

JSP

-쿠키 세션

- 1.쿠키
- 2.세션
- 3.application 내장객체

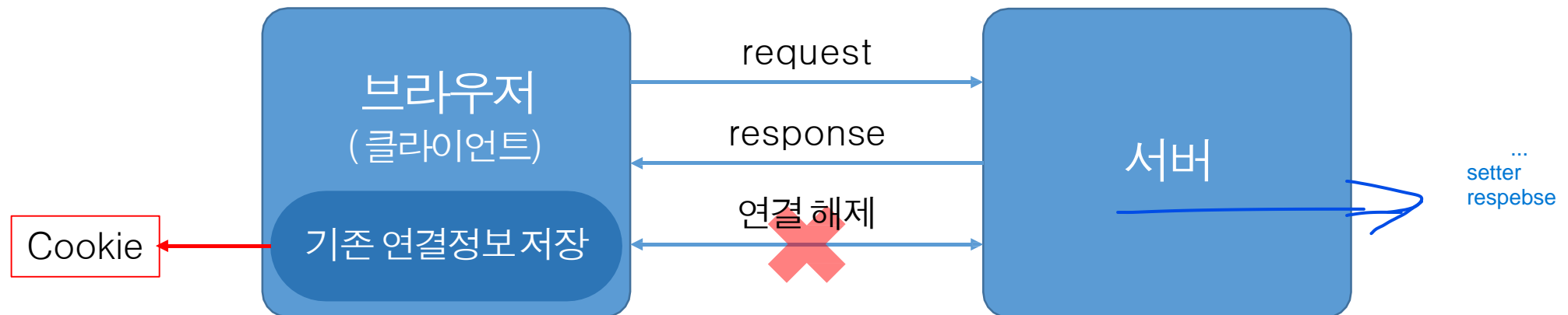
ex)

..

* 쿠키(Cookie)

- 웹 브라우저에서 서버로 어떤 데이터를 요청하면, 서버측에서는 알맞은 로직을 수행한 후 데이터를 웹 브라우저에 응답합니다.
- 그리고 Http 프로토콜은 응답 후에 웹 브라우저와의 **관계를 종료**합니다.
- 연결이 끊겼을 때, 어떤 정보를 **지속적으로 유지하기 위한 수단**으로 쿠키라는 방식을 사용합니다.
- 쿠키는 서버에서 생성하여, 서버가 아닌 **클라이언트측(local)**에 정보를 **저장**합니다.
- 서버에서 요청할 때마다 쿠키의 속성값을 참조 또는 변경할 수 있습니다.
- 쿠키는 개당 4kb로 용량이 제한적이며, 300개까지(1.2MB) 데이터 정보를 가질 수 있습니다.

- 쿠키문법: 쿠키클래스에서 쿠키 생성 -> setter메서드로 쿠키의 속성 설정 -> response객체에 쿠키 탑재 -> 로컬 환경에 저장

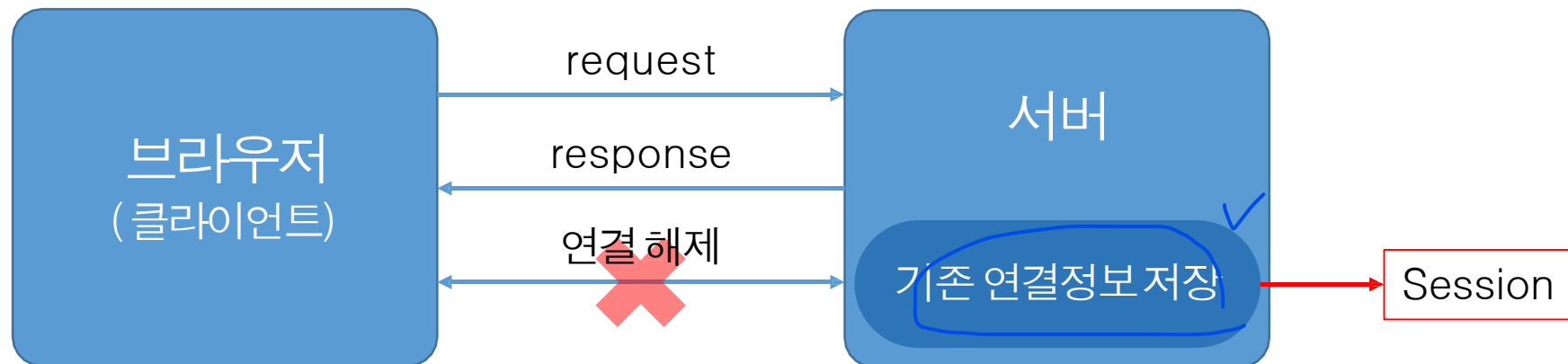


- Cookie 객체 관련 메서드

메서드	기능
1. <code>setMaxAge()</code>	쿠키의 유효시간을 설정합니다.
2. <code>setPath()</code>	쿠키사용의 유효디렉토리를 설정합니다.
3. <code>setValue()</code>	쿠키의 값을 설정합니다.
4. <code>setVersion()</code>	쿠키 버전을 설정합니다.
5. <code>getMaxAge()</code>	쿠키 유효기간 정보를 얻습니다.
6. <code>getName()</code>	쿠키의 이름을 얻습니다.
7. <code>getPath()</code>	쿠키사용의 유효디렉토리 정보를 얻습니다.
8. <code>getValue()</code>	쿠키의 값을 얻습니다.
9. <code>getVersion()</code>	쿠키 버전을 얻습니다.

* 세션(Session)

- 세션도 쿠키와 마찬가지로 서버와의 관계를 유지하기 위한 수단입니다.
- 단, 쿠키와 달리 클라이언트의 특정 위치에 저장되는 것이 아니라, 서버상에 객체형태로 존재합니다.
- 서버당 하나의 세션 객체를 가질 수 있습니다. (브라우저 별 서로 다른 세션 사용)
- 세션 객체는 브라우저 창을 종료하면 삭제됩니다.
- 따라서 세션은 서버에서만 접근이 가능하여 보안이 좋고, 저장할 수 있는 데이터에 한계가 없습니다.
- 세션은 클라이언트의 요청이 발생하면 자동생성되어 고유한 ID값을 클라이언트에 넘겨주며 이것은 쿠키에 저장됩니다.
- JSP에서는 session이라는 내장 객체를 지원하여 세션의 속성을 설정할 수 있습니다.



- session 객체 관련 메서드

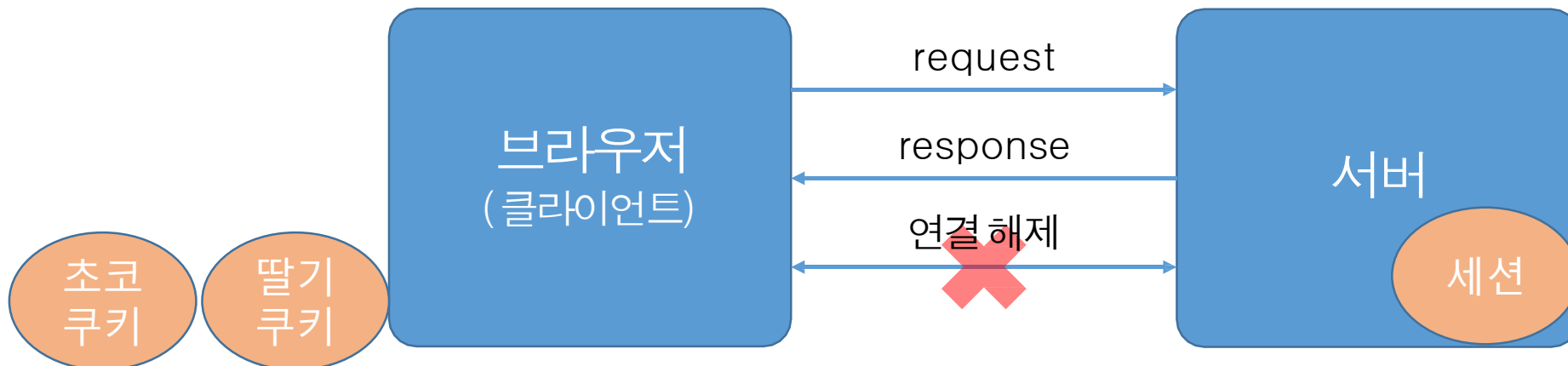
? 30 or

메서드	기능
1. <code>setAttribute()</code>	세션에 <u>데이터를 저장</u> 합니다.
2. <code>getAttribute()</code>	세션에 저장되어 있는 <u>데이터를 얻</u> 습니다.
3. <code>getAttributeNames()</code>	세션에 저장되어 있는 모든 데이터의 세션 이름(key)을 얻습니다.
4. <code>getId()</code>	자동생성된 세션의 유니크한 아이디를 얻습니다.
5. <code>getCreationTime()</code>	세션이 생성된 시간을 구합니다.
6. <code>getLastAccessedTime()</code>	웹 브라우저가 가장 마지막에 세션에 접근한 시간을 구합니다.
7. <code>setMaxInactiveInterval()</code>	세션의 유효시간을 설정합니다. 초 단위로 기록합니다.
8. <code>getMaxInactiveInterval()</code>	세션의 유효시간을 얻습니다. 가장 최근 요청시점을 기준으로 카운트됩니다.
9. <code>removeAttribute()</code>	<u>특정 세션을 삭제</u> 합니다.
10. <code>invalidate()</code>	<u>모든 세션을 삭제</u> 합니다.

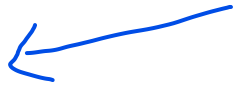
* 쿠키 vs 세션

=>

- 쿠키 대신에 세션을 사용하는 가장 큰 이유는 **세션이 쿠키보다 보안에서 앞서기 때문**입니다.
 - 쿠키의 이름이나 데이터는 네트워크를 통해 전달되기 때문에 HTTP 프로토콜을 사용하는 경우 중간에서 누군가가 쿠키의 값을 읽어올 수 있습니다.
 - 그러나 세션은 오직 서버에만 저장되기 때문에 중요한 데이터를 저장하기에 좋습니다.
 - 세션을 사용하는 또 다른 이유는 웹 브라우저가 쿠키를 지원하지 않거나 강제로 사용자가 쿠키를 차단한 경우에도 사용할 수 있다는 점입니다.
 - **세션은 여러 서버에서 공유할 수 없는 단점**이 있습니다. 그러나 쿠키는 도메인을 이용해 **쿠키를 여러 도메인에서 공유할 수 있기 때문**에 Naver, Daum과 같은 포털사이트들은 쿠키에 로그인 방식을 저장하는 것을 선호합니다.
- ex) ww.naver.com과 mail.naver.com, blog.naver.com의 서버는 각각 다름.



* application 기본 객체



- 특정 웹 어플리케이션에 포함된 모든 JSP페이지는 하나의 application 기본 객체를 공유합니다.
- application 객체는 웹 어플리케이션 전반에 걸쳐서 사용되는 정보를 담고 있습니다.

* 생명주기

- request 객체는 요청영역마다 생성되고,
- session 객체는 브라우저별로 생성되고,
- application은 프로그램 전체에서 딱 한번 최초 가동시 생성됩니다.

가
가

??

/개

