쿠키와 세션

목차

HTTP의 특성

- 클라이언트와 서버 사이에서 HTTP통신을 할 때, 클라이언트의 상태를 저장하지 않는 Stateless, 응답 후 바로 연결을 종료하는 Connectionless 특성이 있다.
- 이에 따라 상태를 유지해야하는 기능은 별도의 방법이 필요함.

쿠키 (Cookie)

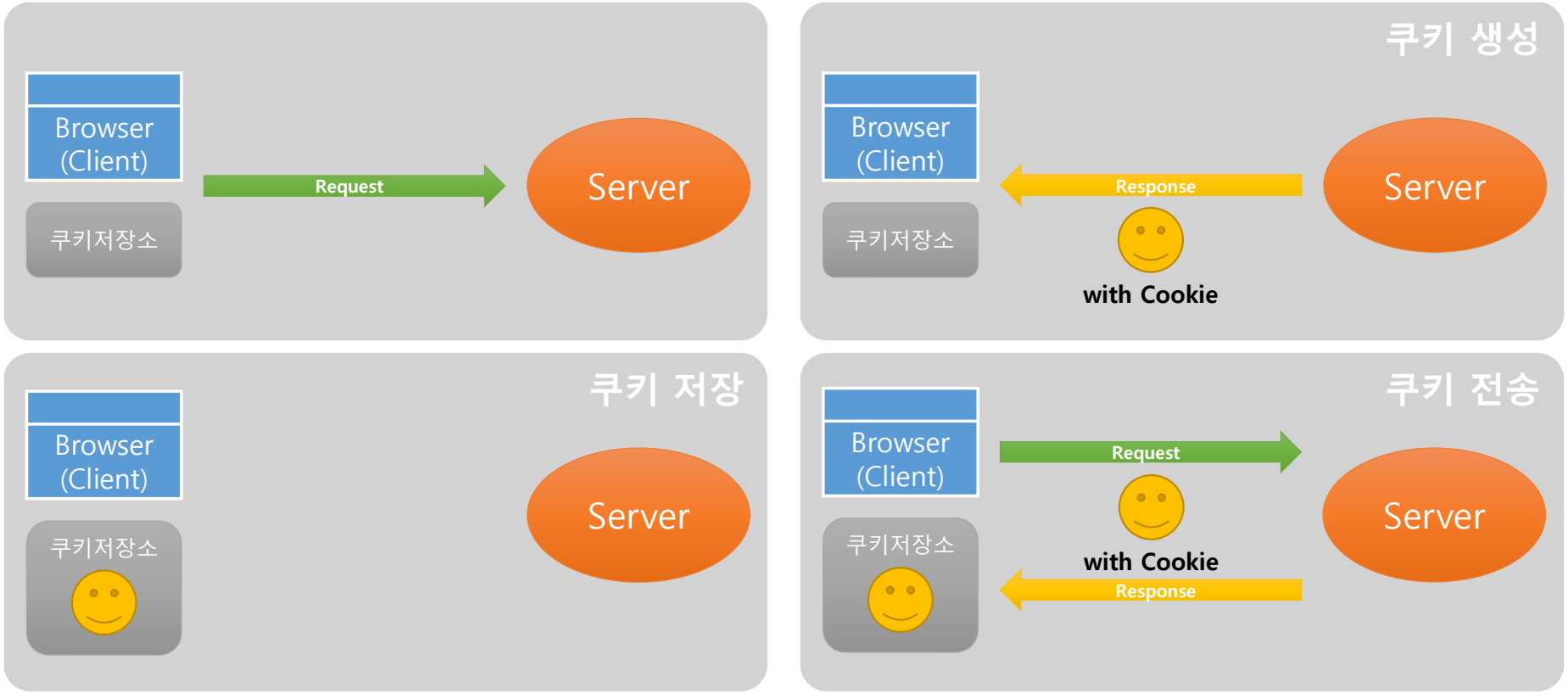
- **쿠키란? (Cookie)**

  Request와 Response에 쿠키 정보를 추가하여 클라이언트의 상태를 파악하기 위한 시스템으로, 클라이언트(브라우저) 로컬에 저장되는 데이터 파일

- **동작 방식**

  1. 클라이언트가 페이지를 요청한다. (사용자가 웹사이트 접근)
  2. 요청을 받은 웹 서버는 쿠키를 생성한다. – *쿠키 생성 단계*
  3. 생성한 쿠키에 정보를 담아 HTTP 화면을 돌려줄 때, 같이 클라이언트에게 돌려준다.
  4. 넘겨 받은 쿠키는 클라이언트가 가지고 있다가(로컬 PC에 저장)
     다시 서버에 요청할 때 요청과 함께 쿠키를 전송한다. – *쿠키 저장 단계, 쿠키 전송 단계*
  5. 동일 사이트 재방문시 클라이언트 PC에 해당 쿠키 존재 여부를 확인한 후,
     요청 페이지와 함께 쿠키를 전송한다. 쿠키가 삭제되기 전까지 웹 서버에 쿠키를 전송.

쿠키(Cookie)



<동작 방식>

쿠키 (Cookie)

- **구성**
  - **이름** : 각각의 쿠키를 구별하는 데 사용되는 이름
  - **값** : 쿠키의 이름과 관련된 값
  - **유효시간** : 쿠키의 유지시간
  - **도메인** : 쿠키를 전송할 도메인
  - **경로** : 쿠키를 전송할 요청 경로
- **예시**
  - 방문했던 사이트에 다시 방문 하였을 때 아이디와 비밀번호 자동 입력
  - 팝업창을 통해 "오늘 이 창을 다시 보지 않기" 체크 등
- **단점 및 한계**
  - 쿠키의 값은 임의변경이 가능하다.
  - 데이터 파일이 웹 브라우저에 저장되고, 요청마다 매번 클라이언트와 서버에서 주고받기 때문에 보안에 취약하다. (탈취 가능)

세션 (Session)

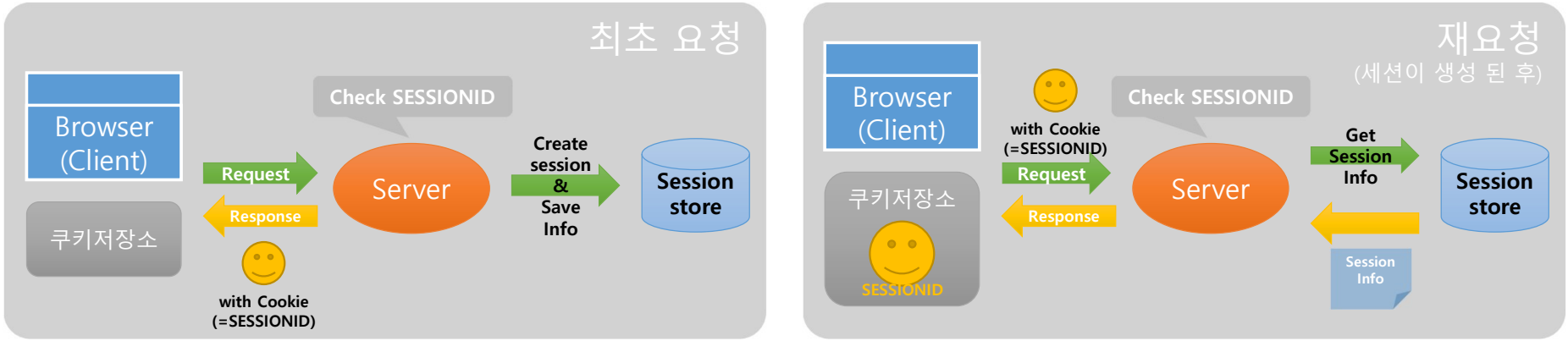
- **세션이란?**

  쿠키를 기반으로 하지만, 클라이언트의 정보를 서버에 따로 저장하는 방식. 쿠키는 쿠키 데이터 값을 통해 클라이언트의 상태를 기억한다면, 세션은 서버가 클라이언트에 부여한 세션 ID를 통해 클라이언트 상태를 기억하게 한다. 이 세션 ID는 쿠키형태로 관리한다.

- **세션 동작 방법**

  1. 클라이언트가 페이지를 요청한다. (사용자가 웹사이트 접근)
  2. 서버는 접근한 클라이언트의 요청(Request)의 Header 필드 중 Cookie를 확인하여, 클라이언트의 SESSIONID가 있는지 확인한다.
  3. SESSIONID 가 존재하지 않는다면, 서버는 SESSIONID 를 생성한 후 세션 저장소에 SESSIONID 에 따른 세션 정보를 저장하고, 클라이언트에 SESSIONID를 보낸다.
  4. 서버에서 클라이언트로 돌려준 SESSIONID 를 쿠키를 사용해 로컬에 저장한다.
  5. 클라이언트는 재접속 시, 이 쿠키를 이용하여 SESSIONID 값을 서버에 전달한다.

세션 (Session)



<동작 방식>

세션 (Session)

- **예시**
  - 화면을 이동해도 로그인이 풀리지 않고 로그아웃 전까지 유지

- **단점 및 한계**
  - 쿠키보다 속도가 느림
  - 사용자가 많아질수록 서버 메모리를 많이 차지하게 됨

  (새로운 인증방식의 등장 – JWT 등의 토큰 방식 등)

쿠키 vs 세션

구분	Cookie	Session
저장 위치	Client	Server
저장 형식	Text	Object
종료 시점	쿠키 생성 시 설정 (미설정 시 브라우저 종료 시)	정확한 시점을 파악하기 힘듦 (Java: HttpSessionListner를 이용)
자원	Client Resource	Server Resource
제한	도메인 당 20개, 쿠키 당 4KB, 한 클라이언트 당 300개	제한 없음

- 쿠키는 로컬에서 관리하며 정보를 서버에 바로 전달하므로 세션에 비해 속도가 빠르다는 장점과 함께, 보안성이 떨어진다는 단점이 있다.
- 세션은 보안성이라는 장점과 함께 세션 ID를 통해 서버에 있는 정보를 가져와야 해서 비교적 느리고, 사용자가 늘어날수록 세션을 관리하는 서버의 부담이 증가한다는 단점이 있다.
- 따라서 용도에 따라 쿠키와 세션을 병행하여 사용한다.