Prof. Me. Hélio Esperidião

Rotas e Roteamento

Rotas na web

Uma rota em web é um “caminho” que será “chamado” por uma aplicação

- Este caminho é responsável por **um** ou mais serviços.

Cada rota pode ter uma ou mais funções.

Ao receber uma chamada a rota faz todo o processamento necessário para retornar os dados que foram solicitados.

Roteamento

O Roteamento refere-se à determinação de como um aplicativo responde a uma solicitação do cliente por um endpoint específico, que é uma URI (ou caminho) e um método de solicitação HTTP específico (GET, POST, etc).

Cada rota pode ter uma ou mais funções de manipulação, que são executadas quando a rota é correspondida.

Exemplo

- **(URI):** http://localhost:8080/retangulos/areas/5/10
  - **Rota**: /retangulos/areas/5/10
  - **Função**: Retorna a área de um retângulo com base 5 altura 10
- **(endpoint/uri):** http://localhost:8080/retangulos/perimetros/5/10
  - **Rota**: /retangulos/perimetros/5/10
  - **Função**: Retorna o perímetro de um retângulo com base 5 altura 10

Arquivo ***.htaccess***

O arquivo .htaccess é um arquivo de configuração do Apache.

Seu conteúdo dará instruções ao Apache para que o servidor se comporte de uma determinada maneira.

Pode ser utilizado para o tratamento e manipulação de rotas em php.

Redireciona todas as rotas para: index.php

.htaccess

```
# Ativa o mecanismo de reescrita de URL
RewriteEngine on
# Se a requisição não for um arquivo existente...
RewriteCond %{REQUEST_FILENAME} !-f
# ... E se a requisição não for um diretório existente...
RewriteCond %{REQUEST_FILENAME} !-d
# Redireciona todas as requisições para roteador.php
RewriteRule ^(.*)$ /roteador.php
```

C:\xampp\htdocs\.htaccess

```
#Arquivo: .htaccess:
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ /roteador.php
```

- O arquivo ***.htaccess deve ser posicionado exatamente em:***
  - C:\xampp\htdocs\.htaccess
- O arquivo ***.htaccess*** fará o redirecionamento de qualquer requisição para o arquivo roteador.php. O arquivo roteador.php será um roteador.
- http://localhost/teste
- http://localhost/teste/tesete2
- http://localhost/teste/n
- Todas as rotas acima serão processadas pelo arquivo index.php

Roteamento

- O processo de roteamento pode ser complexo se construído do zero, portanto iremos utilizar uma pequena classe de roteamento de um terceiro.
  - O bramus/router é um projeto de um micro roteador para PHP, desenvolvido por Bramus Van Damme.
  - Fornece uma maneira simples e leve de manipular rotas em aplicações PHP sem precisar de frameworks robustos como Laravel ou Symfony.
  - Github do projeto:
    - https://github.com/bramus/router

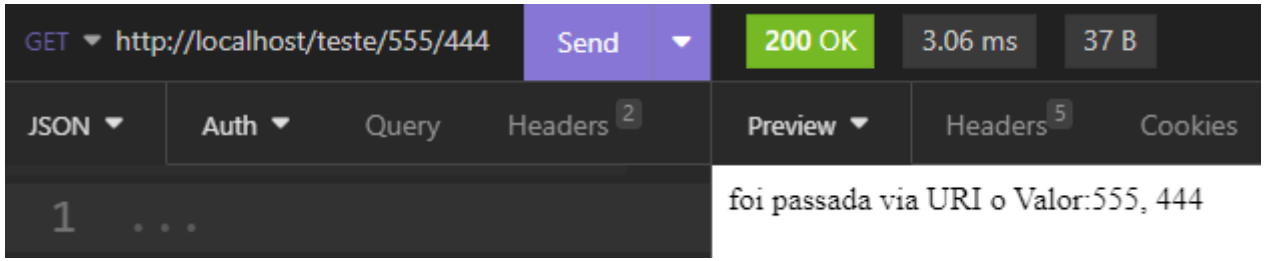
Utilização

- Para utilizar faça a o download do arquivo:
  - https://github.com/bramus/router/blob/master/src/Bramus/Router/Router.php
- Posicione esse arquivo exatamente na pasta:
  - C:\xampp\htdocs\
- Abra o arquivo e remova a seguinte linha:
  namespace Bramus\Router;

Passagem de paramento via URI.

```php
<?php
//https://github.com/bramus/router
require_once "Router.php";

//Cria uma instância da classe Router
$router  = new Router();

//mapeia a URI /teste/{int}
$router->get('/teste/(\d+)', function($v1) {
  echo "foi passada via URI o Valor:$v1 ";
});
//mapeia a URI /teste/{int}/{int}
$router->get('/teste/(\d+)/(\d+)', function($v1,$v2) {
   echo "foi passada via URI o Valor:$v1, $v2 ";
 });


//inicializa o roteamento
$router->run();
?>
```

Padrão	Significado
(\d+)	Dígitos de 0-9
(-?\d+)	Permite negativos



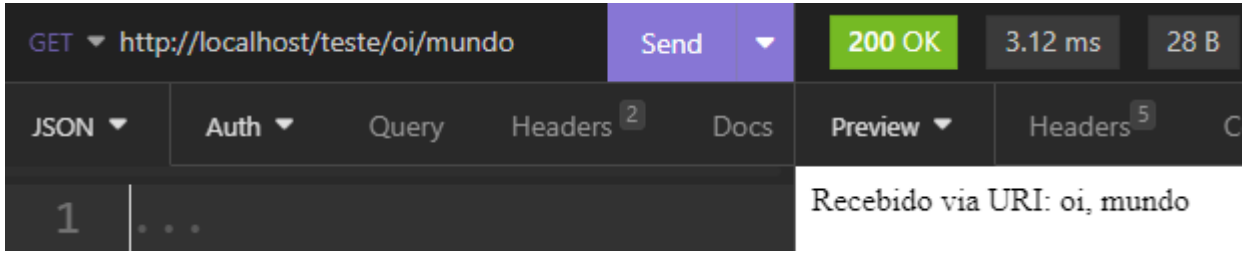
roteador.php

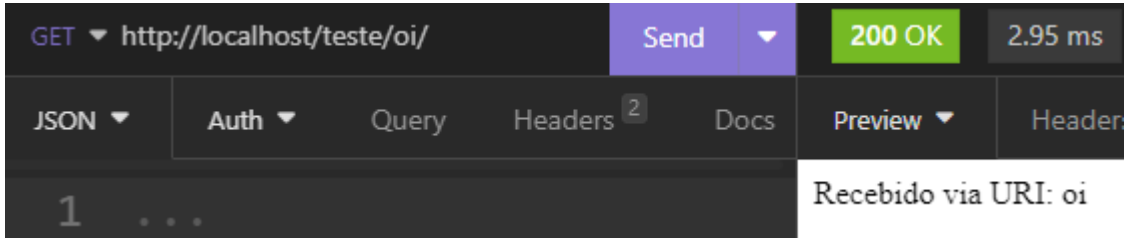
```php
<?php
require_once "Router.php"; //https://github.com/bramus/router
$router = new Router(); //Cria uma instância da classe Router
$router->get('/retangulos/area/(\d+)/(\d+)', function ($base, $altura) {
     echo "$base - $altura"; //http://localhost:8080/retangulos/area/50/5
});
$router->get('/retangulos/perimetro/(\d+)/(\d+)', function ($base, $altura) {
    echo "$base - $altura"; //http://localhost:8080/retangulos/perimetro/50/5
});
$router->get('/retangulos/diagonal/(\d+)/(\d+)', function ($base, $altura) {
    echo "$base - $altura"; //http://localhost:8080/retangulos/diagonal/50/5
});
$router->get('/retangulos/(\d+)/(\d+)', function ($base, $altura) {
    echo "$base - $altura"; //http://localhost:8080/retangulos/50/5
});
$router->run(); //inicializa o roteamento
?>
```

Passagem de paramento via URI.

```php
<?php
//https://github.com/bramus/router
require_once "Router.php";

//Cria uma instância da classe Router
$router  = new Router();

//mapeia a URI /teste/{string}
$router->get('/teste/(\w+)', function($v1) {
  echo "Recebido via URI: $v1 ";
});
//mapeia a URI /teste/{string}/{string}
$router->get('/teste/(\w+)/(\w+)', function($v1,$v2) {
   echo "Recebido via URI: $v1, $v2 ";
 });

//inicializa o roteamento
$router->run();
?>
```

Padrão	Significado
(\w+)	Caracteres : a-z 0-9 _

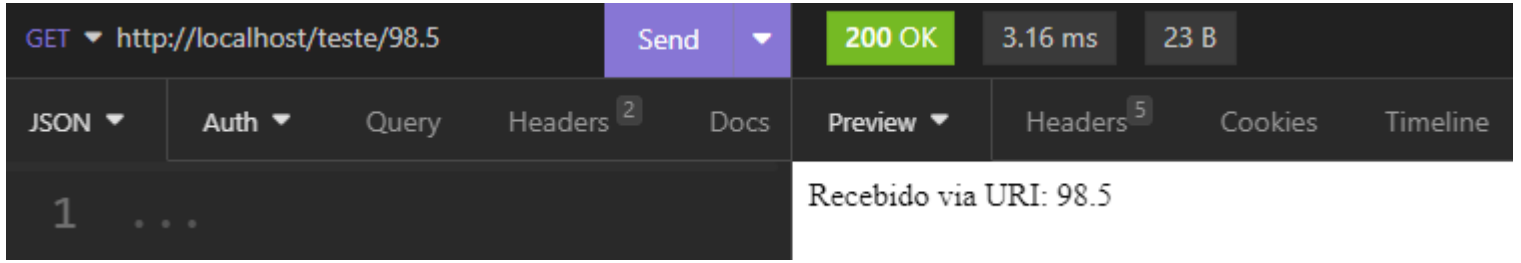




Passagem de paramento via URI.

```php
<?php
//https://github.com/bramus/router
require_once "Router.php";

//Cria uma instância da classe Router
$router  = new Router();

//mapeia a URI /teste/{números e pontos}
$router->get('/teste/([0-9.]+)', function($v1) {
  echo "Recebido via URI: $v1 ";
});

//inicializa o roteamento
$router->run();
?>
```

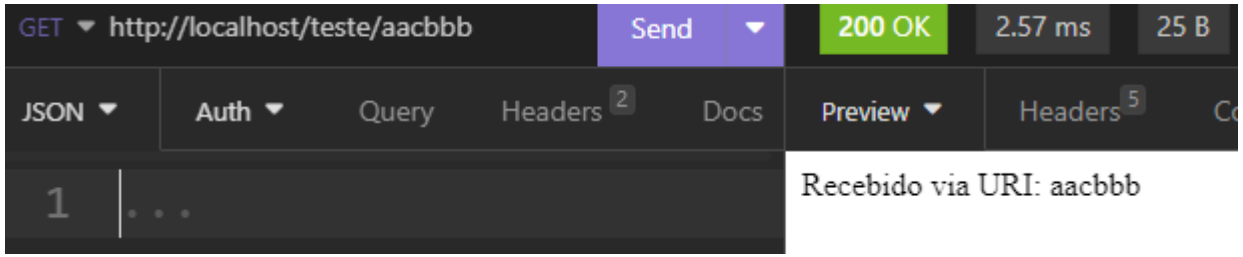
Padrão	Significado
(\w+)	Caracteres : a-z 0-9 _
([0-9.]+)	Float com “.”
([0-9,]+)	Float com “,”



```php
<?php
//https://github.com/bramus/router
require_once "Router.php";

//Cria uma instância da classe Router
$router  = new Router();

//mapeia a URI /teste/{números e virgula}
$router->get('/teste/([abc]+)', function($v1) {
  echo "Recebido via URI: $v1 ";
});

//inicializa o roteamento
$router->run();
?>
```

Passagem de paramento via URI.

Padrão	Significado
[abc]+	São aceitos apenas os caracteres ab c



POST, PUT, DELETE

```php
<?php
//https://github.com/bramus/router
require_once "Router.php";
//Cria uma instância da classe Router
$router  = new Router();
//mapeia a URI /
$router->post('/cliente/', function() {
   $jsonRecebido = file_get_contents('php://input');
   echo "post:  $jsonRecebido";
 });
 $router->put('/cliente/(\d+)', function($id) {
   $jsonRecebido = file_get_contents('php://input');
   echo "put: $id - $jsonRecebido ";
 });
  $router->delete('/cliente/(\d+)', function($id) {
   echo "delete: $id";
 });
//inicializa o roteamento
$router->run();
?>
```