۱- رابط گرافیکی برای کاربر (GUI)

GUI (رابط گرافیکی برای کاربر) نوعی رابط تصویری برای برنامه است که نمونهٔ خوب آن می‌تواند با فراهم کردن شکل و صورتی ثابت برای برنامه و همچنین با کنترلگرهای آشنا، مثل pushbuttons (دکمه‌های فشاری)، list boxes (جعبه‌های لیست) و sliders و menus (منوها) و مانند اینها استفاده از برنامه را آسان‌تر کند. رابط گرافیکی باید رفتاری قابل فهم و پیش‌بینی داشته 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 دربارهٔ خصوصیات همه اجزای رابط‌های گرافیکی نیست، ولی اصول کلی لازم برای ایجاد GUI های کاربردی برای برنامه‌های کاربران، در آن گنجانیده شده است.

۱-۱ یک GUI چگونه کار می‌کند؟

رابط گرافیکی (GUI) محیطی آشنا برای کاربر فراهم می‌کند. این محیط حاوی pushbutton ها، togglebutton ها، list ها، menu ها، text box ها، و . . . می‌باشد که برای همهٔ 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 برنامه‌ای که بر پایهٔ GUI طراحی شده باید در هر زمان آمادهٔ ورودی‌های ماوس و (یا احتمالاً ورودی‌های کیبرد) روی هر یک از عناصر خود باشد. این ورودی‌ها به event ها معروفند. برنامه‌ای که به این event ها پاسخ گوید، event driven نامیده می‌شود.

سه عنصر اساسی لازم برای ایجاد رابط گرافیکی (GUI) MATLAB عبارتند از:

۱- اجزا (Components)

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

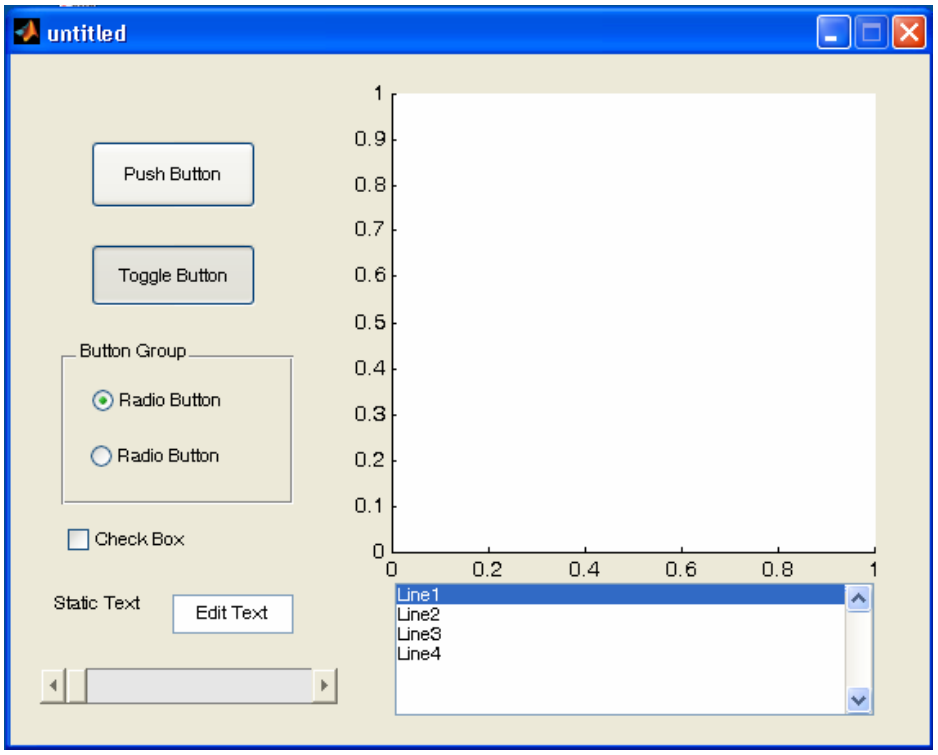
۲- اشکال (Figures)

اجزای GUI باید درون یک figure مرتب شوند، که پنجره‌ای 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.

۳- فراخوان‌ها (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 برنامهٔ 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 یه callback وجود داشته باشد.

عناصر اصلی GUI ها در زیر به خلاصه و نمونه‌هایی از آنها در شکل ۱-۱ 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.



شکل ۱-۱ یک پنجرهٔ Figure نشان دهندهٔ 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 می‌کند.

Toggle button : (uicontrol) : جزئی گرافیکی است که کار یک کلید دو حالته 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 می‌شود.

Radio button : (uicontrol) : نوعی از toggle button هاست که به صورت دایرهٔ کوچکی است و هنگام "روشن" بودن نقطه‌ای در مرکز آن قرار می‌گیرد. گروهی از radio button ها را می‌توان برای پیاده‌سازی گزینه‌های مستقل استفاده کرد. هر کلیک ماوس روی این جزء callback آن را فعال می‌کند.

Check box : (uicontrol) : check box نوعی از toggle button ها است که به شکل مربعی کوچک با علامت تیک (✓) در درون آن به منزلهٔ "روشن" بودن می‌باشد. هر کلیک ماوس روی آن، callback آن را فعال می‌کند.

Edit box : (uicontrol) : edit box متنی را نمایش می‌دهد و به کاربر اجازه می‌دهد اطلاعات نشان داده شده در آن را تغییر دهد. Callback مربوط به آن با فشار دکمه enter فعال می‌شود.

List box : (uicontrol) : کنترلی گرافیکی است که یک سری از متن‌های رشته‌ای (text string) را نمایش می‌دهد کاربر می‌تواند با یک یا دو بار کلیک روی هر یک از این متن‌های رشته‌ای آنها را انتخاب کند. به هنگام انتخاب یک متن رشته‌ای callback آن فعال می‌شود.

Popupmenu : (uicontrol) : کنترلی گرافیکی است که در پاسخ به کلیک ماوس یک دسته از متن‌های رشته‌ای را نمایش می‌دهد. تا هنگامی که روی یک منوی popup کلیک نشده است، تنها رشتهٔ انتخاب شده فعلی آن قابل مشاده است.

Slider : (uicontrol) : slider کنترل گرافیکی دیگری است که نقش آن تنظیم یک مقدار به طور منظم و پیوسته با کشیدن کنترل آن به وسیله ماوس است. هر تغییر در slider ، callback اش را فعال می‌کند.

Frame : (uicontrol) : یک قاب ایجاد می‌کند که در حقیقیت جعبهٔ مربعی شکل درون figure می‌باشد. قابها برای گروه بندی مجموعه‌ای از کنترل‌های گرافیکی استفاده می‌شوند. قابها هرگز callback ی را فعال نمی‌کنند.

Text field : (uicontrol) : برچسبی (label) ایجاد می‌کند که متنی رشته‌ای واقع در نقطه‌ای روی figure است. text field ها هرگز callback ی را فعال نمی‌کنند.

Menu items : (uimenu) : یک منو ایجاد می‌نماید و callback این منوها به هنگام کلیک ماوس روی آنها فعال می‌شود.

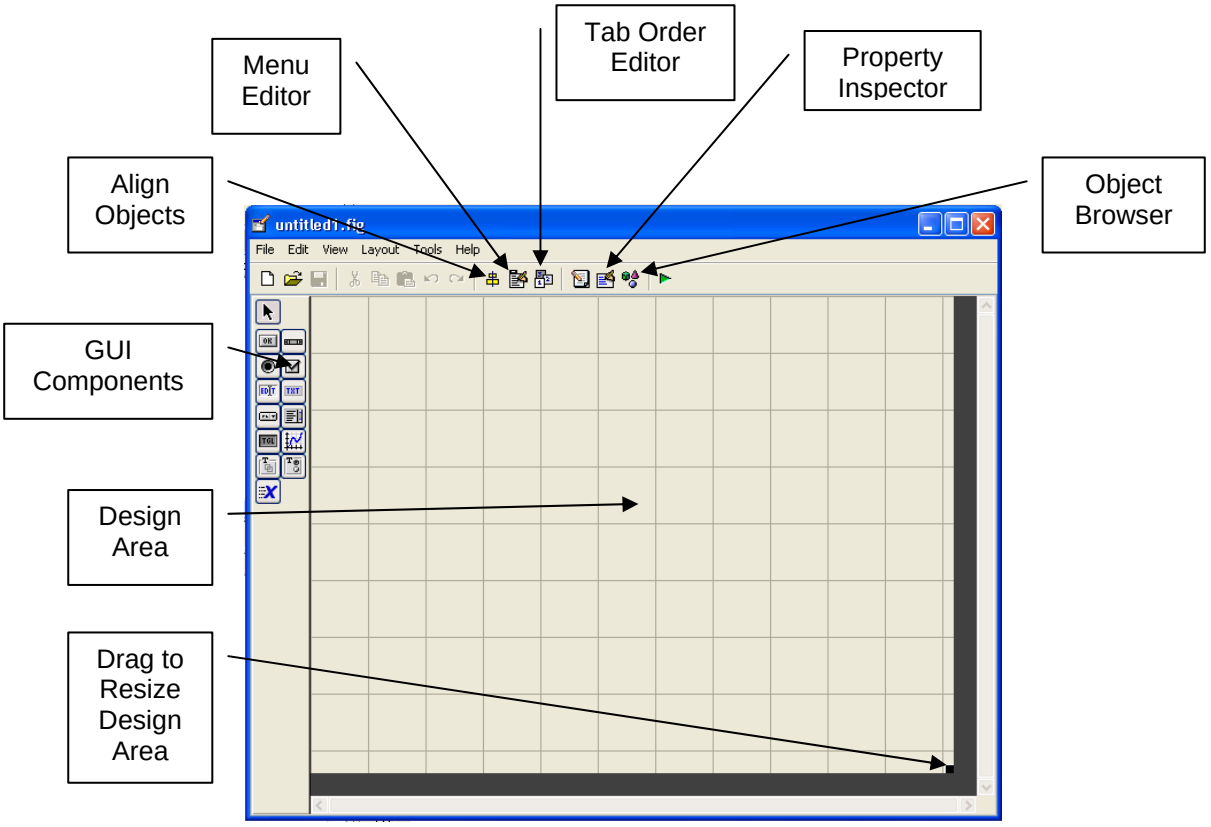
Context menu : (uicontextmenu) : یک منوی context ایجاد می‌نماید. هنگامی که کاربر روی شیء مورد نظر با کلیک سمت راست ماوس، کلیک می‌کند این منو ظاهر می‌شود.

Axes : (axes) : یک دستگاه مختصات برای نمایش اطلاعات در آن، ایجاد می‌کند. Axes ها هیچ گاه callback ی را فعال نمی‌کنند.

۲-۱ ایجاد و نمایش یک GUI

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 نوشتهٔ روی آن و . . . ویرایش کند. هنگامی که guide یک GUI را ذخیره می‌کند، برنامه‌ای حاوی مجموعه‌ای 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 ۲-۱ نیز به همراه آن ظاهر می‌شود.



شکل ۲-۱ پنجرهٔ ابزار guide

ناحیهٔ روشن چهار خانه، layout (کادر و ناحیهٔ طراحی) نام دارد، ناحیه‌ای که برنامه نویس، GUI را در آن طراحی می‌کند. در قسمت چپ پنجرهٔ layout editor ، مجموعه‌ای 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 ناحیهٔ طراحی پخش کند و یا خصوصیات (propertise) این اجزا را تغییر داده و یا به GUI منو اضافه کند یا . . .

گام‌های 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.

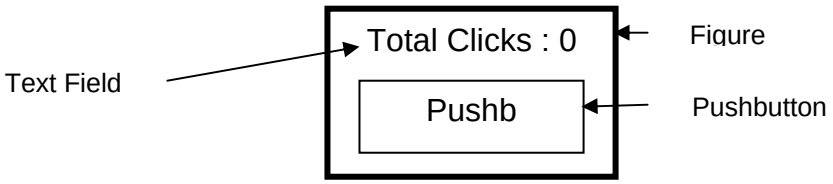
۳- از یکی دیگر از ابزارهای MATLAB به نام Property Inspector (واقع درون guide) استفاده کنید تا به هر کدام از اجزا، یک لقب(یک "tag ") نسبت دهید و ویژگی‌های هر یک را که شامل رنگ، متن نمایش داده شده، غیره می‌باشد، تنظیم نمائید.

۴- figure را در یک فایل ذخیره کنید. بعد از اینکه figure زا 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 دیگرM-File می‌باشد، که حاوی کد آن و بدنهٔ callback های مربوط به عناصر GUI است.

۵- کدی بنویسید که رفتار مربوط به هر تابع callback را انجام می‌دهد.

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 است را نمایش دهد.

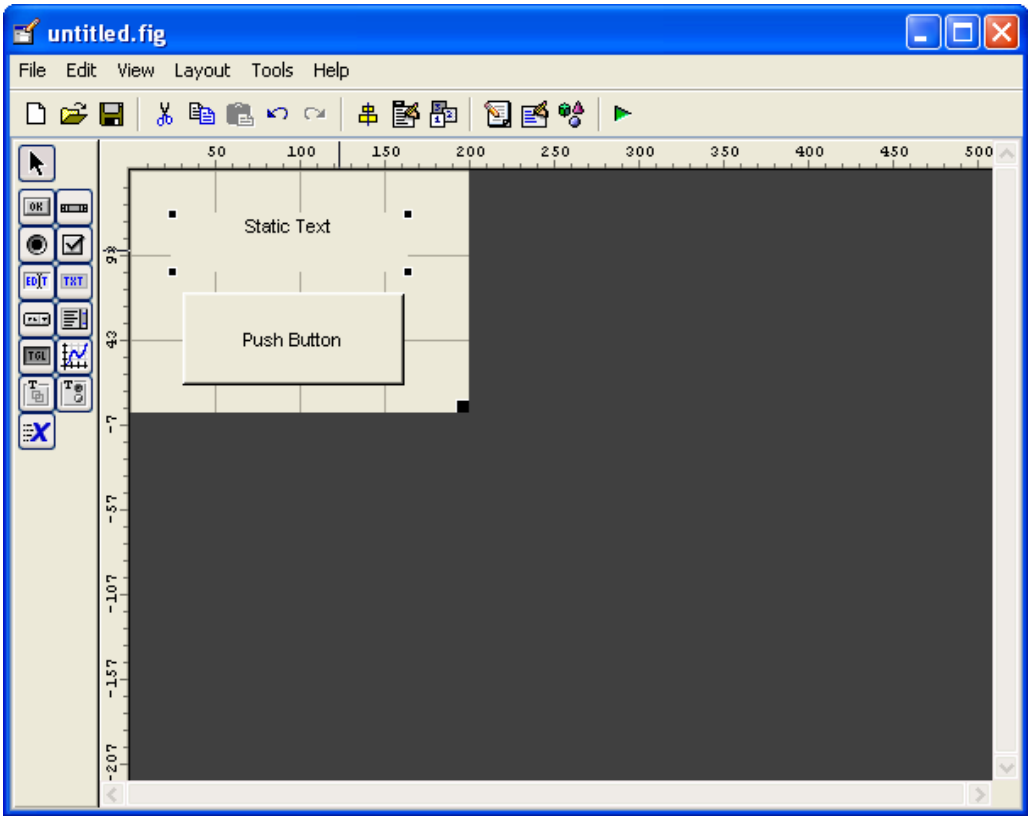
قدم اول: طراحی این GUI بسیار ساده است. این GUI شامل یک دکمهٔ فشاری و یک text field است callback مربوط به pushbutton سبب خواهد شد که عدد داخل text field با فشار کلیک یک واحد اضافه شود. طرح اولیه از این GUI در شکل ۱-۳ نشان داده شده است.



شکل ۳-۱ 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 ۲-۱ ظاهر می‌شود.

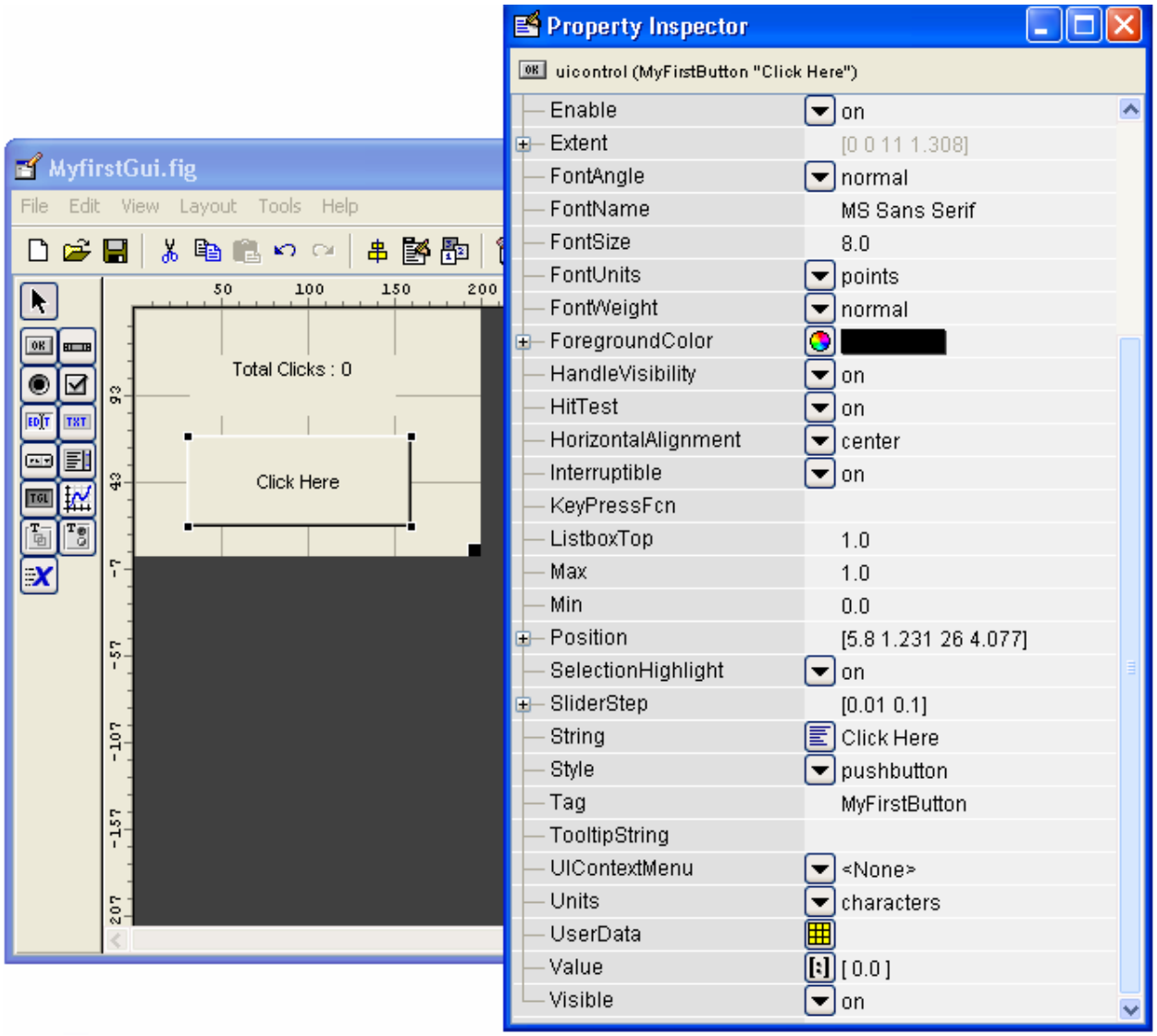
در ابتدای امر باید اندازهٔ ناحیهٔ طراحی (کادر GUI) را تنظیم کنیم که در واقع، اندازه GUI نهایی خواهد بود. این کار را با کشیدن مربع کوچک واقع در گوشهٔ پایین سمت راست ناحیه طراحی تا رسیدن به اندازه و شکل دلخواه، انجام می‌دهیم. سپس، در لیست اجزای GUI روی گزینه "pushbutton" کلیک و آن را در ناحیه طراحی قرار می‌دهیم. شکل حاصل از انجام این مراحل در شکل ۴-۱ دیده می‌شود. حال می‌توانیم با استفاده از ابزار Alignment جایگاه و هم راستایی این دو عنصر را بطور دلخواه تنظیم کنیم.



شکل ۴-۱ کادر GUI نهایی درون پنجرهٔ guide

قدم سوم: برای تنظیم property های دکمهٔ 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 دکمهٔ فشاری با ماوس right-click کرده و در منویی که ظاهر می‌شود،

گزینهٔ "Inspect Properties" را انتخاب کنید. پنجرهٔ "property Inspector" همان طور که در شکل ۵-۱ نشان داده شده است، ظاهر خواهد شد. توجه کنید که این پنجره، لیستی از تمام property های pushbutton نمایش می‌دهد و می‌توان مقدار هر یک از آنها را تغییر داد. "Property Inspector" در واقع همان توابع set و get را انجام می‌دهد، البته به صورت کاملاً ساده و قابل فهم.



شکل ۵-۱ پنجرهٔ property Inspedtor نشان دهندهٔ property های یک دکمهٔ فشاری که string آن روی Click Here وTag آن روی MyFirstButton تنظیم شده است.

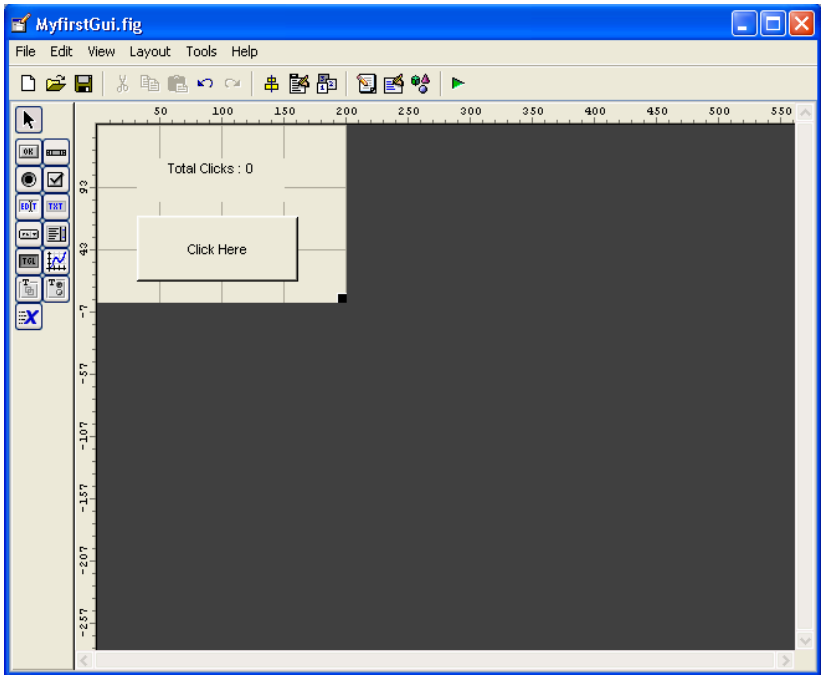
در مورد Pushbutton می‌توان property های زیادی مانند رنگ، انـدازه، فونـت، جایگـاه متن نمایش داده شده روی آن و . . . را تنظیم کرد، ولی دو مورد زیر ضروری هستند :

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

Tag property : که نام دکمه فشار می‌باشد.

در ایـن مـورد مثـال مقـدار string دکمـه فـشاری " click here" و Tag آن نیـز بـه `My First Button` تغییر داده می‌شود.

بـرای text field نیـز بایـد دو property را تنظـیم کنـیم؛ کـه یکـی از آنهـا String property می‌باشد که دربرگیرنده متنی است که قرار است نمـایش داده شـود و دیگـری Tag property است که نام text field مـی‌باشـد. در واقـع تـابع callback بـه ایـن نـام احتیاج دارد تا text field را پیدا کند و متن درون آن را تغییر دهد. در ایـن مثـال String ایـن عنصر به `Total click:0` و Tag آن به `My First Text` تغییـر داده شـده‌انـد. ناحیـه طراحی پس از طی این مراحل در شکل ۶-۱ نشان داده شده است.

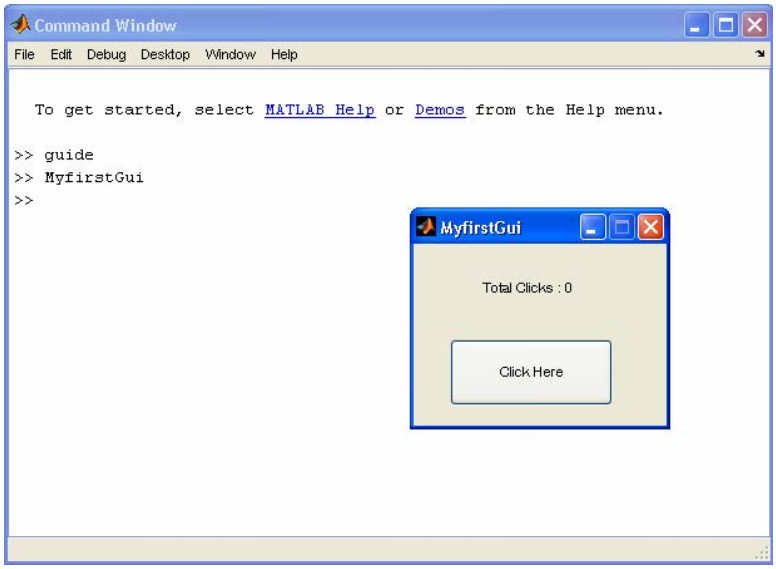


شکل ۶-۱ کادر طراحی پس از تنظیم property های یک pushbutton و textfield.

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 رشتهٔ درون Name property هنگام اجرای GUI به صورت عنوان در بالای پنجره شکل ظاهر می‌شود. البته این کار ضرورتی ندارد و تنها به زیبایی کار کمک می‌کند.

قدم چهارم: هم اکنون ناحیه طراحی را تحت نام MyFirstGUI ذخیره می‌کنیم. برای این کار از منوی File ، گزینهٔ MyFirstGUI را به عنوان نام فایل تایپ کنید، سپس روی دکمه Save کلیک کنید، این عمل به طور اتوماتیک دو فایل به نام‌های 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.

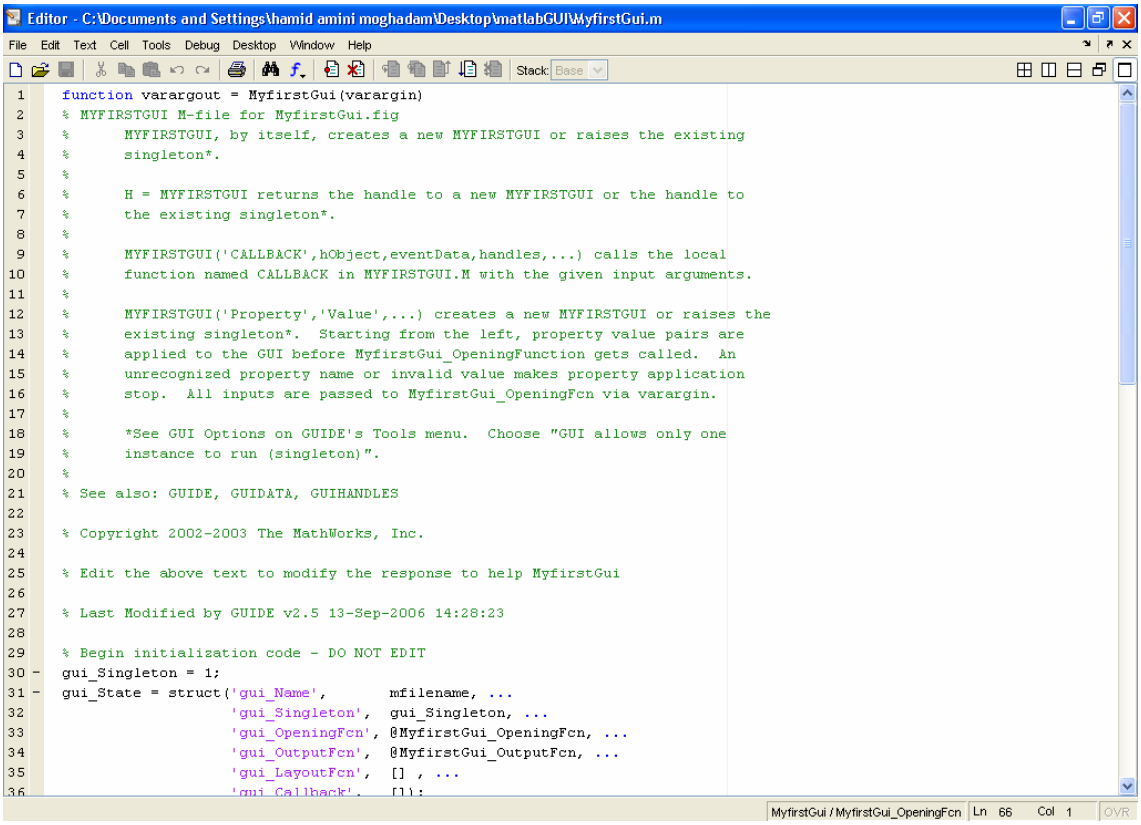
هم اکنون این GUI کامل شده است ولی هنوز کاری را که بدان محول شده انجام نمی‌دهد. این GUI را می‌توان با تایپ MyFirstGUI در پنجره فرمان، همان طور که در شکل ۷-۱ دیده می‌شود، اجرا کرد. اگر در این GUI روی دکمه کلیک شود، پیغام زیر در پنجرهٔ فرمان ظاهر می‌شود.



شکل ۷-۱ 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 ۸-۱ نشان داده شده است.



شکل ۸-۱ 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 می‌کند و بقیه آرگومان‌ها را به آن تابع می‌فرستد.

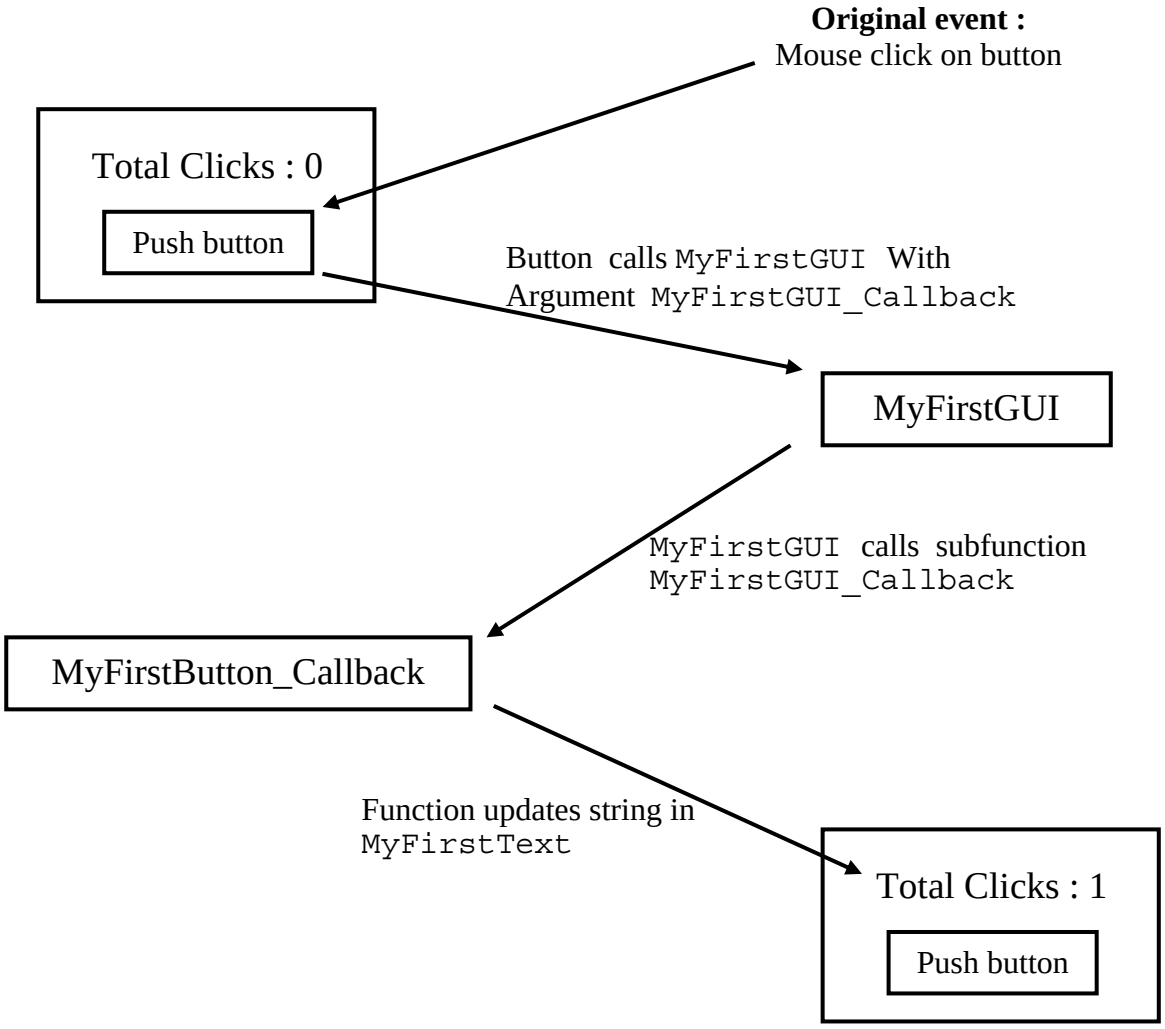
وظیفه هر تابع callback ادارهٔ پیش آمدهای یک عنصر GUI است. اگر کلیک ماوس روی یک جزء GUI (یا ورودی صفحه کلید برای Edit Field ها) اتفاق بیافتد، آنگاه تابع callback

مربوط به آن جزء، به طور خودکار توسط MATLAB فراخوانی می‌شود. نام تابع callback همان مقدار Tag property برای آن جزء GUI به اضافهٔ 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 ballback ها فقط پیغامی را نمایش می‌دهند مبنی بر اینکه هنوز چیزی در تابع callback منظور نشده است.

قدم پنجم: اکنون، وقت پیاده‌سازی callback مربوط به دکمهٔ فشاری فرا رسیده است، این تابع شامل یک متغیر دائمی است که برای شمارش تعداد کلیک‌های انجام شده به کار می‌رود. وقتی روی pushbutton کلیک می‌شود، MATLAB تابع MyFirstGUI را با MyFirstButton_Callback به عنوان آرگومان اول آن فراخوانی می‌کند. سپس همان طور که در شکل ۹-۱ 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 رشته‌ای جدیدی با این عدد جدید بسازد. سپس باید رشتهٔ جدید را در String property مربوط به MyFirstText ذخیره کند. تابعی که این مرحله را انجام می‌دهد در زیر آورده شده است:

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

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



شکل ۹-۱ ادارهٔ وقایع در برنامهٔ 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 می‌شود، مقدار count را یک واحد افزایش داده و رشتهٔ جدیدی حاوی مقدار جدید count ایجاد می‌کند. سپس تابع، رشتهٔ نمایش داده شده در جای متن MyFirstText را update می‌کند.

برنامهٔ نهایی با تایپ MyFirstGUI در پنجرهٔ فرمان اجرا می‌شود. هنگامی که کاربر روی دکمه کلیک کند، MATLAB به طور اتوماتیک تابع MyFirstGUI را با MyFirstButton_Callback به عنوان آرگومان اول فراخوانی می‌کند 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 می‌دهد. GUI حاصل بعد از سه فشار دکمه در شکل ۱۰-۱ نشان داده شده است.

شکل ۱۰-۱ برنامه حاصل بعد از سه بار فشار کلید

۱-۲-۱ نگاهی عمیقتر

شکل ۸-۱ 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 می‌کند.

ابتدا بیایید نگاه دقیق‌تری به طرز اعلان function بیاندازیم. توجه داشته باشید که این تابع از متغیر varargin برای معرفی آرگومان‌های ورودی‌اش و از varargout برای معرفی نتایج خروجی‌اش 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

اگر کاربر MyFirstGUI را بدون آرگومان فراخوانی کند، مقداری که توسط nargin برگرداننده می‌شود، صفر خواهد بود. در این صورت، برنامه با استفاده از تابع openfig ، GUI درون فایل MyFirstGUI.fig را باز می‌کند. 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 این تابع 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 گزینهٔ `reuse` فراخوانی شود در حالی که همان شکل از قبل وجود داشته باشد، آنگاه شکل موجود، بدون تغییر به بالای صفحه کامپیوتر برده می‌شود. در مقابل، اگر این آرگومان `new` باشد، چندین نسخه از شکل می‌تواند، در آن واحد اجرا شود. هر بار که openfig با گزینهٔ `new` فرا خوانده شود، نسخهٔ جدید از شکل ایجاد خواهد شد. به طور پیش فرض، GUI ی ایجاد شده توسط guide ، گزینهٔ `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 می‌کند:

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

این تابع رنگ پس زمینهٔ شکل را با رنگ پیش فرض پس زمینهٔ مورد استفاده بوسیله کامپیوتری که MATLAB روی آن در حال اجرا است، تطبیق می‌دهد. در واقع این تابع، رنگ GUI را با رنگ دیگر پنجره‌های کامپیوتر یکی می‌کند، بنابراین یک GUI را می‌توان روی کامپیوترهای با ویندوز 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 داده‌ای منحصر به فرد به خود شکل در آن ذخیره می‌کنند.

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 علاوهٔ یک text field و یک pushbutton. تابع guidata ساختار handles را به عنوان داده‌ای مربوط به شکل در آن ذخیره می‌کند و این کار را تابع setappdata انجام می‌دهد.

عبارت پایانی در این GUI، در صورت اختصاص آرگومان خروجی در هنگام فراخوانی MyFirstGUI، ساختار handles را به فراخوان باز می‌گرداند.

```
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 آن را توسط تابع fevel اجرا می‌کند. تابع fevel تابعی را که نام آن در varagin{1} آمده است اجرا می‌کند و بقیه آرگومان‌ها را (varagin{2} , varagin{3} و غیره) به آن تابع می‌فرستد. این مکانیزم سبب می‌شود که توابع callback به زیر توابعی تبدیل شوند بطوریکه امکان فراخوانی اتفاقی آنها از جایی دیگر خارج از M-File وجود نداشته باشد.

۲-۲-۱ 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 می‌باشد). آرگومان‌های این زیر تابع عبارتند از:

hObject- که handle شکل مادر (parent) می‌باشد

eventdata- در نسخه فعلی MATLAB از این آرایه استفاده نمی‌شود.

handles- ساختار handles حاوی تمام handle های اجزای درون شکل می‌باشد.

varagin- یک آرگومان اضافی برای فرستادن آرگومان‌های دیگر به تابع callback می‌باشد. برنامه‌نویس در صورت نیاز می‌تواند از این ویژگی برای ارائهٔ 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 دکمهٔ فشاری در برنامه MyFirstGUI ، در آنجا که می‌خواستیم تابع callback دکمهٔ فشاری، متن نمایش داده شده در text field را تغییر دهد، بهره گرفتیم:

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

۳-۲-۱ 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 دادهٔ global مربوط به GUI را تغییر دهند و چند نسخهٔ یکسان GUI که در یک زمان اجرا می‌شوند نیز نمی‌توانند در کار یکدیگر خلل ایجاد کنند.

برای اضافه کردن دادهٔ محلی به ساختار handles ، باید M-File را پس از ایجاد آن با دستور guide ، به طور دستی تغییر داد. برنامه نویس باید دادهٔ مربوط به برنامه را (application data) بعد از فراخوانی guihandles و قبل از guidata به ساختاری handles اضافه کند. به عنوان مثال، برای اضافه کردن تعداد کلیک‌های ماوس (count) به ساختار handles ، برنامه را به صورت زیر تغییر می‌دهیم:

```
%Generete a structure of handles to pass to callbacks
Handles = guihandles(fig);

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

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

اکنون دادهٔ مربوط به برنامه، همراه ساختار handles بـه تمـام توابـع callback فرسـتاده می‌شود و در جای لازم از آن استفاده می‌شود.

نسخه‌ای از تابع callback دکمهٔ فشاری که از مقدار متغیر count در سـاختار handles استفاده می‌کند، در زیر آورده شـده اسـت. توجـه کنیـد کـه هرگونـه تغییـر در اطلاعـات درون سـاختار 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);

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

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

۴-۲-۱ چند تابع مفید دیگر

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

```
gcbo
gcbf
findobj
```

تابع gcbo (*get callback object*)، handle شیای را که آنcallback را ایجاد کرده است، باز می‌گرداند و تابع gcbf (*get callback figure*)، handle شکل حاوی آن شیء را باز می‌گرداند. این توابع می‌توانند برای تعیین شیء و شکل بوجود آورندهٔ 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 گزینهٔ مورد جستجو منطبق باشد، handle آنها را بر می‌گرداند. فرم معمول این تابع به 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.

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

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

۱-۳ 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 می‌کند. Property های کلیهٔ اشیاء در Online Help Browser ثبت شده‌اند 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 می‌توان با ابزار Property Inspector و یا توابع get و set تغییر داد. ولی با اینکه استفاده از Property Inspector برای تنظیم property های یک شیء آسان‌تر است، برای تنظیم property اشیاء از درون برنامه، مثلاً از درون یک تابع callback، باید از توابع get و set استفاده کنیم.

۱-۴ اجزای GUI

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

Text Field(جای متن)

Edit Boxes

Frames(قاب‌ها)

Pushbuttons(دکمه‌های فشاری)

Toggle Button(دکمه‌های دو حالته)

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 کنندهٔ نسبت به سه رنگ اصلی قرمز، سبز و آبی با مقیاس بین ۰- ۱ . مثلاً رنگ magenta با بردار [1 0 1] مشخص می‌شود.

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

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

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 صفحهٔ مانیتور در مقیاسی که در property `units` تعیین شده است، مشخص می‌کند. 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 می‌باشند.

SelectionType : نوع انتخاب را برای آخرین کلیک ماوس روی شکل، تعیین می‌کند. تک کلیک ماوس نوع `normal` و دو بار کلیک نوع `open` را باز می‌گرداند. گزینه‌های دیگری نیز وجود دارند. برای دیدن آنها به پوشه‌های Online در MATLAB رجوع کنید.

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

Units : مقیاس و واحدی است ک شکل در آن تعریف می‌شود و گزینه‌های ممکن برای آن عبارتند از `centimeters` و `normalized` و `points` و `pixels` و `characters` و `inches` . واحد پیش فرض `pixels` می‌باشد.

Visibe : مرئی یا نامرئی بودن شکل را مشخص می‌کند. مقادیر ممکن برای آن `on` و `off` می‌باشند.

Windowstyle : تعیین کنندهٔ normal یا modal بودن شکل است. مقادیر ممکن، `normal` و `modal` هستند.

مشخصات مهم اشیاء uicontrol

BackgroundColor : تعیین کنندهٔ رنگ پس زمینهٔ 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 کنندهٔ نسبت به سه رنگ اصلی قرمز، سبز و آبی در مقیاس بین ۰ تا ۱ باشد. مثلاً رنگ magenta با بردار [1 0 1] مشخص می‌شود.

Callback : تعیین کنندهٔ نام و پارامترهای تابع فراخوانی شده در هنگام فعال شدن شیء مربوط به آن (توسط صفحه کلید یا ورودی نوشتاری) می‌باشد.

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` می‌باشد.

FontAngle : رشته‌ای حاوی زاویهٔ فونت نمایش داده شده روی شیء است. مقدار آن می‌تواند `normal` و `italic` و `oblique` باشد.

FontName : رشته‌ای حاوی نام فونت برای متن نمایش داده شده روی شیء است.

FontSize : یک عدد مشخص کنندهٔ اندازهٔ فونت نمایش داده شده روی شیء است. اندازهٔ فونت به طور پیش فرض در واحد points می‌باشد.

FontWeight : رشته‌ای حاوی ضخامت فونت نمایش داده شده روی شیء است و مقدار آن می‌تواند `light` و `normal` و `demi` یا `bold` باشد.

ForegroundColor : تعیین کننده رنگ پیش زمینهٔ شیء می‌باشد.

HorizontalAligment : تعیین کنندهٔ جایگاه افقی متن درون شیء است. مقادیر ممکن عبارتند از: `left` و `center` و `right` .

Max : حداکثر مقدار value property برای شیء.

Min : حداقل مقدار value property برای شیء.

Parent : handle شکل دربرگیرندهٔ این شیء است.

Position : مکان شیء را روی صفحه، در مقیاس تعیین شده در مشخصهٔ `units` مشخص می‌کند و مقدار آن یک بردار با چهار عنصر است که دو عنصر اول آن مختصات x و y از گوشهٔ پایین سمت چپ شکل دربرگیرندهٔ آن هستند. دو عنصر بعدی طول و عرض شکل هستند.

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

Tooltipstring : مشخص کنندهٔ متن راهنمایی است که وقتی کاربر اشاره‌گر ماوس را روی یک شیء نگاه می‌دارد، نمایش داده می‌شود.

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 `chracters`. واحد پیش فرض `pixels` می‌باشد.

value : مقدار فعلی uicontrol می‌باشد. 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 می‌باشد. برای دیگر کنترل‌ها این property می‌تواند معانی متفاوتی داشته باشد.

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

۱-۴-۱ Text Field ها

یک text field شیای گرافیکی است که یک متن رشته‌ای را درون خود نمایش می‌دهد. می‌توان راستا و جایگاه متن را درون ناحیه نمایش، با تنظیم property ، Horizontal Alignment تعیین کرد. به طور پیش فرض، متن در مرکز text field قرار می‌گیرد. با ایجاد یک uicontrol که property ی style ش `edit` است، یک text field بوجود می‌آید. text field را همچنین می‌توان با استفاده از ابزار Layout Editor به GUI اضافه کرد.

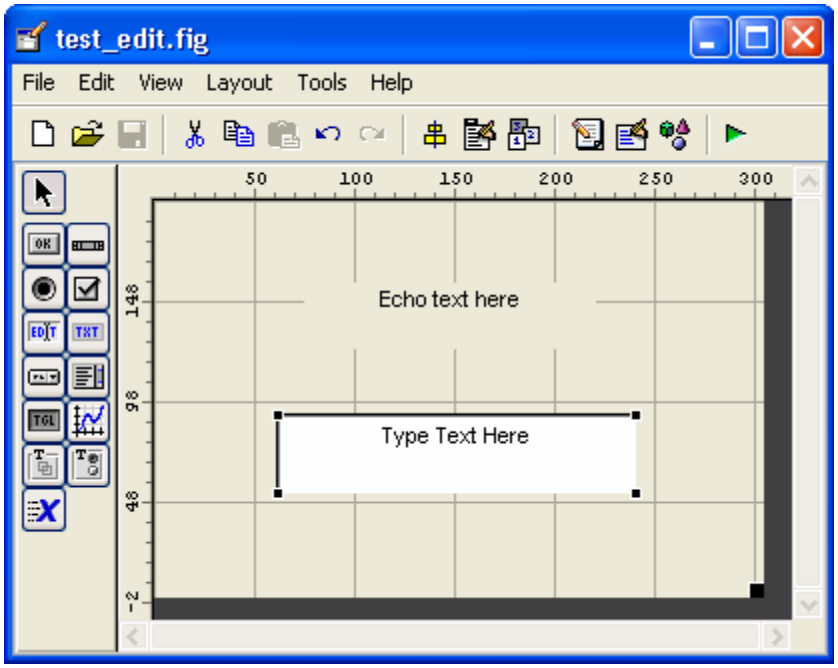
Text field ها callback ی را فعال نمی‌کنند، ولی می‌توان مقدار نمایش داده شدهٔ درون آنها را با تغییر String property آن از درون یک تابع callback ، همان طور که در بخش ۲-۱ دیدید، تغییر داد.

۱-۴-۲ Edit Box ها

یک edit box شیای گرافیکی است که به کاربر امکان وارد کردن یک متن رشته‌ای را می‌دهد. هنگامی که کاربر کلید Enter را پس از تایپ رشته درون جعبه، فشار می‌دهد، callback این عنصر فعال می‌شود. یک edit box را می‌توان با ایجاد یک uicontrol که property style آن `edit` می‌باشد، تولید کرد. edit box را همچنین می‌توان با استفاده از ابزار edit box در Layout Editor با GUI اضافه کرد.

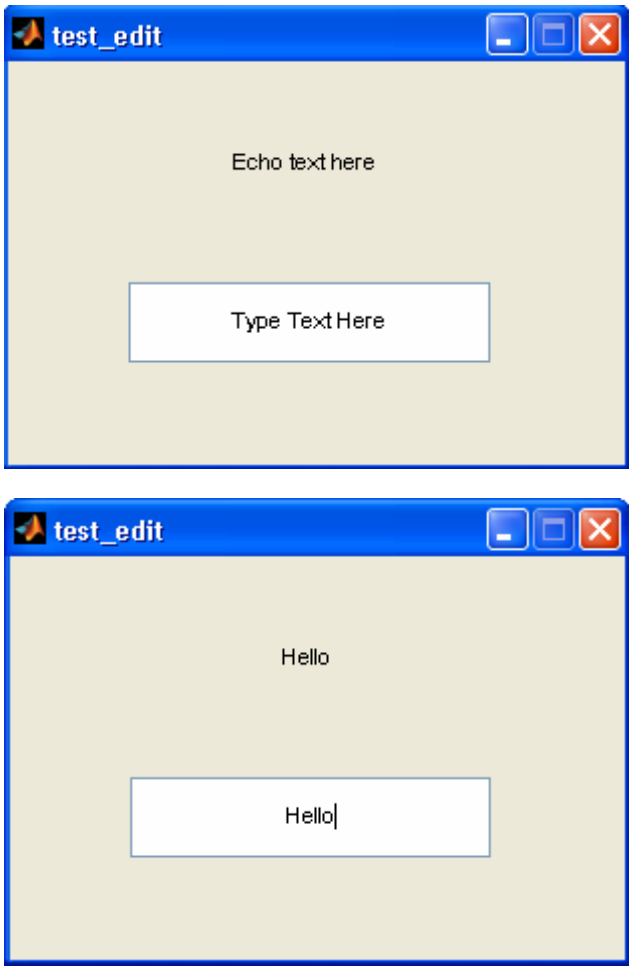
شکل ۱۱-۱ یک GUI ساده، حاوی یک edit box با نام `EditBox` و یک text field با نام `TextBox` را نشان می‌دهد. هنگامی که کاربر یک رشته را درون edit box تایپ می‌کند، این شیء بطور خودکار تابع EditBox_Callback را فراخوانی می‌کند. این تابع به کمک ساختار handles موقعیت و شناسایی edit box را مشخص می‌کند و رشتهٔ تایپ شده را از سوی کاربر دریافت می‌کند. 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);
```



شکل ۱۱-۱ یک GUI ساده با یک edit box و یک text field

شکل ۱۲-۱ این GUI را درست بعد از شروع به کارش، یعنی بعد از اینکه کاربر کلمهٔ 'Hello' را در edit box تایپ می‌کند نشان می‌دهد.



شکل ۱۲-۱ GUI تولید شده توسط برنامهٔ test_edit

۳-۴-۱ Frame ها

Frame (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 ۱۰-۱ دیده می‌شود، می‌توان از یک قاب برای قرار دادن یک گروه از radio button ها در کنار یکدیگر، استفاده کرد.

یک قاب را با ایجاد یک uicontrol که style property آن `frame` می‌باشد، می‌توان ایجاد کرد. همچنین فریم‌ها را می‌توان با استفاده از ابزار frame در Layout Editor به GUI افزود. قاب‌ها callback ی تولید نمی‌کنند. البته در MATLAB نسخهٔ ۷ اثری از Frame ها مشاهده نمی‌شود و باید از Panel که دارای عملکردی کاملاً مشابه است، به جای frame استفاده کرد.

۱-۴-۴ Pushbutton ها

یک pushbutton (دکمهٔ فشاری) عنصری است که کاربر می‌تواند با کلیک روی آن، عملیات خاصی را فعال کند. هنگامی که کاربر روی pushbutton کلیک می‌کند، callback آن فعال می‌شود. این عنصر را می‌توان 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 ۱۰-۱ تصویری از کاربرد pushbutton ارائه می‌دهد.

۱-۴-۵ Toggle Button ها

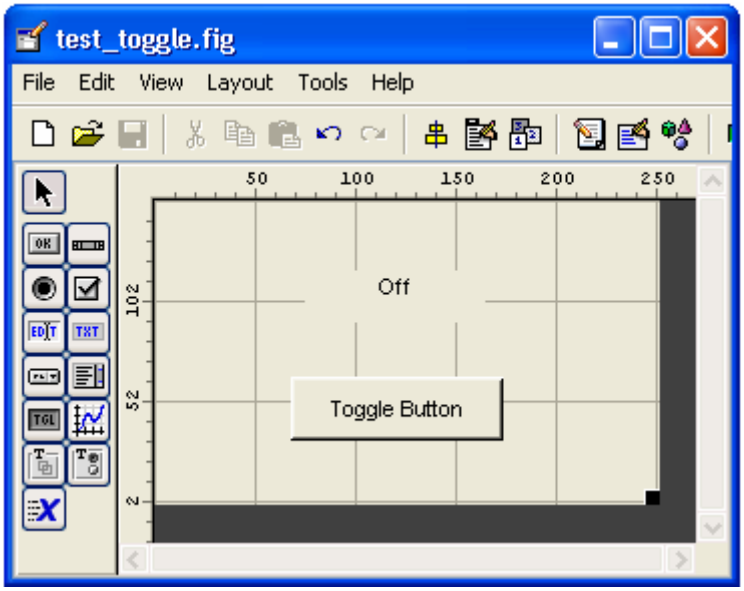
toggle button نوعی از دکمه است که دو حالت دارد: on (فشرده شده) وoff (رها). یک toggle button با کلیک ماوس روی آن، بین دو حالت تغییر وضعیت می‌دهد. مشخصهٔ `value` این عنصر وقتی کلید در حالت on قرار دارد max (که معمولاً 1 است) و هنگامی که off است min (که معمولاً 0 است) می‌شود.

toggle button را می‌توان با ایجاد یک uicontrol که style property آن، `togglebutton` می‌باشد، خلق کرد. همچنین آن را می‌توان با استفاده از ابزار toggle button در Layout Editor ایجاد کرد.

شکل ۱۳-۱ یک GUI ساده حاوی یک toggle button با نام `ToggleButton` و یک textfield با نام `TextBox` را نشان می‌دهد. هنگامی که کاربر روی

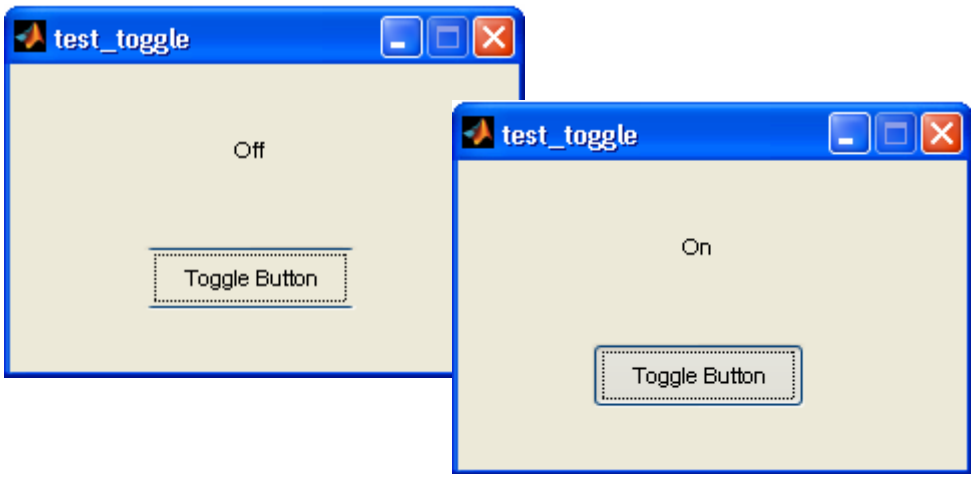
togglebutton کلیک می‌کند، این عنصر به طور خودکار تـابع ToggleButton_Callback را فراخوانی می‌کند. این تابع با بکار گیری ساختار handles ، موقعیت و مکـان toggle button را شناسایی کرده و حالت آن را از Value property اش دریافت می‌کند. سپس این تابع بـا تعیـین موقعیت text field حالت اخذ شده در مرحلهٔ قبل را درون text field نمایش می‌دهد.

```
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
```



شکل ۱۳-۱ یک GUI ساده حاوی یک toggle button و یک text field

شکل ۱-۱۴ تصویری از این GUI را درست بعد از شروع به کار و بعد از اینکه کاربر برای بـار اول روی toggle button کلیک می‌کند، نمایش می‌دهد.



شکل ۱-۱۴ تصویری از GUI تولیدی به وسیلهٔ برنامهٔ test_togglebutton هنگامی که togglebutton خاموش و روشن می شود .

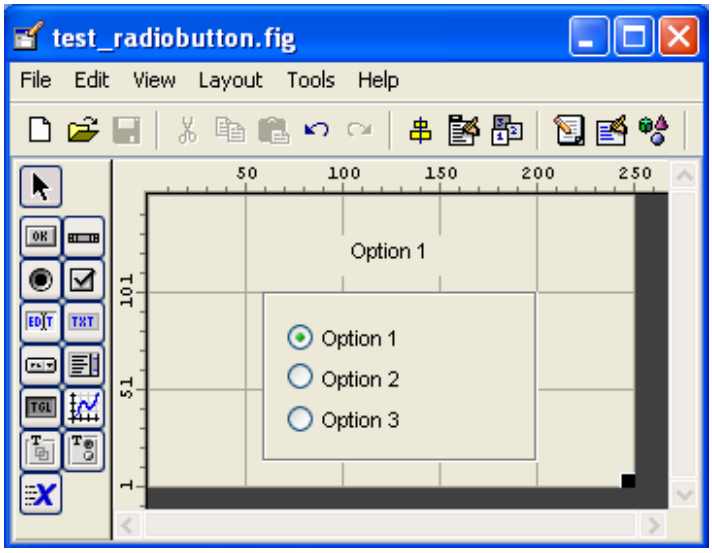
۱-۴-۶ Checkbox ها و Radio button ها

اساس کار Checkbox ها و Radio button ها مشابه toggle button هـا اسـت و تنها شکل ظاهری آنها فرق می‌کند. Checkbox ها و Radio button هـا نیـز مـشابه toggle button ها دارای دو وضعیت on و off هستند و با هر کلیک ماوس روی آنها بـین ایـن دو حالـت تغییر وضعیت داده و در هر مرتبه callback آنها فعـال مـی‌شـود. Value property ایـن اجـزا وقتی که on هستند max (که معمولاً 1 است) و وقتـی کـه off هـستند min (کـه معمـولاً 0 اسـت) می‌باشد. نمونه‌ای از یک checkbox و radio button در شکل ۱-۱۰ نشان داده شده است.

یـک checkbox را مـی‌تـوان بـا ایجـاد uicontrol ی کـه style property آن `checkbox` می‌باشد، ایجاد کرد. همچنین این عنصر را می‌توان با استفاده از ابـزار Layout Editor در checkbox ، ایجاد کرد. برای radio button نیـز وضـع بـه همـین منـوال اسـت. یـک radio button را مـی‌تـوان بـا ایجـاد uicontrol ی کـه style آن `radiobutton`

مـی‌باشـد، ایجـاد کـرد و همچنـین آن را مـی‌تـوان بـا اسـتفاده از ابـزار radio button در Layout Editor ایجاد کرد.

متداول است که از checkbox ها بـرای نمـایش گزینـه‌هـای on/off اسـتفاده مـی‌شـود و مجموعه‌ای از radio button برای انتخاب گزینه‌ای از میان گزینه‌های مستقل استفاده می‌شود.



شکل ۱۵-۱ یک GUI ساده حاوی سه radio button به همراه یک text field برای نمایش انتخاب کنونی.

شکل ۱۵-۱ مثالی از چگونگی ایجاد گروهی از گزینه‌های مستقل را بـا radio button هـا نـشان مـی‌دهـد. GUI نـشان داده شـده در ایـن سـه شـکل radio button بـا برچـسب‌هـای "Option1" و "Option2" و "Option3" در خـود دارد. هـر radio button از یـک callback مشابه ولی با پارامتر مستقل استفاده می‌کند.

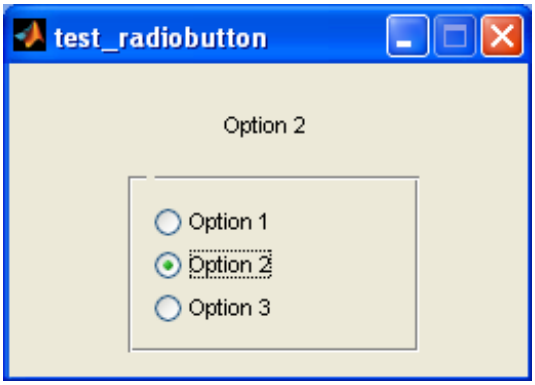
توابع callback مربوط به هر radio button :

```
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');
```

هنگامی که کاربر روی یکی از radio button ها کلیک می‌کند، تابع callback مربوط به آن اجزا می‌شود. این تابع متن نمایش داده شده در text box را به گزینه‌ای که هم اکنون انتخاب شده تغییر می‌دهد و radio button فعلی را روشن (on) و بقیه radio button ها را خاموش (off) می‌کند.

توجه کنید که این GUI از یک قاب برای قرار دادن radio button ها در کنار هم، برای تأکید بر اینکه اینها جزء یک مجموعه هستند، بهره گرفته است. شکل ۱-۱۶ تصویر این GUI را پس از انتخاب Option 2 نشان می‌دهد.

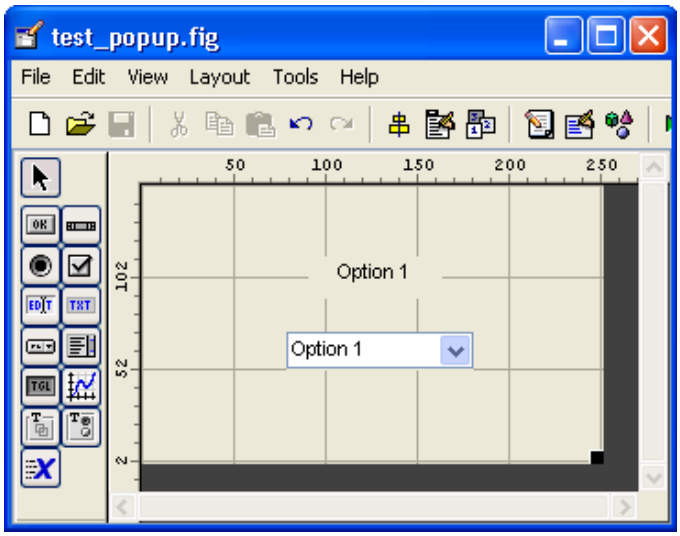


شکل ۱-۱۶ تصویر GUI ی تولید شده به وسیلهٔ برنامهٔ test_radiobutton

۱-۴-۷ منوهای Popup

منوهای popup اجزای گرافیکی هستند که به کاربر اجازهٔ انتخاب یک گزینه از میان لیستی از گزینه‌های مستقل را می‌دهند. این لیست که کاربر از میان آن گزینهٔ مورد نظر را انتخاب می‌کند، به وسیله آرایه‌ای از نوع cell که حاوی رشته‌های گزینه‌هاست، مشخص می‌شود. مشخصهٔ `value` برای این منو تعیین می‌کند که کدام گزینه هم اکنون انتخاب شده است. یک منوی popup را می‌توان به وسیلهٔ ابزار popup menu در Layout Editor به GUI اضافه نمود.

شکل ۱۷-۱ نمونه‌ای از یک منوی popup را نشان می‌دهد. این GUI حاوی یک منوی popup با پنج گزینه با برچسب‌های "Option 1" و "Option 2" و . . . است.

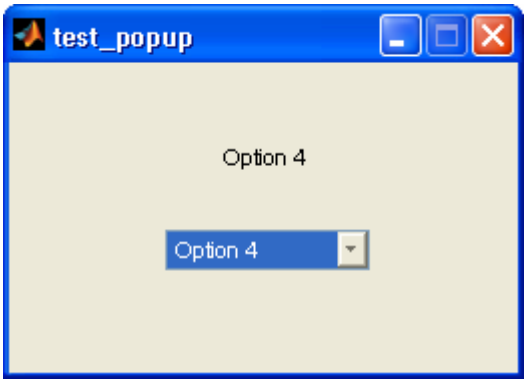


شکل ۱۷-۱ یک GUI ساده حاوی یک منوی popup و یک text field برای نمایش گزینهٔ انتخاب شده .

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);
```

این تابع گزینهٔ 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 ۱۸-۱ تصویری را از این GUI پس از انتخاب Option 4 نشان می‌دهد.



شکل ۱۸-۱ تصویری از GUI ایجاد شده به وسیلهٔ برنامهٔ test_popup

۱-۴-۸ List Box ها

list box ها اشیایی گرافیکی هستند که چند خط نوشته را در خود نمایش می‌دهند و به کاربر اجازهٔ انتخاب یک یا چند خط از این خطوط را می‌دهند. 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 می‌دهد. خطوطی که کاربر می‌تواند انتخاب کند، به وسیلهٔ یک آرایهٔ سلولی مشخص می‌شود و مقدار Value property مشخص می‌کند که کدام رشته انتخاب شده است.

یک list box را می‌توان با ایجاد uicontrol ی که style property آن `listbox` است، بوجود آورد. list box را همچنین می‌توان به کمک ابزار listbox در Layout Editor ایجاد نمود.

از list box ها می‌توان برای انتخاب یک گزینه از میان مجموعه‌ای از گزینه‌های ممکن استفاده نمود. در کاربردهای متداول GUI ، تک کلیک ماوس روی یکی از موارد لیست تنها باعث انتخاب

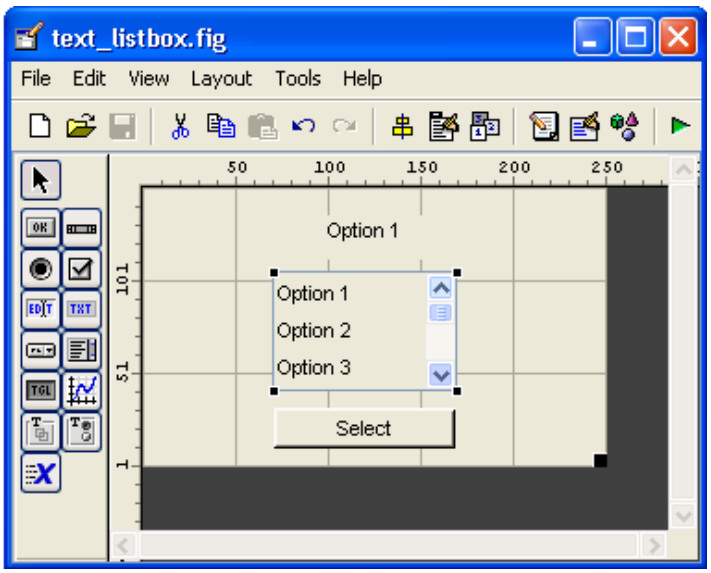
آن می‌شود و منجر به اتفاق خاص دیگری نمی‌شود. با این وجود، عملیات منتظر و آمادهٔ تحریک‌های دیگر از طرف سایر عناصر، مثل یک pushbutton می‌شود. نوع پیشامدهای تک-کلیک و دوبار-کلیک را می‌توان با استفاده از SelectionType property شکلی که عمل کلیک کردن روی آن اتفاق می‌افتد، از هم تشخیص داد. یک کلیک ماوس، رشتهٔ `normal` را در SelectionType property قرار می‌دهد و دوبار کلیک رشتهٔ `open` را در SelectionType property جای می‌دهد.

البته انتخاب چندین گزینه از درون لیست نیز میسر است. اگر اختلاف میان property های max و min از یک بیشتر باشد، آنگاه انتخاب چند گزینه امکان پذیر است. در غیر اینصورت تنها یک مورد را می‌توان از لیست انتخاب کرد. شکل ۱۹-۱ نمونه‌ای از یک list box را که تنها قابلیت انتخاب یک مورد را داراست، نشان می‌دهد. GUI نشان داده شده در این شکل حاوی یک list box به هشت گزینه، با برچسب‌های "option 1" و "option 2" و ... است. به علاوه، این GUI دارای یک pushbutton برای انجام عمل انتخاب و یک text field برای نمایش گزینهٔ انتخاب شده، می‌باشد. list box و pushbutton هر دو تولید callback می‌کنند.

توابع callback مربوطه در زیر آورده شده‌است. اگر انتخابی در list box صورت گیرد، آنگاه تابع listbox1_callback اجرا خواهد شد. این تابع شکل تولید کنندهٔ 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 نمی‌دهد، ولی اگر دو کلیک بود، این تابع مقدار انتخاب شده در list box را دریافت می‌کند و رشتهٔ متناسب با آن درtext field قرار می‌دهد.

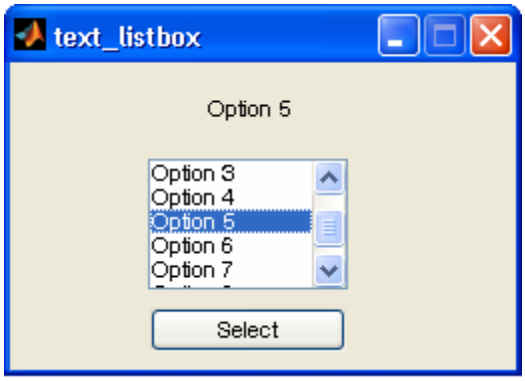
```
function button1_Callback(hObject, eventdata, handles)

%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');


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



شکل ۱۹-۱ 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 ، رشتهٔ مربوط به آن را درون text field می‌نویسد. GUI تولید شده بوسیلهٔ برنامهٔ test_listbox در شکل ۲۰-۱ نشان داده شده است.



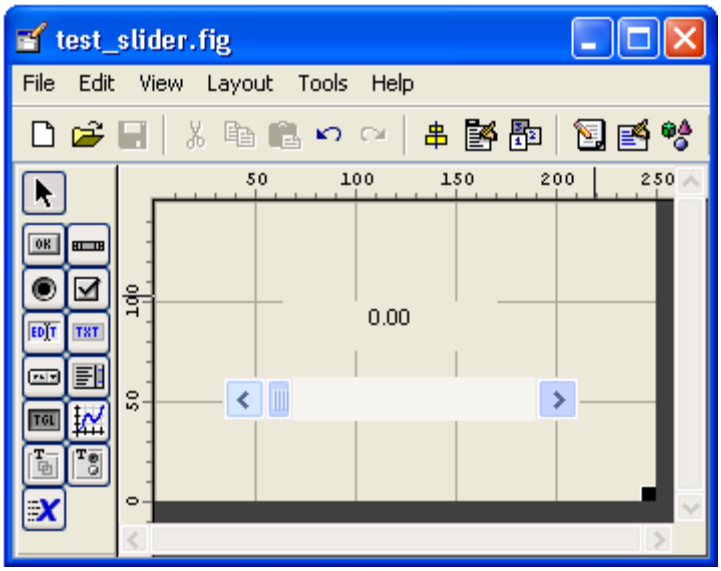
شکل ۲۰-۱ GUI تولید شده به وسیلهٔ برنامهٔ test_listbox

۹-۴-۱ Slider ها

slider اشیاء گرافیکی‌ای هستند که به کاربر این امکان را می‌دهند تا مقداری را از میان دامنهٔ پیوسته‌ای از مقادیر، با حرکت یک bar به وسیلهٔ ماوس، انتخاب کند. این مقدار بین مینیمم و ماکسیمم مقادیر پیش فرض تغییر می‌کند. Value property برای slider مقداری بین max و min ، بسته به موقعیت آن، به خود می‌گیرد.

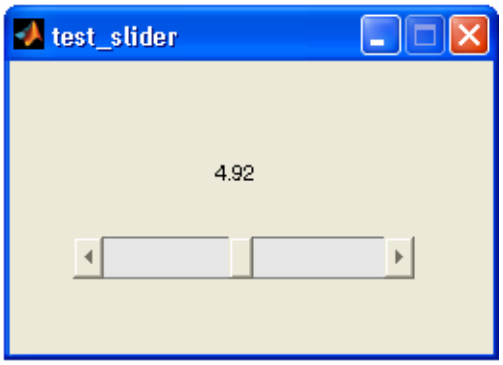
یک slider را می‌توان با ایجاد یک uicontrol که style property اش `slider` می‌باشد، ایجاد نمود. البته آنرا می‌توان به وسیلهٔ ابزار slider در Layout Editor نیز تولید کرد.

شکل ۲۱-۱ یک GUI ساده حاوی یک slider و یک text field را نشان می‌دهد. min property برای این slider ، صفر و Max property آن، ۱۰ انتخاب شده است. وقتی کاربر Slider را حرکت می‌دهد، این عنصر بطور خودکار تابع Slider_Callback را فراخوانی می‌کند. این تابع با دریافت مقدار slider از مشخصهٔ `Value` آن، آنرا در text field نمایش می‌دهد.



شکل ۱-۲۱ شمایی از یک GUI حاوی یک slider و یک text field

شکل ۱-۲۲ این GUI را به همراه slider آن که در موقعیت وسط خود قرار دارد، نشان می‌دهد.



شکل ۱-۲۲ GUI تولید شده به وسیلهٔ برنامهٔ test_slider

۱-۵ Dialog Box ها (جعبه‌های محاوره‌ای)

یک dialog box نوع خاصی از اشیاء figure است که از آن برای نمایش اطلاعات یا دریافت ورودی از کاربر، استفاده می‌شود. dialog box ها معمولاً برای نمایش پیغام‌های خطا، هشدار، پرسیدن سؤالات و دریافت ورودی از کاربر، مورد استفاده قرار می‌گیرند. از آنها همچنین برای انتخاب فایل و تنظیم propertyهای چاپگر استفاده می‌شود.

dialog box ها می‌توانند modal یا non-modal باشند. نوع modal آن تا زمانی که باز است و بسته نشده است، اجازهٔ دسترسی به دیگر پنجره‌های درون برنامه را به کاربر نمی‌دهد. از این نوع dialog box ها معمولاً برای نمایش پیام‌های خطا و هشدار که به توجه و پاسخ فوری نیاز دارند و نمی‌توان از آنها بی‌تفاوت گذشت، استفاده می‌شود. تمام dialog box ها از پیش non-modal فرض می‌شوند.

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 می‌کند.

errordlg : یک پیغام خطا در dialog box نشان می‌دهد. کاربر برای ادامهٔ کار، باید روی دکمهٔ OK کلیک کند.

helpdlg : یک پیغام help در dialog box نمایش می‌دهد. کاربر برای ادامهٔ کار، باید روی دکمهٔ OK کلیک کند.

inputdlg : یک پیغام که درخواست وارد نمودن داده را می‌نماید، نمایش می‌دهد و مقدار ورودی را از کاربر دریافت می‌کند.

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

printdlg : یک dialog box ، برای انتخاب چاپگر نمایش می‌دهد.

questdlg : یک سؤال می‌پرسد! این dialog box می‌تواند دارای دو یا سه دکمه باشد، که بطور پیش فرض Yes و No و Cancel نام‌گذاری شده‌اند.

uigetfile : یک dialog box برای باز کردن فایل نمایش می‌دهد. این پنجره در حقیقت به کاربر اجازهٔ انتخاب یک فایل را می‌دهد ولی این فایل را باز نمی‌کند.

uiputfile : یک dialog box برای ذخیرهٔ فایل نمایش می‌دهد. این پنجره نیز در حقیقت به کاربر اجازهٔ انتخاب یک فایل را برای ذخیره کردن می‌دهد ولی آنرا ذخیره نمی‌کند.

uisetcolor : یک dialog box برای انتخاب رنگ نمایش می‌دهد.

uisetfont : یک dialog box برای انتخاب رنگ نمایش می‌دهد.

warndlg : یک پیغام هشدار در یک dialog box نمایش می‌دهد. کاربر باید برای ادامه کار، روی دکمه OK کلیک کند.

۱-۵-۱ Dialog Box های Error و Warning

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. متداول‌ترین طریقهٔ فراخوانی این dialog box ها به صورت زیر است:

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

warning_string یا error_string پیغامی است که قرار است به کاربر نشان داده شود، box_title عنوان dialog box می‌باشد و create_mode هم رشته‌ای است که بسته به نوع dialog box ی که شما می‌خواهید ایجاد کنید، `modal` یا `non-modal` می‌باشد.

به عنوان مثال عبارت زیر یک پیغام خطا از نوع modal ایجاد می‌کند به طوری که کاربر نمی‌تواند آنرا نادیده بگیرد و از آن بگذرد. dialog box تولید شده به وسیلهٔ عبارت زیر، در شکل ۱-۲۵ نشان داده شده است.

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

شکل ۲۵-۱ یک error dialog box

۲-۵-۱ Input Dialog Box ها

یک input dialog box از کاربر می‌خواهد که یک یا چند مقدار مورد نیاز برنامه را وارد کند. input dialog box را می‌توان 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,defalt_answer)
```

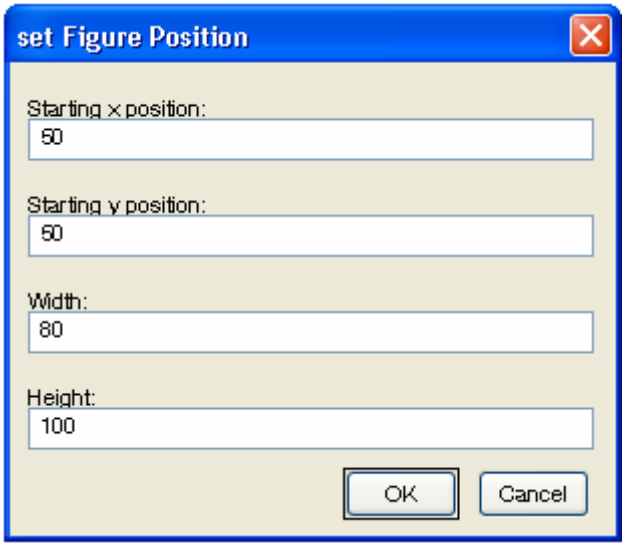
در اینجا prompt یک آرایهٔ سلولی می‌باشد. عناصر این آرایه رشته‌هایی هستند که هر یک از آنها متناظر با مقداری است که از کاربر خواسته می‌شود که آنها را وارد کند. پارامتر title عنوان dialog box را تعیین می‌کند و line_no تعداد خطوط مجاز برای جواب را مشخص می‌کند و آخر از همه، default_answer یک آرایهٔ سلولی، حاوی جواب‌های از پیش مشخص شده است و هنگامی مورد استفاده قرار می‌گیرد که کاربر دادهٔ مربوط به گزینه‌ای را وارد نکند. توجه کنید که جواب‌های از پیش تعیین شده باید به تعداد prompt ها باشد.

وقتی کاربر روی دکمه OK کلیک می‌کند، جواب‌هایی که او وارد کرده است به صورت یک آرایه سلولی حاوی رشته‌های 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);
```

dialog box حاصل در شکل ۲۶-۱ نشان داده شده است.



شکل ۲۶-۱ یک input dialog box

۳-۵-۱ Dialog Box های uigetfile و uisetfile

جعبه‌های محاوره‌ای uigetfile و uisetfile به منظور فراهم کردن امکان انتخاب فایل به طور بصری طراحی شده‌اند. این dialog box ها تنها نام و محل فایل را باز می‌گردانند و در واقع فایل را باز و ذخیره نمی‌کند. این برنامه نویس است که مسؤل نوشتن کد برای ذخیره کردن فایل است.

عبارات ایجاد کنندهٔ این dialog box به شکل زیر هستند:

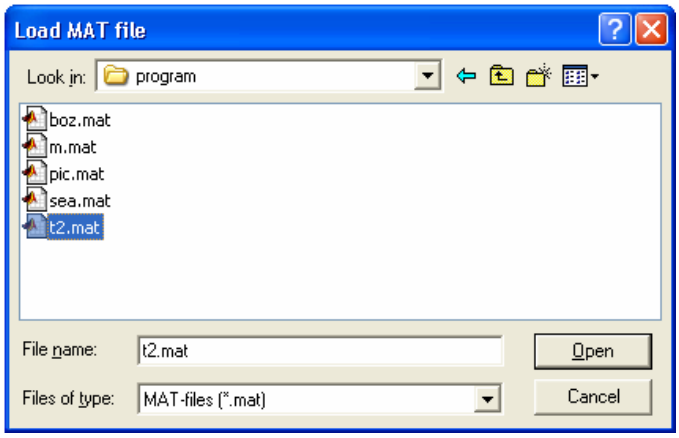
```
[filename , pathname]=uigetfile(filter_spec,title);
[filename , pathname]=uisetfile(filter_spec,title);
```

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

script file زیر چگونگی استفاده از این dialog box را نشان می‌دهد.

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

این عبارت از کاربر درخواست می‌کند که نام یک mat-file را وارد کند و سپس محتوای آن فایل را می‌خواند. شکل ۲۷-۱ dialog box ایجاد شده به وسیلهٔ این کد را در سیستم عامل Windows XP نشان می‌دهد.



شکل ۲۷-۱ یک dialog box برای باز کردن فایل ، که به وسیلهٔ دستور uigetfile ایجاد شده است.

۶-۱ Menu ها

Menu ها را نیز می‌توان به GUI در MATLAB اضافه کرد. یک منو به کاربر اجازهٔ انتخاب گزینه‌ای را بدون ظهور عنصر دیگری در GUI ، می‌دهد. برای جلوگیری از پر شدن GUI از دکمه‌های اضافی و برای انتخاب گزینه‌هایی 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 با دکمهٔ سمت راست ماوس کلیک می‌کند ظاهر می‌شوند.

منوهای استاندارد بوسیلهٔ 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 کنندهٔ کلید معادل در صفحه کلید برای یک گزینه در منو است. کاربر با فشردن کلیدهای CTRl + key به طور همزمان، می‌تواند گزینهٔ مورد نظر را از طریق صفحه کلید، فعال کند.

Callback : تعیین کننده نام و پارامترهای تابعی است که با فعال شدن گزینهٔ مربوط به آن در 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 می‌شود.

Checked : وقتی این property روشن (`on`) باشد، یک علامت تیک (✓) در سمت چپ 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 گزینهٔ منو قابل انتخاب است یا خیر. اگر یک گزینه منو بوسیلهٔ این property ، از کار افتاده باشد، دیگر به کلیک‌های ماوس و کلیدهای میان بر پاسخ نمی‌دهد. مقادیر ممکن برای این property ، `on` و `off` هستند.

Lable : متن نمایش داده شده روی منو را مشخص می‌کند. برای اختصاص یک کلید مخفف به گزینه‌ای از منو می‌توان از کاراکتر آمپرسند (&) در ابتدای نام منو، استفاده کرد. این علامت در نام منو ظاهر نمی‌شود. به عنوان مثال، رشتهٔ `&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 یک منوی دیگر باشد .

Position : موقعیت و مکان گزینه منو را روی menu bar یا درون منو ، مشخص می کند . موقعیت ۱ ، برای یک منوی سطح بالا ، منتهاالیه سمت چپ در menu bar و برای زیر منوها بالاترین موقعیت در منوی دربرگیرنده آنها می باشد .

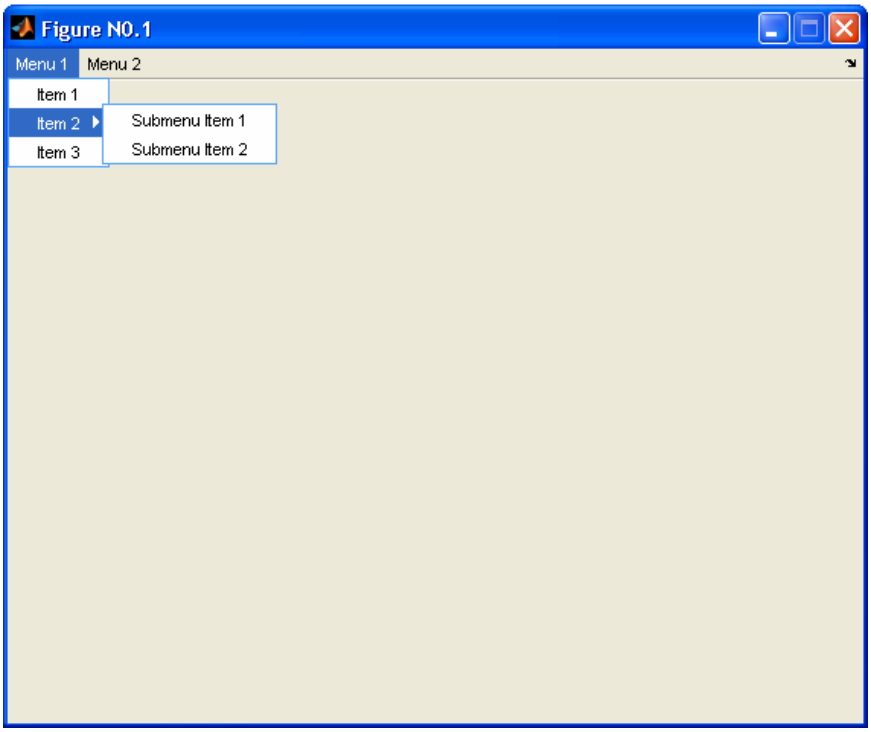
Seperator : وقتی این Property ، `on` است ، یک خط بالای این گزینه در منو ظاهر می شود که آنرا از بقیه جدا می کند . مقادیر ممکن برای آن ، `on`و `off` هستند .

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

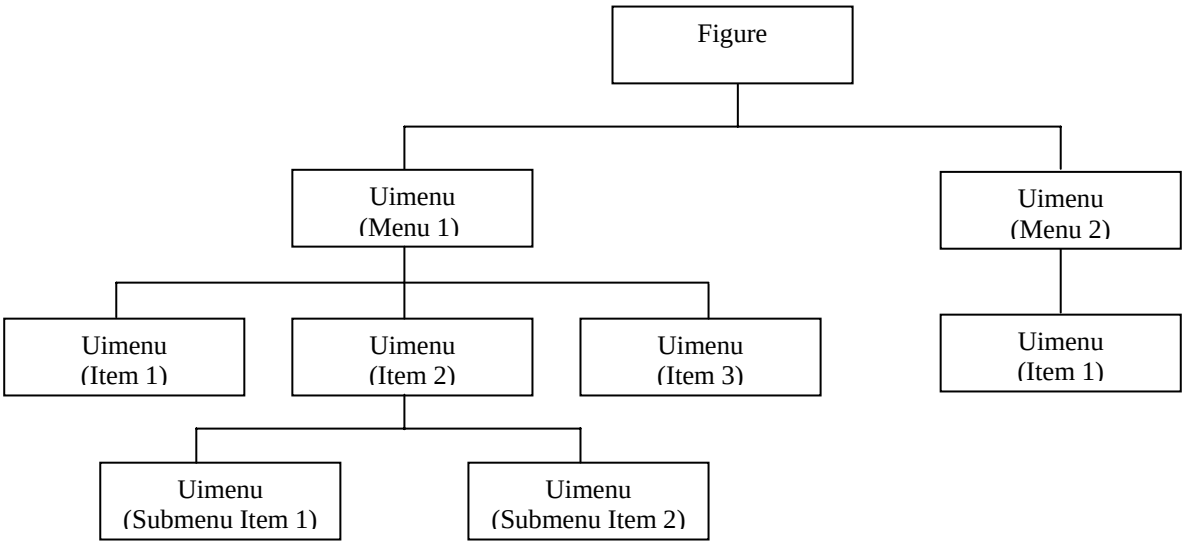
Visible : مرئی یا نامرئی بودن یک گزینهٔ منو را تعیین می‌کند مقدار آن می‌تواند `on` یا `off` باشد.

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 ۱-۲۸(a) یک نمونه منو را در حال کار نشان می‌دهد و شکل ۱-۲۸(b) رابطهٔ بین اشیاء سازنده این منو را نشان می‌دهد.



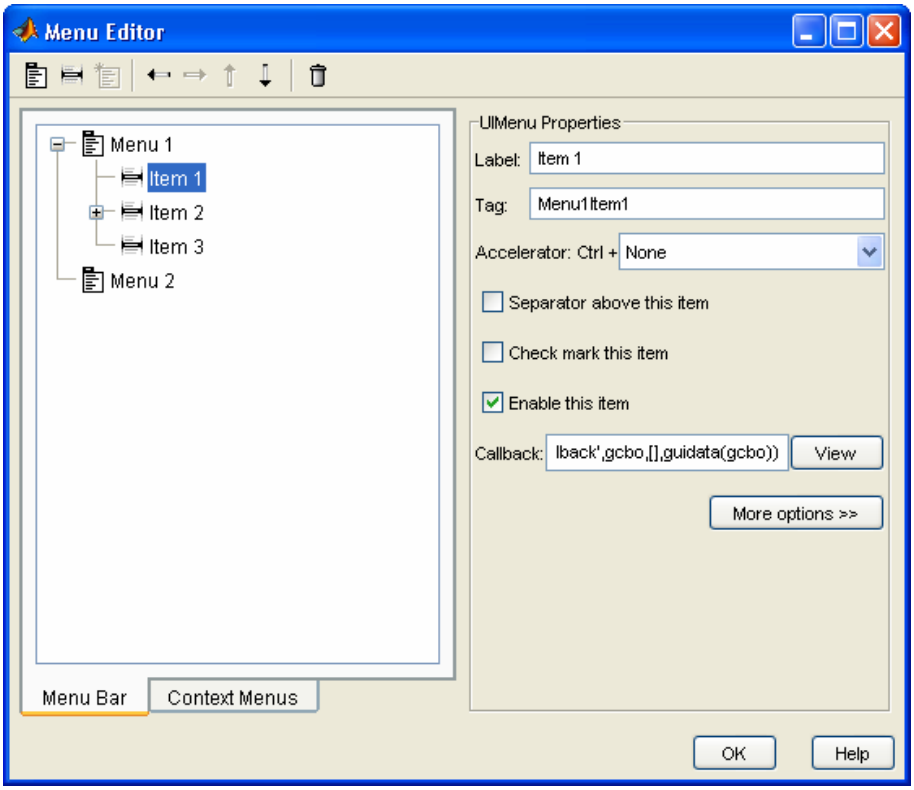
(a)



(b)

شکل ۱-۲۸ (a) ساختاری از منوها و گزینه های آن (b) رابطهٔ بین اجزای تشکیل دهندهٔ منو

منوهای MATLAB را می‌توان بـه کمـک Menu Editor ایجـاد نمـود. شـکل ۲۹-۱ پنجـرۀ Menu Editor را با گزینه‌های تولید کنندۀ ساختار این منو نشان می‌دهد. تمام مشخصه‌های موجـود در Menu Editor نـشان داده نمـی‌شـوند و بـرای تغییـر آنهـا بایـد از Property Editor (propedit) استفاده نمود.



شکل ۲۹-۱ نمایی از Menu Editor ایجاد کنندۀ این منوها

منوهای context سطح بالا با اشیاء uicontextmenu سـاخته مـی‌شـوند و گزینـه‌هـای سطح پائین در آنها با اشیاء uimenu ایجاد می‌شـوند.اصـول و عملکـرد منوهـای context مـشابه بـا منوهای استاندارد است، جز اینکه می‌توان آنها را با هر شیء GUI (مثل متن، خط، محورهـای مختـصات، اشکال) مرتبط کرد. ایستی از مشخصه‌های مهم اشیاء uicontextmenu در زیرداده شده است.

مشخصه‌های مهم اشیاء uicontextmenu

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

parent : handle شیء مادر برای منوی context

Tag : نام منوی context است که از آن برای تعیین موقعیت منو استفاده می‌شود.

Visible : مرئی یا نامرئی بودن منوی context را تعیین می‌کند. این مشخصه بطور خودکار مقدار دهی می‌شود و معمولاً نباید مقدار آنرا تغییر داد.

۱-۶-۱ از بین بردن اثر منوهای پیش فرض

هر شکل MATLAB مجموعه‌ای 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 ممکن برای 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 پیش فرض از بین می‌روند. شما می‌توانید این کار را با کمک Property Inspector در هنگام خلق GUI ، انجام دهید.

۱-۶-۲ چگونه منوهای مورد نظرمان را بسازیم؟

برای ساختن منوهای استاندارد مورد نظرتان 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 گزینه‌های منو یک Label برای نمایش روی آن و یک Tag یکتا نسبت دهید.

بهترین راه برای نوشتن callback برای یک منو، بررسی و مدل کردن callback ی است که بوسیلهٔ یک uicontrol بطور خودکار ایجاد می‌شود. 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. مهم‌ترین مشخصه‌هایی که باید برای یک گزینهٔ منو تنظیم شوند، Tag ، Callback ، Label آن هستند. که می‌توان آنها را بدون نیاز به Property Editor از درون Menu Editor تنظیم کرد. با این وجود اگر قصد تغییر مشخصه‌ها را دارید باید از Property Editor استفاده کنید، تعداد مشخصه‌هایی را که می‌توان از داخل Menu Editor تغییر داد در MATLAB نسخهٔ ۷ بیشتر شده است و تقریباً دیگر نیازی به Property Editor احساس نمی‌شود.

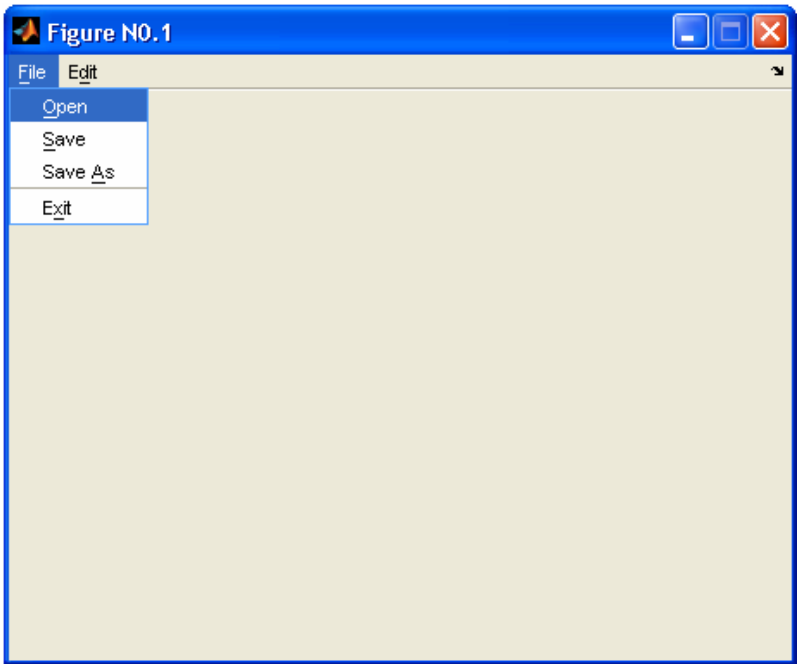
قدم سوم، پیاده‌سازی تابع callback برای انجام عملیات مورد نظر برای هر گزینهٔ منو است. توجه داشته باشید در این مرحله باید توابع callback را خودتان بطور دستی ایجاد کنید.

۳-۶-۱ کلیدهای میان‌بر و کلیدهای مخفف

MATLAB قابلیت کار با کلیدهای میان‌بر و کلیدهای مخفف را دارد. کلیدهای میان‌بر در واقع 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 اختصاص داد. در این صورت با فشردن همزمان دو کلید CTRL و O تابع callback گزینهٔ File/Open اجرا می‌شود.

کلیدهای میان‌بر را می‌توان با تنظیم Accelerator property در یک شیء uimenu تعریف کرد.

کلیدهای مخفف حروف تکی هستند که با فشار آنها در صفحه‌کلید هنگامی که منو باز است ، می‌توان گزینهٔ مربوطه را در منو اجرا کرد. زیر این حروف در گزینهٔ مربوطه ، یک خط تیره کشیده می‌شود. (شکل ۳۰-۱ را ببینید : Open). منوهای سطح بالا (مثلاً در شکل ۳۰-۱ ، 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 ، گزینه‌های درون آنها را می‌توان تنها با فشار کلید مخفف مربوطه اجرا کرد.



شکل ۳۰-۱ چگونگی استفاده از کلیدهای مخفف را نشان می‌دهد. منوی File با فشار کلیدهای ALT+f باز می‌شود و وقتی باز شد، با فشار کلید "x" می‌توان گزینهٔ Exit را اجرا نمود.

کلیدهای مخفف را می‌توان با قرار دادن کاراکتر (&) قبل از حروف مخفف مورد نظر در Label Property تولید نمود. علامت (&) در نام منو ظاهر نمی‌شود ، ولی زیر حرف بعد از آن ، در نام منو یک خط تیره ظاهر می‌شود که در واقع به کاربر می‌گوید این حرف یک کلید مخفف است . برای مثال Label property منوی Exit در شکل ۳۰-۱ به صورت 'E&xit' است .

۴-۶-۱ ساخت منوهای Context

منوهای Context به طریقی مشابه با منوهای معمولی ایجاد می‌شوند ، جز اینکه گزینهٔ 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 گزینهٔ "context Menu" در Menu Editor ایجاد نمود . پس از ایجاد یک منوی context می‌توان 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 می‌دهند ولی انجام آن با فرمان set نیز امکان پذیر است . (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 وسیلهٔ plot به وجود آمده ، مرتبط می‌سازد .

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

۱-۷ نکاتی برای خلق GUI های کارآمدتر

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

۱-۷-۱ tool tips

Tool tipها پنجره‌های کمکی کوچکی هستند که هنگام نگاه داشتن اشاره‌گر ماوس روی یک شیء uicontrol خودبه‌خود ظاهر می‌شوند و از آنها برای راهنمایی سریع کاربر دربارهٔ عملکرد آن شیء استفاده می‌شود.

یک tool tip را می‌توان با قراردادن متنی که قرار است نمایش داده شود در tooltipstring property برای یک شیء ایجاد نمود.

۱-۷-۲ 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 می‌کند 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 با افزایش اندازه و حجم توابع، زمان کامپایل اولیه به مراتب افزایش می‌یابد. از آنجا که توابع تعریف کنندهٔ یک GUI معمولاً بزرگ هستند، زمان کامپایل کردن برنامه‌هایی که بر اساس GUI طراحی شده‌اند، به مراتب 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 برنامه در آینده 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 زیر را 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 نسخهٔ 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 لـو رفـتن ایـده‌هایتـان محافظـت کنید. pcode ها می‌تواند به راحتی اجرا شود ولی دیدن کد درون آنها و ایجاد تغییرات در آنهـا از عهـده هر کسی بر نمی‌آید.