

Estrutura de Projetos

- Projetos em Quasar
- Middleware de Pagamento
- Máscara de Código Interno do Produto
- Migração para PHP 8.2 - Principais Mudanças e Ajustes Necessários
- Tabela Propriedades

Projetos em Quasar

Como estruturar pastas e componentes nos projetos

Componentes

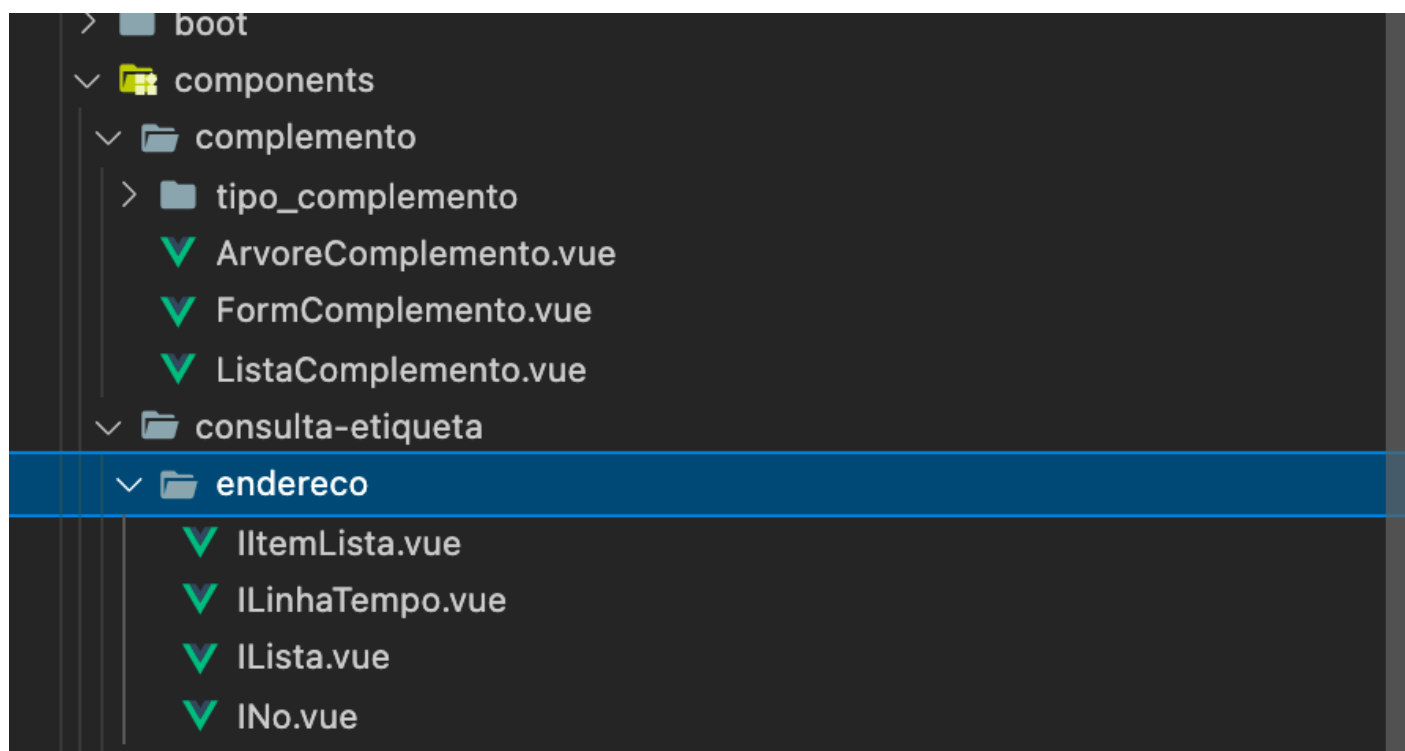
Para a nomeação dos componentes usamos a estrutura PascalCase. Quando o componente é personalizado pela equipe incluímos a letra "i" na frente

Exemplo de componente comum: AbrirModalCaixa
Exemplo de componente da illimitar: IAbrirModalCaixa

Pastas

Para a nomeação das pastas usamos a estrutura snake_case

Exemplo: minha_pasta



Middleware de Pagamento

Processo intermediador (middleware) de pagamento, baseado no fluxo da biblioteca CliSitef, para integração com sistemas terceiros durante o processo de intermediação e finalização de pagamento.

Durante o processo, apresentado no arquivo em anexo, o fluxo deverá ser compreendido da seguinte forma:

```
IniciaFuncao() > ContinuaFuncao() > [... FICA EM LOOPING ATÉ INDICAR CONCLUIDO ...] > FinalizaFuncao()
```

O processo deverá executar antes das transações, no início do processo da forma de pagamento com o método

```
IniciaFuncao()
```

e finalizar após o registro da venda.

Caso o processo seja interrompido entre o processo de pagamento e o registro de venda, deve ser enviado ao fluxo através do método

```
InterrompeFuncao().
```

Layout das Funções ▣

Endpoint:

```
IniciaFuncao()
```

Parâmetros a ser passado ao endpoint:

```
{
  "data": {
    "cliente": {
      "id": 1,
      "nome": "Cliente Padrão",
      "documento": "000.000.000-33",
      "uf": "RJ"
    },
    "tabelaVenda": "0",
    "tabelaAtacado": "0",
    "vendedor": {
      "id": 1,
      "nome": "Vendedor Padrão",
    },
    "documento": "",
    "observacao": "",
    "conta": {
      "id": 1,
      "nome": "Conta Caixa",
    },
    "quantidadeVenda": 1,
    "totalVenda": 29.9,
    "totalDesconto": 0.00,
    "totalPagamento": 29.9,
    "totalTroco": 0.00,
    "itens": [
      {
        "id": 1,
        "id_produto_grade": 129447,
        "id_produto": 129447,
        "codigo": "925818196",
        "descricao": "TOP SUPLEX SUPLEX BICOLOR TRANCADO COSTA ",
        "unidade": "UND",
        "quantidade": 1,
        "valor": 29.9,
        "valor_venda": 29.9,
        "valor_custo": 15.77,
        "valor_atacado": 29.9,
        "oculta_custo": "",
        "idPreco": 0,
```

```
    "valor_pago": 29.9,
    "valor_total": 29.9,
    "qtdeAtacado": 0,
    "ncm": "61044900-00-00000000",
    "codigo_ean": "",
    "id_codigobarra": "",
    "limite_desconto": 0,
    "promocao": [],
    "produto_grade_imagem": false,
    "situacao": "ATIVO"
  }
],
"cancelados": [],
"pagamento": {
  "prazo": 6,
  "condicao": 11
},
"nfce": "1",
"operador": {
  "id": 1,
  "nome": "Operador Caixa",
},
"tokenVenda": "7e7b5cab-1ddd-6472-0b0f-a3ff2a583b55",
"dataEmissao": "2021-11-24 10:03:17"
}
}
```

Retorno do endpoint

```
{
  "success": true,
  "message": "Iniciado com Sucesso",
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
  }
}
```

Endpoint: ContinuaFuncao ▣

Parâmetros a ser passado ao endpoint ▣

```
{
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
    "comando": 1,
    "valor": false,
    "voltar": false,
    "interromper": false,
    "tokenVenda": "7e7b5cab-1ddd-6472-0b0f-a3ff2a583b55",
    "dataEmissao": "2021-11-24 10:03:17",
  }
}
```

▣

Se o parâmetro "voltar" for informado Sim [true], o processo está indicando que volte o fluxo.

Se o parâmetro "interromper" for informado Sim [true], o processo está indicando que interrompa o fluxo.

Retorno do endpoint ▣

```
{
  "success": true,
  "continue": true,
  "message": "Executado com Sucesso",
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
    "comando": 1,
    "tipoCampo": 0,
    "tamanhoCampo": {
      "minimo": 0,
      "maximo": 255,
      "decimais": 2,
      "truncar": false,
    }
  }
}
```

```

    },
    "titulo": "Título a Exibir Quando Houver",
    "buffer": "",
  }
}

```

Segue as tabelas de comando e tipo de campo.

Tabela com os códigos de Comando

Comando	Descrição
1	Mensagem para o visor do operador
2	Mensagem para o visor do cliente
11	Deve remover a mensagem apresentada no visor do operador
12	Deve remover a mensagem apresentada no visor do cliente
2000-2999	Deve apresentar o texto em buffer, e obter uma resposta do tipo SIM [true] / NÃO [false].
3000-3999	Deve apresentar um menu de opções e permitir que o usuário selecione uma delas. Na chamada o parâmetro buffer contém as opções no formato JSON (anexo abaixo).
4000-4999	Deve apresentar a mensagem em buffer, e aguardar uma tecla do operador. É utilizada quando se deseja que o operador seja avisado de alguma mensagem apresentada na tela.
5000-5999	Deve ser lido um campo cujo tamanho está entre tamanho.minimo e tamanho.maximo.
50	Está devolvendo o valor de <code>documento</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.
51	Está devolvendo o valor de <code>cliente</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém as opções no formato JSON (anexo abaixo).
52	Está devolvendo o valor de <code>desconto</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.
53	Está devolvendo o valor de <code>observacao</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.

54	Está devolvendo o valor de <code>acrécimo</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.
55	Está devolvendo o valor de <code>delivery</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.
56	Está devolvendo o valor de <code>emissão de NF-e</code> a ser definido pela aplicação, na chamada o parâmetro buffer contém o valor.
60	Informa a aplicação a interromper o processo.
7000-7999	Deve ser armazenado as informações em buffer para ser apresentado após ser executado o <code>FinalizaFuncao()</code> e aguardar uma tecla do operador, é utilizada quando se deseja que o operador seja avisado de alguma mensagem apresentada na tela no processo de finalização da venda.
100	Informa a aplicação que o processo foi concluído.

Anexo 1 - Formato de Lista do Comando = 21 retornado pelo buffer■

```
{
  "titulo": "Título",
  "itens": [
    { id: 1, label: "texto 1" },
    { id: 2, label: "texto 2" },
    { id: 3, label: "texto 3" },
  ]
}
```

Anexo 2 - Retorno do Comando = 51 retornado pelo buffer■

```
{
  "id": 1,
  "nome": "Cliente Padrão",
  "documento": "000.000.000-33",
  "uf": "RJ"
}
```

■

Tabela com os códigos de Tipo de Campo (tipoCampo)■

tipoCampo	Descrição
1	Alfa-numérico (string)
2	Inteiro
3	Valor com ponto flutuante (double), neste caso é possível informar a quantidade de casas decimais e se irá truncar
4	Monetário, sendo valor com ponto flutuante (double), neste caso é possível informar a quantidade de casas decimais e se irá truncar
5	Lista (array, de valor único)
6	Data (AAAA-MM-DD)
7	Data e Hora (AAAA-MM-DD HH:MM:SS)

Endpoint: InterrompeFuncao■

Parâmetros a ser passado ao endpoint■■■

```
{
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
    "tokenVenda": "7e7b5cab-1ddd-6472-0b0f-a3ff2a583b55",
    "dataEmissao": "2021-11-24 10:03:17"
  }
}
```

Retorno do endpoint

```
{
  "success": true,
  "message": "Interropido com Sucesso",
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
  }
}
```

```
}
```

Endpoint: FinalizaFuncao■

Parâmetros a ser passado ao endpoint■

```
{
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
    "totalVenda": 29.9,
    "totalDesconto": 0.00,
    "totalPagamento": 29.9,
    "totalTroco": 0.00,
    "pagamento": [
      {
        "id": 1,
        "valor": 29.9,
        "codigo": "TEF",
        "prazo": 14,
        "prazo_descricao": "REDE DEBITO",
        "condicao": false,
        "condicao_descricao": false,
        "descricao": "REDE DEBITO",
        "flag_pdv_pagamento": 3,
        "numero": "008018358",
        "complemento": {
          "numero": "008018358",
          "tef": ["..."],
          "raw": ["..."]
        },
        "troca": false,
        "vale": false
      }
    ],
    "tokenVenda": "7e7b5cab-1ddd-6472-0b0f-a3ff2a583b55",
    "dataEmissao": "2021-11-24 10:03:17"
  }
}
```

Retorno do endpoint ▣

```
{
  "success": true,
  "message": "Finalizado com Sucesso",
  "data": {
    "token": "cbb97b94-4dfd-11ec-8be9-000c291f9967",
    ["..."]
  }
}
```

Máscara de Código Interno do Produto

Este recurso permite a modificação da máscara padrão do **código interno** ao cadastrar um produto, determinando como será sua composição.

Para especificar a máscara do **código interno**, na **Configuração Geral do Sistema** (*Sistema > Configuração Geral*), no quadro **Produto**, na sessão Configuração da Máscara do Código Interno, existem parâmetros a serem especificados:

Tipo Produto

Determina qual o **Tipo** do **Produto** que será observado para a geração do código através da máscara determinada, podendo ser:

- Acabado;
- Grade Simples;
- Grade;

Máscara do Código

Máscara que será aplicada no **código interno** aos produtos onde o **Tipo** for igual ao definido no campo **Tipo Produto**;

Formatação da Máscara

A máscara do código interno deverá ser montada através dos campos relacionados a tabela `produto_grade` e seus relacionamentos, predeterminados por chaves {}, compostos por 3 parâmetros: { `parametro1` : `parametro2` , `parametro3` } .

parametro1

É o campo que será observado na construção da máscara.

Ele pode ser composto pelo nome do campo disponível na coluna `produto_grade` ou o nome **tabela** mais o **nome do campo** disponível na tabela relacionada a `produto_grade`.

Exemplo: `{campo}` ou `{tabela.campo}`.

parametro2

Campo opcional, que ao ser especificado se comportará das seguintes formas:

- 1. { parametro1 : tamanho } : insere zeros no início do valor string com base em um tamanho especificado.
- 2. { parametro1 : posição inicial - posição final } : retorna a parte do valor entre os índices inicial (posição inicial) e final (posição final).

parametro3

Campo opcional, que ao ser especificado se comportará da seguinte forma:

- 1. { parametro1 : parametro2 , tamanho } : insere zeros no início do valor string com base em um tamanho especificado.

Exemplos de Aplicação

Máscara	ID (Produto)	Código Interno
{produto.id:2-4,13}	3	00000000000000
	30	00000000000000
	301	00000000000001
	3015	00000000000015
	30153	00000000000015

Máscara	ID (Produto)	Código Interno
{produto.id:2,13}	3	00000000000000
	30	00000000000000
	301	00000000000001
	3015	00000000000015
	30153	00000000000153

Máscara	ID (Produto)	Código Interno
{produto.id:13}	3	00000000000003
	30	00000000000030
	301	00000000000301
	3015	0000000003015
	30153	0000000030153

Máscara	ID (Produto)	Código Interno
{produto.id:2-4}	3	
	30	0
	301	01
	3015	015
	30153	015

Máscara	Tipo de Produto	Descrição	Código Interno
{tipo_produto.nome}- {produto.descricao}	ADULTO	HUANITO MALAQUIAS	ADULTO-HUANITO MALAQUIAS

Máscara	Data de Cadastro	Referência	Código Interno
{produto.data_cadastro}- {produto.codigo}	2022-09-20 14:40:29	1122334455	2022-09-20 14:40:29- 1122334455

Máscara	Tipo de Produto	Descrição	Marca	Código Interno
{tipo_produto.nome} ABC{produto.descricao}@- 123{marca.nome}	algodão	Calça	adidas	algodãoABCCalça@- 123adidas

Máscara	ID Departamento	ID Marca	ID (SKU)	Código Interno
{departamento.id}-{marca.id}-{produto_grade.id}	8438	174	89192	8438-174-89192

Máscara Padrão

Por padrão, o código interno do produto segue a seguinte máscara:

Acabado

{produto_grade.id}

Desta forma, indica que o código interno será composto pelo **ID (SKU)** gerado ao **Produto**.

Exemplos:

	ID (SKU)	Código Interno
1	30153	30153
2	98762948	98762948

Grade / Grade Simples

{produto.id:6}{gradex.id:4}{gradey.id:4}

Neste formato, o código interno será composto pelo **ID (Produto)** gerado ao **Produto**, complementando com **6 dígitos** caso tamanho seja inferior a 6 e acrescido do **ID da Grade X (Horizontal)** e **ID da Grade Y (Vertical)**, complementando com 4 dígitos caso o tamanho for inferior a 4, respectivamente.

Quando o tamanho é superior ao definido, neste caso não irá efetuar o limite mínimo estabelecido.

Exemplos:

	ID (Produto)	ID Grade X (Horizontal)	ID Grade Y (Vertical)	Código Interno
1	11750	42	13	01175000420013
2	11686	65	25	01168600650025

Migração para PHP 8.2 - Principais Mudanças e Ajustes Necessários

A versão do PHP utilizada pelo sistema Illi está passando por uma atualização significativa.

A versão será migrada de **PHP 7.3.33** para **PHP 8.2.28**.

Com essa mudança, é esperado que enfrentemos algumas incompatibilidades e a necessidade de ajustes no código.

Além disso, mensagens de erro podem surgir nos logs, exigindo análise e correções pontuais.

Para auxiliar nesse processo, destacamos algumas das mudanças mais relevantes:

1. Acesso a índices inexistentes em arrays agora gera Warning

Antes (PHP 7.3):

Ao acessar um índice inexistente em um array associativo, era gerado um "Notice".

```
// Exemplo 1
$array = array();
echo $array['chave'];

// Exemplo 2
$valor['filhos'] = false;
$valor['filhos'][] = "Um Valor Qualquer";

// Exmplo 3
$array[$row["id_acesso"]][$row["id"]] = true;
```

Agora, esse mesmo acesso gera um *Critical*.

```
$array = [];  
echo $array['chave'];
```

Automatic conversion of false to array is deprecated

Depois (PHP 8.2):

Solução Recomendada:

Para evitar esse erro, utilize o operador de coalescência nula (`??`) ou verifique a existência da chave com `isset()`, conforme abaixo:

```
// Exemplo 1
$array = [];  
  
// Exemplo 2
$valor['filhos'] = [];  
$valor['filhos'][] = "Um Valor Qualquer";  
  
// Exemplo 3
$array = [];  
if (empty($array[$row["id_acesso"]])) {  
    $array[$row["id_acesso"]] = [];  
}  
$array[$row["id_acesso"]][$row["id"]] = true;
```

Referências:

[RFC: Deprecate implicit conversion of incompatible float keys](#)

[RFC: Deprecate implicit conversion of array keys](#)

2. Mudanças no retorno de funções de ordenação (`sort()`, `asort()`, etc.)

Antes (PHP 7.3):

As funções `sort()`, `asort()`, `ksort()`, entre outras, modificavam o array e retornavam `true` ou `false`, mas também podiam ser usadas diretamente no `var_dump()`.

```
$array = [3, 1, 2];  
var_dump(sort($array));
```

Depois (PHP 8.2):

Essas funções continuam modificando o array, mas agora retornam apenas `true` ou `false`.

```
$array = [3, 1, 2];  
if (sort($array))  
{  
    print_r($array);  
}
```

Solução Recomendada:

Evite usar `var_dump(sort($array))` para visualizar o resultado. Em vez disso, use `if (sort($array))` e imprima o array modificado separadamente.

Referências:

3. Novas funções para manipulação de arrays

A partir do PHP 8.1, foram introduzidas novas funções que facilitam a manipulação de arrays.

Antes (PHP 7.3):

Não existiam funções nativas para verificar se um array era indexado sequencialmente ou para obter a primeira e a última chave de um array.

Depois (PHP 8.1+):

Agora, contamos com:

- `array_is_list()`: Verifica se um array é uma lista sequencial.
- `array_key_first()`: Retorna a primeira chave de um array.
- `array_key_last()`: Retorna a última chave de um array.

Exemplo de uso:

```
$array = [10, 20, 30];  
if (array_is_list($array))  
{  
    echo "O array é uma lista válida";  
}
```

```
$array = ['a' => 1, 'b' => 2];  
echo array_key_first($array); // Retorna "a"  
echo array_key_last($array); // Retorna "b"
```

Referências:

PHP 8.1: `array_is_list()`

PHP 8.0: `str_contains()`, `str_starts_with()`, `str_ends_with()`

PHP 8.3: `array_take()`, `array_drop()`

4. Erro: "Creation of dynamic property ... is deprecated"

Antes (PHP 7.3):

A partir do **PHP 8.2**, a criação dinâmica de propriedades em classes que não declaram explicitamente essas propriedades se tornou **obsoleta** (*deprecated*).

Isso significa que, se um objeto tentar atribuir um valor a uma propriedade que não foi previamente definida na classe, o PHP gerará um aviso de depreciação.

```
class Vendarapida {  
    // Nenhuma propriedade $md5Hash declarada aqui  
}  
  
$obj = new Vendarapida();  
$obj->md5Hash = 'abc123'; // ⚠ ERRO: Criando propriedade dinamicamente!
```

Saída do log:

Destaque para as linhas 4 e 7.

```
[critical] -> require [/index.php:2]
[critical] --> require_once [/backend/index.php:191]
[critical] ---> vendarapida::__construct [/backend/system/core/CodeIgniter.php:309]
[critical] ----> errorBygeneral [/backend/application/controllers/pdv/vendarapida.php:132]
[critical] -----> dump [/backend/application/libraries/dump.php:76]
[critical] -----> __get_backtrace [/backend/application/libraries/dump.php:477]
[critical] [string] Creation of dynamic property vendarapida::$md5Hash is deprecated
```

Depois (PHP 8.1+):

Declare a propriedade explicitamente dentro da classe antes de usá-la:

```
class Vendarapida {
    public string $md5Hash; // ☐ Correto: propriedade definida na classe
}

$obj = new Vendarapida();
$obj->md5Hash = 'abc123'; // Agora funciona sem erro
```

Referências:

[RFC: Deprecate Dynamic Properties](#)

[PHP 8.2 - Dynamic Properties Deprecated](#)

5. Erro: "Return type of ... with IteratorAggregate::getIterator(): Traversable, or the

#[\ReturnTypeWillChange] attribute should be used to temporarily suppress the notice"

Antes (PHP 7.3):

A partir do PHP 8.1, métodos que implementam interfaces internas do PHP (como `IteratorAggregate::getIterator()`) devem declarar explicitamente seus tipos de retorno.

Caso contrário, o PHP gerará um aviso de depreciação

Destaque para as linhas 8 e 20.

```
[critical] -> require [/index.php:2]
[critical] --> require_once [/backend/index.php:191]
[critical] ---> MY_Controller::__construct [/backend/system/core/CodeIgniter.php:309]
[critical] ----> MY_Controller::apiAuth [/backend/application/core/MY_Controller.php:2]
[critical] -----> usuario::autenticar [/backend/application/core/MY_Controller.php:2]
[critical] -----> usuario_grupo_usuario::autenticar [/backend/application/controllers/usuario/usuario.php:1099]
[critical] -----> dispositivo::alterar [/backend/application/controllers/usuario/usuario_grupo_usuario.php:179]
[critical] -----> MY_Controller::salvar [/backend/application/controllers/usuario/dispositivo.php:110]
[critical] -----> Doctrine\ORM\EntityManager::flush [/backend/application/core/MY_Controller.php:2]
[critical] -----> Doctrine\ORM\UnitOfWork::commit
[/backend/vendor/doctrine/orm/lib/Doctrine/ORM/EntityManager.php:378]
[critical] -----> Doctrine\ORM\UnitOfWork::executeInserts
[/backend/vendor/doctrine/orm/lib/Doctrine/ORM/UnitOfWork.php:423]
[critical] -----> Doctrine\ORM\Persisters\Entity\BasicEntityPersister::executeInserts
[/backend/vendor/doctrine/orm/lib/Doctrine/ORM/UnitOfWork.php:1122]
[critical] -----> Doctrine\DBAL\Connection::prepare
[/backend/vendor/doctrine/orm/lib/Doctrine/ORM/Persisters/Entity/BasicEntityPersister.php:273]
[critical] -----> Composer\Autoload\ClassLoader::loadClass
[/backend/vendor/doctrine/dbal/lib/Doctrine/DBAL/Connection.php:1244]
[critical] -----> Composer\Autoload\includeFile [/backend/vendor/composer/ClassLoader.php:428]
[critical] -----> include [/backend/vendor/composer/ClassLoader.php:571]
[critical] -----> errorBygeneral [/backend/vendor/doctrine/dbal/lib/Doctrine/DBAL/Statement.php:27]
[critical] -----> dump [/backend/application/libraries/dump.php:76]
```

[critical] -----> __get_backtrace [/backend/application/libraries/dump.php:477]

[critical] [string] Return type of Doctrine\DBAL\Statement::getIterator() should either be compatible with IteratorAggregate::getIterator(): Traversable, or the #[\ReturnTypeWillChange] attribute should be used to temporarily suppress the notice

```
class MinhaClasse implements IteratorAggregate {  
    public function getIterator() { // ⚠ Erro: falta o tipo de retorno  
        return new ArrayIterator(["maça", "banana", "uva"]);  
    }  
}
```

Antes (PHP 8.1+):

Adicione o tipo de retorno `Traversable`, garantindo compatibilidade com a interface `IteratorAggregate`:

```
class MinhaClasse implements IteratorAggregate {  
    public function getIterator(): Traversable { // ✅ Correto: tipo de retorno declarado  
        return new ArrayIterator(["maça", "banana", "uva"]);  
    }  
}
```

Referências:

[RFC: Deprecate Implicit Incompatible Method Signatures](#)

[PHP 8.1 - Changes to IteratorAggregate::getIterator\(\)](#)

Documentação Oficial do PHP 8.x

PHP 8.0 Release Notes:

<https://www.php.net/releases/8.0/>

PHP 8.1 Release Notes:

<https://www.php.net/releases/8.1/>

PHP 8.2 Release Notes:

<https://www.php.net/releases/8.2/>

Agradecemos a colaboração de todos nesse processo de migração.

Caso encontrem outras inconsistências ou precisem de suporte, por favor, entrem em contato.

Tabela Propriedades

Salvar / Alterar Registro

```
$tabela = 'pessoa';  
$idTabela = 1;  
$propriedades = [  
    ['nome' => 'x',  
     ['valores' => [  
         ['junho' => false,  
          ['setembro' => date('i'),  
           ['dezembro' => date('H'),  
            []],  
          ['teste' => [  
              ['a' => RAND(0, 99), 'b' => RAND(0, 99)], RAND(0, 99), RAND(0, 99)  
              []  
          ]],  
        ],  
      ];
```

Retorno

```
{"success" => true, "message" => "Salvo com sucesso!", "data" => []}
```

Carregando / Utilizando o Registro

```
$tabela = 'pessoa';  
$idTabela = 1;  
  
$propriedades = $this->propriedade->carregar($tabela, $idTabela);
```

Retorno

```
Array
(  
  [nome] => x  
  [teste] => Array
```

```
(  
  [0] => Array  
    (  
      [a] => 56  
      [b] => 96  
    )  
  
  [1] => 22  
  [2] => 97  
)  
  
[valores] => Array  
(  
  [abril] => Array  
    (  
      [date] => 2024-09-13 15:19:58.605768  
      [timezone] => America/Sao_Paulo  
      [timezone_type] => 3  
    )  
  
  [junho] =>  
  [dezembro] => 16  
  [setembro] => 47  
)  
)
```