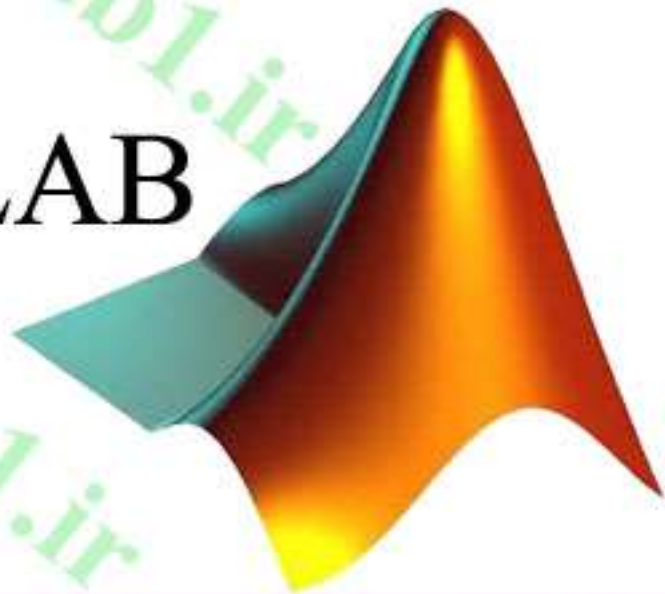


دوره جامع آموزش

برنامه نویسی

متلب

MATLAB



گروه برنامه نویسی ایران متلب MATLAB1
از حرفه ای ها متلب را یاد بگیرید

۱- رابط گرافیکی برای کاربر (GUI)

GUI (رابط گرافیکی برای کاربر) نوعی رابط تصویری برای برنامه است که نمونه خوب

آنمیتواند با فراهم کردن شکل و صورتی ثابت برای برنامه و همچنین با کنترلگرهای آشنا،

مثل

دکمه های فشاری)، list boxes (جعبه های لیست) و sliders و menus

pushbuttons

Graphical User Interface (GUI)

(منوها) و مانند اینها استفاده از برنامه را آسانتر کند. رابط گرافیکی باید رفتاری قابل فهم و پیشبینیداشته باشد، بدین معنی که کاربر بداند در ازای انجام عملی خاص، چه اتفاقی خواهد افتاد. برای مثال، هنگامی که ماوس روی یک pushbutton کلیک میکند، GUI باید عملی را که روی آن نوشته شده، آغاز کند. این فصل به معرفی عناصر اصلی رابطهای گرافیکی MATLAB اختصاص دارد. با اینکه این فصل حاوی توضیحات کاملی درباره خصوصیات همه اجزای رابطهای گرافیکی نیست، ولی اصول کلی لازم برای ایجاد GUI های کاربردی برای برنامه های کاربران، در آن گنجانیده شده است.

1-1 یک GUI چگونه کار می کند؟

رابط گرافیکی (GUI) محیطی آشنا برای کاربر فراهم می کند. این محیط حاوی pushbutton ها، togglebutton ها، list ها، menu ها، text box ها، و ... می باشد که برای همه کاربران آشناست و این موجب میشود که کاربر به جای مشغول کردن ذهن خود با چند و چون اجرای برنامه و پیچیدگی آن، تنها روی استفاده از آن تمرکز کند. ایجاد رابطهای گرافیکی برای برنامه نویس کار مشکلی است. زیرا برنامههای که بر پایه GUI طراحی شده باید در هر زمان آماده ورودیهای ماوس و (یا احتماً ورودیهای کیبرد) روی هر یک از عناصر خود باشد. این ورودیها به event ها معروفند. برنامه ای که به این event ها پاسخ گوید، event driven نامیده می شود. سه عنصر اساسی لازم برای ایجاد رابط گرافیکی (GUI) MATLAB عبارتند از:

اجزا (Components) - 1

عناصر درون GUI (pushbutton ها، label ها، editbox ها) اجزای گرافیکی نام دارند. انواع این اجزا شامل کنترلهای گرافیکی، (مانند pushbutton ها، editbox ها، list ها، slider ها و . . .) عناصر ثابت و بدون تغییر (مانند قابها و نوشتهها)، منوها و محورهای

Graphical User Interface (GUI)

مختصات هستند. کنترل‌های گرافیکی و عناصر ثابت توسط تابع `uncontenxmenu` به وجود می‌آیند. در نهایت، محورهای مختصات که وظیفه نمایش داده‌های گرافیکی را بر عهده دارند، توسط تابع `axes` به وجود می‌آیند.

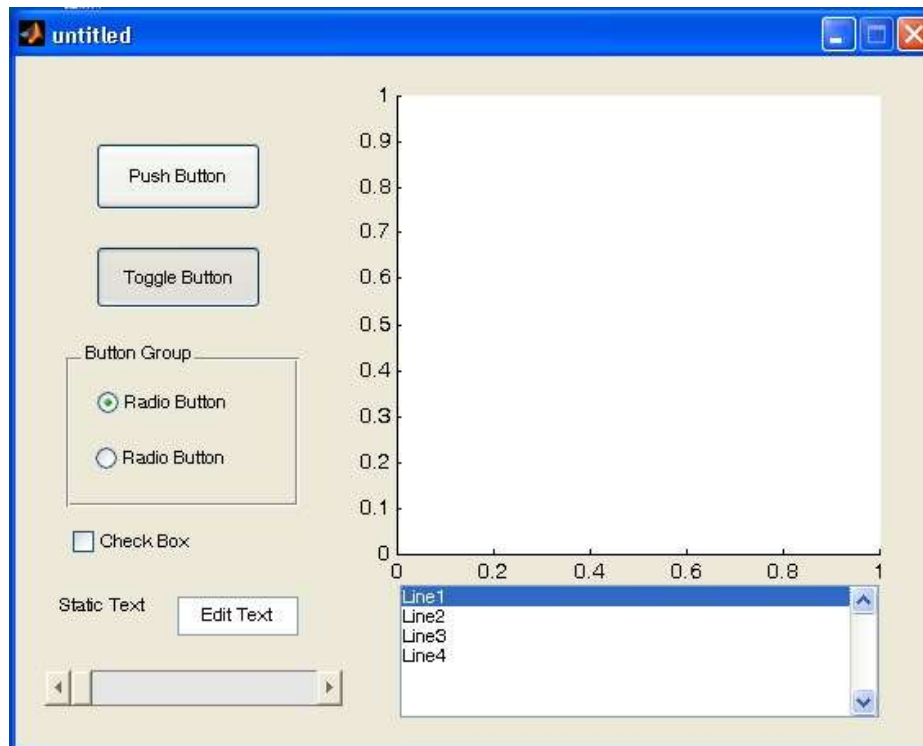
2- اشکال (Figures)

اجزای GUI باید درون یک `figure` مرتب شوند، که پنجره‌های روی صفحه کامپیوتر است. پیش از این `figure` ها به طور خودکار هنگام ترسیم داده‌ها بوجود می‌آمدند. با این وجود، `figure` های خالی را نیز میتوان با دستور `figure` ایجاد کرد و از آنها میتوان برای نگهداری و کنار هم گذاشتن اجزای گرافیکی استفاده کرد.

3- فراخوان ها (Callbacks)

باید راهی برای انجام عملی خاص هنگامی که کاربر با ماوس روی یک دکمه کلیک یا اطلاعاتی را توسط کیبرد تایپ میکند، وجود داشته باشد. هر کلیک ماوس یا فشار کلید از صفحه کلید یک `event` تلقی میشود و برنامه `MATLAB` باید با اجرای تابع مربوطه، به این `event` پاسخ گوید. به عنوان مثال، اگر کاربر روی یک دکمه کلیک کند، این پیش آمد باید سبب اجرای کد مربوط به `function` آن دکمه شود. کد اجرا شده در پاسخ به این پیش آمد، `callback` نام دارد. در حقیقت باید برای عملکرد هر جزء گرافیکی GUI به `callback` وجود داشته باشد. عناصر اصلی GUI ها در زیر به خلاصه و نمونه‌هایی از آنها در شکل 1-1 نشان داده شده است. در ادامه مثالهایی از این عناصر را مطالعه کرده و سپس با استفاده از آنها به ایجاد GUI های کاربردی خواهیم پرداخت.

Graphical User Interface (GUI)



شکل 1-1 یک پنجره Figure نشان دهنده مثالهایی از عناصر GUI. از بالا به پایین و از چپ به راست، عناصر عبارتند از: دکمه فشاری (pushbutton)، یک toggle button در وضعیت "روشن"، دو radio button درون یک قاب، یک check box، یک text field و یک edit box، یک slider، محور مختصات، یک list box.

مشخصات بعضی از عناصر اصلی GUI:

Pushbutton: (uicontrol) این جزء گرافیکی کار یک دکمه فشاری را انجام میدهد.

هنگامی که با ماوس روی آن کلیک شود، callback مربوطه را فعال می کند.

Graphical User Interface (GUI)

Toggle button : (uicontrol) : جزئی گرافیکی است که کار یک کلید دو حالتی

را انجام میدهد. این کلید دو وضعیت یا "روشن" است یا "خاموش" و هر بار که با ماوس روی آن کلیک شود، تغییر وضعیت داده و callback مربوط به آن فعال می شود.

Radio button : (uicontrol) : نوعی از toggle button است که به

صورت دایره کوچکی است و هنگام "روشن" بودن نقطه ای در مرکز آن قرار می گیرد. گروهی از radio button ها را میتوان برای پیاده سازی گزینه های مستقل استفاده کرد. هر کلیک ماوس روی این جزء callback آن را فعال می کند.

Check box : (uicontrol) : نوعی از toggle button است که

به شکل مربعی کوچک با علامت تیک (☑) در درون آن به منزله "روشن" بودن میباشد. هر کلیک ماوس روی آن، callback آن را فعال می کند.

Edit box : (uicontrol) : edit box متنی را نمایش می دهد و به کاربر

اجازه میدهد اطلاعات نشان داده شده در آن را تغییر دهد. Callback مربوط به آن با فشار دکمه enter فعال می شود.

List box : (uicontrol) : کنترلی گرافیکی است که یک سری از متن های رشته ای

(text string) را نمایش می دهد کاربر می تواند با یک یا دو بار کلیک روی هر یک از این متن های رشته ای آنها را انتخاب کند. به هنگام انتخاب یک متن رشته ای callback آن فعال می شود.

Graphical User Interface (GUI)

PopupMenu : (uicontrol) : کنترلی گرافیکی است که در پاسخ به کلیک ماوس

یکدسته از متنهای رشتهای را نمایش میدهد. تا هنگامی که روی یک منوی popup کلیک نشده است، تنها رشته انتخاب شده فعلی آن قابل مشاهده است.

Slider : (uicontrol) : slider کنترل گرافیکی دیگری است که نقش آن تنظیم

یکمقدار به طور منظم و پیوسته با کشیدن کنترل آن به وسیله ماوس است. هر تغییر در slider، callback، اش را فعال می کند.

Frame : (uicontrol) : یک قاب ایجاد میکند که در حقیقت جعبه مربعی شکل

درون figure میباشد. قابها برای گروه بندی مجموعههای از کنترلهای گرافیکی استفاده میشوند. قابها هرگز callbackی را فعال نمی کنند.

Text field : (uicontrol) : برچسبی (label) ایجاد می کند که متنی رشته ای

واقع در نقطهای روی figure است. text field ها هرگز callbackی را فعال نمیکنند.

Menu items : (uimenu) : یک منو ایجاد می نماید و callback این منوها به

هنگامکلیک ماوس روی آنها فعال می شود.

Context menu : (uicontextmenu) : یک منوی ایجاد می نماید.

هنگامی که کاربر روی شیء مورد نظر با کلیک سمت راست ماوس، کلیک می کند این منو ظاهر می شود.

Axes : (axes) : یک دستگاه مختصات برای نمایش اطلاعات در آن، ایجاد میکند.

هاهیچ گاه callbackی را فعال نمی کنند.

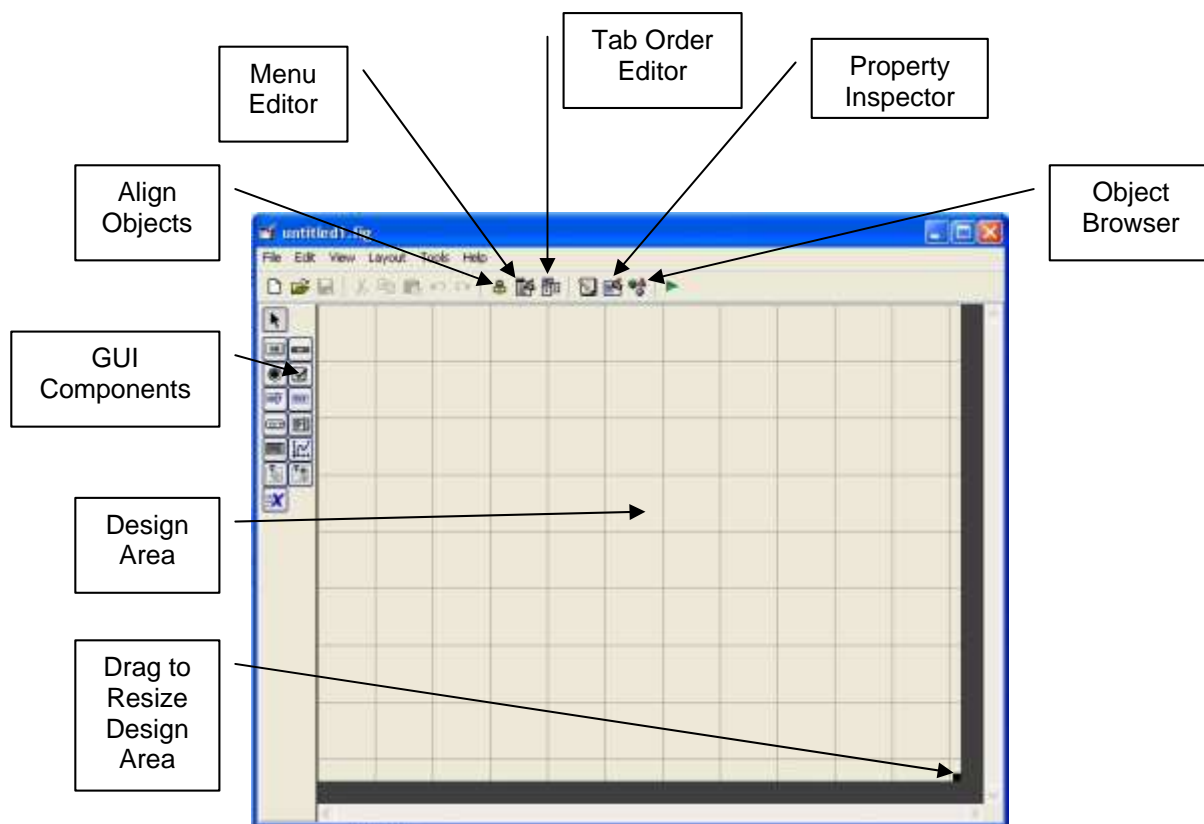
Graphical User Interface (GUI)

1-2 ایجاد و نمایش یک GUI های MATLAB را میتوان با ابزاری به نام `guide`، ایجاد کرد. این ابزار به برنامه نویس امکان پیاده‌سازی GUI، انتخاب و مرتب کردن اجزای درون آن را میدهد. بعد از اینکه اجزا در جاهایشان قرار گرفتند، برنامه نویس میتواند خصوصیات (`properties`) هر یک را اعم از اسم، رنگ، اندازه، فونت، و نوشته روی آن و... ویرایش کند. هنگامی که `guide` یک GUI را ذخیره می‌کند، برنامه‌های حاوی مجموعه‌های از توابع کلیدی ایجاد میکند که برنامه نویس میتواند با تغییر در این توابع رفتار GUI را تنظیم کند.

وقتی `guide` اجرا می‌شود، `Layout editor` نشان داده شده در شکل 1-2 نیز به

همراه آن ظاهر می‌شود.

Graphical User Interface (GUI)



شکل 1-2 پنجره ابزار guide

ناحیه روشن چهار خانه، layout (کادر و ناحیه طراحی) نام دارد، ناحیه ای که برنامه نویسی، GUI را در آن طراحی می کند. در قسمت چپ پنجره layout editor، مجموعه ای از عناصر GUI قرار دارند. کاربر میتواند هر تعداد از این اجزا را ابتدا با کلیک روی جزء مورد نظر و سپس کشیدن آن به درون ناحیه layout ایجاد کند. بالای این پنجره یک toolbar حاوی یک سری از ابزارهای مفید وجود دارد که به کاربر اجازه میدهد تا اجزای GUI را هم راستا کرده یا روی ناحیه طراحی پخش کند و یا خصوصیات (propertise) این اجزا را تغییر داده و یا به GUI منو اضافه کند یا ...

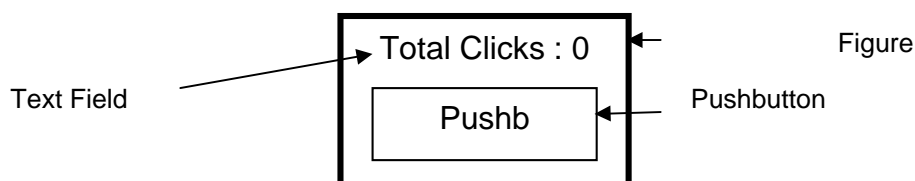
Graphical User Interface (GUI)

گام های اساسی لازم برای ایجاد یک GUI در MATLAB به قرار زیر است:

- ۱- ابتدا باید تصمیم بگیرید که به چه عناصری برای کارتان احتیاج دارید و نقش هر یک را تعیین کنید، سپس طراحی اولیه و درهم برهمی از این اجزا با دست روی کاغذ بیاورید.
- ۲- از ابزار `guide` (محیط توسط یافته GUI) برای چیدن اجزا درون `figure` کمک بگیرید.
- ابعاد و اندازه `figure`، هم راستایی و فضای بین اجزا را میتوان با ابزارهای درون `guide` تنظیم کرد.
- ۳- از یکی دیگر از ابزارهای MATLAB به نام `Property Inspector` (واقع درون `guide`) استفاده کنید تا به هر کدام از اجزا، یک لقب (یک `tag`) نسبت دهید و ویژگیهای هر یک را که شامل رنگ، متن نمایش داده شده، غیره می باشد، تنظیم نمائید.
- ۴- `figure` را در یک فایل ذخیره کنید. بعد از اینکه `figure` را ذخیره کردید، دو فایل با اسامی یکسان ولی با پسوندهای متفاوت روی دیسکت بوجود میآیند. فایل با پسوند `.fig` خود GUI های ایجادشده، و فایل دیگر `M-File` میباشد، که حاوی کد آن و بدنه `callback` های مربوط به عناصر GUI است.
- ۵- کدی بنویسید که رفتار مربوط به هر تابع `callback` را انجام می دهد.
- به عنوان نمونه برای این مراحل، بیائید یک GUI ساده را در نظر بگیریم که حاوی یک `pushbutton` ساده و یک متن رشتهای میباشد. با هر بار کلیک روی `pushbutton` متن رشتهای طوری تغییر میکند تا تعداد کل دفعاتی که از ابتدای کار GUI روی `pushbutton` کلیک شده استرا نمایش دهد.

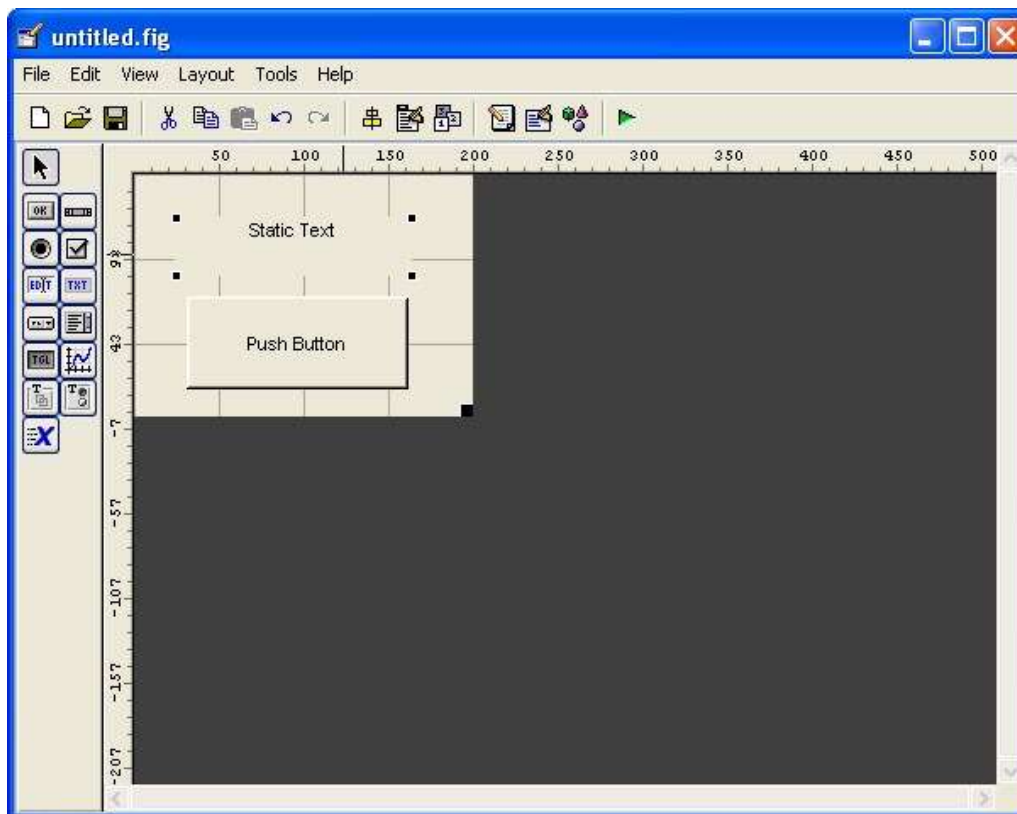
Graphical User Interface (GUI)

قدم اول: طراحی این GUI بسیار ساده است. این GUI شامل یک دکمه فشاری و یک text field است. callback مربوط به pushbutton سبب خواهد شد که عدد داخل text field با فشار کلیک یک واحد اضافه شود. طرح اولیه از این GUI در شکل 1-3 نشان داده شده است.



شکل 1-3 طرح دستی اولیه برای یک GUI حاوی یک pushbutton و جایی برای نوشته قدم دوم: برای چیدن اجزا روی یک GUI، guide را اجرا کنید. وقتی guide اجرا شود، پنجره نشان داده شده در شکل 1-2 ظاهر می شود. در ابتدای امر باید اندازه ناحیه طراحی (کادر GUI) را تنظیم کنیم که در واقع، اندازه GUI نهایی خواهد بود. این کار را با کشیدن مربع کوچک واقع در گوشه پایین سمت راست ناحیه طراحی تارسیدن به اندازه و شکل دلخواه، انجام می دهیم. سپس، در لیست اجزای GUI روی گزینه "pushbutton" کلیک و آن را در ناحیه طراحی قرار می دهیم. شکل حاصل از انجام این مراحل در شکل 1-4 دیده می شود. حال می توانیم با استفاده از ابزار Alignment جایگاه و هم راستایی این دو عنصر را بطور دلخواه تنظیم کنیم.

Graphical User Interface (GUI)

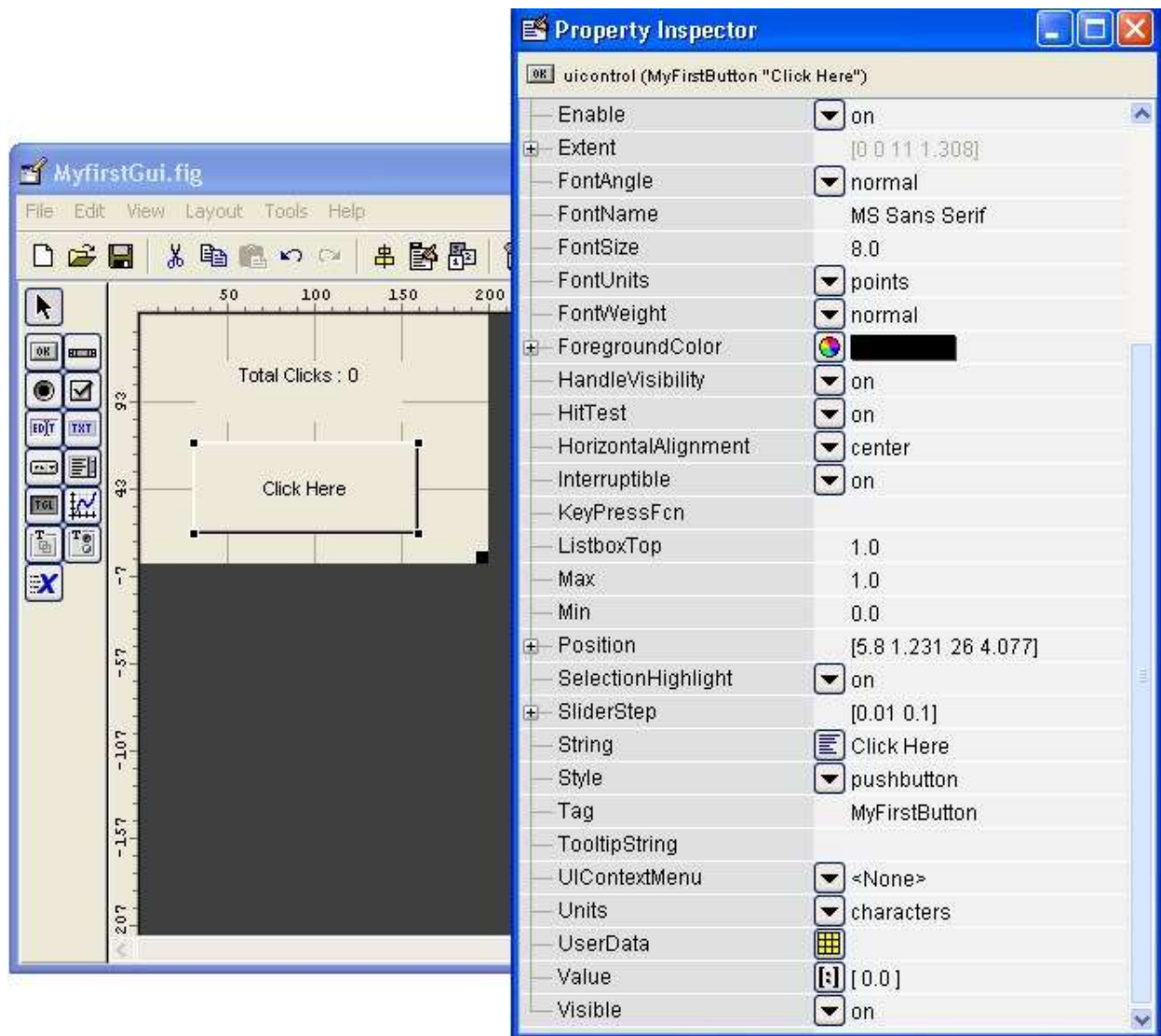


شکل 4-1 کادر GUI نهایی درون پنجره guide

قدم سوم : برای تنظیم property های دکمه فشاری، روی آن یک بار کلیک و سپس گزینه "Property Inspector" را از toolbar انتخاب کنید. برای این کار راه دیگری نیز وجود دارد. بدین ترتیب که روی دکمه فشاری با ماوس right-click کرده و در منویی که ظاهر میشود، گزینه "Inspect Properties" را انتخاب کنید. پنجره property Inspector همان طور که در شکل 5-1 نشان داده شده است، ظاهر خواهد شد. توجه کنید که این پنجره، لیستی از تمام property های pushbutton نمایش می دهد و می توان مقدار هر یک از آنها را تغییر داد.

Graphical User Interface (GUI)

"Property Inspector" در واقع همان توابع `get` و `set` را انجام می دهد، البته به صورت کاملاً ساده و قابل فهم.



شکل 5-1 پنجره property Inspector نشان دهنده property های یک دکمه فشاری که

string آن روی Click Here و Tag آن روی MyFirstButton تنظیم شده است.

در مورد Pushbutton میتوان property های زیادی مانند رنگ، اندازه، فونت،

جایگاهمتن نمایش داده شده روی آن و ... را تنظیم کرد، ولی دو مورد زیر ضروری هستند:

string property: که حاوی متنی است که قرار است روی دکمه فشاری ظاهر شود.

Graphical User Interface (GUI)

Tag property : که نام دکمه فشار می باشد.

در این مورد مثال مقدار string دکمه فشاری "click here" و Tag آن نیز به

`My First Button` تغییر داده می شود.

برای text field نیز باید دو property را تنظیم کنیم؛ که یکی از

آنها String property می باشد که دربرگیرنده متنی است که قرار است نمایش داده

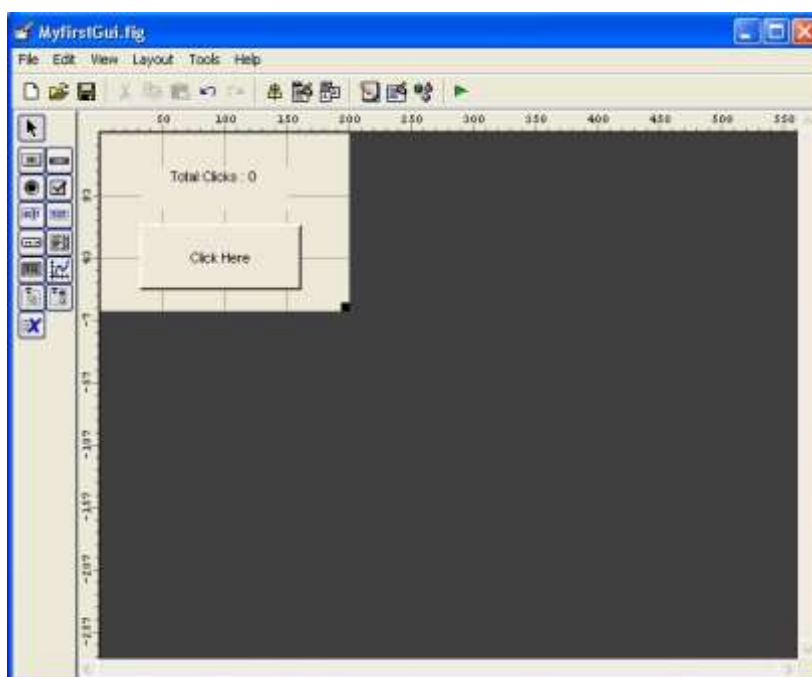
شود و دیگری Tag property است که نام text field می باشد. در واقع

تابع callback به این ناما احتیاج دارد تا text field را پیدا کند و متن درون آن را تغییر

دهد. در این مثال String اینعنصر به `Total click:0` و Tag آن به `My First`

`Text` تغییر داده شده اند. ناحیه طراحی پس از طی این مراحل در شکل 6-1 نشان داده شده

است.



شکل 6-1 کادر طراحی پس از تنظیم property های یک pushbutton

Graphical User Interface (GUI)

و `textfield`.

همچنین امکان تنظیم `property` های خود شکل اصلی نیز وجود دارد و این کار را میتوان با کلیک روی یک نقطه خالی (به طوری که روی عنصر نباشد) در `Layout Editor` و سپس انتخاب ابزار `Property Inspector` انجام داد. تغییر نام `Figure` نیز ایده خوبی است، زیرا رشته درون `name property` هنگام اجرای `GUI` به صورت عنوان در بالای پنجره شکل ظاهر می شود. البته این کار ضرورتی ندارد و تنها به زیبایی کار کمک می کند.

قدم چهارم: هم اکنون ناحیه طراحی را تحت نام `MyFirstGUI` ذخیره میکنیم. برای این کار از منوی `File`، گزینه `MyFirstGUI` را به عنوان نام فایل تایپ کنید، سپس روی دکمه `Save` کلیک کنید، این عمل به طور اتوماتیک دو فایل به نام های `MyFirstGUI.fig` و `MyFirstGUI.m` ایجاد میکند. فایل شکل (`.fig`) حاوی `GUI` طراحی شده میباشد و `M-File` حاوی کدی است که فایل شکل را `load` و `GUI` را ایجاد میکند. درون این `M-File` برای هر عنصر فعال در `GUI` یک تابع `callback` وجود دارد.

هم اکنون این `GUI` کامل شده است ولی هنوز کاری را که بدان محول شده انجام نمیدهد. این `GUI` را میتوان با تایپ `MyFirstGUI` در پنجره فرمان، همان طور که در شکل 7-1 دیده میشود، اجرا کرد. اگر در این `GUI` روی دکمه کلیک شود، پیغام زیر در پنجره فرمان ظاهر میشود.

Graphical User Interface (GUI)



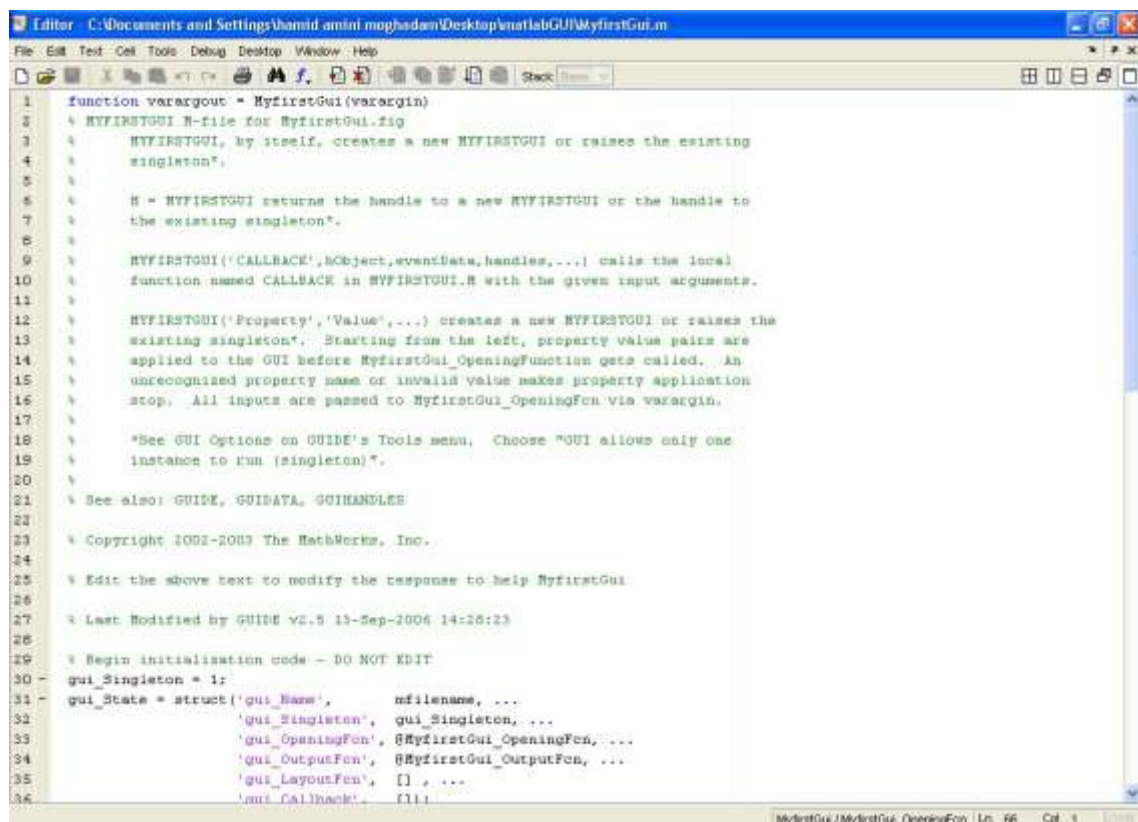
شکل 7-1 با نوشتن عبارت MYFirstGui در پنجره فرمان، GUI کار خود را آغاز می کند.
MyFirstButton Callback not implemented yet.

بدین معنی که callback مربوط به MyFirstButton هنوز مشخص نشده است.

بخشی از M-File که خود به خود توسط guide ایجاد شده است، در شکل 8-1 نشان داده شده

است.

Graphical User Interface (GUI)



```
1 function varargout = MyFirstGui(varargin)
2 % MYFIRSTGUI M-file for MyFirstGui.fig
3 %
4 % MYFIRSTGUI, by itself, creates a new MYFIRSTGUI or raises the existing
5 % singleton*.
6 %
7 % H = MYFIRSTGUI returns the handle to a new MYFIRSTGUI or the handle to
8 % the existing singleton*.
9 %
10 % MYFIRSTGUI('CALLBACK', hObject,eventData,handles,...) calls the local
11 % function named CALLBACK in MYFIRSTGUI.H with the given input arguments.
12 %
13 % MYFIRSTGUI('Property','Value',...) creates a new MYFIRSTGUI or raises the
14 % existing singleton*. Starting from the left, property value pairs are
15 % applied to the GUI before MyFirstGui_OpeningFcn gets called. An
16 % unrecognized property name or invalid value makes property application
17 % stop. All inputs are passed to MyFirstGui_OpeningFcn via varargin.
18 %
19 % *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only one
20 % instance to run (singleton)".
21 %
22 % See also: GUIDE, GUIDATA, GUIHANDLES
23 %
24 % Copyright 1991-2003 The MathWorks, Inc.
25 %
26 % Edit the above text to modify the response to help MyFirstGui
27 %
28 % Last Modified by GUIDE v2.5 13-Sep-2006 14:26:23
29 %
30 % Begin initialization code - DO NOT EDIT
31 gui_Singleton = 1;
32 gui_State = struct('gui_Name',       mfilename, ...
33                  'gui_Singleton',   gui_Singleton, ...
34                  'gui_OpeningFcn', @MyFirstGui_OpeningFcn, ...
35                  'gui_OutputFcn',  @MyFirstGui_OutputFcn, ...
36                  'gui_Callback',    []);
```

شکل 1-8 M-File مربوط به MyFirstGUI که به طور خودکار تولید می شود .

این فایل شامل تابع MyFirstGUI و تعدادی زیر توابع خام برای پیاده سازی و اجرای callback های اجزای فعال GUI است. اگر تابع MyFirstGUI بدون آرگومان فراخوانی شود، آنگاه این تابع، GUI درون فایل MyFirstGUI.fig را نمایش میدهد. ولی اگر این تابع با آرگومان فراخوانی شود، آنگاه تابع فرض میکند که آرگومان اولش نام یک زیر تابع است و با استفاده از fevel آن تابع را فراخوانی می کند و بقیه آرگومان ها را به آن تابع می فرستد. وظیفه هر تابع callback اداره پیش آمدهای یک عنصر GUI است. اگر کلیک ماوس روی یک جزء GUI (یا ورودی صفحه کلید برای Edit Field ها) اتفاق بیافتد، آنگاه تابع callback مربوط به آن جزء، به طور خودکار توسط MATLAB فراخوانی میشود. نام

Graphical User Interface (GUI)

تابع callback همان مقدار Tag property برای آن جزء GUI به اضافه پسوند
"_Callback" در انتهای آن است، یعنی نام تابع callback برای MyFirstButton به
صورت MyFirstButton_Callback خواهد بود.

M-File هایی که توسط guide ایجاد شده‌اند، حاوی callback برای هر عنصر
فعال GUI است، ولی این callback ها فقط پیغامی را نمایش می دهند مبنی بر اینکه هنوز چیزی
در تابع callback منظور نشده است.

قدم پنجم : اکنون، وقت پیاده‌سازی callback مربوط به دکمه فشاری فرا رسیده است،
اینتابیع شامل یک متغیر دائمی است که برای شمارش تعداد کلیک های انجام شده به کار می رود. وقتی
روی pushbutton کلیک می شود، MATLAB تابع MyFirstGUI را
با MyFirstButton_Callback به عنوان آرگومان اول آن فراخوانی میکند. سپس همان
طور که در شکل 1-9 مشاهده میشود تابع MyFirstGUI زیر
تابع MyFirstButton_Callback را

فراخوانی میکند. این تابع باید شمار کلیکهای زده شده را یک واحد افزایش دهد و متن رشته‌های
جدیدی با این عدد جدید بسازد. سپس باید رشته جدید را در String property مربوط
به

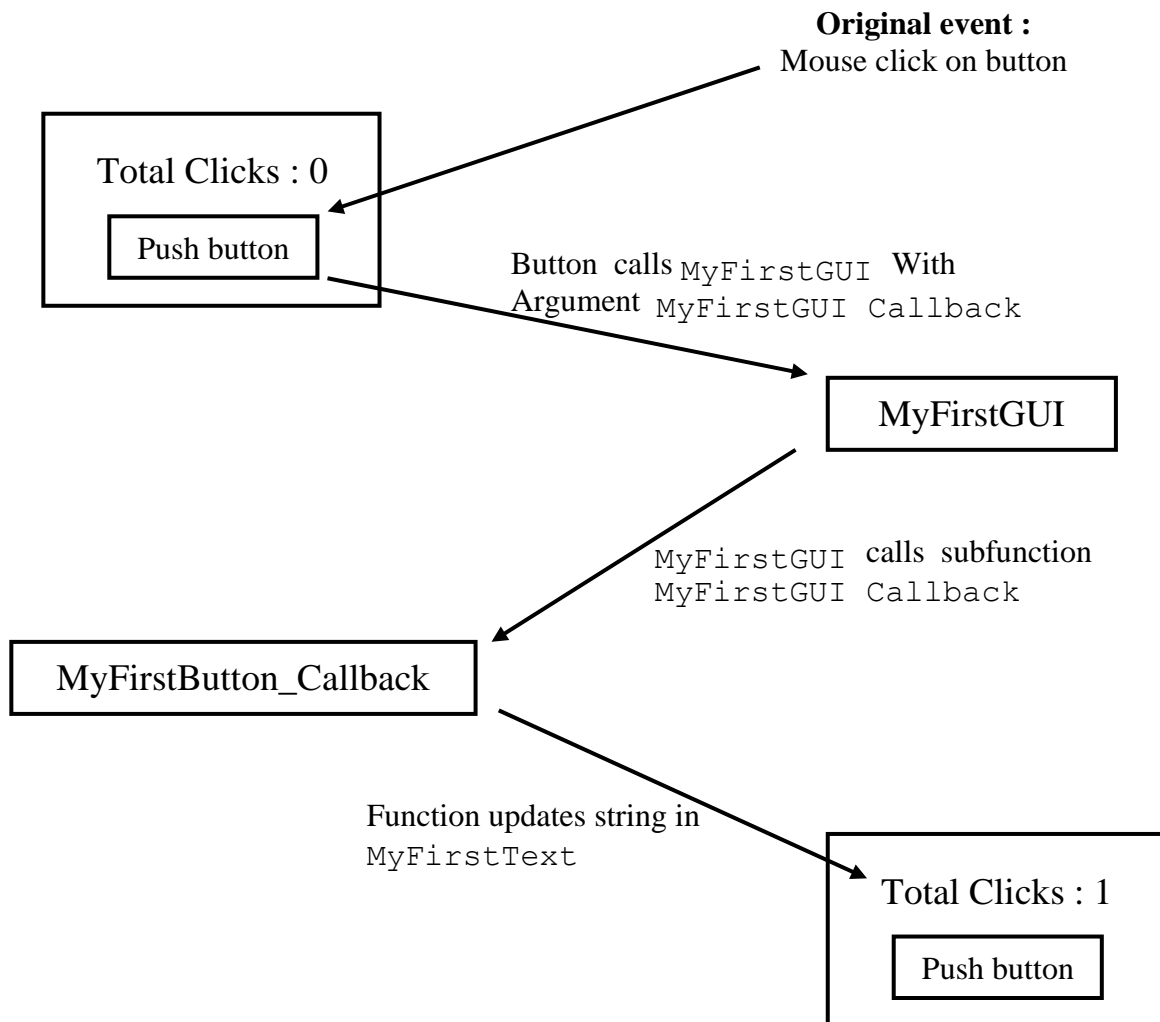
MyFirstText ذخیره کند. تابعی که این مرحله را انجام می دهد در زیر آورده شده است:

```
function MyFirstButton_Callback(hObject, eventdata, handles)
```

```
%Declare and initialize variable to store the count persistent count
```

Graphical User Interface (GUI)

```
if isempty(count)
count=0;
end %Update count count = count + 1;
%Create new string str=sprintf('Total
Clicks : %d',count);
%Update the text field set(handles.MyFirstText,'string',str);
```



Graphical User Interface (GUI)

شکل 9-1 اداره وقایع در برنامه MyFirstGUI . هنگامی که کاربر با ماوس روی دکمه کلیک می کند ، تابع MyFirstGUI با آرگومان MyFirstButton_Callback به طور خودکار فراخوانی می شود . تابع MyFirstGUI به خودی خود ، زیر تابع MyFirstButton_Callback را فراخوانی می کند . این تابع مقدار count را یک واحد افزایش می دهد و سپس مقدار جدید count را در textfield ذخیره می کند .

توجه داشته باشید که این تابع متغیر count را از نوع persistent اعلان می کند و مقدار اولیه آن را صفر قرار می دهد . هر بار که تابع فراخوانی می شود ، مقدار count را یک واحد افزایش داده و رشته جدیدی حاوی مقدار جدید count ایجاد میکند . سپس تابع ، رشته نمایش داده شده در جای متن MyFirstText را update می کند .

برنامه نهایی با تایپ MyFirstGUI در پنجره فرمان اجرا می شود . هنگامی که کاربر روی دکمه کلیک کند ، MATLAB به طور اتوماتیک تابع MyFirstGUI را با MyFirstButton_Callback به عنوان آرگومان اول فراخوانی می کند و تابع MyFirstGUI نیز زیر تابع MyFirstButton_Callback را فرا میخواند . این تابع نیز مقدار متغیر count را یک واحد افزایش داده و متن نمایش داده شده در text field را با مقدار جدید تطابق می دهد .

GUI حاصل بعد از سه فشار دکمه در شکل 10-1 نشان داده شده است .

Graphical User Interface (GUI)



شکل 10-1 برنامه حاصل بعد از سه بار فشار کلید

1-2-1 نگاهی عمیقتر

شکل 1-8 M_File مربوط به MyFirstGUI را که به طور اتوماتیک توسط guide ایجاد شد، نشان میدهد. اکنون قصد داریم که این M-File را از نزدیک بررسی کنیم تا بفهمیم چگونه کار می کند.

ابتدا بیایید نگاه دقیقتری به طرز اعلان function بیاندازیم. توجه داشته باشید که این تابع از متغیر varargin برای معرفی آرگومانهای ورودی و از varargout برای معرفی نتایج خروجی استفاده میکند. تابع varargin میتواند معرف تعداد دلخواهی از آرگومانهای ورودی و تابع varargout میتواند معرف تعداد متغیری از آرگومانهای خروجی باشد. بنابراین کاربر آزاد است که تابع MyFirstGUI را با هر تعداد آرگومان فراخوانی کند.

فراخوانی M-File بدون آرگومان

Graphical User Interface (GUI)

اگر کاربر MyFirstGUI را بدون آرگومان فراخوانی کند، مقداری که توسط nargin برگردانده میشود، صفر خواهد بود. در این صورت، برنامه با استفاده از تابع openfig ، GUI درونفایل MyFirstGUI.fig را باز می کند. شکل تابع openfig بدین صورت است:

```
Fig=openfig ( `mfilename`,`reuse`);
```

که در آن mfilename نام فایل شکلی است که قرار است load شود. آرگومان دوم در اینتابع مشخص میکند که در یک زمان چند نسخه از این شکل می تواند اجرا شود. اگر این آرگومان، `reuse` باشد، بدان معناست که در هر لحظه تنها یک نسخه از شکل میتواند اجرا شود. اگر تابع openfig با گزینه `reuse` فراخوانی شود در حالی که همان شکل از قبل وجود داشته باشد، آنگاه شکل موجود، بدون تغییر به بالای صفحه کامپیوتر برده می شود. در مقابل، اگر این آرگومان `new` باشد، چندین نسخه از شکل میتواند، در آن واحد اجرا شود. هر بار که openfig با گزینه `new` فراخوانده شود، نسخه جدید از شکل ایجاد خواهد شد. به طور پیش فرض، GUI ی ایجاد شده توسط guide، گزینه `reuse` را دارا میباشد، بنابراین تنها یک نسخه از آن در هر لحظه میتواند وجود داشته باشد. اگر که میخواهید چند نسخه از GUI در هر لحظه قابل نمایش باشد، باید تابع openfig را خودتان به صورت دستی تغییر دهید.

وقتی که شکل load شود، تابع MyFirstGUI عبارت زیر را اجرا می کند:

```
Set(fig,'color',get(0,'defaultUicontrolBckgroundcolor'));
```

این تابع رنگ پس زمینه شکل را با رنگ پیش فرض پس زمینه مورد استفاده بوسیله کامپیوتریکه MATLAB روی آن در حال اجرا است، تطبیق میدهد. در واقع این تابع، رنگ GUI را با رنگ دیگر پنجره های کامپیوتر یکی میکند، بنابراین یک GUI را میتوان روی کامپیوترهای با ویندوز

Graphical User Interface (GUI)

نوشت و روی کامپیوترهای با سیستم عامل UNIX اجرا نمود و بر عکس، به طوری که در هر دو محیط کاملاً طبیعی به نظر برسد.

دو عبارت بعدی یک ساختار حاوی handle های اشیای درون شکل فعلی تولید میکنند و این ساختار را به عنوان داده ای منحصر به فرد به خود شکل در آن ذخیره می کنند.

ایجاد ساختار handles :

```
Handles = guihandles(fig);
```

ذخیره این ساختار در داخل شکل:

```
Guidata(fig,handles);
```

تابع guihandles یک ساختار حاوی handle مربوط به تمام اشیای درون شکل مورد نظر، ایجاد میکند. نام عناصر درون این ساختار با Tag هر یک از اجزای GUI متناظر است و مقدار آنها نیز با handle هر یک از اجزا متناظر است. به عنوان مثال، ساختار handle در MyFirstGUI.m به صورت زیر است:

```
Handles = guihandles (fig)
```

```
Handles =
```

```
Figure1          99.0005
```

```
MyFirstText:  3.002
```

```
MyFirstButton: 100.0007
```

سه جزء در این شکل وجود دارد: خود شکل (figure)، به علاوه یک text field و

یک pushbutton. تابع guidata ساختار handles را به عنوان داده ای مربوط به شکل

در آن ذخیره می کند و این کار را تابع setappdata انجام می دهد.

عبارت پایانی در این GUI، در صورت اختصا آرگومان خروجی در هنگام فراخوانی

Graphical User Interface (GUI)

MyFirstGUI، ساختار handles را به فراخوان باز می گرداند.

```
If nargin > 0  
    Varargin{1} = fig; end
```

فراخوانی M-File با آرگومان

اگر کاربر، MyFirstGUI را با آرگومان فراخوانی کند، مقدار بازگردانده شده

بوسیلهٔ nargin از صفر بزرگتر خواهد شد. در این صورت، با آرگومان اول به عنوان یک نام

تابع callback رفتار و آنرا توسط تابع fevel اجرا میکند. تابع fevel تابعی را که نام آن

در varargin{1} آمده است اجرا میکند و بقیه آرگومانها را (varargin{2} , varargin{3} و

غیره) به آن تابع میفرستد.

این مکانیزم سبب میشود که توابع callback به زیر توابعی تبدیل شوند بطوریکه امکان

فراخوانیاتیفاقی آنها از جایی دیگر خارج از M-File وجود نداشته باشد.

1-2-2 ساختار یک زیر تابع callback

هر زیر تابع callback فرم استاندارد زیر را دارد:

```
Function componentTag_callback(hObject, eventdata, handles,varargin);
```

که در آن componentTag نام جزئی است که این callback را بوجود آورده است

(که همان رشتهٔ درون Tag property آن جزء می باشد). آرگومان های این زیر تابع عبارتند از:

hObject - که handle شکل مادر (parent) می باشد

eventdata - در نسخه فعلی MATLAB از این آرایه استفاده نمی شود.

Graphical User Interface (GUI)

handles - ساختار حاوی تمام handle های اجزای درون شکل میباشد.
varargin - یک آرگومان اضافی برای فرستادن آرگومانهای دیگر به تابع callback
میباشد. برنامه‌نویس در صورت نیاز میتواند از این ویژگی برای ارائه اطلاعات بیشتر تابع callback بهره بگیرد.

باید به این نکته توجه داشت که هر تابع callback دسترسی تمام و کمال به ساختار handles دارد، بنابراین تابع callback میتواند اجزای GUI درون figure را تغییر دهد. ما از این ویژگی در تابع callback دکمه فشاری در برنامه MyFirstGUI، در آنجا که میخواستیم تابع callback دکمه فشاری، متن نمایش داده شده در text field را تغییر دهد، بهره گرفتیم:

```
%Update the text field Set(handles.MyFirstText.'string',str);
```

1-2-3 اضافه کردن Application Data به یک شکل

این امکان وجود دارد که اطلاعات بخصوصی را که مورد نیاز برنامه GUI است، به جای ذخیره در حافظه دائم یا سراسری، در ساختار handles ذخیره کرد. طراحی GUI حاصل از این روش بسیار مقاومتر و مطمئنتر است، زیرا برنامه‌های دیگر MATLAB نمی‌توانند به طور تصادفی داده global مربوط به GUI را تغییر دهند و چند نسخه یکسان GUI که در یک زمان اجرا میشوند، نیز نمیتوانند در کار یکدیگر خلل ایجاد کنند.

برای اضافه کردن داده محلی به ساختار handles، باید M-File را پس از ایجاد آن با دستور guide، به طور دستی تغییر داد. برنامه نویس باید داده مربوط به برنامه را (application data) بعد از فراخوانی guidata و قبل از guidata به

Graphical User Interface (GUI)

ساختاری handles اضافه کند. بهعنوان مثال، برای اضافه کردن تعداد کلیکهای ماوس (count) به

ساختار handles، برنامه را بهصورت زیر تغییر می دهیم:

```
%Generate a structure of handles to pass to callbacks Handles  
= guidata(fig);
```

```
%Add count to the structure. Handles.count  
= 0;
```

```
%Store the structure  
Guidata(fig,handles);
```

اکنون دادهٔ مربوط به برنامه، همراه ساختار handles به تمام توابع callback

فرستاده می شود و در جای لازم از آن استفاده می شود.

نسخه‌های از تابع callback دکمهٔ فشاری که از مقدار متغیر count در

ساختار handles استفاده میکند، در زیر آورده شده است. توجه کنید که هرگونه تغییر در

اطلاعات درون ساختار handles را باید با فراخوانی guidata ذخیره کرد.

```
Function componentTag_callback(hObject, eventdata, handles,varargin);
```

```
%Update count  
Handles.count = handles.count+1
```

```
%Save the update handles structure guidata(hObject,handles);
```

Graphical User Interface (GUI)

```
%Creat new string
Str=sprintf('Total Clicks: %d',handles.count);

%Update the text field Set
(handles.MyFirstText,'string',str);
```

1-2-4 چند تابع مفید دیگر

سه تابع بخصوص دیگر نیز گاهی در طراحی توابع callback مورد استفاده قرار می گیرند:

`gcbf findobj` تابع `gcbf` (`get callback object`)، `handle` شی ای را که آن callback را ایجاد کرده است، باز می گرداند و تابع `get callback`، `handle` (`figure`) شکل حاوی آن شیء را باز میگرداند. این توابع میتوانند برای تعیین شیء و شکل بوجود آورنده یک `callback`، توسط تابع `callback` مورد استفاده قرار گیرند.

تابع `findobj` در میان اشیاء فرزند واقع درون یک شیء مادر به دنبال آنهایی که دارای یک مقدار مشخص از یک `property` معلوم هستند، میگردد و به محض پیدا کردن اشیایی که خصوصیاتشان با گزینه مورد جستجو منطبق باشد، `handle` آن ها را بر میگرداند. فرم معمول این تابعه صورت زیر است:

```
Hndl = Findobj(parent,'property',value);
```

که در آن `parent`، `handle` شیء مادر است. `property` خصوصیتی است که قرار است چک شود. `value` مقداری از آن `property` است که قرار است مورد جستجو قرار گیرد.

Graphical User Interface (GUI)

به عنوان مثال، فرض کنید که یک برنامه نویس می خواهد متن روی یک دکمه فشاری با نام `button1` را هنگامی که یک تابع callback اجرا میشود، تغییر دهد. برنامه نویس این کار را می تواند با پیدا کردن دکمه فشاری مورد نظر و جایگزینی متن آن با متن جدید به صورت زیر انجام دهد:

```
Hndl = findobj(gcf,'Tag','Button1');  
Set(Hndl,'string',New text');
```

3-1 property های یک شیء

هر شیء GUI شامل طیف وسیعی از property هایی میباشد که کاربر میتواند آنها را بسته به سلیقه و نیاز خود تغییر دهد. این property ها برای انواع مختلف اشیاء (مثل figure ها، axe ها، uicontrol ها و غیره) کمی فرق می کند. Property های کلیه اشیاء در Online Help Browser ثبت شده اند و قابل دسترسی هستند. با این حال، چند property مهم برای شیء figure و اشیای uicontrol در زیر به آن اشاره شده است.

Property اشیاء را می توان با ابزار Property Inspector و یا توابع get و set تغییر داد. ولی با اینکه استفاده از Property Inspector برای تنظیم property های یکشیء آسانتر است، برای تنظیم property اشیاء از درون برنامه، مثلاً از درون یک تابع callback، باید از توابع get و set استفاده کنیم.

Graphical User Interface (GUI)

GUI اجزای 1-4

این بخش خلاصه‌های از ویژگیهای اصلی اجزای متداول GUI ارائه میکند . چگونگی ایجاد و استفاده از هر جزء به همراه انواع پیش آمدهایی که هر کدام از آنها می توانند ایجاد کنند، به تفصیل در این قسمت آمده است. اجزای مورد بحث در این فصل عبارتند از:

Text Field (جای متن) Edit

Boxes

Frames (قاب ها)

Pushbuttons (دکمه های فشاری)

Toggle Button (دکمه های دو حالتی)

Chekboxes

Radio Bations

Popup Menus

List Boxes

Sliders

Property های مهم یک شکل

color: رنگ شکل را مشخص میکند این مقدار یا میتواند یک رنگ از پیش تعریف شده،

مثل ``r`` و ``g`` و ``b`` باشد یا اینکه یک بردار با سه عنصر مشخص کننده نسبت به سه رنگ

اصلی قرمز، سبز و آبی با مقیاس بین 0-1. مثلاً رنگ magenta با بردار $[1 \ 0 \ 1]$ مشخص می

شود.

Graphical User Interface (GUI)

MenuBar : مشخص می کند که آیا مجموعه منوهای پیش فرض باید روی شکل ظاهر شوند یاخیر. مقادیر ممکن برای این `property` و `figure` برای نمایش منوهای پیش فرض و یا `none` برای پاک کردن آنها می باشد.

Name : یک رشته حاوی نامی است که در عنوان شکل ظاهر می شود.

NumberTitle : مشخص مینماید که آیا شماره شکل در عنوان شکل ظاهر شود یا خیر. مقادیر ممکن برای آن `on` و `off` هستند.

position : مکان و موقعیت شکل را روی صفحه مانیتور در مقیاسی که در `property` `units` تعیین شده است، مشخص می کند. این مقدار یک بردار با چهار عنصر است که دو عنصر اول آن معرف مختصات x و y گوش پایین سمت چپ شکل و دو عنصر بعدی معرف عرض و طول شکل می باشند.

SelectionType : نوع انتخاب را برای آخرین کلیک ماوس روی شکل، تعیین میکند. تککلیک ماوس نوع `normal` و دو بار کلیک نوع `open` را باز می گرداند. گزینه های دیگری نیز وجود دارند. برای دیدن آنها به پوشه های Online در MATLAB رجوع کنید.

Tag : "نام" شکل است که از آن برای شناسایی شکل استفاده می شود.

Units : مقیاس و واحدی است که شکل در آن تعریف میشود و گزینه های ممکن برای آن عبارتند از `centimeters` و `normalized` و `points` و `pixels` و

`characters` و `inches`. واحد پیش فرض `pixels` می باشد. **Visible** : مرئی یا نامرئی بودن شکل را مشخص میکند. مقادیر ممکن برای آن `on` و `off` می باشند.

Graphical User Interface (GUI)

Windowstyle: تعیین کننده normal یا modal بودن شکل است. مقادیر ممکن،

`normal` و `modal` هستند.

مشخصات مهم اشیاء uicontrol

BackgroundColor: تعیین کننده رنگ پس زمینه شیء است که مقدار آن میتواند یک رنگ

از پیش تعریف شده، مثل `r` و `g` و `b`، یا اینکه برداری با سه عنصر، مشخص کننده نسبتبه

سه رنگ اصلی قرمز، سبز و آبی در مقیاس بین 0 تا 1 باشد. مثلاً رنگ magenta با بردار [1

0 1 مشخص می شود.

Callback: تعیین کننده نام و پارامترهای تابع فراخوانی شده در هنگام فعال شدن

شیء مربوط به آن (توسط صفحه کلید یا ورودی نوشتاری) می باشد.

Enable: مشخص میکند که آیا یک شیء قابل انتخاب است یا خیر. اگر

این property غیر فعال باشد، آنگاه شیء به ماوس و صفحه کلید پاسخ نخواهد داد. مقادیر

ممکن برای آن `on` و

`off` می باشد.

FontAngle: رشتهای حاوی زاویه فونت نمایش داده شده روی شیء است. مقدار آن میتواند

`normal` و `italic` و `oblique` باشد. **FontName**: رشته ای حاوی نام

فونت برای متن نمایش داده شده روی شیء است.

FontSize: یک عدد مشخص کننده اندازه فونت نمایش داده شده روی شیء است.

اندازه فونت به طور پیش فرض در واحد points می باشد.

Graphical User Interface (GUI)

FontWeight: رشته‌های حاوی ضخامت فونت نمایش داده شده روی شیء است و مقدار آن

می تواند `light`` و `normal`` و `demi`` یا `bold`` باشد.

ForegroundColor: تعیین کننده رنگ پیش زمینه شیء می

باشد.

HorizontalAlignment: تعیین کننده جایگاه افقی متن درون شیء است. مقادیر ممکن

عبارتند: `left`` و `center`` و `right``.

Max: حداکثر مقدار `value property` برای شیء. از

Min: حداقل مقدار `value property` برای شیء.

Parent: `handle` شکل دربرگیرنده این شیء است.

Position: مکان شیء را روی صفحه، در مقیاس تعیین شده در مشخصه `units``

مشخص میکند و مقدار آن یک بردار با چهار عنصر است که دو عنصر اول آن مختصات `x` و `y` از

گوشه پایین سمت چپ شکل دربرگیرنده آن هستند. دو عنصر بعدی طول و عرض شکل هستند.

Tag: "نام" شیء است که از آن برای تعیین موقعیت و شناسایی شیء استفاده می شود.

Tooltipstring: مشخص کننده متن راهنمایی است که وقتی کاربر اشارهگر ماوس

را روی یک شیء نگاه می دارد، نمایش داده می شود.

Units: مقیاس و واحدی است که شیء در آن تعریف میشود و گزینه‌های ممکن برای

آن عبارتند از: `inches`` و `centimeters`` و `pixels`` و

`points`` و `normalized`` یا `characters``. واحد پیش فرض `pixels`` می باشد.

Graphical User Interface (GUI)

value: مقدار فعلی uicontrol می باشد. برای toggle button ها، check box ها و radio button ها در وضعیت on این مقدار، مقدار Max property و در وضعیت off ، مقدار آن، مقدار Min property می باشد. برای دیگر کنترل ها این property می تواند معانی متفاوتی داشته باشد.

visible: مرئی یا نامرئی بودن شیء را مشخص میکند و مقدار آن میتواند `on` یا

`off` باشد.

1-4 Text Field ها

4-1

یک text field شیء ای گرافیکی است که یک متن رشته ای را درون خود نمایش می دهد. می توان راستا و جایگاه متن را درون ناحیه نمایش، با تنظیم Horizontal Alignment property تعیین کرد. به طور پیش فرض، متن در مرکز text field قرار می گیرد. با ایجاد یک uicontrol که property style ش `edit` است، یک text field بوجود

می آید. text field را همچنین می توان با استفاده از ابزار در Layout Editor به GUI اضافه کرد.

Text field ها callback ی را فعال نمی کنند، ولی می توان مقدار نمایش داده شده درون آنها را با تغییر String property آن از درون یک تابع callback ، همان طور که در بخش 2-1 دیدید، تغییر داد.

2-4-1 Edit Box ها

Graphical User Interface (GUI)

یک edit box شی ای گرافیکی است که به کاربر امکان وارد کردن یک متن رشته ای را میدهد. هنگامی که کاربر کلید Enter را پس از تایپ رشته درون جعبه، فشار میدهد، callback این عنصر فعال می شود. یک edit box را می توان با ایجاد یک uicontrol که property style آن 'edit' می باشد، تولید کرد. edit box را همچنین می توان با استفاده از ابزار edit box در Layout Editor با GUI اضافه کرد.

شکل 1-11 یک GUI ساده، حاوی یک edit box با نام 'EditBox' و یک text field با نام 'TextBox' را نشان می دهد. هنگامی که کاربر یک رشته را درون edit box تایپ میکند، این شیء بطور خودکار تابع EditBox_Callback را فراخوانی می کند. این تابع به کمک ساختار handles موقعیت و شناسایی edit box را مشخص می کند و رشته تایپ شده را از سوی کاربر دریافت می کند. سپس با تعیین موقعیت و مکان text field، این رشته را در آن نمایش میدهد.

```
function EditBox_Callback(hObject, eventdata, handles)
```

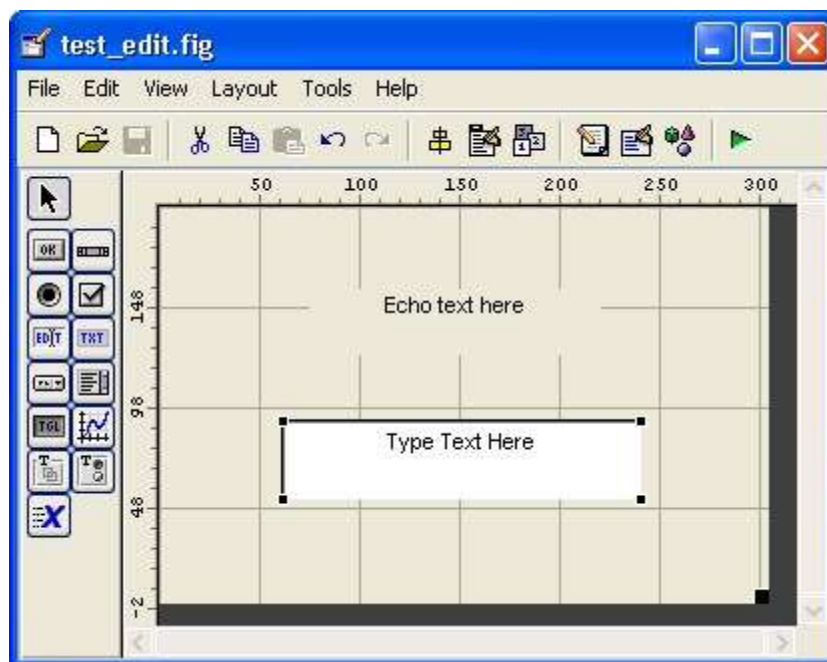
```
%Find the value typed into the edit box
```

```
str = get(handles.EditBox,'string');
```

```
%Place the value into the text field set
```

```
(handles.TextBox,'string',str);
```

Graphical User Interface (GUI)



یک-1 شکل GUI ساده با یک edit box و یک text field

شکل 1-12 این GUI را درست بعد از شروع به کارش، یعنی بعد از اینکه کاربر کلمه

'Hello' را در edit box تایپ می کند نشان می دهد.

Graphical User Interface (GUI)



شکل 1-12 GUI تولید شده توسط برنامه `test_edit`

3-4-1 **Frame** ها (قاب) نیز شیای گرافیکی است که مستطیلی را در GUI نمایش

میدهد . میتوان از قابها برای قرار دادن گروهی از اشیای گرافیکی مربوط به هم در درون یک جعبه و قاب استفاده کرد.

برای مثال، همان طور که در شکل 1-10 دیده می شود، میتوان از یک قاب برای قرار دادن یک گروه

از `radio button` ها در کنار یکدیگر، استفاده کرد.

Graphical User Interface (GUI)

یک قاب را با ایجاد یک `uicontrol` که `style property` آن `'frame'` می باشد، می توان ایجاد کرد. همچنین فریم ها را می توان با استفاده از ابزار `frame` در `Layout Editor` به `GUI` افزود. قابها `callback` ی تولید نمیکنند. البته در `MATLAB` نسخه 7 اثری از `Frame` ها مشاهده نمی شود و باید از `Panel` که دارای عملکردی کاملاً مشابه است، به جای `frame` استفاده کرد. **1-4-4 Pushbutton ها**

یک `pushbutton` (دکمه فشاری) عنصری است که کاربر میتواند با کلیک روی آن، عملیات خاصی را فعال کند. هنگامی که کاربر روی `pushbutton` کلیک می کند، `callback` آن فعال می شود. این عنصر را می توان با ایجاد `uicontrol` ی که `style property` آن `'pushbutton'` است، ایجاد کرد. همچنین آنها را میتوان با استفاده از ابزار `pushbutton` در `Layout Editor` به `GUI` اضافه کرد.

تابع `MyFirstGUI` در شکل 1-10 تصویری از کاربرد `pushbutton` ارائه می دهد.

1-4-5 Toggle Button ها

`toggle button` نوعی از دکمه است که دو حالت دارد: `on` (فشرده شده) و `off` (رها). یک `toggle button` با کلیک ماوس روی آن، بین دو حالت تغییر وضعیت می دهد. مشخصه `'value'` این عنصر وقتی کلید در حالت `on` قرار دارد `max` (که معمولاً 1 است) و هنگامی که `off` است `min` (که معمولاً 0 است) می شود. `toggle button` را می توان با ایجاد یک `uicontrol` که `style property` آن،

Graphical User Interface (GUI)

`togglebutton` می باشد، خلق کرد. همچنین آن را می توان با استفاده از ابزار

toggle button در Layout Editor ایجاد کرد.

یک 1-13 شکل GUI ساده حاوی یک toggle button با نام `ToggleButton` و

یک textfield با نام `TextBox` را نشان می دهد. هنگامی که کاربر

روی togglebutton کلیک میکند، این عنصر به طور خودکار

تابع ToggleButton_Callback را فراخوانی می کند. این تابع با بکارگیری ساختار handles

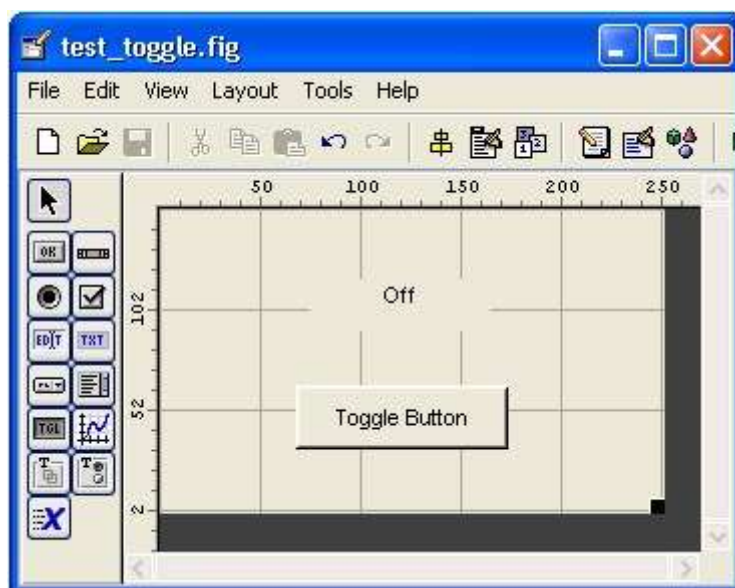
، موقعیت و مکان toggle button را شناسایی کرده و حالت آن را از Value property اش

دریافت می کند. سپس این تابع با تعیین موقعیت text field حالت اخذ شده در مرحله قبل را

درون text field نمایش می دهد.

```
function togglebutton1_Callback(hObject, eventdata, handles)
%Find the state of the toggle button state
= get (handles.ToggleButton, 'Value');
%Place the value into the text field if
state == 0
    set (handles.TextBox, 'string', 'Off'); else
    set (handles.TextBox, 'string', 'On'); end
```

Graphical User Interface (GUI)



یک-13 شکل GUI ساده حاوی یک toggle button و یک text field

شکل 1-14 تصویری از این GUI را درست بعد از شروع به کار و بعد از اینکه کاربر برای بار اول روی toggle button کلیک می کند، نمایش می دهد.



شکل 1-14 تصویری از GUI تولیدی به وسیله برنامه `test_togglebutton` هنگامی که

`togglebutton` خاموش و روشن می شود .

Graphical User Interface (GUI)

1-4-6 Checkbox ها و Radio button ها

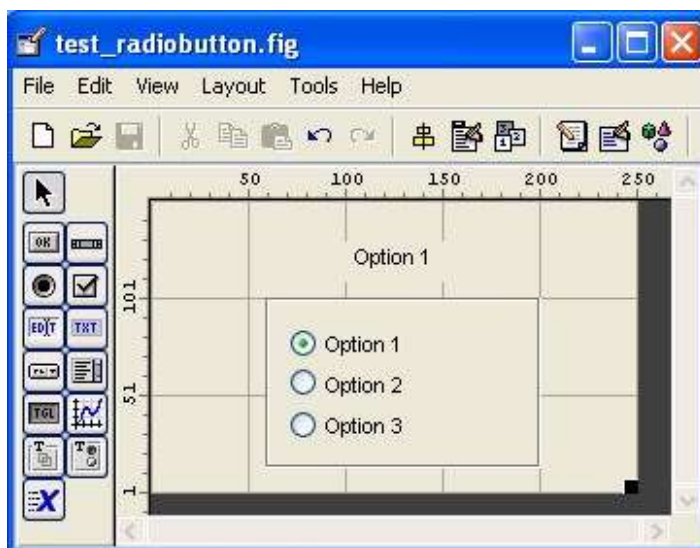
اساس کار Checkbox ها و Radio button ها مشابه toggle button ها است و تنها شکل ظاهری آنها فرق می کند. Checkbox ها و Radio button ها نیز مشابه toggle button ها دارای دو وضعیت on و off هستند و با هر کلیک ماوس روی آنها بین این دو حالت تغییر وضعیت داده و در هر مرتبه callback آنها فعال می شود. Value property این اجزا وقتی که on هستند max (که معمولاً 1 است) و وقتی که off هستند min (که معمولاً 0 است) می باشد. نمونه ای از یک checkbox و radio button در شکل 1-10 نشان داده شده است.

یک checkbox را می توان با ایجاد uicontrol ی که style property آن checkbox `checkbox` میباشد، ایجاد کرد. همچنین این عنصر را میتوان با استفاده از ابزار checkbox در Layout Editor، ایجاد کرد. برای radio button نیز وضع به همین منوال است. یک

radio button را می توان با ایجاد uicontrol ی که style آن `radiobutton` می باشد، ایجاد کرد و همچنین آن را می توان با استفاده از ابزار radio button در Layout Editor ایجاد کرد.

متداول است که از checkbox ها برای نمایش گزینه های on/off استفاده میشود و مجموعه ای از radio button برای انتخاب گزینه ای از میان گزینه های مستقل استفاده می شود.

Graphical User Interface (GUI)



شکل 1-15 یک GUI ساده حاوی سه radio button به همراه یک text field برای نمایش انتخاب کنونی.

شکل 1-15 مثالی از چگونگی ایجاد گروهی از گزینه های مستقل را با radio button نشان می دهد. GUI نشان داده شده در این سه radio button با برچسب های "Option1" و "Option2" و "Option3" در خود دارد. هر radio button از یک callback مشابه ولی با پارامتر مستقل استفاده می کند.

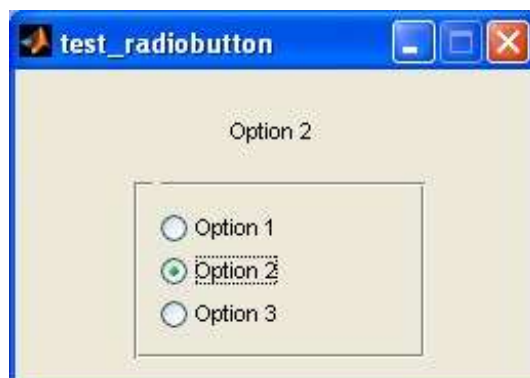
توابع callback مربوط به هر radio button :

```
function radiobutton1_Callback(hObject, eventdata, handles)
    set(handles.Label1,'string','Option 1');
```

```
function radiobutton2_Callback(hObject, eventdata, handles)
    set(handles.Label1,'string','Option 2');
    function radiobutton3_Callback(hObject, eventdata,
handles)
        set(handles.Label1,'string','Option 3');
```


Graphical User Interface (GUI)

هنگامی که کاربر روی یکی از radio button ها کلیک می کند، تابع callback مربوطه آن اجرا می شود. این تابع متن نمایش داده شده در text box را به گزینه ای که هم اکنون انتخاب شده تغییر می دهد و radio button فعلی را روشن (on) و بقیه radio button ها را خاموش (off) می کند. توجه کنید که این GUI از یک قاب برای قرار دادن radio button ها در کنار هم، برایتأکید بر اینکه اینها جزء یک مجموعه هستند، بهره گرفته است. شکل 16-1 تصویر این GUI را پس از انتخاب Option 2 نشان می دهد.



شکل 16-1 تصویر GUI ی تولید شده به وسیله برنامه test_radiobutton

1-4-7 منوهای Popup

منوهای popup اجزای گرافیکی هستند که به کاربر اجازه انتخاب یک گزینه از میان لیستی از گزینههای مستقل را میدهند. این لیست که کاربر از میان آن گزینه مورد نظر را انتخاب میکند، به وسیله آرایه های از نوع cell که حاوی رشته های گزینههاست، مشخص میشود. مشخصه `value` برای این منو تعیین میکند که کدام گزینه هم اکنون انتخاب شده است. یک منوی popup را می توان به وسیله ابزار popup menu در Layout Editor به GUI اضافه نمود.

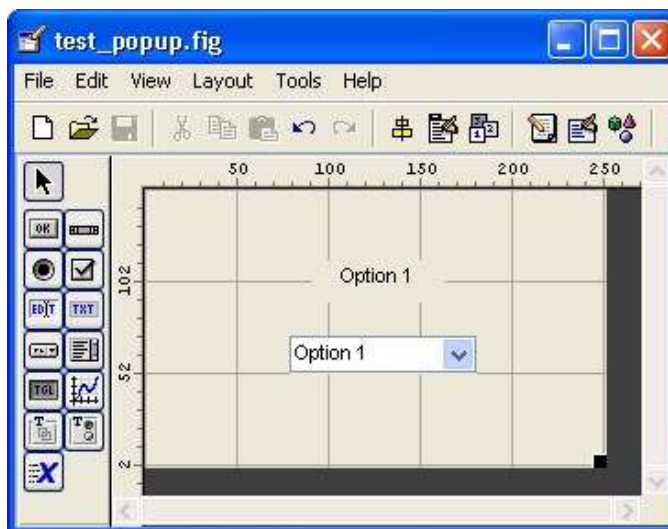
Graphical User Interface (GUI)

شکل 1-17 نمونه‌های از یک منوی popup را نشان می‌دهد. این GUI حاوی یک منوی

و ... است.

"Option

برچسب‌های



"Option 2" و

با پنج گزینه با " 1

popup

شکل 1-17 یک GUI ساده حاوی یک منوی popup و یک text field برای

نمایش گزینه انتخاب شده.

تابع callback مربوط به منوی

: popup

```
function Popup1_Callback(hObject, eventdata, handles)
```

```
%Find the value of the popup menu
```

```
Value = get(handles.Popup1, 'Value');
```

```
%Place the value into the text field
```

```
str = ['Option ' num2str(Value) ]; set
```

```
(handles.Label, 'string', str);
```

Graphical User Interface (GUI)

این تابع گزینه انتخاب شده را با چک کردن پارامتر `value` تشخیص میدهد و یک رشته حاوی این مقدار ایجاد نموده و آنرا در text field نمایش میدهد. شکل 1-18 تصویری را از این GUI پس از انتخاب 4 Option نشان می دهد.



شکل 1-18 تصویری از GUI ایجاد شده به وسیله برنامه test_popup

ها

List Box 1-4-8

list box ها اشیایی گرافیکی هستند که چند خط نوشته را در خود نمایش می دهند و به کاربر اجازه انتخاب یک یا چند خط از این خطوط را می دهند. اگر تعداد این خطوط از فضای list box بیشتر باشد به طوری که در آن جای نگیرند، در کنار آن یک scroll bar ایجاد خواهد شد که به کاربر امکان بالا و پائین رفتن در list box را می دهد. خطوطی که کاربر می تواند انتخاب کند، به وسیله یک آرایه سلولی مشخص می شود و مقدار Value property مشخص می کند که کدام رشته انتخاب شده است. یک list box را می توان با ایجاد uicontrol ی که style property آن

Graphical User Interface (GUI)

`listbox` است، بوجود آورد. `list box` را همچنین می توان به کمک ابزار `listbox` در

Layout Editor ایجاد نمود.

از `list box` ها می توان برای انتخاب یک گزینه از میان مجموعه ای از گزینه های ممکن استفاده نمود. در کاربردهای متداول GUI، تک کلیک ماوس روی یکی از موارد لیست تنها باعث انتخاب آن میشود و منجر به اتفاق خاص دیگری نمیشود. با این وجود، عملیات منتظر و آماده تحریرهای دیگر از طرف سایر عناصر، مثل یک `pushbutton` میشود. نوع پیشامدهای تک-کلیک و دوبار-کلیک را می توان با استفاده از `SelectionType` property شکلی که عمل کلیک کردن روی آن اتفاق می افتد، از هم تشخیص داد. یک کلیک ماوس، رشته ``normal`` را در

`SelectionType`

`property` قرار می دهد و دوبار کلیک رشته ``open`` را در `SelectionType`

`property` جای می دهد.

البته انتخاب چندین گزینه از درون لیست نیز میسر است. اگر اختلاف میان `property`

های `min` و `max` از یک بیشتر باشد، آنگاه انتخاب چند گزینه امکان پذیر است. در غیر اینصورت

تنها یک مورد را می توان از لیست انتخاب کرد. شکل 1-19 نمونه ای از یک `list box` را که

تنها قابلیت انتخاب یک مورد را داراست، نشان می دهد. GUI نشان داده شده در این شکل حاوی

یک `list box` به هشت گزینه، با برچسب های `"option 1"` و `"option 2"` و... است.

به علاوه، این GUI دارای یک `pushbutton` برای انجام عمل انتخاب و یک `text field`

برای نمایش گزینه انتخاب شده، می باشد. `list box` و `pushbutton` هر دو تولید `callback`

می کنند.

Graphical User Interface (GUI)

توابع callback مربوطه در زیر آورده شده است. اگر انتخابی در list box صورت گیرد، آنگاه تابع listbox1_callback اجرا خواهد شد. این تابع شکل تولید کننده این callback را (با استفاده از تابع gcbf) بررسی میکند تا بفهمد عملیات انتخاب با یک کلیک یا دو کلیک انجام شده است. اگر تک کلیک بود، تابع callback کاری انجام نمی دهد، ولی اگر دو کلیک بود، این تابع مقدار انتخاب شده در list box را دریافت می کند و رشته متناسب با آن در text field قرار می دهد.

```
function button1_Callback(hObject, eventdata, handles)
```

```
%Find the value of the listbox  
value = get(handles.listbox1, 'value');
```

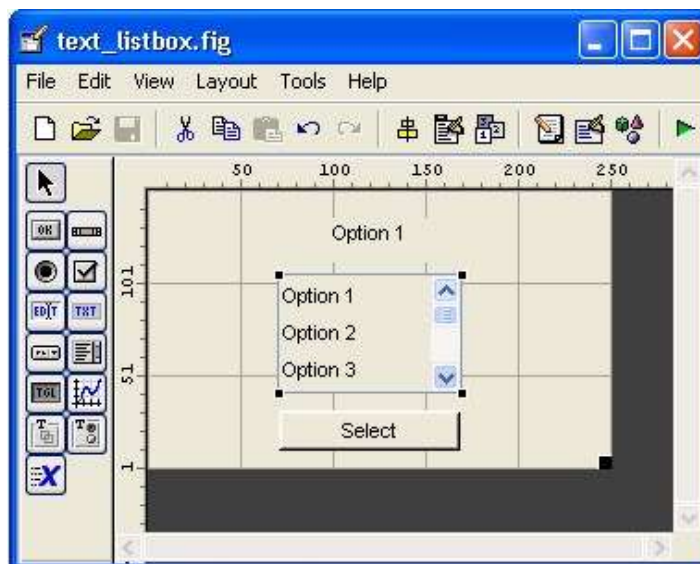
```
%Update text label1 str = ['Option  
' num2str(Value) ]; set  
(handles.Label, 'string', str);
```

```
function listbox1_Callback(hObject, eventdata, handles)
```

```
selectiontype=get(gcbo, 'SelectionType');  
if selectiontype(1) == 'o' %Find the  
value of the listbox  
value = get(handles.listbox1, 'value');
```

Graphical User Interface (GUI)

```
%Update text label1      str =  
['Option ' num2str(Value) ];      set  
(handles.Label,'string',str); end
```



شکل 19-1 یک GUI ساده با یک listbox، یک pushbutton و یک text field. اگر pushbutton انتخاب شود، آنگاه تابع Button1_Callback اجرا خواهد شد. این تابع با دریافت مقدار انتخاب شده از list box، رشته مربوط به آن را درون text field می نویسد. GUI تولید شده بوسیله برنامه test_listbox در شکل 20-1 نشان داده شده است.



Graphical User Interface (GUI)

شکل 20-1 GUI تولید شده به وسیله برنامه

test_listbox 1-4-9 slider ها

slider اشیاء گرافیکی هستند که به کاربر این امکان را میدهند تا مقداری را از

میاندامنه پیوسته‌های از مقادیر، با حرکت یک bar به وسیله ماوس، انتخاب کند. این مقدار بین

مینیمم و ماکسیمم مقادیر پیش فرض تغییر می کند. Value property برای slider مقداری

بین min و max، بسته به موقعیت آن، به خود می گیرد.

یک slider را می توان با ایجاد یک uicontrol که style property

اش `slider` می باشد، ایجاد نمود. البته آنرا می توان به وسیله ابزار slider در Layout

Editor نیز تولید کرد.

شکل 21-1 یک GUI ساده حاوی یک slider و یک text field را نشان می دهد.

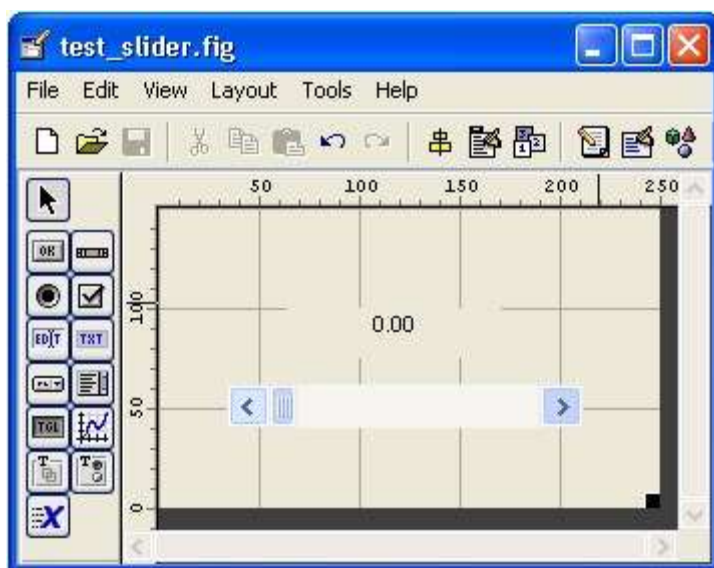
min property برای این slider، صفر و Max property آن، 10 انتخاب شده است.

وقتی کاربر Slider را حرکت میدهد، این عنصر بطور خودکار تابع Slider_Callback را

فراخوانی می کند. این تابع با دریافت مقدار slider از مشخصه `Value` آن، آنرا در text field

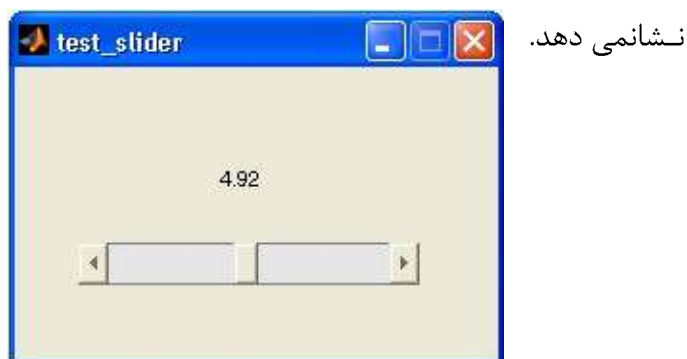
نمایش می دهد.

Graphical User Interface (GUI)



شکل 21-1 شمایی از یک GUI حاوی یک slider و یک text field

شکل 22-1 این GUI را به همراه slider آن که در موقعیت وسط خود قرار دارد،



شکل 22-1 GUI تولید شده به وسیله برنامه test_slider

5-1 Dialog Box ها (جعبه های محاوره ای)

یک dialog box نوع خاصی از اشیاء figure است که از آن برای نمایش اطلاعات

یادریافت ورودی از کاربر، استفاده می شود. dialog box ها معمولاً برای نمایش پیغام های

Graphical User Interface (GUI)

خطا، هشدار، پرسیدن سؤالات و دریافت ورودی از کاربر، مورد استفاده قرار میگیرند. از آنها همچنین برای انتخاب فایل و تنظیم `property` های چاپگر استفاده می شود.

`dialog box` ها می توانند `modal` یا `non-modal` باشند. نوع `modal` آن تا زمانی که باز است و بسته نشده است، اجازه دسترسی به دیگر پنجره های درون برنامه را به کاربر نمی دهد. از این نوع `dialog box` ها معمولاً برای نمایش پیام های خطا و هشدار که به توجه و پاسخ فوری نیاز دارند و نمی توان از آنها بی تفاوت گذشت، استفاده می شود. تمام `dialog box` ها از پیش `non-modal` فرض می شوند.

MATLAB شامل انواع متنوعی از `dialog box` ها است، که مهمترین آنها در زیر به طور خلاصه آورده شده است.

dialog box های منتخب :

dialog : یک `dialog box` بدون عنوان ایجاد می کند.

errordlg : یک پیغام خطا در `dialog box` نشان می دهد. کاربر برای ادامه کار،

باید روی دکمه OK کلیک کند.

helpdlg : یک پیغام `help` در `dialog box` نمایش میدهد. کاربر برای ادامه کار،

باید روی دکمه OK کلیک کند.

inputdlg : یک پیغام که درخواست وارد نمودن داده را مینماید، نمایش می دهد و

مقدار ورودی را از کاربر دریافت می کند.

listdlg : به کاربر اجازه انتخاب یک یا چند گزینه را از یک لیست می دهد.

Graphical User Interface (GUI)

printdlg: یک dialog box ، برای انتخاب چاپگر نمایش می دهد.

questdlg: یک سؤال می پرسد! این dialog box می تواند دارای دو یا سه دکمه

باشد، که بطور پیش فرض Yes و No و Cancel نام گذاری شده اند.

uigetfile: یک dialog box برای باز کردن فایل نمایش می دهد . این پنجره

درحقیقت به کاربر اجازه انتخاب یک فایل را می دهد ولی این فایل را باز نمی کند.

uiputfile: یک dialog box برای ذخیره فایل نمایش می دهد . این پنجره نیز

درحقیقت به کاربر اجازه انتخاب یک فایل را برای ذخیره کردن می دهد ولی آنرا ذخیره نمی کند.

uicolor: یک dialog box برای انتخاب رنگ نمایش میدهد.

uisetfont: یک dialog box برای انتخاب رنگ نمایش می دهد.

warndlg: یک پیام هشدار در یک dialog box نمایش می دهد. کاربر باید برای

ادامه کار، روی دکمه OK کلیک کند.

Warning و Error های Dialog Box 1-5-1

warning dialog box ها و error dialog box ها دارای پارامترهای فراخوانی و

رفتار مشابه هستند. در حقیقت تنها تفاوت آنها در تصویر نمایش داده شده روی آنهاست. متداول

ترین طریقه فراخوانی این dialog box ها به صورت زیر است:

```
errordlg(error_string,box_title,create_mode);
```

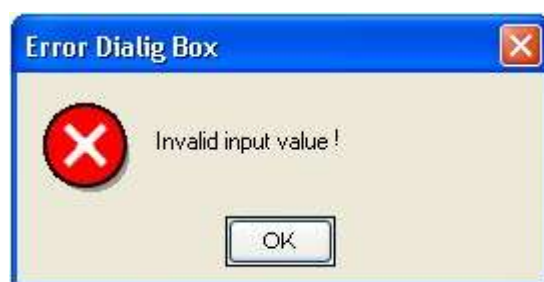
```
warningdlg(warning_string,box_title,create_mode);
```

Graphical User Interface (GUI)

warning_string یا error_string پیغامی است که قرار است به کاربر نشان داده شود، box_title عنوان dialog box می باشد و create_mode هم رشته ای است که بستهبه نوع dialog box می که شما می خواهید ایجاد کنید، 'modal' یا 'non-modal' می باشد.

به عنوان مثال عبارت زیر یک پیغام خطا از نوع modal ایجاد میکند به طوری که کاربر نمی تواند آنرا نادیده بگیرد و از آن بگذرد. dialog box تولید شده به وسیله عبارت زیر، در شکل 1-25 نشان داده شده است.

```
errordlg('Invalid input value !','Error Dialig Box','modal');
```



یک-1-25 شکل error dialog box

2-5-1 Input Dialog Box ها

Graphical User Interface (GUI)

یک input dialog box از کاربر می خواهد که یک یا چند مقدار مورد نیاز برنامه را

وارد کند. input dialog box را می توان با یکی از عبارات زیر ایجاد نمود:

```
answer = inputdlg(prompt) answer =  
inputdlg(prompt,title) answer =  
inputdlg(prompt,title,line_no) answer =  
inputdlg(prompt,title,line_no,default_answer)
```

در اینجا prompt یک آرایه سلولی میباشد. عناصر این آرایه رشتههایی هستند که هر یک

از آنها متناظر با مقداری است که از کاربر خواسته می شود که آنها را وارد کند. پارامتر title

عنوان dialog box را تعیین می کند و line_no تعداد خطوط مجاز برای جواب را مشخص

می کند و آخر از همه، default_answer یک آرایه سلولی، حاوی جواب های از پیش مشخص

شده است و هنگامی مورد استفاده قرار میگیرد که کاربر داده مربوط به گزینه های را وارد نکند.

توجه کنید که جواب های از پیش تعیین شده باید به تعداد prompt ها باشد.

وقتی کاربر روی دکمه OK کلیک می کند، جواب هایی که او وارد کرده است به صورت یک آ

رایه سلولی حاوی رشته های جواب در متغیر answer بازگردانده میشود.

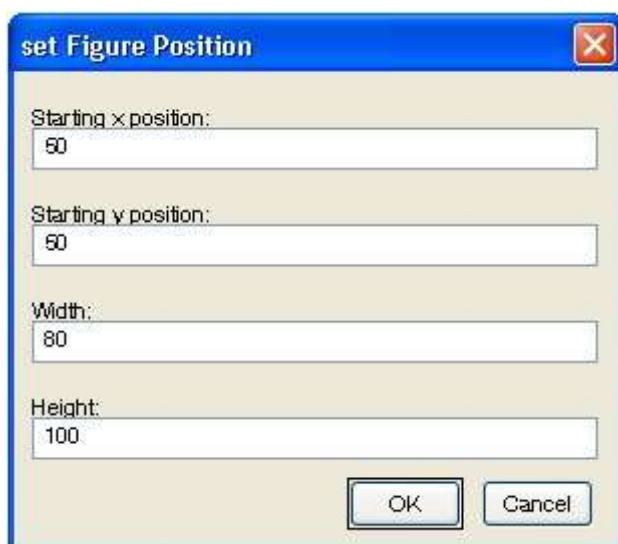
به عنوان مثال فرض کنید که میخواهیم مکان و موقعیت یک figure را با استفاده از

یک input dialog تنظیم کنیم. کد این عملیات به صورت زیر است:

```
prompt{1}='Starting x position:';  
prompt{2}='Starting y position:';  
prompt{3}='Width: '; prompt{4}='Height: ';  
title='set Figure Position';  
default_ans={'50','50','80','100'};  
answer=inputdlg(prompt,title,1,default_ans)  
;
```

Graphical User Interface (GUI)

dialog box حاصل در شکل 1-26 نشان داده شده است.



و `uigetfile` یک 1-26 شکل `input dialog box`

3-5-1 Dialog Box های `uigetfile`

جعبه‌های محاوره‌های `uigetfile` و `uisetfile` به منظور فراهم کردن امکان انتخاب

فایله به طور بصری طراحی شده اند. این `dialog box` ها تنها نام و محل فایل را باز می گردانند و در

واقعفایل را باز و ذخیره نمی کند. این برنامه نویس است که مسؤل نوشتن کد برای ذخیره کردن فایل

است.

عبارات ایجاد کننده این `dialog box` به شکل زیر هستند:

```
[filename , pathname]=uigetfile(filter_spec,title);
```

```
[filename , pathname]=uisetfile(filter_spec,title);
```

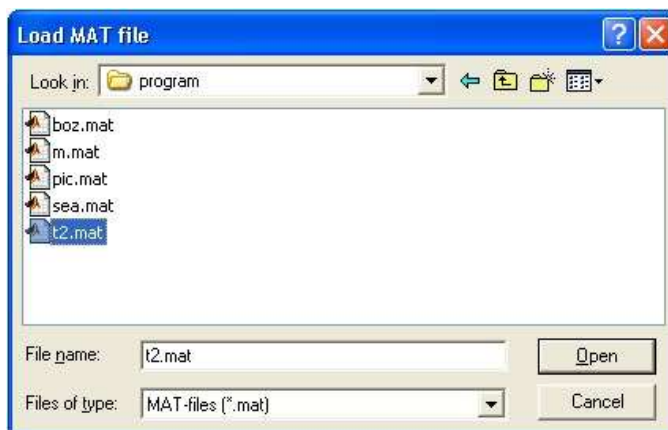
Graphical User Interface (GUI)

پارامتر `filter_spec` یک رشته مشخص کننده نوع فایل‌های نمایش داده شده در `dialog box` است. مثل ``*.m`` و ``*.mat`` و غیره. پارامتر `title`، رشته تعیین کننده عنوان `dialog box` می باشد. بعد از اجرای `dialog box`، `filename` حاوی نام فایل انتخاب شده و `pathname` حاوی مسیر فایل خواهد شد. اگر کاربر دکمه `Cancel` را فشار دهد، مقدار `filename` صفر می شود.

`script file` زیر چگونگی استفاده از این `dialog box` را نشان می دهد.

```
[filename , pathname]=uigetfile('*.mat','Load MAT file');  
    if filename ~= 0    load(  
        [pathname filename])  
    end
```

این عبارت از کاربر درخواست می کند که نام یک `mat-file` را وارد کند و سپس محتوای آن فایل را می خواند. شکل 1-27 `dialog box` ایجاد شده به وسیله این کد را در سیستم عامل `Windows XP` نشان می دهد.



شکل 1-27 یک `dialog box` برای باز کردن فایل ، که به وسیله دستور

`uigetfile` ایجاد شده است.

Graphical User Interface (GUI)

1-6 **Menu ها** را نیز میتوان به GUI در MATLAB اضافه کرد. یک منو به کاربر اجازه انتخابگزینههای را بدون ظهور عنصر دیگری در GUI، میدهد. برای جلوگیری از پر شدن GUI از دکمههای اضافی و برای انتخاب گزینه هایی که کمتر با آنها سر و کار داریم، بهتر است از منوها استفاده کنیم.

در MATLAB دو نوع منو وجود دارد: منوهای استاندارد که در بالای شکل در menu bar قرار دارند و با کلیک روی آنها به پایین میآیند و منوهای Context که وقتی کاربر روی یک شیء گرافیکی دکمه سمت راست ماوس کلیک می کند ظاهر می شوند. منوهای استاندارد بوسیله اشیاء uimenu ایجاد میشوند. هر گزینه در یک منو به همراه گزینههای درون زیر منوی آن، یک شیء uimenu محسوب میشوند. اشیاء uimenu شبیه به اشیاء uicontrol هستند و بسیاری از property های آنها اعم از parent و callback و Enable و ... یکسان هستند.

Property های مهم uimenu : Accelerator : یک کاراکتر مشخص کننده کلید معادل در صفحه کلید برای یک گزینه در منو است. کاربر با فشردن کلیدهای `CTRL + key` به طور همزمان، می تواند گزینه مورد نظر را از طریق صفحه کلید، فعال کند.

Callback : تعیین کننده نام و پارامترهای تابعی است که با فعال شدن گزینه مربوط به آندر منو، فراخوانی میشود. اگر منو، زیر منو نیز داشته باشد، callback آن قبل از ظاهر شدن زیر منو اجرا میشود. اگر منو، زیر منویی نداشته باشد، callback آن به محض اینکه کاربر دکمه ماوس را رها کند، اجرا می شود.

Graphical User Interface (GUI)

Checked : وقتی این property روشن (`on`) باشد، یک علامت تیک (☑) در

سمتچپ گزینه مربوطه در منو، ظاهر میشود. به کمک این ویژگی میتوان منویی ایجاد نمود که بین دو وضعیت معین، تغییر حالت دهد. مقادیر ممکن برای این property، `on` و `off` هستند.

Enable : مشخص میکند که آیا یک گزینه منو قابل انتخاب است یا خیر. اگر یک گزینه

منوبوسیله این property، از کار افتاده باشد، دیگر به کلیکهای ماوس و کلیدهای میان بر پاسخنمی دهد. مقادیر ممکن برای این property، `on` و `off` هستند.

Lable : متن نمایش داده شده روی منو را مشخص میکند. برای اختصاص یک کلید مخففبه

گزینههای از منو میتوان از کاراکتر آمپرسند (&) در ابتدای نام منو، استفاده کرد. این علامت در نام منو

ظاهر نمی شود. به عنوان مثال، رشته `&file` برای label property سبب نمایش

متن `File` روی منو شده و منو را به کلید F حساس می کند.

Parent : handle شی مادر برای گزینه منو است. شی مادر می تواند یک شکل یا

یکمنوی دیگر باشد.

Position : موقعیت و مکان گزینه منو را روی menu bar یا درون منو، مشخص می

کند. موقعیت 1، برای یک منوی سطح بالا، منتهالیه سمت چپ در menu bar و برای زیر

منوها بالاترین موقعیت در منوی دربرگیرنده آنها می باشد.

Seperator : وقتی این Property، `on` است، یک خط بالای این گزینه در منو

ظاهر می شود که آنرا از بقیه جدا می کند. مقادیر ممکن برای آن، `on` و `off` هستند.

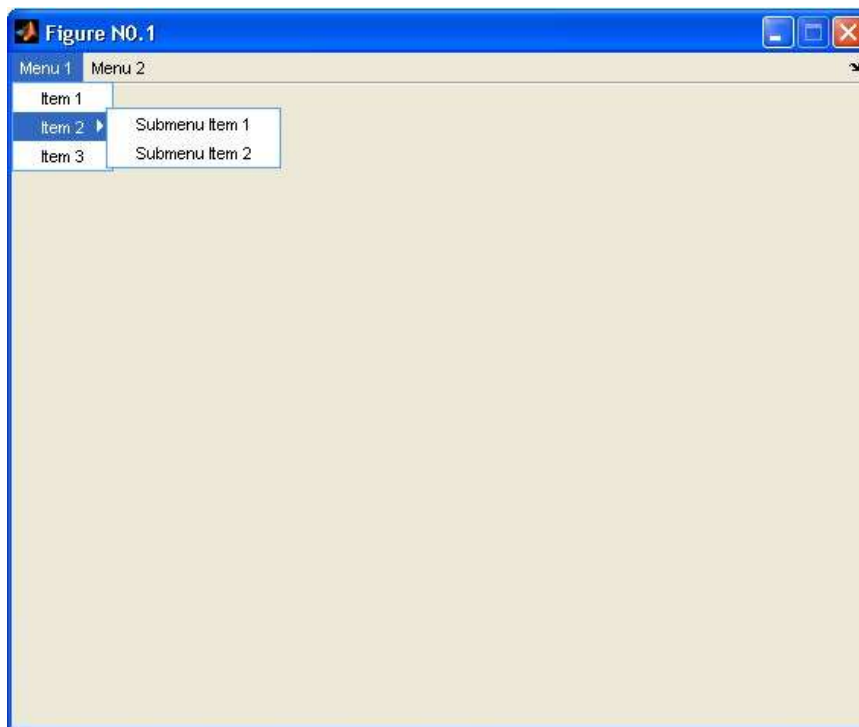
Tag : نام گزینه منو است که برای شناسایی آن استفاده می شود.

Visible : مرئی یا نامرئی بودن یک گزینه منو را تعیین میکند مقدار آن میتواند `on` یا

Graphical User Interface (GUI)

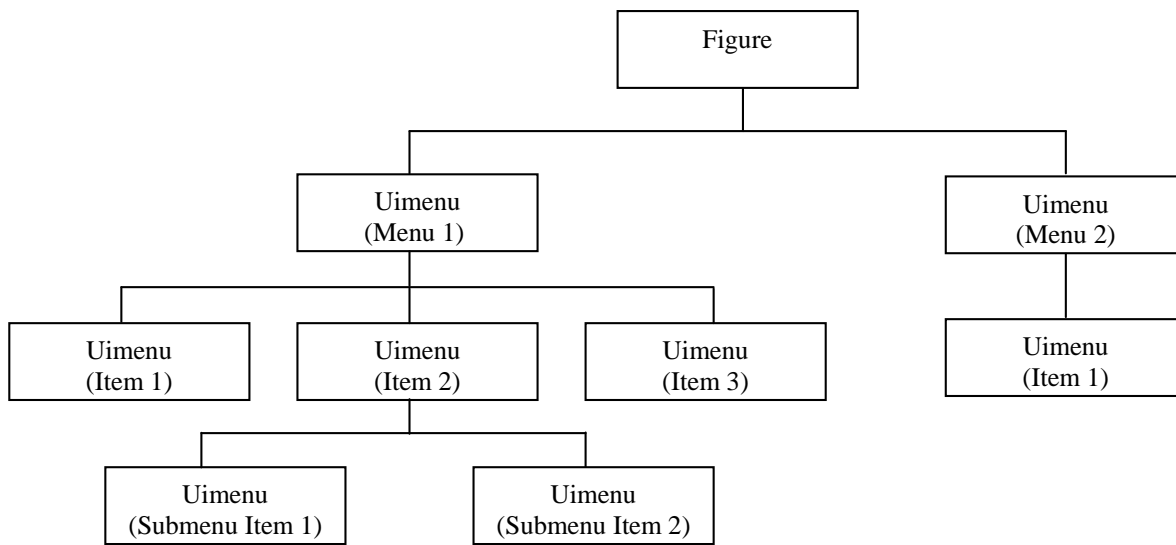
باشد. `off`.

هر گزینهٔ منو به یک شیء مادر متصل است که این شیء مادر برای منوهای سطح بالا همان `figure` و برای زیر منوها، یک منوی دیگر است. تمام `uimenu` هایی که به یک شیء مادر متصل هستند، روی یک منو نمایش داده میشوند و اتصال متوالی گزینهها، یک درخت از زیر منوها بوجود میآورد. شکل (a) 1-28 یک نمونه منو را در حال کار نشان می دهد و شکل (b) 1-28 رابطهٔ بین اشیاء سازنده این منو را نشان می دهد.



(a)

Graphical User Interface (GUI)



(b)

شکل 1-28 (a) ساختاری از منوها و گزینه های آن (b) رابطه بین اجزای تشکیل دهنده منو

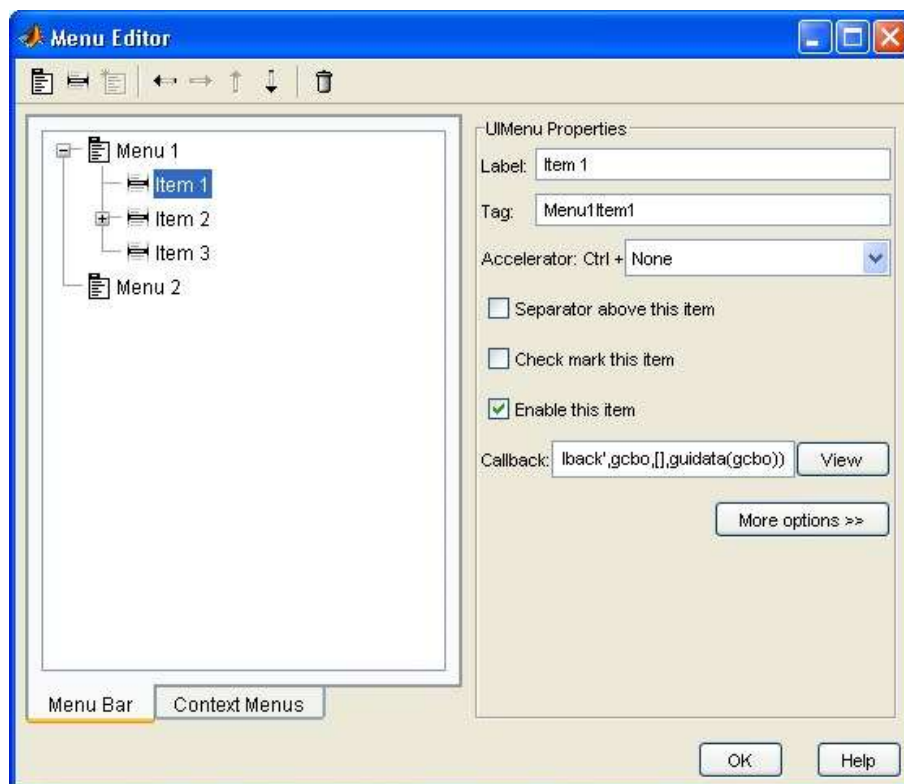
منوهای MATLAB را می توان به کمک Menu Editor ایجاد نمود. شکل 1-29

پنجره Menu Editor را با گزینه های تولید کننده ساختار این منو نشان می دهد. تمام مشخصه

های موجود در Menu Editor نشان داده نمی شوند و برای تغییر آنها باید از Property

Editor (propedit) استفاده نمود.

Graphical User Interface (GUI)



شکل 29- 1 نمایی از Menu Editor ایجاد کننده این منوها

منوهای context سطح بالا با اشیاء uicontextmenu ساخته میشوند و گزینههای سطح پایین در آنها با اشیاء uimenu ایجاد میشوند. اصول و عملکرد منوهای context مشابه با منوهای استاندارد است، جز اینکه می توان آنها را با هر شیء GUI (مثل متن، خط، محورهای مختصات، اشکال) مرتبط کرد. ایستی از مشخصه های مهم اشیاء uicontextmenu در زیر داده شده است. **مشخه های مهم اشیاء uicontextmenu**

callback : نام و پارامترهای تابع فراخوانی شونده هنگام فعال شدن منوی context را تعیین می کند. تابع قبل از نمایش منوی context اجرا می شود.

Graphical User Interface (GUI)

parent : handle شیء مادر برای منوی context **Tag** : نام منوی context

است که از آن برای تعیین موقعیت منو استفاده می شود.

Visible : مرئی یا نامرئی بودن منوی context را تعیین میکند . این مشخصه

بطور خودکار مقدار دهی می شود و معمولاً نباید مقدار آنرا تغییر داد.

1-6-1 از بین بردن اثر منوهای پیش فرض

هر شکل MATLAB مجموعه‌های از منوهای استاندارد به همراه دارد. اگر قصد دارید که این

منوها را پاک کنید و منوها خودتان را بجای آنها بگذارید، بایستی ابتدا منوهای پیش فرض را خاموش

کنید.

نمایش منوهای پیش فرض، بوسیلهٔ MenuBar property ی شکل کنترل میشود. مقادیر

ممکن‌برای این مشخصه، `figure` و `none` هستند. در صورتی که این مشخصه

روی figure تنظیم شود، منوهای پیش فرض نمایش داده میشوند و در صورتی که روی none

تنظیم شود منوهای پیش‌فرض از بین می روند. شما می توانید این کار را با کمک Property

Inspector در هنگام خلق GUI، انجام دهید.

1-6-2 چگونه منوهای مورد نظرمان را بسازیم؟

برای ساختن منوهای استاندارد مورد نظرمان برای یک GUI، باید عملاً سه مرحلهٔ زیر را

طی کنید:

ابتدا به کمک Menu Editor یک ساختار برای منوی جدید ایجاد کنید و پس از تعریف

آن، به هر کدام از گزینه های منو یک Label برای نمایش روی آن و یک Tag یکتا نسبت دهید.

Graphical User Interface (GUI)

بهترین راه برای نوشتن callback برای یک منو، بررسی و مدل کردن callback ی است که بوسیلهٔ یک uicontrol بطور خودکار ایجاد می شود. فرم صحیح یک uimenu callback بصورت زیر است:

```
MyGui ( `MenuItemTag_Callback` , gcbo , [ ] , guidata(gcbo))
```

شما باید نام GUI خودتان را بجای MyGui بنویسید و Tag گزینهٔ منو را بجای MenuItemTag بنویسید.

در قدم بعدی در صورت لزوم مشخصه هر گزینه را با استفاده از Property Editor تنظیم کنید. مهم ترین مشخصه هایی که باید برای یک گزینهٔ منو تنظیم شوند، Label ، Tag Callback آن هستند. که می توان آنها را بدون نیاز به Property Editor از درون Menu Editor تنظیم کرد. با این وجود اگر قصد تغییر مشخصه ها را دارید باید از Property Editor استفاده کنید، تعداد مشخصه هایی را که می توان از داخل Menu Editor تغییر داد در MATLAB نسخهٔ 7 بیشتر شده است و تقریباً دیگر نیازی به Property Editor احساس نمی شود.

قدم سوم، پیاده سازی تابع callback برای انجام عملیات مورد نظر برای هر گزینهٔ منو است. توجه داشته باشید در این مرحله باید توابع callback را خودتان بطور دستی ایجاد کنید.

3-6-1 کلیدهای میانبر و کلیدهای مخفف MATLAB قابلیت کار با کلیدهای میانبر و کلیدهای مخفف را دارد. کلیدهای میانبر در واقعه ترکیبهای "CTRL+Key" هستند که سبب اجرای یک گزینهٔ منو بدون باز کردن منو میشوند. برای مثال کلید میانبر "o" را میتوان به گزینهٔ File/Open

Graphical User Interface (GUI)

اختصاص داد. در این صورت با فشردن همزمان دو کلید CTRL و O تابع callback گزینه File/Open اجرا می شود.

کلیدهای میان بر را می توان با تنظیم Accelerator property در یک شیء

uimenu تعریف کرد.

کلیدهای مخفف حروف تکی هستند که با فشار آنها در صفحه کلید هنگامی که منو باز است

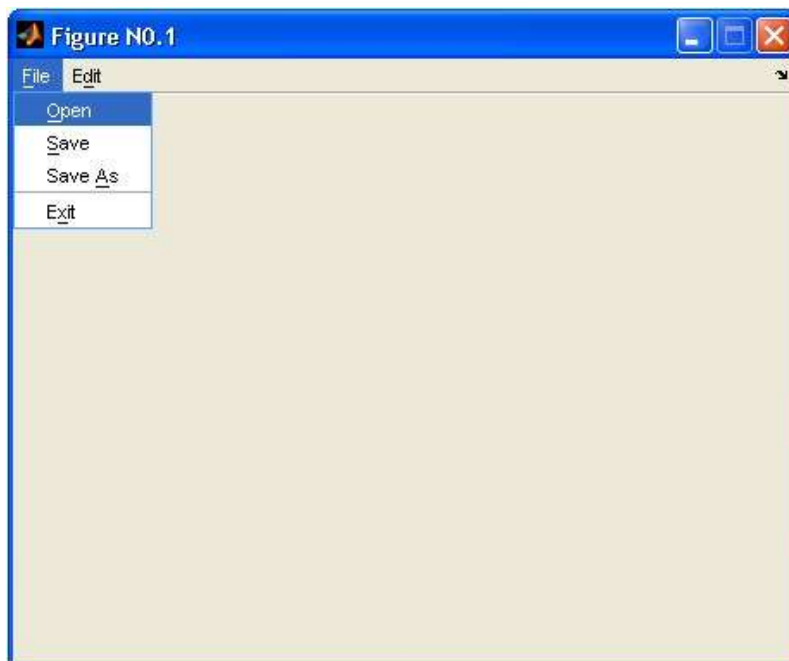
، میتوان گزینه مربوطه را در منو اجرا کرد. زیر این حروف در گزینه مربوطه ، یک خط تیره

کشیده میشود. (شکل 1-30 را ببینید: Open). منوهای سطح بالا (مثلاً در شکل 1-30 ، File

و Edit منوهای سطح بالا تلقی میشوند) را میتوان با فشار حرف مخفف مربوطه به همراه کلید ALT

اجرا کرد.

پس از اینکه منوها باز شدند ، گزینه های درون آنها را می توان تنها با فشار کلید مخفف مربوطه اجرا کرد.



شکل 1 30 چگونه استفاده از کلیدهای مخفف را نشان می دهد. منوی File با فشار کلیدهای

Graphical User Interface (GUI)

ALT+f باز می شود و وقتی باز شد، با فشار کلید "x" می توان گزینه Exit را اجرا نمود.
کلیدهای مخفف را می توان با قرار دادن کاراکتر (&) قبل از حروف مخفف مورد نظر در Label Property تولید نمود. علامت (&) در نام منو ظاهر نمی شود، ولی زیر حرف بعد از آن، در نام منو یک خط تیره ظاهر میشود که در واقع به کاربر می گوید این حرف یک کلید مخفف است.

برای مثال Label property منوی Exit در شکل 1-30 به صورت 'E&xit' است.

1-6-4 ساخت منوهای Context

منوهای Context به طریقی مشابه با منوهای معمولی ایجاد می شوند، جز اینکه گزینه منوی سطح بالا برای آنها یک uicontextmenu است. شیء مادر برای یک uicontextmenu باید شکل (figure) باشد ولی میتوان آنرا به کلیک راست ماوس روی هر شیء گرافیکی حساس نمود.

پس از ایجاد یک منوی context می توان با انتخاب گزینه "context Menu" در Menu Editor ایجاد نمود. پس از ایجاد یک منوی context می توان هر تعداد گزینه در زیر آن قرار داد.

برای اتصال یک منوی context به یک شیء گرافیکی شما بایستی Uicontextmenu Property آن شیء را با handle منوی context مقدار دهی کنید. معمول است که این کار را با Property Inspector انجام می دهند ولی انجام آن با فرمان set نیز امکان پذیر است.

Graphical User Interface (GUI)

(همانطور که در زیر نشان داده شده است) اگر Hcm ، handle یک منوی context باشد ، عبارت زیر این منوی context را به یک خط که به وسیله plot به وجود آمده ، مرتبط می سازد .

```
H1=plot(x,y); set(H1,  
`UiContextmenu` , Hcm) ;
```

1-7 نکاتی برای خلق GUI های کارآمدتر

در این بخش چند نکته دیگر برای GUI های کارآمدتر آورده شده است.

1-7-1 tool tips ها پنجره های کمکی کوچکی هستند که هنگام نگاه داشتن اشاره

گر ماوس روی یک شیء uicontrol خودبه خود ظاهر میشوند و از آنها برای راهنمایی سریع کاربر درباره عملکرد آن شیء استفاده می شود.

یک tool tip را می توان با ا ق راردادن متن ی ک ه ق رار است نمایش داده شود در

property tooltipstring برای یک شیء ایجاد نمود.

1-7-2 Pcode

MATLAB هنگامی که در طول اجرای یک برنامه، تابعی را برای بار اول اجرا میکند، آنرا به یک

کد واسط به نام pcode کامپایل می کند و سپس این pcode را در run-time

interpreter خود اجرا میکند. پس از اینکه تابع برای بار اول کامپایل شد، در

حافظه MATLAB باقی میماند و میتوان آنرا بارها بدون نیاز به کامپایل مجدد، اجرا نمود. با این

وجود، اگر MATLAB بسته شود، دفعه بعد تابع باید دوباره کامپایل شود.

ضرری که کاربر بابت این کامپایل اولیه میبیند برای برنامههای کوچک محسوس نیست ولی

باافزایش اندازه و حجم توابع، زمان کامپایل اولیه به مراتب افزایش مییابد. از آنجا که توابع تعریف

کننده یک GUI معمولاً بزرگ هستند، زمان کامپایل کردن برنامههایی که بر اساس GUI طراحی

Graphical User Interface (GUI)

شده‌اند، به‌مراتب از انواع دیگر برنامه‌ها بیشتر است. به بیان دیگر، برنامه‌های GUI بسیار کند اجرا میشوند.

خوشبختانه، یک راه برای رهایی از این مشکل وجود دارد. به این صورت که فایل‌های نوشتاری و توابع MATLAB را میتوان به pcode کامپایل کرد و فایل pcode حاصل را برای اجرای سریع برنامه‌ها آینه ذخیره نمود. فایل‌های pcode سبب میشوند که برنامه بدون نیاز به انجام کامپایل اولیه، با سرعت بیشتری اجرا شود.

MATLAB با دستور pcode فایل‌های pcode را تولید میکند. این دستور یکی از دو شکل

زیرا به خود می‌گیرد:

```
pcode fun1.m fun2.m fun3.m . . .  
pcode *.m
```

شکل اول این دستور، فایل‌های نام برده شده را کامپایل میکند و شکل دوم آن تمام M-File

درون مسیر کنونی را کامپایل میکند. فایل خروجی کامپایل شده، با پسوند ".p" ذخیره میشود.

برای مثال، اگر شما فایل foo.m را کامپایل کنید خروجی عملیات در فایل foo.p ذخیره می‌شود.

اگر یک تابع در دو فایل هم نام یکی با پسوند P-File و دیگری با پسوند M-File

وجود داشته باشد، MATLAB بطور خودکار نسخه P-File را اجرا میکند. زیرا که سریعتر اجرا

خواهد شد.

با این وجود اگر M-File را تغییر دهید باید به خاطر داشته باشید که آنرا بطور دستی دوباره

کامپایل کنید، در غیر این صورت برنامه، کد قدیمی را اجرا می‌کند.

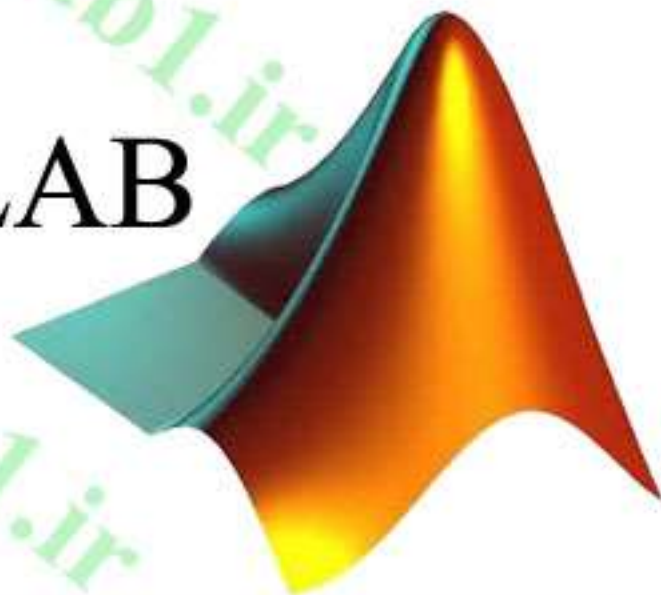
کامپایل کردن فایلها به pcode یک مزیت دیگر نیز دارد. شما میتوانید حاصل زحمات خود را که همان کد برنامه است در عرضه برنامه به دیگران، از گزند تغییرات و لو رفتن ایدههایتان محافظت کنید. pcode را میتوان به راحتی اجرا شود ولی دیدن کد درون آنها و ایجاد تغییرات در آنها از عهده هر کسی بر نمی آید.

دوره جامع آموزش

برنامه نویسی

متلب

MATLAB



گروه برنامه نویسی ایران متلب MATLAB1 
از حرفه ای ها متلب را یاد بگیرید