ENGATINHANDO NO GUI DO MATLAB OU COMO APRENDER A CRIAR INTERFACES GRÁFICAS NO MATLAB EM UMA TARDE

20 de março de 2013
v1.1

Marcial Guerra de Medeiros
{marcial.guerra} EM gmail PONTO com

Dedico essa apostila a todos os animais que vivem na UFRN, incluindo alguns colegas e professores.
– O Autor

”Eu falei Windows? Ah..chega me deu um aperto no coração [...]”
–Paulo Sérgio da Motta Pires A.K.A Paulo Motta

Mensagem ao Leitor

Olá, meu nome é Marcial, tudo bom? Espero que sim. Se você está lendo isso é porque quer aprender a criar interfaces gráficas no Matlab, ótimo! Você realmente acertou em cheio ao achar essa minúscula (mas útil!) apostila. Decidi fazer este documento sobre o tenebroso assunto (para muitos) porque sofri na pele a ausência de um material objetivo, conciso e gradativo, ou seja, um material bem passo-a-passo. Claro, tem bons materiais, no entanto, só consegui aprender realmente a usar o GUIDE através de intensas buscas (O Google é meu pastor e nada me faltará; Help do Matlab tem poder, amém). Depois de muitas buscas e "tentativa e erro" terminei minha primeira interface (que eu entendi o que estava fazendo). A alegria de ver aquilo funcionando é indescritível [...] tanto que decidi fazer esta apostila (ou quem sabe, exemplo único?) para que outros possam compartilhar o mesmo sentimento.

O que segue nas próximas páginas são exemplos bem mastigados, isto é, passo-a-passo, da maneira mais didática possível. Esta primeira versão tem apenas um exemplo, mas já serve para dar um "V_0" aos interessados. Cada versão apresenta um novo exemplo que é construído tendo por base o anterior (ao menos, é o que tenho planejado).

Se você leu até aqui, parabéns! Acaba de ler umas 400 palavras que fizeram você aprender absolutamente *nada* sobre o GUIDE.

Hahaha!

Tá, é sério. Pode fazer sua cara de mau, tomar uma água e abrir o Matlab (não necessariamente nessa ordem)!

Marcial Guerra de Medeiros
19/03/2013 - 01:43 AM, em algum canto de parede do meu Quarto-cubículo.

Sumário

1 Exemplo 1: Do nada à senóide **7**
1.1 Breve introdução 7
1.2 Etapa 1: Criando a interface 8
1.2.1 Considerações sobre design 8
1.2.2 Refinando o enunciado 8
1.2.3 Blocos de construção 9
1.3 Etapa 2: Editando o código da GUI 15
1.3.1 Explorando a implementação 15
1.3.2 *Handles*: tornando sonhos realidade 16
1.3.3 Considerações sobre a variável tempo 17
1.3.4 Rascunhando o gráfico do seno 17
1.3.5 Rascunhando a transformada de Fourier 18
1.3.6 O problema da mudança simultânea 18
1.4 Exemplo 1: Concluído 19

Lista de Figuras

1.1 O que aparece quando você digita guide 7
1.2 Ambiente de criação de uma GUI, um arquivo do tipo ".fig" 8
1.3 Introdução de uma janela para *display* de gráficos. 9
1.4 Introdução de *sliders* para permitir alteração de parâmetros. 10
1.5 Introdução de um *static text*, para indicação de nomes, observações, etc. 10
1.6 Editando a *string* de um bloco de texto. 11
1.7 Texto renomeado para "amplitude". 11
1.8 GUI após inserção de todos os textos auxiliares 12
1.9 Alterando as *tags* das janelas de gráficos (axes). 13
1.10 Ajuste de parâmetros do *slider* responsável pela amplitude. 13
1.11 Ajuste de parâmetros do *slider* responsável pela frequência. 14
1.12 Salvando o ".fig". 14
1.13 Salvando o ".fig", com o nome "exemplo1". 15
1.14 Dai-me forças, Santo IGNUcius! 15
1.15 *Handles*, *Handles* pra todos os lados! 16
1.16 Inclusão do tempo como variável externa às funções do tipo "Callback". 17
1.17 Código 18
1.18 Interface Gráfica criada, distinta e louvável! 19

Capítulo 1

Exemplo 1: Do nada à senóide

Enunciado: Deve ser feita uma interface com dois gráficos, um na esquerda e outro na direita. No primeiro deve ser traçado o gráfico duma senóide e no outro, o gráfico da transformada rápida de fourier desta mesma senóide. A senóide deve assumir a forma $A \cdot \sin(\Omega t)$ onde Ω é a frequência angular $(2 \cdot \pi \cdot f)$; e deve ser possível para o usuário alterar de maneira fácil os valores de A e Ω.

1.1 Breve introdução

Antes de confrontar o enunciado, vamos lá, digite "guide" no *prompt* de comando do Matlab, vai aparecer essa tela:

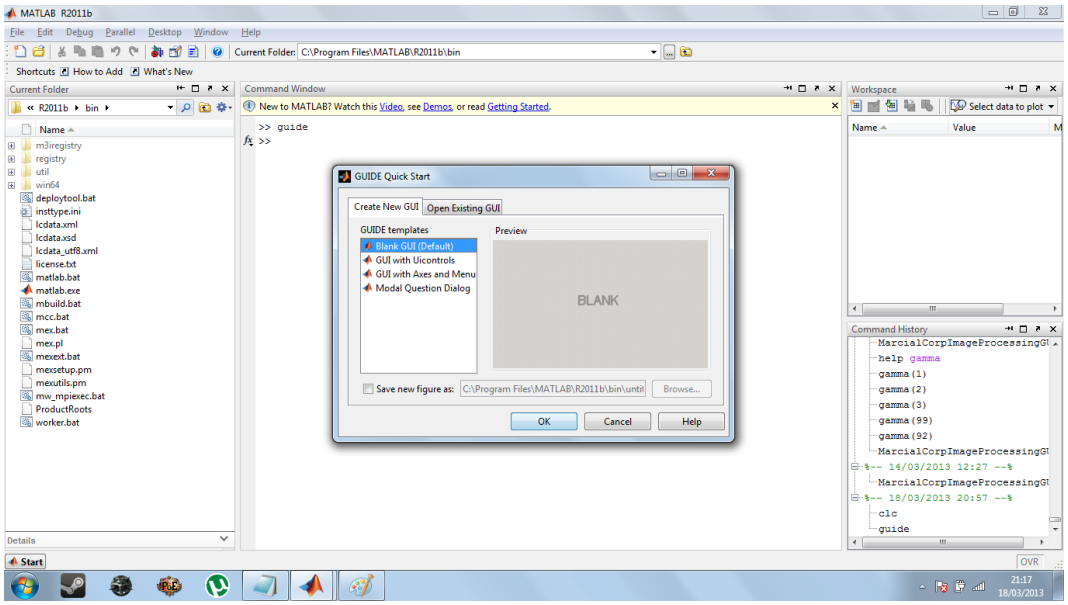

Figura 1.1: O que aparece quando você digita guide

Como pode ver, na Figura 1.2 tem-se duas abas, uma chamada *Create New GUI* (lit.: Criar Nova GUI) e outra de nome *Open Existing GUI* (lit.: Abrir GUI existente). Na primeira tem-se várias opções, vamos usar a primeira, chamada *Blank GUI (Default)* (lit.: GUI em branco (Padrão)), isto quer dizer que é uma interface sem nada.

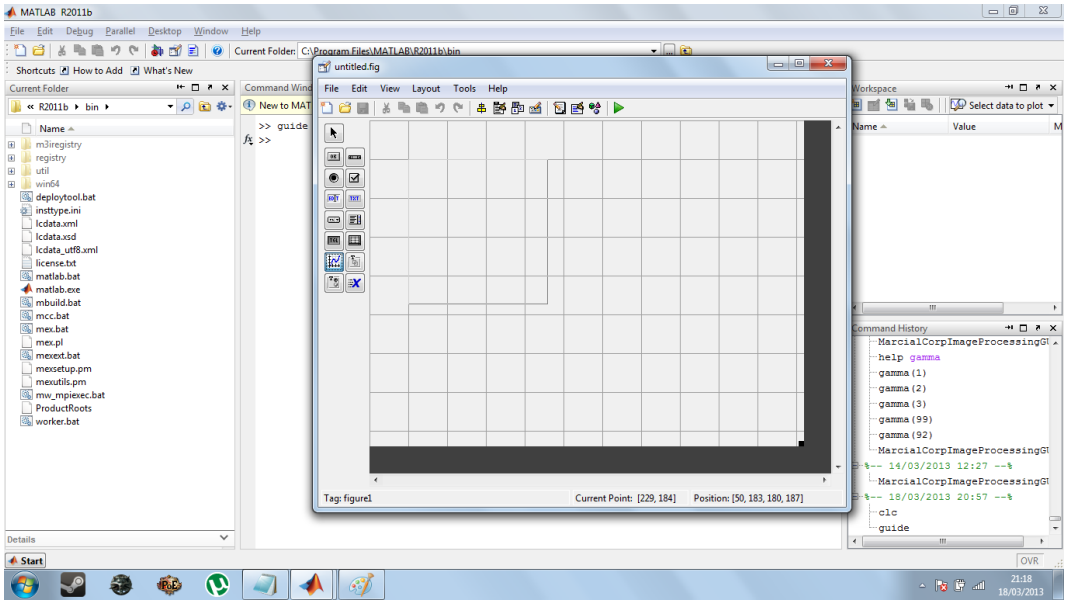

Figura 1.2: Ambiente de criação de uma GUI, um arquivo do tipo ".fig"

Essa tela que você está vendo agora é o coração da interface gráfica, todo o design é feito aqui (existem outros jeitos de fazer, no entanto, estes fogem o escopo deste texto).

A tela tem uma estrutura em grade já para facilitar a organização simétrica dos componentes (ou blocos) da interface. A caixa de ferramentas em vertical ao lado esquerdo, possuí os blocos para criação da interface. Tem *pushbutton*, *sliders* (barra de rolagem, vertical ou horizontal), *axes* (para gráficos), *checkbox*, etc; e todos podem ser inseridos como você bem entender.

A mágica aqui é que ao finalizar o design da sua interface, e salvar esse "untitled.fig" para "algumnome.fig", o Matlab vai gerar o código que implementa a GUI e salvar em "algumnome.m"; e a partir deste arquivo você descreve realmente o que o Matlab faz quando aquele botão é clicado, ou aquela barra é deslizada, etc.

O processo de criação de uma interface gráfica pode ser dividido em duas etapas: O design da GUI (trabalhando em cima do ".fig") e a implementação dos botões, *sliders*, etc (trabalhando em cima do ".m" gerado).

Vamos atacar a **primeira etapa** na proxima seção.

1.2 Etapa 1: Criando a interface

1.2.1 Considerações sobre design

A priori, o design em si, é muito subjetivo, entretanto, é bom ter em mente duas perguntas ao se criar uma interface gráfica:

1. Qual a usabilidade da minha GUI?, isto é, quão facilmente um leigo consegue usá-la?

2. Está a GUI, além de satisfatória, fácil de modificar?

Sobre a primeira, é bom ter em mente que o usuário final pode não ser tão perito no assunto quanto você; e sobre a segunda, vale para qualquer prática de programação, não só matlab ou gui, manter o código inteligível e bem comentado/documentado garante uma reutilização de código mais suave, caso venha a ser necessária no futuro.

1.2.2 Refinando o enunciado

Dito isto, vamos segmentar o enunciado no tocante a esta primeira etapa, deixando a parte interessante para o GUI (neste primeiro momento) em **negrito**:

”Deve ser feita uma interface com **dois gráficos, um na esquerda e outro na direita**. No primeiro deve ser traçado o gráfico duma senóide e no outro, o gráfico da transformada rápida de Fourier desta mesma senóide. A senóide deve assumir a forma $A \cdot \sin(\Omega t)$ onde Ω é a frequência angular $(2 \cdot \pi \cdot f)$ e **deve ser possível para o usuário alterar de maneira fácil os valores de A e Ω**.”

Enfim:

1. Deve ser feita uma interface com dois gráficos, um na esquerda e outro na direita,
2. [...] deve ser possível para o usuário alterar de maneira fácil os valores de A e Ω.

1.2.3 Blocos de construção

Assim sendo, vamos criar uma janela gráfica: clique em Axes (um dos blocos ao lado esquerdo) e crie uma janela para gráficos (formalmente, uma janela para saída de gráficos) do tamanho que achar interessante; você deve ter algo parecido com figura abaixo:

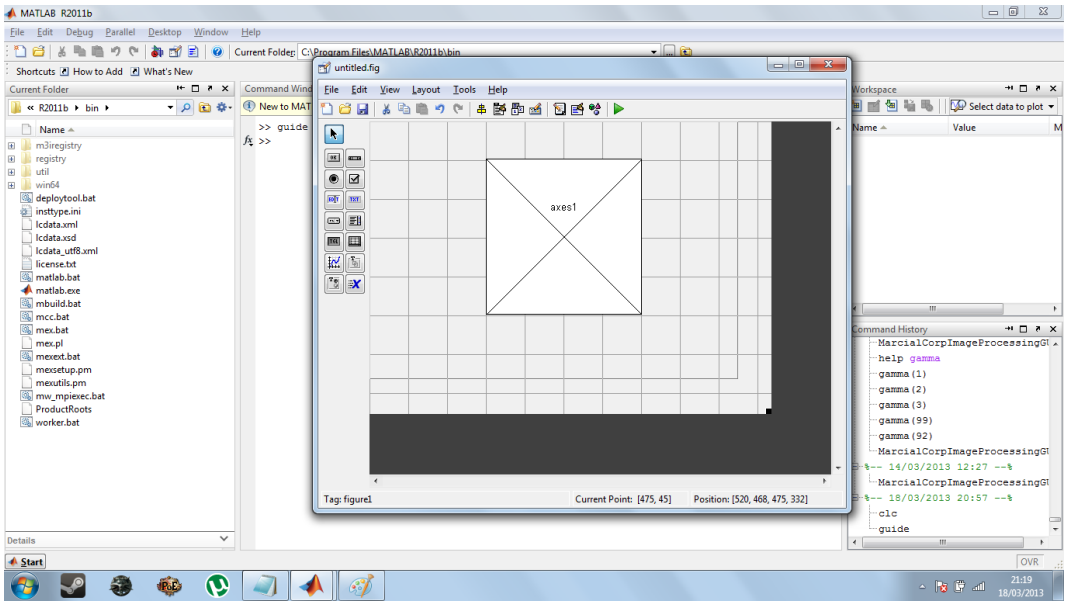

Figura 1.3: Introdução de uma janela para *display* de gráficos.

De mesma forma, use as ferramentas do lado esquerdo para criar duas barras de rolagem (ou *sliders*), que serão responsáveis pela comunicação do usuário com a interface (uma para mudar a amplitude da senóide e outra para mudar a frequencia angular (Ω). Você deve obter algo semelhante a essa imagem:

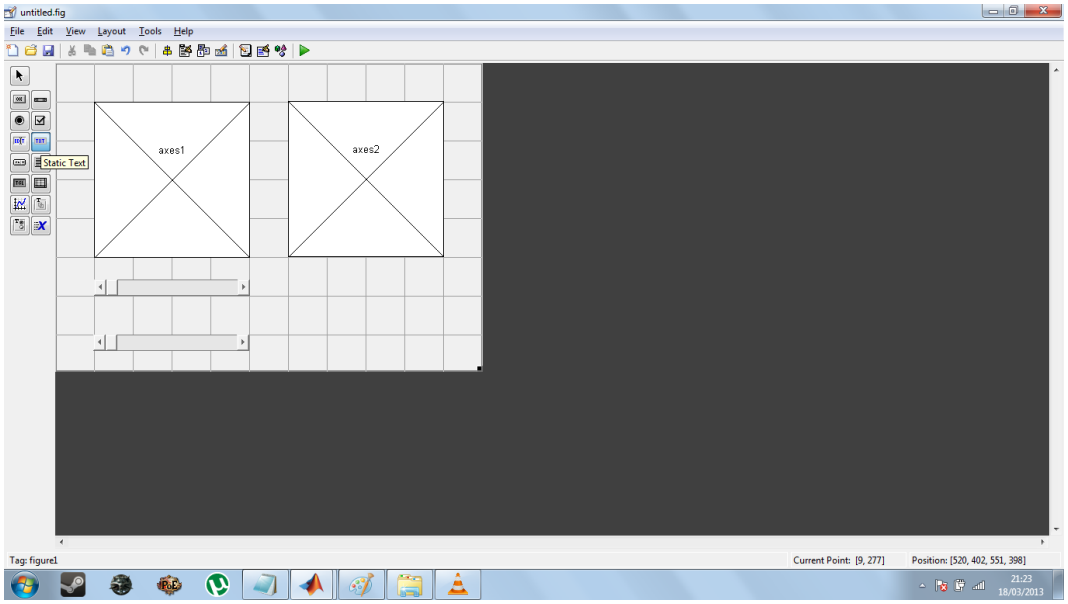

Figura 1.4: Introdução de *sliders* para permitir alteração de parâmetros.

Consegue ver o *Static Text* (lit.: texto estático)? Esse nome não é à toa, ele serve justamente para isso: ficar paradão! Falando sério, serve para colocar-se títulos, observações, etc. No nosso caso, é interessante deixar bem claro para o usuário qual barra de rolagem altera qual parâmetro; então vamos fazer isso e dar nome aos bois.

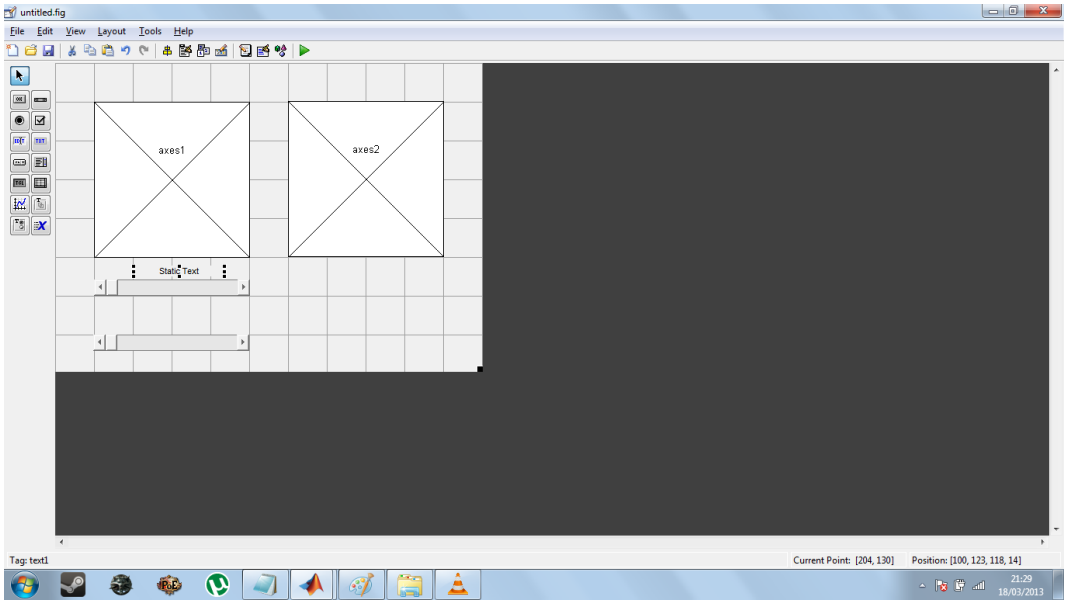

Figura 1.5: Introdução de um *static text*, para indicação de nomes, observações, etc.

De modo a mudar o nome de *static text*, deve-se alterar o parâmetro *string* do bloco. Para fazer isto, basta clicar duas vezes com o botão esquerdo em cima do bloco *static text* ou clicar com o botão direito e selecionar *Property-inspector* na caixa de opções. Tendo feito isto, é fácil mudar o nome:

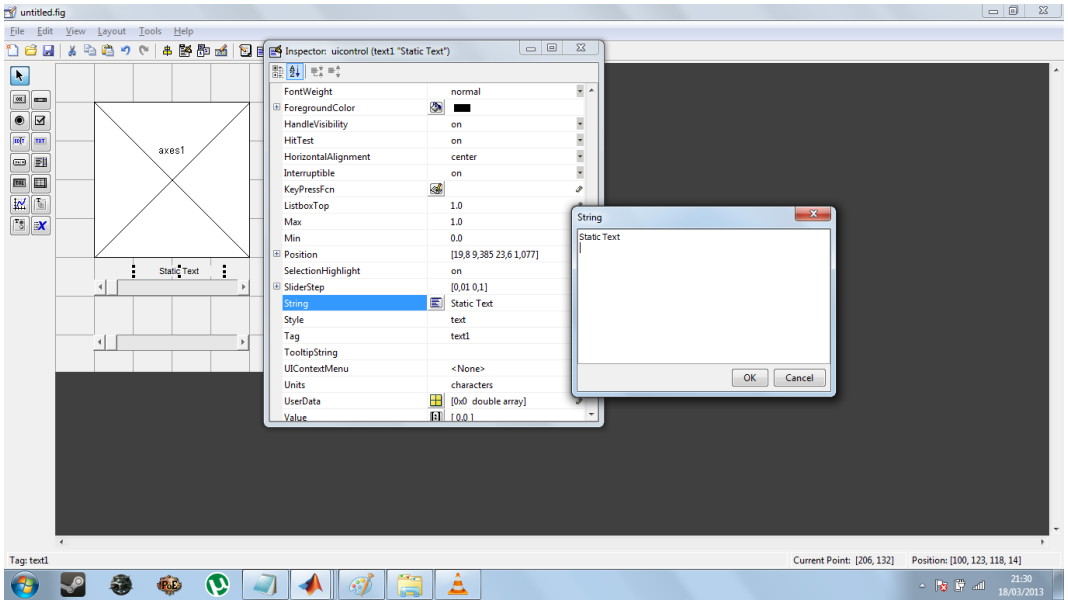

Figura 1.6: Editando a *string* de um bloco de texto.

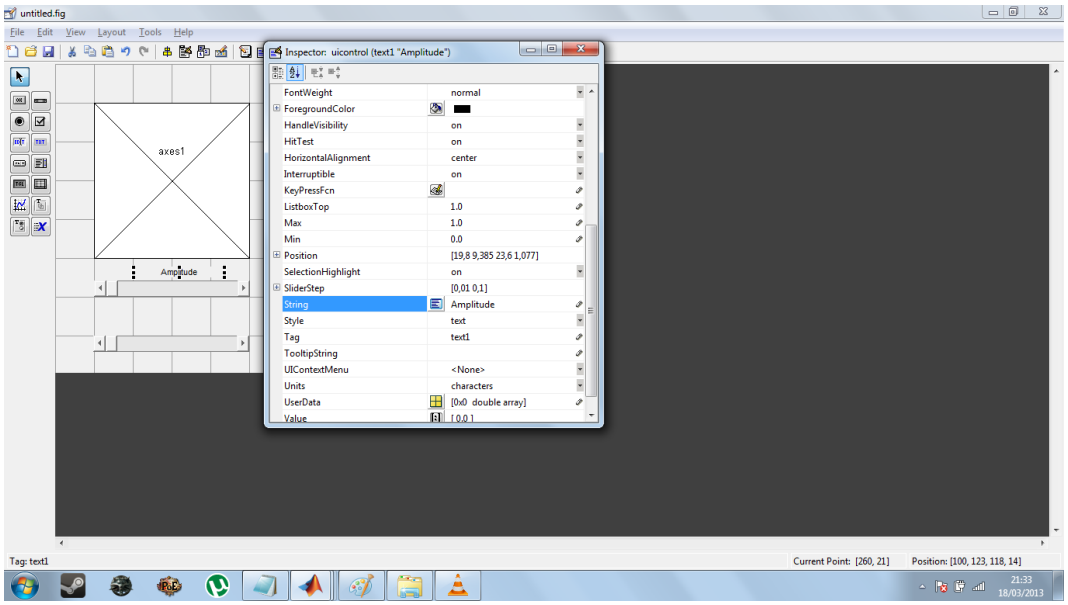

Figura 1.7: Texto renomeado para ”amplitude”.

Note que o tamanho da letra pode ser alterado no parâmetro ”font”.

Nossa GUI então fica com a seguinte cara:

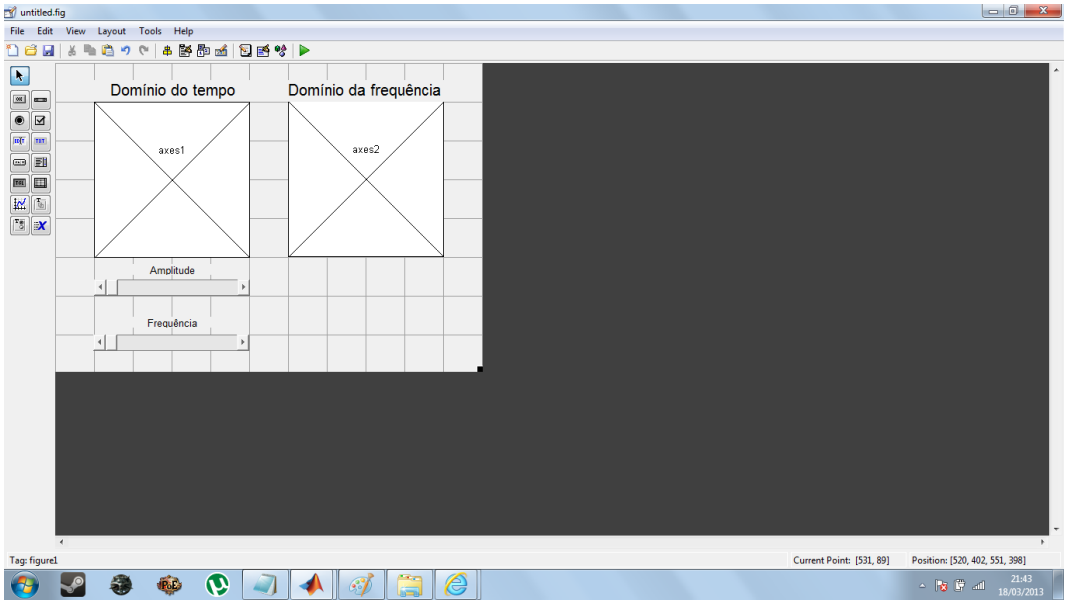

Figura 1.8: GUI após inserção de todos os textos auxiliares

"PARA TUDO!! Domínio do tempo, domínio da frequência?! Não tem nada passo-a-passo aqui! Buaaaaaa..." Calma! Como dito, só estou colocando os nomes em lugares interessantes baseado no enunciado do nosso exemplo. Esta etapa poderia acabar aqui mas por completude vamos agora explorar 4 parâmetros de muitíssima utilidade, são eles: *tag*, *max*, *min*, *SliderStep*. Vamos definir o que eles fazem:

1. *tag*: é um marcador para o bloco, pense nele como o "nome" do bloco.
2. *max*: valor máximo do bloco.
3. *min*: valor máximo do bloco.
4. *SliderStep*: Passo de incremento do *slider* (barra de rolagem).

Existem duas formas de alterar estes parâmetros, direto no código (que será gerado pelo ".fig") ou pelo *Property-inspector* do ".fig"; vamos alterar pelo segundo. Ao abrir as propriedades para as janelas de gráficos de nome axes1 e axes2, é possível mudar suas *tags* para um nome mais próximo da sua tarefa no problema. Neste caso, é apropriado mudar o *tag* de axes1 (um nome que não diz muito) para dominio_tempo; e o mesmo se faz para o axes2:

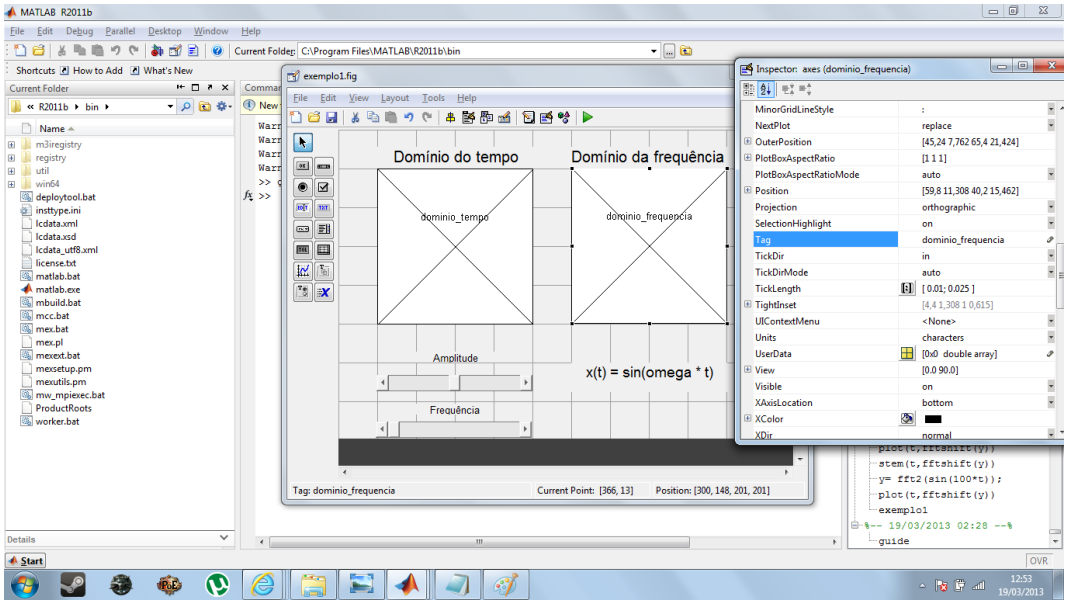

Figura 1.9: Alterando as *tags* das janelas de gráficos (axes).

O mesmo pode ser então feito para os *sliders*. Para o *slider* responsável pela amplitude, modificamos sua *tag* para um nome mais próximo da sua tarefa, neste caso, ”amplitude” e definimos um máximo e um mínimo:

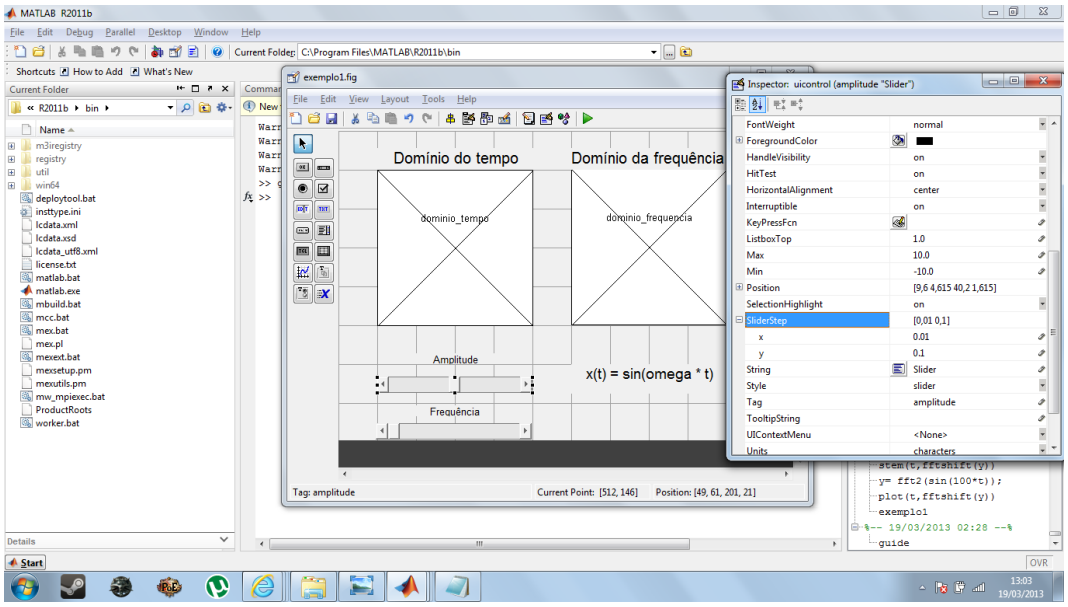

Figura 1.10: Ajuste de parâmetros do *slider* responsável pela amplitude.

Os parâmetros *tag*, *max* e *min* são bem intuitivos. O *SliderStep* também seria, se não dependesse de duas variáveis x e y. Como pode ser visto na Figura 1.10. Essas variáveis do *SliderStep* funcionam da seguinte maneira:

1. O x indica a porcentagem do valor total que a barra deve se deslocar quando se pressiona alguma das setas.

2. O y indica a porcentagem do valor total que a barra deve se deslocar quando se clica em um ponto qualquer da barra.

Ajustando os parâmetros para o *slider* responsável pela frêquencia:

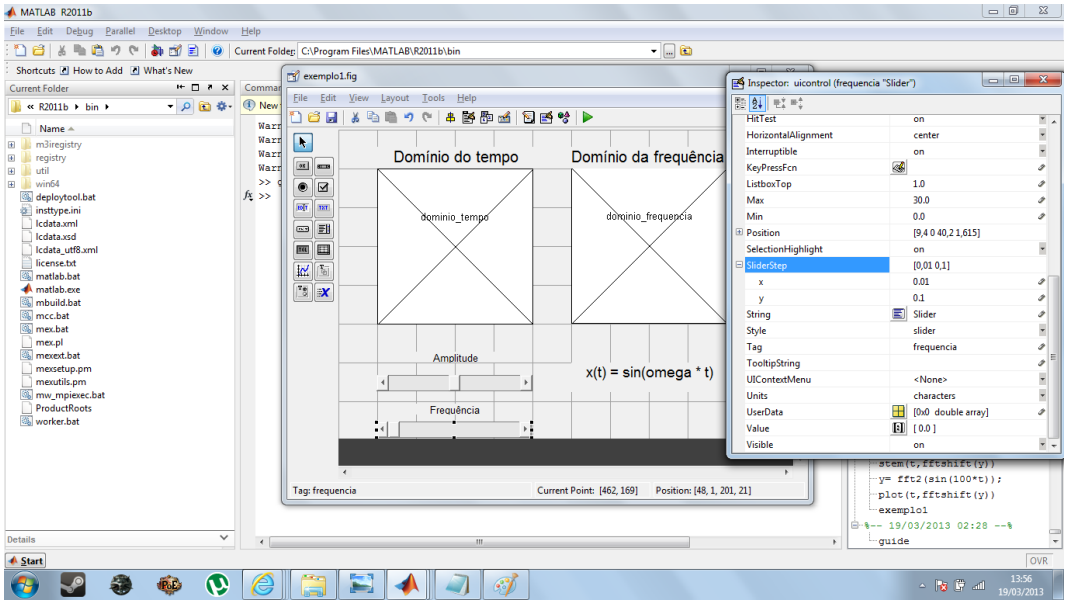

Figura 1.11: Ajuste de parâmetros do *slider* responsável pela frequência.

Podemos finalmente salvar a GUI:

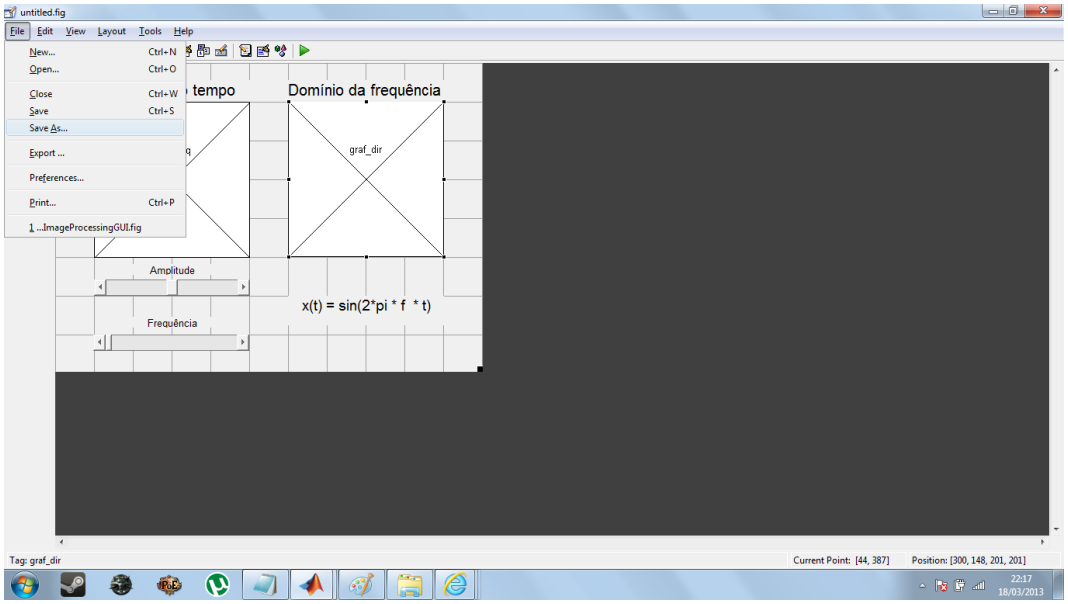

Figura 1.12: Salvando o ".fig".

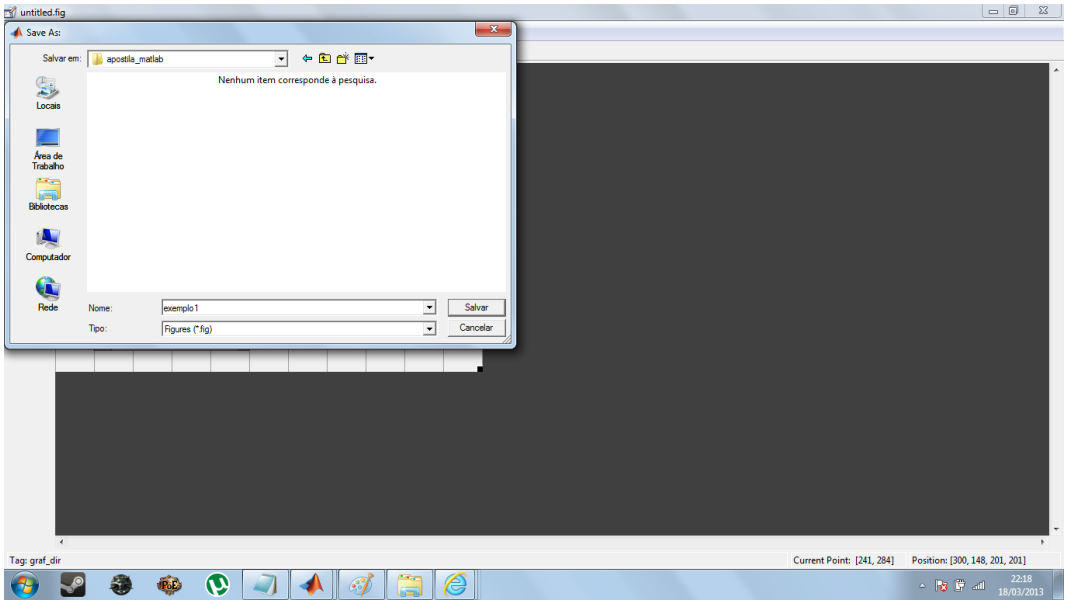

Figura 1.13: Salvando o ".fig", com o nome "exemplo1".

1.3 Etapa 2: Editando o código da GUI

Então você salvou lá o exemplo1.fig e aparece essa tela:

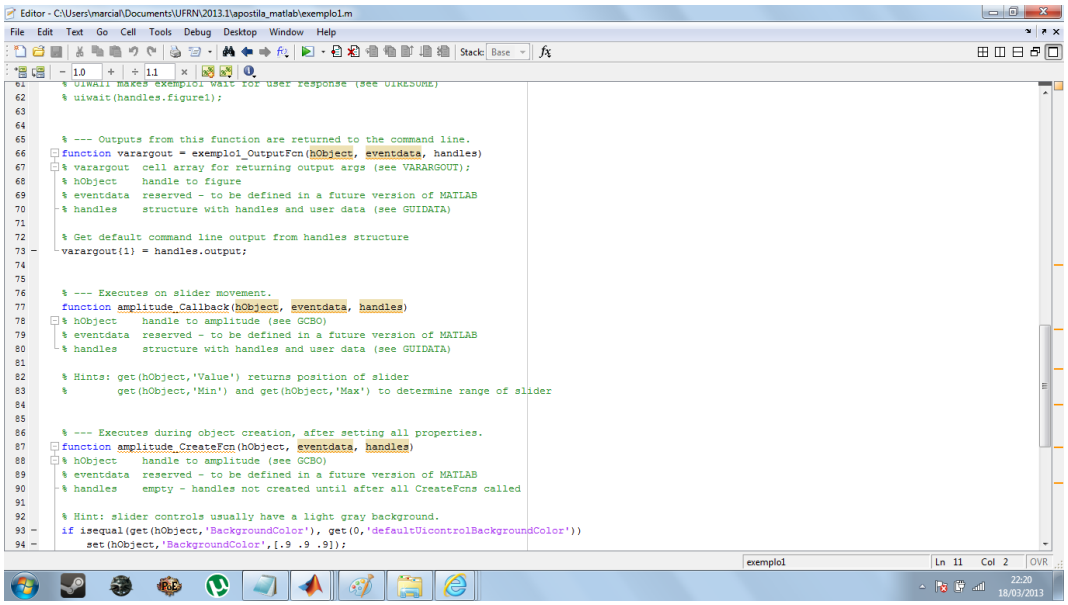

Figura 1.14: Dai-me forças, Santo IGNUcius!

Segure suas calças! Isso não é nada mais que a implementação da bela interface que você acabou de fazer no .fig! "Muito bem, Marcial, mas e aí começo por onde?" Ora, do começo, Alice!

1.3.1 Explorando a implementação

Vamos primeiro observar o arquivo ".m" gerado, podemos verificar que este arquivo consiste de diversas funções. O arquivo é meio grande, cheio de comentários que você provavelmente não entende e parâmetros estranhos (hObject, handles...). Existem funções que terminam em "*callback*", essas são as funções chamadas quando existe uma alteração no bloco, isto é, quando o usuário "clica" no mesmo. Se você tiver mudado as *tags* dos *sliders* de acordo com a etapa 1 do exemplo, você deverá ter: "amplitude_Callback" e "frequencia_Callback".

Está enrolado? Pense assim, quando você quer que ocorra alguma mudança nos gráficos? Justamente quando o usuário mudar os valores de amplitude e frequência correto? Então é simples, todas as suas energias devem estar concentradas no que o programa deve fazer quando "clicarem" ou na barra de frequência ou na barra de amplitude! Ora, muito fácil agora, basta colocar a fórmula do seno e pá!

"Mas peraí Marcial, como faço essa ligação entre barra de rolagem e uma variável do programa? e como as variáveis de uma função podem ser usadas em outra sem serem variáveis globais?"

Ótimas perguntas, para fazer isso, deve-se usar variáveis com "*Handles*"; vamos entender o conceito de *handles* e como funcionam na proxima subseção.

1.3.2 *Handles*: tornando sonhos realidade

Figura 1.15: *Handles*, *Handles* pra todos os lados!

Variáveis na forma handles.nomedavariavel são variáveis especiais que podem ser acessadas por qualquer função do programa (mesmo estando fora do escopo da função). Ao criar o ".m" baseado no ".fig", o Matlab gera automaticamente algumas destas variáveis baseado nos blocos escolhidos, por exemplo: um slider tem como variável handles.tag_que_você_deu_para_o_slider e pode-se extrair o seu valor com a função *get*, desta forma: "get(handles.tag_que_você_deu_para_o_slider, 'Value')".

Vê onde quero chegar? se você pode extrair o valor de um slider (e também, de qualquer outro bloco!) usando funções, você pode armazenar esse valor em uma variável e utilizá-la em outra parte do programa! Três observações devem ser feitas agora:

1. Se lembra que os parâmetros Max, Min e SliderStep podem ser alterados no código? pois bem, basta utilizar a função "set"; e se estiver realmente curioso, use o Help!

2. Os gráficos funcionam de maneira um pouco diferente, você apenas declara qual a janela que vai utilizar e a partir deste comando: axes(handles.tag_da_janela), tudo que vier depois envolvendo desenho de gráficos (plot) vai sair na janela cuja tag foi utilizada no handle.

3. Para utilizar uma variável na forma handles.nome_da_variavel, que não seja uma criada pelo Matlab automaticamente, você além de criá-la, deve exportá-la para as outras funções poderem "vê-la". Para tal tarefa, usa-se guidata(handles.nome_da_variavel_que_voce_quer_exportar) e voilá! Agora você pode utilizá-la em outra função!

Agora que já falamos de *handles* vamos falar da texpix a melhor câmera do Bras– ops..

1.3.3 Considerações sobre a variável tempo

Como usar a variável tempo em duas funções (”amplitude_Callback”, ”frequencia_Callback”) simultaneamente? Se você chutou ”*handles*!”, acertou! O problema aqui é: **o tempo não pode estar definido dentro de nenhuma das duas funções de ”callback”**, porque ao fazer isto, estaríamos, a cada clique, calculando o tempo novamente e se você quiser usar um intervalo de tempo grande o programa teria uma execução muito lenta. A solução é criar o tempo em outra função, uma que sirva apenas como inicialização, assim, só seria feito uma atribuição para a variável de tempo; e utilizando a mágica dos *handles*, podemos usar tal variável em qualquer outra função do programa. Vamos criar esta variável na função *OpeningFnc*, como mostrado na figura abaixo:

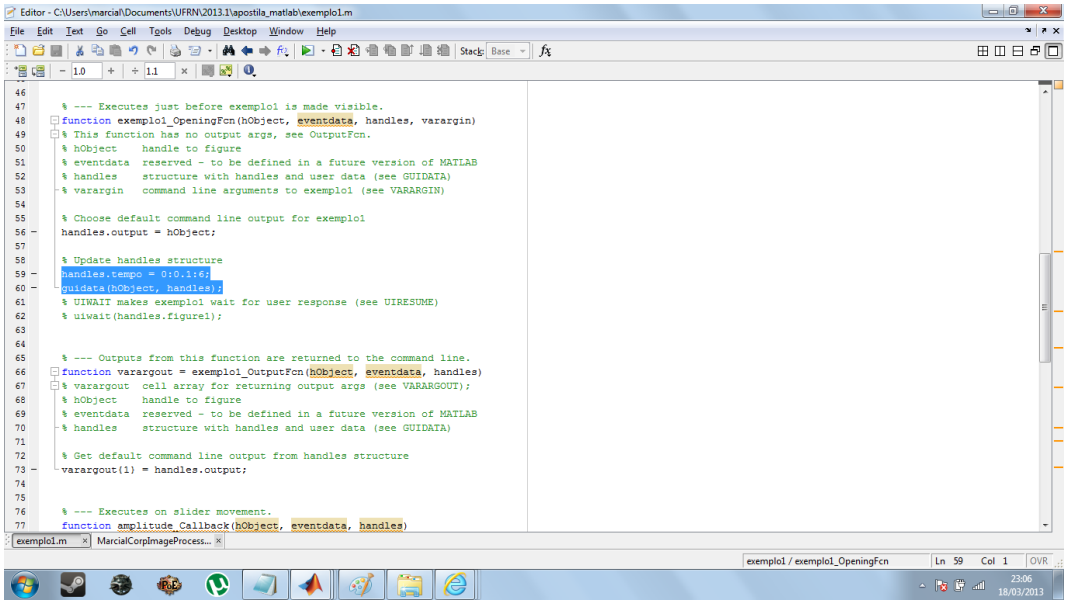

Figura 1.16: Inclusão do tempo como variável externa às funções do tipo ”Callback”.

Note que o comando *guidata* não serve apenas para exportar uma variável para o resto do programa, na figura é utilizado guidata(hObject, handles) que serve para exportar os *handles* (todos) que estiverem naquele escopo, neste caso tem-se apenas a variável handles.tempo.

1.3.4 Rascunhando o gráfico do seno

Como queremos o gráfico do seno na janela da esquerda quando um dos *sliders* seja deslocado, devemos escrever algum código em alguma das funções *Callback* dos *sliders*. Digamos, que a escolhida seja a ”amplitude_Callback”:

1. Começamos o código com o comando ”axes(handles.dominio_tempo)” que numa linguagem mais humana: ”As saídas gráficas a partir daqui vão para a janela com a tag dominio_tempo”;

2. Então, atribuimos o valor do seno a uma variável qualquer com o comando ”y = get(handles.amplitude,’Value’).*sin(get (handles.frequencia), ’Value’) * handles.tempo);” que significa: ”Atribua a y o valor dado pela multiplicação do valor do slider amplitude vezes o seno do valor dado pelo slider de frequência vezes o tempo.”;

3. Finalmente, faz-se o gráfico com o simples ”plot(handles.tempo,y)”.

1.3.5 Rascunhando a transformada de Fourier

De mesma forma que na subseção anterior, faz-se a FFT. Se estiver curioso quanto ao código, dá um *zoom* na Figura 1.17 e veja!

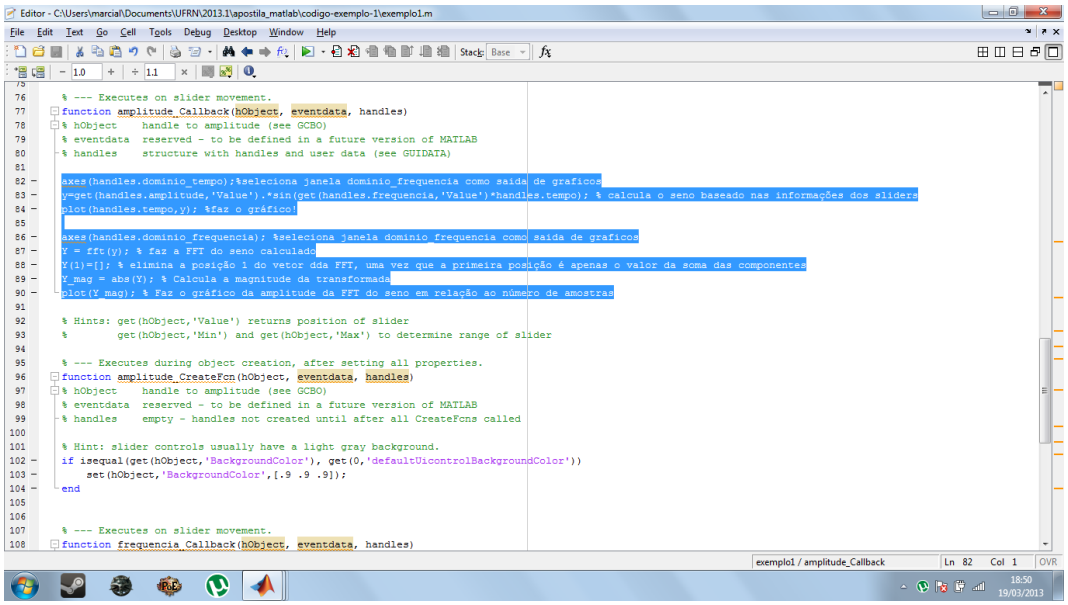

Figura 1.17: Código

1.3.6 O problema da mudança simultânea

Como agora fazer o código da "frequencia_Callback" ? Muito simples, ele é exatamente igual ao código da "amplitude_Callback"! Isto se deve ao fato de que não existem operações que uma barra irá fazer e a outra não, mantendo a frequência parada, queremos ver o que a mudança de amplitude traz ou vice-versa e para tanto, ambas funções vão fazer a mesma coisa quando seus respectivos blocos forem "clicados".

Se você não entendeu, pense: o que aconteceria se você não escrevesse nada na "frequencia_Callback"? teste você mesmo e veja o que acontece!

1.4 Exemplo 1: Concluído

Eis a interface gráfica gerada!

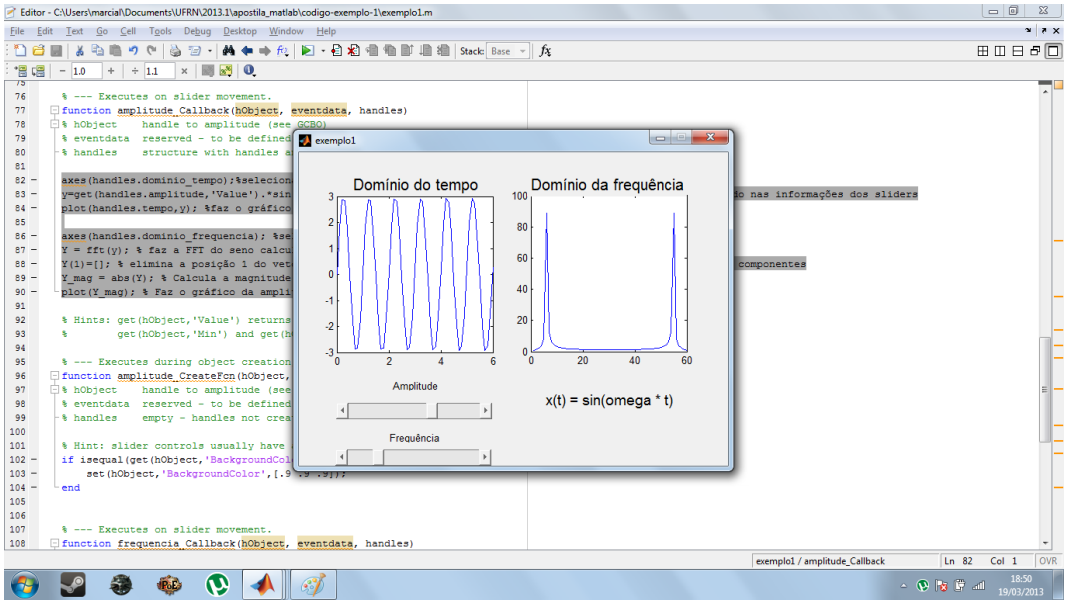

Figura 1.18: Interface Gráfica criada, distinta e louvável!

Nota final

Espero que esta apostila tenha sido útil para você, porque me deu um trabalho desgraçado de fazer! Agradeço críticas, sugestões e elogios; não que eu aceite, mas agradeço por qualquer contribuição para este ou outro trabalho meu. Se for ganhar algum dinheiro em cima disso, peço que por favor me dê uma porcentagem (ou um doce!), sabe como é ... estudante!

Só peço por favor que não enviem dúvidas poque não sei se terei tempo de responder e não quero que isso passe como grosseria da minha parte.

Tudo de bom, nos vemos por aí!