

APIs WEB

prof. Me. Hélio Esperidião

O que é Front End?

- A parte visual de um site, que podemos interagir, é conhecida como Front-End. Os profissionais que trabalham nessa área são responsáveis por desenvolver a interface gráfica usando tecnologias como HTML, CSS e JavaScript.
- É comum confundir o trabalho de um Front-End com o de um designer, pois no passado existia a figura do Webmaster que desempenhava várias funções. No entanto, a diferença está no fato de que o designer utiliza ferramentas visuais, como Photoshop ou Sketch, para criar a interface, enquanto o desenvolvedor Front-End trabalha mais próximo do código em si, que será executado em navegadores como Chrome, Firefox ou Safari.

Front-end

- Atualmente a curva de aprendizado do front-end é maior
- São inúmeras tecnologias onde é necessário possuir domínio.
 - Html, css, Javascript, ux Design.



The screenshot shows two LinkedIn job listings. The first listing is for 'Desenvolvedor Front End' with 1,000 vacancies and 76 new ones. The second listing is for 'Front End' with 5,000 vacancies and 313 new ones. Both listings include a brief description of the job and a link to the full listing.

linkedin.com
https://br.linkedin.com › jobs › desenvolvedor-front-e... ⋮

1.000 vagas de Desenvolvedor Front End em: Brasil (76 novas)
1.000 principais **vagas** de Desenvolvedor **Front End** hoje em **Brasil**. Aproveite sua rede profissional e seja contratado. Novas **vagas** de Desenvolvedor **Front End** ...

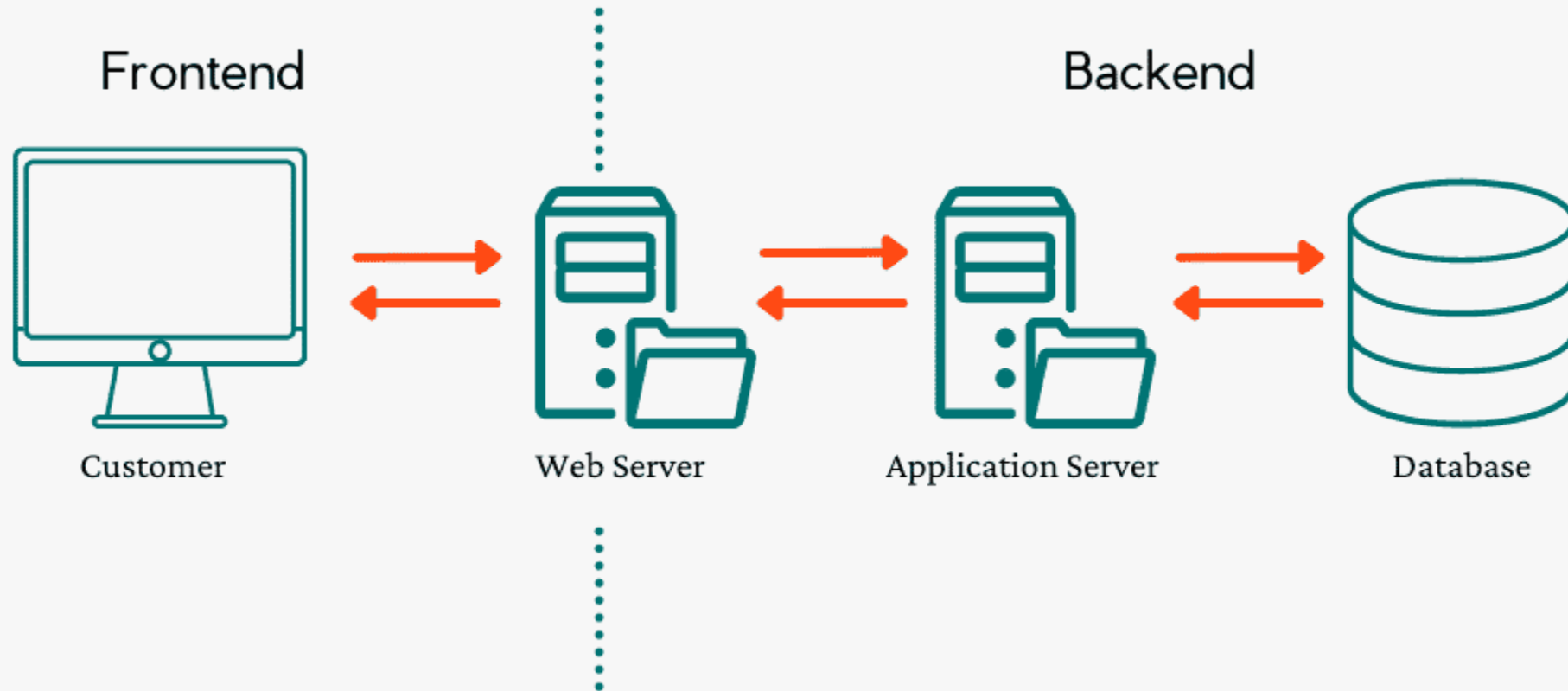
https://br.linkedin.com › jobs › front-end-vagas ⋮

5.000 vagas de Front End em: Brasil (313 nova(s)) - LinkedIn
5.000 principais **vagas** de **Front End** hoje em **Brasil**. Aproveite sua rede profissional e seja contratado. Novas **vagas** de **Front End** adicionadas diariamente.

O que é Back End?

- O Back-End, como o próprio nome sugere, refere-se à parte que está por trás de uma aplicação. Embora possa parecer abstrato inicialmente, pense no Facebook, por exemplo. Para que você possa usar o Facebook no seu dia a dia, os dados do seu perfil, amigos e publicações precisam estar armazenados em algum lugar e serem processados a partir desse local, que é um banco de dados.
- O trabalho do Back-End, na maioria dos casos, envolve a conexão entre os dados que são enviados do navegador para o banco de dados e vice-versa. Ele aplica regras de negócio, validações e garante a integridade dos dados em um ambiente que é restrito ao usuário final, ou seja, onde o usuário não tem acesso direto ou controle para manipular informações.

Frontend vs. Backend



Back-end

- É mais fácil para quem é bom com lógica de programação e não tem tanta aptidão com questões visuais;
- Os salários para back-end normalmente são maiores que os de front-end;
- O back-end é melhor para os iniciantes, pois conhecendo apenas uma linguagem de programação já é possível atuar.

Full Stack

- Full Stack é um profissional da área de Tecnologia da Informação com habilidades versáteis para trabalhar em várias atividades relacionadas ao desenvolvimento e programação web.
- Esse tipo de profissional é altamente valorizado no mercado devido ao seu amplo conhecimento em diversas linguagens, códigos e tecnologias.

Full Stack

- O termo "Full Stack", traduzido para o português, significa "pilha completa", não no sentido de energia, mas sim em referência a alguém que possui todas as habilidades necessárias.
- Podemos fazer uma analogia com um jogador de futebol, dizendo que o Full Stack "joga em todas as posições", ou seja, ele é capaz de atuar em todas as áreas.

Full Stack

- Full Stack é responsável por liderar todo o processo de desenvolvimento de uma solução digital, desde o back-end até o front-end.
- Para desempenhar essas diversas tarefas, o profissional precisa ter um domínio de tecnologias, linguagens e códigos como HTML, Python, CSS, PHP, JavaScript, banco de dados MySQL, entre outros.

Vagas em aberto



linkedin.com

<https://br.linkedin.com> > jobs > desenvolvedor-full-stack... ⋮

3.000 vagas de Desenvolvedor Full Stack em - LinkedIn


3.000 principais **vagas** de Desenvolvedor **Full Stack** hoje em **Brasil**. Aproveite sua rede profissional e seja contratado. Novas **vagas** de Desenvolvedor Full ...

Salários: Full - Stack

<https://www.glassdoor.com.br/>

Salários do cargo de Full Stack Developer - Brasil

Atualizado em 22 de mai. de 2023

 Confiança muito alta

R\$ 6.000 /mês

Média salarial

611 salários



Outras remunerações ⓘ

Média: **R\$ 6.000** Intervalo: R\$ 1.460 - R\$ 15.200


Qual é o salário de Full Stack Developer? O salário médio nacional de Full Stack Developer é de R\$6.000 em Brasil. Filtre por localização para ver os salários de Full Stack Developer na sua região. As estimativas de salários têm como base os 611 salários enviados de forma sigilosa ao Glassdoor por funcionários com o cargo de Full Stack Developer.

Salários: back-end

<https://www.glassdoor.com.br/>

Salários do cargo de Desenvolvedor Back-end – Brasil

Atualizado em 22 de mai. de 2023

 Confiança muito alta

R\$ 5.027 /mês

Média salarial

412 salários



Outras remunerações ⓘ

Média: **R\$ 7.325** Intervalo: R\$ 1.750 - R\$ 43.099


Qual é o salário de Desenvolvedor Back-end? O salário médio nacional de Desenvolvedor Back-end é de R\$5.026 em Brasil. Filtre por localização para ver os salários de Desenvolvedor Back-end na sua região. As estimativas de salários têm como base os 412 salários enviados de forma sigilosa ao Glassdoor por funcionários com o cargo de Desenvolvedor Back-end.

Salários: front - end

<https://www.glassdoor.com.br/>

Salários do cargo de Desenvolvedor Front End - Brasil

Atualizado em 22 de mai. de 2023

 Confiança muito alta

R\$ 4.500 /mês

Média salarial

3.467 salários



Outras remunerações ⓘ

Média: **R\$ 6.100** Intervalo: R\$ 860 - R\$ 25.729

Qual é o salário de Desenvolvedor Front End? O salário médio nacional de Desenvolvedor Front End é de R\$4.500 em Brasil. Filtre por localização para ver os salários de Desenvolvedor Front End na sua região. As estimativas de salários têm como base os 3.467 salários enviados de forma sigilosa ao Glassdoor por funcionários com o cargo de Desenvolvedor Front End.

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, e ela deve ser única na aplicação web, ao receber uma chamada ela 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

- Url: <http://localhost:8080/mostrarClientes>
 - Rota: /mostrarClientes
 - Função: Retornar uma tabela com todos os clientes cadastrados.
- Url: <http://localhost:8080/calcularMedia>
 - Rota: /calcularMedia
 - Função: Recebe duas notas e retorna a média das notas.

APIs



A sigla API (Application Programming Interface) que, traduzida para o português, pode ser compreendida como uma interface de programação de aplicação.



API é um conjunto de normas que possibilita a comunicação entre plataformas através de uma série de padrões e protocolos.



Por meio de APIs, desenvolvedores podem criar novos softwares e aplicativos capazes de se comunicar com outras plataformas

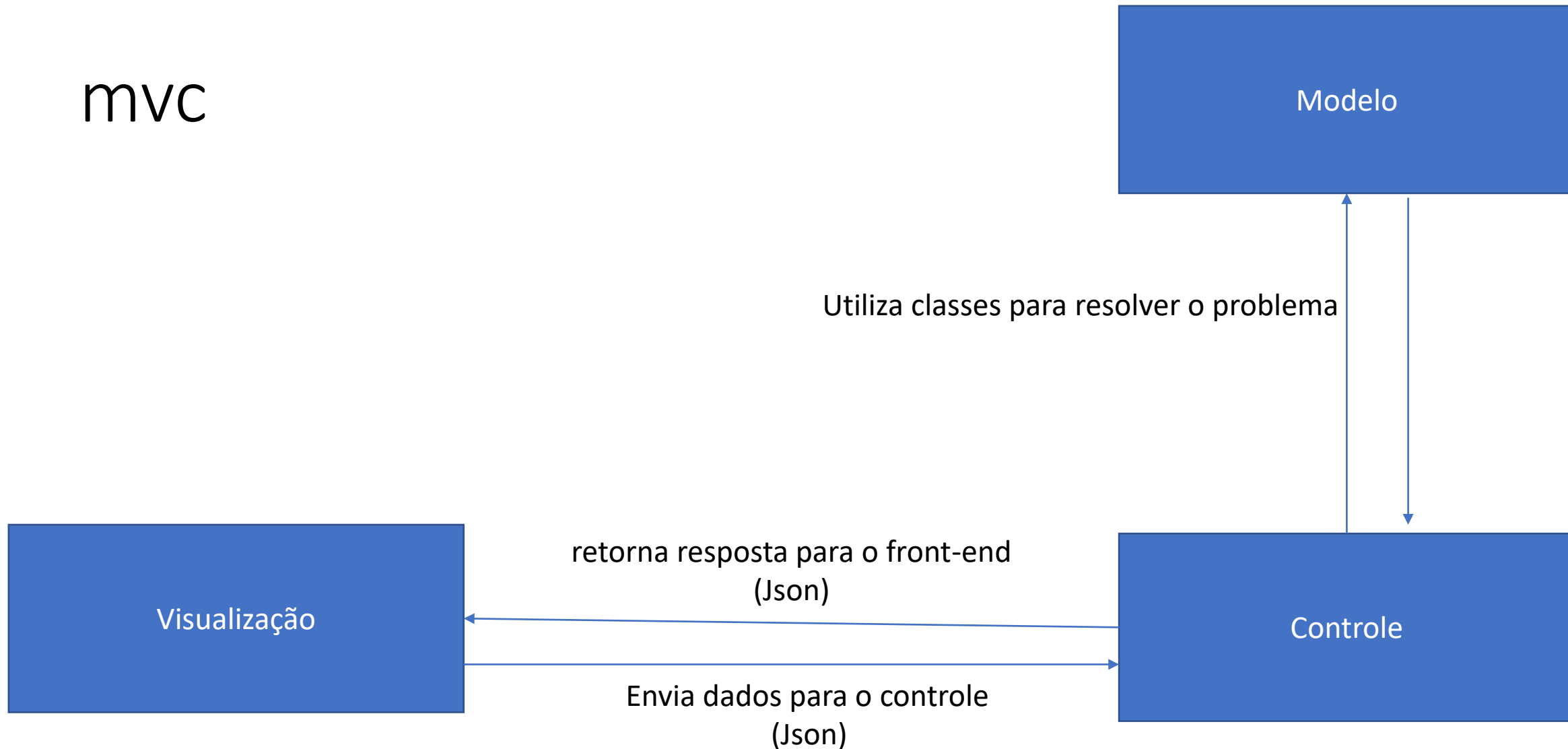
Esquema de funcionamento de apis



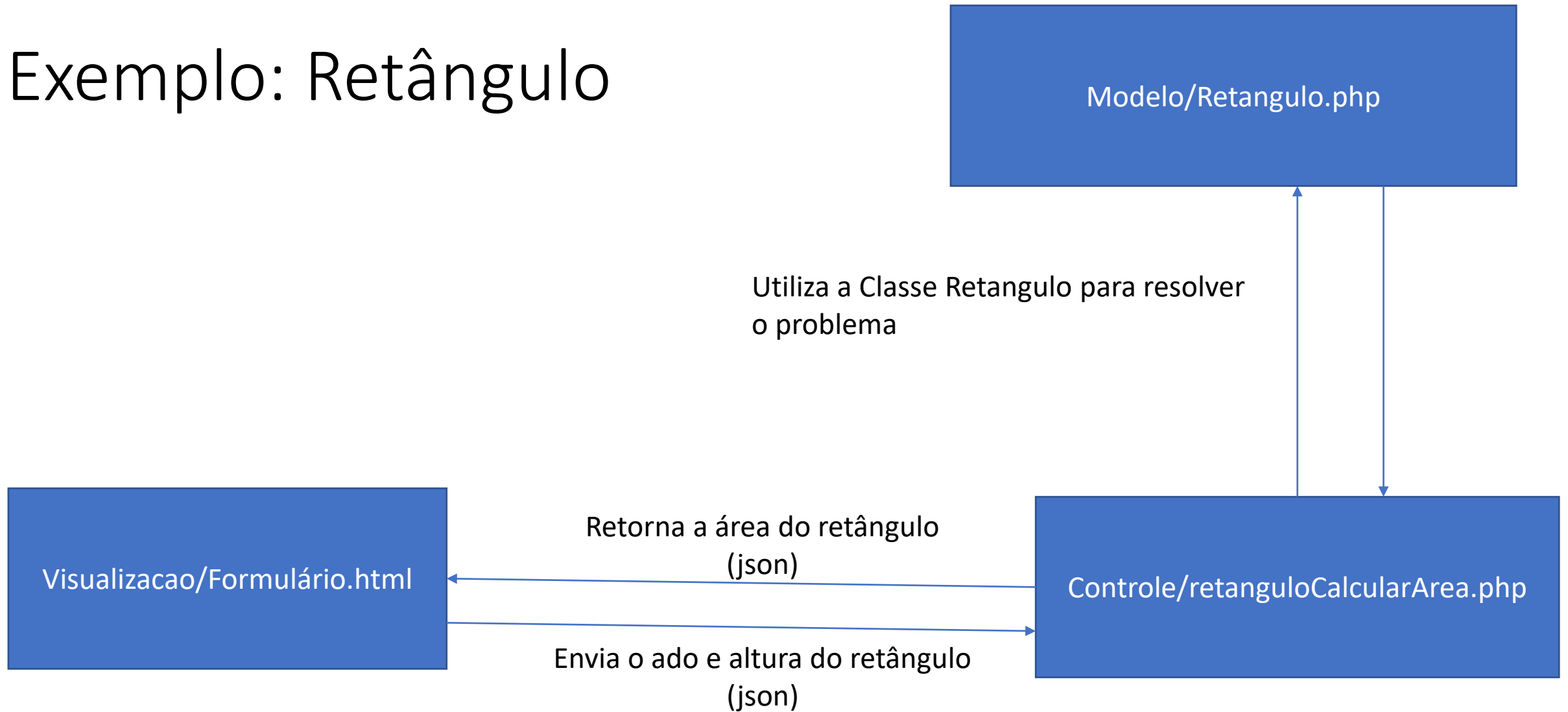
Padrão mvc

- Modelo
 - Todas as classes da aplicação ficarão armazenadas nesse diretório
- Visualização
 - Todos os arquivos de front-end ficarão armazenados nesse diretório.
- Controle.
 - Arquivos que recebem dados do front-end, utilizam as classes e enviam respostas para o front-end.

mvc



Exemplo: Retângulo



JSON - JavaScript Object Notation

É uma formatação leve de troca de dados.

Para seres humanos, é fácil de ler e escrever.

Para máquinas, é fácil de interpretar e gerar.

É baseado em um subconjunto da linguagem de programação JavaScript, Standard ECMA-262 3a Edição -Dezembro - 1999.

JSON

01

JSON é em formato texto e completamente independente de linguagem

02

Formato ideal de troca de dados

03

é um formato compacto, de padrão aberto independente, de troca de dados simples e rápida (parsing) entre sistemas

ESTRUTURA

Uma coleção de pares nome/valor. Em várias linguagens, isto é caracterizado como um *object*, record, struct, dicionário, hash table, keyed list, ou arrays associativas.

Uma lista ordenada de valores. Na maioria das linguagens, isto é caracterizado como uma *array*, vetor, lista ou sequência.

Exemplos de json

- `{"altura":57.7,"largura":55}`
- `{"nome":"helio","cargo":"professor"}`

Aplicações modernas.

- Tipicamente em uma aplicação moderna o front end envia um json com os dados para um controle que utiliza as classes necessárias para resolver o problema.
- O controle também responde para o front-end um json.

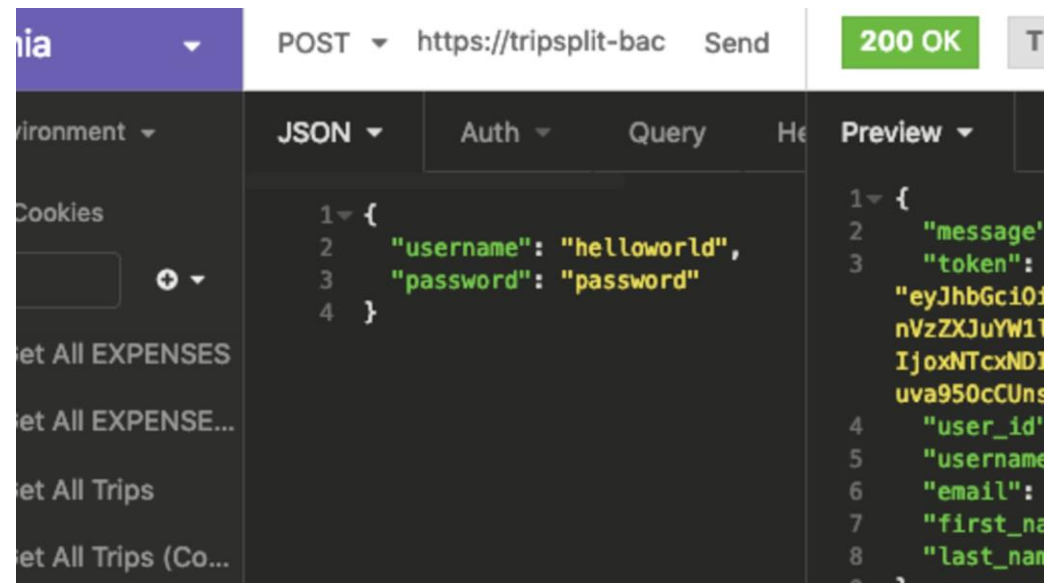
apis interessantes.

- <https://api.adviceslip.com/advice>
- <https://deividfortuna.github.io/fipe/>
 - <https://parallelum.com.br/fipe/api/v1/carros/marcas>
 - <https://parallelum.com.br/fipe/api/v1/carros/marcas/59/modelos>

insomnia

- é um software que nos permite fazer requisições à APIs. Ajuda no desenvolvimento do Backend (ex.: testar as chamadas para cada endpoint sem necessitar de um Frontend)
- Download free:

<https://insomnia.rest/>



Via cep:
<https://viacep.com.br/ws/12245020/json/>

The screenshot displays the Insomnia REST client interface. The top menu bar includes 'Application', 'Edit', 'View', 'Window', 'Tools', and 'Help'. Below the menu, there's a navigation bar with 'Insomnia / New Document', 'DESIGN', 'DEBUG', 'TEST', 'Setup Git Sync', 'Login', and 'Sign Up' buttons. The main interface is divided into several panels:

- Left Panel:** Shows 'No Environment' and 'Cookies'. A search filter is present. Below, a 'GET New Request' entry is visible.
- Request Panel:** Shows the method 'GET' and the URL 'https://viacep.com.br/ws/12245020/json/'. A 'Send' button is located to the right of the URL. Below the URL, tabs for 'Plain', 'Auth', 'Query', 'Headers', and 'Docs' are visible.
- Response Panel:** Shows the status '200 OK', response time '498 ms', and size '261 B'. Below this, tabs for 'Preview', 'Headers', 'Cookies', and 'Timeline' are visible. The 'Preview' tab is active, displaying a JSON response:

```
1 {  
2   "cep": "12245-020",  
3   "logradouro": "Rua Paraibuna",  
4   "complemento": "até 1501 - lado ímpar",  
5   "bairro": "Jardim São Dimas",  
6   "localidade": "São José dos Campos",  
7   "uf": "SP",  
8   "ibge": "3549904",  
9   "gia": "6452",  
10  "ddd": "12",  
11  "siafi": "7099"  
12 }
```

Testando apis: <https://api.adviceslip.com/advice>

The screenshot displays the Insomnia API client interface. At the top, the application name "Insomnia" is visible, along with a menu bar (Application, Edit, View, Window, Tools, Help) and a search bar. The main area shows a request configuration for a GET method to the URL `https://api.adviceslip.com/advice`. The request is in the "Send" state, and the response is displayed as "200 OK" with a response time of "216 ms" and a size of "148 B". The response body is shown in a "Preview" tab, displaying a JSON object with a "slip" property containing an "id" and an "advice".

```
1 {
2   "slip": {
3     "id": 68,
4     "advice": "The number of vampires in the average home, is direct
5   }
6 }
```

https://api.adviceslip.com/advice/1

The screenshot displays the Insomnia REST client interface. The main window shows a GET request to `https://api.adviceslip.com/advice/1` that has been successfully executed. The response status is `200 OK`, with a response time of `751 ms` and a body size of `101 B`. The response body is a JSON object:

```
1 {
2   "slip": {
3     "id": 1,
4     "advice": "Remember that spiders are more afraid of you, than you
are of them."
5   }
6 }
```

The interface includes a sidebar on the left with a 'New Request' button, a top navigation bar with 'DESIGN', 'DEBUG', and 'TEST' tabs, and a right-hand pane for viewing the response details.

Testar Minhas APIs

- Exemplo API Retangulo.
 - Classe:
 - Retangulo.php
 - Controles:
 - retangulo_CalcularArea.php
 - retangulo_CalcularPerimetro.php
 - retangulo_CalcularDiagonalPrincipal.php

Classe Retangulo.php.

Considerar que GETs e Sets foram programados

```
class Retangulo{
    private $base;
    private $altura;
    public function calcularArea(){
        return ($this->altura * $this->base);
    }
    public function calcularDiagonal(){
        return sqrt(pow($this->altura, 2) + pow($this->base, 2));
    }
    public function calcularPerimetro(){
        return ($this->altura * 2 + $this->base * 2);
    }
}
```

retangulo_CalcularArea.php

```
<?php
require_once "../modelo/Retangulo.php";
$altura = $_GET['altura'];
$base = $_GET['base'];

$r1 = new Retangulo();

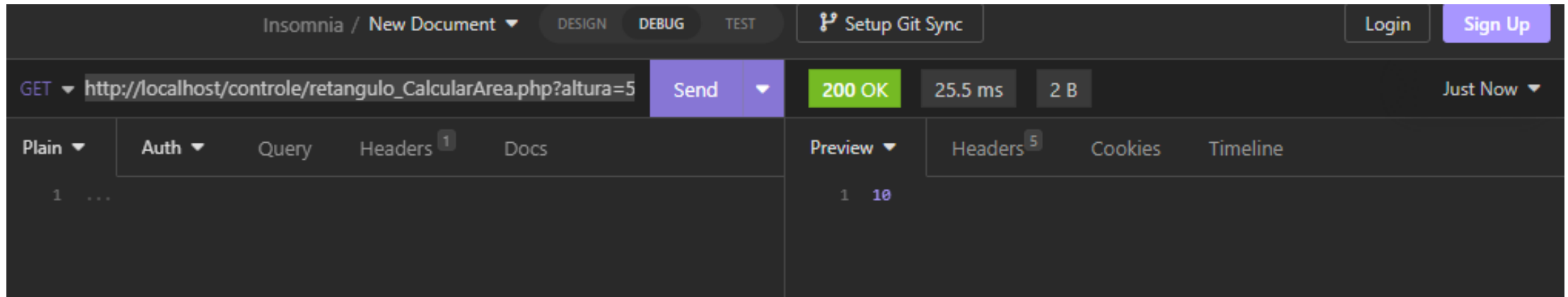
$r1->setAltura($altura);
$r1->setBase($base);
$area = $r1->calcularArea();

header("HTTP/1.1 200 OK");
echo $area;
?>
```

- `header("HTTP/1.1 200 OK");`
 - Significa que o processamento e resposta estão ok. “deu tudo certo com a requisição do usuário”

Testar no insomnia

- http://localhost/controle/retangulo_CalcularArea.php?altura=5&base=2



Teste utilizado o método post.

Alterar o arquivo: retangulo_CalcularArea.php

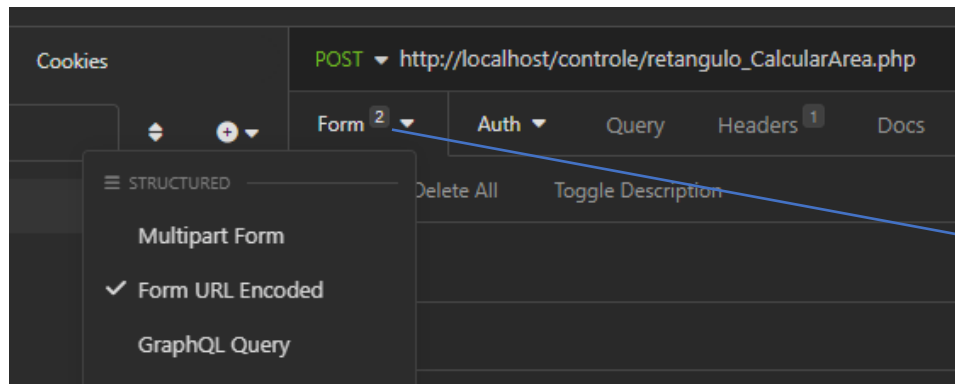
```
<?php
require_once "../modelo/Retangulo.php";
$altura = $_POST['altura']; //Trocou para post
$base = $_POST['base']; //Trocou para post

$r1 = new Retangulo();

$r1->setAltura($altura);
$r1->setBase($base);
$area = $r1->calcularArea();

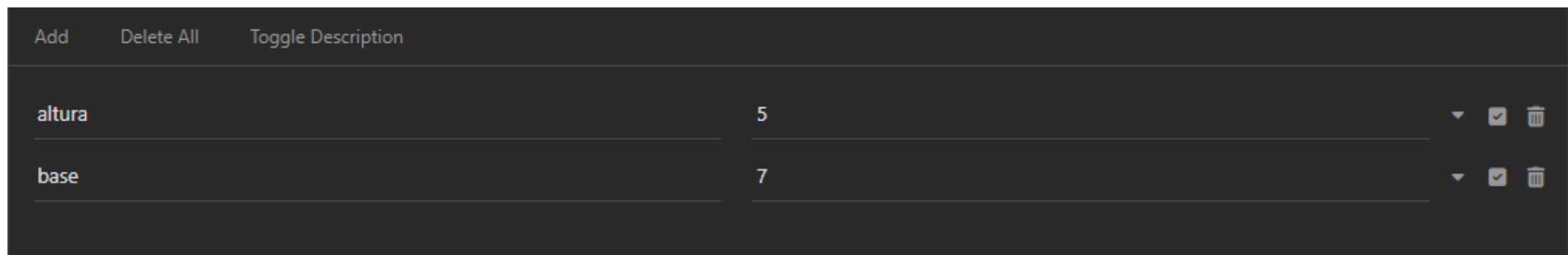
header("HTTP/1.1 200 OK");
echo $area;
?>
```

Caso seja enviado um post. Configure a ferramenta



Click nessa posição e escolha a opção “multipart form”

Adicione os “names” que virão do formulário e seus valores



Exemplo teste post:

The screenshot displays the Insomnia REST client interface. At the top, the application name 'Insomnia / New Document' is visible, along with tabs for 'DESIGN', 'DEBUG', and 'TEST'. A 'Setup Git Sync' button is also present. On the right side of the top bar, there are 'Login' and 'Sign Up' buttons.

The main interface shows a request configuration for a POST method to the URL `http://localhost/control/retangulo_CalcularArea.php`. The response status is `200 OK`, with a response time of `19 ms` and a response size of `3 B`. The request was sent 'Just Now'.

The request body is configured as a 'Form' with two fields:

Field Name	Value
altura	32
base	25

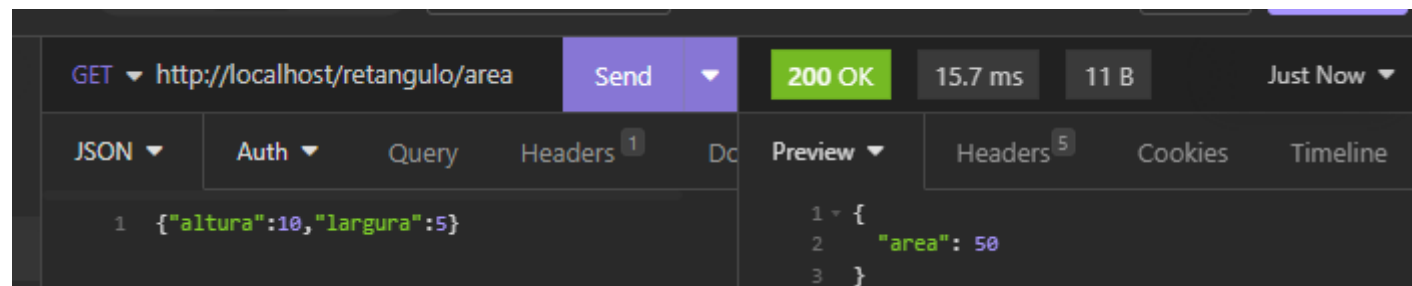
The response preview shows a status of `1 800`. The interface also includes tabs for 'Auth', 'Query', 'Headers', and 'Docs' for the request, and 'Preview', 'Headers', 'Cookies', and 'Timeline' for the response.

Apis e Aplicações Modernas

- Tipicamente quando construímos APIs web um dos objetivos é separar completamente o front-end do back end.
- Utilizando O **MVC** é possível separar o front-end do back end.
- Muitos desenvolvedores tem preferido trabalhar com o padrão **Json** para comunicação entre o front-end e o back-end.
- Os fronts (sites, televisores, carros, dispositivos eletronicos) enviam para o back-end dados utilizando o formato **JSON**. Isso garante um isolamento ainda maior do front-end.

Os dados não são enviados por meio da url nem por meio do post da forma tradicional.

- Observe que a rota não envia parâmetros via get
 - <http://localhost/retangulo/area>
- Tipicamente aplicações enviam dados em JSON no corpo da requisição.
- Veja como usar: insira os dados formatados no padrão **JSON**
- **Observe que a resposta da api também é em JSON**



The screenshot shows a web browser's developer tools interface. At the top, it displays the request method as 'GET' and the URL as 'http://localhost/retangulo/area'. The response status is '200 OK', with a response time of '15.7 ms' and a size of '11 B'. Below this, the 'JSON' tab is selected, showing the request body as a JSON object: `{ "altura": 10, "largura": 5 }`. The 'Preview' tab is also visible, showing the response body as a JSON object: `{ "area": 50 }`.

Arquivo .htaccess

- O arquivo .htaccess é um arquivo de configuração do Apache, o software que seu servidor usa para executar.
- 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.

Redirecciona todas as rotas para: index.php

```
RewriteEngine on
```

```
RewriteCond %{REQUEST_FILENAME} !-f
```

```
RewriteCond %{REQUEST_FILENAME} !-d
```

```
RewriteRule ^(.*)$ /index.php?path=$1 [NC,L,QSA]
```

C:\xampp\htdocs.htaccess

Arquivo: .htaccess:

RewriteEngine on

RewriteCond %{REQUEST_FILENAME} !-f

RewriteCond %{REQUEST_FILENAME} !-d

RewriteRule ^(.*)\$ /index.php?path=\$1 [NC,L,QSA]

- O arquivo htaccess fará o redirecionamento de qualquer requisição para o arquivo **index.php**.
- <http://localhost/teste>
- <http://localhost/teste/tesete2>
- <http://localhost/teste/n>
- Todas as rotas acima serão processadas pelo arquivo index.php

Variável: `$_SERVER['REQUEST_URI']`

- Essa variável é responsável por armazenar a URL digitada pelo usuário.
- <http://localhost/teste/n>
 - `$_SERVER["REQUEST_URI"] = /teste/n`
- <http://localhost/retangulo/area>
 - `$_SERVER["REQUEST_URI"] = /retangulo/area`
- <http://localhost/retangulo/perimetro>
 - `$_SERVER["REQUEST_URI"] = /retangulo/perimetro`

Index.php

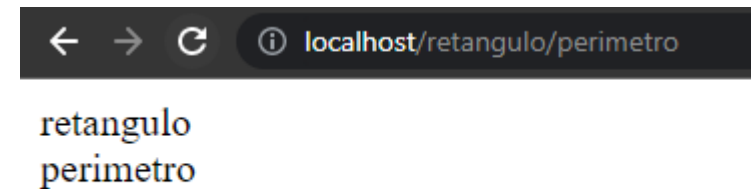
Observe a url digitada no navegador.

Observe a explosão da url e as posições dos vetores

```
<?php
```

```
$partesRota = explode("/", $_SERVER['REQUEST_URI']);  
echo $partesRota[1]."<br>";  
echo $partesRota[2];
```

```
?>
```



Index.php

//Testar com a rota : http://localhost/retangulo/perimetro

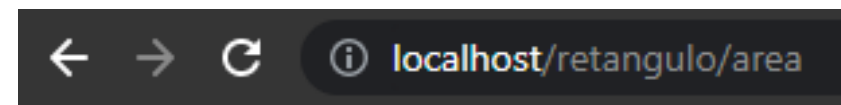
<?php

```
$partesRota = explode("/", $_SERVER['REQUEST_URI']);
```

```
echo $partesRota[1]."<br>";
```

```
echo $partesRota[2];
```

?>



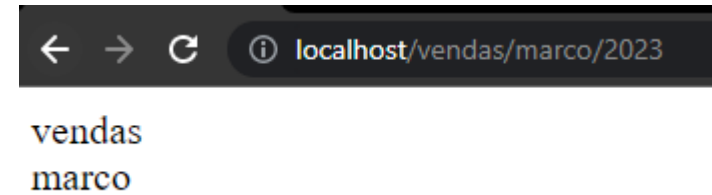
retangulo
area

Qual o valor da posição 3 do vetor?

```
<?php
```

```
$partesRota = explode("/", $_SERVER['REQUEST_URI']);  
echo $partesRota[1]."<br>";  
echo $partesRota[2];
```

```
?>
```



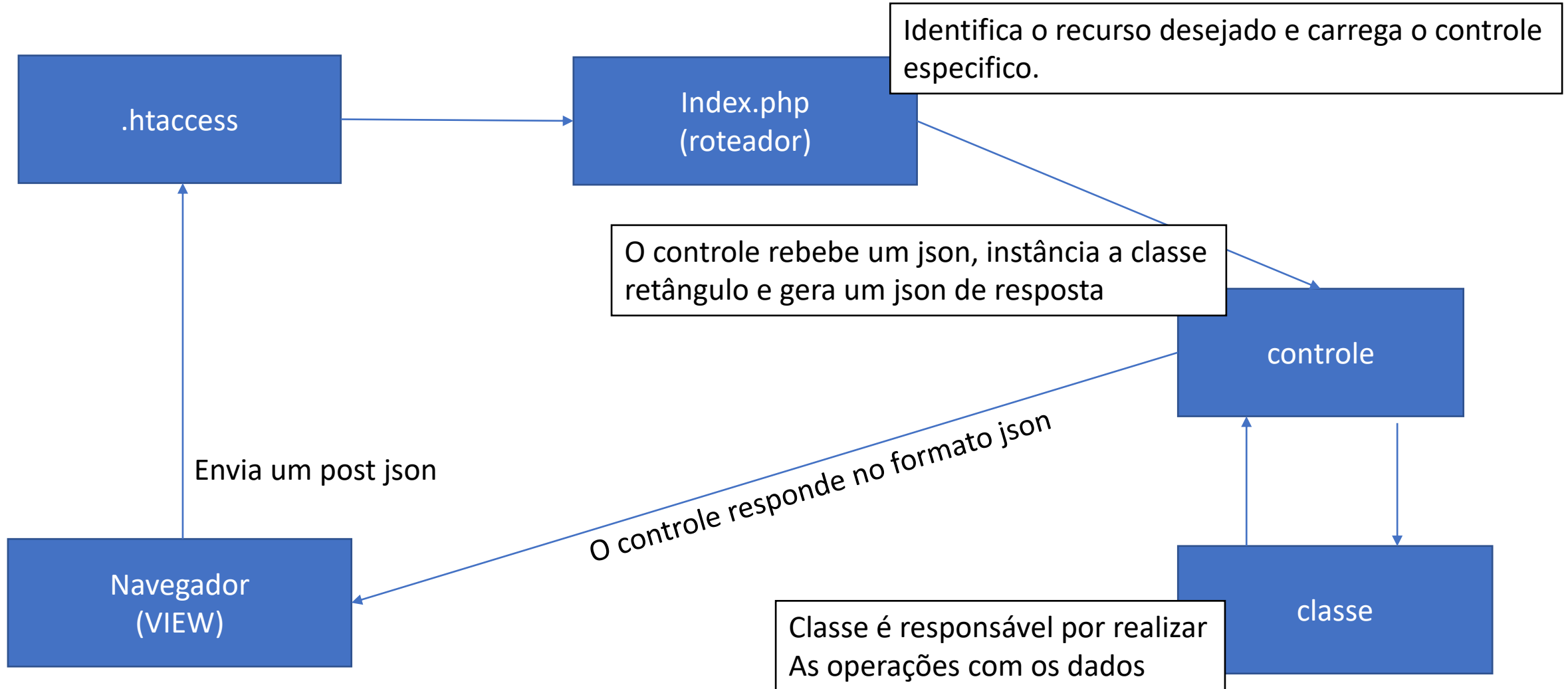
Exemplo 1:

- Será apresentada a criação de uma API no Modelo MVC.
- O front-end envia um post contendo um json para a api.
- A api recebe do cliente um json e responde um JSON.
- A api lida com todas as operações básicas de um retângulo.

Exemplo: Arquivos

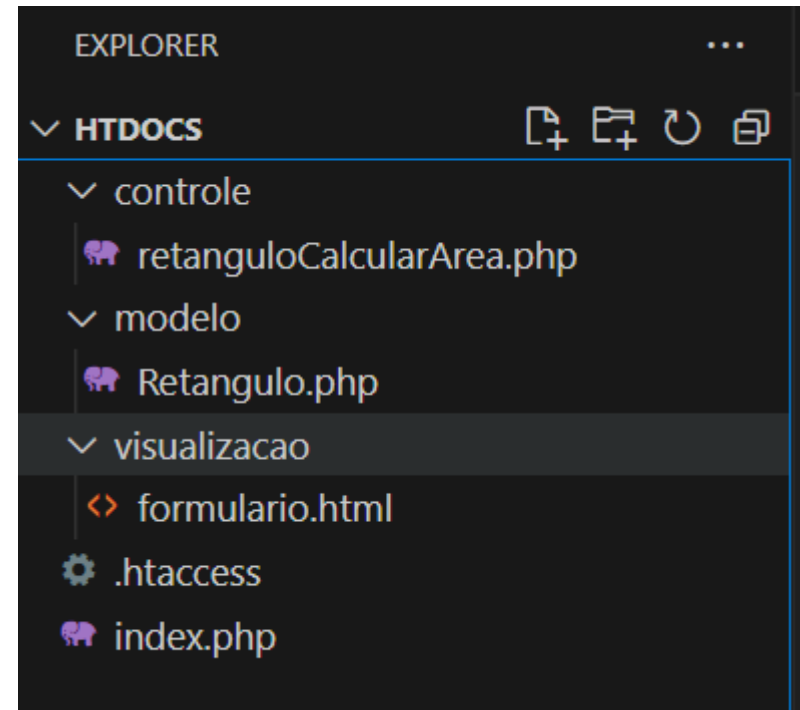
- Arquivos:
 - Configuração e roteamento
 - .htaccess
 - Index.php
 - Controle
 - retanguloCalcularArea.php
 - Modelo
 - Retangulo.php

Arquitetura do sistema



Exemplo 1:

- Para fins didáticos apague todos os arquivos da pasta htdocs e posicione todos os arquivos no diretório: **C:\xampp\htdocs**
- Crie a pasta **modelo**.
 - retanguloCalcularArea.php
- Crie a pasta **controle**.
- Crie a pasta **visualização**
 - Formulário.html
- Crie o arquivo **.htaccess**
- Crie o arquivo **index.php**



.htaccess

redireciona todas as requisições para o arquivo
index.php

```
RewriteEngine on
```

```
RewriteCond %{REQUEST_FILENAME} !-f
```

```
RewriteCond %{REQUEST_FILENAME} !-d
```

```
RewriteRule ^(.*)$ /index.php?path=$1 [NC,L,QSA]
```

Index.php

```
<?php
```

```
$metodo = $_SERVER['REQUEST_METHOD'];  
$partesRota = explode("/", $_SERVER['REQUEST_URI']);  
if($partesRota[1]=="retangulo"){  
    if($partesRota[2]=="area"){  
        if( $metodo=="POST"){ //Verifica o método de envio  
            require_once "controle/retanguloCalcularArea.php";  
        }  
    }elseif($partesRota[2]=="perimetro"){  
        require_once "controle/retanguloCalcularPerimetro.php";  
    }else{  
        header("HTTP/1.1 404 Not Found");  
    }  
}else{  
    header("HTTP/1.1 404 Not Found");  
}  
}
```

Verifica se a posição 1 do vetor é igual a retângulo
A posição 1 do vetor corresponde ao 2º texto
depois de http:domínio.com.br/1ºTexto

Verifica se a posição 2 do vetor é igual a retângulo
A posição 2 do vetor corresponde ao 2º texto
depois de http:domínio.com.br/1ºTexto/2ºTexto

```
?>
```

Retangulo.php

```
class Retangulo{
    private $base;
    private $altura;

    public function calcularArea(){
        return ($this->altura * $this->base);
    }
    public function calcularDiagonal(){
        return sqrt(pow($this->altura, 2) + pow($this->base, 2));
    }
    public function calcularPerimetro(){
        return ($this->altura * 2 + $this->base * 2);
    }
}
```

retanguloCalcularArea.php

```
require_once "modelo/Retangulo.php";

$jsonRecebido = file_get_contents('php://input');
$obj = json_decode($jsonRecebido);
$altura = $obj->altura;
$largura = $obj->largura;

$r1 = new Retangulo();
$r1->setAltura($altura);
$r1->setBase($largura);
$area = $r1->calcularArea();

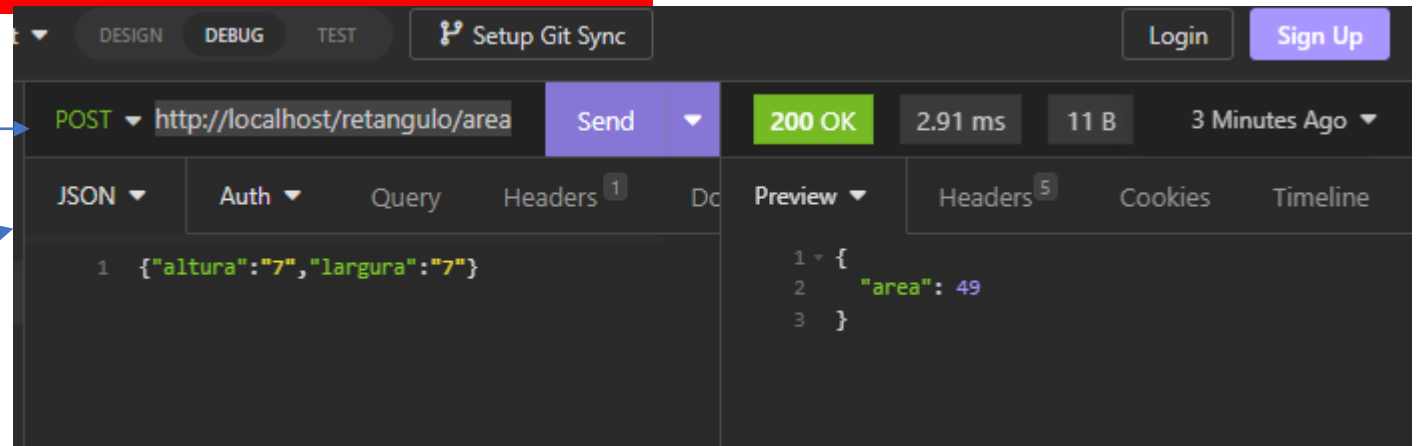
$resposta = array();
$resposta['area'] = $area;
header("HTTP/1.1 200 OK");
header('Content-Type: application/json; charset=utf-8');
echo json_encode($resposta);
?>
```

Teste de API

- Antes de construir o front-end é necessário testar a API já construída.
- É uma **péssima** prática construir tudo e testar no final.
 - A aplicação é complexa e encontrar problemas fica quase impossível
- **Apenas após funcionar no teste construa o front.**

Troque o método de envio para post

Digite um json válido par testar



Teste de API

- Antes de construir o front-end é necessário testar a API já construída.
- É uma **péssima** prática construir tudo e testar no final.
 - A aplicação é complexa e encontrar problemas fica quase impossível

Troque o método de envio para post

Digite um json válido par testar

```
POST http://localhost/retangulo/area 200 OK 2.91 ms 11 B 3 Minutes Ago
```

```
JSON Auth Query Headers 1 Docs Preview Headers 5 Cookies Timeline
```

```
{ "altura": "7", "largura": "7" }
```

```
1 {  
2   "area": 49  
3 }
```

É fácil visualizar problemas e corrigir quando é utilizado o insomnia

```
POST http://localhost/retangulo/area 200 OK 15.6 ms 166 B Just Now
```

```
JSON Auth Query Headers 1 Docs Preview Headers 5 Cookies Timeline
```

```
1 { "altura": "7", "largura": "7" }
```

```
Parse error: syntax error, unexpected '$jsonRecebido' (T_VARIABLE) in C:\xampp\htdocs\controle\retanguloCalcularArea.php on line 4
```

Front-end

- O HTML não é capaz de processar, gerar e enviar dados json.
- Para isso utilizaremos *Javascript*.
 - *O Javascript enviará uma requisição assíncrona para a rota desejada.*
 - *Requisição assíncrona significa que você não será redirecionado para o arquivo de requisição.*
 - *O Javascript envia os dados para a página destino e a aplicação continua na página atual enquanto a requisição é processada.*
- Neste momento do curso é importante saber apenas como recuperar os dados digitados no formulário montar um json e enviar o json no corpo da requisição.
- Portanto não serão abordadas agora detalhes da linguagem Javascript.

Front-end

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
</head>
<body>
  <div>
    Altura: <input type="text" id="txtAltura" value="5"><br>
    Largura: <input type="text" id="txtLargura" value="6"><br>
    <button onclick="enviarDados()">Enviar Json</button>
    <div id="divResposta"></div>
    <script>
      function enviarDados(){
        // próximo slide
      }
    </script>
  </div>
</body>
</html>
```

```
var alturaDigitada = document.getElementById("txtAltura").value; //recuperar valor digitado no input
var larguraDigitada = document.getElementById("txtLargura").value; //recuperar valor digitado no input
var Meujson = {"altura":alturaDigitada,"largura":larguraDigitada }; //monta objeto json para envio
var meuJsonConvertidoString= JSON.stringify(Meujson); //Converte o objeto json em texto puro
fetch("http://localhost/retangulo/area", { // Define a rota da requisição
  body: meuJsonConvertidoString, //define o json no formato texto que será enviado
  method: "POST", //define o método de envio
  headers: { "Content-Type": "application/json" }, //define o conteúdo que será enviado(json)
}).then((response) => {
  return response.json(); //converte os dados recebidos em um objeto json
}).then((data) => {
  console.log(data); //imprime no console o json recebido
  var area = data.area; //recuperar o json.area
  document.getElementById("divRespota").innerHTML= area; //escreve na div a área.
}).catch((error) => {
  console.error("Error:", error); //imprime no consolo caso aconteça algum erro.
});
```

- Para testar acesse o arquivo formulário.html
- Pressione f12 para abrir o painel do desenvolvedor.
- Selecione a opção console para verificar o que foi impresso no console.
- O console é muito útil para debug.

The screenshot shows a web browser window with the address bar displaying `localhost/visualizacao/formulario.html`. The page content includes a form with two input fields: "Altura:" with the value "5" and "Largura:" with the value "6". Below the inputs is a button labeled "Enviar Json". The number "30" is displayed below the button. The developer tools panel is open, showing the "Console" tab. The console displays a single message: `{area: 30}` from `formulario.html:27`. The console settings on the right are configured with "messages in console", "errors in console", "Eager evaluation", "Autocomplete from history", and "evaluation as user action" all checked.