

Windows – Prompt de Comando e Powershell

Prompt de Comando

1. Aula 1 – Os Primeiros Comandos No Prompt:

- echo: Printa na tela tudo o que for escrito a seguir

- >: indica onde o texto que foi escrito será salvo, caso queira. Colocar nome do arquivo e sua extensão para que seja criado.

```
PS A:\Users\bruno\Desktop\Dicas> echo O comando cd alterna entre pastas! > dica4.txt
```

- dir: Lista tudo o que está em uma pasta.

- /O: Uma opção de visualização do dir. Ele ordena a saída de acordo com o que for escolhido depois:

- D: Data;

- S: size.

Podemos ainda escolher qual pasta queremos que ele mostre o conteúdo, organizado ou não, apenas colocando o caminho dela:

```
C:\Users\bruno>dir /O:D c:\
O volume na unidade C não tem nome.
O Número de Série do Volume é 7E22-A2D0

Pasta de c:\

05/06/2021  09:10  <DIR>          PerfLogs
20/01/2022  16:23  <DIR>          Intel
20/01/2022  18:09  <DIR>          Riot Games
20/01/2022  21:27  <DIR>          Program Files
22/01/2022  15:19  <DIR>          Program Files (x86)
29/01/2022  13:08  <DIR>          Users
04/02/2022  15:45  <DIR>          Windows
              0 arquivo(s)              0 bytes
              7 pasta(s) 270.914.818.048 bytes disponíveis
```

- cd (change diretor) + nome da pasta: Entra na pasta desejada que foi listada através do dir.

- cd + ..: Volta para diretório anterior;

- cd + /: Volta para a raiz.

- mkdir + nome_pasta: Cria um diretório.

- move + nome do que vai ser movido + para onde será movido: Move itens, pastas e afins para outros lugares.
- Seta pra cima: Navega pelo histórico de comandos que já usamos nessa sessão.
- type: Lê o que tá dentro de arquivos txt.
- copy: Copia o item que quiser. Sempre lembrar de colocar a extensão que quer que a cópia tenha.
- rename: renomeia o item. Sempre colocar a extensão do arquivo.
- del + nome arquivo: deleta arquivo.
- Tab: completa os nomes dos comandos ou arquivos que tenhamos começado a digitar. Pode apertar várias vezes até aparecer o que está procurando.
- cls: limpa a tela do prompt.
- help ou help + nome do comando: Exibe uma lista com todos os comandos ou a descrição do comando escolhido.

Para saber mais:

1. Adicionando conteúdo a um arquivo:

Vimos neste capítulo do treinamento que podemos criar um arquivo utilizando o sinal >, mas e se quisermos adicionar mais linhas a um arquivo de texto já existente, por exemplo para guardar o resultado de diversas execuções de um programa em um único arquivo?

Para isto existe o >>, quando colocamos o sinal de maior, duas vezes, o Prompt entende que só deve criar um arquivo novo quando o arquivo que pedimos não existir! Caso ele já exista, ele adiciona o novo conteúdo ao final do arquivo, sem sobrescrevê-lo!

Veja só como funciona:

Digamos que primeiro criamos um novo arquivo de texto com o nosso conhecido comando `echo` :

```
echo Ola mundo > arquivo.txt
```

Ao abrirmos nosso `arquivo.txt` , vemos o texto que esperávamos:

```
type arquivo.txt
//arquivo.txt
Ola MundoCOPIAR CÓDIGO
```

Porém se agora desejarmos adicionar um novo texto abaixo de "Ola Mundo"? Utilizamos o '>>'!

Desta forma:

```
echo Novos dados! >> arquivo.txt
```

Quando abrimos nosso `arquivo.txt`, vemos que o texto foi adicionado corretamente:

```
type arquivo.txt
//arquivo.txt
Ola Mundo
Novos dados!
```

2. Pasta Home do usuário:

- a. Você já reparou que quando abrimos o prompt ele sempre abre na mesma pasta? Essa é a pasta do seu usuário, onde a maioria das pessoas salvam os Documentos, Fotos e Videos. Lá também ficam os arquivos particulares do seu usuário, como de configuração dos programas específicos e favoritos do seu navegador. A grande vantagem de cada usuário ter sua pasta é que temos uma separação dos dados do sistema e dos dados de cada usuário, facilitando assim o backup dos dados dos usuários caso seja necessário e aumentando a segurança também. A pasta do seu usuário é conhecida como pasta Home, e essa nomenclatura é adotada por muitos desenvolvedores quando querem se referir a pasta do usuário, independentemente do sistema operacional.

3. O comando tree:

- a. Dependendo do diretório atual podem aparecer muitas informações mas repare que o comando `tree` mostra as pastas e subpastas organizadas em uma árvore.

- b. O tree pode ser útil para entender a estrutura de um projeto. Muitas vezes você precisa baixar um projeto na Alura para importar alguns arquivos iniciais. Com o comando tree você já pode ver facilmente como o projeto está organizado! Muito útil :).

4. Comando more:

- a. O comando more funciona de um jeito semelhante ao comando type, com a diferença de exibir página por página do arquivo no terminal, em vez de mostra-lo todo de uma vez.

- b. O seu uso é análogo ao comando type, podendo ser chamado assim:

c. more arquivo.txt

- d. Ele exibirá uma página de cada vez do arquivo, sendo muito útil quando queremos exibir arquivos de texto com várias linhas para ir lendo lentamente, em vez de abri-lo todo no terminal de uma vez. Um exemplo disso é quando queremos ler os logs de uma aplicação que está em um servidor na nuvem, neste caso é preciso ler grandes arquivos de texto linha a linha, para identificar um bug ou realizar algum teste.

- e. Agora que você já conhece o comando more e um exemplo prático de sua aplicação, vou lhe mostrar como controlar a exibição das páginas:

- i. Para passar de página em página, utilize a tecla espaço do seu teclado.
- ii. Para passar de linha a linha, utilize a tecla enter do seu teclado.
- iii. Para sair da exibição do comando, sem chegar ao final do arquivo utilize a tecla q do seu teclado.

- f. Como você já sabe, sempre que quiser aprender mais sobre um comando, pode utilizar o help, então caso tenha a curiosidade de aprender ainda mais sobre o comando more, é só digitar help more no terminal!

O que aprendemos:

- para que serve o Prompt
- criar (mkdir) e remover (rmdir) diretórios
- copiar (copy) e apagar arquivos (del)
- imprimir no prompt com o comando echo

- saber a diferença entre . e ..
- criar arquivos com >
- ver o conteúdo do arquivo (type)
- mover pastas e arquivos (move)
- renomear diretórios e arquivos (rename)

2. Aula 2 – Um Novo Prompt e Executando Scripts:

2.1. Baixamos o cmdex para executar os códigos com maior facilidade que o cmd do windows.

2.2. Criamos nosso primeiro script de backup. Ele basicamente pega todos os arquivos e subpastas de uma pasta e copia com o xcopy /e /y para a pasta de backup.

2.3. Pause: Comando que espera o usuário fazer alguma interação com o terminal antes de continuar.

2.4. Scripts são feitos em documentos .bat e podem ser editados pelo bloco de notas.:

2.4.1. O batch script é um arquivo simples que agrupa uma série de comandos, normalmente utilizado para automatizar tarefas cotidianas. Um arquivo batch (.bat) é um script que é executado sequencialmente pelo prompt (um comando depois do outro).

2.4.2. No desenvolvimento usamos scripts de diversas formas. Por exemplo, para limpar uma máquina antes de instalar arquivos novos, guiar o usuário pela instalação, preparar arquivos de configuração e importar dados dentro da aplicação ou de um banco de dados. E é claro, o exemplo clássico de um script para automatizar o backup!

2.5. @echo off:

2.5.1. Você pode notar que é exibido duas vezes cada mensagem, uma vez exibindo o comando e outra na saída dele, pois quando executamos um script no prompt, ele exibe o nome do comando que está dentro do script e o seu resultado. Para desabilitar essa exibição dos comandos na hora de executar um script, devemos

começar o script com o comando `@echo off`, ficando desta maneira:

```
@echo off
```

```
cls
```

```
echo Dia de hoje:
```

```
echo %date%
```

```
echo Hora atual:
```

```
echo %time%
```

2.5.2. O comando `@echo off` faz com que os comandos que estão nos scripts não apareçam durante sua execução, mostrando apenas seus resultados.

2.5.3. É útil quando queremos deixar a execução dos scripts mais limpas, ou quando temos alguma informação sensível no script que não desejamos que usuário veja.

2.6. Para saber mais - `cmdr`:

2.6.1. Além do terminal colorido o `cmdr` traz algumas funcionalidades extras no terminal que ajudam muito o dia a dia do desenvolvedor, vamos dar uma olhada em algumas delas!

2.6.2. Copiar e Colar:

2.6.2.1. Se você já tentou copiar e colar no prompt do Windows pode ter reparado que é extremamente trabalhoso, precisando que usar o mouse para essa tarefa que é tão trivial. Uma das facilidades que o `cmdr` nos traz é que agora é possível colar no Prompt com um atalho do teclado, basta apertamos `CTRL + V`. Bem mais prático, não acha? (Caso este atalho não funcione com você, experimente utilizar o `CTRL + SHIFT + V`, pois dependendo da configuração do seu teclado o atalho pode mudar!)

2.6.2.2. Para copiarmos qualquer coisa que está no terminal, basta apenas selecionarmos ela com o mouse! Isso mesmo, ao

selecionar o conteúdo ele vai automaticamente para a área de transferência, podendo ser colado em qualquer lugar!

2.6.2.3.Outra opção é usar o atalho SHIFT+INSERT para colar o texto que estiver na área de transferência. (dica do aluno Wilton Santana pelo fórum).

2.6.3. Múltiplas abas:

2.6.3.1.Quando estamos trabalhando em um projeto, é muito comum precisarmos navegar entre várias pastas diferentes ou ter que executar múltiplos comandos simultaneamente. Quando utilizamos o prompt tradicional do Windows, muitos desenvolvedores acabam abrindo diversas instâncias do mesmo, para não ter que ficar trocando de diretório toda hora, mas isso acaba gerando uma confusão pois temos que usar o ALT + TAB repetidas vezes para trocar entre os prompts abertos e as aplicações que utilizamos para desenvolvimento.

2.6.3.2.No cmdr, nós temos uma solução para esse problema, as famosas abas! Inspirado nas abas que temos hoje nos navegadores, o cmdr implementou algo bem semelhante. Podemos ter várias instâncias do cmdr, lado a lado, organizada por abas. Para criar uma aba, basta utilizarmos o atalho de teclado CTRL + T e dar um ENTER para confirmar. Ele abre uma aba com um terminal novinho para nós! Isso ajuda muito na produtividade! Para alternar entre abas, é só usar o atalho CTRL + TAB que ele troca para a próxima aba, ficando muito fácil ter um terminal em cada pasta que necessita de atenção, um para consultar o banco de dados e uma para acessar um possível servidor na nuvem!

2.6.4. Configurações extras:

2.6.4.1.O cmdr é um terminal extremamente personalizável, se você usar o atalho Windows + ALT + P , você abrirá as configurações dele, indo na parte de Features, você verá que é possível alterar quase tudo do terminal. Lá tem

configuração das cores que ele utiliza, do nível de transparência, do tipo de cursor e muitas outras coisas que podem deixar o terminal do seu jeito.

2.6.4.2. Ele também tem como exportar e importar as configurações, então caso um dia você precise formatar ou trocar de computador, você pode levar as suas configurações com você.

2.7. O que aprendemos:

- 2.7.1. Instalar o cmd para usar no local do prompt padrão.
- 2.7.2. Como criar um script no Windows (.bat).
- 2.7.3. Como executar scripts .bat.
- 2.7.4. O comando xcopy, para copiar pastas e sub-pastas.
- 2.7.5. O comando pause, para esperar uma interação do usuário com o terminal.
- 2.7.6. A criar um script de backup, para automatizar uma tarefa repetitiva.

3. Aula 3 – Variáveis de ambiente e instalação do JDK:

3.1. Para fazer qualquer alteração no %PATH% da nossa máquina é importante sempre fazer um backup dele antes e guardar em um lugar seguro para que se algo der errado você tenha para onde correr.

3.1.1. Para criar um backup do PATH podemos simplesmente dar um echo %PATH% em um arquivo backupPATH.txt utilizando o cmd/er.

3.1.2. Não tente fazer pelo powershell, ele irá simplesmente criar o arquivo com a palavra %PATH% escrito dentro, ao invés de realmente fazer o backup dele.

3.2. Quando criamos um script e queremos que ele possa ser executado de qualquer lugar diretamente na linha de comando, sem precisar estar na pasta em que salvamos o nosso script, precisamos adicionar ele nesse PATH.

3.2.1. Ao analisar o nosso PATH com o 'echo PATH' ou somente 'PATH' na linha de comando podemos notar que a maioria dos caminhos existentes lá possuem uma pasta bin, então podemos

criar a nossa própria pasta em qualquer lugar e jogar todos os nossos scripts dentro dela, adicionando o caminho dela no PATH.

3.2.2. Para setar um novo caminho ou script no PATH utilizamos o 'set PATH=%PAHT%;c:\caminho\para\a\nova\pasta\bin'. NÃO ESQUECER DE SEMPRE COLOCAR O %PATH% DEPOIS DO 'PATH=' E O ';' POIS QUEREMOS ADICIONAR ESSA NOVA PASTA BIN E NÃO SUBSTITUIR TODO O NOSSO PATH POR ELA!!!!!!.

3.3. O set também serve para criar variáveis de ambiente, ou seja, indicamos caminhos que queremos salvar com um nome específico e assim ao invés de digitar aquele caminho toda vez que quisermos ir para ele, podemos simplesmente evocar com o nome que demos para a variável de ambiente.

3.3.1. Como criar a variável de ambiente:

```
set PASTA_CODIGO=a:\Users\bruno\Documents\GitHub\Alura\Windows\codes
```

3.3.2. Para ver todas as variáveis de ambiente podemos simplesmente usar o set sem nenhum outro argumento na linha de comando. Ele irá devolver todas as variáveis, seus caminhos e afins. Para conferir onde está sua variável recém criada basta procurar pelo nome, uma vez que está em ordem alfabética.

3.3.3. Para evocar uma variável fazemos como o %PATH%:

```
a:\Users\bruno>echo %pasta_codigo%  
a:\Users\bruno\Documents\GitHub\Alura\Windows\codes
```

3.4. O comando set infelizmente só funciona temporariamente, ou seja, durante todo o tempo que o cmd estiver aberto. A partir do momento que fechamos e reabrimos, todas as variáveis de ambiente e o nosso novo caminho adicionado no PATH somem.

3.4.1. Para que uma dessas alterações seja feita permanentemente, utilizamos o comando setx.

3.4.2. Ele não funciona no cmd, somente no cmd oficial do windows, pois uma alteração permanente no PATH é algo muito importante e perigoso de se fazer, então tem que ser oficialmente e em modo de administrados.

- 3.4.3. Ele funciona parecido com o set, mas ao invés de colocar '=' utilizamos aspas duplas:

```
C:\Users\bruno>setx PATH "%PATH%;a:\Users\bruno\Documents\bin" /M
ÊXITO: o valor especificado foi salvo.
```

- 3.4.4. O /M no fim indica que é para a alteração ser feita no sistema.
- 3.4.5. Quando for criar uma variável não precisa a contra barra pós última pasta, caso sua variável seja somente para levar até uma pasta.
- 3.4.6. Após setar essas alterações permanentemente, reinicie o cmd para que elas comecem a valer.

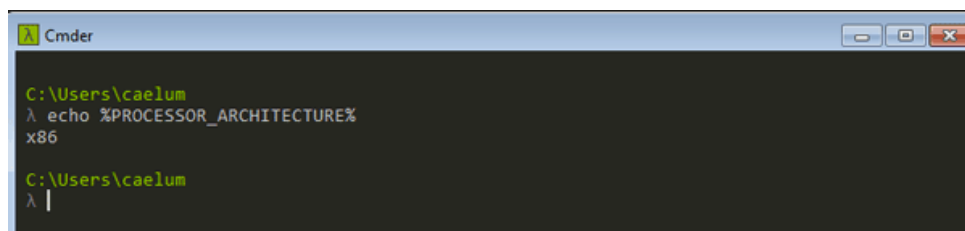
3.5. Alterando o PATH pela interface:

- 3.5.1. Clique no ícone Pesquisar e digite *Painel de Controle*
- 3.5.2. Clique em -> Painel de Controle -> Sistema de Segurança -> Sistema-> Configurações Avançadas
- 3.5.3. Na aba Avançado das *Propriedades do sistema* clique em *Variáveis de Ambiente*,
- 3.5.4. Em *Variáveis do Sistema* localize PATH e clique nele.
- 3.5.5. Na janela *Editar* você pode modificar o PATH adicionando a localização da classe para o valor de PATH.
- 3.5.6. Para ver as alterações feitas no PATH é preciso reabrir o mesmo (não precisa reiniciar).

3.6. Arquitetura do processador:

- 3.6.1. Podemos descobrir a arquitetura do computador usando 2 métodos diferentes:

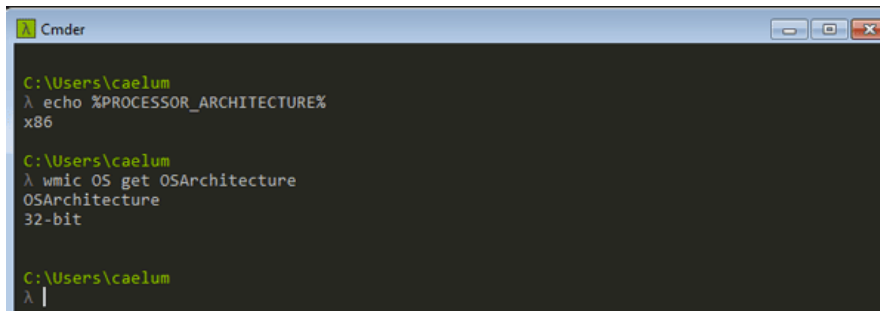
3.6.1.1. echo %PROCESSOR_ARCHITECTURE%



```
C:\Users\caelum
λ echo %PROCESSOR_ARCHITECTURE%
x86

C:\Users\caelum
λ |
```

3.6.1.2. wmic OS get OSArchitecture



```
Cmdr
C:\Users\caelum
λ echo %PROCESSOR_ARCHITECTURE%
x86
C:\Users\caelum
λ wmic OS get OSArchitecture
OSArchitecture
32-bit
C:\Users\caelum
λ
```

3.6.2. wmic é um comando que dá acesso ao Windows Management Instrumentation. O wmic é mais importante para administradores de infraestrutura e possui muito mais funções. Para gente basta saber se é 32bit ou 64bit!

3.7. Arquitetura pela interface gráfica:

3.7.1. Acesse o *Painel de Controle* (ou *Configurações* no Windows 10) e clique no *Sistema* para *Ver o nome do computador*. Depois clique no *Sobre*. Nessa tela também encontramos a informação sobre x86 (32bit) ou x64 (64bit).

3.7.2. Dica: Se seu teclado tiver a tecla *Pause*, pode usar o atalho *Windows + Pause*.

3.7.3. Isso é um bom exemplo que mostra o poder do prompt. Dá muito mais trabalho navegar entre as janelas do que digitar um comando no prompt. Outra vantagem do prompt é que esses comandos mudam com muito menos frequência do que as janelas do Windows.

3.7.4. A grande desvantagem é que não podemos exigir do um usuário comum executar comandos no prompt. Mas a gente, sendo ou se tornando desenvolvedor, pode encarar a linha de comando!

3.8. Para saber mais – Uma ferramenta para as variáveis de ambiente:

3.8.1. RapidEE: programa que permite alterar as variáveis de ambiente de maneira gráfica muito mais fácil.

3.8.2. Ele exibe tanto as variáveis de ambiente do sistema quanto as de usuário.

3.8.3. Link para download: <http://www.rapidee.com/en/download>.

3.8.4. Para adicionar uma nova variável, basta clicar com o botão direito e selecionar *Add Variable*. Irá aparecer uma caixinha com o nome

da variável que você deseja criar, e a opção de seleccionar o conteúdo como uma *String normal* ou *Expanded String*. A diferença é que devemos usar *Expanded String* quando queremos usar uma variável que referencie outra, por exemplo, caso queiramos criar uma variável chamada **PASTA_TEMP**, que tenha o valor **%TEMP%**, referenciado o valor da variável TEMP.

- 3.8.5. Para editar o valor de alguma variável, basta expandir seu conteúdo clicando no símbolo de + ao lado do nome dela, e dar um duplo clique no valor que desejamos alterar.
- 3.8.6. Para adicionar novos valores, clicamos com o botão direito na variável que queremos alterar e seleccionamos a opção *Add value....* Podemos escrever o caminho na mão, como fazíamos no prompt, ou seleccionar entre as opções *Insert directory path...* e *Insert file path...*, para inserir uma pasta ou um arquivo, respectivamente, através do sistema de janelas do Windows.
- 3.8.7. Quando estamos alterando o PATH, podemos cometer algum erro, como inserir duas vezes a mesma variável ou inserir uma variável em branco, e isto é um pouco difícil de detectar quando estamos na linha de comando, pois se o PATH for muito grande, pode ficar difícil de notar algum descuido nosso.
- 3.8.8. O RapidEE tem uma função que corrige isto automaticamente para gente, é a *Cleanup Paths*, que remove valores repetidos e valores em branco. Para usá-la basta clicar com o botão direito em cima de uma variável e seleccioná-la.
- 3.8.9. Outro erro bastante comum quando estamos instalando programas ou alterando a variável PATH é que acabamos criando certos caminhos inválidos. O rapidEE detecta isto para a gente e coloca uns caminhos em vermelho. Ficando muito fácil de identificá-lo, agora basta seleccionarmos o caminho inválido e apagá-lo com a tecla delete do seu teclado.
- 3.8.10. Uma coisa importante ao utilizar a ferramenta rapidEE é lembrar de salvar as alterações antes de fechá-la, clicando no pequeno ícone de salvar da barra de ferramentas dele.

3.8.11. Ele dá um pequeno aviso caso você tente fechar sem salvar, mas é bom te informar com antecedência para você não se esquecer!

3.9. O que aprendemos:

- 3.9.1. O que são as variáveis de ambiente;
- 3.9.2. Mostrar as variáveis de ambiente pelo comando set;
- 3.9.3. A importância da variável PATH;
- 3.9.4. Alterar a variável PATH com set e setx;
- 3.9.5. Criar novas variáveis de ambiente;
- 3.9.6. Descobrir a arquitetura do sistema (32bit ou 64bit);
- 3.9.7. Instalar o JDK do Java;
- 3.9.8. Configurar o PATH para usar o comando javac no prompt;
- 3.9.9. Configurar a JAVA_HOME.

4. Aula 4 – Gerenciando Pacotes Com Chocolatey:

4.1. O chocolatey é um gerenciador de pacotes como o apt do Linux que facilita nossa vida durante a instalação de programas e pacotes em si no geral.

4.2. Site para instalar o chocolatey: <https://chocolatey.org/install>.

4.2.1. **Get-ExecutionPolicy**: Rodar no powershell, se o resultado for: Restricted, então rode: *Set-ExecutionPolicy AllSigned* ou *Set-ExecutionPolicy Bypass -Scope Process*. O resultado do get deve ser *AllSigned*.

4.2.2. Quando o resultado for o certo, rode esse comando para instalar o chocolatey:

4.2.3. `Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))`

4.2.4. Lembrando que tudo deve ser feito no powershell rodando como administrador.

4.3. Para começar a usar basta digitar `choco -command-`.

4.3.1. Para instalar um pacote `choco install nome-do-pacote`.

4.3.2. Para saber quais pacotes existem e como instalar eles:
<https://community.chocolatey.org/packages>.

4.3.3. Passar o -y depois do comando de instalar faz com que ele aceite todas as perguntas que possam surgir direto.

4.4. Choco list -l: lista todos os pacotes instalados na máquina.

4.5. Choco uninstall nome-do-pacote: desinstala um pacote na mesma facilidade que instalou.

4.6. Existem inúmeros pacotes sobre tudo e grande parte dos programas, não somente as ferramentas de desenvolvimento.

4.7. Para saber mais – choco list:

4.7.1. Vimos no treinamento que podemos verificar os pacotes disponíveis baixados através do site do chocolatey, porém também é possível fazer isto pela linha de comando! O comando choco list lista todos os pacotes que podem ser instalados pelo chocolatey mas antes de executa-lo tome cuidado por quê o chocolatey possui milhares de pacotes e seu computador vai passar um bom tempo listando-os! Para executar uma busca por pacotes mais específica, podemos utilizar o comando choco list com um nome do pacote que queremos buscar, por exemplo:

4.7.1.1.choco list git

4.7.2. Isto nos exibirá diversos pacotes que possuem git no nome ou em suas dependências! Bem mais fácil do que procurar o pacote desejado em uma lista gigantesca.

4.7.3. Podemos encontrar pessoas que usem o choco Search, mas ele faz exatamente a mesma coisa que o list. Inclusive podemos utilizar os mesmos argumentos em ambos.

4.8. Para saber mais – Gerenciador OneGet:

4.8.1. Gerenciador de pacotes nativo do windows que vem a partir do win10.

4.8.2. Página do projeto para mais informações:
<https://github.com/OneGet/oneget/wiki/cmdlets>.

4.9. Para saber mais – Outros gerenciadores:

4.9.1. Assim como existe o chocolatey para Windows, no mundo Linux nós temos o apt-get. O apt-get também é um gerenciador de

pacotes, e inclusive foi ele que inspirou a criação do chocolatey. Ele já vem por padrão nas distribuições mais comuns do Linux, como o Ubuntu e o Mint e do mesmo modo do chocolatey, facilita muito nossa vida quando queremos instalar programas no Linux.

4.9.2. Já quando desenvolvemos aplicações, por exemplo usando Java, é muito comum usarmos código (dependências) de outros desenvolvedores no nosso código para ajudar em alguma tarefa. No mundo Java possuímos algumas ferramentas que nos ajudam a baixar e a lidar com essas dependências como o maven, ivy e o gradle. Elas agilizam o processo de desenvolvimento, já que, com um pequeno código, qualquer desenvolvedor que baixar nossa aplicação pode baixar as dependências do mesmo e ter o projeto funcionando rapidamente.

4.9.3. Links para os cursos de maven e ivy:

4.9.3.1. <https://cursos.alura.com.br/course/maven-build-do-zero-a-web>.

4.9.3.2. <https://cursos.alura.com.br/course/ivy>.

4.10. Para instalar uma versão específica de um programa/pacote: `choco install nome-do-pacote -version x.x.x`.

4.11. O que aprendemos:

4.11.1. O que é um gerenciador de pacotes.

4.11.2. Instalar o chocolatey, o gerenciador de pacotes para o Windows.

4.11.3. Como instalar programas e ferramentas com apenas um comando.

4.11.4. Especificar a versão correta dos programas.

4.11.5. Remover pacotes com o chocolatey.

4.11.6. Como instalar a JDK, python, Node, Ruby, Git, PHP e muitos outros.

5. Aula 5 – Comandos do Linux Bash no Windows:

5.1. O cmdr possui 2 versões, a mini e a completa. Na versão completa ele vem com todas as funcionalidades de comandos do Linux tais como:

5.1.1. Cp: para copiar arquivos;

5.1.2. Ls: listagem de diretório (como o dir);

- 5.1.2.1. -a: mostra os arquivos ocultos desse diretório;
- 5.1.2.2. -al: mostra os arquivos ocultos e ainda lista como no dir;
- 5.1.2.3. -lt: ordena pela data de modificação;
- 5.1.2.4. -lth: mostra o tamanho dos arquivos;
- 5.1.2.5. -lSh: ordena por tamanho de arquivo;
- 5.1.2.6. -hl: Transforma unidades de tamanho = 4096 -> 4K;
- 5.1.2.7. -rl: lista em ordem reversa a normal;
- 5.1.2.8. -sl: lista por ordem de tamanho.

5.1.3. Mv: move arquivos e pastas.

5.1.4. Rm: deleta arquivos e pastas como o del.

5.1.5. Cat: exibe o que tem escrito em um arquivo.

5.2. Grep -conteúdo-: Comando que busca o conteúdo especificado em um texto passado para ele. Se quiser encontrar todos as pastas e arquivos que possuam doc no nome no diretório home, você pode fazer um `ls -al | grep Doc` e ele devolverá esse resultado:

```
a:\Users\bruno
λ ls -al | grep Doc
drwxr-xr-x 1 bruno 197609 0 fev 23 19:01 Documents/
```

5.2.1. O '|' serve justamente para conectar a saída de um comando com a entrada de outro, ou seja, no nosso código acima pedimos para o grep procurar a palavra doc em todo o resultado da listagem de todos os arquivos, inclusive ocultos, da pasta home que era onde estávamos.

5.2.2. Podemos combinar o grep com qualquer coisa, inclusive o cat para procurar palavras em documentos.

5.3. O que aprendemos:

- 5.3.1. Comandos do Linux bash no Windows;
- 5.3.2. Como habilitar os comandos do Linux bash no Windows;
- 5.3.3. Como instalar a versão completa do Cmder;
- 5.3.4. Os comandos ls, rm, rmdir, mv, cp, cat;
- 5.3.5. As diferenças entre os comandos de Windows e de Linux;
- 5.3.6. Buscar dados com o comando grep.