

Treinamento de VBA

EMPRESA
JÚNIOR

FGV
consultoria
desde 1988

Embaixadores:

Enzo Paes

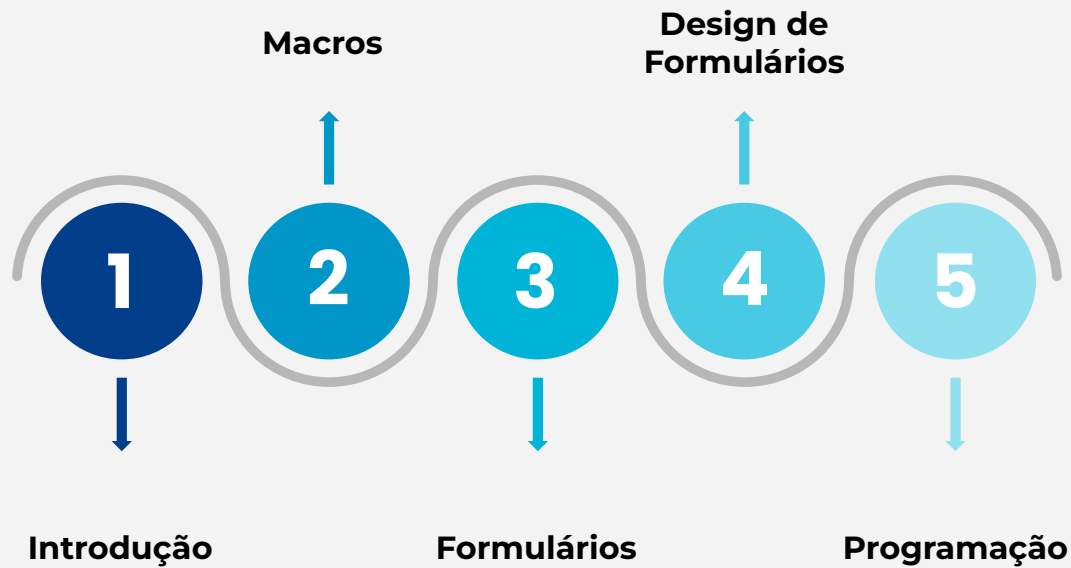
Henrique Melquiades

Manoela Brandi

Coordenador:

Esthevão Marttioly

Sumário



A person's hand is shown typing on a laptop keyboard. The laptop screen and the surrounding area are overlaid with a complex, futuristic digital interface. This interface includes various icons such as a shield, gears, a target, and a magnifying glass, along with snippets of code and data visualizations like bar charts and line graphs. The background is dark with blurred city lights, creating a high-tech, cybernetic atmosphere.

1

Introdução

Introdução

O **Visual Basic for Applications (VBA)** é a linguagem de programação por trás do **Excel** e tem como objetivo **automatizar** alguns comandos repetidos por meio de **botões** programados.

Há **dois** tipos de ferramentas principais que podem ser feitas pelo VBA:

Macros: é possível gravar uma sequência de ações no Excel que rodam automaticamente após clicar em algum botão.

Formulários: são telas em que se permite colocar caixas de texto para depois puxar suas informações para o Excel.

UserForm1

Insira os valores de Gastos e Receitas para o Projeto.

CUSTO DE AQUISIÇÃO	R\$		DESPESAS TRIBUTÁRIAS	R\$	
CUSTOS DE INCORPORAÇÃO	R\$		PIS	R\$	
Mão de Obra própria e Encargos (Incorporação)	R\$		COFINS	R\$	
Serviços de Terceiros (Incorporação)	R\$		ITBI	R\$	
Materiais Aplicados (Incorporação)	R\$		IRPJ	R\$	
Equipamentos de Obra (Incorporação)	R\$		CSLL	R\$	
CUSTOS DE REFORMA	R\$		IPTU	R\$	
Mão de Obra própria e Encargos (Reforma)	R\$		ADIANTAMENTO AOS FORNECEDORES	R\$	
Serviços de Terceiros (Reforma)	R\$		Adiantamento aos Fornecedores	R\$	
Materiais Aplicados (Reforma)	R\$		Adiantamento de compra de Imóveis e Terreno	R\$	
Equipamentos de Obra (Reforma)	R\$		DESPESAS JUDICIAIS	R\$	
DESPESAS ADMINISTRATIVAS	R\$		Taxas Judiciais	R\$	
Despesas Gerais	R\$		DESPESAS COM INCORPORAÇÃO DE EMPREENDIMENTO	R\$	
Imobilizado	R\$		Cartórios e Registros (Incorporação)	R\$	
DESPESAS FINANCEIRAS	R\$		DESPESAS COM REFORMA DO IMÓVEL	R\$	
Juros de Mora	R\$		Cartórios e Registros (Reforma)	R\$	
Taxas Bancárias	R\$		DESPESAS COM VENDAS	R\$	
Multas	R\$		Brindes (Vendas)	R\$	
Descontos Obtidos	R\$		Vale Alimentação (Vendas)	R\$	
RECEITA	R\$				
Valor de Venda do Imóvel (Próprio)	R\$				
Comissão Recebida (Investidor)	R\$				
Valor de Venda do Imóvel (Investidor)	R\$				
Comissão Cobrada (Investidor)	R\$				

Após incluir todos os Gastos e Receitas do Projeto:

- Certifique-se de que utilizou "." (ponto) para separar os centavos
- Clique no botão abaixo para adicionar os dados à planilha.

Inserir Projeto

Projeto Homem-hora

No PO Início

Site Type SLA

Cliente Status

Margem de Lucro

Imposto

Inserir

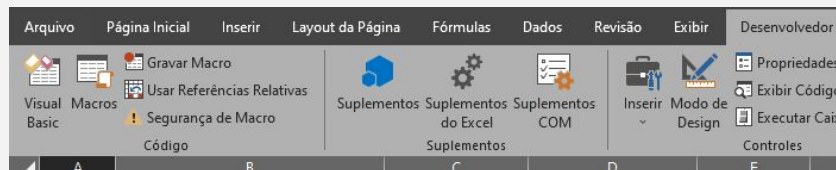
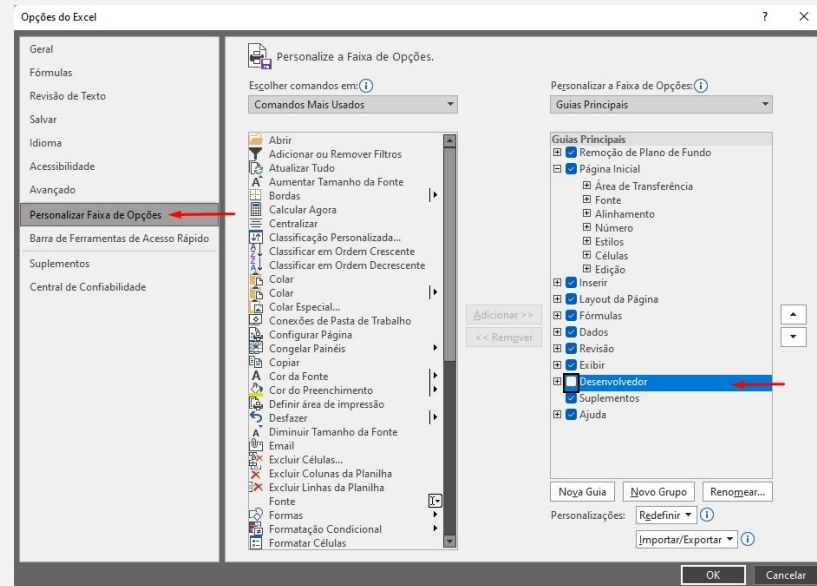
Limpar

Introdução

Primeiramente, deve-se habilitar o **guia desenvolvedor** no Excel. Deve-se ir em “Arquivo” -> “Opções” -> “Personalizar a faixa de Opções” e depois habilitar o **Desenvolvedor** na parte direita, assim como mostra a imagem.

Na guia, aparece a parte de **Visual Basic** (VBA para criar formulários ou editar macros) e uma parte de **Macros**, que permite encontrar as macros.

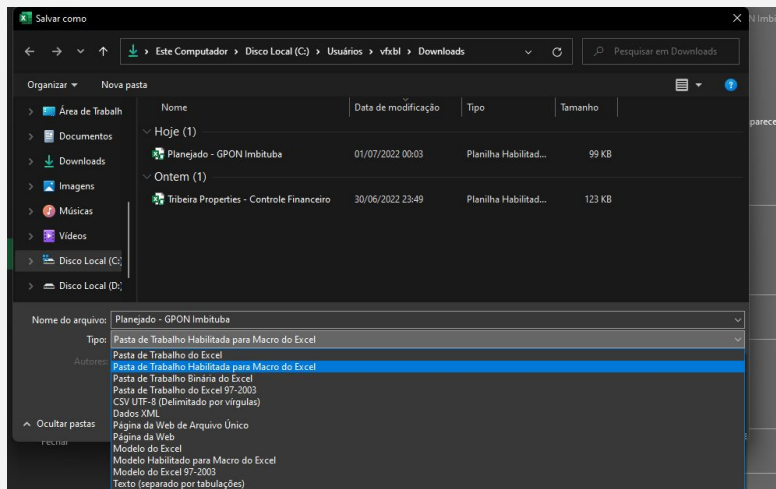
Ainda, é importante dizer que, quando um arquivo com VBA ou macros é salvo pela primeira vez, deve-se **mudar o tipo do arquivo**.



Introdução

Para **salvar o arquivo** com base na permissão para **macros**, deve-se ir em **“Arquivos” -> “Salvar como”** e depois, em tipo, deve selecionar a parte de **“Pasta de Trabalho Habilitada para Macro do Excel”**.

Dado isso, o formato do arquivo vai ser diferente e ele permite a **habilitação para Macros e formulários do VBA**. Depois da primeira vez, não é necessário mais fazer esse processo novamente, pois ele já salva assim automaticamente.



2

Macros

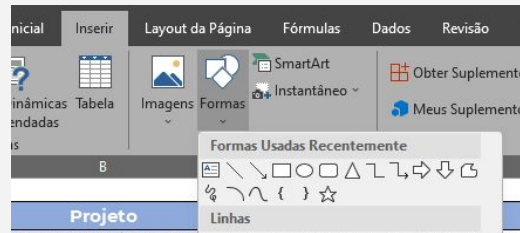
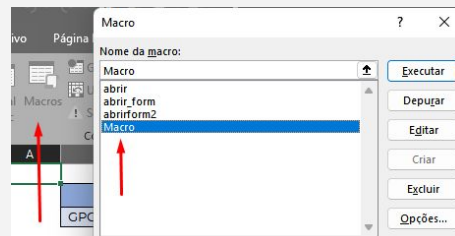
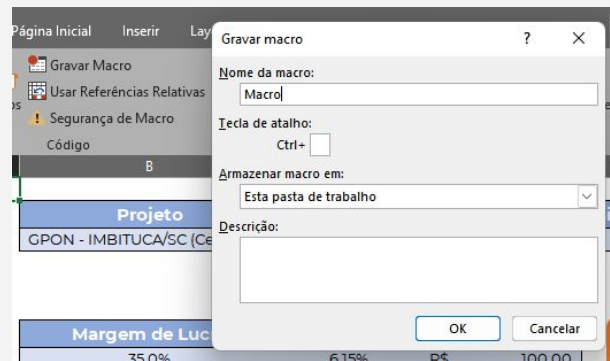


Macros

Como já dito, uma **macro** permite salvar uma **sequência de ações** para depois **executá-la** a partir de um único botão. Para gravar uma macro, deve-se ir em **“Desenvolvedor”** e clicar em **“Gravar uma macro”** e, com isso, ela começa a ser gravada.

Durante a **gravação**, ela **salva todas as ações** que serão feitas por você e, quando **finalizar**, basta clicar em **“Parar gravação”**. A macro será salva em **“Macros”** e é possível **executá-la** (realizar os comandos feitos), **editá-la** (alterar o código feito em VBA), **depurar** (verificar a sequência de comandos em VBA) e **Excluir**.

Para criar um botão, basta ir em **“Inserir” -> “Formas”** e criar uma **forma** de sua preferência, também a editando para ficar do seu agrado.

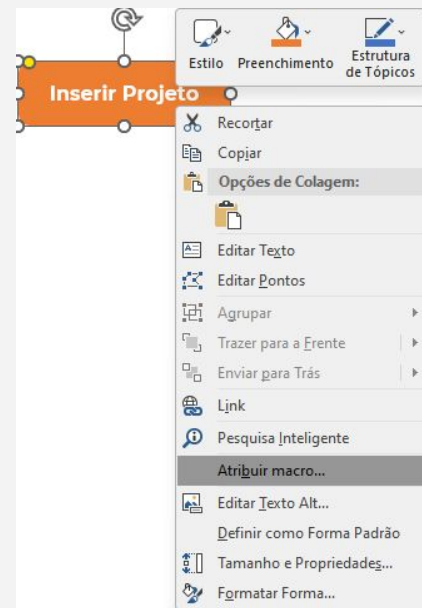


Macros

Depois disso, é só **clicar** na forma criada com o **botão direito** e ir em **“Atribuir macro...”**, selecionar a macro criada ou a que quer executar com aquele botão e ir em **“Ok”**. Agora, sempre que **apertar no botão (forma)**, o Excel roda a **macro** que você fez.

Isso permite várias **automações**, por exemplo, caso queira fazer **lançamentos** de alguns itens de uma **planilha** para **outra**, caso queira acrescentar **linhas automaticamente**, caso queira **remover duplicatas**, transformar **texto para colunas**, etc...

E, sempre que quiser alterar a forma novamente, basta **clicar com o botão direito** nela e ir em **“Página Inicial”**.



A person's hand is shown typing on a laptop keyboard. The laptop screen displays a complex digital interface with various icons, charts, and text. A semi-transparent blue rectangular box is overlaid on the screen, containing the number '3'. The background is dark with blurred city lights and a large red sphere on the right side.

3

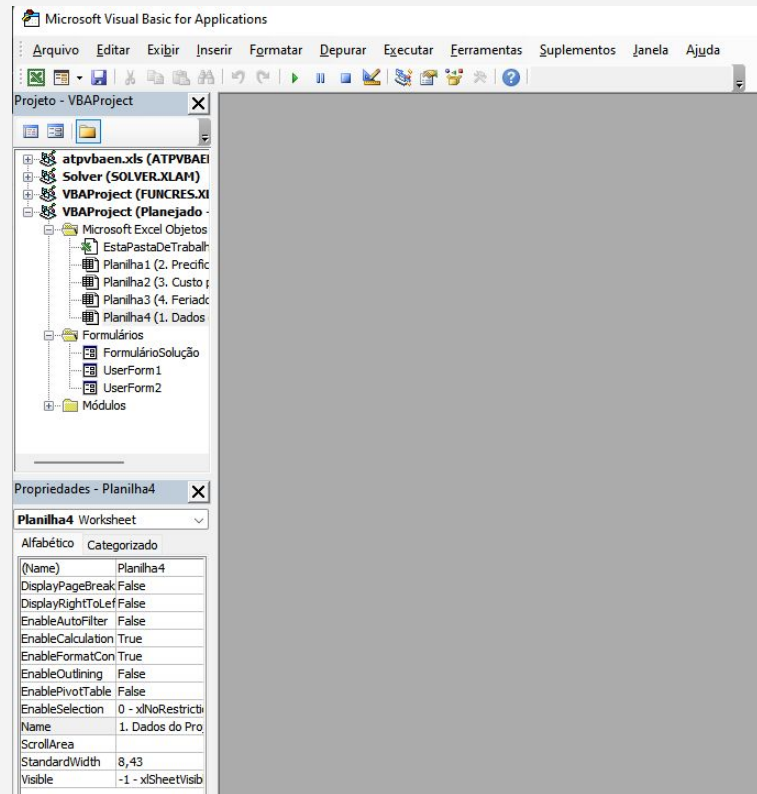
Formulários

Formulários

Para abrir o **VBA**, basta ir em “**Desenvolvedor**” e clicar em “**Visual Basic**”. O VBA tem um formato muito característico, em cima, há várias abas de “**arquivo**”, “**editar**”, “**exibir**”, **etc.** Também há opções de **ativar** as **janelas** da esquerda caso sejam fechadas sem querer.

Na **primeira janela (projeto)**, há todos os **arquivos criados**, sejam eles **formulários (UserForm)** ou **macros (módulos)**. Em baixo, há as propriedades, que servem para alterar o **design** dos formulários e serão de grande importância.

Para exibir **formulários ou módulos**, basta clicar neles duas vezes. Para inserir algum formulário ou módulo, basta ir em “**Inserir**” -> “**UserForm**” ou “**Módulo**”. As macros gravadas serão os **módulos** dentro do VBA.

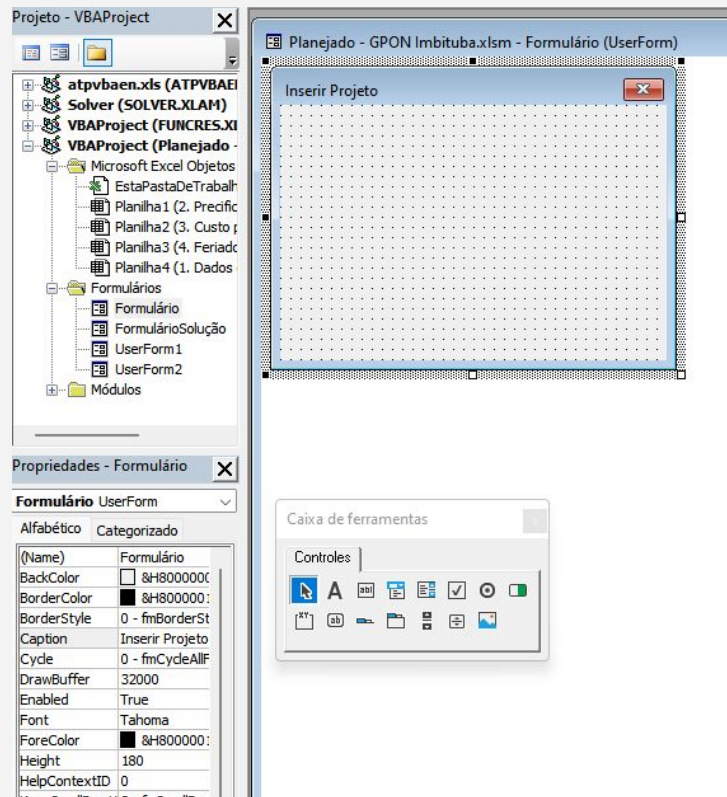


Formulários

Depois de criado um **formulário**, ele aparecerá na esquerda com um nome clássico de **UserForm** e aparecerá o **formulário em branco** e uma **caixa de ferramentas** para controlá-lo.

Para alterar seu **nome**, deve-se mudar o nome em **(Name)**, por exemplo, para **“Formulário”**. Para alterar a escrita em cima da caixa de formulário, altera-se a **“Caption”** dele. Depois disso, pode-se alterar seu design conforme desejar, o que será comentado no próximo tópico.

Além disso, também há a opção de **criar novas caixas** a partir da **caixa de ferramentas**.

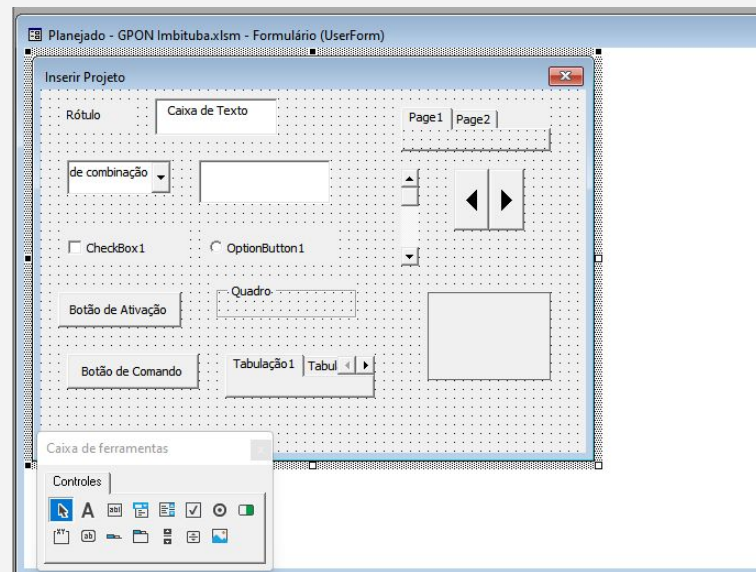


Formulários

Na **caixa de ferramentas**, primeiro aparece o **rótulo**, que é a **escrita normal** para qualquer texto que se deseja colocar. Depois, há a **caixa de texto**, que representa uma **caixa** em que é possível escrever quando o formulário é aberto.

Na **caixa de combinação**, há a **listagem** de diversos itens que podem ser colocados nela. A **caixa de listagem** funciona como uma **tabela** dentro do VBA. Abaixo, também há a **caixa de seleção** e o **botão de opção**, para assinalar se está feito.

Também há um **botão de ativação**, que é um botão que continua pressionado após clicado. Em paralelo, o **botão de comando** não continua pressionado após clicado e é mais usado como o **“Ok”** final do formulário.

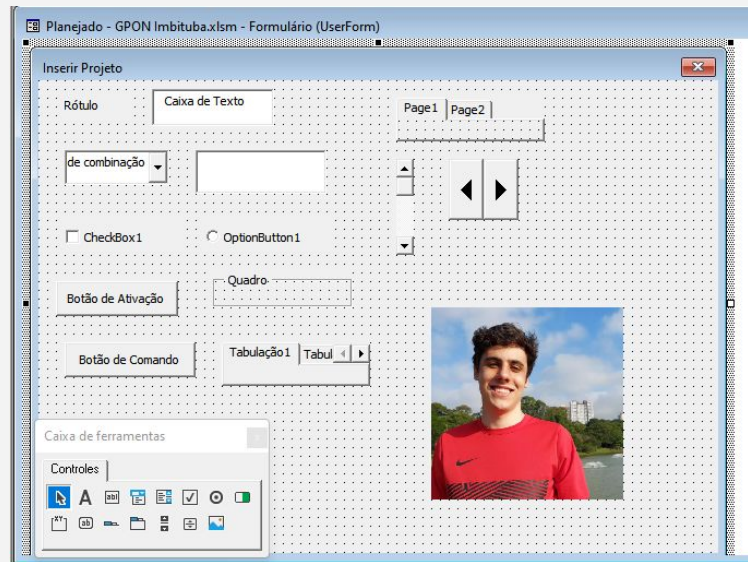


Formulários

Por fim, há o **quadro**, que apresenta informações dentro dele. O **TabStrip** e o **MultiPage** servem para **alterar os dados dentro de várias informações** quando selecionado um ou outro e não são tão utilizados.

A **barra de rolagem** serve para ir para cima ou para baixo nas informações. O botão de rotação serve para alterar as informações para a **direita ou para a esquerda**.

E, finalmente, também há a **imagem**, que pode ser **importada** pelo **VBA** por essa opção. Para ajustar o tamanho da foto, deve ir em **PictureSizeMode** e selecionar o valor **3**. Depois, basta **tirar a parte de sobra** em um dos lados e colocar o **PictureSizeMode 1**.



MousePointer	0 - fmMousePoin
Picture	(Bitmap)
PictureAlignment	2 - fmPictureAlig
PictureSizeMode	3 - fmPicture ▾
PictureTiling	False
SpecialEffect	0 - fmSpecialEffe

4

Design de Formulários

Design de Formulários

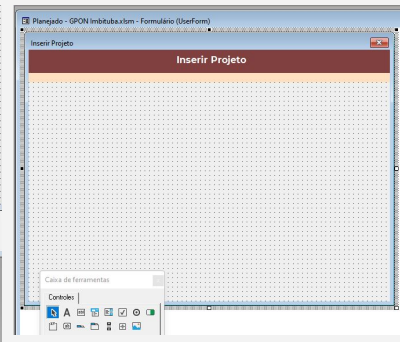
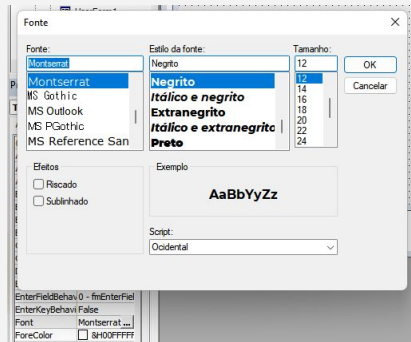
Primeiramente, para criar **alguma forma** para conter espaços no formulário, **basta criar um rótulo** (texto) e pintar a **BackColor** da cor desejada (pode-se ir em **Paleta** para aparecer mais opções).

Caso ainda queira escrever algum texto na caixa, basta ir em **Caption** e escrever o **nome**, por exemplo **“Inserir Projeto”**. Para centralizar o texto, vá em **TextAlign** e escolha a opção **2**. Para **alterar a fonte e colocar negrito**, vá em **Font** e escolha a fonte, o estilo e o tamanho desejados.

Para **alterar a cor de fundo do formulário**, clique no **formulário** e altere **BackColor**. No exemplo da direita, há 3 textos, dois para o fundo e um para o texto de fato, e o fundo está pintado de cinza claro.



Tabstop	true
Tag	
Text	Inserir Projeto
TextAlign	2 - fmText#
Top	0
Value	Inserir Projeto
Visible	True




Design de Formulários

Depois de **acrescentar todos os textos, caixas de texto** e **caixas de combinação** (para listagem), é encontrado o resultado a seguir. Além disso, também são colocados **dois botões de comando** para **inserir** os dados ou **limpá-los**.

Porém, ainda falta a **inteligência** por trás do formulário, para que ele funcione como deseja. Para entrar no **código do formulário**, basta clicar nele, ir em **“Exibir” -> “Código”**. Caso queira fazer aparecer novamente o formulário e ele não esteja aparecendo, basta clicar duas vezes no **UserForm** à esquerda ou ir em **“Exibir” -> “Objeto”**.

O **código das macros (módulos)** tem o mesmo **formato** e a mesma **linguagem**.



The image shows a screenshot of a web application window titled "Inserir Projeto". The window has a dark red header bar with the title "Inserir Projeto" and a close button (X) in the top right corner. Below the header, there is a light orange horizontal bar. The main content area is white and contains several input fields and two buttons. The input fields are arranged in two columns. The left column contains: "Projeto" (text input), "No PO" (text input), "Site Type" (text input), "Cliente" (dropdown menu), "Margem de Lucro" (text input), and "Imposto" (text input). The right column contains: "Homem-hora" (text input), "Início" (text input), "SLA" (text input), "Status" (dropdown menu), and two buttons: "Inserir" and "Limpar". The buttons are dark red with white text.

Inserir Projeto	
Projeto	Homem-hora
No PO	Início
Site Type	SLA
Cliente	Status
Margem de Lucro	Inserir
Imposto	Limpar

A person's hand is shown typing on a laptop keyboard. The laptop screen and the surrounding area are overlaid with a complex, futuristic digital interface. This interface includes various icons such as a shield, gears, a target, and a network diagram, along with snippets of code and data visualizations like bar charts and line graphs. The background is dark with out-of-focus city lights in red, yellow, and blue, creating a bokeh effect. A semi-transparent dark horizontal band runs across the middle of the image, serving as a backdrop for the title.

5

Programação

Programação

Primeiramente, os **códigos do VBA** são separados em **subs**, de modo que, sempre que se inicia um código novo, deve-se escrever **sub nome()** e, depois que der **enter**, aparece um **End Sub** no final.

O primeiro **módulo** que deve ser criado é uma macro para abrir o formulário para que depois atribua-se essa **macro** a um **botão** (forma) dentro do **Excel**.

Então, no VBA, vá em **“Inserir” -> “Módulo”** e, nele, escreve **sub mostrar_formulário()**, sendo que **“mostrar_formulário”** pode ser o nome que **preferir** (é o nome da macro). Agora, dentro da **sub**, coloca-se o **código** que deve ser **rodado** quando ocorrer a **macro**.

```
Planejado - GPON Imbituba.xlsm - Formulário (Código)
(Geral)
Sub nome()
End Sub
```

```
Planejado - GPON Imbituba.xlsm - Módulo4 (Código)
(Geral)
Sub mostrar_formulário()
End Sub
```

```
Planejado - GPON Imbituba.xlsm - Módulo4 (Código)
(Geral)
Sub mostrar_formulário()
    Formulário.Show
End Sub
```

Programação

No **VBA**, os **códigos** normalmente são escritos da seguinte forma. Primeiro, deve-se dizer o **objeto** ao qual você quer fazer a **ação** e, como queremos **referir** ao **formulário**, escrevemos primeiro seu **nome**.

Depois, escreve-se **ponto (.)** e depois a **ação** que deve ser **feita** (aparece **várias caixas** com o que pode ser **feito**). Com isso, escreve-se então a **fórmula**:

Formulário.Show

```
Planejado - GPON Imbituba.xlsm - Formulário (Código)
(Geral)
Sub nome()
End Sub
```

```
Planejado - GPON Imbituba.xlsm - Módulo4 (Código)
(Geral)
Sub mostrar_formulário()
End Sub
```

```
Planejado - GPON Imbituba.xlsm - Módulo4 (Código)
(Geral)
Sub mostrar_formulário()
    Formulário.Show
End Sub
```

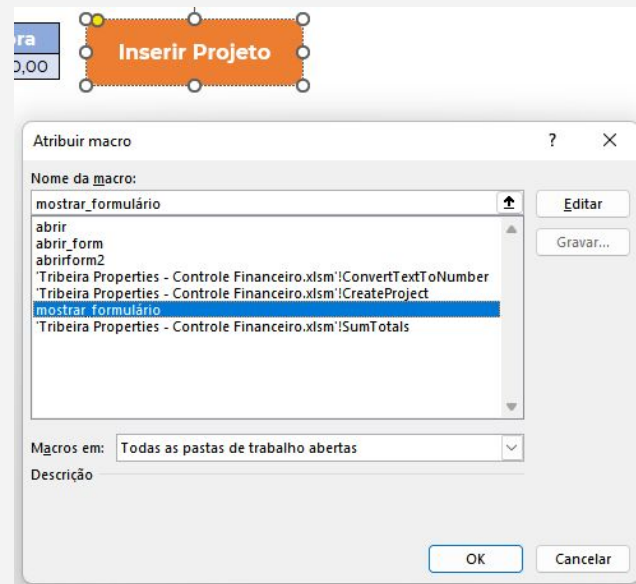

Programação

Com isso, a **macro** é criada, então basta criar uma forma para **inserir o projeto**, clicar com o botão direito, ir em **“Atribuir macro”** e selecionar o **mostrar_formulário** (macro criada para abrir).

Agora, é necessário dizer para o VBA que os termos que estão nas **caixas de texto do formulário** vão parar em cada uma das **células** do Excel. Então, vá no **formulário**, **“Exibir”** e **“Código”**.

Há duas opções de fazer o texto do formulário que deve ir para o Excel: (i) ao clicar no botão **“Inserir”** ou **“Ok”**, (ii) ou após mudar cada **caixa de texto**.

O primeiro método permite que **o que estava na célula anteriormente continue** mesmo sem mudar no formulário, enquanto o segundo não. Porém, o **segundo é mais dinâmico e não exige** o comando de **“Ok”**.



Programação

Em relação ao primeiro modo, como **o código de inserir o texto vai ser executado quando clicar no botão**, na caixa da esquerda deve escolher o **Botão de Comando**, no caso com nome **CommandButton1** (o nome pode ser descoberto ao ir no formulário e clicar nele), e depois ir em **“Click”** na direita.

Isso vai criar uma **Sub** que vai ser executada quando alguém **clicar no botão inserir**. Porém, deve-se especificar o que acontece.

Para facilitar o código, **deve ir em cada caixa de texto e alterar o nome dele** para algum mais fácil. No caso, coloquei o nome de cada escrito que está do lado.

```
Planejado - GPON Imbituba.xlsm - Formulário (Código)
CommandButton1 Click
Private Sub CommandButton1_Click()
End Sub
```

The screenshot displays the VBA development environment for an Excel spreadsheet. On the left, the **Project Explorer** shows a hierarchy with 'Formulário' containing 'FormulárioSoluçã', 'UserForm1', and 'UserForm2'. Below this, 'Módulos' contains 'Módulo1' through 'Módulo4'. The 'VBAProject (Tribeira)' folder is expanded, showing 'Microsoft Excel Objects' and 'EstaPastaDeTral'. The **Properties - Projeto** window is open, showing the 'Projeto' TextBox with its '(Name)' property set to 'Projeto'. A red arrow points to this property. On the right, a user form is visible with several input fields: 'Projeto' (a dropdown menu), 'No PO' (a text box), 'Site Type' (a text box), 'Cliente' (a dropdown menu), and 'Margem de Lucro' (a text box). The form has a dotted background and a title bar that says 'Inserir Proj'.

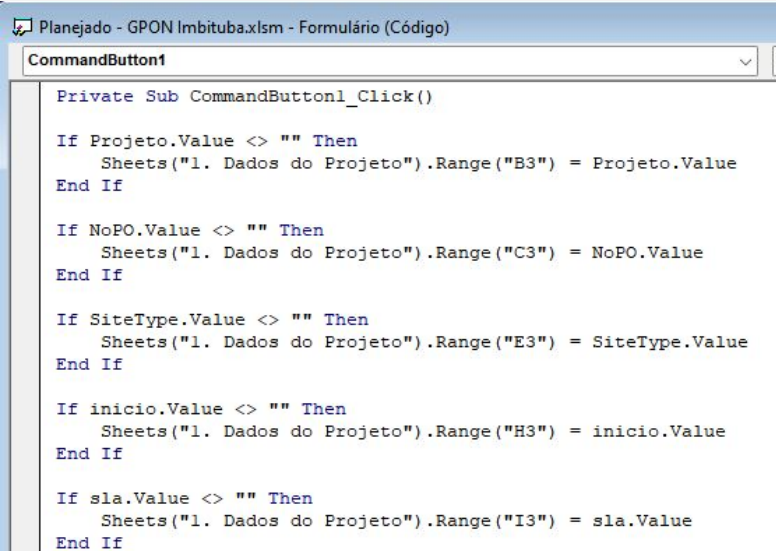
Programação

Primeiramente, para se referir a alguma célula do Excel, escreva **Range(“código da célula”)**, sendo que o código da célula é por exemplo **“A21”**, sendo coluna A e linha 21. E, para referir ao **valor** que está na **caixa de texto**, escreve-se **Caixa_de_Texto.Value**, sendo **Caixa_de_Texto** o nome colocado na caixa de texto.

Além disso, caso o **botão de executar o formulário esteja na mesma aba de onde serão incluídos o valores**, não é necessário dizer a **aba**, pois por padrão ele considera **ActiveSheet** (planilha atual).

Porém, quando há outra planilha, o código se torna:

Sheets(“Nome da Planilha”).Range(“código da célula”) = Caixa_de_Texto.Value



```
Planejado - GPON Imituba.xlsm - Formulário (Código)
CommandButton1

Private Sub CommandButton1_Click()

    If Projeto.Value <> "" Then
        Sheets("1. Dados do Projeto").Range("B3") = Projeto.Value
    End If

    If NoPO.Value <> "" Then
        Sheets("1. Dados do Projeto").Range("C3") = NoPO.Value
    End If

    If SiteType.Value <> "" Then
        Sheets("1. Dados do Projeto").Range("E3") = SiteType.Value
    End If

    If inicio.Value <> "" Then
        Sheets("1. Dados do Projeto").Range("H3") = inicio.Value
    End If

    If sla.Value <> "" Then
        Sheets("1. Dados do Projeto").Range("I3") = sla.Value
    End If

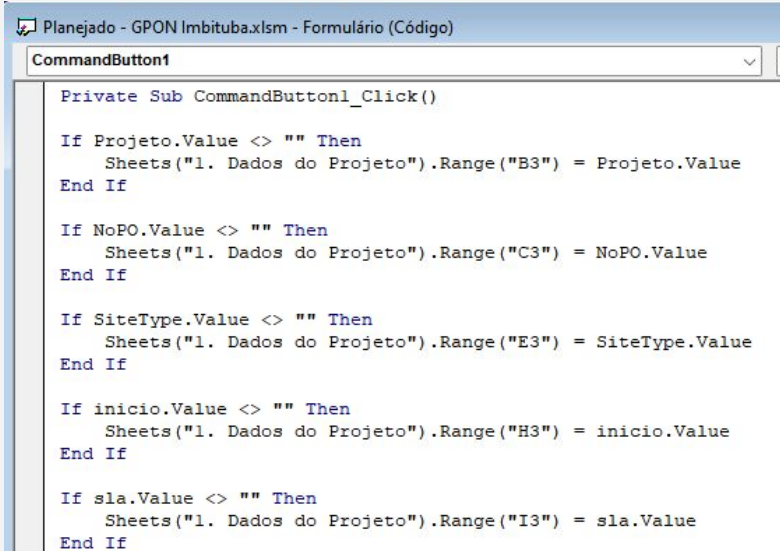
End Sub
```

Programação

Nesse caso, foi inserido um comando **If**. O comando **If é igual ao =se()** do **Excel**, ele estabelece uma **condição** e diz o que deve ser **feito se a condição é verdadeira**.

Aqui, a condição é que a caixa de comando deve ser **diferente de nada**, senão, quando não for preenchido um valor, a célula tomará um valor **nulo**, substituindo o **valor antigo**.

Portanto, coloca-se **If Caixa_de_texto.Value <> "" Then** (se o valor da caixa de texto for diferente de nulo então...). Além disso, todo comando **If** no VBA **deve acabar** com **End If**, então também coloca-se esse comando. Caso queira colocar outra condição dentro do **If**, antes de **End If** deve colocar **Else**



```
Private Sub CommandButton1_Click()  
  
If Projeto.Value <> "" Then  
    Sheets("1. Dados do Projeto").Range("B3") = Projeto.Value  
End If  
  
If NoPO.Value <> "" Then  
    Sheets("1. Dados do Projeto").Range("C3") = NoPO.Value  
End If  
  
If SiteType.Value <> "" Then  
    Sheets("1. Dados do Projeto").Range("E3") = SiteType.Value  
End If  
  
If inicio.Value <> "" Then  
    Sheets("1. Dados do Projeto").Range("H3") = inicio.Value  
End If  
  
If sla.Value <> "" Then  
    Sheets("1. Dados do Projeto").Range("I3") = sla.Value  
End If  
  
End Sub
```


Programação

Além disso, para programar o botão de “**Limpar**”, basta **formatar** todos os valores das **Caixas de Texto** como **nulo**, ou seja, para todas as caixas de texto:

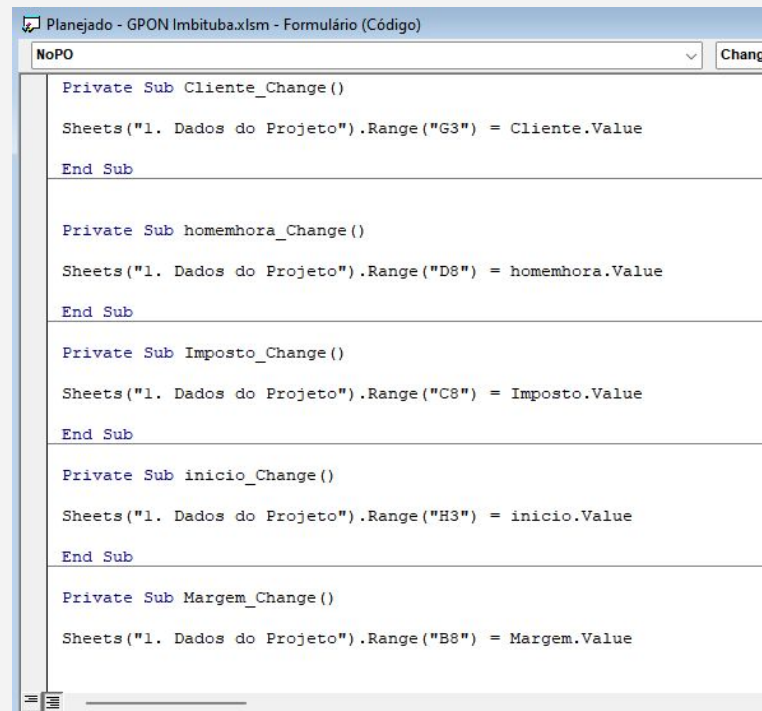
Caixa_de_Texto.Value = ""

```
Private Sub CommandButton2_Click()  
  
    Solucao.Value = ""  
    Cliente.Value = ""  
    homemhora.Value = ""  
    Imposto.Value = ""  
    inicio.Value = ""  
    Margem.Value = ""  
    PO.Value = ""  
    SiteType.Value = ""  
    sla.Value = ""  
  
End Sub
```

Programação

Em paralelo a isso, também há um modo de **alterar os valores das células** enquanto se **escreve o comando**. Para isso, deve-se **selecionar cada uma das Caixas de Comando** criadas ao invés de “**CommandButton1**” e na direita selecionar “**Change**” (mudança).

Nesse caso, não é necessário mais colocar o **If**, pois o **comando só considera mudanças no formulário**. O grande problema desse método é que ele **não guarda** as informações **anteriores** caso haja uma mudança no formulário.



```
Planejado - GPON Imbituba.xlsm - Formulário (Código)

NoPO
Change

Private Sub Cliente_Change()

    Sheets("1. Dados do Projeto").Range("G3") = Cliente.Value

End Sub

Private Sub homemhora_Change()

    Sheets("1. Dados do Projeto").Range("D8") = homemhora.Value

End Sub

Private Sub Imposto_Change()

    Sheets("1. Dados do Projeto").Range("C8") = Imposto.Value

End Sub

Private Sub inicio_Change()

    Sheets("1. Dados do Projeto").Range("H3") = inicio.Value

End Sub

Private Sub Margem_Change()

    Sheets("1. Dados do Projeto").Range("B8") = Margem.Value

End Sub
```

Programação

Por fim, também é necessário definir qual a **lista** que entrará nas **caixas de listagem**. Para isso, deve haver uma tabela no **Excel** mostrando todas as **opções**. Na direita, há **duas listas** que entrarão no **formulário**.

Para isso, as listas serão puxadas quando o **formulário** for **aberto**, então deve-se selecionar **UserForm** e **Initialize** na hora de escrever o **código**.

No caso, **cliente** será puxada a **lista inteira**, então basta escrever

Cliente.RowSource = "Aba!célula_início:célula_fim"

Pois, no Excel, **referindo-se a uma célula em outra aba** usa-se essa formatação.

Cliente	Status
RECURSUS	Produção
Claro	Finalizado
Telefônica	Aguardando
Motorola	Vistoria
Engemon	
Banco Central	
IHS Tower	
Phoenix Fibra	
Sandin Engenharia	
FIBRASIL	

```
UserForm
Initialize

Private Sub UserForm_Initialize()

    Cliente.RowSource = "'1. Dados do Projeto'!B11:B20"

    ultimalinha = Sheets("1. Dados do Projeto").Range("C10").End(xlDown).Row

    status.RowSource = "'1. Dados do Projeto'!C11:C" & ultimalinha

End Sub
```

Programação

Entretanto, os **status** podem mudar caso sejam **adicionados mais status** abaixo de “**vistoria**”. Portanto, deve-se puxar a **última linha de status** e isso pode ser feito da seguinte forma:

Primeiro, **vai na célula C11** (célula que tem a palavra status) e **depois utiliza Ctrl + seta para baixo**. Isso pode ser feito por meio do comando **End(xlDown)**. Depois disso, **refere-se a linha dessa célula** (por isso **.row**) e guarda isso na variável **ultimalinha**.

Assim, o código do **RowSource** é o mesmo de **Cliente**, porém, nas células, utiliza-se “**C11:C**” & **ultimalinha**, pois a letra **&** no Excel **une dois textos** e **ultimalinha** guarda o número da última linha de status, como definido anteriormente.

UserForm

Initialize

```
Private Sub UserForm_Initialize()  
    Cliente.RowSource = "'1. Dados do Projeto'!B11:B20"  
  
    ultimalinha = Sheets("1. Dados do Projeto").Range("C10").End(xlDown).Row  
  
    status.RowSource = "'1. Dados do Projeto'!C11:C" & ultimalinha  
  
End Sub
```

Inserir Projeto

Projeto	<input type="text"/>	Homem-hora	<input type="text"/>
No PO	<input type="text"/>	Início	<input type="text"/>
Site Type	<input type="text"/>	SLA	<input type="text"/>
Cliente	<div><div></div><div>RECURSUS</div><div>Claro</div><div>Telefônica</div><div>Motorola</div><div>Engemom</div><div>Banco Central</div><div>IHS Tower</div><div>Phoenix Fibra</div></div>	Status	<div><div></div><div></div></div>
Margem de L			
Imposto			

Inserir

Limpar

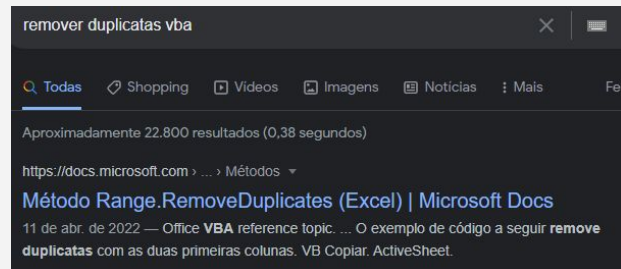
Comandos Importantes

Há alguns **comandos importantes** que em muitos códigos são **utilizados** no **VBA**. Para encontrar algum **comando mais específico** como **“separar texto em colunas”** ou **“remover duplicados”**, há duas opções:

A **primeira opção** é criar uma **macro** que realiza essa **determinada função** no Excel e, depois disso, **abrir o VBA com o código da determinada macro**. No Exemplo abaixo foi feito isso. Agora só basta **interpretar** o código.

Primeiramente, **seleciona-se a primeira casa**, depois **seleciona todas a partir do Ctrl e Seta para baixo (xlDown)** e o terceiro código é realmente o **Remover Duplicatas**. Porém, ao que se pode perceber, para remover duplicatas, deve-se dizer a aba **(ActiveSheet)**, as células em que o comando aparecerá **(J2:J15)** e depois há o **comando** desejado.

```
Sub Macro2 ()  
'  
' Macro2 Macro  
'  
'  
Range("J2").Select  
Range(Selection, Selection.End(xlDown)).Select  
ActiveSheet.Range("$J$2:$J$15").RemoveDuplicates Columns:=1, Header:=xlYes  
End Sub
```



Comandos Importantes

O modo anterior é interessante pois **basta copiar e colar o código no lugar desejado e mudar as células** a serem alteradas. O outro modo, que considero o mais simples, é **procurar na Internet**. Basta procurar **“remover duplicatas vba”** que o **Google** já mostra qual comando deve ser feito e isso serve para quaisquer **comandos**, por isso não é necessário decorar tantas fórmulas. Quando forem **programar, abusen** do **Google!** Algumas fórmulas importantes são (tentem entender a lógica dos códigos):

Copiar > **Range("célula").Copy**

Colar > **Range("célula").PasteSpecial**

Retirar a seleção das células após copiar > **Application.CutCopyMode = False**

Última linha > **ultimalinha = Range("primeira célula").End(xlDown).Row**

Remover duplicatas > **ActiveSheet.Range(células).RemoveDuplicates Columns := 1, Header := _xlNo**

Deslocar x células para baixo e y para direita da selecionada > **Range(célula).Offset(x,y)**

[illegible]