http://www.lycato.com

Scopes & Cookie

Nguyễn Nghiệm

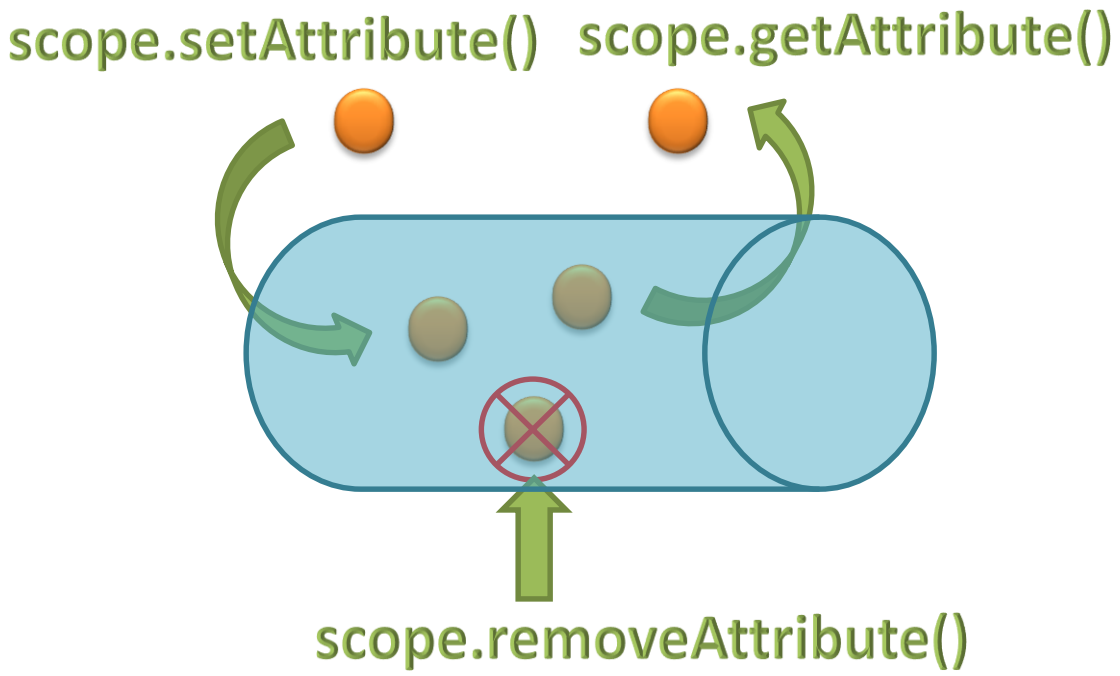
Agenda

- request: HttpServletRequest
- session: HttpSesssion
- application: ServletContext
- Cookie

Scopes

PHẠM VI CHIA SẺ DỮ LIỆU (SCOPE)

- Trong Spring Boot, ngoài **Model** (chia sẻ dữ liệu giữa Controller và View) còn có 3 scope khác:
  - **HttpServletRequest** (gọi là ***request***)
  - **HttpSession** (gọi là ***session***)
  - **ServletContext** (gọi là ***application***)
- Scope API
  - **<scope>.setAttribute**(name, value)
    - *Thêm mới hoặc thay thế một attribute*
  - **<scope>.getAttribute**(name): Object
    - *Lấy giá trị của một attribute*
  - **<scope>.removeAttribute**(name)
    - *Xóa một attribute*



THAM CHIẾU ĐẾN CÁC SCOPE

- Sử dụng *@Autowired* để "tiêm" Scope vào Component

```
@Controller
public class HomeController {

    @Autowired ServletContext application;
    @Autowired HttpSession session;
    @Autowired HttpServletRequest request;

    @RequestMapping("/url")
    public String share( HttpSession session, HttpServletRequest request ) {
        // lập trình thao tác scopes
        …
    }
}
```

1. Tiêm vào các fields

${#servletContext} | ${application}
${#httpSession} | ${session}
${#request}

Thymeleaf

2. Tiêm vào đối số của Action Method.

Chỉ áp dụng cho session và request

Scopes Example

```java
@Controller
public class ScopeController {
    @Autowired HttpServletRequest request;
    @Autowired HttpSession session;
    @Autowired ServletContext application;
    @RequestMapping("/scope/demo")
    public String demo() {
        request.setAttribute("r", "Request");
        session.setAttribute("s", "Session");
        application.setAttribute("a", "Application");
        return "scope/demo";
    }
}
```

.../scope/demo

scope/demo.html

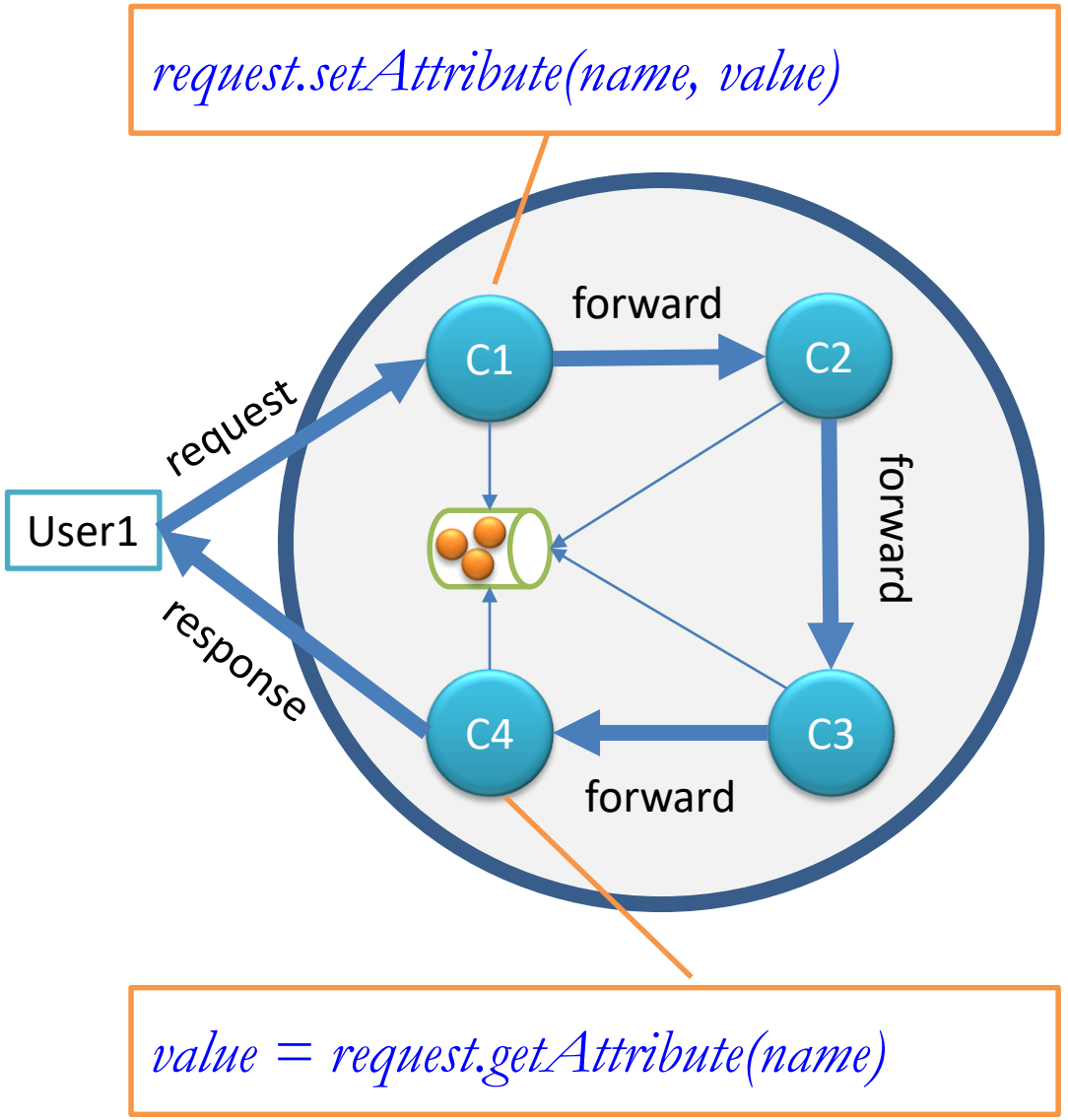
```
<ul>
    <li th:text="${#request.getAttribute('r')}"></li>
    <li th:text="${r}"></li>
    <li th:text="${#httpSession.getAttribute('s')}"></li>
    <li th:text="${session.s}"></li>
    <li th:text="${#servletContext.getAttribute('a')}"></li>
    <li th:text="${application.a}"></li>
</ul>
```

Response

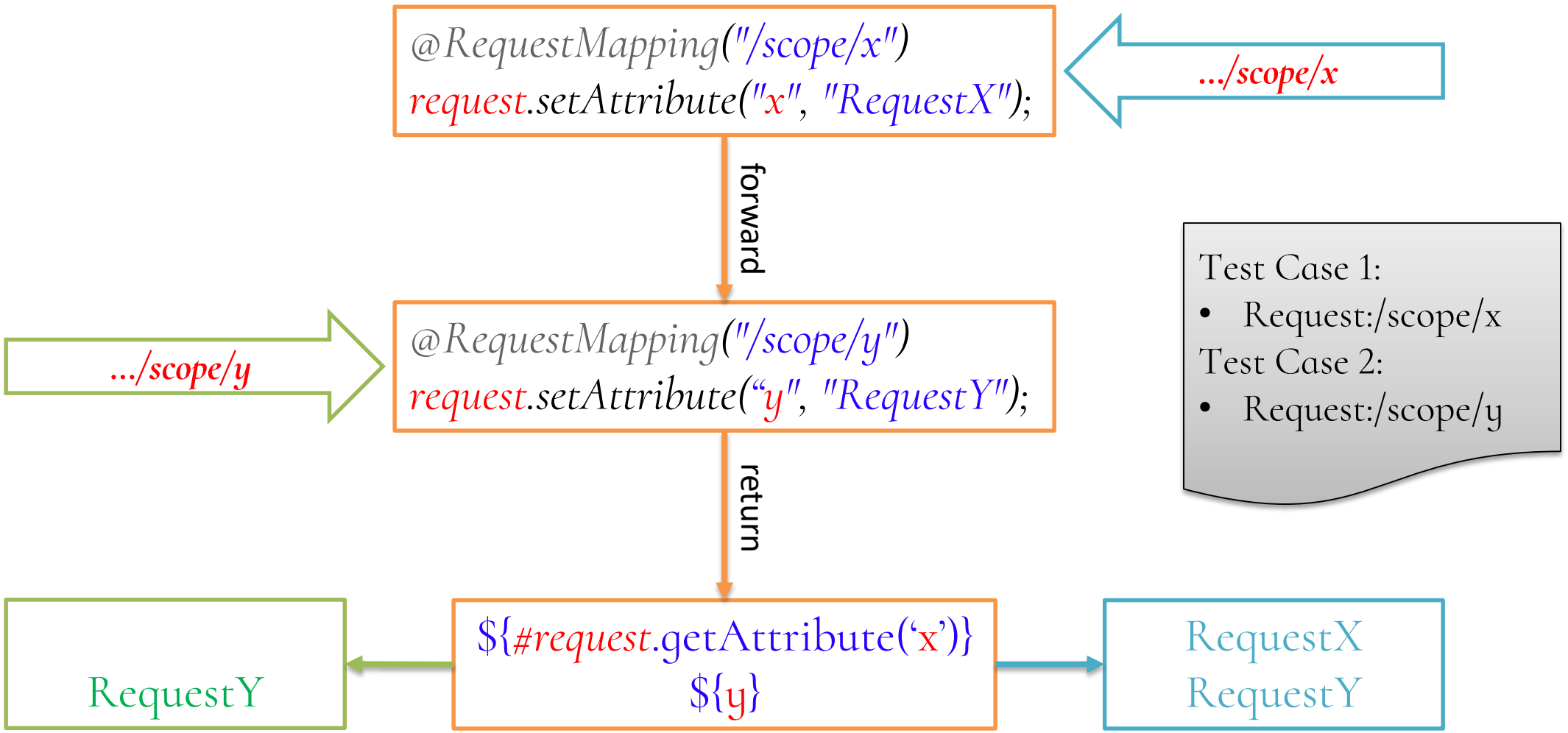
```
<ul>
    <li>Request</li>
    <li>Request</li>
    <li>Session</li>
    <li>Session</li>
    <li>Application</li>
    <li>Application</li>
</ul>
```

REQUEST SCOPE

- Dữ liệu trong ***request*** được sử dụng để chia sẻ giữa các Component hoạt động trên *cùng một request*
  - Các component đứng trước trong chuỗi dây chuyền forward có thể sử dụng *request.setAttribute(name, value)* để tạo attribute và các component đứng sau có thể sử dụng *value = request.getAttribute(name)* để đọc giá trị của attribute
  - Request sẽ được giải phóng khỏi bộ nhớ sau khi request kết thúc

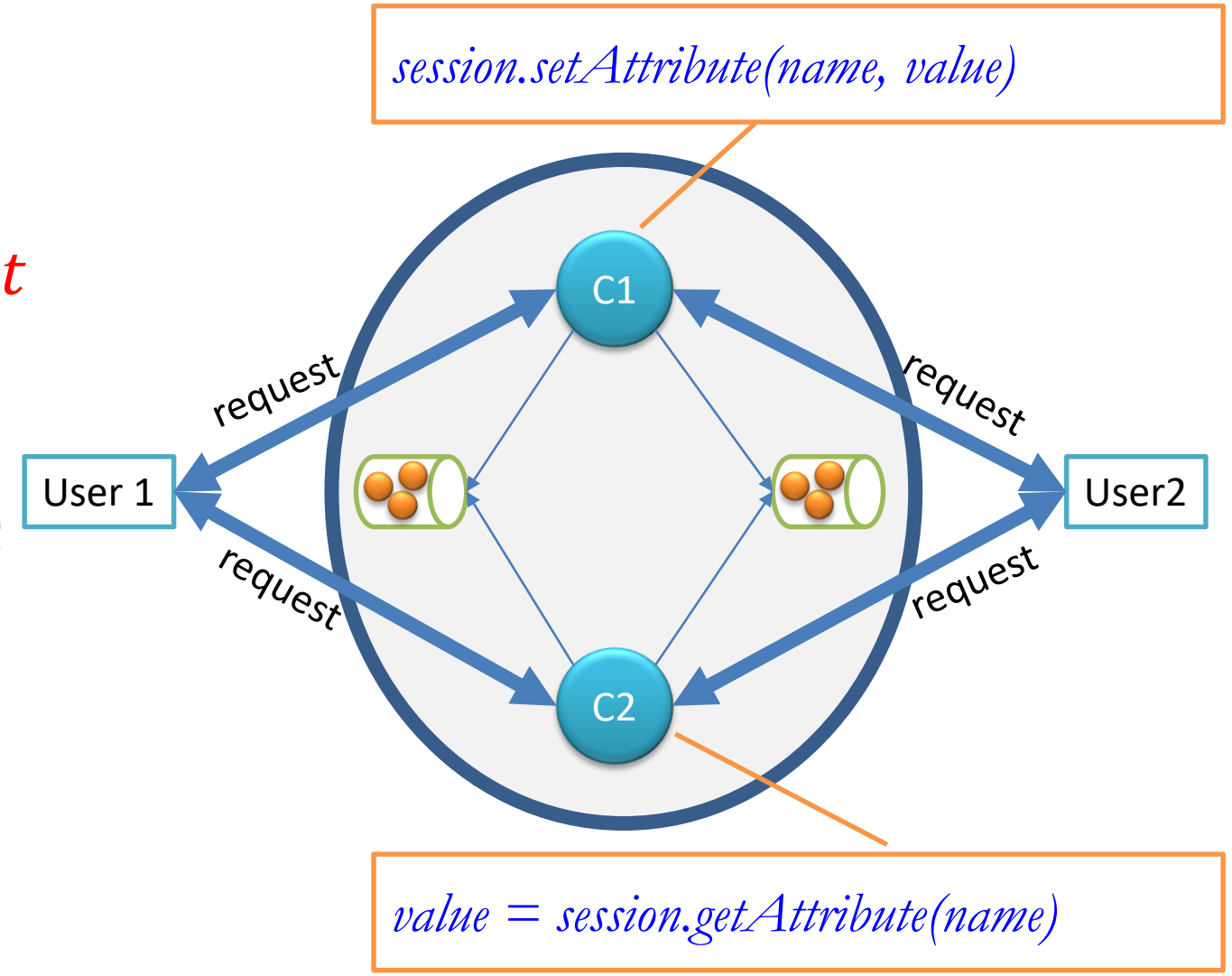


REQUEST SCOPE

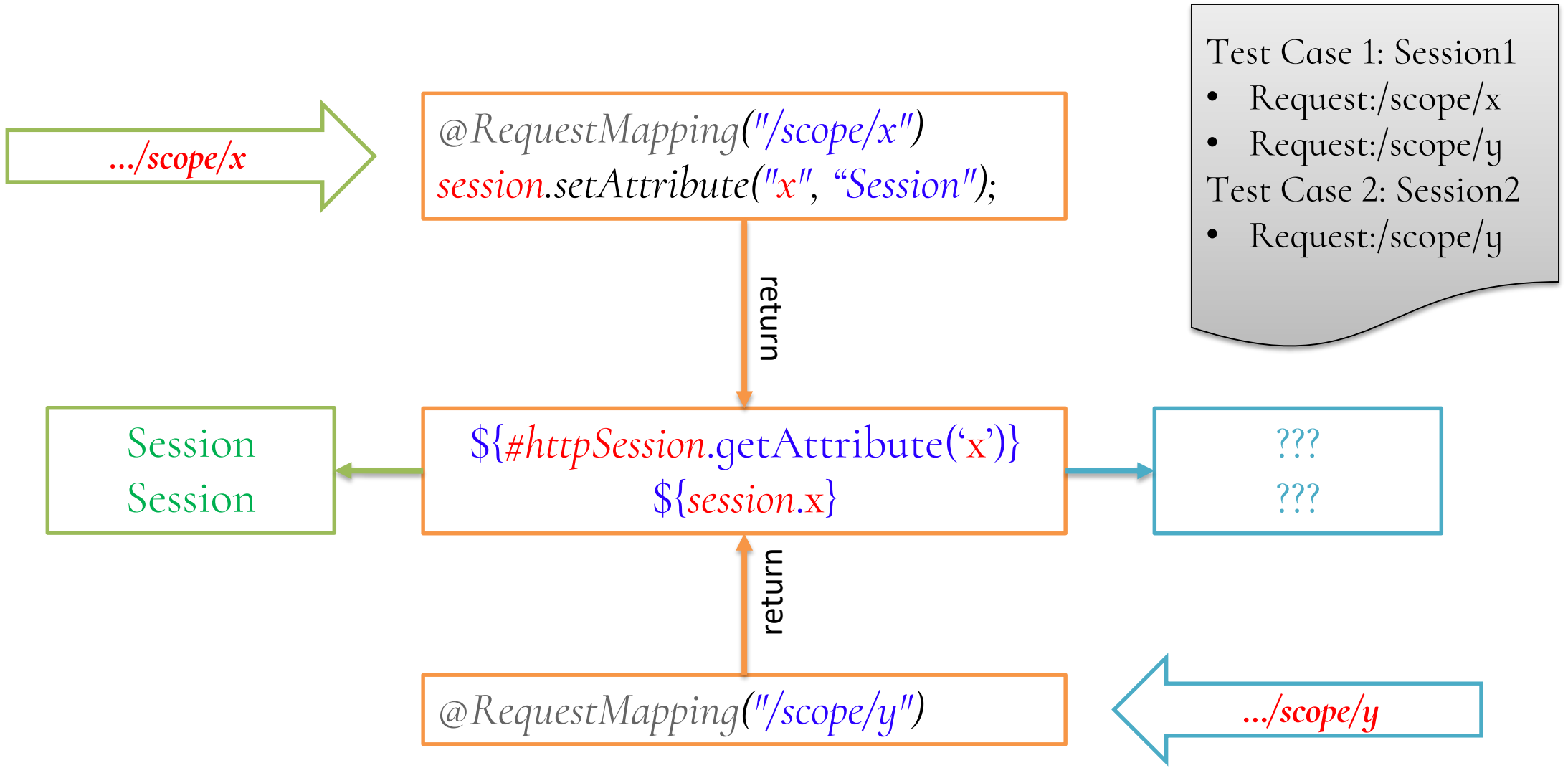


SESSION SCOPE

- ***Session*** là nơi lưu trữ dữ liệu theo phiên làm việc, có hiệu lực từ lúc kết nối đến website cho đến lúc *timeout* (sau một khoảng thời gian không truy xuất website)
- Dữ liệu trong ***session*** được sử dụng để chia sẻ giữa các Component hoạt động trên *cùng một phiên làm việc*

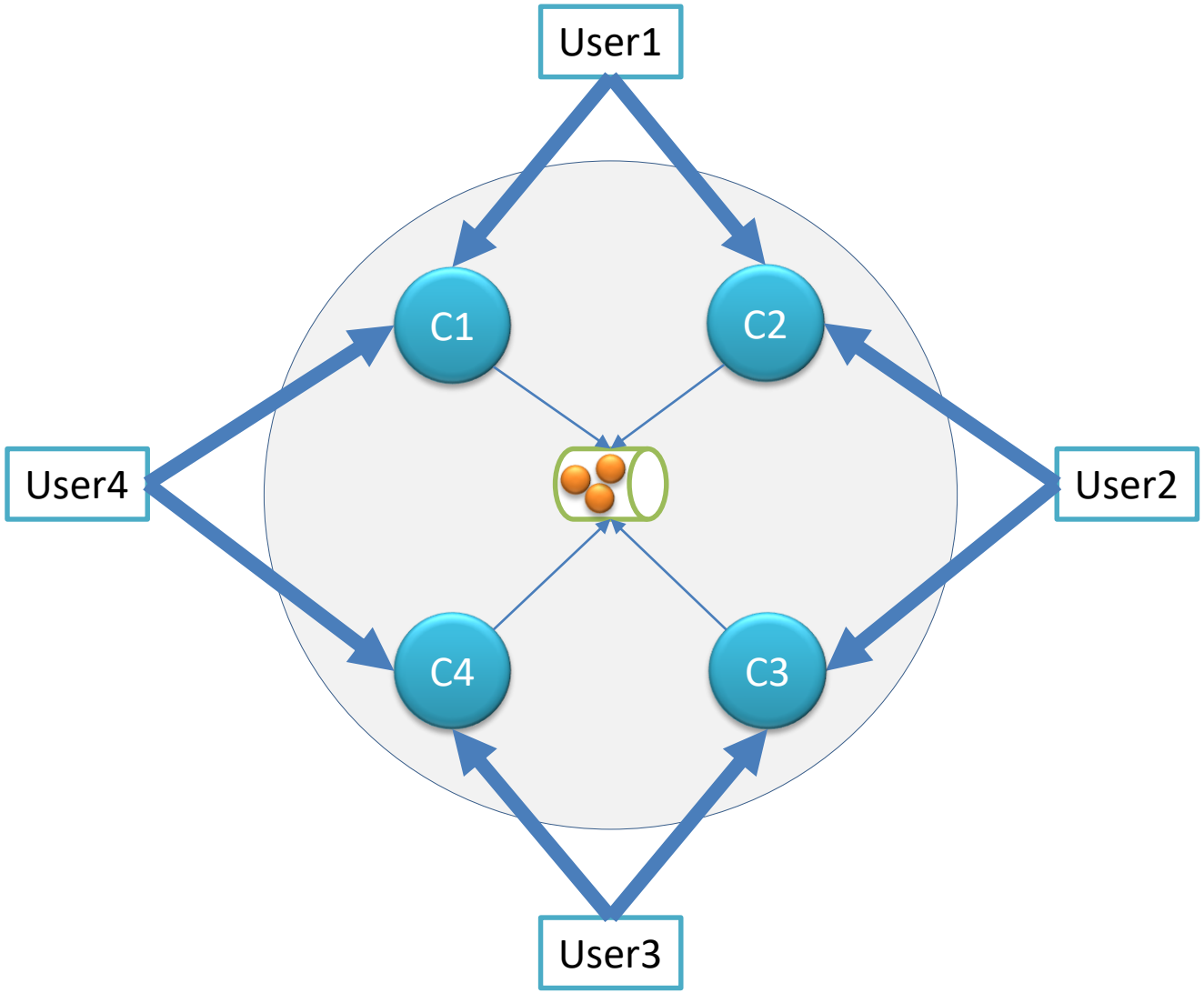


SESSION SCOPE

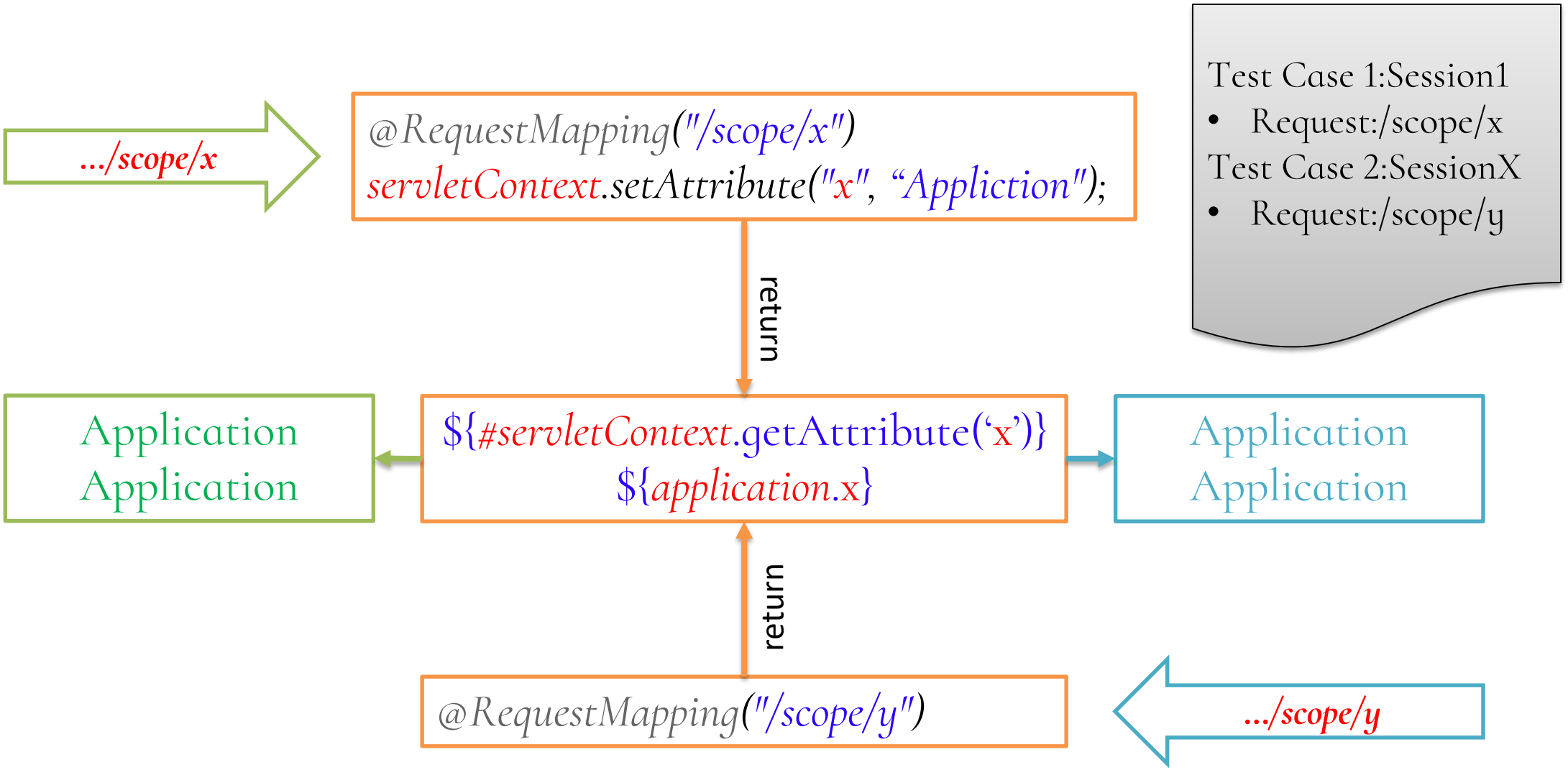


APPLICATION SCOPE

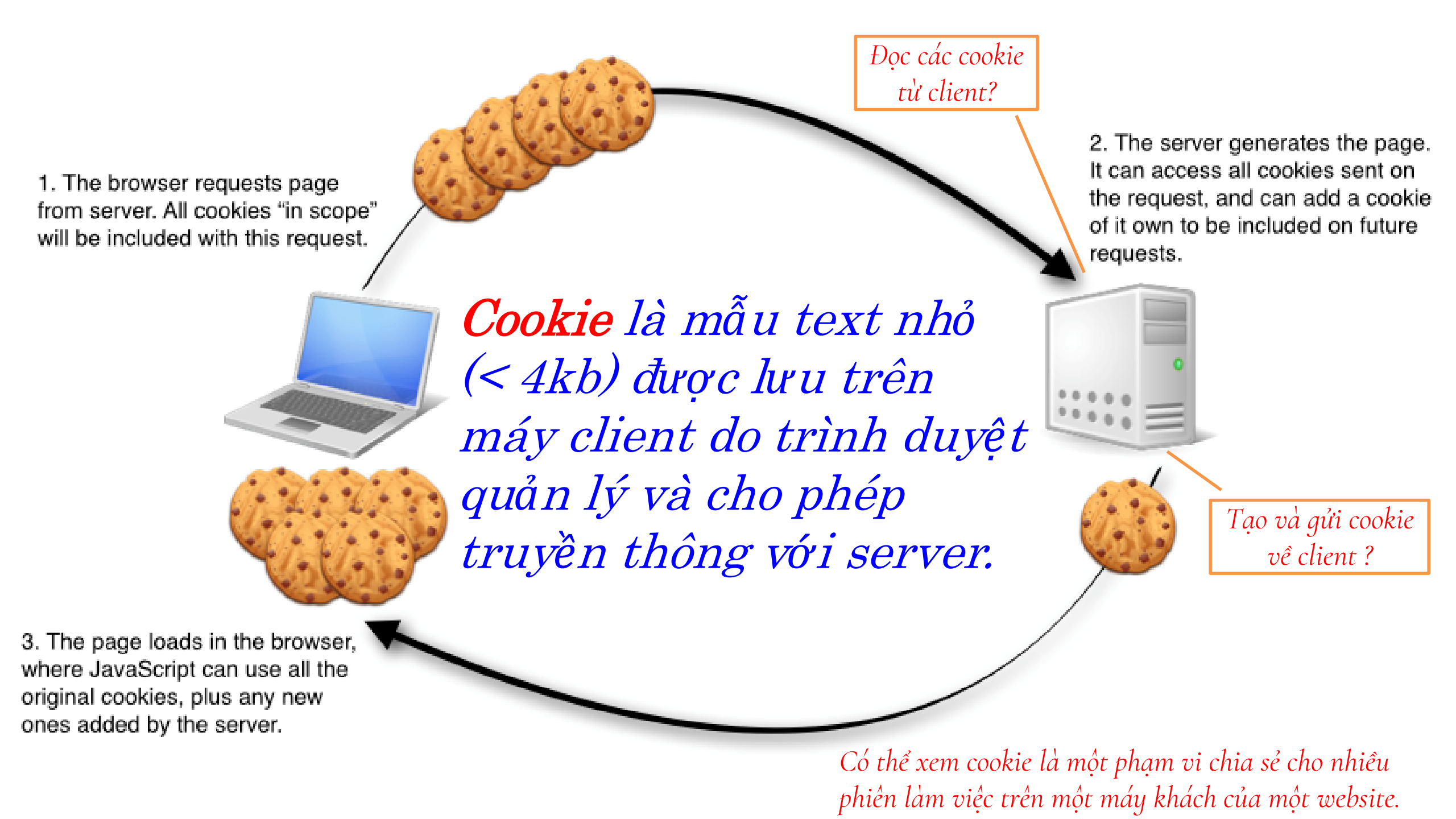
- ***Application*** là nơi lưu trữ dữ liệu dung chung cho các hoạt động trên toàn ứng dụng web (bất kỳ component nào cũng có thể tạo và đọc attribute)
  - *application.setAttribute(name, value)*
  - *value = application.getAttribute(name)*



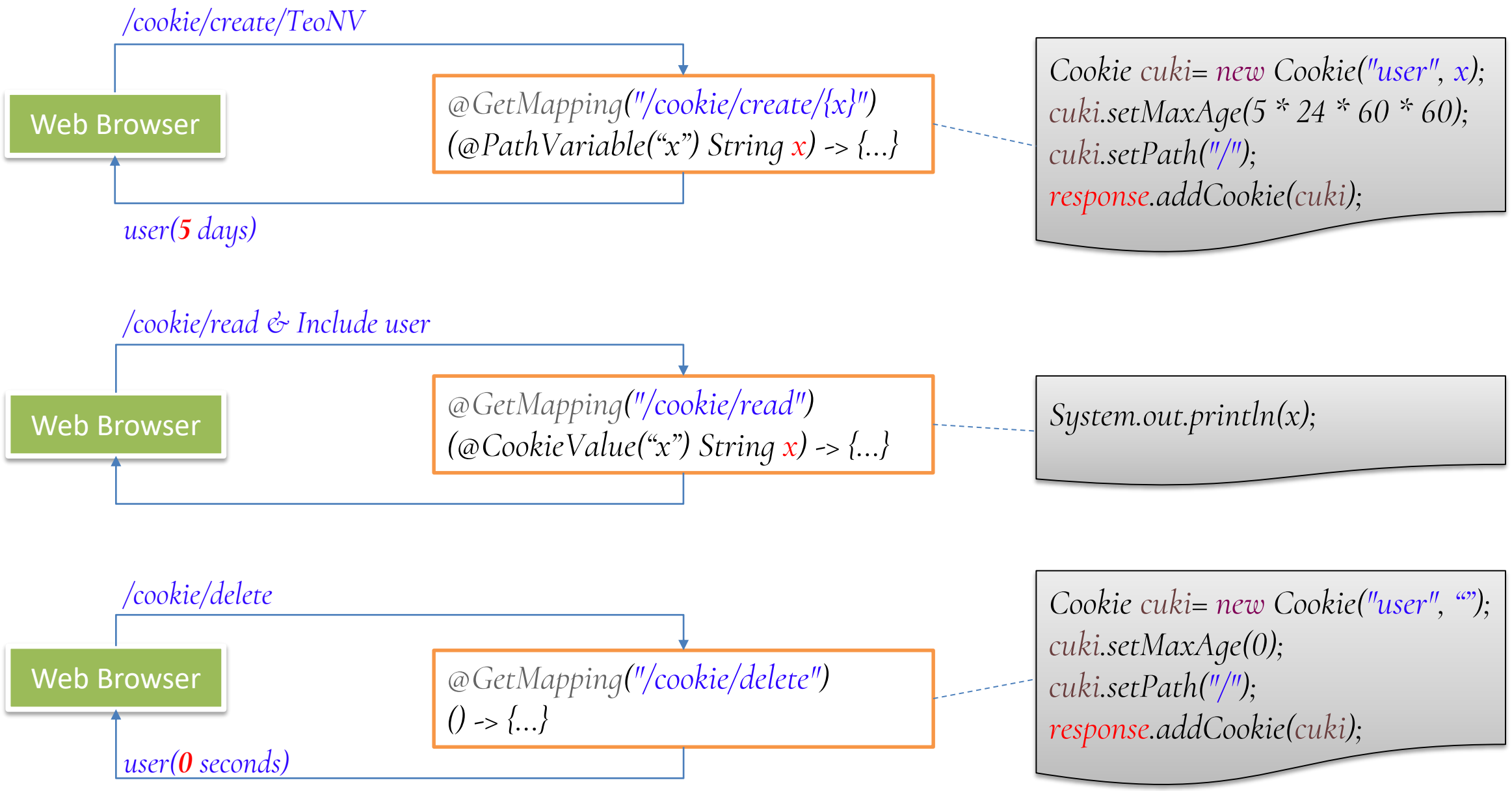
Application Scope



Cookie


Đọc các cookie từ client?
1. The browser requests page from server. All cookies "in scope" will be included with this request.
2. The server generates the page. It can access all cookies sent on the request, and can add a cookie of it own to be included on future requests.
Cookie là mẫu text nhỏ (< 4kb) được lưu trên máy client do trình duyệt quản lý và cho phép truyền thông với server.
Tạo và gửi cookie về client ?
3. The page loads in the browser, where JavaScript can use all the original cookies, plus any new ones added by the server.
Có thể xem cookie là một phạm vi chia sẻ cho nhiều phiên làm việc trên một máy khách của một website.

COOKIE


/cookie/create/TeoNV
Web Browser
@GetMapping("/cookie/create/{x}")
(@PathVariable("x") String x) -> {...}
user(5 days)
Cookie cuki= new Cookie("user", x);
cuki.setMaxAge(5 * 24 * 60 * 60);
cuki.setPath("/");
response.addCookie(cuki);
/cookie/read & Include user
Web Browser
@GetMapping("/cookie/read")
(@CookieValue("x") String x) -> {...}
System.out.println(x);
/cookie/delete
Web Browser
@GetMapping("/cookie/delete")
() -> {...}
user(0 seconds)
Cookie cuki= new Cookie("user", "");
cuki.setMaxAge(0);
cuki.setPath("/");
response.addCookie(cuki);

TẠO COOKIE VÀ GỬI VỀ CLIENT

```
@RequestMapping("/create/{user}")
public String create(@PathVariable("user") String username,
                     HttpServletResponse response) {
    Cookie cookie = new Cookie("user", username); // tạo cookie
    cookie.setMaxAge(5 * 24 * 60 * 60); // 5 ngày
    cookie.setPath("/"); // toàn website
    cookie.setHttpOnly(true); // không cho sửa đổi phía client

    response.addCookie(cookie); // gửi về client để lưu lại

    return "redirect:/read";
}
```

ĐỌC GIÁ TRỊ COOKIE

2

```
@ResponseBody
@RequestMapping("/read")
public String read(HttpServletRequest request) {
    Cookie[] cookies = request.getCookies();
    for(Cookie cookie: cookies) {
        if(cookie.getName().equals("user")) {
            return cookie.getValue();
        }
    }
    return "not found";
}
```

1

```
@ResponseBody
@RequestMapping("/read")
public String read(@CookieValue("user") String value) {
    return value;
}
```

Summary

- ☒ HttpServletRequest
- ☒ HttpSession
- ☒ ServletContext
- ☒ Cookie
  - ☒ Read
  - ☒ Create
  - ☒ Delete


Thank you