Micro Teaching | Aula 01

Json CRUD - Reading

DigitalHouse > Coding School

JSON

Java**S**cript **O**bject **N**otation

CRUD

Create **R**ead **U**pdate **D**elete

Introdução ao JSON

DigitalHouse >
Coding School

Para que serve?

Serve para a transferência de dados de uma forma simples e ágil, permitindo acessá-los como arrays ou objetos.

Um pouco da história

O Json não foi desenvolvido, mas sim descoberto! Isso em 1996, na Netscape. Mas só se popularizou em 2001.

Vantagens

1. Simplicidade:
Simples de ler e de escrever (só utiliza [], {} e ,)

Vantagens

2. Versatilidade:

Pode ser interpretado como um **array** ou como um **objeto**

Vantagens

3. Independência:
O **Json** interage com diferentes linguagens

Vantagens

4. Agilidade:

Por conta de sua simplicidade, acaba sendo muito ágil

XML x JSON

O **Json** acaba sendo mais fácil de se trabalhar (mas em termos de performance, têm o mesmo desempenho).

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<youtube>
    <canal>
        <nome>Canal 01</nome>
        <data_inscricao>25/01/2019</data_inscricao>
    </canal>
    <canal>
        <nome>Canal 02</nome>
        <data_inscricao>25/12/2019</data_inscricao>
    </canal>
</youtube>
```

JSON

```
{
    "youtube": {
        "canais": [
            {
                "nome": "Canal 01",
                "data_inscricao": "25/01/2019"
            },
            {
                "nome": "Canal 02",
                "data_inscricao": "25/12/2019"
            }
        ]
    }
}
```

Acessando o arquivo JSON

Atrelamos nosso arquivo **Json** à uma variável:

```
$meuArquivoJson = “nomeDoArquivo.json”;
```

Acessando os dados

Capturamos o conteúdo do arquivo com a função **file_get_contents()**

```
$meuJson = file_get_contents($meuArquivoJson);
```

Transformando em Array

Transformamos nosso **Json** (que é uma **string**) em **array** com **json_decode()**

```
$meuJsonArray = json_decode($meuJson, true);
```

Transformando em Objeto

Também podemos usar **json_decode()** para transformarmos o **Json** em Objeto, basta trocarmos ***true*** por ***false***

```
$meuJsonObj = json_decode($meuJson, false);
```

Como o false é o valor default, não é necessário declará-lo.

Acessando os Arrays

Acessamos os valores do nosso **array** normalmente, utilizando recursos como **loops** ou simplesmente acessando os **índices**.

```
foreach($itens as $item)
foreach($itens as $chave => $valor)
for, while, do while, $array[0], $array["nome"]...
```

DigitalHouse > Coding School

Acessando o Array do Array

Atrelando um **índice** do nosso **array** a uma variável

```
$indexYouTube = $meuJsonArray['youtube'];
```

Acessando o Array do Array

Atrelando mais um **índice** do nosso **array** a uma variável

```
$canais = $indexYouTube['canais'];
```

Percorrendo um Array Simples

Usando o **foreach()** para percorrer um **array simples**

```
foreach( $canais as $canal ){
    var_dump($canal);
}
```

Variável Variável

Podemos usar a seguinte sintaxe para criar variáveis variáveis

```
for( $i = 0; $i < count($canais); $i++){
    ${“canal”.$i} = $canais[$i];
}
```

Se tivermos 2 canais, criamos as variáveis $canal0 e $canal1

Percorrendo um Array Associativo

Usando o **foreach()** para percorrer um **array associativo**

```
foreach( $canalo as $key => $value ){
    echo $key . " = " . $value . "<br/>";
}
```

Exibindo os valores no HTML

Poderíamos estilizar a exibição desses valores com HTML/CSS

```
<ul>
<?php foreach( $canal0 as $key => $value ){
    echo “<li>” . $key . “ = ” . $value . “</li>”;
} ?>
</ul>
```

Obrigado!