

Programação II

Formulários



Formulários

```
<form action="arq.php" method="tipoEnvio">
```



para onde os dados do formulário serão submetidos.



como os dados do formulário serão submetidos.

Métodos GET e POST

- **GET**

- Os dados do formulário são enviados ao servidor pela URL da página, como pares nome/valor:

`nome_campo1=valor_campo1&nome_campo2=valor_campo2&...`

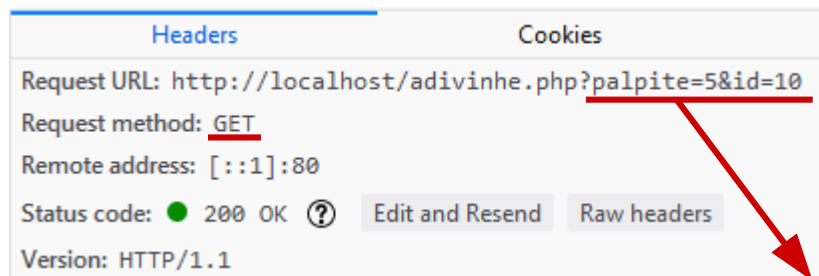
- Ao serem recebidos pelo servidor através do protocolo HTTP, são armazenados numa variável de ambiente do sistema operacional denominada QUERY_STRING.
- Uma URL de uma página que recebe dados via GET fica no seguinte formato:

`http://www.site.com.br/calcula.php?nome=fulano&ano=1990`

onde o caracter “?” separa o nome da página da cadeia de caracteres vinda do formulário. Os nomes dos campos são associados aos seus valores através do sinal de “=”, e são separados uns dos outros pelo caracter “&”.

Métodos GET e POST

- GET



Headers Cookies

Request URL: http://localhost/adivinhe.php?palpite=5&id=10

Request method: GET

Remote address: [::1]:80

Status code: 200 OK ? Edit and Resend Raw headers

Version: HTTP/1.1



Headers Cookies Params

Filter request parameters

Query string

id: 10

palpite: 5

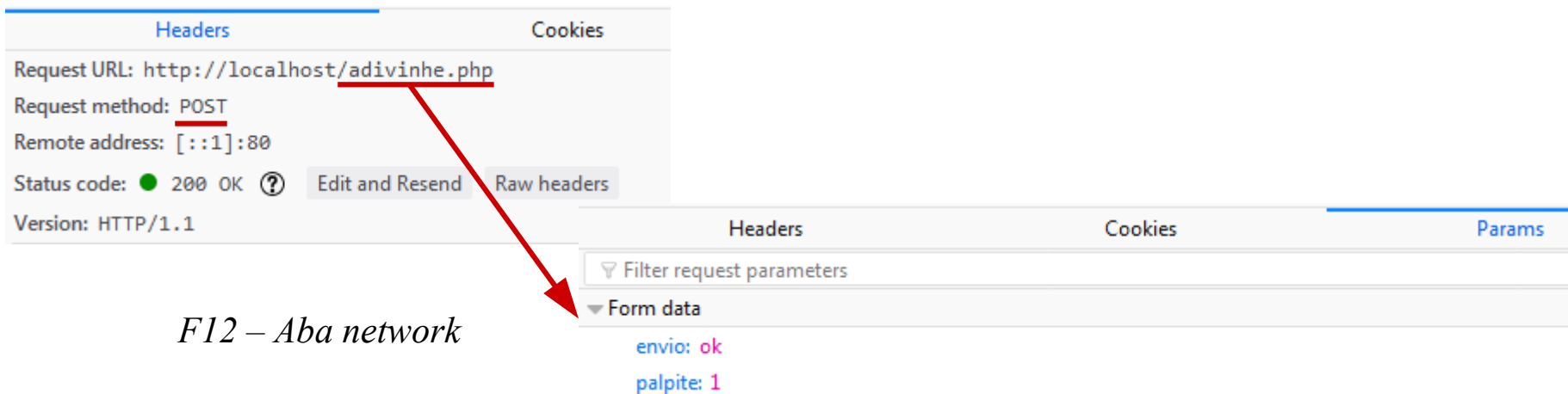
F12 – Aba network

- Como as informações são passadas pela URL, todos os dados podem ser vistos pelo usuário (inclusive valores de campos do tipo password, com senhas).
- O PHP armazena as variáveis recebidas pelo GET num vetor superglobal chamado **`$_GET`**.
- Para acessar determinada variável, basta colocar seu nome no índice do vetor: **`$_GET['nome']`**, **`$_GET['ano']`**, etc.

Métodos GET e POST

- **POST**

- O método POST envia os dados do formulário encapsulados dentro do corpo da mensagem, em uma área chamada FORM DATA. Portanto, os dados não ficam visíveis através da URL.



- O PHP armazena as variáveis recebidas pelo POST num vetor chamado **\$_POST**. Para acessar determinada variável, basta colocar seu nome no índice do vetor: **\$_POST['nome']**, **\$_POST['ano']**, etc.

Qual escolher?

- Utilize **POST** quando os dados estão sendo criados ou modificados
 - Ex: cadastros (insert), alterações (update)
- Utilize **GET** quando o conteúdo está sendo lido ou buscado
 - Ex: resultados de busca, exibição de dados vindos do BD (select)
 - A mesma URL, com os mesmos parâmetros GET, deve prover o mesmo resultado quando acessada. Isso permite compartilhar ou salvar nos favoritos um link para aquele item específico (ex: produto específico em uma loja, notícia, etc)
 - Além disso, os indexadores dos mecanismos de busca seguem URLs com GET (o que não ocorre com POST)

PHP e formulários

- Cada elemento (INPUT, SELECT ou TEXTAREA) de um formulário HTML submetido a um script PHP, cria no escopo deste uma variável:
 - o identificador da variável é o nome do elemento (seu atributo “name”)
 - o valor é o dado digitado/selecionado pelo usuário.
- Estas variáveis ficam armazenadas dentro de vetores (arrays) superglobais do PHP:
 - `$_POST['nomeCampo']`: armazena dados submetidos via POST;
 - `$_GET['nomeCampo']`: armazena dados submetidos via GET ou passados pela URL (*query string*);
 - `$_REQUEST['nomeCampo']`: é uma espécie de atalho, que contém os valores de `$_POST`, `$_GET` e `$_COOKIE`. É útil para páginas que recebem dados de dois ou mais locais por diferentes métodos;
 - `$_FILES['nomeCampo']`: armazena arquivos enviados por upload.

Campos Textuais

- Os valores das variáveis relativas a campos abertos à digitação (inputs do tipo *text*, *password*, *email*, etc. e *textarea*) corresponde ao valor digitado pelo usuário no respectivo campo, e poderá ser acessado no array `$_POST`, `$_GET` ou `$_REQUEST` por meio do nome especificado no atributo "name".
 - *OBS: Considere, para todos os exemplos a seguir, que o formulário utiliza POST.*
- Os campos:

```
<input type="text" name="nome" size="20">
```

ou

```
<textarea name="nome"></textarea>
```
- serão acessados no PHP como `$_POST['nome']` (ou entre chaves, `{$_POST['nome']}`, se a variável for utilizada dentro de um contexto de aspas duplas).
 - No caso do *textarea*, para que as quebras de linha sejam convertidas em tags `
`, o PHP dispõe da função `nl2br()`

Campos Radio

- Todos os radio que pertencem ao mesmo grupo deverão ter o mesmo valor no atributo **name**.
- O valor recebido pelo PHP será o valor do atributo **value** do radio selecionado.

Selecione a operação:


```
<input name="op" type="radio" value="ad">Adição<br>
<input name="op" type="radio" value="sub">Subtração<br>
<input name="op" type="radio" value="mult">Multiplicação<br>
<input name="op" type="radio" value="div">Divisão<br><br>
```

mesmo name

diferentes value

Selecione a operação:

- ☒ Adição
- ☐ Subtração
- ☐ Multiplicação
- ☐ Divisão

- No exemplo acima, como os 4 radio possuem o mesmo nome "op", o PHP recebe a variável **\$_POST['op']**, contendo o value do item selecionado.
 - Como apenas um dos botões pode estar selecionado, será um único valor.
 - Se nenhuma opção for selecionada, **\$_POST['op']** não será criada.

Campos Checkbox

- O PHP receberá o valor de um checkbox apenas se este tiver sido marcado pelo usuário.
- No exemplo abaixo, se o usuário assinalar o checkbox e submeter o formulário, o PHP criará a variável `$_POST['concordo']` com o valor "sim":

```
<label>  
<input name="concordo" type="checkbox" value="sim">  
Li e estou de acordo com os termos de uso do site  
</label>
```

- Para verificar no PHP se a opção foi ou não marcada, podemos utilizar a função `isset`:


```
if(!isset($_POST['concordo'])) {  
    echo "É necessário concordar com os termos de uso!";  
}
```

- OBS: Se um checkbox não tiver o atributo `value`, seu valor quando marcado será "on"

Campos Checkbox

- Quando há vários checkboxes que pertencem a um mesmo grupo de opções, especialmente quando gerados dinamicamente, não temos como prever quantos checkboxes serão criados, nem quantos serão marcados. A verificação individual com `isset` se torna inviável.
- Por isso, podemos nomear todos os checkboxes como um array. Assim, apenas os values das opções que forem marcadas serão recebidas pelo PHP num vetor, onde poderemos então acessá-los.

```
Selecione as disciplinas nas quais deseja matricular-se:<br><br>
<input name="disciplina[]" type="checkbox" value="Banco de Dados II">Banco de
  Dados II<br>
<input name="disciplina[]" type="checkbox" value="Programação II">Programação
  II<br>
<input name="disciplina[]" type="checkbox" value="TCC I">TCC I<br>
<input name="disciplina[]" type="checkbox" value="Sistemas Operacionais">Sistemas
  Operacionais<br>
<input name="disciplina[]" type="checkbox" value="Algoritmos">Algoritmos<br>
<input name="disciplina[]" type="checkbox" value="Estatística Básica">Estatística
  Básica<br><br>
```



[] indica que os checkboxes marcados farão parte de um array com este nome

Campos Checkbox

Selecione as disciplinas nas quais deseja matricular-se:

- ☒ Banco de Dados II
- ☒ Programação II
- ☐ TCC I
- ☒ Sistemas Operacionais
- ☐ Algoritmos
- ☒ Estatística Básica

<h2>Matrícula realizada com sucesso</h2>

```
<?php
```

```
    if (isset($_POST['disciplina'])) {  
        echo "Você escolheu as seguintes disciplinas:<br><br>";  
        foreach($_POST['disciplina'] as $selecionada) {  
            echo $selecionada;  
            echo "<br>";  
        }  
    }
```

```
?>
```

Campos Select

- Como os campos Select só permitem selecionar um elemento, só receberemos no PHP uma variável com o nome do campo (`$_POST['cidade']`, no exemplo abaixo), contendo o “value” da opção selecionada (ou o que estiver entre `<option>` `</option>`, caso não sejam definidos os atributos “value” das opções).
- Para deixar uma das opções previamente selecionada, utilize o atributo vazio `selected`.

```
<select name="cidade">
  <option value="xap">Chapecó</option>
  <option value="xe" selected>Xanxerê</option>
  <option value="xax">Xaxim</option>
</select>
```

se o value for omitido, será enviado o
valor do conteúdo do elemento `<option>`

Selecione sua cidade:

Chapecó	▼
Chapecó	
Xanxerê	
Xaxim	