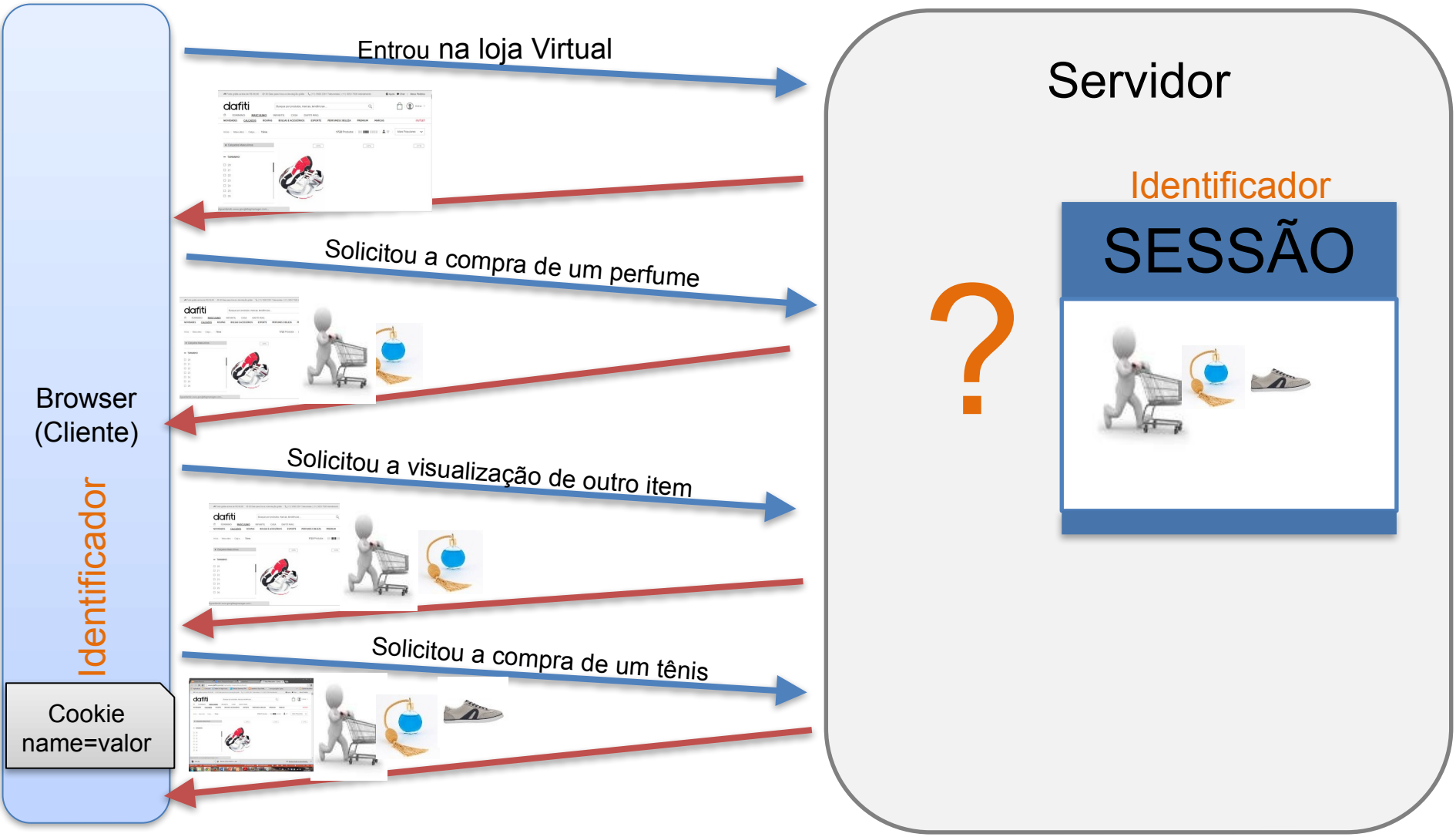
INSTITUTO FEDERAL
SANTA CATARINA
Câmpus Canoinhas

Sessão

INSTITUTO FEDERAL
SANTA CATARINA

Exemplo loja virtual

Compartilhando dados entre diferentes requisições



INSTITUTO FEDERAL
SANTA CATARINA

Cookies

Cookie
nome=valor

Browser
(Cliente)

Solicitação de adição de um item ao carrinho

HTTP response

Set-Cookie: nome=valor
[expires =DATA ;]
[path=CAMINHO ;]
[domain=NOME_DOMINIO ;]

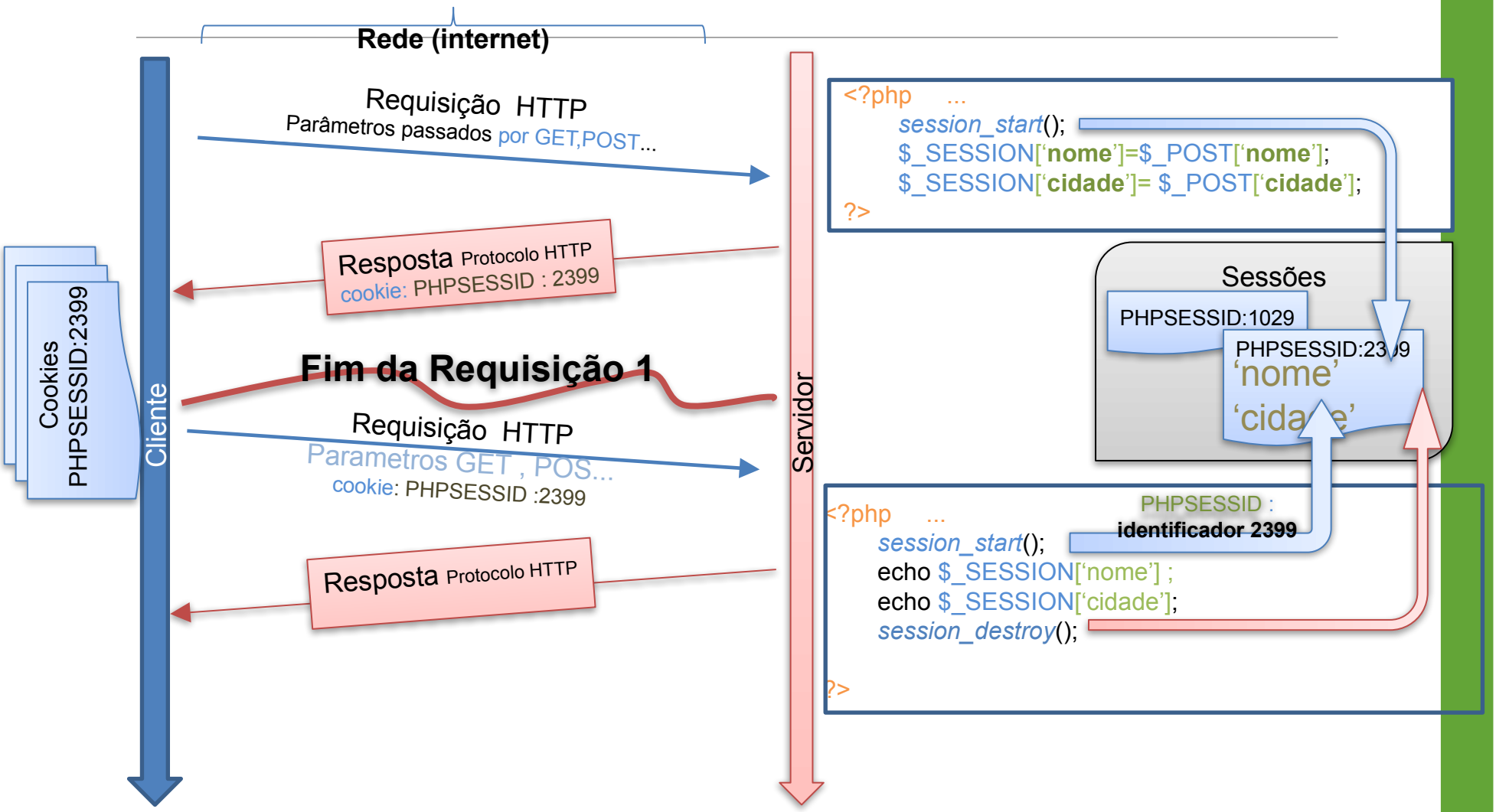
Clicou em outro item da loja
nome:valor

Servidor

SESSÃO

INSTITUTO FEDERAL
SANTA CATARINA

Como funciona a sessão do PH



Adaptado de : http://cscie12.dce.harvard.edu/lecture_notes/2011/20110504/handout.html

Sessão

Como vimos na aula passada, o HTTP não possui estado. Para descobrir qual cliente está fazendo a requisição, guardamos um cookie no cliente. Toda vez que ele realizar a requisição, ele deverá enviar os cookies.

Se quisermos guardar o nome do usuário, a idade, o saldo na conta corrente, então podemos criar um cookie para cada um dos dados. Isso é uma boa ideia?

Sessão

Claro que não. Todo dado que é armazenado no cliente pode ser fraudado, portanto é inseguro utilizar cookies para armazenar estes dados.

Então aonde deveriam ser armazenados os dados?

Sessão

Os dados do cliente devem ser armazenados no servidor. Desta forma eles ficam mais protegidos. Alguém precisará ter a senha de acesso ao servidor para conseguir alterar os dados.

Mas como podemos fazer para armazenar os dados no servidor?

Sessão

Podemos gravar os dados em arquivos ou em um banco de dados. Sempre que o cliente acessar a aplicação, nós recuperamos os dados do arquivo.

Mas como iremos saber qual é o cliente que está acessando, para então conseguir recuperar seus dados?

Sessão

Como vimos na aula passada, para identificar o cliente, precisamos utilizar cookies. Inveś de armazenar o saldo do cliente no cookie, vamos armazenar somente um identificador (ID) no cookie.

Quando o cliente enviar uma requisição, ele enviará o cookie com o ID. Nós utilizamos este ID para saber qual arquivo buscar e assim recuperar os dados do cliente.

Será que é muito complicado implementar esse mecanismo em PHP?

Sessão

Na verdade, é bem simples. O PHP já vem com um mecanismo padrão que faz tudo isso que foi explicado. Este mecanismo chamase **sessão**. Para criar e utilizar a sessão no PHP basta seguir 2 passos:

- iniciar a sessão: deve ser chamada a função session_start();
- utilizar a sessão: os dados são armazenados e recuperados utilizando a variável superglobal **$_SESSION**. Ela é um vetor e conforme alteramos ela, a sessão é automaticamente alterada.

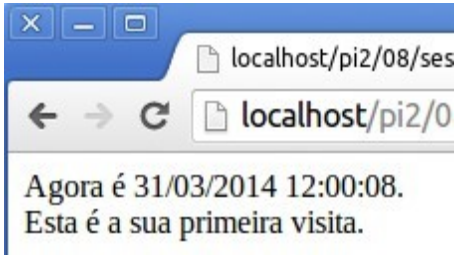
Sessão

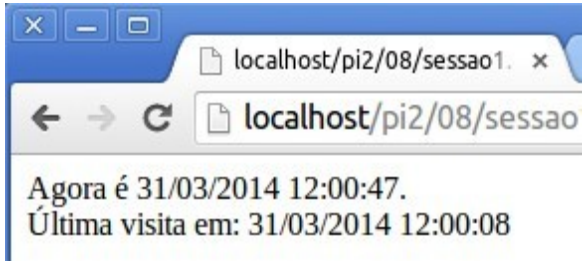
Vamos ver o mesmo exemplo que foi utilizado com cookies, só que desta vez, com sessão.

```php
<?
header('Content-Type: text/html; charset=utf-8');

session_start();
if (isset($_SESSION['ultima_visita'])) {
    $mensagem = "Última visita em: $_SESSION[ultima_visita]";
} else {
    $mensagem = "Esta é a sua primeira visita.";
}

$agora = date('d/m/Y H:i:s');
$_SESSION['ultima_visita'] = $agora;
echo "Agora é $agora.<br />$mensagem";
```





Sessão

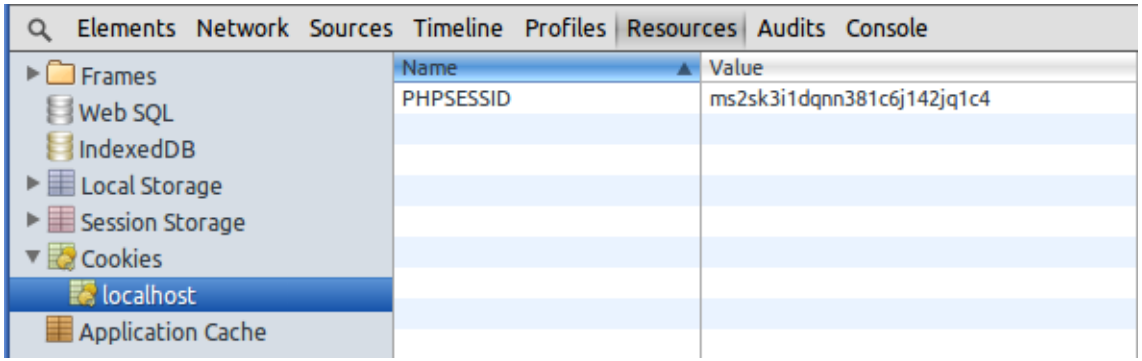
Se compararmos com cookie, temos uma linha a mais para chamar a função que inicia a sessão: session_start(). Outra diferença é que para gravar um valor na sessão, não é necessário chamar uma função, bastando apenas gravar um valor na variável $_SESSION.

Se quisermos deletar um valor da sessão, podemos utilizar a função unset(), da mesma forma que faríamos para deletar uma variável.

O índice do vetor $_SESSION deve ser uma string. Ela pode conter qualquer tipo de valor (até outros vetores).

Sessão

Vamos dar uma olhada nos cookie salvos no navegador quando utilizamos a sessão.



Quando iniciamos a sessão, o PHP automaticamente recupera ou cria o cookie PHPSESSID. Ele possui uma string com caracteres aleatórios e serve como ID para recuperar os dados em um arquivo no servidor. Você consegue ver alguma falha de segurança nesse mecanismo?

Sessão

Para alguém conseguir burlar a sessão, ele teria que alterar o cookie para outro valor que fosse de outro usuário logado. Ele poderia tentar strings aleatórias até conseguir, apesar de que isso demoraria muito tempo.

Outra forma seria tentar roubar os cookies do navegador. O navegador tenta proteger os cookies para que isso não aconteça, porém um vírus no computador conseguiria ler os cookies.

A sessão do PHP é bem simples. Existem outras implementações na internet e em frameworks que são mais seguras.

Sessão

Antes de finalizarmos a aula, vamos ver como destruir uma sessão. Para que uma sessão seja destruída, ela primeiro deve ser iniciada (session_start). Depois disso, basta chamarmos a função **session_destroy()**. A partir daí, o cookie será destruído no navegador e o arquivo será deletado. Tome cuidado, pois mesmo depois de chamar session_destroy a variável $_SESSION continuará a mesma. É aconselhável redirecionar o cliente depois de destruir a sessão, pois na próxima requisição a variável $_SESSION estará limpa.

Exercício

Crie um blog de sessão. Terá somente um endereço para o cliente acessar. No topo da página, ele poderá enviar uma mensagem e seu nome. Em baixo, deverá ter uma listagem das mensagens enviadas anteriormente. Salve as mensagens na sessão. A listagem deverá ter somente as últimas 10 mensagens. A listagem deve ser ordenada de tal forma que a mensagem mais antiga apareça mais em cima.

Referências

NIEDERAUER, Juliano. **Desenvolvendo websites com PHP**. 2. ed. São Paulo: Novatec, 2011. 301 p. ISBN 9788575222348.