spring

20강_세션, 쿠키

클라이언트와 서버의 연결을 유지하는 방법을 학습합니다.

20-1 세션(Session)과 쿠키(Cookie)
20-2 HttpServletRequest를 이용한 세션 사용
20-3 HttpSession을 이용한 세션 사용
20-4 세션 삭제
20-5 세션 주요 메소드 및 플로어
20-6 쿠키(Cookie)

2018. 01 / blogstudy@naver.com

20-1 : 세션(Session)과 쿠키(Cookie)

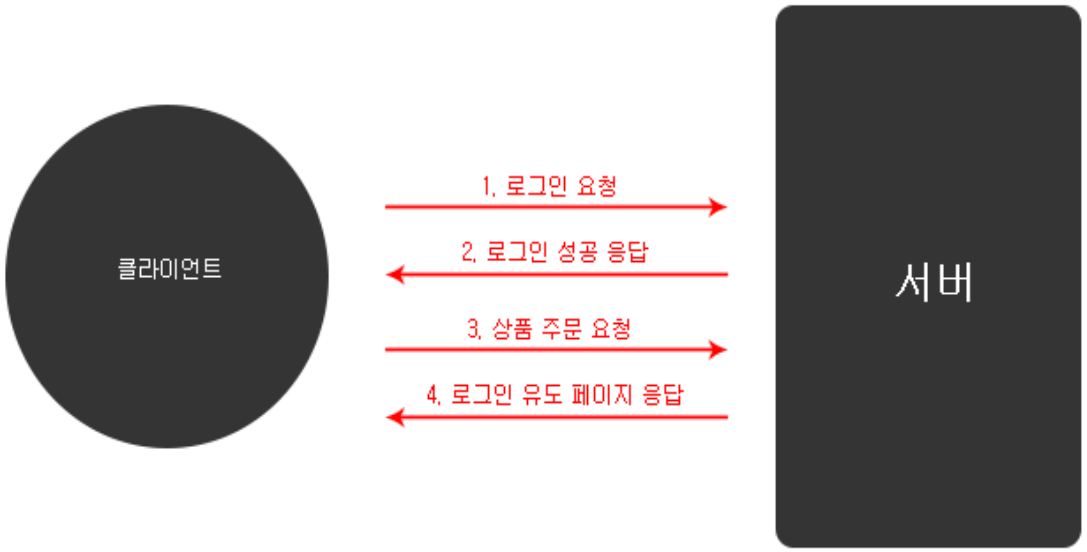
Connectionless Protocol

웹 서비스는 HTTP 프로토콜을 기반으로 하는데, HTTP 프로토콜은 클라이언트와 서버의 관계를 유지 하지 않는 특징이 있다.

20-1 : 세션(Session)과 쿠키(Cookie)

Connectionless Protocol

서버의 부하를 줄일 수 있는 장점은 있나, 클라이언트의 요청 시마다 서버와 매번 새로운 연결이 생성되기 때문에 일반적인 로그인 상태 유지, 장바구니 등의 기능을 구현하기 어렵다.

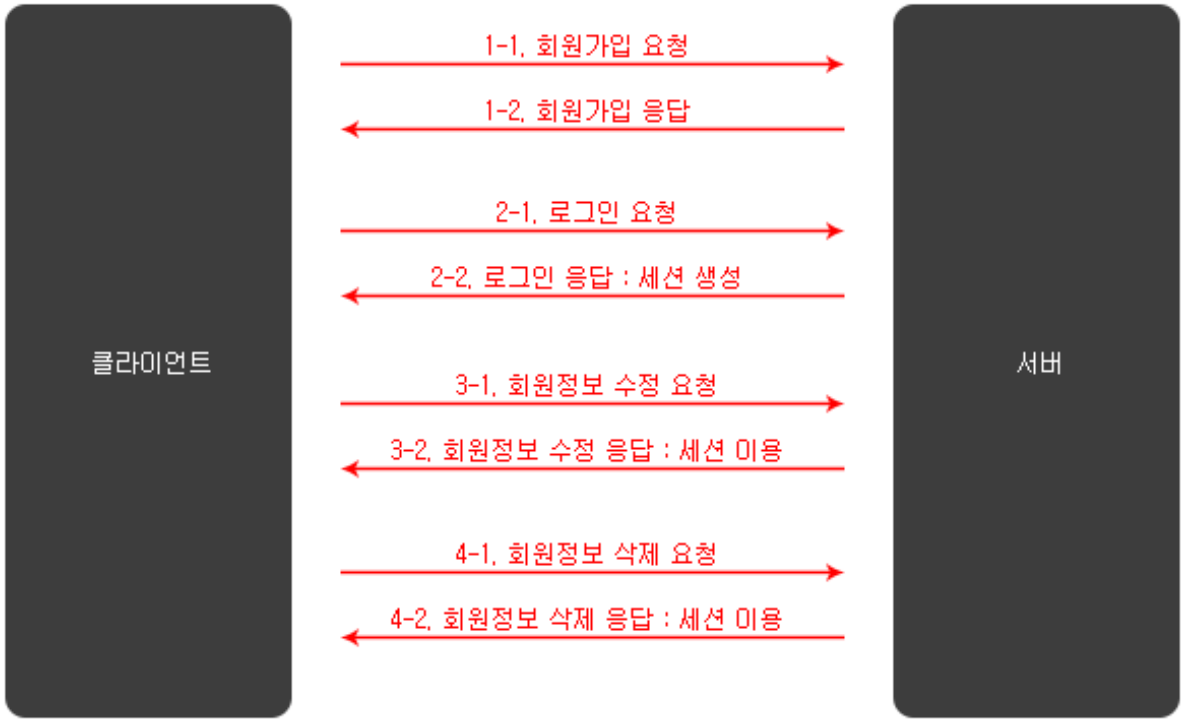


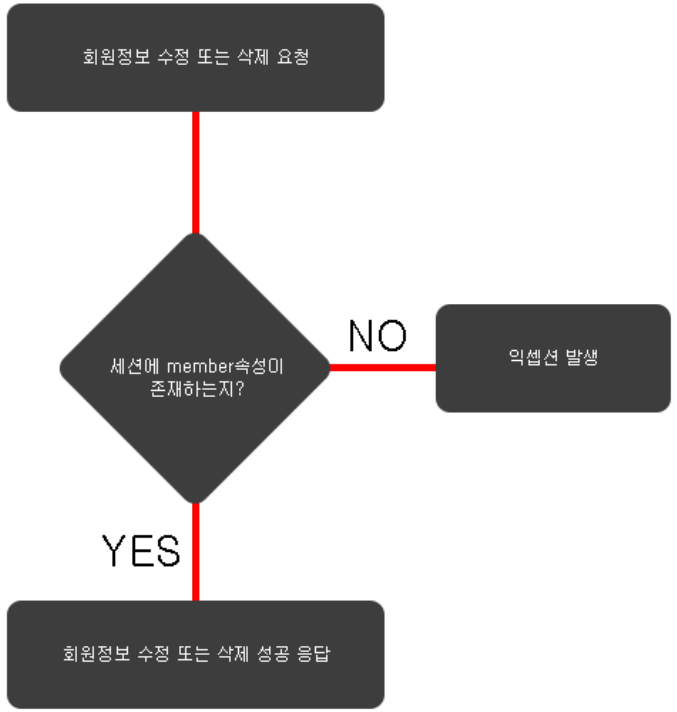
이러한 Connectionless Protocol의 불편함을 해결하기 위해서 세션과 쿠키를 이용한다.

세션과 쿠키는 클라이언트와 서버의 연결 상태를 유지해주는 방법으로, 세션은 서버에서 연결 정보를 관리하는 반면 쿠키는 클라이언트에서 연결 정보를 관리하는데 차이가 있다.

20-2 : HttpServletRequest를 이용한 세션 사용

예제: lec20Pjt001





스프링 MVC에서 HttpServletRequest를 이용해서 세션을 이용하려면 컨트롤러의 메소드에서 파라미터로 HttpServletRequest를 받으면 된다

20-3 : HttpSession을 이용한 세션 사용

HttpServletRequest와 HttpSession의 차이점은 거의 없으며, 단지 세션객체를 얻는 방법에 차이가 있을 뿐이다

HttpServletRequest
파라미터로 HttpServletRequest를 받은 후 getSession()으로 세션 얻음.

HttpSession
파라미터로 HttpSession을 받아 세션 사용.

```
public String memLogin(Member member, HttpServletRequest request) {…}

public String memLogin(Member member, HttpSession session) {…}
```

20-4 : 세션 삭제

세션을 삭제하는 방법은 세션에 저장된 속성이 더 이상 필요 없을 때 이루어지는 과정으로 주로 로그아웃 또는 회원 탈퇴 등에 사용된다.

로그아웃

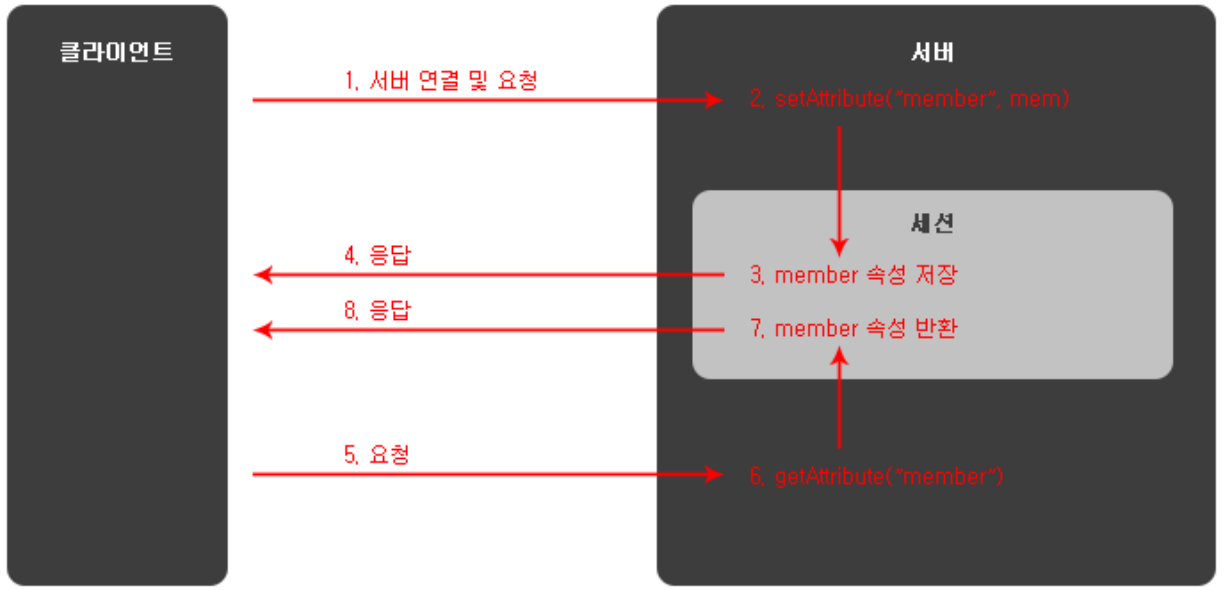
```
@RequestMapping("/logout")
public String memLogout(Member member, HttpSession session) {

 session.invalidate();

 return "/member/logoutOk";
}
```

회원탈퇴

```
@RequestMapping(value = "/remove", method = RequestMethod.POST)
public String memRemove(Member member, HttpServletRequest request) {

service.memberRemove(member);

HttpSession session = request.getSession();
session.invalidate();

return "/member/removeOk";
}
```

20-5 : 세션 주요 메소드 및 플로어

세션 메소드	기능
getId()	세션 ID를 반환한다.
setAttribute()	세션 객체에 속성을 저장한다.
getAttribute()	세션 객체에 저장된 속성을 반환한다.
removeAttribute()	세션 객체에 저장된 속성을 제거한다.
setMaxInactiveInterval()	세션 객체의 유지시간을 설정한다.
getMaxInactiveInterval()	세션 객체의 유지시간을 반환한다.
invalidate()	세션 객체의 모든 정보를 삭제한다.



20-6 : 쿠키(Cookie)

예제: lec20Pjt002

- mallMain()에서 쿠키를 생성하고, 파라미터로 받은 HttpServletResponse에 쿠키를 담고 있다.
- 쿠키를 생성할 때는 생성자에 두 개의 파라미터를 넣어주는데 첫 번째는 쿠키이름을 넣어 주고 두 번째는 쿠키값을 넣어 준다.

```
@RequestMapping("/main")
public String mallMain(Mall mall, HttpServletResponse response) {

  Cookie genderCookie = new Cookie("gender", mall.getGender());

  if(mall.isCookieDel()) {
   genderCookie.setMaxAge(0);
   mall.setGender(null);
  } else {
   genderCookie.setMaxAge(60*60*24*30);
  }
   response.addCookie(genderCookie);

  return "/mall/main";
}
```

20-6 : 쿠키(Cookie)

예제: lec20Pjt002

- mallMain()에서 생성된 쿠키를 mallIndex()에서 사용한다.
- 쿠키를 사용할 때는 @CookieValue 를 사용한다.

```
@RequestMapping("/index")
public String mallIndex(Mall mall,
@CookieValue(value="gender", required=false) Cookie genderCookie,
HttpServletRequest request) {

 if(genderCookie != null)
  mall.setGender(genderCookie.getValue());

 return "/mall/index";
}
```

@CookieValue 어노테이션의 value 속성은 쿠키 이름을 나타내는데, 만약 value에 명시한 쿠키가 없을 경우 익셉션이 발생한다.
익셉션을 막는 방법이 있는데, 바로 required 속성이다. Required 속성은 기본값으로 true를 가지고 있는데 required가 true인 경우 value값에 해당하는 쿠키가 없으면 익셉션이 발생한다. 따라서 required 속성값을 false로 설정해서 value값에 해당하는 쿠키가 없어도 익셉션이 발생하지 않도록 한다.