Módulo 8: Backend com C# e Python

Dia 03: Controle de Fluxo e Debugging Profissional

João Quentino

29 de dezembro de 2025

Roteiro do Dia 03

Objetivo da Aula

Ao final desta aula, você será capaz de:

- Usar `for`, `foreach` e `while` para repetir ações.
- Criar funções para organizar melhor o código.
- Tratar erros com `try/except` (Python) e `try/catch` (C#).
- Usar o debugger do VS Code para inspecionar o código passo a passo.
- Automatizar a leitura e processamento simples de arquivos CSV.

Aquecimento: Por que loops?

Problema

Ler 100 linhas de um arquivo e mostrar apenas as que interessam. Fazer isso copiando e colando código 100 vezes não faz sentido.

Ideia-chave:

- Loops = repetir uma ação para cada item de uma coleção.
- Funções = dar nome para um pedaço de lógica.

Loop `for` em Python

Percorrendo uma lista

```
nomes = ["Ana", "Bruno", "Carlos"]

for nome in nomes:
    print(nome)
```

Leitura: "para cada `nome` dentro de `nomes`, execute o bloco".

Loop foreach em C#

Percorrendo uma lista

```
var nomes = new List<string> { "Ana", "Bruno", "Carlos
    " };

foreach (var nome in nomes)
{
    Console.WriteLine(nome);
}
```

Loop `while` em Python

Repetindo até a condição parar

```
contador = 0

while contador < 3:
    print(contador)
    contador = contador + 1
```

Loop while em C#

Repetindo até a condição parar

```
int contador = 0;

while (contador < 3)
{
    Console.WriteLine(contador);
    contador = contador + 1;
}
```

Funções em Python

Separando responsabilidades

```
def somar(a, b):
    resultado = a + b
    return resultado

x = somar(10, 5)
print(x)
```

Ideia: uma função recebe entradas, faz algo e devolve um resultado.

Métodos em C#

Funções dentro de uma classe

```
static int Somar(int a, int b)
{
    int resultado = a + b;
    return resultado;
}

int x = Somar(10, 5);
Console.WriteLine(x);
```

Python: try/except

Evitando que o programa quebre

```
texto = input("Digite um numero: ")

try:
    numero = int(texto)
    print("Numero valido:", numero)
except ValueError:
    print("Valor invalido, nao é inteiro.")
```

C#: try/catch

Mesmo conceito em C#

```
Console.Write("Digite um numero: ");
var texto = Console.ReadLine();

try
{
    int numero = int.Parse(texto ?? "0");
    Console.WriteLine("Numero valido: " + numero);
}
catch (FormatException)
{
    Console.WriteLine("Valor invalido, nao e inteiro."
        );
}
```

Debugging: Conceitos Rápidos

- **Breakpoint**: ponto onde a execução para.
- **Step Over**: executa a próxima linha.
- **Watch/Locals**: ver o valor das variáveis.
- **Call Stack**: ver quem chamou quem.

Objetivo em aula:

- Rodar o código de leitura de arquivo em modo debug.
- Enxergar o conteúdo de cada linha do CSV.

Desafio Guiado: Leitura de CSV

Contexto

Temos um arquivo `vendas.csv` com colunas: `data`, `produto`, `quantidade`, `preco`.

Tarefas:

- Ler o arquivo linha por linha.
- Ignorar o cabeçalho.
- Somar o valor total de vendas.

Python (Visão Geral)

Leitura simples de CSV

```
with open("vendas.csv", encoding="utf-8") as arquivo:
    for linha in arquivo:
        print(linha.strip())
```

No gabarito, essa lógica vira funções bem organizadas.

C# (Visão Geral)

Leitura simples de CSV

```
var linhas = File.ReadAllLines("vendas.csv");

foreach (var linha in linhas)
{
    Console.WriteLine(linha);
}
```

Resumo do Dia 03

- Loops: `for`, `foreach`, `while`.
- Modularização com funções/métodos.
- Tratamento básico de erros em Python e C#.
- Noções práticas de debugging no VS Code.
- Automação de leitura de arquivos CSV.

Próximos Passos

Bora automatizar!

- Completar o desafio de somar o total de vendas por produto.
- Criar funções novas para filtrar vendas por data.
- Preparar a base para, depois, expor esses dados em uma API.