
```

```
app.get('/',function(req,res,next){  
  
  if(req.session.num === undefined){  
    req.session.num = 1;  
  }  
  else{  
    (req.session.num)++;  
  }  
  res.send("hello");  
  var num = req.session.num;  
  console.log(num);  
})
```

```
[nodemon] restarting due to changes...  
[nodemon] starting `node app.js`  
8001  
1  
2   Session {  
3     cookie: { path: '/', _expires: null, originalMaxAge: null, httpOnly: true },  
4     num: 1  
5   }  
6   Session {  
    cookie: { path: '/', _expires: null, originalMaxAge: null, httpOnly: true },  
    num: 2  
  }  
  Session {  
    cookie: { path: '/', _expires: null, originalMaxAge: null, httpOnly: true },  
    num: 3  
  }  
}
```

### Session 사용하기

접속할 때마다 사용자를 식별하는 쿠키값을 서버로 전송  
서버는 이를 받아서 사용자의 데이터 조작

Filter <span>⌵</span> <span>⌵</span> <input type="checkbox"/> Only blocked				
Name	Value	Domain	Path	Expires / ...
connect.sid	s%3Aun6nq1_bigiUGPWvgIBUcyToWpYm9iky.5lxlOJyx8pD...	localhost	/	Session

### 파일에 데이터 저장하기

세션 데이터는 기본적으로는 컴퓨터 '메모리'에 저장됨

서버가 꺼지면 데이터가 다 날아가게 됨 - 따라서 별도로 저장해서 관리해야 함

`npm install -s session-file-store`

```
app.use(session({
  secret: '123456',
  resave: false,
  saveUninitialized: true,
  store: new FileStore()
}));
```

```
PS C:\Users\ohda\Desktop\세송내학교\openyearround\node> npm install -s session-file-store
+ session-file-store@1.4.0
updated 1 package and audited 212 packages in 3.74s

9 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

## 파일에 데이터 저장하기

```

  sessions
    { } 9wz3FHxBvAm_i...

JS template.js JS app.js { } 9wz3FHxBvAm_i7yF4S7oRkdGoMuZBKJN.json X JS main.js JS auth.js test.html
sessions > { } 9wz3FHxBvAm_i7yF4S7oRkdGoMuZBKJN.json > ...
1 [{"cookie":{"originalMaxAge":null,"expires":null,"httpOnly":true,"path":"/"},"num":2,"__lastAccess":1590937943148}]

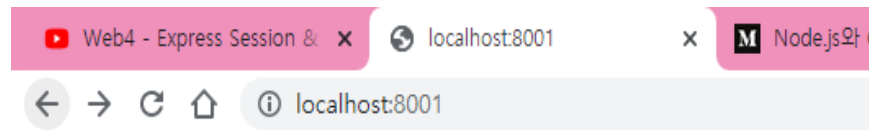
```

Name	×	Headers	Preview	Response	Initiator	Timing	Cookies
localhost		<b>Accept:</b> text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9 <b>Accept-Encoding:</b> gzip, deflate, br <b>Accept-Language:</b> ko,en-US;q=0.9,en;q=0.8,ko-KR;q=0.7 <b>Cache-Control:</b> max-age=0 <b>Connection:</b> keep-alive <b>Cookie:</b> connect.sid=s%3A9wz3FHxBvAm_i7yF4S7oRkdGoMuZBKJN.foN05D%2BT8w0XkB6DjHpPD5gkCU7oJRYU%2F%2BpJ3WU5B8Y <b>Host:</b> localhost:8001					

## 로그인창 만들기

### 1. 로그인 버튼 만들기

```
app.get('/',function(req,res,next){  
    var text = `  
        <html>  
            <head>  
            </head>  
            <body>  
                <a href="/login_page"> LOGIN </a>  
            </body>  
        </html>  
    `;  
    res.send(text);  
})
```

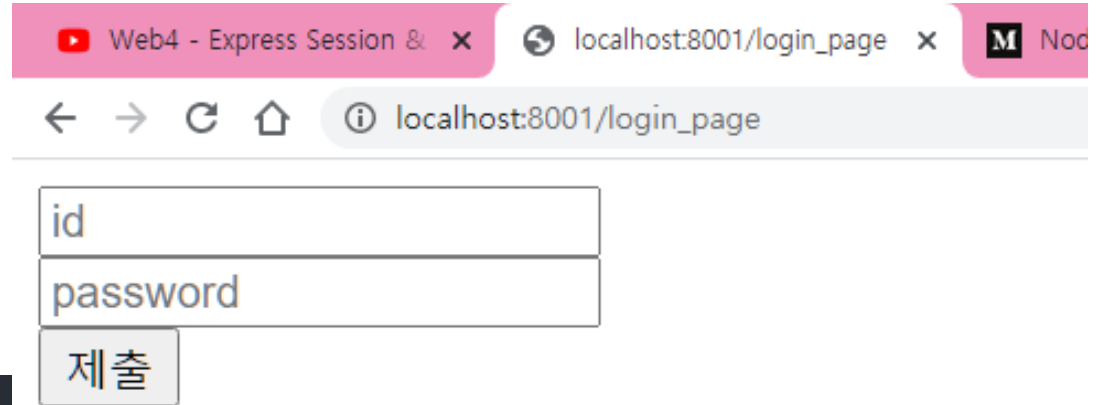


## 로그인창 만들기

### 2. 폼을 만들고 post 형식으로 정보 넘기기

```
app.get('/login_page',function(req,res,next)
{
    var text = `
        <html>
        <head>

        </head>
        <body>
            <form action='/login' method='post'>
                <input type="text" name="id" value="" placeholder="id"><br>
                <input type="password" name="password" value="" placeholder="password"><br>
                <input type="submit">
            </form>
        </body>
        </html>
    `;
}
```



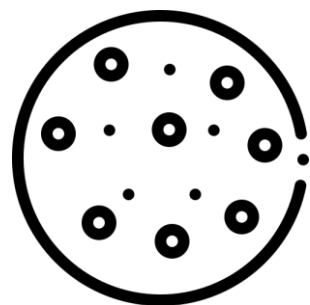
The screenshot shows a web browser window with the address bar displaying 'localhost:8001/login\_page'. The page content includes a form with two text input fields. The first field is labeled 'id' and the second is labeled 'password'. Below these fields is a button labeled '제출' (Submit).



로그인창 만들기

3. 정보 확인하기

```
app.post('/login', function (req, res) {  
    if(user.id == req.body.id && user.password == req.body.password){  
        req.session.is_logged = true;  
        res.send(`Success ${user.id}`);  
    }  
    else{  
        res.send('try again');  
    }  
});  
  
var user = {  
    id: 'abc',  
    password: '1234'  
};
```



**Q & A**

