**Git-> autoload\_classmap\_files\_psr\_4\_v3**

**Inserir classes customizadas ao autoload do composer.**

1 – Adicionar o seguinte bloco de códogo ao autoload.php do vendor.

"autoload": {

"psr-4": {

(Passando o namespace) e o (sub namespace) depois a pasta ondem devem ficar as classes.

"Stywill\\BuscadorDeCursos\\": "src/"

}

}

2 – Agora é preciso atualizar o autoload.php para isso é necessário executar o comando **composer dumpautoload** no prompt dentro da pasta do projeto.

**Classmap e Files**

("classmap": ["src/TesteClassMap.php"]) Adiciona classes ao autoload

("files":["funcoes/functions.php"]) Adiciona arquivos ao autoload

Depois de cada alteração no composer.json é necessário executar na linha de comando o **composer dumpautoload.**

**Executar teste com phpunit**

**Baixar a instalar a classe na aula da alura.**

**Para usar o php phan**

O --dev instala apenas na maquina para teste e não adiciona em produção.

Executar o composer --dev require phan/phan .

Para que ele pegue todos os paramentos instalados é necessário criar uma pasta na raiz do projeto com o arquivo de configuração da seguinte forma “.phan/config.php” com o seguinte conteúdo.

return [

"target\_php\_version" => '7,1', (Aqui é necessário informar a versão do php)

'directory\_list' => [

'src',

(Aqui vai a lista de diretórios das classes usadas no projeto para que o phan analise junto )

'vendor/symfony/dom-crawler',

'vendor/guzzlehttp/guzzle',

'vendor/psr/http-message'

],

"exclude\_analysis\_directory\_list" => [

'vendor/'

],

'plugins' => [

'AlwaysReturnPlugin',

'UnreachableCodePlugin',

(Aqui vai alguns plugins já definidos)

'DollarDollarPlugin',

'DuplicateArrayKeyPlugin',

'PregRegexCheckerPlugin',

'PrintfCheckerPlugin',

],

];

E por fim executar o comando no prompt vendor/bin/phan --allow-polyfill-parser src\Buscador.php Esse é o nome do arquivo que quero verificar

**Comandos Scripts**

Assim é possível simplificar os comandos passados no prompt e até definir sequencias

"scripts":{

"test":"phpunit: tests\\TestBuscadorDeCursos.php",

"cs":"phpcs --standard=PSR12 src/",

"phan": "phan --allow-polyfill-parser src\\Buscador.php",

"check":[

"@phan",

"@cs",

"@test"

]

},

Quando usado o “composer list” temos o retorno de todos os comandos validos do composer

E com esse parâmetro abaixo é possível customizar a descrição dos nossos scripts de verificação e teste.

"scripts-descriptions":{

"check": "Roda as verificações do código. PHAN, PHPCS, PHPUNIT"

}

Um exemplo de como executar comando customizados antes ou depois dos comandos do composse.

No caso esse script é executado sempre depois do composer update

"post-update-cmd":["@teste"]

Para a lista completa de quais os eventos disponíveis, acesse <https://getcomposer.org/doc/articles/scripts.md>