Tutorial 1 - Básico do VBA do Excel

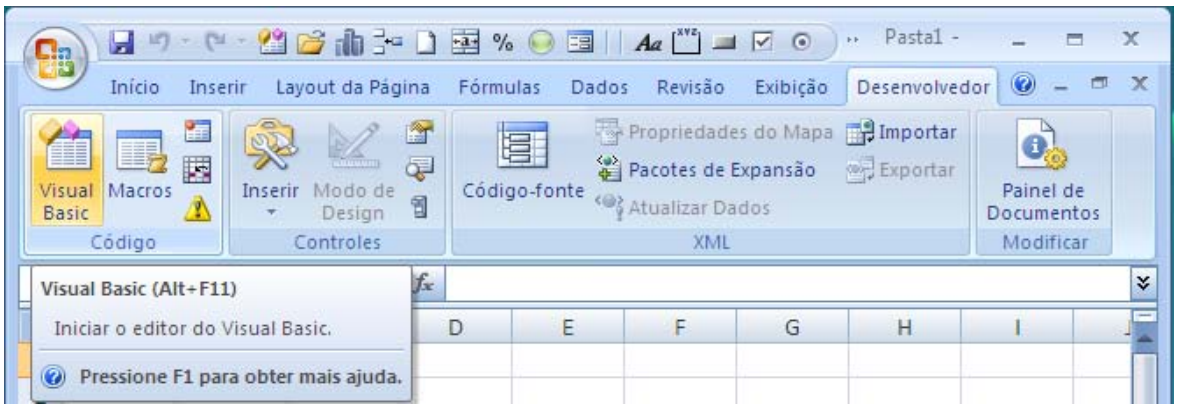
Este tutorial contém a 1ª lição sobre a série Básico do VBA do Excel. Ele cobre tópicos de criação e gerenciamento de matrizes e o entendimento de estruturas de decisão e laço do VBA. Os iniciantes na programação de VBA serão encorajados a percorrerem de cabo a rabo as lições anteriores desta série se eles já não tiverem feito isto. Este documento contém informações sobre os seguintes tópicos.

O site de Suporte da Microsoft ou a seção de Ajuda (Help) do VBA Excel no seu computador contém exemplos compreensíveis sobre a maioria das coisas cobertas neste tutorial. Para mais informação, por favor, refira-se a eles.

Criando Sua Primeira Macro Microsoft Support

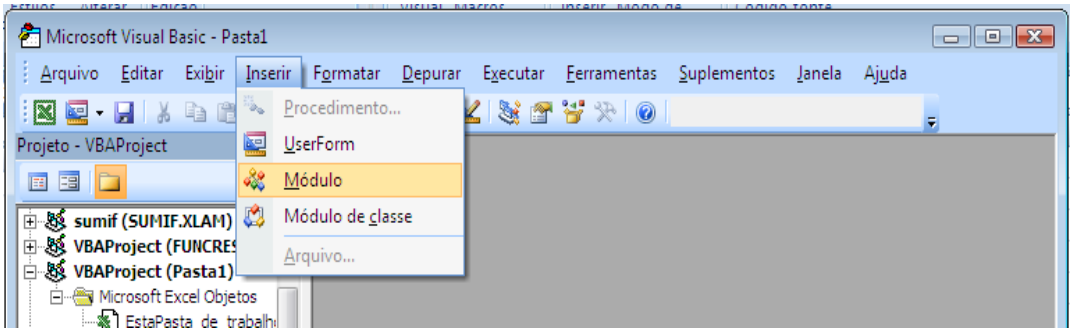
Nesta subseção lhe mostraremos como criar sua primeira macro (programa VBA). Nós usaremos o clássico exemplo mundial "Alô Mundo!". Para criar o exemplo, por favor, siga os seguintes passos:

1. Abra o Visual Basic Editor indo à guia Desenvolvedor, no grupo Código e clique no botão Visual Basic ou apenas pressione as teclas [Alt] e [F11] ao mesmo tempo.



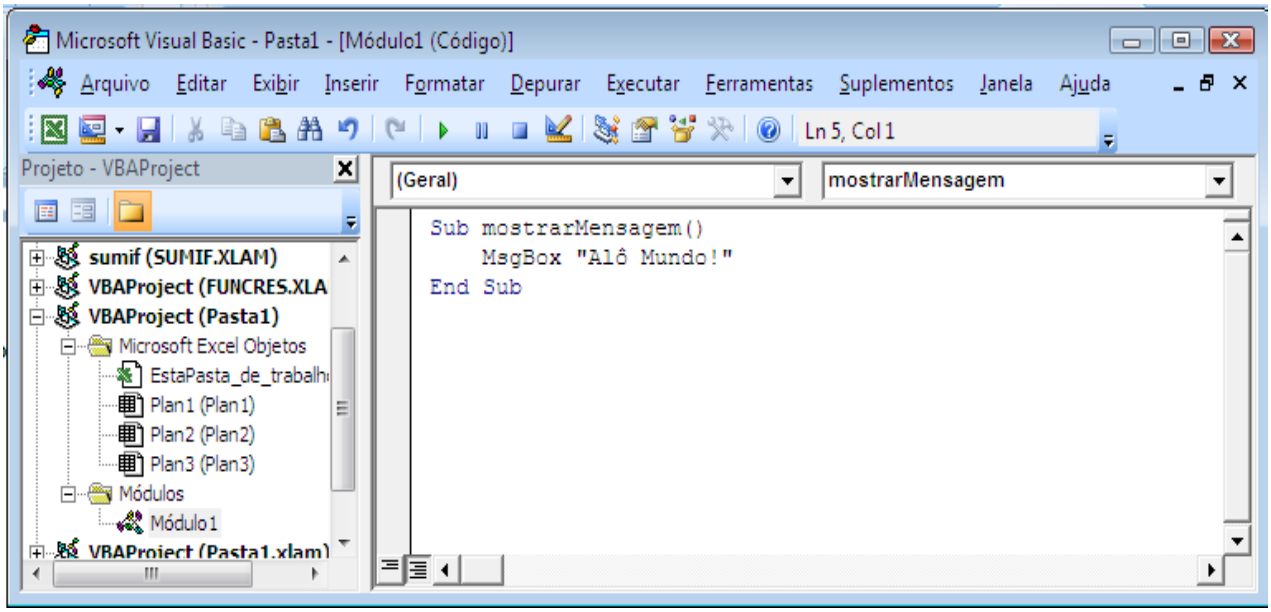
2. No menu **Inserir** no topo do Visual Basic Editor, selecione Módulo para abrir a **janela Módulo** (janela

de código).

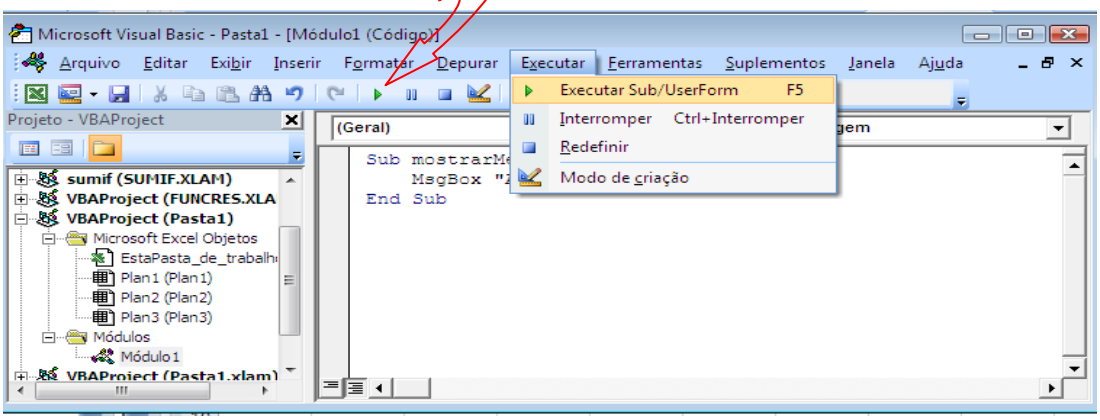


3. Na janela Módulo, digite o seguinte:

```
Sub mostrarMensagem()
    MsgBox "Alô Mundo!"
End Sub
```



4. Clique o botão **Executar Sub/UserForm**, , ou pressione [F5], ou vá para **Executar...Executar Sub/UserForm** para executar o programa



5. A caixa de mensagem aparece com a saudação "Alô Mundo!".

Este é o seu primeiro programa VBA.

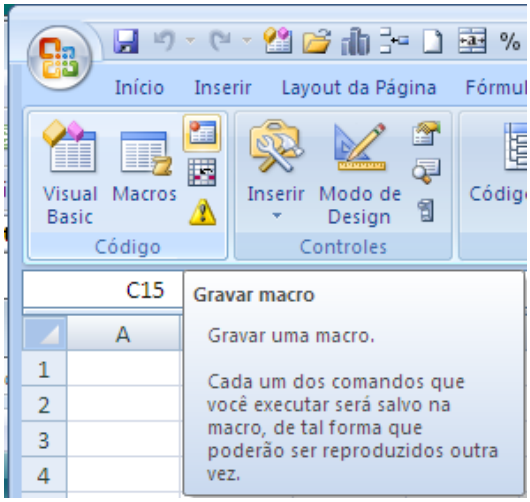
Gravando a Sua Primeira Macro

Gravando uma Macro

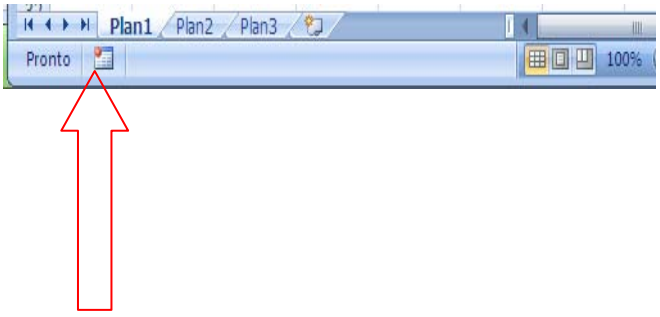
O Microsoft Excel tem um gravador de macro embutido que traduz as suas ações em comandos de macro do VBA. Depois de você ter gravado a macro, você será capaz de ver o layout e a sintaxe. Antes de você gravar ou escrever uma macro, planeje os passos e comandos que você quer que a macro realize. Cada ação que você fizer durante a gravação da macro será gravada – incluindo a correção que você fez.

Neste exemplo, gravaremos uma macro que configura a cor de fundo da célula para amarelo claro. Para gravar a macro, siga os passos abaixo:

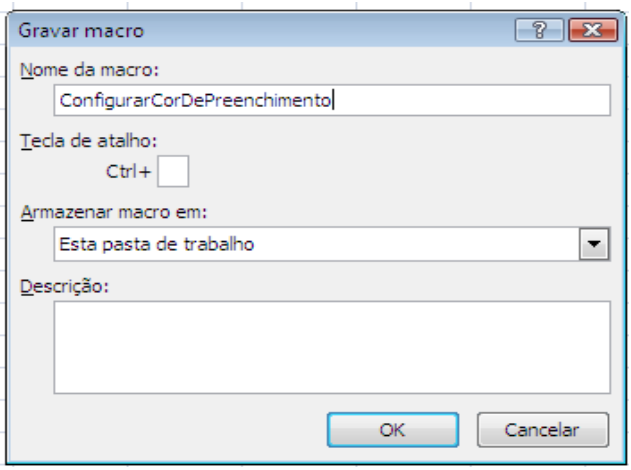
1. Selecione o botão **Gravar macro**  no grupo **Código**, da guia **Desenvolvedor**.



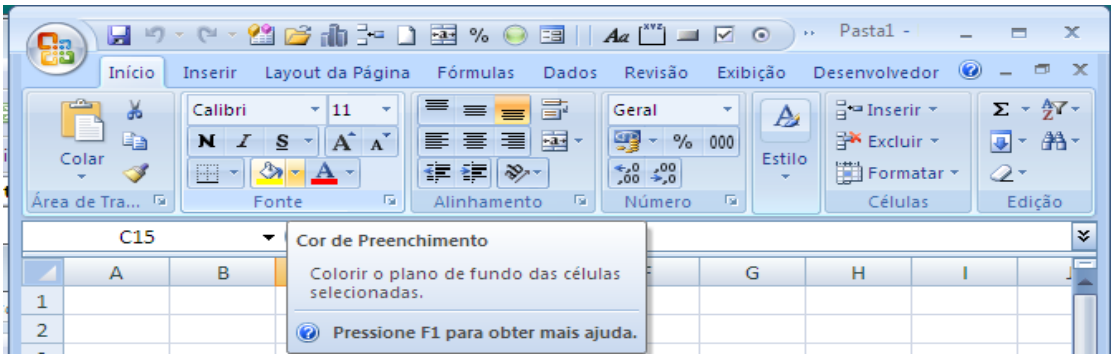
Ou vá ao final da planilha e clique no botão

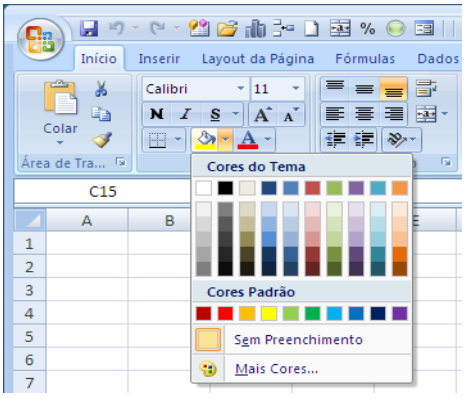


2. Na caixa de diálogo *Gravar macro*, digite “ConfigurarCorDePreenchimento" na caixa de texto **Nome da macro** para designar o nome da macro. Deixe todas as outras opções como default e daí então clique no botão Ok. Isto começará a gravação da macro.

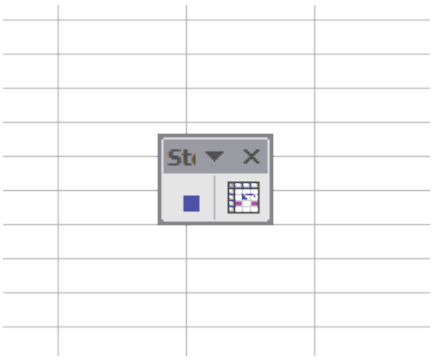


3. No Painel de Cor de Fundo (Background), selecione a caixa de cor Amarelo Claro. Esta ação configurará o fundo da célula atual (A1) na cor amarelo claro.





4. Para parar a gravação da macro, clique no botão Parar (o retângulo azul marinho) na barra de ferramentas Gravador de Macro.

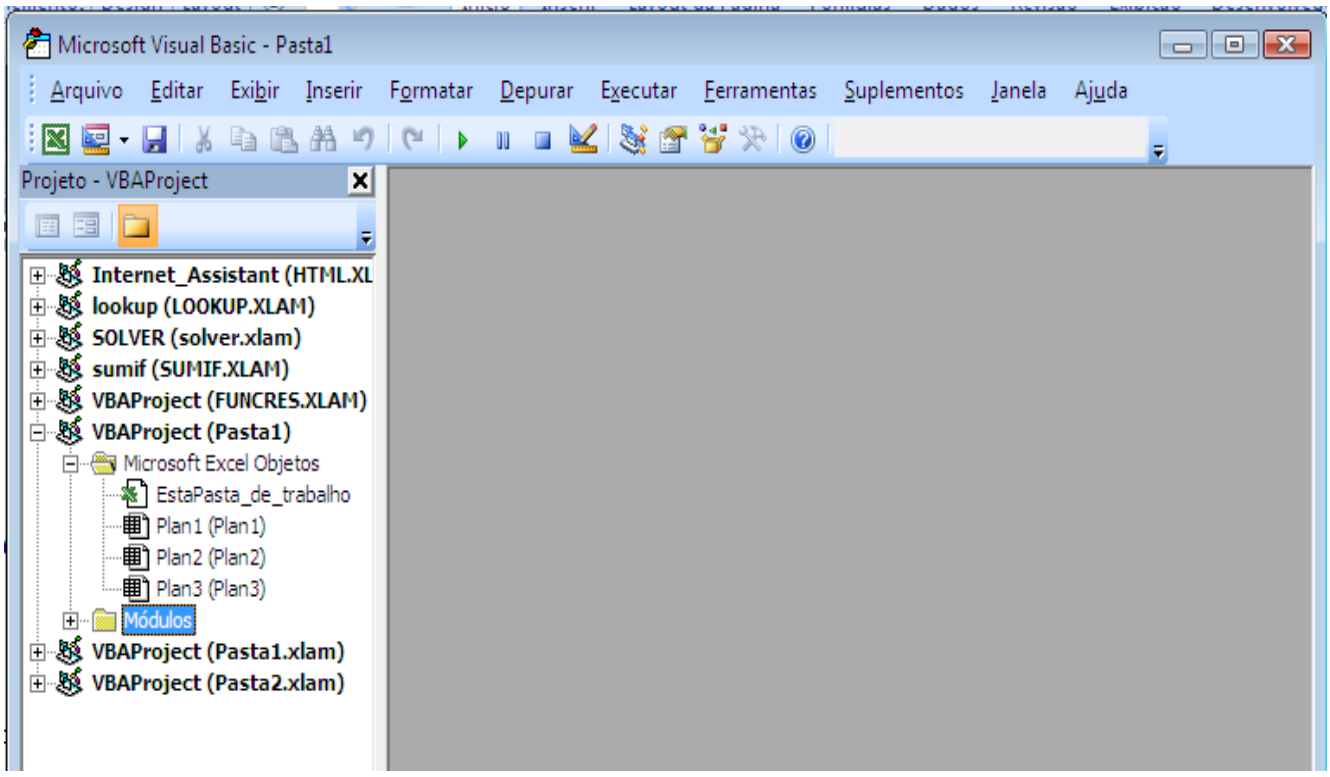


Agora você tem uma macro gravada que configura o fundo da célula para amarelo claro.

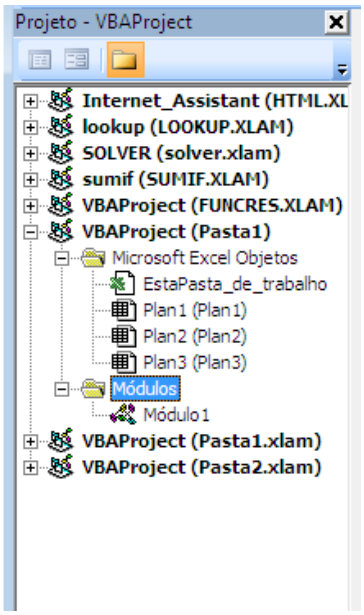
Ver a Sintaxe Gravada

A macro gravada está pronta para uso. Antes de rodarmos a marco, vamos investigar a sintaxe.

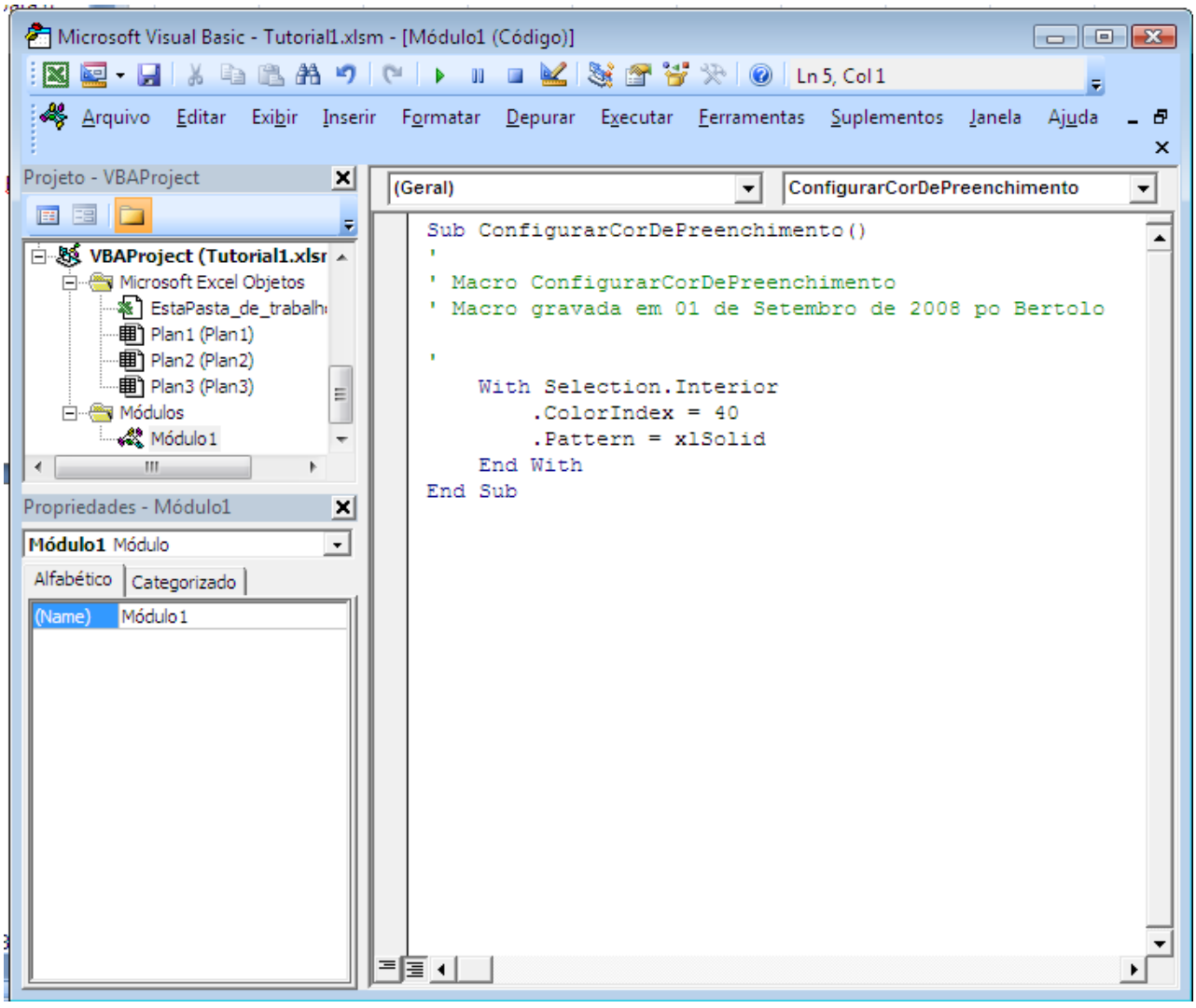
1. Para carregar o Visual Basic Editor, pressione [Alt] e [F11] ao mesmo tempo. (Lembre-se de nossa lição anterior?) O Visual Basic Editor aparece.



2. Expandir a pasta **Módulos** no **Project Explorer** clicando no sinal de mais (+).



3. Duplo clique na pasta **Módulo1** para ver a sub-rotina (macro).

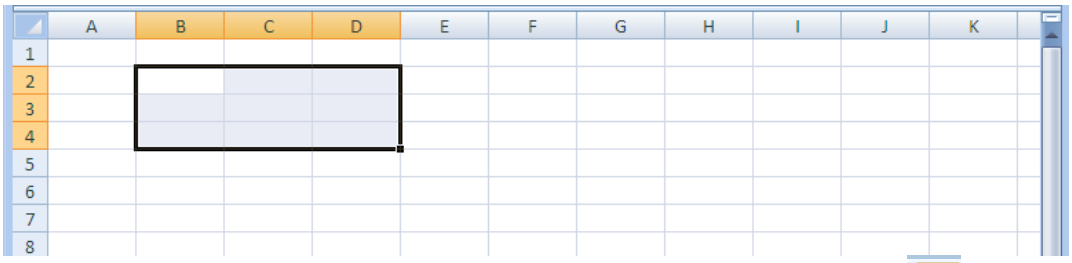


Como a figura mostra, o nome da sub-rotina é "ConfigurarCorDePreenchimento". O índice de cor para o laranja claro é 40. O padrão do fundo (background) é sólido.

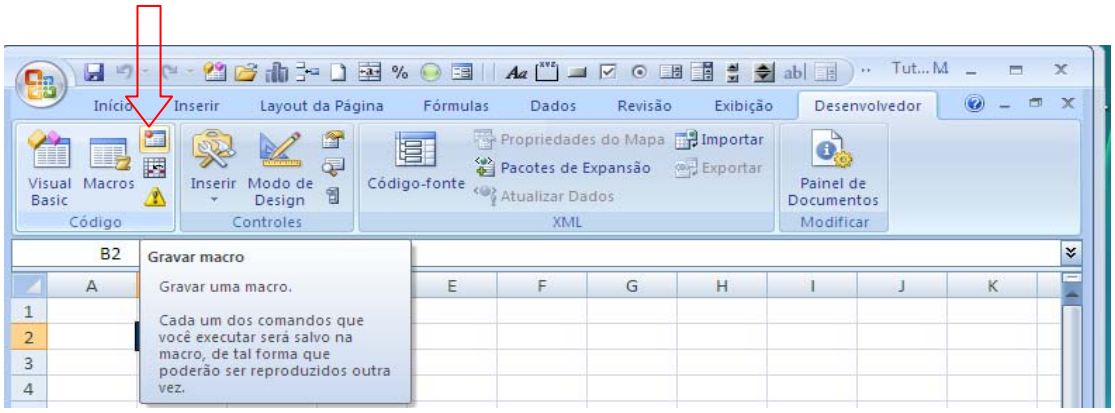
Executar a Macro Gravada

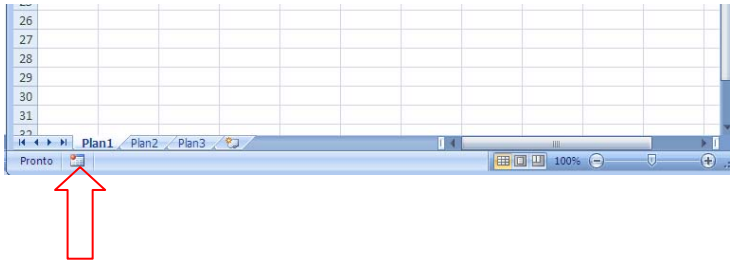
No nosso exemplo anterior, criamos uma macro "Alô Mundo!". Nós rodamos a macro dentro do Visual Basic Editor. Desta vez nós rodamos o gravador de macro na planilha.

1. Em qualquer planilha, selecione de B2 a D4.

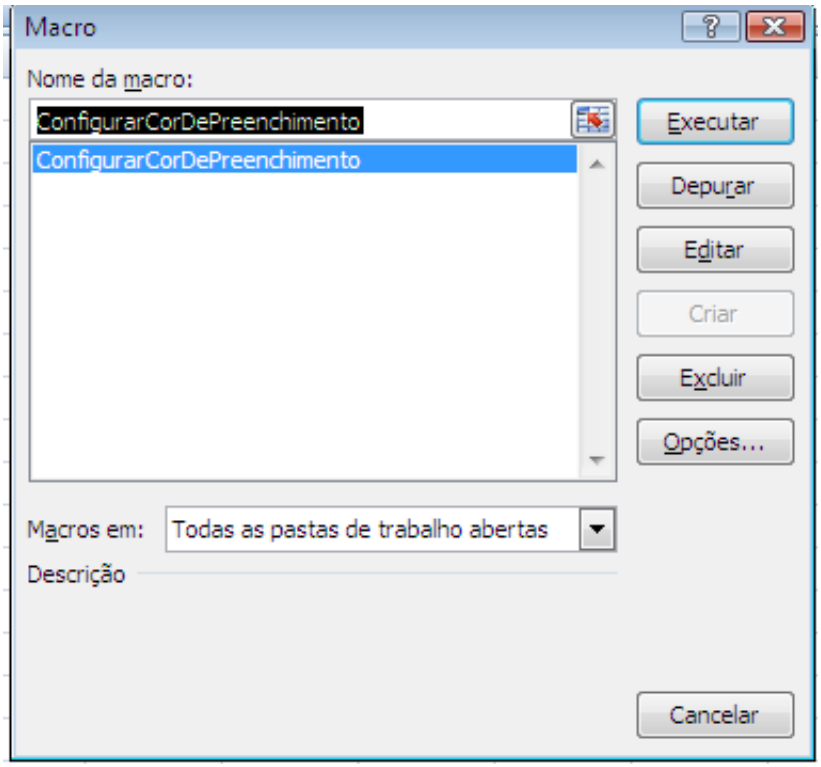


2. Rodar a macro gravada selecionando a guia **Desenvolvedor**...o grupo **Código**...o ícone  ... ou pressionando [Alt] e [F8] ao mesmo tempo. Ou ainda junto as guias de planilha, no fundo da planilha selecione o ícone 

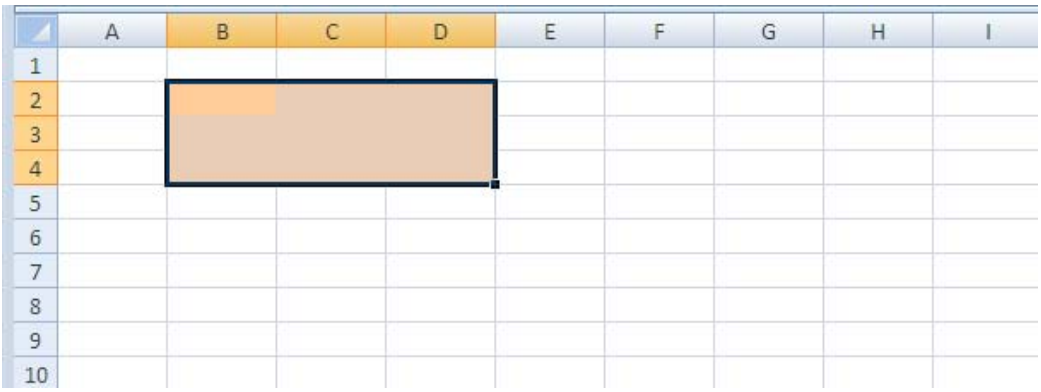




3. A caixa de diálogo Macro é mostrada. Desde que existe somente uma macro no módulo, por default a única macro, ConfigurarCorDePreenchimento é selecionada. Clique no botão **Executar** para rodar a macro.



4. As células do intervalo B2:D4 agora tem cor de fundo (background) laranja claro.



Módulos e Procedimentos

Módulos e Procedimentos e Seus Escopos

Um **módulo** é um recipiente (container) para procedimentos como mostrado nos nossos exemplos anteriores. Um **procedimento** é uma unidade de código confinado entre as declarações **Sub** e **End Sub** ou entre as declarações **Function** e **End Function**.

O procedimento sub (ou sub-rotina) seguinte imprime a data e o tempo atual na célula C1:

```
Sub MostrarTempo()

    Range("C1") = Now()
```

```
End Sub
```

A função seguinte soma até dois números:

```
Function somaNo(x, y)

     somaNo = x + y

End Function
```

Procedimentos no Visual Basic podem ter ou o escopo private ou public. Um procedimento com escopo private é somente acessível aos outros procedimentos no mesmo módulo; um procedimento com escopo public é acessível a todos procedimentos em cada módulo na pasta de trabalho na qual o procedimento é declarado, e em todas as pastas que contenham uma referência àquela pasta. Por default, procedimentos tem escopo public.

Aqui estão exemplos de definição do escopo para procedimento.

```
Public Sub MostrarTempo()
    Range("C1") = Now()
End Sub

Private Sub MostrarTempo ()
    Range("C1") = Now()
End Sub
```

Chamando Procedimentos Sub e Procedimentos Function

Existem duas maneiras de se chamar um procedimento sub. O exemplo seguinte mostra como um procedimento sub pode ser chamado pelos outros procedimentos sub.

```
Sub z(a)
  MsgBox a
End Sub

Sub x()
    Call z("ABC")
End Sub

Sub y()
    z "ABC"
End Sub
```

O procedimento sub **z** tem um argumento (a) e exibe o valor argumento ("ABC") numa caixa de mensagem. Rodar ou Sub x ou Sub y conduzirá ao mesmo resultado.

O exemplo seguinte chama um procedimento function de um procedimento sub.

```
Sub MostrarSoma()
     msgbox somaNo(3,5)
End Sub
```

```
Function somaNo(x, y)
     somaNo = x + y
End Function
```

O procedimento sub MostrarSoma chama a função somaNo e retorna um "8" numa caixa de mensagem.

Se houverem procedimentos com nomes duplicados em diferentes módulos, você deve precisar incluir um qualificador de módulo antes do nome do procedimento quando chamar o procedimento.

Por exemplo:

Módulo1.MostrarSoma

Passando Argumento por Referência ou por Valor

Se você passar um argumento **por referência** quando chamar um procedimento, o procedimento acessa à variável atual na memória. Como resultado, o valor da variável pode ser mudado pelo procedimento. Passar por referência é o default no VBA. Se você não especificar explicitamente em passar um argumento **por valor**, o VBA o passará por referência. As duas declarações seguintes conduzem ao mesmo resultado.

```
   Sub AdicionaNo(ByRef x as integer)
Sub AdicionaNo(x as integer)
```

Aqui está um exemplo para mostrar o comportamento por referência. O procedimento sub, TestePassagem1 chama AdicionaNo1 por referência e mostra "60" (50 + 10) na caixa de mensagem.

```
Sub TestePassagem1()
    Dim y As Integer
    y = 50
    AdicionaNo1 y
    MsgBox y
End Sub

Sub AdicionaNo1(ByRef x As Integer)
    x = x + 10
End Sub
```

O exemplo seguinte mostra o comportamento por valor. O procedimento sub, TestePassagem2 chama AdicionaNo2 por valor e mostra "50" na caixa de mensagem.

```
Sub TestePassagem2()
    Dim y As Integer
    y = 50
    AdicionaNo2 y
    MsgBox y
End Sub

Sub AdicionaNo2(ByVal x As Integer)
    x = x + 10
End Sub
```