Universidade Federal do Espírito Santo
Centro de Ciências Agrárias – CCA UFES
Departamento de Computação

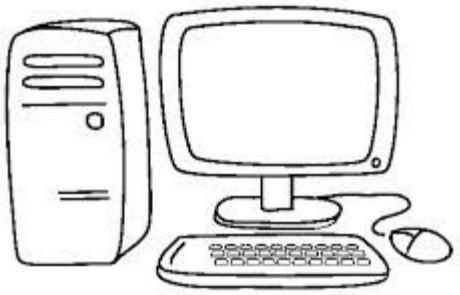
Trabalhando com PHP

Desenvolvimento de Sistemas para WEB

Site: http://jeiks.net
E-mail: jacsonrcsilva@gmail.com

Formulários

Formulários: HTML ↔ PHP



Cliente
Possui a página HTML com o formulário.

Envia as informações ao servidor

Servidor
Possui a página que receberá as informações do formulário

Processa as informações do formulário. Gera a resposta

Fornece a resposta para o Cliente

Formulários HTML

- Formulários HTML são utilizados para enviar dados para o servidor.
- Eles podem conter muitos campos, como:
  - Texto;
  - Checkboxes;
  - Listas de seleção;
  - etc.;
- A tag utilizada é a *form*:

```
<form>
   elementos de entrada
</form>
```

Formulários HTML

- Os principais atributos dos formulários são:
  - **action**: especifica onde enviar os dados quando um formulário é submetido;
  - **method**: especifica o método HTTP que será utilizado para enviar o formulário;
  - **name**: especifica o nome do formulário
  - Mais atributos em: <http://www.w3schools.com/tags/tag_form.asp>
- Exemplo:

```
<form action="destino.php" method="GET" name="fnome">
  Primeiro nome: <input type="text" name="pNome"><br>
  Último nome: <input type="text" name="uNome"><br>
  <input type="submit" value="Enviar">
</form>
```

Métodos de envio

- GET:
  - Anexa os dados do formulário na URL em pares de nome/valor. Ex:
    - http://jeiks.net/processa.php?nome=jeiks&senha=123
    - Por isso, nunca deve-se usar GET para enviar dados confidenciais!
  - O comprimento de um URL é limitado (cerca de 3000 caracteres);
  - Útil para submissões simples.
  - GET é melhor para dados que não precisam de proteção.
- POST:
  - Anexa os dados do formulário dentro do corpo da solicitação HTTP (os dados não são mostrados na URL);
  - Não tem limitações de tamanho;
  - POST deve ser utilizado para dados que precisam demais proteção.

Recebendo os formulários – PHP

- Para receber os dados do formulário, o PHP possui arrays que são variáveis superglobais:
  - $_POST;
  - $_GET;
  - $_REQUEST;
- Exemplo:

```
<p>Dados obtidos com POST</p>
<pre><?php print_r($_POST); ?></pre>

<p>Dados obtidos com GET</p>
<pre><?php print_r($_GET); ?></pre>

<p>Dados obtidos com REQUEST</p>
<pre><?php print_r($_REQUEST); ?></pre>
```

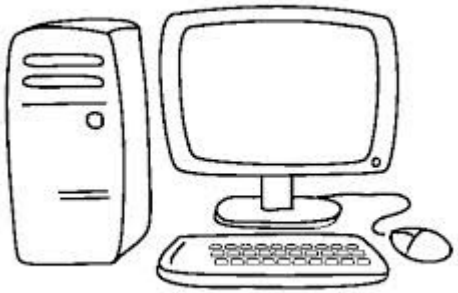
Hora de praticar.

Abram e estudem os exemplos disponíveis em:
formularios.zip

Banco de Dados

Banco de Dados HTML e PHP

Obtém as informações do banco de dados



Cliente
Possui a página HTML com o formulário.

Envia a requisição ao servidor

Servidor
Possui a página que receberá a requisição

Fornece a resposta para o Cliente

Processo de comunicação

- Ocorre em etapas:
  1. Conectar ao banco de dados:

```
$con = mysqli_connect(   "servidor", "usuario",
                         "senha" , "nome_do_banco" );
```

  2. Efetuar a pesquisa no banco de dados:

```
$query = "SELECT * FROM ÁtabelaÁ";
$resposta = mysqli_query( $con, $query );
```

  3. Percorrer a resposta do banco de dados:

```
while ( $linha = mysqli_fetch_array($resposta) )
   echo "<li>($linha[0]) $linha[1]";
```

  4. Encerrar a conexão ao banco de dados:

```
mysqli_close( $con )
```

Mão na massa

Estudem o exemplo: mysql.php

Sessões em PHP

O que é uma sessão

- Suporte a sessões no PHP:
  - maneira de preservar dados através de acessos subsequentes.
- Características:
  - permite a criação de aplicações mais personalizadas;
  - permite que certas características do site sejam apresentadas somente aos usuários pertencentes a determinada sessão.
- É determinada através de uma variável superglobal ($_SESSION) chamada “variável de sessão PHP”.

Variável de sessão PHP

- É utilizada para
  - Armazenar informações sobre a sessão;
  - Modificar configurações da sessão de um usuário.
- Armazenar informação sobre um único usuário;
- Permanecer disponível para todas as páginas de uma aplicação web.
- Ela trabalha criando uma única identificação (UID) para cada usuário e pode armazenar outras variáveis/informações utilizando essa UID.

Sessões

- Permitem trabalhar como em *desktops*:
  - Abrir um aplicativo associado a um usuário;
  - Utilizar o aplicativo;
  - Fechar o aplicativo.
- As sessões permitem armazenar informações sobre o usuário no servidor para utilizar depois.
- Porém, são temporárias, ou seja, terminam assim que o site é abandonado.

Criando uma sessão

```
<?php
session_start();

if( isset( $_SESSION[©views©] ) )
   $_SESSION[©views©]++;
else
   $_SESSION[©views©]=1;

echo "Session ID=".session_id()."<br/>";
echo "Visualizacoes=".$_SESSION[©views©];
?>
```

Após criar a sessão, também é necessário utilizar essa instrução para acessar as variáveis que foram declaradas.

Destruindo uma sessão

- Apagando uma variável de sessão:

```
<?php
session_start();
if(  isset( $_SESSION[©views©] )  )
  unset( $_SESSION[©views©] );
?>
```

- Destruindo toda a sessão:

```
<?php
session_destroy();
?>
```

Mãos a obra.

Crie o php com o código do slide 18 e execute-o. Após isso, insira mais variáveis e teste sua sessão.

Cookies

Cookies

- São formas utilizadas por website para identificar um usuário.
- Com isso, consegue salvar informações específicas para que este usuário seja tratado de uma forma diferente na próxima vez que entrar no site:
  - Cor de fundo do site;
  - Pesquisas realizadas;
  - Nick utilizado no chat;
  - etc.
- Assim, na próxima vez que entrar no site, ele abrirá essas variáveis e saberá como tratar sua visita.
  - Geralmente os sites mantém um campo chamado “Lembre-se de mim” ou “Remember Me” para isso.

Cuidados de segurança

- Suponha que um cookie possua as seguintes variáveis:
  - Usuario;
  - Nivel_Acesso;
- Se o usuário modificar o valor do Nivel_Acesso e entrar no site de novo, ele pode se tornar um administrador.
- Em casos onde a sessão seja gravada no cookie, uma pessoa pode roubar essa informação e acessar a conta da pessoa em outro computador.

Criando um cookie

setcookie(name, value, expiration,
path, domain, secure, httponly);

- Name: nome que seu cookie terá no computador do usuário;
- Value: valor que será armazenado no cookie;
- Expiration: define quando o cookie vai expirar (ser apagado);
- Path: O caminho no servidor onde o cookie estará disponível;
- Domain: O domínio para qual o cookie estará disponível;
- Secure: Indica que o cookie só poderá ser transmitido sob uma conexão segura HTTPS do cliente;
- Httponly: Quando for TRUE, o cookie será acessível somente sob o protocolo HTTP.

Exemplos

```
<?php
  $value = ©alguma coisa de algum lugar©;

  setcookie("CookieTeste", $value);
  /* expira em 1 hora */
  setcookie("CookieTeste", $value, time()+3600);
  /* utilizando caminho e domínio */
  setcookie("CookieTeste", $value, time()+3600,
                  "/~usuario/", ".exemplo.com", 1);
?>
```

Obtendo os valores do cookie

- Os valores são obtidos automaticamente.
- Para acessá-los, basta utilizar a variável $_COOKIE:

```
<?php

// Mostra um cookie individual
echo $_COOKIE[ "CookieTeste" ];
echo $HTTP_COOKIE_VARS["CookieTeste"];

// Outra maneira de testar é vendo todos os cookies:
print_r( $_COOKIE );

?>
```

Cookies e Arrays

```
<?php
// envia os cookies
setcookie("cookie[tres]", "cookie tres");
setcookie("cookie[dois]", "cookie dois");
setcookie("cookie[um]", "cookie um");

// Depois que a página recarregar, mostra eles

if (  isset( $_COOKIE["cookie"] )  ) {
    foreach ($_COOKIE["cookie"] as $nome => $valor ) {
        echo "$nome : $valor <br />\n";
    }
}
?>
```

Qual utilizar: sessão ou cookies?

Qual utilizar: sessão ou cookies?

Os cookies são mais fáceis de serem modificados do que as sessões.

Exemplo de uma função login

```
function login($username, $password) {
  $query = "SELECT id, user_level FROM users WHERE ".
         "password = ©$password© AND ".
         "username = ©$username© LIMIT 1";
  $sql = mysql_query( $query );
  if($sql === false) return false;
  else {
         $resp = mysql_fetch_array( $sql );
         session_regenerate_id(true);

         $_SESSION[©user_id©] = $resp[ ©id© ];
         $_SESSION[©user_level©] = $resp[ ©user_level© ];
         $_SESSION[©user_name©] = $username;
         $_SESSION[©user_lastactive©] = time();
        return true;
  }}
```

Segurança
Lembre-se de tratar as variáveis $username e $password antes de fazer a query.

Se for utilizar o “Lembre-se de mim”

```
function login($username, $password, $remember = false)
...
if($remember) {
   // Gera uma nova chave de identificação com um cookie
   $cookie_auth = "algoRandomico" . $username;
   $auth_key = sha1( $cookie_auth );
   $auth_query = mysql_query(
            " UPDATE users SET auth_key = ©$auth_key© ".
            " WHERE username = ©$username© "  );
   exp = time()+60*60*24*7; //sete dias após o dia atual
   setcookie( "auth_key", $auth_key, exp, "/",
            "exemplo.com", false, true );
}
```

Mãos na massa

Crie cookies e utilize-os para gravar informações do seu site.