UFU/FACOM
Disciplina: GBC025
Ref: Trabalho de Programação

Objetivo: assimilar conceitos básicos para construção de um servidor web em Prolog

Desenvolvimento: o procedimento a ser seguido consiste na realização de vários passos na ordem especificada. Tais passos são descritos a seguir. Contudo antes de se descrever os passos a serem executados iremos descrever o processo que permite a comunicação entre computadores objetivando mostrar como uma página web armazenada num computador poda ser visualizada em outro computador. É também objetivo deste texto apresentar noções básicas para criação de páginas web.

No que diz respeito a WWW a maioria das páginas são escritas numa mesma linguagem, chamada HTML, e são transmitidas pela internet através de um protocolo denominado HTTP (*hypertext transfer protocol*). O protocolo HTTP permite que um computador executando um sistema operacional, como o Windows, possa "conversar" com um computador funcionando com um sistema operacional igual ou diferente tal como o LINUX. Usando um navegador (ou *browser*) para internet - tal como o internet explorer, que interpreta conteúdo especificado em HTML - é possível visualizar no monitor o conteúdo HTML de uma forma tal que possamos entendê-lo.

Embora falando a mesma linguagem os vários dispositivos acessando a web devem obedecer a certas regras para que a comunicação aconteça. É como levantar o braço na sala de aula para poder fazer alguma pergunta para o professor. O protocolo HTTP especifica tais regras para que a comunicação ocorra. Quando você está interessado em ver uma determinada página web no seu computador (chamado de computador ou máquina cliente) inicia-se uma comunicação com o computador responsável pelo conteúdo daquela página (o computador servidor ou simplesmente servidor). O servidor é o computador onde o web site "mora". Quando você digita um endereço no seu navegador o servidor recebe seu pedido, encontra a página web desejada e a envia para seu computador de tal modo que você possa vê-la.

Vejamos agora com mais detalhes o processo de comunicação cliente/servidor.

- Todo pedido/resposta começa pela digitação de um endereço URL (*Universal Resource Locator*) na barra de endereços do seu navegador (ou browser). Eis um exemplo de URL: www.facom.ufu.br. Na verdade os navegadores não usam URL´s e sim endereços IP (I*nternet Protocol*). Você pode fazer uma analogia de IP´s com números de telefones ou endereços postais. Por exemplo, o endereço IP de www.ufu.br é 192.168.3.2.

- Tente abrir um navegador e digite http://www.facom.ufu.br no campo de especificação de endereços. Abra uma nova aba, digite http://192.168.3.2/ e veja o conteúdo exibido. Você verá o mesmo conteúdo para ambas as especificações. A URL http://www.facom.ufu.br está servindo como um apelido para http://192.168.3.2/, mas por que? Como? Ora é mais fácil nos lembrarmos de palavras do que de números. O sistema encarregado de converter palavras em números é denominado DNS (*Domain name system*), que é uma "lista" de todas os computadores conectados na internet. Quando se digita http://www.facom.ufu.br este URL é enviado para o servidor de nomes (o sistema DNS) que tenta associá-lo a um endereço IP. Existe uma quantidade enorme de computadores conectados a internet e nem

todo servidor DNS contém uma listagem de toda máquina que está online. Assim sendo, existe um processo em que o servidor DNS apropriado pode ser alcançado e devolver o valor de IP correspondente. O sistema DNS busca pelo endereço IP correspondente a www.facom.ufu.br, encontra o valor 192.168.3.2, que é então devolvido para o seu *browser*. Seu computador então envia o pedido para o computador especificado pelo IP e fica esperando uma resposta. Se tudo estiver correto o servidor envia uma mensagem de volta para o cliente seguido pela página web. Se algo der errado, por exemplo, URL digitado incorretamente, recebe-se um erro HHTP, no caso a famigerada mensagem "page not found". Algumas vezes os desenvolvedores web fazem com que uma dada página web seja retornada quando uma página desejada não existe.

Existem diferentes tipos de conteúdo que podem ser vistos na internet. Vamos considerar neste texto os seguintes tipos: texto (poderá ser visto no passo 1) e HTML (poderá ser visto no passo 2) e em outros lugares. Uma vez tendo conhecimento do processo de comunicação entre computadores vamos dar início a descrição dos passos para que você possa executá-los na sequência especificada.

Passo 1: servindo conteúdo texto.

Neste passo criaremos um servidor web que mostra uma simples página quando um determinado endereço é especificado. A página a ser acessada é especificada por `/minha_pagina`. Quando acessada uma mensagem de texto simples (`plain-text`) é mostrada. Para programarmos um servidor web que apresente tal página devemos especificar três elementos:

1. Devemos declarar um handler, elemento que ligará o caminho HTTP a um predicado prolog. Isto é feito pela declaração `:- http_handler(root(minha_pagina), mensagem, []).` Quando acessarmos a página `minha_pagina` o predicado `mensagem` será executado.
2. É necessário também termos um predicado que quando executado faz com que o servidor web fique ativo para receber nossos pedidos. No nosso programa esta responsabilidade pertence ao predicado `servidor`.
3. Finalmente faz-se necessária a implementação da página, que consiste na definição do predicado `mensagem`. O argumento de `mensagem` diz respeito a detalhes do pedido, o que será ignorado por enquanto. Nossa tarefa é escrever um documento CGI que consiste de um número de linhas formadas por pares caracterizados por *name: value* seguidos por dois caracteres /n e depois pelo conteúdo do documento. A única linha obrigatória representa um cabeçalho formado pelo par `Content-type:` <mime-type>. O conteúdo para este nosso primeiro programa consiste de texto simples a ser escrito. A escrita pode ser feita usando qualquer predicado prolog com este fim, como por exemplo `write/1`. Usaremos um outro predicado: o predicado `format/2`.

Para executarmos o servidor coloque o código que segue num arquivo.

```
:- use_module(library(http/thread_httpd)).
:- use_module(library(http/http_dispatch)).

:- http_handler(root(minha_pagina), mensagem, []).

servidor(Porta) :-
        http_server(http_dispatch, [port(Porta)]).

mensagem(_Request) :-
        format('Content-type: text/plain~n~n'),
        format('Seja bem-vindo!~n').
```

Depois de informar este código a Prolog execute a seguinte consulta:

```
?- servidor(5000).
```

O número 5000 é o número da porta. Na verdade pode ser qualquer número entre 1001 e 65000 (depende de seu sistema operacional). O próximo passo é digitar o endereço http://localhost:5000/minha_pagina no seu navegador e visualizar o conteúdo.

Passo2: servindo conteúdo HTML

Um conteúdo HTML pode ser escrito da mesma como fizemos no passo anterior. A diferença consiste em especificar `Content-type: text/html` e usar format/2 para escrever as tags HTML.Observe o código especificado a seguir.

```
:- use_module(library(http/thread_httpd)).
:- use_module(library(http/http_dispatch)).
:- http_handler(root('minha_pagina.html'), mensagem, []).
servidor(Porta) :-
        http_server(http_dispatch, [port(Porta)]).
mensagem(_Request) :-
      format('Content-type: text/html~n~n'),
      format('<html>~n'),
      format('<head>~n'),
      format('<title>~n'),
      format('Página de Carlos Lopes~n'),
      format('</title>~n'),
      format('</head>~n'),
      format('<body>~n'),
      format('<p> Seja bem-vindo! </p>~n'),
      format('</body>~n'),
      format('</html>').
```

Observe que escrever um conteúdo neste formato é trabalhoso, o que pode levar a erros. Uma alternativa é usar `library(http/html_write)`, que traduz termos Prolog em HTML. Vamos começar com um pequeno exemplo:

```
:- use_module(library(http/thread_httpd)).
:- use_module(library(http/http_dispatch)).
:- use_module(library(http/html_write)).

:- http_handler(root(minha_pagina), mensagem, []).
servidor(Porta) :-
        http_server(http_dispatch, [port(Porta)]).
mensagem(_Pedido) :-
```

```
reply_html_page(title('Página de Carlos Lopes'),
                [ h1('Seja bem-vindo'),
                  p(['Este exemplo demonstra o processo de geração de ',
'conteúdo HTML'])
                ]).
```

O predicado `reply_html_page/2` recebe uma descrição da cabeça e corpo da página HTML e a repassa para `html/1`. O argumento para `html/1` é um simples termo ou uma lista de termos. O nome do funtor de cada termo é uma tag HTML. O funtor pode receber um ou dois argumentos. Se apresenta um argumento então tal argumento é o conteúdo HTML, que é um termo ou uma lista de termos. Caso especifique dois argumentos o primeiro especifica os atributos e o segundo o conteúdo. Existe uma exceção para esta regra: se uma tag HTML não tem conteúdo, como por exemplo a tag <img>, o primeiro argumento corresponde aos atributos.

A seguir dados alguns exemplos para ilustrar a descrição feita anteriormente

```
Exemplo 1:
      span(class(product_class), 'Computers')

Exemplo 2:
      img([width(32),heigth(32),src('/icons/computer.png')])

Exemplo 3:
      table([ tr([ td('celula 1a'), td('celula 1b')]),
              tr([ td('celula 2a'), td('celula 2b')])
            ])
```

O exemplo 1 mostra a especificação de um atributo. O exemplo 2 mostra que `img` tem somente um atributo que é interpretado como conteúdo. Exemplo 3 mostra lista de termos.

Os métodos que usamos até o momento destinam-se a construir páginas web dinâmicas. Páginas estáticas são mais bem geradas como arquivos de texto. Páginas com grande conteúdo estático e pouca informação podem ser construídas pelo uso de PWP. PWP será mostrada adiante. PWP é uma resposta baseada em prolog para PHP, ASP, JSP etc.

Passo 3: Processando parâmetros HTTP

O tratamento de parâmetros em páginas é feito por `http_parameters/2`. O primeiro parâmetro é o pedido passado pelo servidor. O segundo argumento é uma lista de `Nome(Valor, Propriedades)`. `Nome` refere-se à identificação dada um controle enquanto que `Valor` corresponde ao valor digitado pelo usuário relacionado ao controle especificado por `Nome`. No exemplo do formulário em HTML relativo a um pedido de pizza tínhamos a seguinte especificação:

```
...
<p><label>Nome do Cliente: <input name="nomeCliente" ></label></p>
...
```

Pare este exemplo `nomeCliente( carlos,[])` será um dos elementos da lista que aparecerá no segundo argumento de `http_parameters/2` admitindo que no campo Cliente o usuário tenha digitado `carlos`. As `Propriedades` permitem especificar o tipo, se o parâmetro é opcional, etc. Para ilustrar como o processamento de parâmetros é feito considere a página HTML especificada anteriormente contendo um formulário. Tal página é mostrada a seguir.

```
<!DOCTYPE HTML>
<html lang="pt-br">
<head>
  <title>Pizzas Qloucura</title>
 </head>

<body>

<form method="post"
      enctype="application/x-www-form-urlencoded"
      action="http://localhost:8000/processa_pedido">
 <p><label>Nome do Cliente: <input name="nomeCliente" ></label></p>
 <p><label>Telefone: <input type=tel name="telCliente"></label></p>
 <p><label>E-mail: <input type=email name="emailCliente"></label></p>
 <fieldset>
  <legend> Tamanho da Pizza </legend>
  <p><label> <input type=radio name="tamanho" value="p"> Pequena </label></p>
  <p><label> <input type=radio name="tamanho" value="m"> Média </label></p>
  <p><label> <input type=radio name="tamanho" value="g"> Grande </label></p>
 </fieldset>
 <fieldset>
  <legend> Ingrediente/legend>
  <p><label> <input type=checkbox name="ing" value="tom"> Tomate </label></p>
  <p><label> <input type=checkbox name="ing" value="pre"> Presunto</label></p>
  <p><label> <input type=checkbox name="ing" value="ceb"> cebola </label></p>
  <p><label> <input type=checkbox name="ing" value="que"> Queijo </label></p>
 </fieldset>
 <p><label>Horário de entrega desejado: <input type=time min="11:00" max="21:00"
step="900" name="tempo"></label></p>
 <p><label>Instruções para entrega: <textarea name="obs"></textarea></label></p>
 <p><button>Enviar Pedido</button></p>
</form>

</body>
</html>
```

O processamento envolve, por meio de um programa prolog, manipular os dados inseridos nos campos do formulário. No código prolog que segue simplesmente fazemos acesso aos parâmetros e os mostramos ao usuário através de uma página com conteúdo textual.

```
:- use_module(library(http/thread_httpd)).
:- use_module(library(http/http_dispatch)).
:- use_module(library(http/http_parameters)).
:- use_module(library(uri)).


:- http_handler(root(processa_pedido), mensagem, []).
:- dynamic(nomeCliente/2).
:- dynamic(telCliente/2).


servidor(Porta) :-
        http_server(http_dispatch, [port(Porta)]).


mensagem(R) :-
      format('Content-type: text/plain~n~n'),
```

```
format('Listagem dos dados!~n'),
http_parameters(R,[nomeCliente(C,[]),
                   telCliente(T,[]),
                   emailCliente(E,[]),
                   tamanho(S,[]),
                   ing(I,[]),
                   tempo(H,[]),
                   obs(O,[])
                   ]),
write(nome(C)),nl,
write(telefone(T)),nl,
write(email(E)),nl,
write(tamanho(S)),nl,
write(ingrediente(I)),nl,
write(tempo(H)),nl,
write(obs(O)),nl.
```

Observe que caberá à página `processa_pedido` fazer o processamento dos dados e gerar uma página em formato de texto mostrando os dados digitados pelo usuário. Note também que ao invés de usar `format` optamos por `write` para escrever os valores digitados pelo usuário.

Para observar o resultado inicialmente consulte o arquivo contendo o código prolog. Em seguida no prompt prolog o seguinte:

?- servidor(8000).

Agora abra a página contendo o formulário, insira os dados e clique em `enviar pedido`. Você verá que uma nova página aparecerá contendo os valores digitados pelo usuário.

Tarefa 1: Refaça o passo anterior informando os dados digitados por meio de uma página HTML.

Tarefa 2: baseado no exemplo do pedido de pizzas, crie uma página web que contenha um menu com dois itens:

- cadastrar pedido
- cancelar pedido

Ao selecionar `cadastrar pedido`, a página com formulário para especificação do pedido deverá ser mostrada ao visitante. Uma vez submetido o pedido você deverá guardá-lo em um arquivo de nome `pedidos.pl` que deverá ter a descrição do pedido especificada da seguinte forma, como mostrado no exemplo a seguir:

```
pedido(manoel,3231-2231,mane@ufu.br,p,[pre,tom],12:00,bem quentinha).
```

Informe ao visitante que o pedido foi feito com sucesso informando a ele o pedido feito. Ao selecionar `cancelar pedido`, sua tarefa é eliminar o pedido do arquivo `pedidos.pl` e informar ao visitante que o cancelamento foi feito com sucesso.