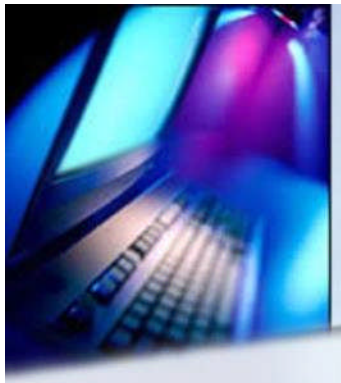
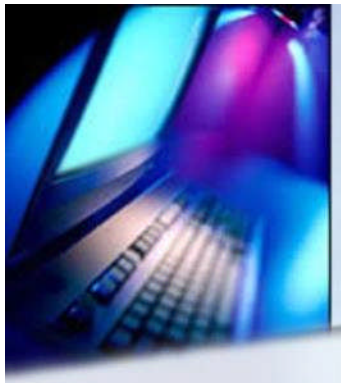
Domínio

Autor : Marcelo Passos dos Santos



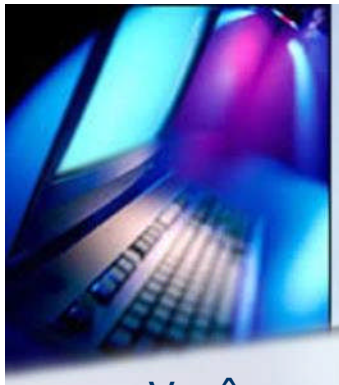
Objetivo

Apresentar o conceito de domínios na WEB.



Conceito de domínio

- O conceito de domínio foi criado com o objetivo de facilitar a memorização dos endereços de computadores na Internet.
- Imagine ter que memorizar um número como 186.192.90.15 (endereço IP) e inseri-lo na barra de navegação para poder consultar o site e depois ainda decorar várias outras sequências de números de site que utilize regularmente.
- Seria uma tarefa bastante difícil e a Web certamente não teria o sucesso que tem hoje.
- Escrever g1.com.br ou google.com torna-se bastante intuitivo e fácil de memorizar.



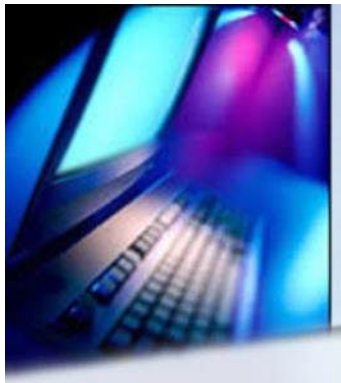
Achando um *Site* na internet

- Você que está usando G1, já conhece o endereço : http://www.g1.com.br.
- Repare que depois do nome do protocolo vem ://seguido pelo nome do site www.g1.com.br.
- No vocabulário de um desenvolvedor o www.g1.com.br é o domínio (ou domain).
- A abreviação www representa o *world wide web*.
- O nome do domínio é organizado em uma hierarquia que foi criada para organizar os sites na internet e para agente ter algo fácil para se lembrar.
- Para ser correto, a internet funciona na verdade sem esses domínios. Aqueles nomes são coisas dos humanos, as máquinas na internet têm uma outra forma de se endereçar.
- Elas usam o que se chama endereços de IP, que são números - muito difíceis para decorar.

Exemplo : nslookup www.g1.com.br

Resulta no IP : 186.192.90.5

Esse comando procura o numero IP do G1 na internet. Podemos até usar esse endereço no navegador:



Portas

- Quando usamos a URL http://www.g1.com.br abrimos uma conexão com o servidor que roda em algum lugar na internet.
- Para estabelecer uma conexão na rede é preciso saber qual é o endereço IP, e já vimos como descobrí-lo.
- Agora imagine que o servidor é uma casa: dependendo da casa há várias portas disponíveis.
- O que é preciso saber é qual porta devemos utilizar quando chegarmos na casa. Ou seja, devemos saber qual porta é utilizada para o protocolo HTTP!
- A porta reservada para o protocolo HTTP é o 80.
- Novamente um número, e como o navegador já sabe essa porta padrão podemos omiti-la, mas nada nos impede adicioná-la no endereço, por exemplo: http://186.192.90.5:80

UniFOA
Centro Universitário de Volta Redonda

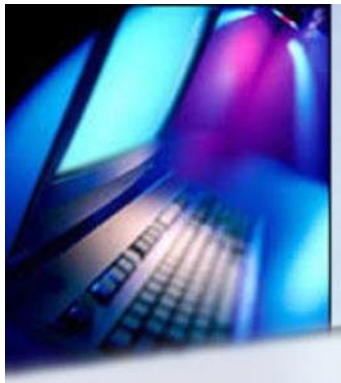
O cliente pede (Request) e o servidor responde (Response)

- **Modelo Requisição Resposta**
  - Realizaremos um teste **efetuando login** na área restrita do UniFOA
  - Quando preenchemos o formulário de acesso e clicamos no botão, o navegador envia o nosso *login* e a nossa *senha* para o servidor através do protocolo HTTP!
  - No mundo HTTP, a requisição enviada pelo navegador para o servidor é chamada de **REQUEST HTTP**.
  - Recebemos a página *PortalSagres/Modules/Portal/Services/ ...* como resposta já que enviamos login e senha válidos. No mundo HTTP essa resposta é chamada de **RESPONSE HTTP**.
  - A comunicação segue sempre esse modelo: o cliente envia uma requisição e o servidor responde.
- *Requisição e Resposta* ou em inglês: **Request-Response**.
  - É importante saber que a comunicação sempre começa com o cliente: é ele quem pede as informações.
  - O servidor responde apenas o que foi requisitado e nunca inicia a comunicação!

O cliente pede (Request) e o servidor responde (Response)

- Reparem que, mesmo após termos realizado o login e termos enviado várias requisições, aparece o topo da área restritra com o mesmo nome e menu principal.

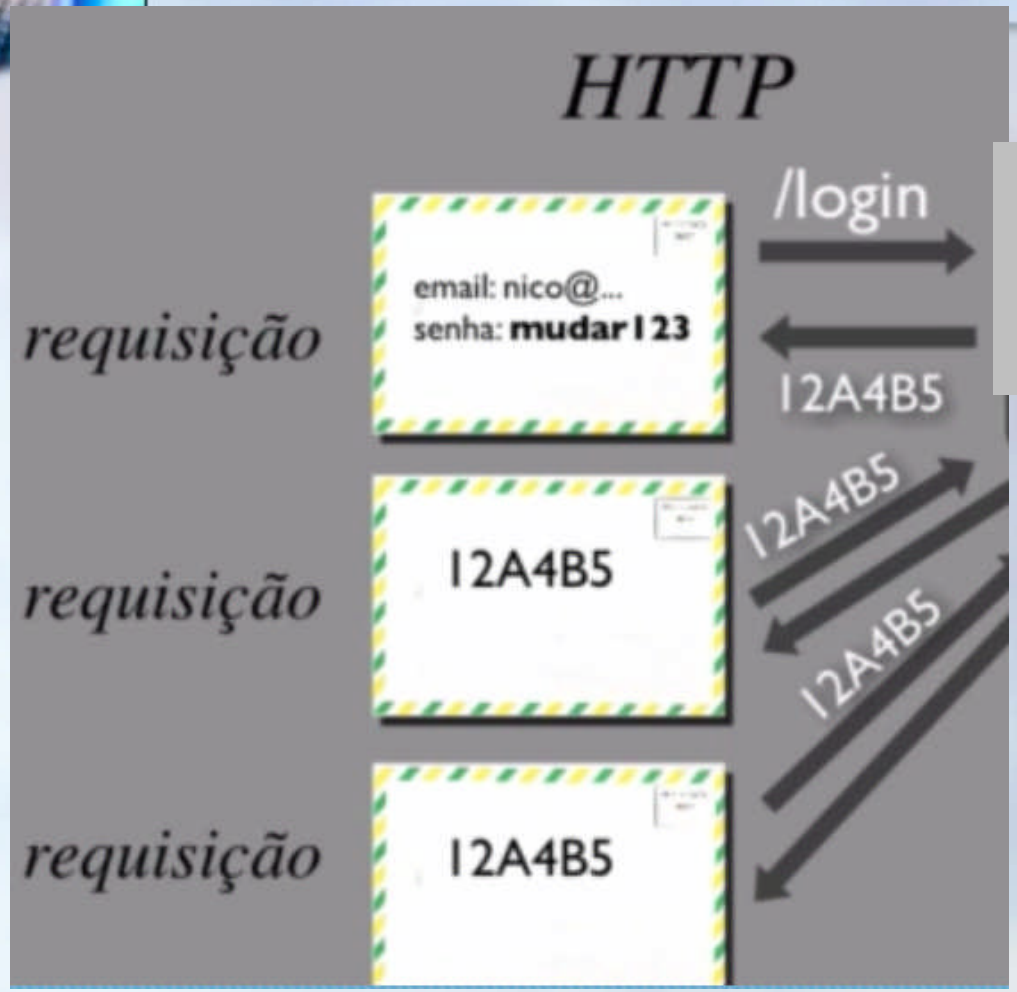
- Ou seja, a área restrita do UniFOA se lembra de alguma forma que eu fiz login em alguma requisição anterior.
- Cada requisição deve enviar todas as informações para gerar a resposta. Isso significa que o navegador envia em cada requisição informações sobre o meu usuário!
- Se cada requisição for independente da outra, e não tem como se lembrar das requisições anteriores, não tem outra explicação a não ser que o navegador envie os dados sobre o meu usuário em cada requisição! Lembre-se de uma carta postal, ela sempre precisa ter os dados do remetente e aqui não é diferente!



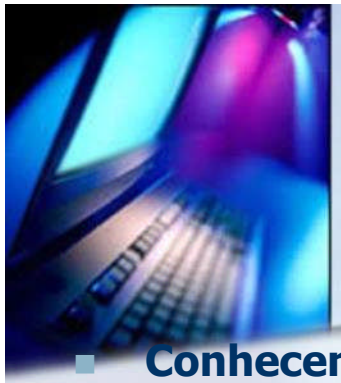
O cliente pede (Request) e o servidor responde (Response)

- Então o navegador *envia login e senha em cada requisição*?
  - Não, não seria muito elegante nem seguro fazer isso, mas ele faz algo parecido.
- Quando efetuamos o login, a área restrita valida os nossos dados, certo?
  - Nesse momento, o servidor tem certeza que o usuário existe e gera uma identificação quase aleatória para o usuário.
- Essa identificação é um número criado na hora e muito difícil de adivinhar denominado hash.
  - Esse número é a identificação temporária do usuário e ele será devolvido na resposta.

O cliente pede (Request) e o servidor responde (Response)



Área Restrita
12A4B5



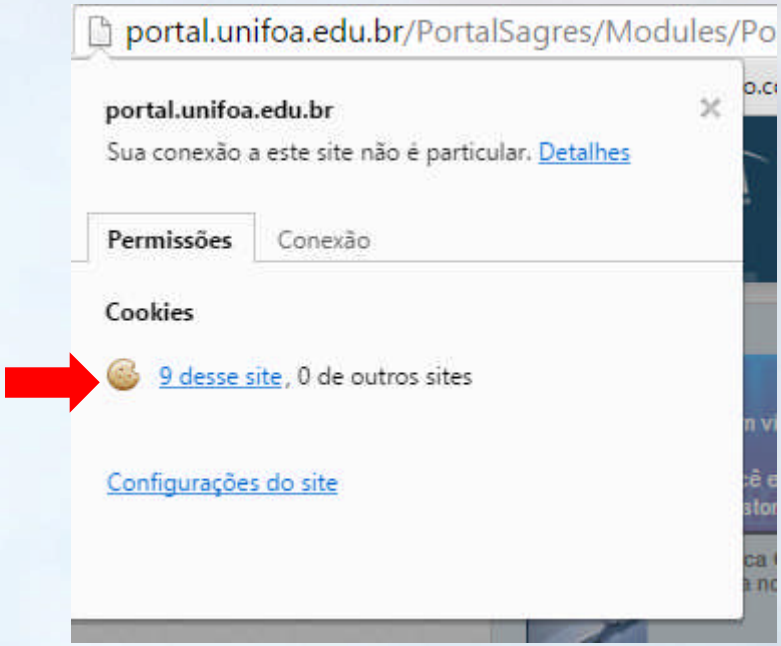
Onde fica armazenado estes números

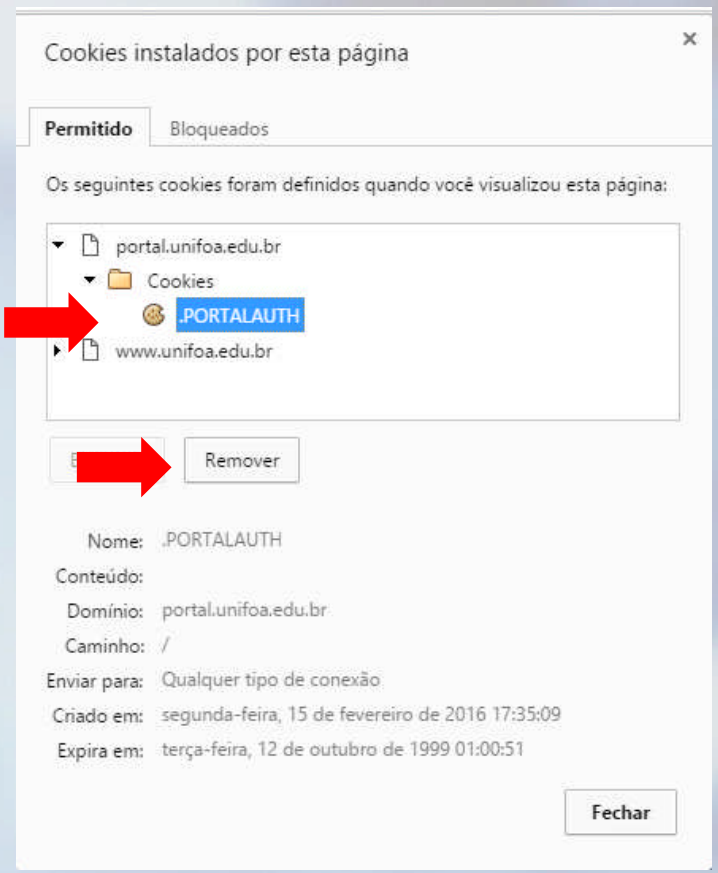
- **Conhecendo cookies**
- Então onde está esse número? O navegador grava esse número em um arquivo especial para cada site, são os famosos **cookies**.
- Como acessar esse cookie depende um pouco do navegador em uso.
- O mais importante é entender o porquê da existência desse número e onde ele foi gravado.
- Normalmente o nome do cookie é algo como session-id, dependendo da plataforma utilizada ele pode se chamar de PHPSESSID ou ASP.NET_SessionId ou JSESSIONID ou outro nome que foi inventado!
- Como o protocolo HTTP é stateless por natureza, ou seja, não sabe das requisições anterioires.
- As plataformas ajudam a gerar esse número e a criar o cookie de maneira transparente.
- Isto é o que as plataformas gerenciam as SESSÕES com o usuário.

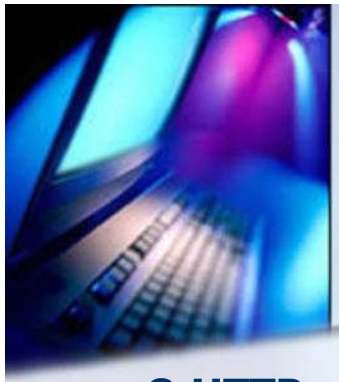
UniFOA
Centro Universitário de Volta Redonda

Excluindo Cookies

- **Conhecendo cookies**
- Para excluir um cookie, basta clica na URL (Barra de endereços), clicar em cookies e após remover.







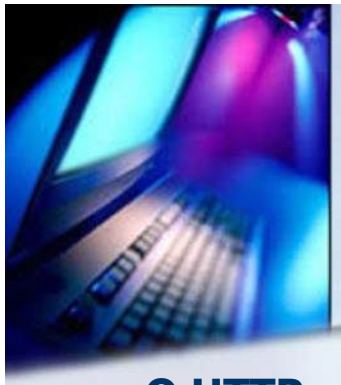
Exercícios

- **O HTTP e o estado das requisições**
- Qual das informações abaixo é verdadeira?

( ) - Uma requisição sempre deve ser enviada com todas as informações necessárias, o que faz uma requisição ser sempre independente das demais.

( ) - O HTTP guarda o estado das requisições no navegador do usuário. Por consequência, a segunda requisição sempre será feita para o mesmo destino da primeira.

( ) - A letra **s** na sigla *HTTPS* significa **stateless**. Usamos HTTPs para trabalharmos com um protocolo sem armazenamento de estado



Exercícios

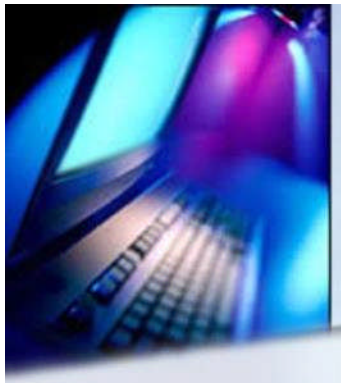
- **O HTTP e o estado das requisições**
- Qual das informações abaixo é verdadeira?

(X) - Uma requisição sempre deve ser enviada com todas as informações necessárias, o que faz uma requisição ser sempre independente das demais.

( ) - O HTTP guarda o estado das requisições no navegador do usuário. Por consequência, a segunda requisição sempre será feita para o mesmo destino da primeira.

( ) - A letra **s** na sigla *HTTPS* significa **stateless**. Usamos HTTPs para trabalharmos com um protocolo sem armazenamento de estado

UniFOA
Centro Universitário de Volta Redonda



Exercícios

- **Sessão HTTP**
- O que é uma sessão HTTP?

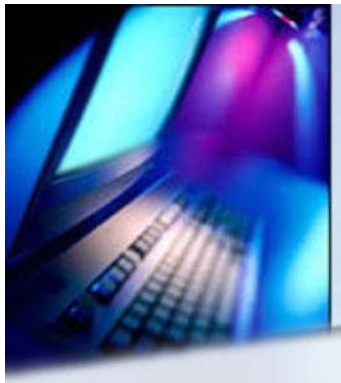
( ) - É o tempo entre requisição e resposta

( ) - É o número gerado para identificar o servidor

( ) - É o número gerado para identificar o cliente

( ) - É o tempo que o cliente utiliza a aplicação Web

UniFOA
Centro Universitário de Volta Redonda



Exercícios

- **Sessão HTTP**
- O que é uma sessão HTTP?

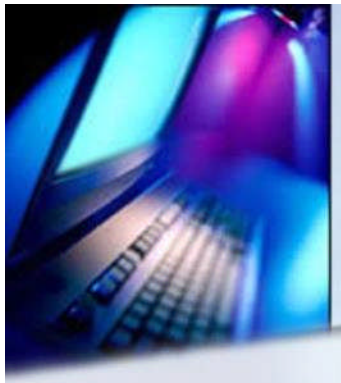
( ) - É o tempo entre requisição e resposta

( ) - É o número gerado para identificar o servidor

( ) - É o número gerado para identificar o cliente

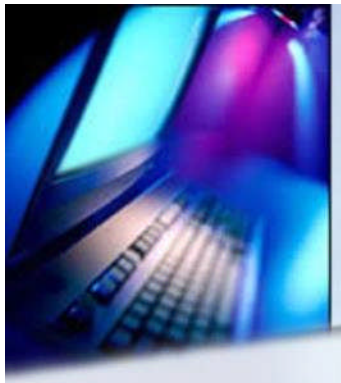
(X) - É o tempo que o cliente utiliza a aplicação Web

UniFOA
Centro Universitário de Volta Redonda



Exercícios

- **O que é um cookie ?**



Exercícios

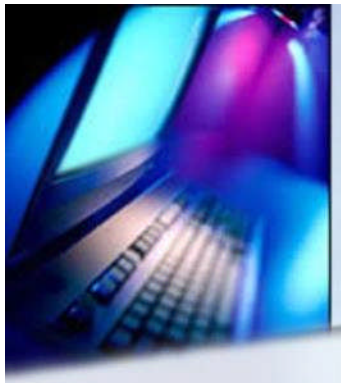
- **Comunicação em HTTP**
- Qual dessas alternativas é verdadeira?

( ) - Em HTTP o servidor sempre envia uma requisição ao cliente para poder alterar algo na tela.

( ) - Uma comunicação com HTTP sempre é iniciada pelo cliente que manda uma requisição ao servidor esperando por uma resposta.

( ) - Quando trabalhamos com HTTP, a comunicação é sempre iniciada pelo lado do cliente que envia uma requisição ao servidor em busca de uma resposta. Mas em alguns casos, o servidor também pode enviar uma requisição ao cliente

UniFOA
Centro Universitário de Volta Redonda



Exercícios

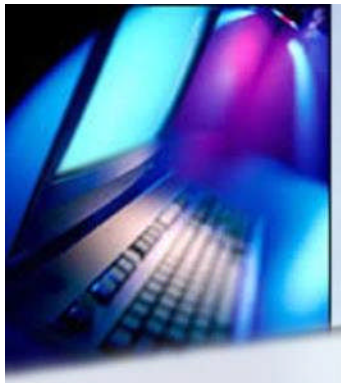
- **Comunicação em HTTP**
- Qual dessas alternativas é verdadeira?

( ) - Em HTTP o servidor sempre envia uma requisição ao cliente para poder alterar algo na tela.

(X) - Uma comunicação com HTTP sempre é iniciada pelo cliente que manda uma requisição ao servidor esperando por uma resposta.

( ) - Quando trabalhamos com HTTP, a comunicação é sempre iniciada pelo lado do cliente que envia uma requisição ao servidor em busca de uma resposta. Mas em alguns casos, o servidor também pode enviar uma requisição ao cliente

UniFOA
Centro Universitário de Volta Redonda



Exercícios

- Pesquisa na internet as categorias de domínios no Brasil
- Quanto custa registrar um domínio o Brasil
- Qual o órgão responsável pelo registro e manutenção dos nomes de domínios que usam o <.br> , e a distribuição de números de Sistema Autônomo (ASN) e endereços IPv4 e IPv6 no País