**Intro**

PHP é uma linguagem de programação interpretada e de uso geral, usada normalmente para programar o backend de uma página web ou ainda serviços de API, apesar de ser capaz de suportar aplicações que funcionem num terminal, por exemplo.

**Ferramentas que EU recomendo**

Lista de recursos que EU, O AUTOR julgo serem bons ao desenvolver aplicações em PHP, obviamente não são obrigatórios, mas é a minha recomendação. Todos os recursos listados são grátis, e os links estão no fim deste arquivo.

• PHP v7.X

A versão 7 do PHP apresenta a melhor performance (eliminando a antiga má-fama de "linguagem lenta") e alguns recursos extras. Os PCs da escola rodam a versão 5.X.

• Banco de dados MySQL ou PostgreSQL

São os bancos de dados relacionais mais populares, e funcionam muito bem com o PHP. Os PCs da escola tem instalado o PostgreSQL.

• Cliente FTP FileZilla

Clientes FTPs (File Transfer Protocol, ou protocolo de transferência de arquivos) são os programas que são utilizados para acessar o servidor da escola e colocar os arquivos do site lá. Os PCs da escola já vem com o WinSCP, que não é ruim, mas desconecta sozinho.

• Editor de texto Visual Studio Code

O VSCode (não confundir com o Visual Studio) e o editor que eu mais gosto, mas editor de texto é algo de gosto extremamente pessoal. Também vale mencionar o Atom e o Sublime, além do clássico Notepad++, que, por ser mais simples, é perfeito para edição ou visualização rápida e prática. Os PCs da escola já vem com o Brackets instalado.

• Pacote de recursos XAMPP

O XAMPP é um pacote de recursos que inclui um servidor Apache, banco de dados MySQL e as linguagens PHP e Perl (embora o último item não seja interessante nesse contexto). A instalação é relativamente simples e utilizá-lo também é tranquilo, o pacote permite rodar aplicações web PHP sem muitos problemas. Útil para rodar o PHP em uma máquina local sem precisar instalar todo um sistema operacional novo nela.

• Recomendação de ouro: sistema operacional Linux Ubuntu

Embora não seja tão familiar quanto o Windows, ele dispõe de diversos recursos que facilitam demais o desenvolvimento. Uma das principais vantagens é que com ele é possível rodar o website na *própria* máquina, ao invés de conectar no servidor da escola toda vez e subir os arquivos, aumentando DEMAIS a produtividade (e gerando independência do servidor da escola, que costuma cair).

• Recomendação de ouro: Git

O Git é uma ferramenta de versionamento de arquivos. Ele possibilita que diversas pessoas mexam no mesmo arquivo de um projeto sem que haja conflito de versões. Facilita DEMAIS o trabalho em equipe, e todos os projetos de qualquer empresa usa o Git. é difícil se habituar a ele começo, mas vale muito a pena aprender a usá-lo. Para utilizar o Git, é preciso utilizar uma plataforma que hospede o seu projeto (como o Github ou Gitlab), e um cliente (como o Git Kraken) pode ser utilizado para facilitar o uso do Git.

**Php na web**

Php significa Php Hypertext Preprocessor (Php: pré-processador de hipertexto) (sim, o nome é esse e eu ADORARIA estar brincando). Isso quer dizer que ela é executada antes de um arquivo hypertext (por exemplo, um arquivo HTML).

Na prática, arquivos de PHP podem conter trechos de HTML, e isso significa que, ao interpretar o arquivo, primeiro todo o PHP será executado, e depois o HTML será carregado no site. Mas a pegada é que é possível gerar trechos de HTML condicionais (quando o HTML está dentro de um if ou switch do PHP) ou trechos repetidos (dentro de um laço de repetição), por exemplo.

Além disso, o PHP anda de mãos dadas com 2 outros recursos importantes no desenvolvimento web: banco de dados e requisições HTTP.

**Banco de dados**

*Nota: essa apostila não vai te ensinar sobre banco de dados a fundo*

Bancos de dados são, grosseiramente falando, aplicações que permitem armazenar diversos dados de uma maneira organizada. O PHP tem diversas funções, já integradas e documentadas, para acesso de banco de dados (em especial os da família SQL).

**Requisições HTTP**

Requisições HTTP são pacotes de dados geralmente usados na web para transmitir informações de um lugar para outro, seja de um site para outro, de um site para si mesmo ou até de um site para a sua própria máquina (ou vice-versa!).

Requisições HTTP são maneiras de uma aplicação web se comunicar com ela mesma e com o resto da web. Por exemplo, quando uma página web é acessada, o navegador realiza um HTTP request para o servidor da página, para que ele possa carregá-la. Ou quando um login e feito em algum website, ele realiza um HTTP request para o próprio site para verificar e logar o usuário.

É possível ver quais HTTP requests são realizados numa página utilizando as developers tools do navegador (F12 > aba "networks")

**Aplicação prática**

**Executando o PHP**

Existem diversas maneiras de executar um script PHP. Para executar um arquivo no navegador, como se fosse uma página da web, é preciso colocá-lo em um servidor.

Caso o servidor esteja na sua máquina, basta colocar o arquivo na pasta em que o seu servidor executa. O XAMPP, mencionado acima, cria por padrão uma pasta com o endereço C:\xampp\htdocs\, e qualquer arquivo ou pasta colocado lá poderá ser acessado pela web. para acessá-los, digite localhost/ na barra de endereço do navegador. De maneira semelhante, instalar o servidor Apache no Linux criará uma pasta com o endereço /var/www/html, e funciona exatamente como a pasta do XAMPP.

Agora para acessar o servidor da escola, é preciso se conectar com ele por meio de um cliente FTP (como o WinSCP ou Filezilla), acessar a pasta /public\_sites/[seu nome]/ e transferir os arquivos do seu computador para lá. Assim, os arquivos vão se comportar exatamente como no XAMPP e no LInux. Para acessá-los, digite o endereço de IP do servidor da escola e seu nome, desta maneira: 200.145.153.175/[seu nome]/.

Tente, por exemplo, arrastar a pasta “apostila” que acompanha este documento até a pasta onde um servidor executa (como as pastas descritas acima) e então execute os arquivos.

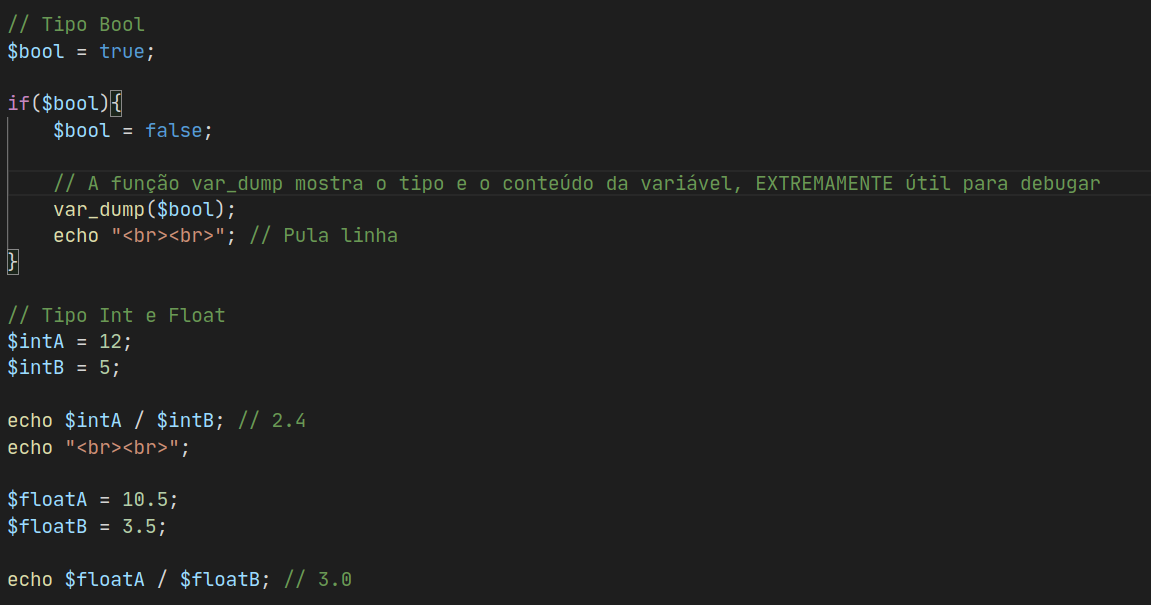
**Variáveis e Tipos**

O PHP é uma linguagem não tipada, e não é necessário declarar variáveis antes do uso. Isso significa que para utilizar uma variável basta atribuir um valor a ela. Também é possível mudar o tipo de valor (de Int para String, por exemplo).

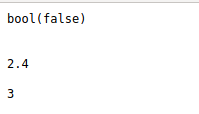
Os tipos do PHP são Bool, Int, Float, String, Array, Object e Resource.

Bool, Int e Float funcionam de maneira similar ao C/C++, sem complicações.

Código (Bools, Ints e Floats):

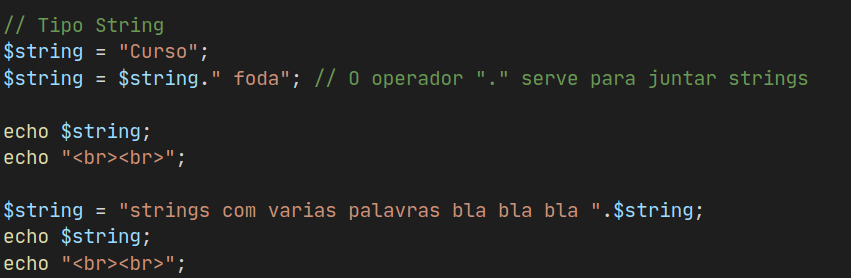


Saída:

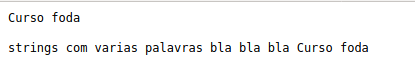


Strings, por sua vez, são muito mais fáceis de se utilizar do que em C/C++. Elas não precisam ser declaradas como arrays, e não precisam ter tamanho limitado. Existe INÚMERAS funções para lidar com strings (como deixar tudo minúsculo, maiúsculo, procurar uma substring, substituir certos elementos, quebrar em um array e mais) também.

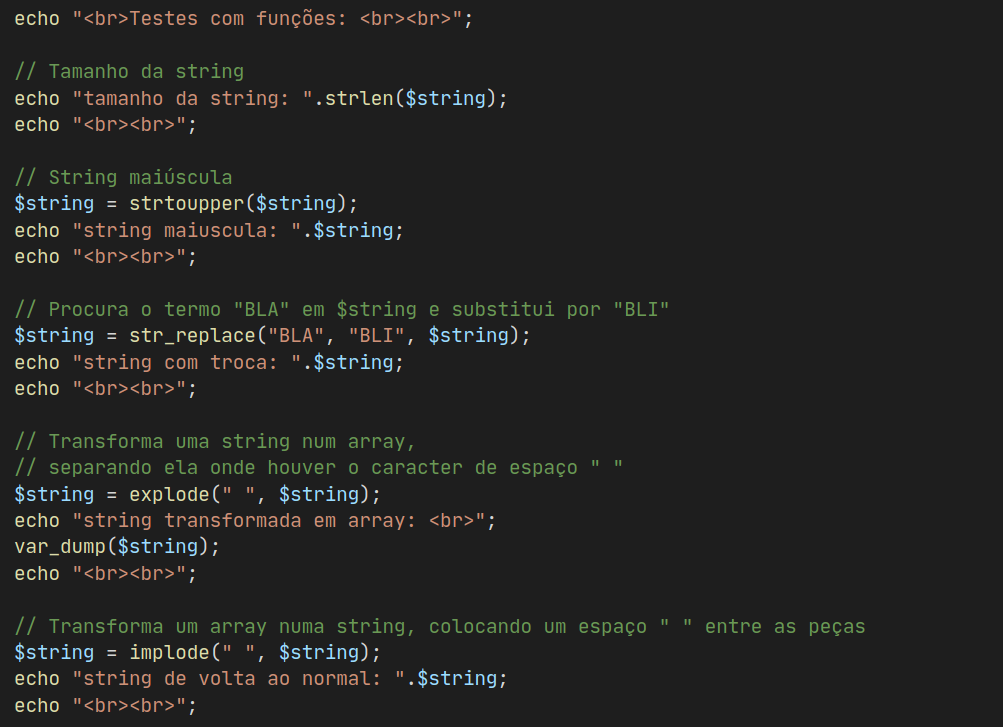
Código (Strings):



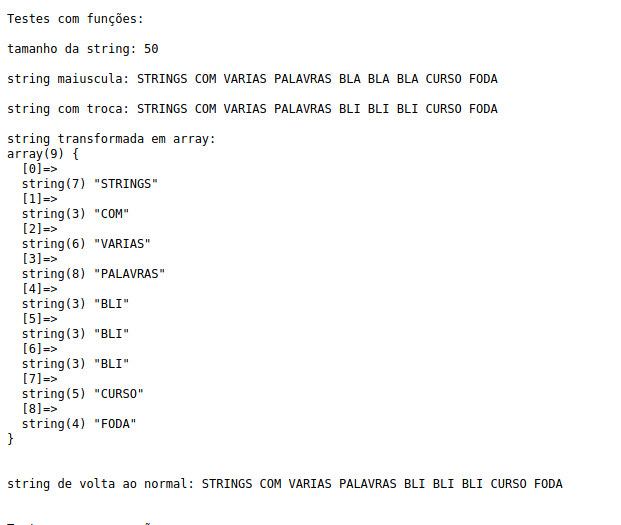
Saída:



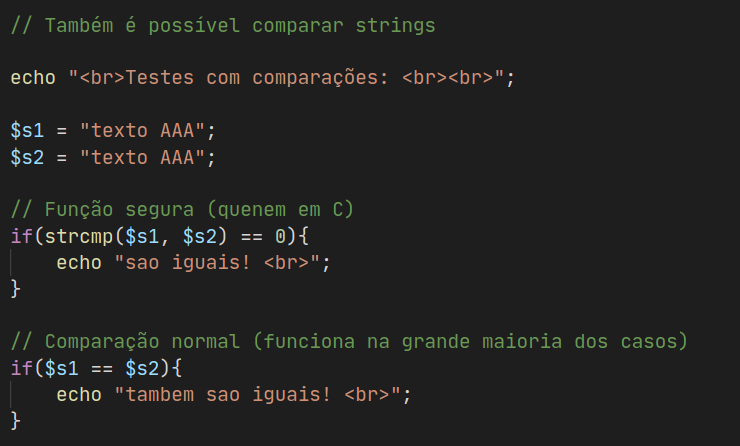
Código (Manipulando Strings):



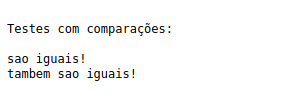
Saída:



Código (Comparação de Strings):

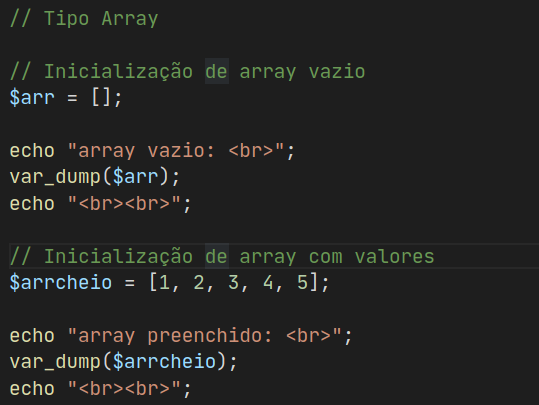


Saída:

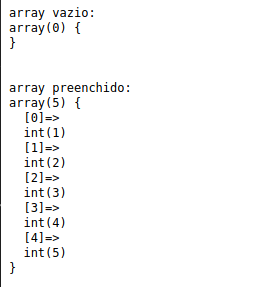


Arrays em PHP também não precisam de um tamanho fixo, e elementos de diversos tipos podem ser armazenados em um mesmo array (ou seja, um mesmo array pode armazenar Ints, Strings, Floats, etc). Arrays podem ter tanto strings quanto números como índices (também chamados de chaves).

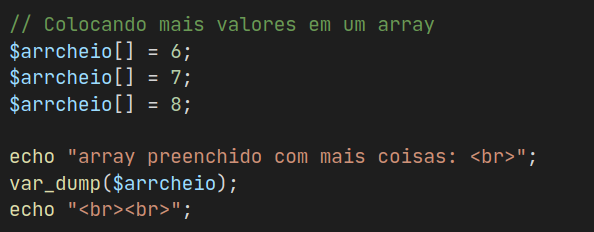
Código (Arrays):



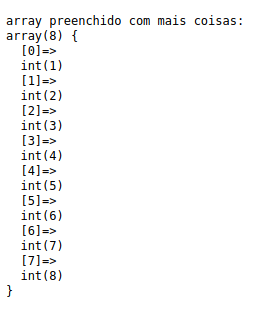
Saída:



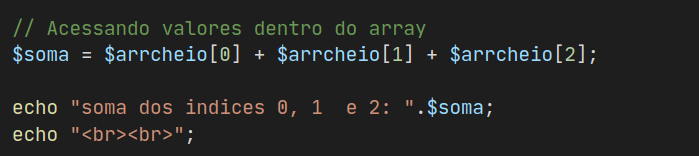
Código (Colocando itens no Array):



Saída:



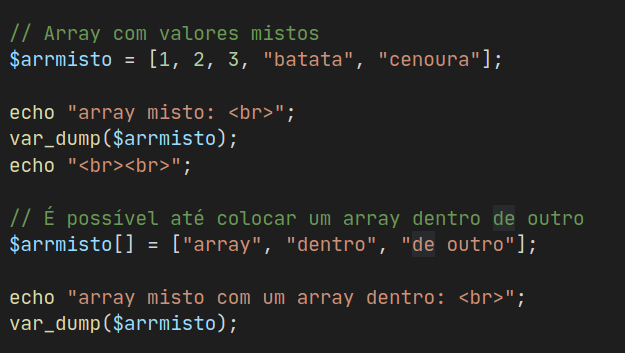
Código (Acessando valores no Array):



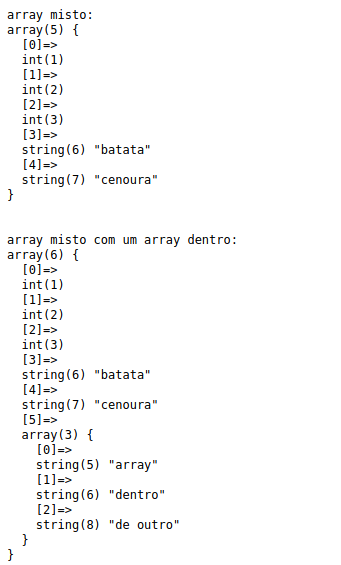
Saída:



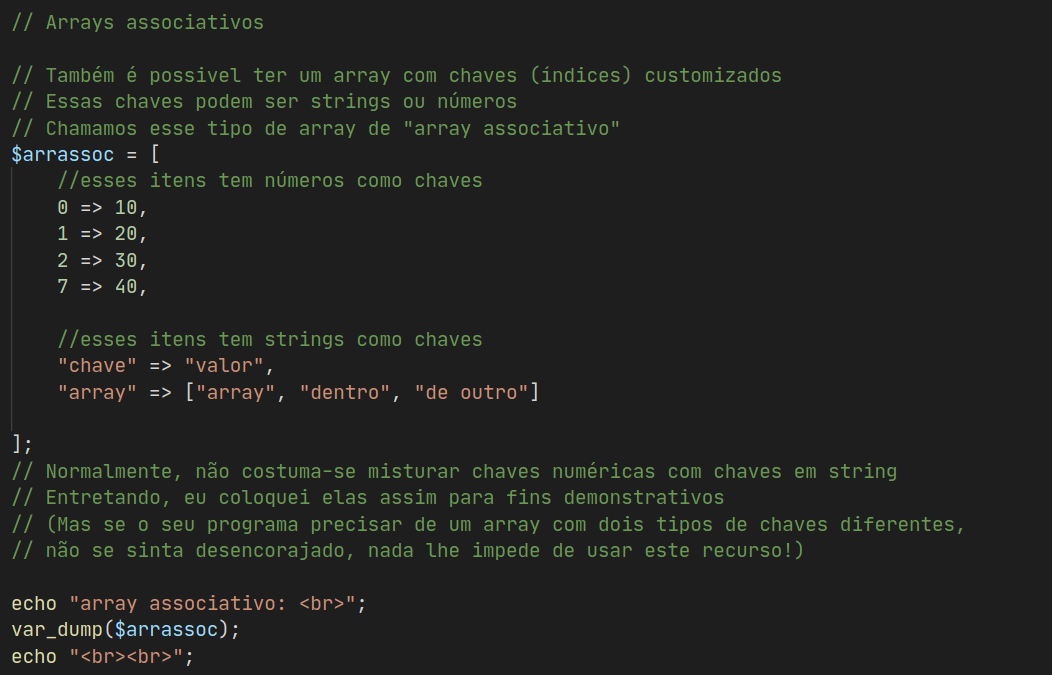
Código (Array misto):



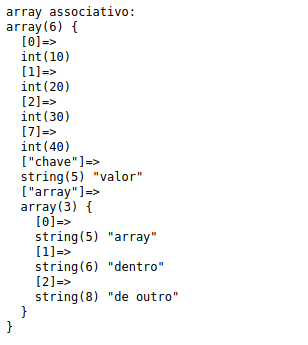
Saída:



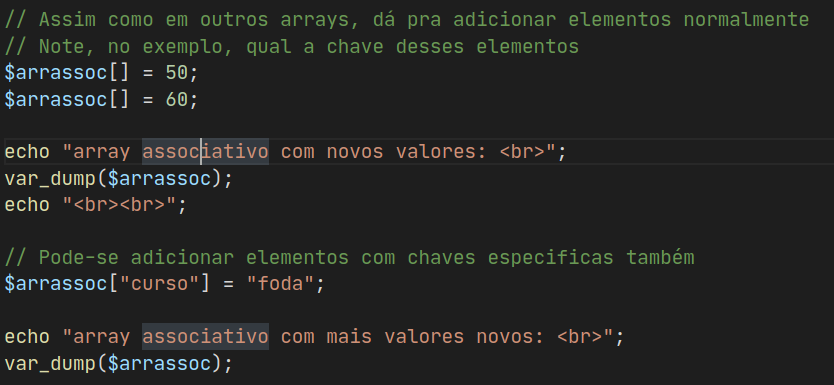
Código (Array associativo):



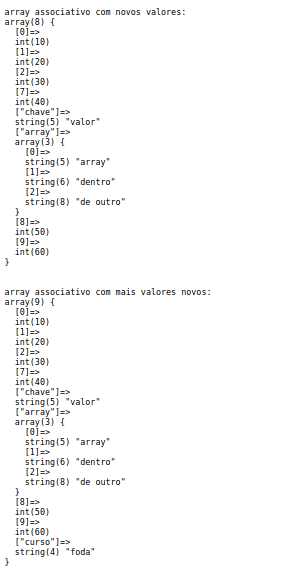
Saída:



Código (Operações com Array associativo):



Saída:



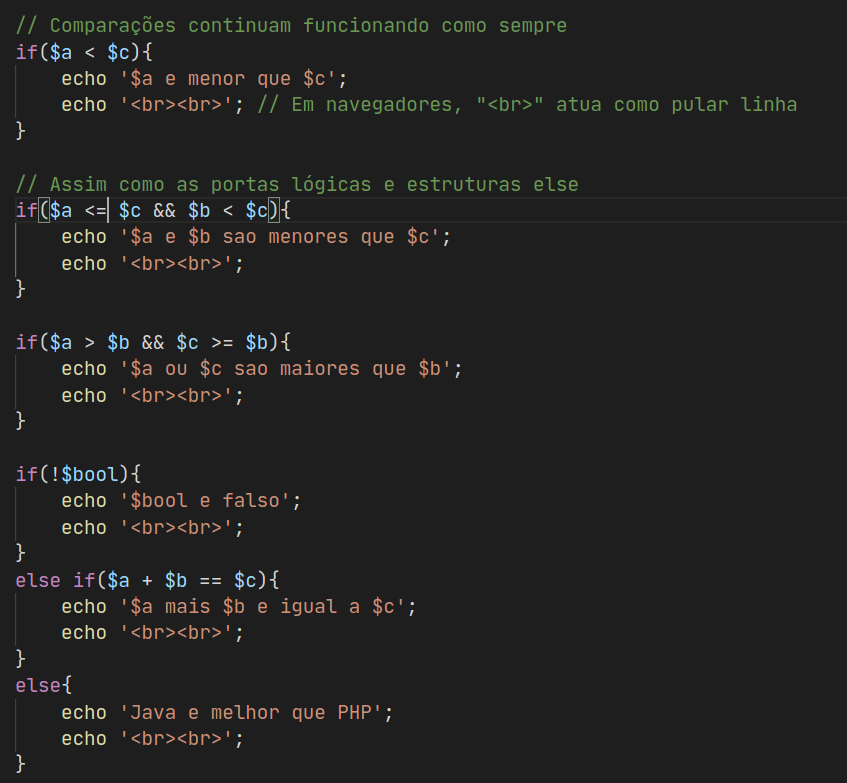
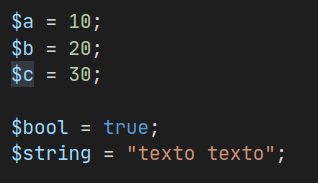
Objects também funcionam de maneira semelhante ao C/C++, tendo seus próprios métodos e propriedades.

Resources são um tipo especial de objeto, que referencia algum acesso à alguma coisa externa ao PHP. Por exemplo, para abrir um arquivo em PHP, é necessário utilizar uma função fopen(), que retorna um Resource. Esse Resource passa a referenciar o arquivo, e com ele é possível utilizar outras funções para manusear este arquivo

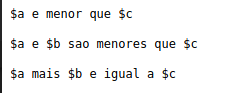
**Condições**

As condições em PHP funcionam de maneira semelhante ao C/C++, assim como outras linguagens. Entretanto, é possível comparar variáveis de tipos diferentes, e o switch funciona com strings além de números.

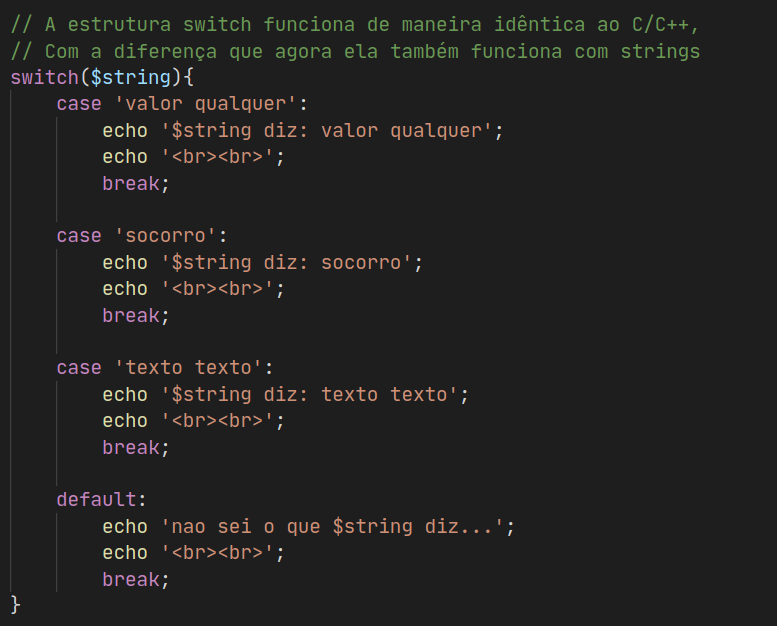
Código (Comparações com If):



Saída:



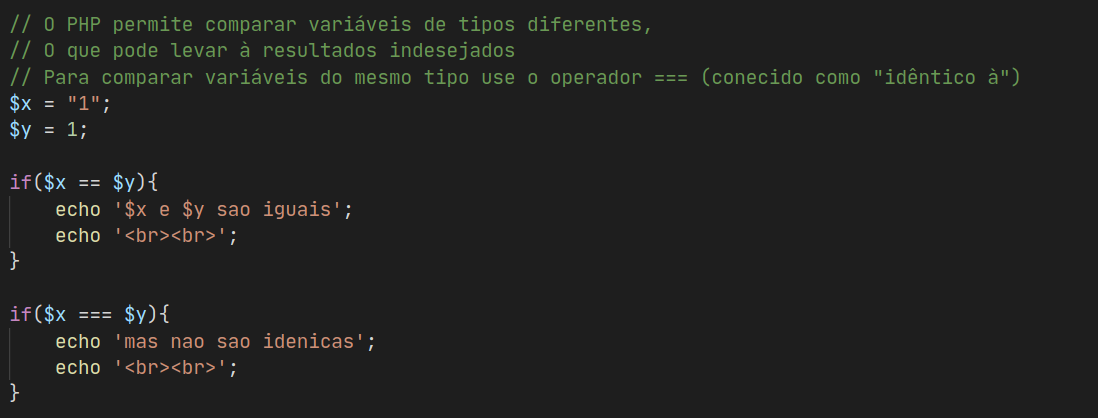
Código (Comparação com Switch):



Saída:



Código (Comparação igual e idêntica):



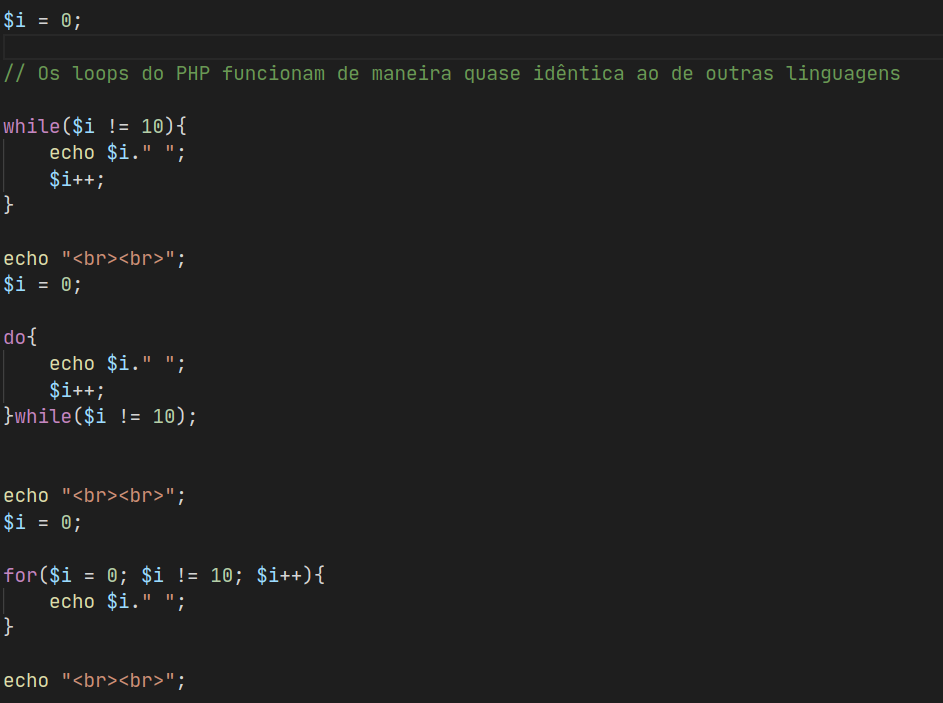
Saída:



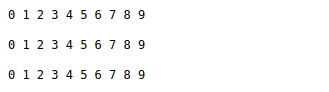
**Loops**

Loops de repetição funcionam de maneira muito parecida com o C/C++ (ou qualquer outra linguagem)

Código (Loops comuns):

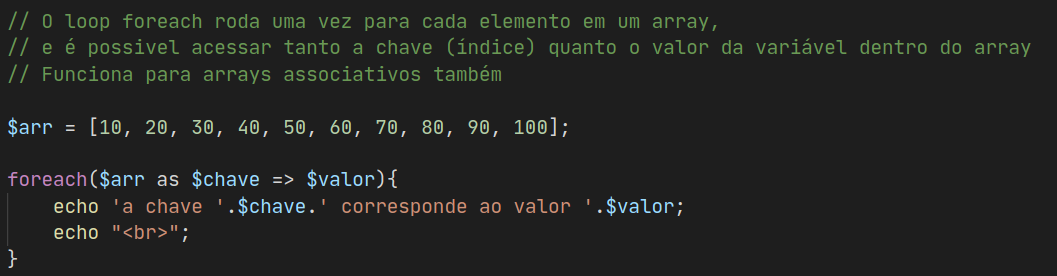


Saída:

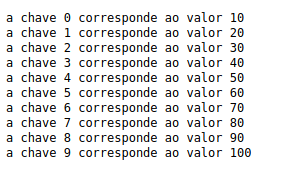


O PHP tem uma implementação de foreach, uma função extremamente útil para iterar sobre um array, com acesso aos elementos e suas chaves

Código (Foreach):



Saída:

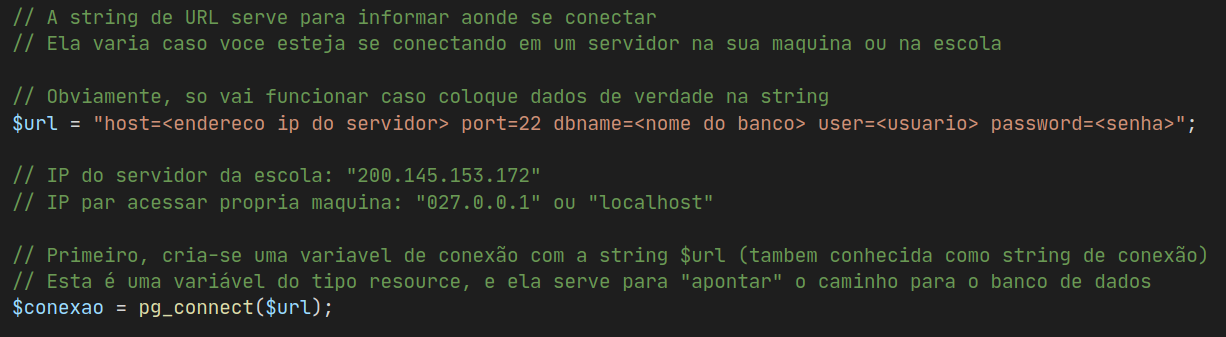


**Acessando o banco de dados**

O PHP vem com funções prontas para acessar banco de dados da família SQL. Nos exemplos à seguir eu vou demonstrar como realizar dois tipos de query: retorno e sem retorno com o PostgreSQL, banco de dados padrão dos servidores da escola.

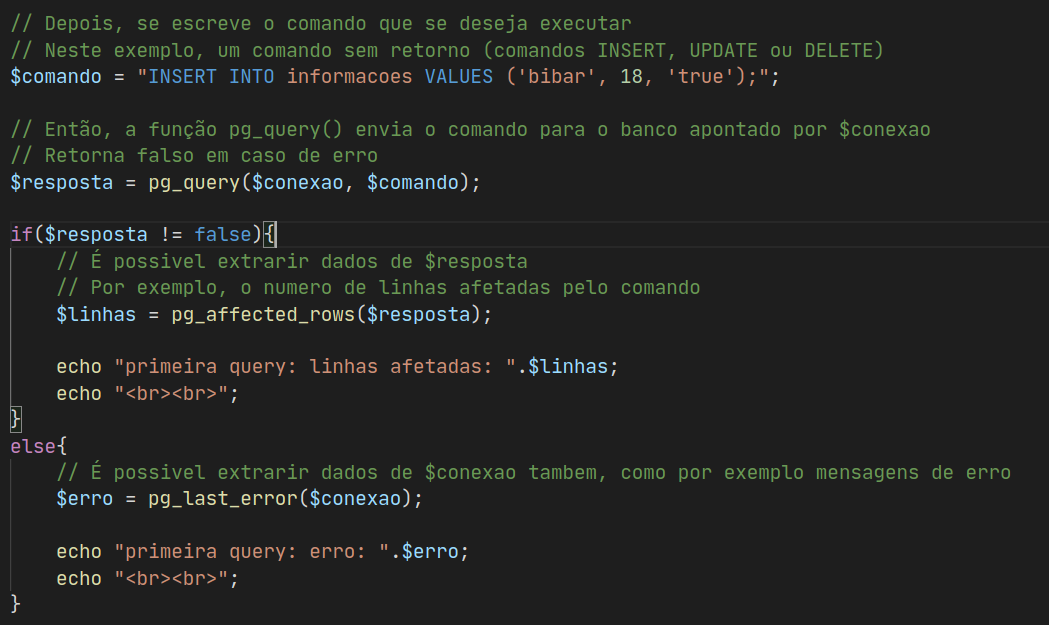
Para realizar uma requisição ao banco, primeiro é criada uma string informando sobre o banco, endereço, senha, etc. e então um resource que representa a conexão à esse banco de dados.

Código (Conexão ao banco):

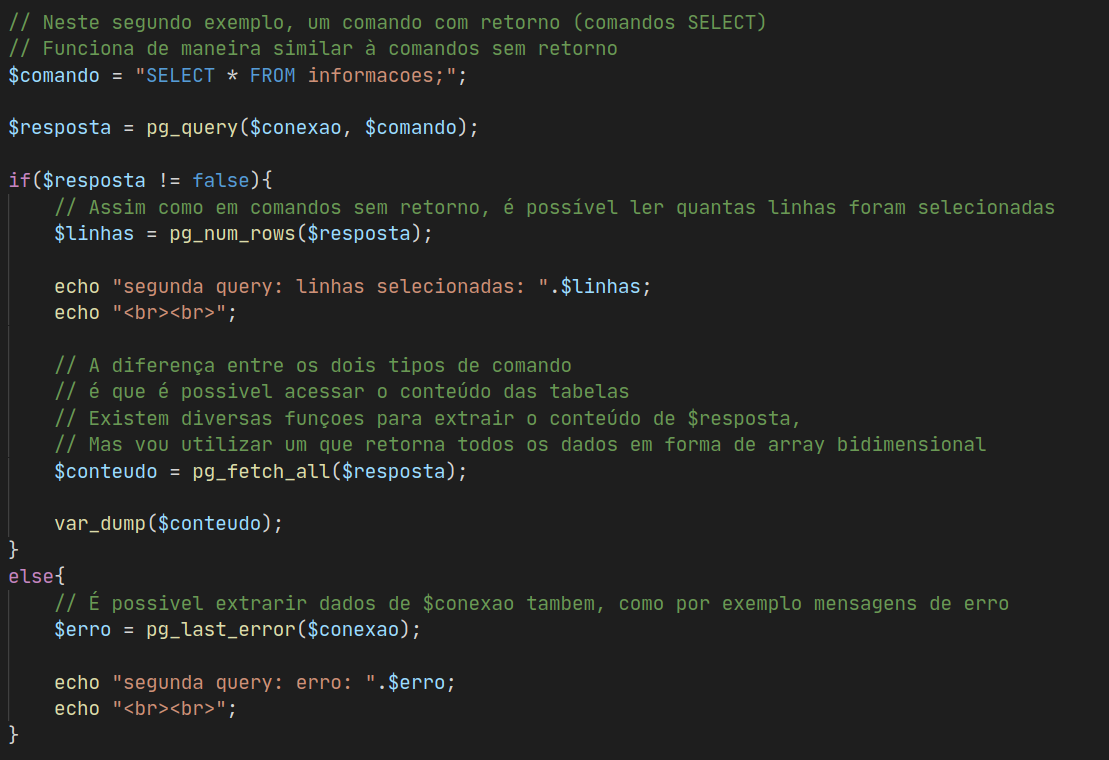


Depois se escreve o comando que se deseja executar e o envia para uma função junto do resource apontando para o banco de dados, assim obtendo uma resposta. Então se extrai dados da resposta sobre o comando (como o número de linhas afetadas, dados trazidos, etc.) para que se possa imprimir os dados ou executar códigos diferentes dependendo resposta.

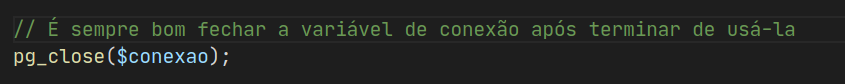
Código (Acessando o banco (sem retorno)):



Código (Acessando o banco (om retorno)):



Uma lista das funções do PostgreSQL do PHP se encontra [aqui](https://www.php.net/manual/pt_BR/ref.pgsql.php).

****

**Recebendo Variáveis**

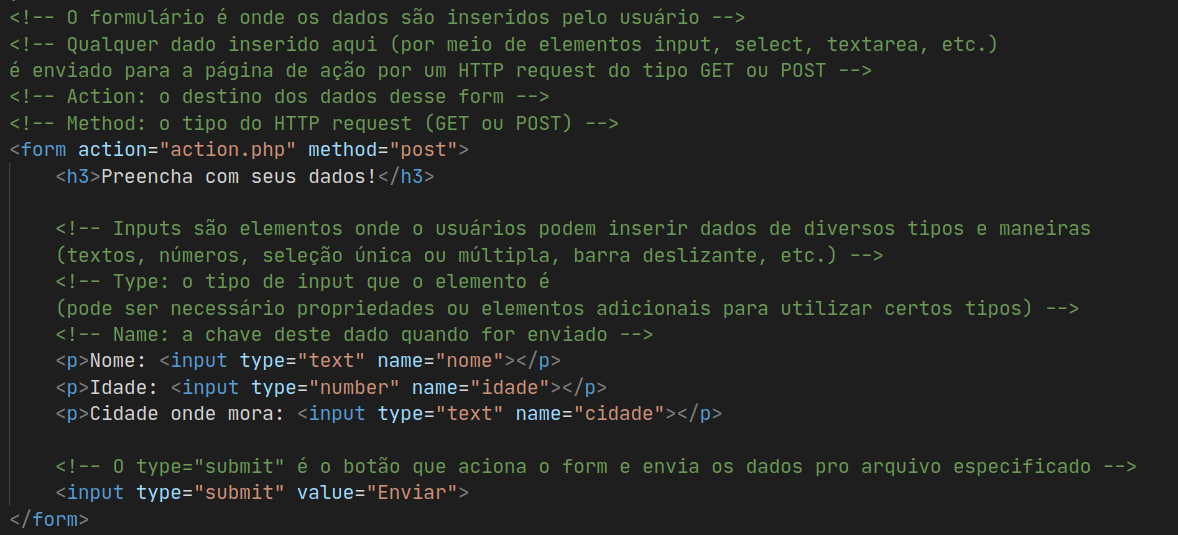
Existem diversas maneiras principais de receber variáveis no PHP. No caso de uma aplicação para o terminal, é possível utilizar recursos parecidos com a função scanf do C, ou realizar uma requisição HTTP para buscar algum recurso pela própria web, por exemplo.

Entretanto, para receber valores de um formulário da sua página, é comum receber por meio de GET ou POST, que são requisições Http. Quando um script recebe dados vindo de uma requisição do tipo GET ou POST, os dados recebidos ficam armazenados em variáveis globais $\_GET e $\_POST. Ou seja, os dados para um script PHP por meio de HTTP podem ser lidos a partir dessas variáveis e então utilizados no código.

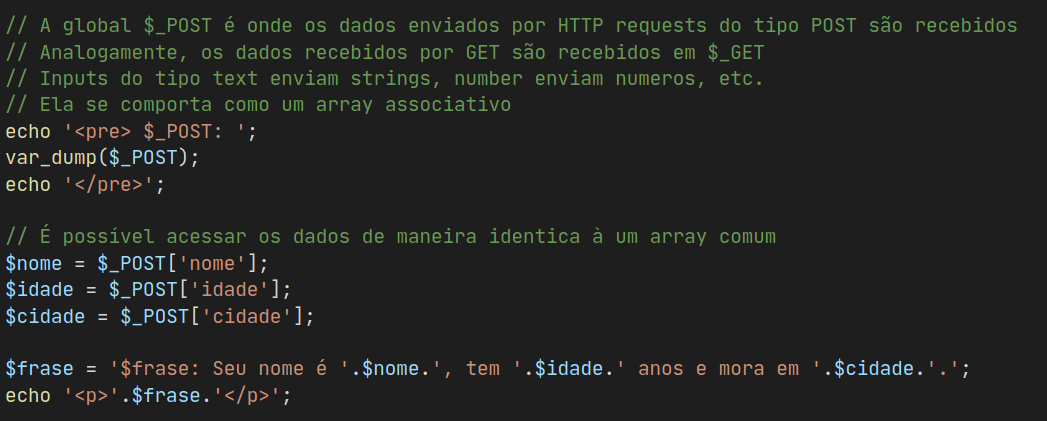
As duas maneiras mais comuns de se enviar dados de uma página para um arquivo PHP são por meio de um formulário HTML ou por meio de AJAX, cada um com suas próprias vantagens e desvantagens.

Pelo formulário HTML é muito mais simples de ser executada, porém ela é limitada e acaba redirecionando o usuário, causando um recarregamento da página. Ela é ativada quando um Input do tipo Submit é ativado em um formulário html, enviando os dados para o destino especificado no formulário

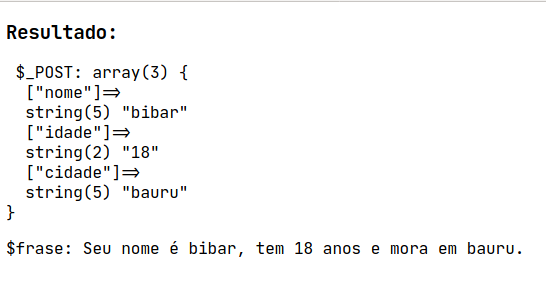
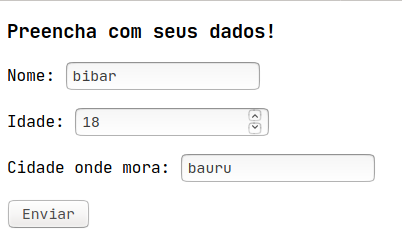
Código (Formulário HTML):



Código (Script action.php):

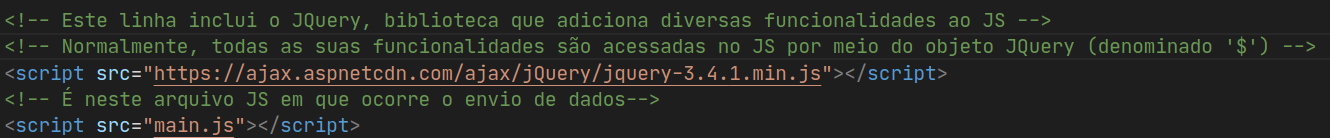


Saída:

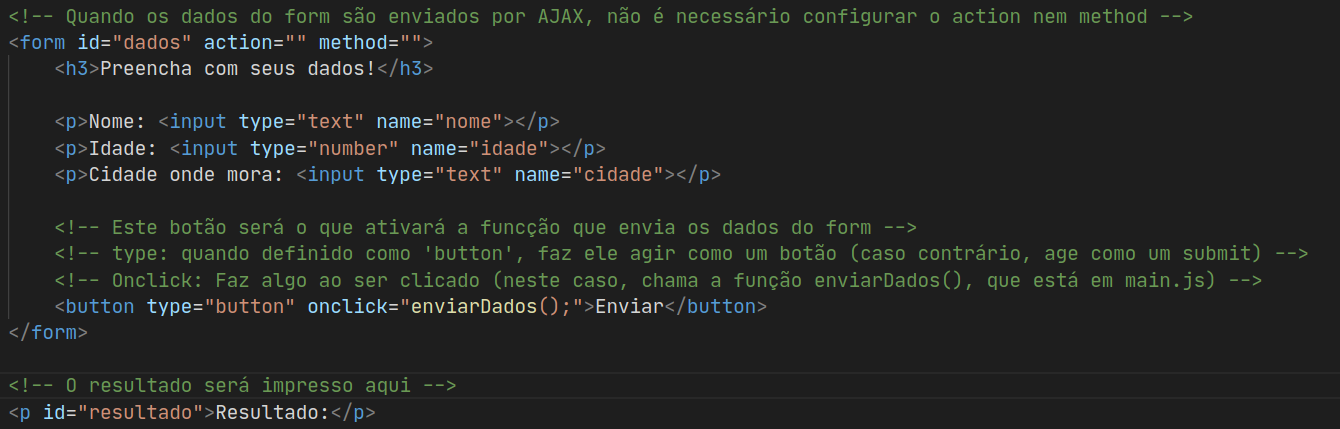


Já pelo AJAX (que significa asynchronous javascript and xml, ou javascript e xml assíncronos), é possível enviar requisições à um arquivo e obter sua resposta sem sair da página, entretanto é necessário um pouco de conhecimento de javascript e costuma ser um pouco chato aprender como funciona o AJAX. Apesar de ser assustador, é interessante aprender como funciona o AJAX porque ele não apenas funciona muito bem com o PHP, mas também com outros recursos da web (como APIs) que podem ser úteis para você no futuro! Então eu peço de bom grado para que se esforce um pouco para entender o AJAX, pois o conhecimento vale a pena.

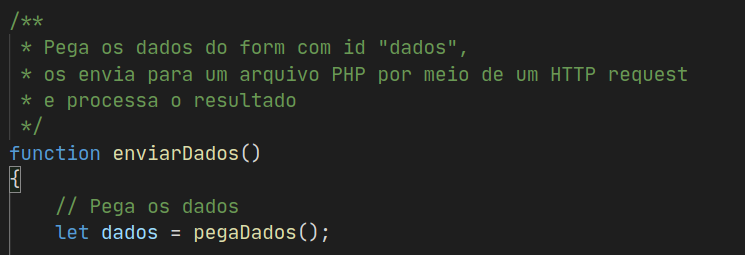
Código (Importando o JQuery e o seu script JS na <head>):



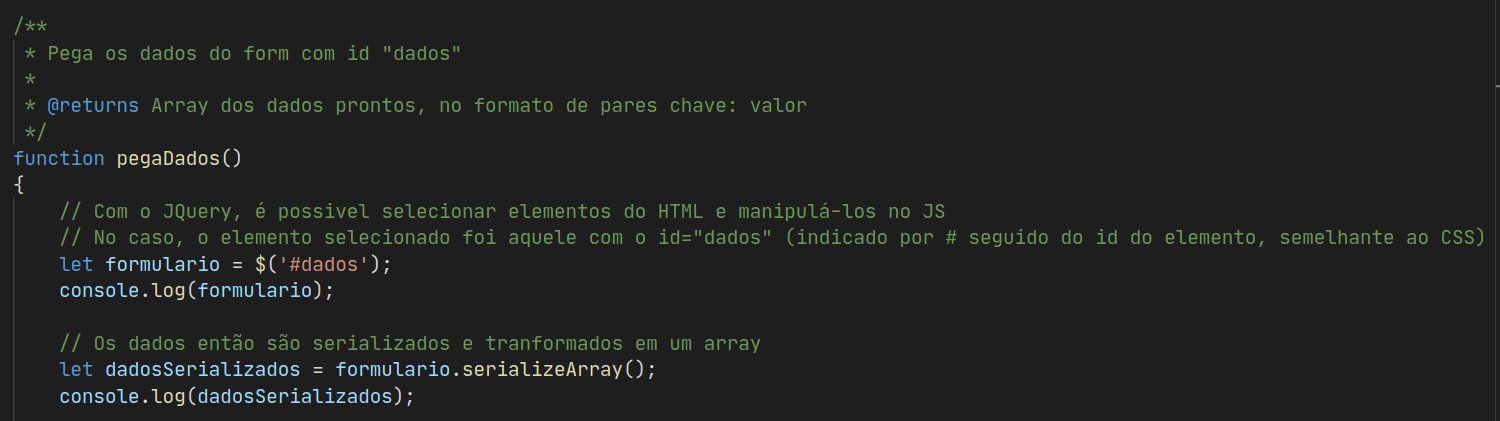
Código (Formulário HTML (levemente diferente do anterior)):



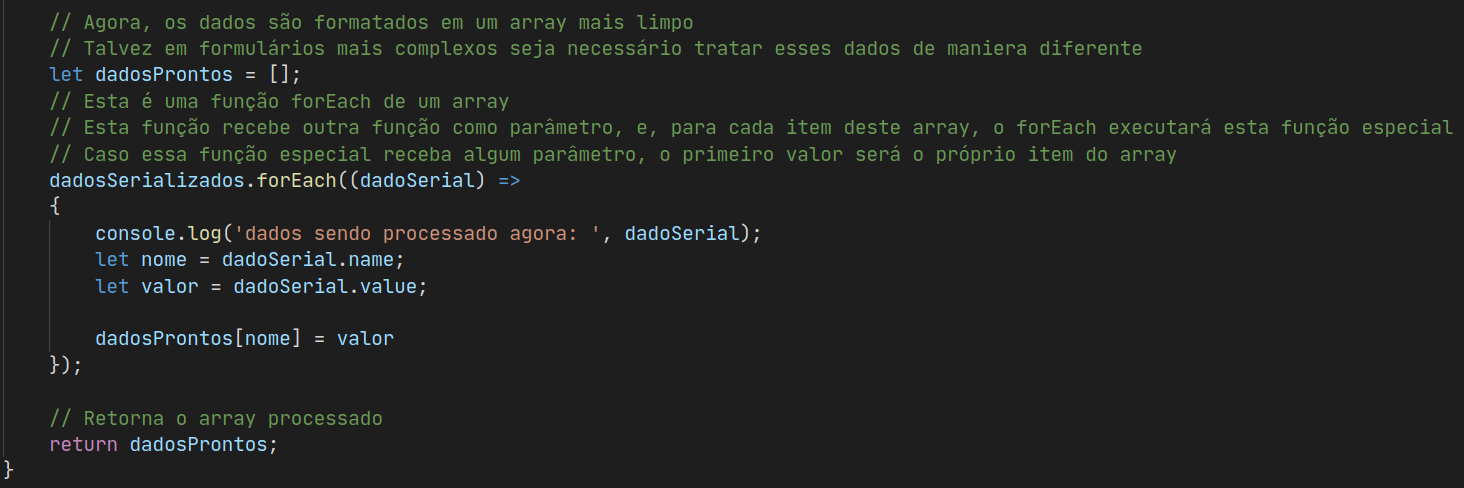
Código (Em main.js: o início da função enviarDados(), chamada pelo botão no HTML):



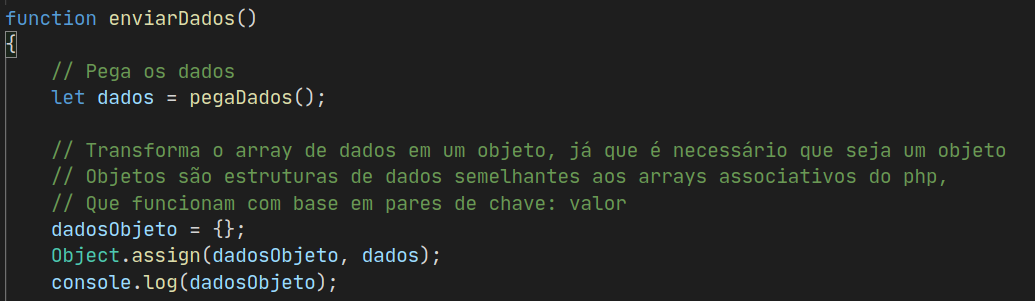
Código (Inicio da função pegaDados(), que é chamada pela função enviarDados()):



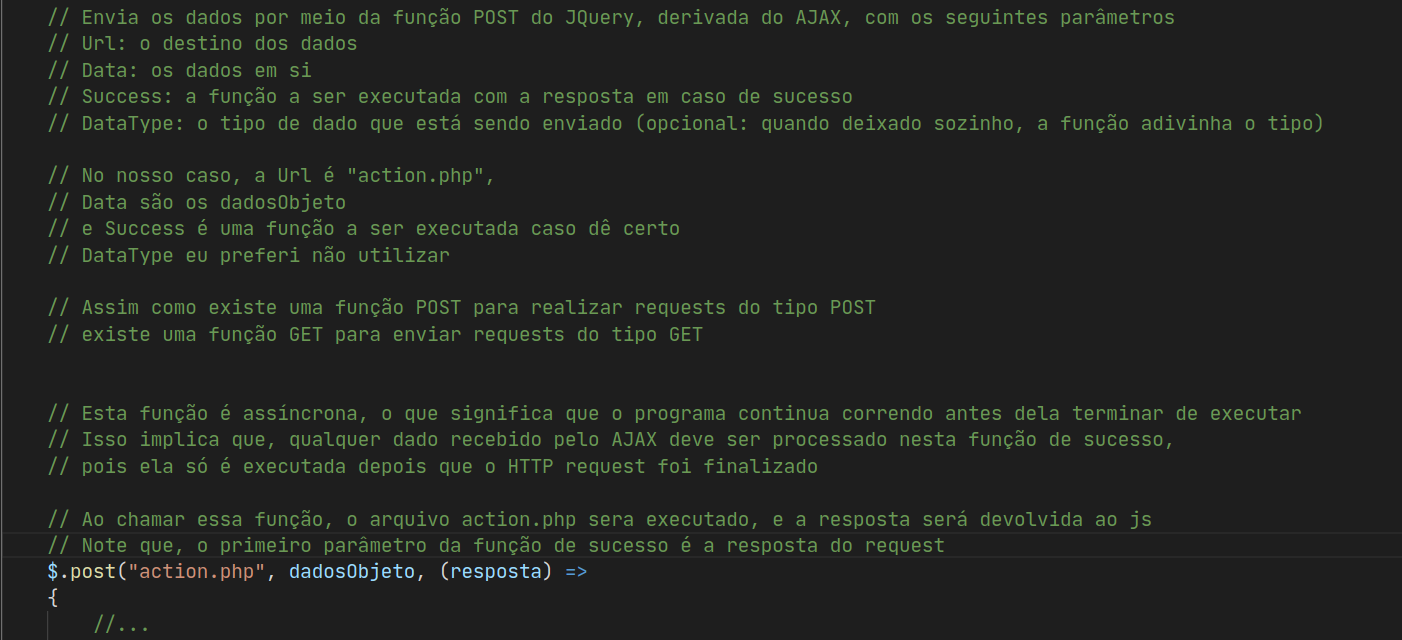
Código (Formatação de dados em pegaDados()):



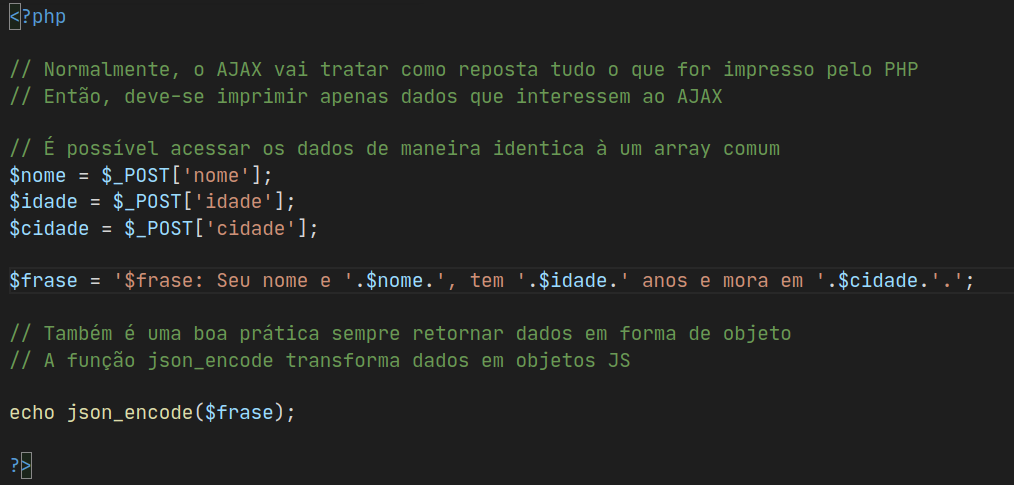
Código (De volta à enviarDados(), formata os dados do form mais um pouco):



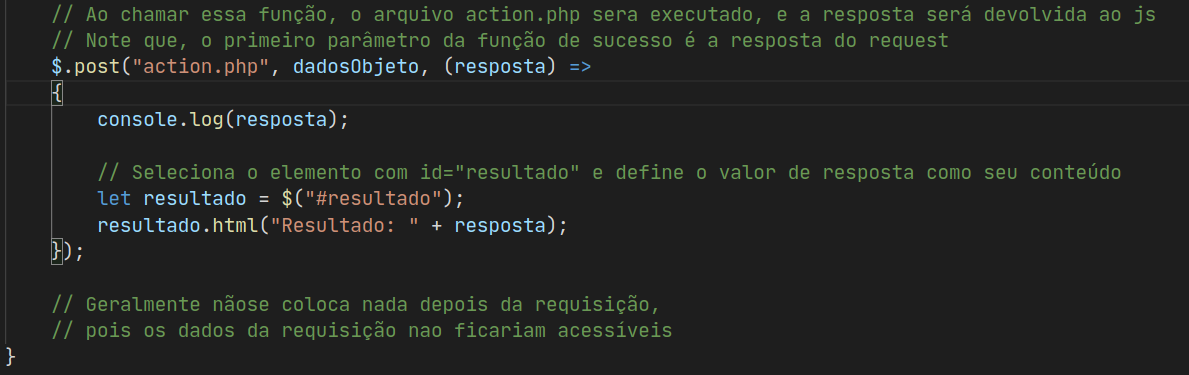
Código (Comentários sobre o AJAX (o AJAX é aquela instrução “$.post(...)”)):



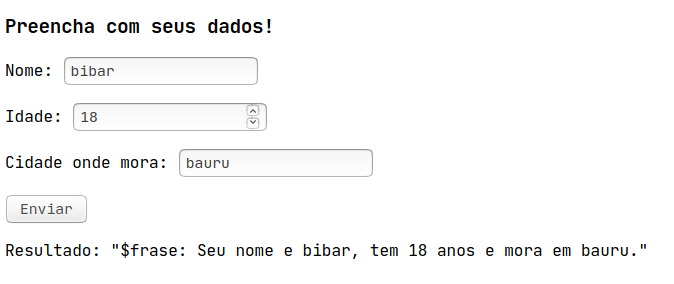
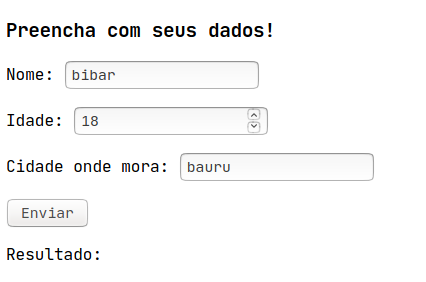
Código (O arquivo action.php, chamado pelo AJAX na função anterior. Tudo que for impresso por esse script será mandado de volta para main.js como retorno):



Código (De volta a main.js: tratamento do retorno):



Saída:



**Links:**

[Xampp](https://www.apachefriends.org/pt_br/download.html) (vem com o PHP e o Mysql já)

[Linux Ubuntu](https://ubuntu.com/download/desktop) ([Tutorial de instalação do SO](https://e-tinet.com/linux/instalar-ubuntu/) e [instalação do PHP e MySQL](https://www.digitalocean.com/community/tutorials/como-instalar-a-pilha-linux-apache-mysql-php-lamp-no-ubuntu-18-04-pt))

[FileZilla](https://filezilla-project.org/)

[WinSCP](https://winscp.net/eng/download.php)

[Visual Studio Code](https://code.visualstudio.com)

[Atom](https://atom.io/)

[Git](https://git-scm.com/downloads)