{sem mágicas}

17.1 O QUE VAI PARA O SERVIDOR

Talvez muitos de vocês tenham tentado entender o que está programado no MAGIC.TXT e não tenha conseguido.Realmente tentar "ler" um programa em PERL não é coisa fácil, principalmente quando desenvolvido por experts que usam vários "truques" da linguagem.

Vamos procurar mostrar nesta aula final aquilo que é o mais importante naquele programa que é como é feita a "leitura" do que é enviado pelo usuários de uma página com FORM TAGS para o programa que roda no servidor.

Se você tiver numa página as FORM TAGS preenchidas seguintes:

Nome: Maria Pia

Endereço: Rua Alfa,57

Gravar

Caso os "names" dos controls de input sejam: **nome** e **end**, o que será enviado para o servidor após a **?** será:

```
nome=Maria+Pia&end=Rua+Alfa%2C57
```

O signo **&** separa cada **dupla**.

O signo **+** aparece no lugar de **espaço/branco**

O mais dificil de entender é aquele **%2C**. Vamos explicar.

Em ASCII cada caractere pode ser representado por um número decimal. O "caractere" **vírgula** por exemplo é representado por: 44

Um número decimal pode ser convertido para a "base" hexadecimal (recorde seus princípios básicos de computação...). O decimal 44, por exemplo, será em hexadecimal:2C.

O CGI transmite determinados caracteres (vírgula é um deles) por um número que é o hexadecimal correspondente ao número que o representa em ASCII. Para mostrar que está fazendo isto, coloca antes do número o sinal **%**.

Por isto é que nossa simples vírgula separando **Rua Alfa** de **57** aparece na transmissão como **%2C**.

Graças a Deus existe uma maneira de se substituir esta representação pelo caractere própriamente dito. Se tivermos numa variável:

```
$carac=%2C
```

podemos colocar propriamente o sinal de vírgula na variável usando:

```
$carac=~s/%(..)/pack("c",hex($1)/
```

Como você deve se lembrar das aulas 12 e 13

```
s/%(..)/XXX/ge
```

substituiria os conjuntos de "% mais os dois dígitos seguintes" por XXX. O uso de **ge** faz com isto aconteça em todas as ocorrências deste conjunto.

No caso do que é transmitido pelo CGI visto acima, usando

```
s/%(..)/pack("c",hex($1)/
```

conseguimos que o conjunto **%2C** seja substituido pela transformação do que vem depois de % de hexadecimal (2C) para decimal (44) e depois por sua representação como caractere ASCII não-numérica: a dita cuja vírgula.

17.2 AS TRANSFORMAÇÕES TODAS

Como vocês veem temos que fazer várias transformações para colocar o que é recebido pelo programa que está no servidor em condições de ser usado.

Para entender as linhas de programação que fazem estas transformaçõpes recomendamos que você releia com cuidado as aulas 12, 13 e aquela sobre arrays (a aula 10) e a que fala sobre split (aula 15) e a que fala sobre loops **for** (aula 11) - enfim: você tem que ter estudado direito o curso.... Na aula 10 comentamos sobre os arrays associativos e dissemos que o que é enviado para o servidor vem num item de um array associativo %ENV e que pode ser lido com:

```
$ENV{"QUERY_STRING"}
```

Isto no caso de usarmos GET (e não POST) , o que é uma das premissas em todos os cursos aqui da DMU.(Você ainda se lembra do que é GET? Isso foi comentado na aula 3). O GET tem umas limitações mas é mais fácil de usar.

O conjunto de linhas de código que colocamos num programa caso não queiramos usar o MAGIC.TXT, será:

```
$temp=$ENV{"QUERY_STRING"};
@pares=split(/&/,$temp);# faz split criando array com duplas
for($i=0;$i<=$#pares;$i++){
 ($chave,$valor)=split(/=/,$pares[$i]);# começa a criar array associativo
 $valor=~tr/+/ /;#substitui caractere + por branco
 $valor=~s/%(..)/pack("c",hex($1))/ge;
 $campos{$chave}=$valor;
```

Depois de executadas estas linhas de código temos o array associativo %campos e poderemos "ler" por exemplo o campo **nome** (do nosso exemplo acima) usando

```
$campos{'nome'};
```

IMPORTANTE: use apóstrofos dentro da chave e não aspas!!!!

17.3 TÁ FALTANDO COISA!

É claro que se eu não uso o MAGIC.TXT, tenho que construir "na unha" o header e as outras

tags do HTML, como visto na aula 1.

17.4 ENFIM UM EXEMPLO/DEMO

Vamos então mostrar um exemplo completo (com os dois programas - HTML e PERL) que lê dados de NOME e ENDEREÇO e os reescrevem numa nova página.

O programa HTML você pode ver aqui

E o programa PERL você pode ver aqui

Para executar a demo use as FORM TAGS abaixo:

Nome:

Endereço:

Enviar

Espero que tenham aproveitado o curso. Mas não deixem de fazer os cursos de Java, no qual usaremos PERL para gravação de arquivos e applets Java para fazer consistência de entradas (o que com PERL não recomendamos).