

# CHECKLIST DE PREPARAÇÃO DE AMBIENTE – WINDOWS 10 (22H2)

## 1. Pré-requisitos do sistema

- ☐ Sistema operacional: **Windows 10 Pro 22H2** atualizado  
*Verifique em:* Configurações → Sistema → Sobre
- ☐ Acesso de **administrador** para instalação de softwares
  - **P@ssw0rd**
- ☐ Conexão com a internet estável
- ☐ Espaço livre mínimo: **10 GB**
- ☐ Desative temporariamente antivírus excessivamente restritivos (se interferirem na instalação)
- ☐ Drivers (Adicionais de convidado no VirtualBox)
- ☐ Navegadores (Chrome e Firefox)

## 2. Instalação de ferramentas base

### 2.1. Git

- ☐ Baixar e instalar [Git for Windows](#)
- ☐ Durante a instalação:
  - Escolha **“Use Git from the Windows Command Prompt”**
  - Escolha **“Checkout as-is, commit Unix-style line endings”**
- ☐ Após a instalação, verifique:  
`git --version`

### 2.2. PHP

- ☐ Baixar o Visual C++:
  - <https://learn.microsoft.com/pt-br/cpp/windows/latest-supported-vc-redist?view=msvc-170#latest-supported-redirectable-version>
- ☐ Baixar PHP (versão 8.4.14) via:
  - <https://windows.php.net/downloads/releases/php-8.4.14-Win32-vs17-x64.zip>
- ☐ Extrair o PHP para um diretório, por exemplo:  
C:\php
- ☐ Adicionar ao **PATH** do sistema:
  - Painel de Controle → Sistema → Configurações avançadas → Variáveis de ambiente
  - Em “Path”, adicionar: C:\php
- ☐ Testar:

```
php -v
```

- ☐ Verificar extensões essenciais no php.ini:

```
extension=fileinfo
extension=mbstring
extension=openssl
extension=pdo_mysql
extension=curl
extension=zip
```

### 3. Composer (Gerenciador de dependências PHP)

- ☐ Baixar o **Composer Setup** para Windows:  
<https://getcomposer.org/Composer-Setup.exe>
- ☐ Durante a instalação:  
Escolher o executável do PHP em C:\php\php.exe
- ☐ Testar:  

```
composer -V
```
- ☐ (Opcional) Atualizar globalmente:  

```
composer self-update
```

### 4. PHPUnit

- Instalar globalmente:  

```
composer global require phpunit/phpunit
```
- Adicionar o diretório global do Composer ao PATH:
  - Geralmente:  

```
%USERPROFILE%\AppData\Roaming\Composer\vendor\bin
```
- Verificar:  

```
phpunit --version
```

### 5. Node.js e npm

- Baixar **Node.js LTS (versão 18 ou superior)**:  
<https://nodejs.org/>
- Instalar com npm incluído
- Testar:
- ```
node -v
npm -v
```
- (Opcional) Atualizar npm:
- ```
npm install -g npm@latest
```

### 6. Laravel

- ☐ Instalar o **Laravel Installer** globalmente:

- ☐

```
composer global require laravel/installer
```

- ☐ Adicionar novamente o diretório global do Composer ao PATH:

```
%USERPROFILE%\AppData\Roaming\Composer\vendor\bin
```

- ☐ Testar:

```
laravel --version
```

- ☐ Criar um novo projeto para validar:

```
laravel new teste  
cd teste  
php artisan serve
```

- Teste acessando: <http://localhost:8000>

## 7. Ferramentas complementares úteis

- ☐ **Visual Studio Code** → <https://code.visualstudio.com/>

- Extensões recomendadas:

- PHP Intelephense
    - Laravel Blade Snippets
    - Laravel Artisan
    - Prettier
    - ESLint
    - **PHPunit Snippet (Winnie Lin)**

- ☐ Server WEB – MySQL - FTP

- **Visual C++**

- [https://wampserver.aviatechno.net/files/vcpackages/all\\_vc\\_redist\\_x86\\_x64.zip](https://wampserver.aviatechno.net/files/vcpackages/all_vc_redist_x86_x64.zip)

- **WAMPP**

- [https://sourceforge.net/projects/wampserver/files/WampServer%203.0.0/wampserver3.3.7\\_x64.exe/download](https://sourceforge.net/projects/wampserver/files/WampServer%203.0.0/wampserver3.3.7_x64.exe/download)

- **FileZilla – FTP Server**

- <https://filezilla-project.org/download.php?type=server>

## 8. TortoiseGit (Interface gráfica para Git)

- Instalar **TortoiseGit** -

- <https://download.tortoisegit.org/tgit/2.17.0.0/TortoiseGit-2.17.0.2-64bit.msi>

- Associar com instalação existente do Git

- (Opcional) Instalar **TortoiseGit Language Pack** para português

- Testar:

- Clique com o botão direito em uma pasta → verifique o menu “Git Clone...”
- Configure nome e e-mail do usuário no primeiro uso.

## 9. Validação final do ambiente

Execute estes testes no terminal (PowerShell ou CMD):

```
php -v
```

```
composer -V
```

```
phpunit --version
```

```
node -v
```

```
npm -v
```

```
laravel --version
```

Se todos retornarem suas versões corretamente → ambiente pronto para desenvolvimento!