

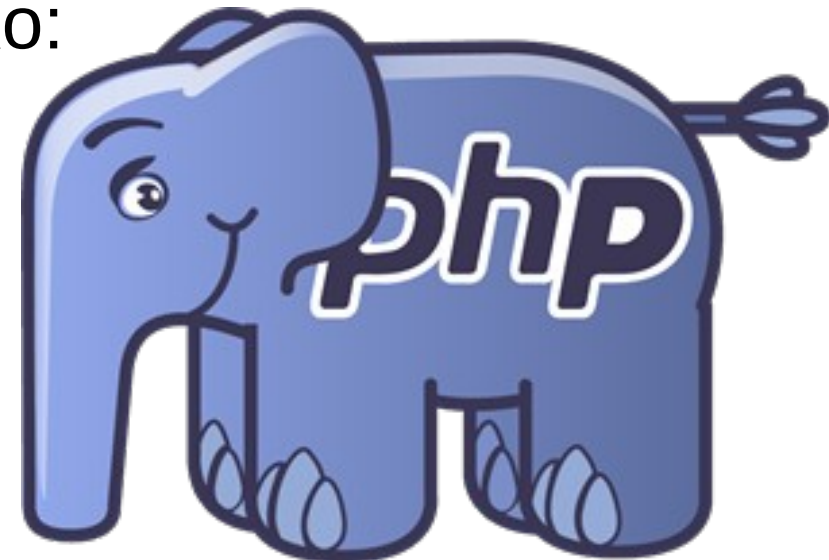
Linguagem PHP

Prof. Daniel Di Domenico

Integração
cliente-servidor



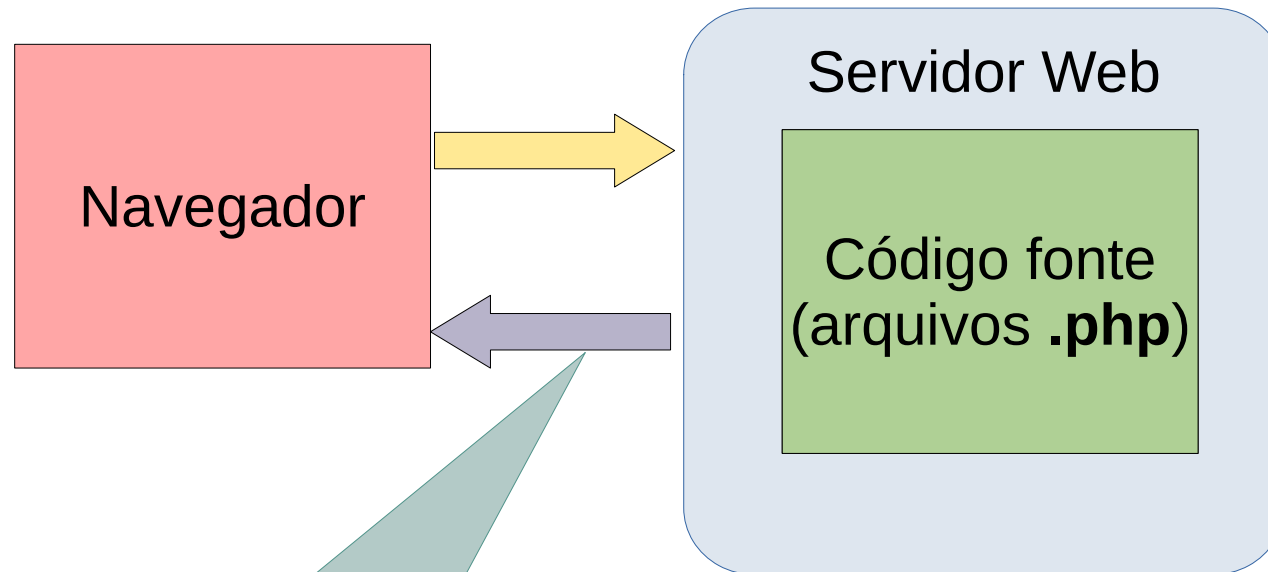
- O que já sabemos sobre PHP
 - Características da linguagem
 - Variáveis, tipos e constantes
 - Comandos de controle de fluxo:
 - IF / ELSE, SWITCH
 - WHILE, FOR / FOREACH
 - Funções
 - Arrays
 - Classes e objetos



PHP: integração cliente-servidor

- **Objetivo da aula:**
 - Integrar o lado cliente (HTML/CSS/JS) da aplicação com o lado servidor (PHP)
 - Até o momento, apenas utilizamos o **processamento no lado servidor**, mostrando a saída do programa no lado cliente
- **Como enviar dados do lado cliente para o lado servidor?**

PHP: integração cliente-servidor



Estamos exibindo a
resposta da requisição ao
servidor no navegador

PHP: funções

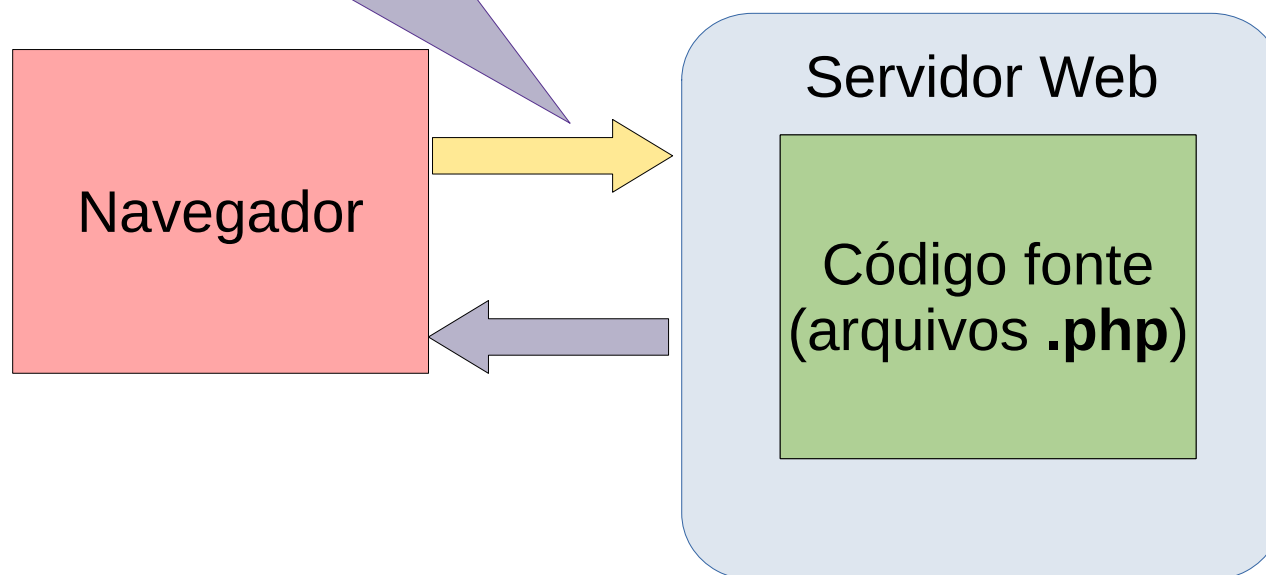


INSTITUTO
FEDERAL

Paraná

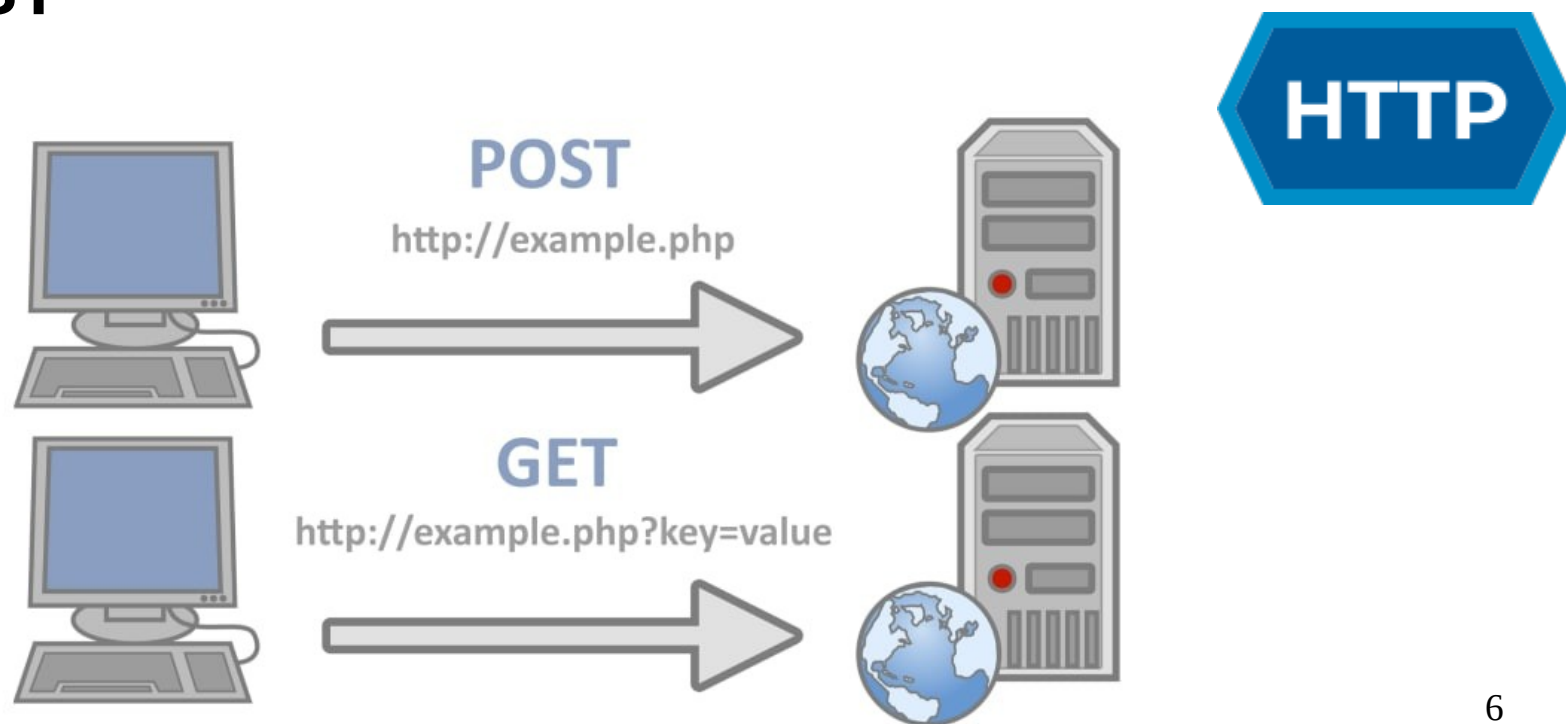
Campus
Foz do Iguaçu

Como **enviar** e **utilizar** os dados de uma requisição no servidor?



PHP: integração cliente-servidor

- **Envio de dados do cliente para o servidor:**
 - Utilizando os verbos/métodos HTTP
 - PHP suporta nativamente os verbos/métodos **GET** e **POST**



HTTP: método GET

- **GET:** envio de informações concatenando dados no campo de endereço (URL)
 - Utiliza os caracteres ? (início) e & (separação) para delimitar os parâmetros
 - Formato: chave=valor
 - Ex.: *pagina.php?nome=Carlos&idade=25*
 - Limitado a 2048 caracteres de informação
 - Menos seguro que o método POST, pois os dados enviados ficam visíveis na URL
 - Mais utilizado para buscar dados do servidor

HTTP: método POST

- **POST:** envio de informações utilizando um pacote de dados separado da requisição
 - O método POST transmite os dados para o servidor no corpo da requisição HTTP
 - Sem limite de tamanho de informações
 - Suporta outros tipos de dados além de texto
 - Mais seguro que o GET, pois os dados enviados ficam “escondidos”
 - Ao utilizar uma requisição HTTPS, esse dados são criptografados
 - Mais utilizado para enviar dados que serão armazenados pelo servidor

PHP: acesso aos dados (servidor)

- Para o PHP ter acesso aos dados recebidos por GET/POST, deve-se utilizar as **variáveis superglobais**:
 - Ambas variáveis são **arrays associativos**, sendo que os valores devem ser acessados pela **chave**
- **\$_GET['chave']** : retorna o valor de *chave* recebido pelo método GET
- **\$_POST['chave']**: retorna o valor de *chave* recebido pelo método POST

PHP: capturando dados

- Exemplo: variáveis globais **\$_GET** e **\$_POST**

```
//Recebe parâmetros por GET
```

```
$nome = $_GET["nome"];  
echo $nome;
```

```
echo "<br>";
```

```
$idade = $_GET["idade"];  
echo $idade;
```

```
//Recebe parâmetros por POST
```

```
$nome = $_POST["nome"];  
echo $nome;
```

```
echo "<br>";
```

```
$idade = $_POST["idade"];  
echo $idade;
```

HTTP: requisição com parâmetros

- Como fazer uma requisição enviando parâmetros GET ou POST?
 - **GET:** basta passar os parâmetros na URL
 - **POST:** precisa-se utilizar uma estrutura denominada **Form data** para adicionar os parâmetros no corpo da requisição HTTP:
 - Para testar, podemos utilizar o extensão **Thunder Client** do VS Code



Exercícios

- **1-** Implemente um programa em PHP que receba dois números e imprima a soma dos mesmos de duas maneiras:
 - Recebendo os parâmetros por GET
 - Recebendo os parâmetros por POST
- **2-** Implemente um programa em PHP que receba 3 números e imprima a média aritmética deles. Codifique dois programas, recendo os parâmetros por:
 - Método GET
 - Método POST
- **3- Desafio:** faça um programa em PHP que receba como parâmetro GET uma cor e gere uma página HTML com o fundo na cor escolhida.