

Tabela de cores

Transcrição

Nesse [link \(https://github.com/alura-cursos/Curso-VBA-Automatizando-tarefas-no-Excel/archive/aula5.zip\)](https://github.com/alura-cursos/Curso-VBA-Automatizando-tarefas-no-Excel/archive/aula5.zip) você vai encontrar a planilha usada nessa aula.

Conforme prometido, montaremos a *tabela de cores da propriedade ColorIndex*. A ideia não é só montar a tabela de cores, mas também realizar um *overview* dos três laços que estudamos em cima dessa tabela.

Com o "Alt + F11" conseguimos acessar o ambiente de desenvolvimento do VBA, e no "Project Explorer" tínhamos o módulo de **laços**. E agora vamos adicionar um novo módulo chamado de **Projeto_Cores**, onde vamos criar a macro que será uma *overview* dos assuntos discutidos.

A princípio, utilizaremos o `While`. Então a macro será chamada de `cores_While()`.

Foi montado uma tabela de exemplo para nós com o *valor* e a *cor* correspondente nas colunas `A` e `B`. Nós montaremos a nossa tabela nas colunas `E` e `F`.

Então, a partir da coluna `E`, iremos iniciar:

```
Sub cores_While()  
    Range("e2").Select  
End Sub
```

Sabemos que é necessário indicar o número das cores de `0` até `56`, e por isso, guardaremos esses números dentro de uma variável chamada `cor` do tipo *Byte*.

```
Sub cores_While()  
    Dim cor As Byte  
    Range("e2").Select  
End Sub
```

Mas por que uma variável do tipo Byte?

O *Byte* aceita até o valor `256`, e por isso ele consegue ser o suficiente e nos atender muito bem, já que temos somente `56` cores.

Quando criamos uma variável numérica, é padrão que ela já seja iniciada com `0`(zero). Para ir do `0` até o `56`, utilizaremos o `While`.

```
Sub cores_While()  
    Dim cor As Byte  
    Range("e2").Select  
    Do While cor <= 56
```

```
Loop  
End Sub
```

Com esse laço, iremos executar algo que será descrever tanto o *valor* da cor quanto a *cor* específica na célula do Excel. Em E2, será escrito o **valor** da variável `cor`, ou seja, `ActiveCell` recebe o valor da variável `cor`. Já a célula F2 será pintada com a cor em questão, ou seja, `ActiveCell.Offset(0, 1).Interior.ColorIndex` recebe o conteúdo da variável `cor`.

```
Sub cores_While()  
    Dim cor As Byte  
    Range("e2").Select  
    Do While cor <= 56  
        ActiveCell = cor  
        ActiveCell.Offset(0, 1).Interior.ColorIndex = cor  
    Loop  
End Sub
```

Salvaremos a nossa nova macro, e em seguida vamos testá-la.

Repare que a macro irá parar em E2, e depois irá validar a célula ativa dentro do `While`, que começará com 0(zero). Zero é *menor ou igual* a 56? Sim! Então a célula ativa recebe o número 0, e a célula da direita será pintada com a cor correspondente a zero. Ao voltarmos ao `while`, qual é o valor da variável `cor`? É zero. Então o valor zero será colocado na célula ativa, e a célula da direita será pintada com a cor correspondente, e assim por diante.

Todavia, o *loop* continua a ser executado sem nenhuma condição de parada, pois o conteúdo de `cor` sempre vai ser zero e nunca vai mudar. Se executarmos uma macro com um *loop* infinito, a nossa máquina irá travar, e assim teremos que usar o comando "Ctrl + Pause" para interromper o *loop*. E as vezes, será preciso fechar o Excel. Por isso é sempre importante salvar as alterações antes de executar as macros.

Por conta disso, percebemos que o valor de `cor` precisa ser alterado, precisa ser incrementado, e além disso, precisamos descer uma célula, caso contrário estaremos sempre pintando em cima do anterior.

```
Sub cores_While()  
    Dim cor As Byte  
    Range("e2").Select  
    Do While cor <= 56  
        ActiveCell = cor  
        ActiveCell.Offset(0, 1).Interior.ColorIndex = cor  
        cor = cor + 1  
        ActiveCell.Offset(1, 0).Select  
    Loop  
End Sub
```

Vamos salvar antes de testar.

Legal! A macro está funcionando assim como planejávamos.