Sessões

Cookies HTTP

Interface HttpSession

Atributos de sessão

Frederico Costa Guedes Pereira © 2006

fredguedespereira@gmail.com

O problema

- O protocolo HTTP **não mantém estado** entre transações distintas
  - Ao término do atendimento da requisição, a conexão TCP é fechada pelo servidor
  - Toda requisição é vista como nova e nunca como continuação de um trabalho anterior
- Problema

  Como implementar aplicações que necessitam de estado entre transações HTTP distintas, tais como, carrinhos de compras, webmails, etc?

O Problema

- Exemplo: uma sessão de web-mail

O Problema

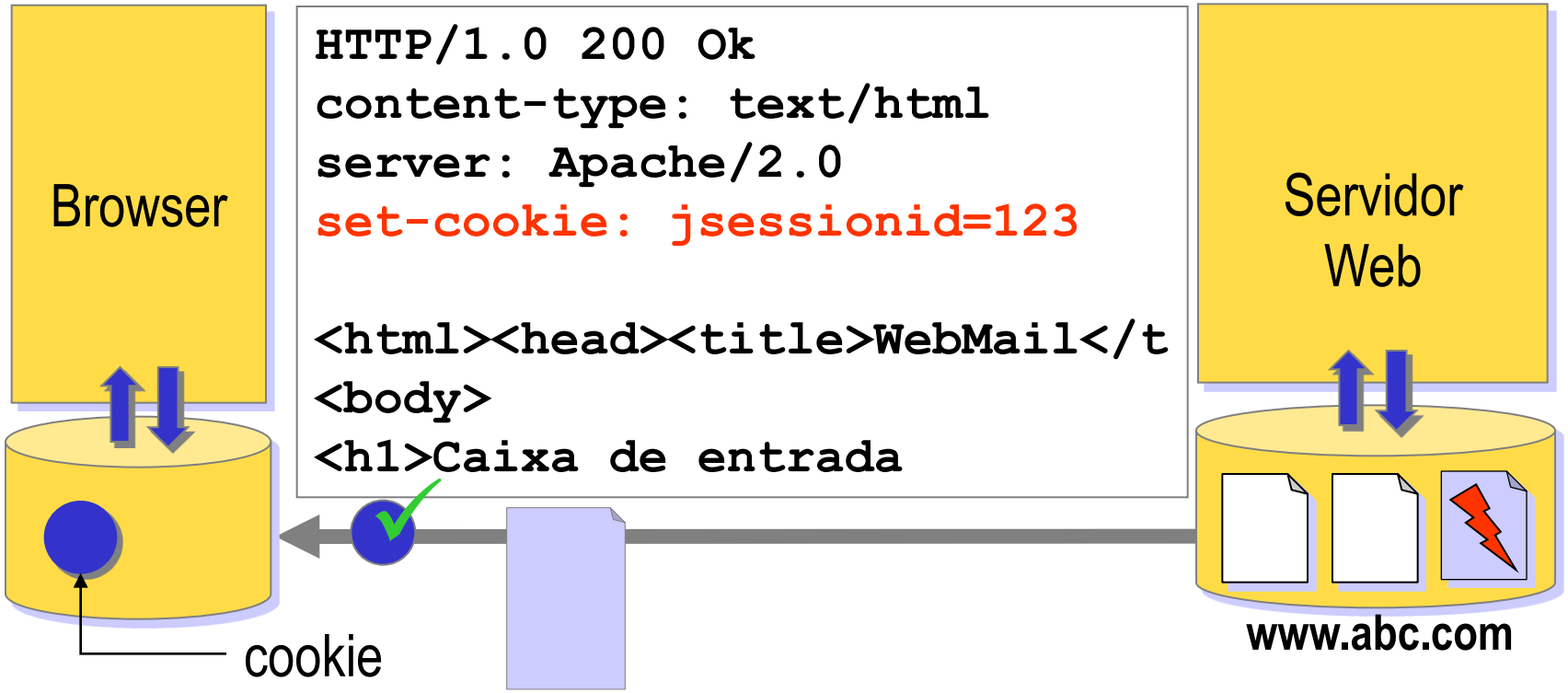
- Problema
  - Como fazer o servidor "entender" que a terceira transação HTTP é um pedido de listagem de um usuário que já se logou no sistema?
- A Netscape definiu o conceito de **cookie**
  - Um *cookie* é um pedaço de informação importante
  - Na API de servlets os cookies são usados para identificar uma sessão de trabalho de um usuário

Cookies

- Identificação de sessões usando *cookies*

Cookies

- Exemplo:
  - Depois que o login do usuário foi validado com sucesso



Cookies

- Exemplo:
  - Ao solicitar a listagem das mensagens, o browser envia o cookie de identificação previamente definido

Sessões

- Onde os produtos do carrinho serão guardados?

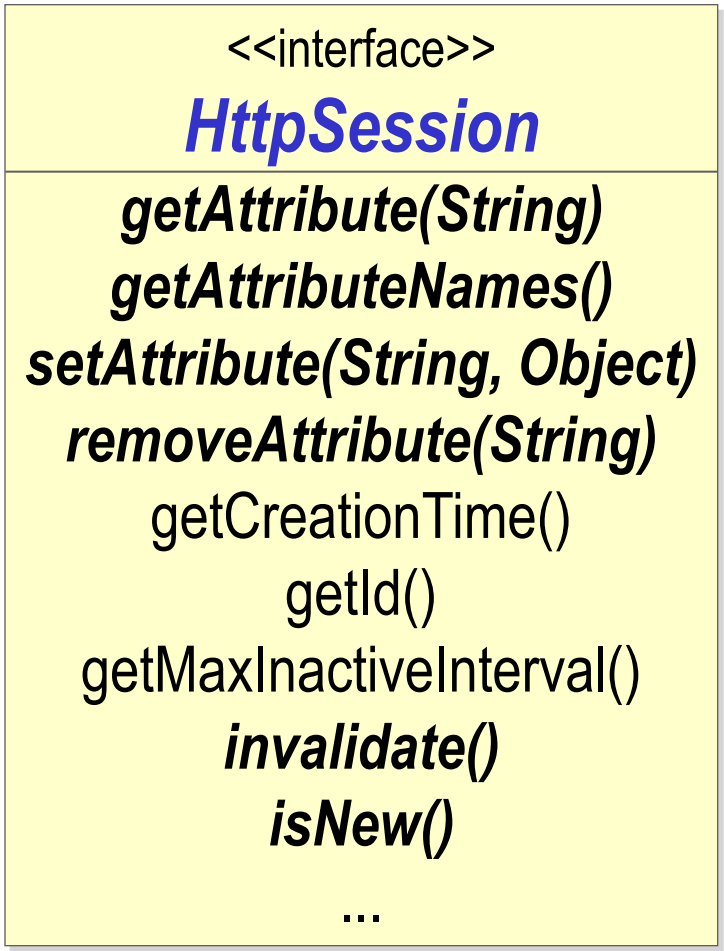
Sessões

- O *cookie* apenas identifica o cliente...
- Algum mecanismo do lado do servidor precisa guardar o "carrinho" com os itens de cada cliente... e isto individualmente para cada cliente!
  - Não é admitido que um cliente inclua um produto no carrinho de outro ou vice-versa
  - O identificador da sessão é uma boa "chave" para pegar o "carrinho" individual de cada cliente

Mas onde vamos guardar os carrinhos de cada cliente?

Sessões

- Num objeto chamado **HttpSession**



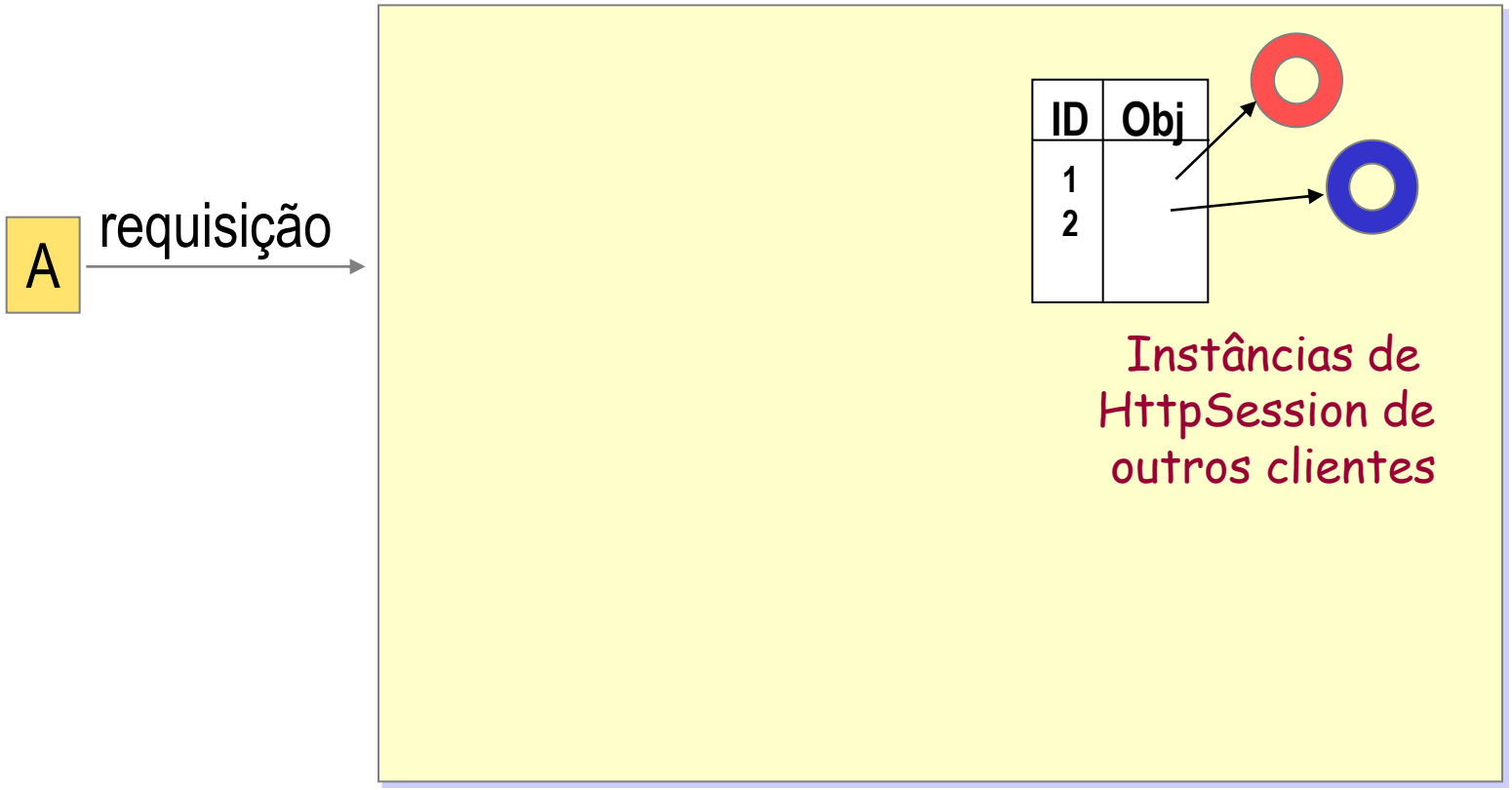
- ✓ Criadas a partir do HttpServletRequest
- ✓ Utilizam um cookie para identificação da sessão
- ✓ Também possuem atributos

Sessões

- Relação entre sessões e cookies:
  - Ao ser criada uma sessão, automaticamente é criado um *cookie* chamado JSESSIONID cujo valor é um número único (ID) de sessão gerado automaticamente para cada usuário
  - O programador não toma conhecimento deste cookie
  - O IP e a porta do browser do usuário são usados para gerar o ID único. Exemplo: 0AAB6784DF35C994FE10D
  - Os objetos HttpSession são mantidos no container e acessados com o ID da sessão em requisições posteriores

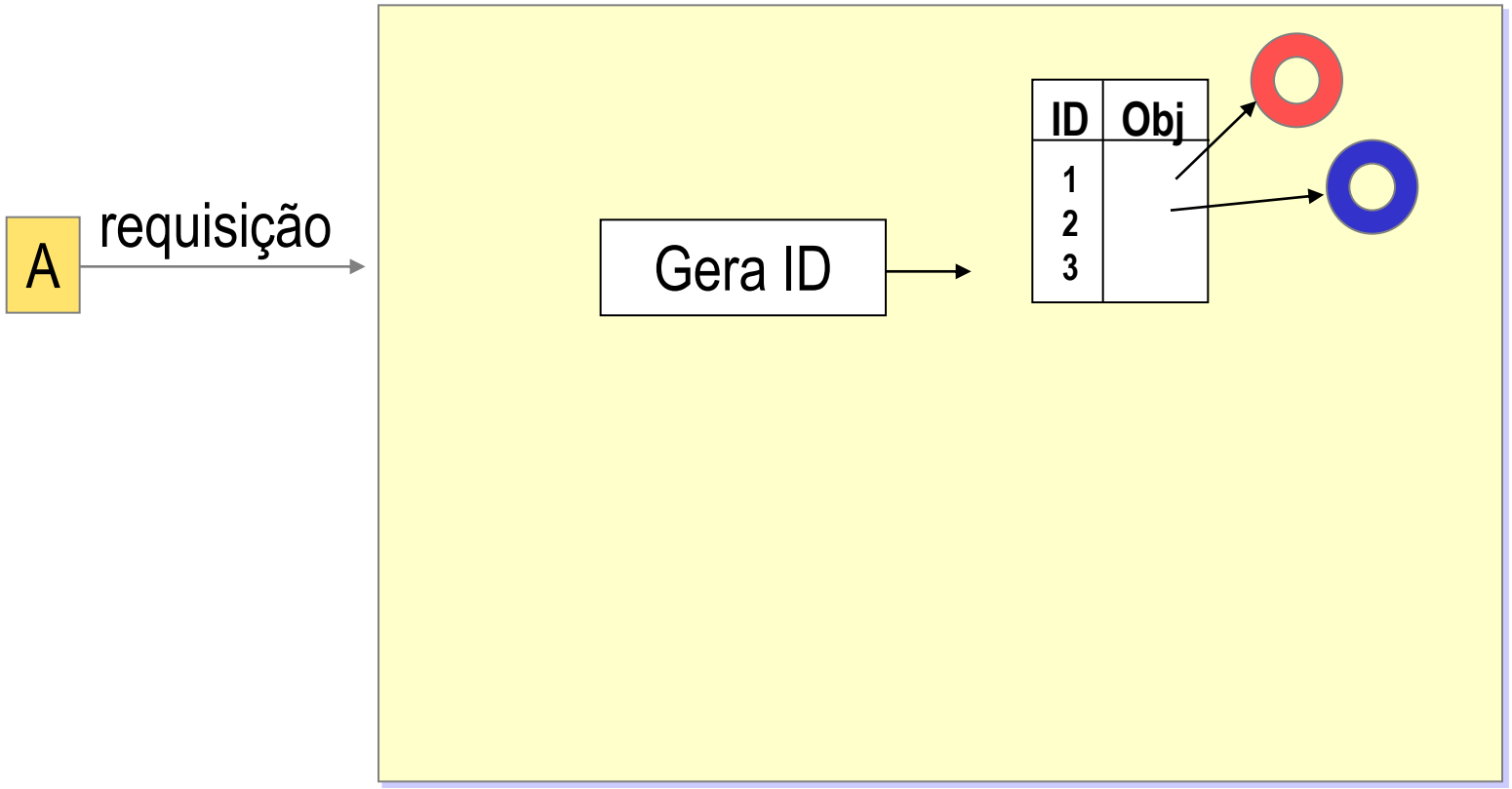
Criação de sessões

- Crie uma nova sessão para o usuário A



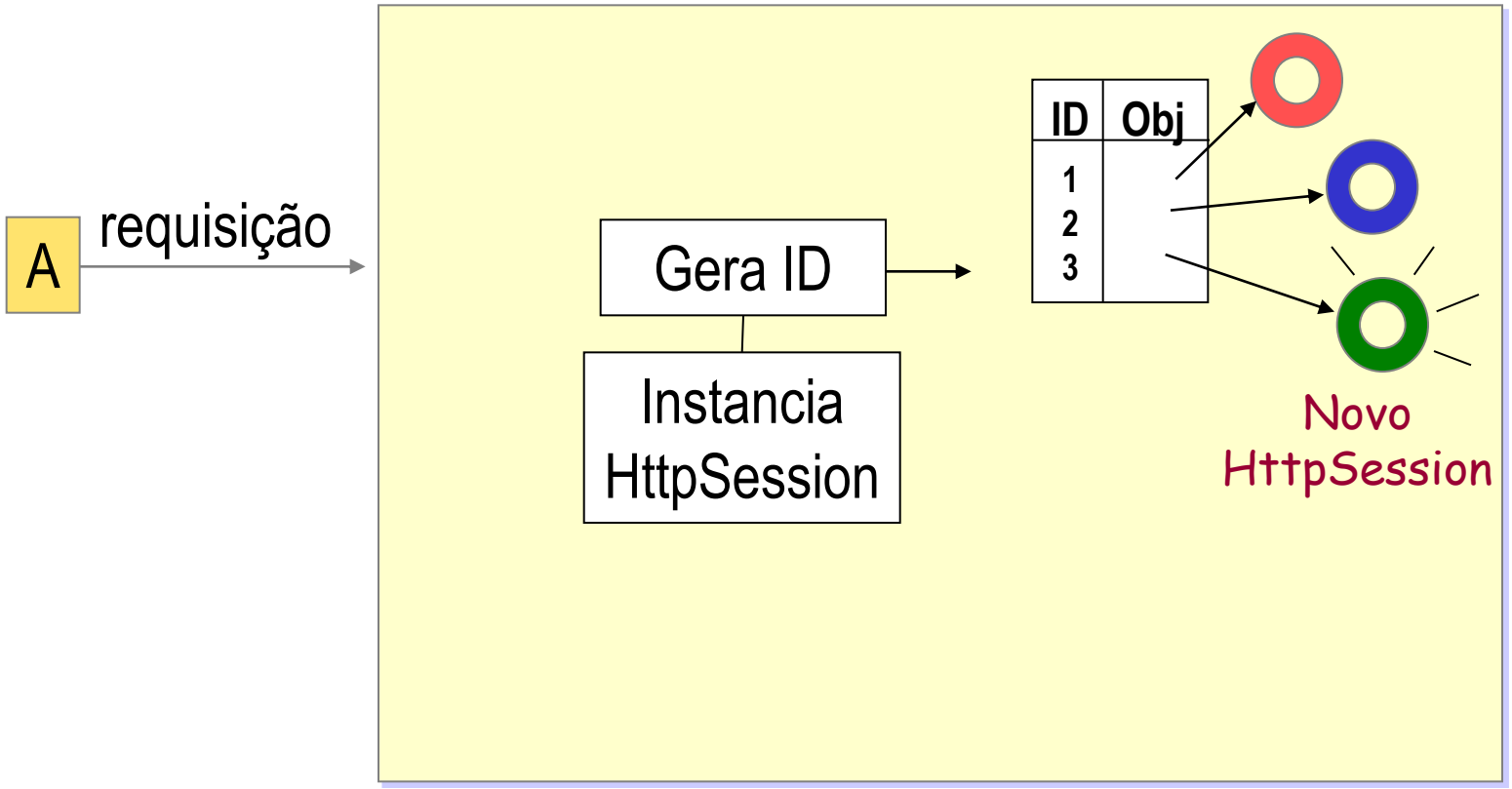
Criação de sessões

- ID único para o usuário A é gerado



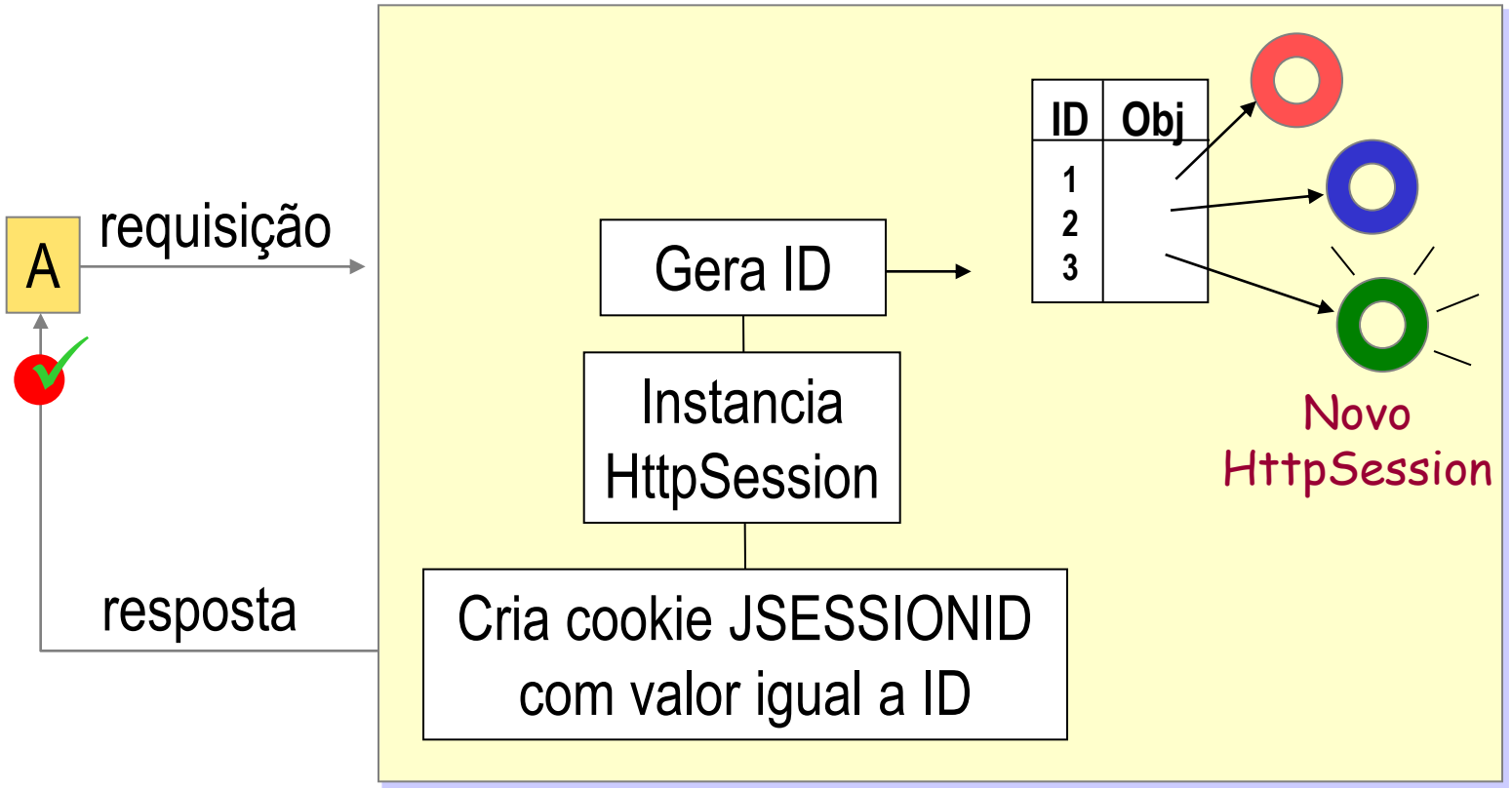
Criação de sessões

- Instancia um objeto HttpSession para o usuário A



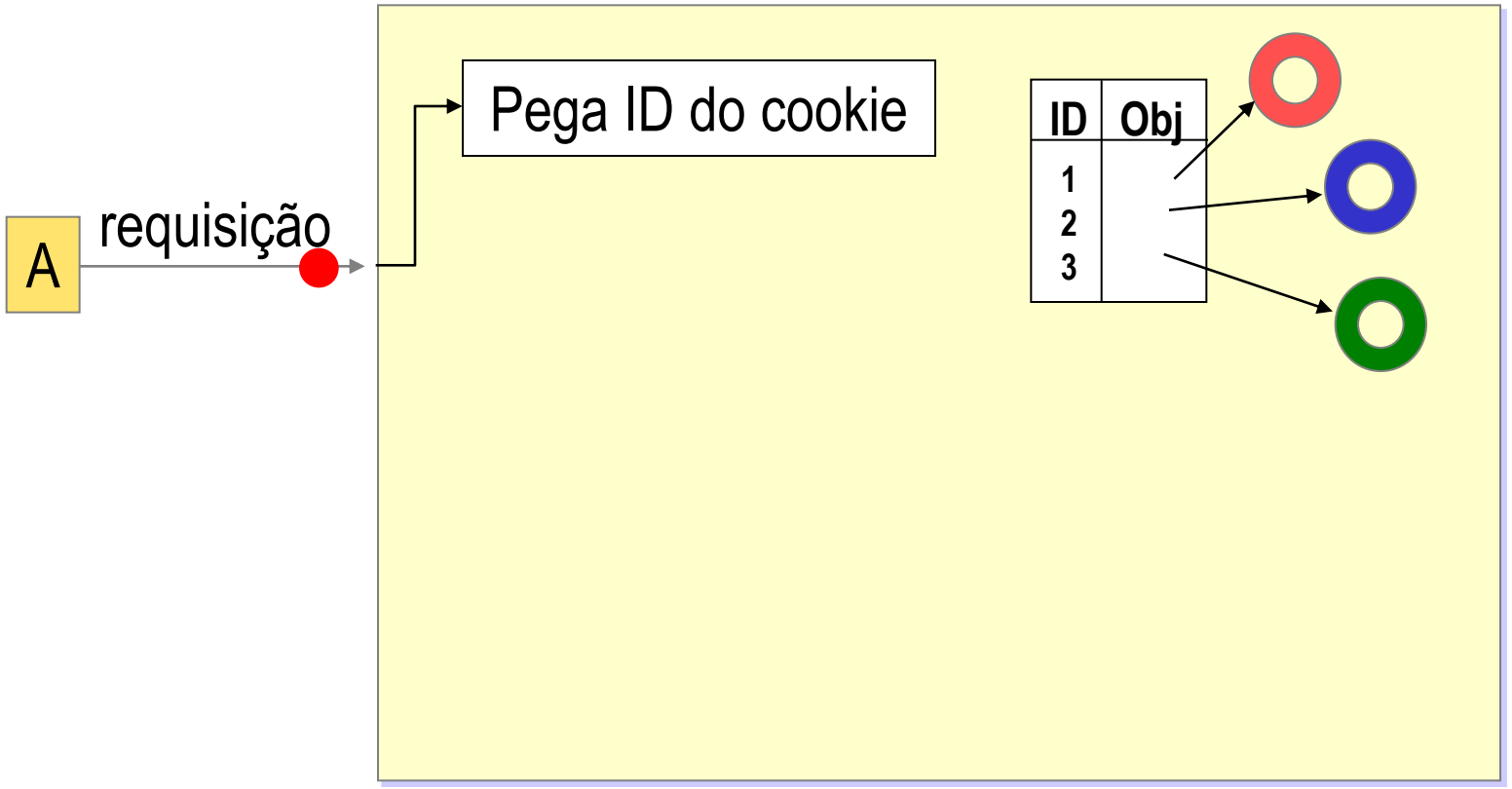
Criação de sessões

- Cria cookie para identificar a sessão do usuário A



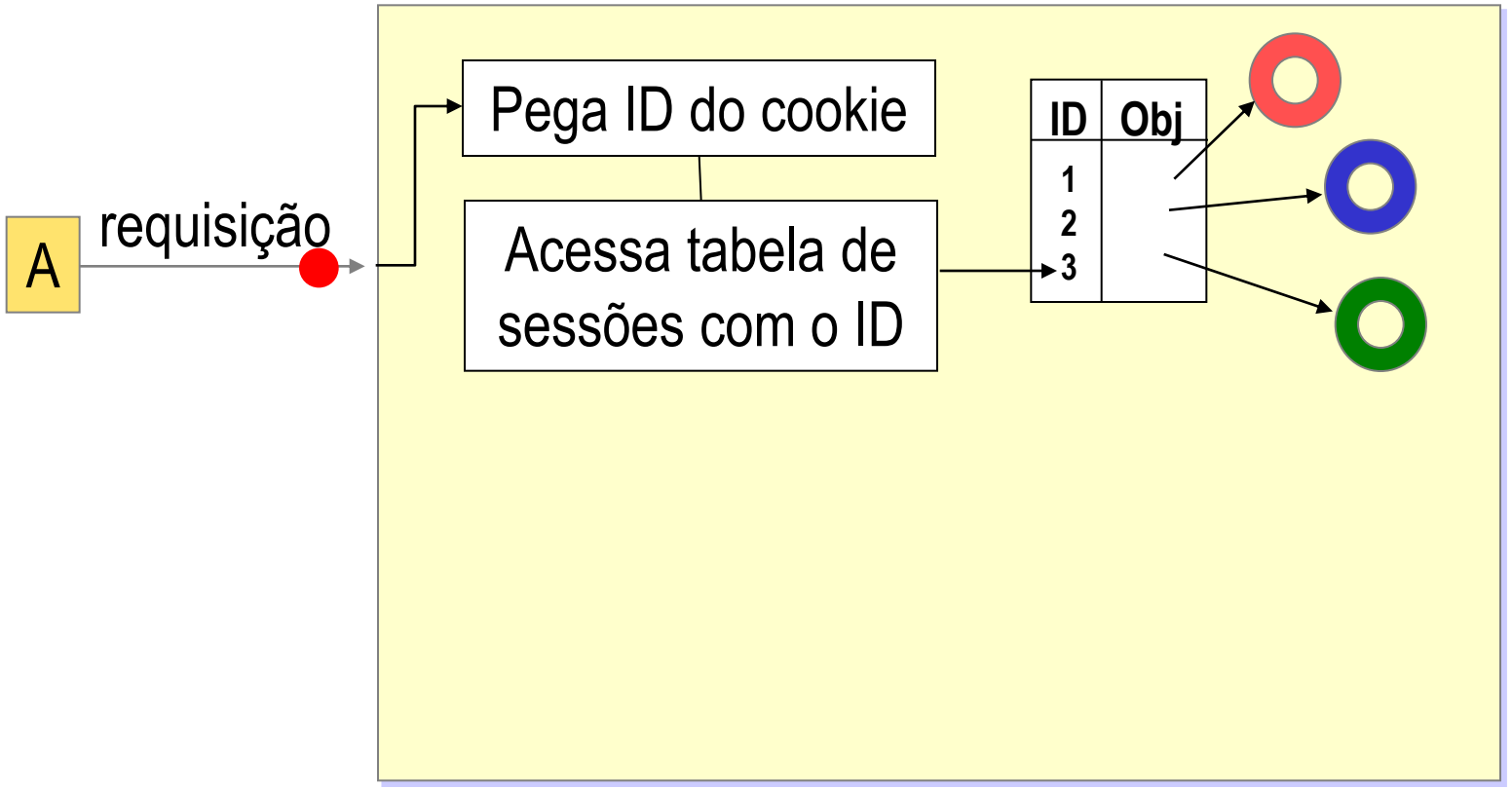
Criação de sessões

- Usuário A põe item no carrinho



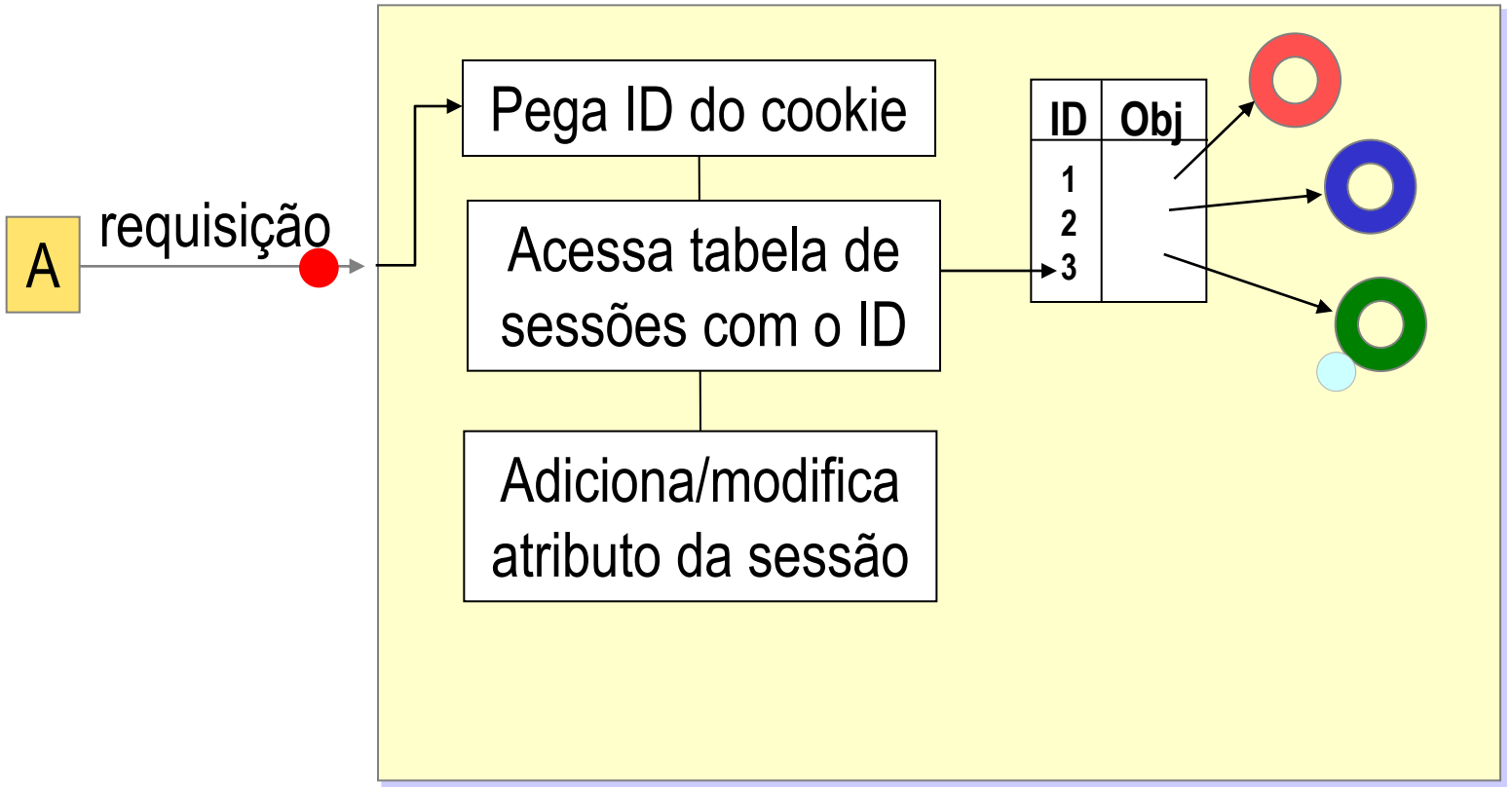
Criação de sessões

- Usuário A põe item no carrinho

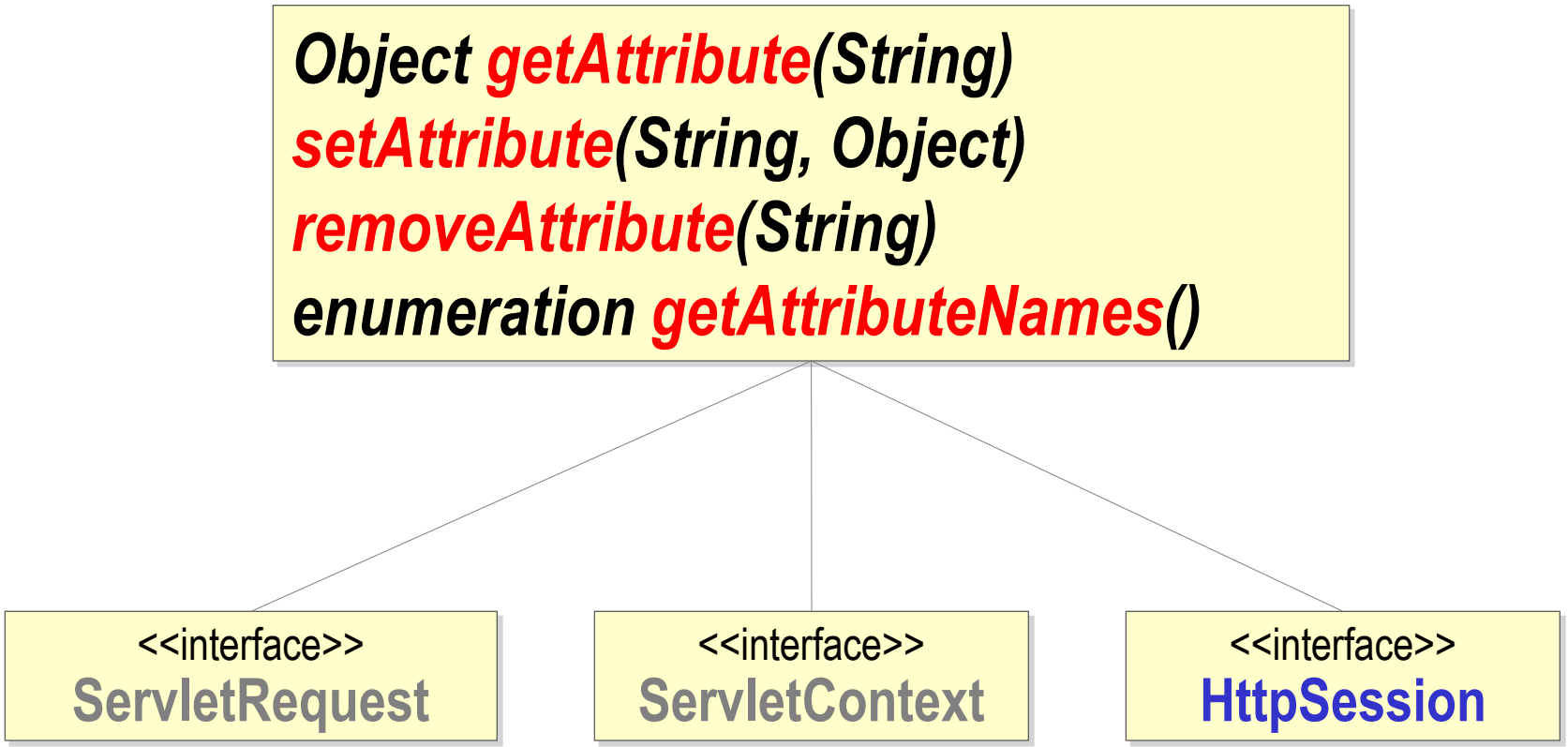


Criação de sessões

- Usuário A põe item no carrinho



API de atributos



Tudo isso em código fica mais simples...

```
public class AddCarrinhoServlet extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse resp)  {
        String item = req.getParameter("item");
        HttpSession session = req.getSession();
        if (session.isNew()) {
            Carrinho c = new Carrinho();
            c.addProduto(item);
            session.setAttribute("carrinho", c);
        } else {
            Carrinho c = (Carrinho) session.getAttribute("carrinho");
            c.addProduto(item);
        }
    }
}
```

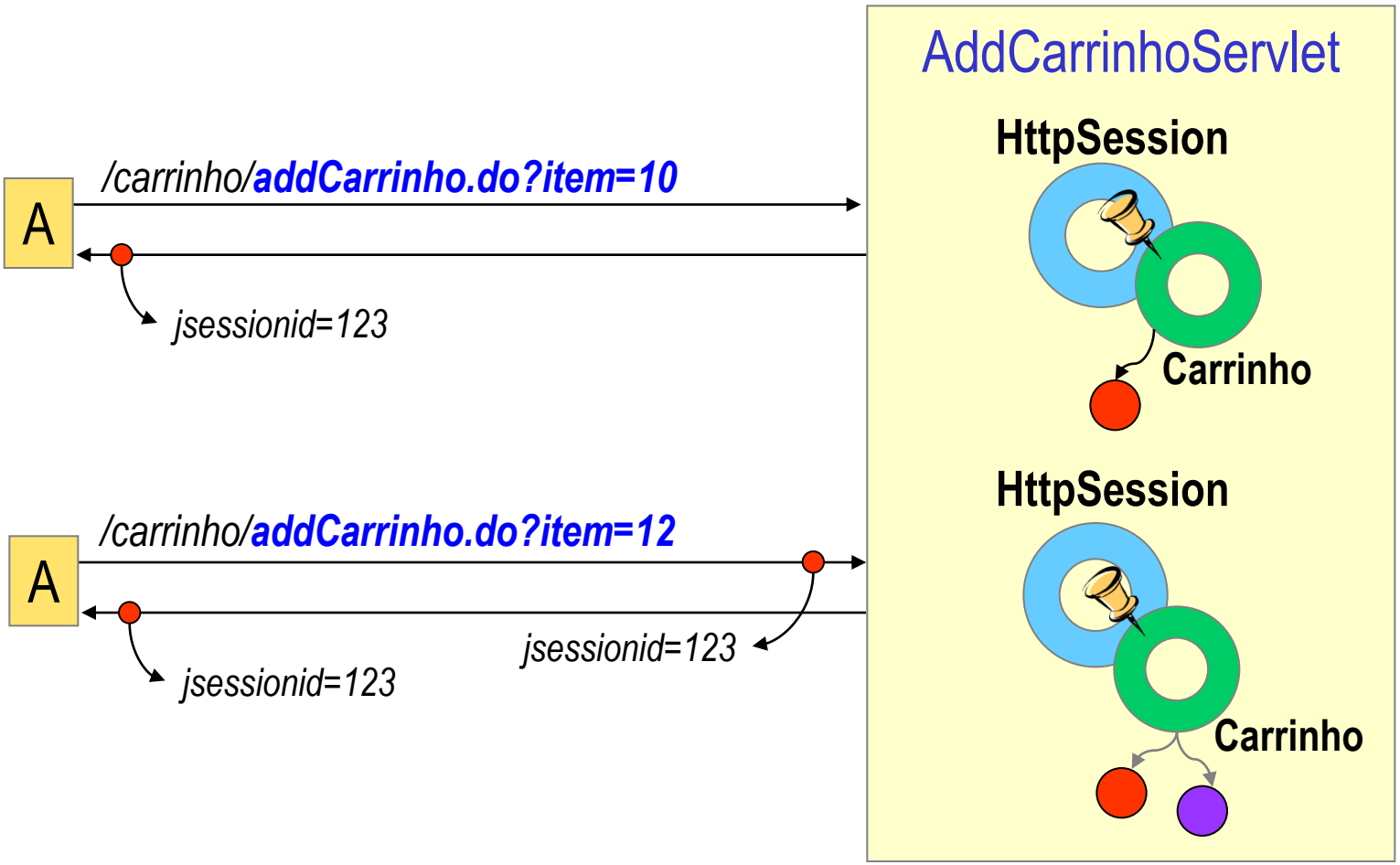
É aqui que o ID é gerado, a sessão criada e o cookie definido

Põe carrinho na sessão

Aponta para o carrinho que já está na sessão

Atributos de sessão

- Exemplo do carrinho de compras



E se o usuário desabilitar *cookies* no browser?

A API de sessões de servlets usa **reescrita de URLs** para identificar a sessão automática e transparentemente

Isto é, quase transparentemente... URLs geradas dentro de servlets ou estaticamente em JSPs precisam ser codificadas pelo programador

Codificação de URLs

- Exemplo de uso:

```
public class MostraCarrinhoServlet extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse resp)  {
        ...
        String url = "http://servidor:8080/carrinho/limpaCarrinho.do";
        urlCod = resp.encodeURL(url);
        out.println("<A HREF=\"" + urlCod + "\">...</A>");
        ...
}
```

Codificação de URLs

- URL antes do encodeURL:

  http://serv:8080/carrinho/limpaCarrinho.do

- URL depois do encodeURL :

  http://serv:8080/carrinho/limpaCarrinho.do;JSESSIONID=0A349FBC108D622E

Como não se sabe qual mecanismo de identificação da sessão será usado, é boa prática passar todas as URLs para recursos que estejam na sessão pelo encodeURL().

Finalização de sessões

- Use os seguinte métodos para invalidar uma sessão
  - void ***invalidate*** ()
  - void ***setMaxInactiveInterval*** (int segundos)
- Outros métodos de HttpSession
  - public String ***getId***(): identificador da sessão
  - public boolean ***isNew***(): `true` se a sessão for recém criada
  - public long ***getCreationTime***(): data da criação da sessão
  - public long ***getLasAccessedTime***(): data do último acesso
  - public int ***getMaxInactiveInterval***(): valor do maior intervalo sem uso para a sessão

Configuração de sessões no DD

- Tempo máximo, em minutos, de todas as sessões da aplicação web

```
<webapp>
  ...
  <session-config>
    <session-timeout>12</session-timeout>
  </session-config>
  ...
</webapp>
```