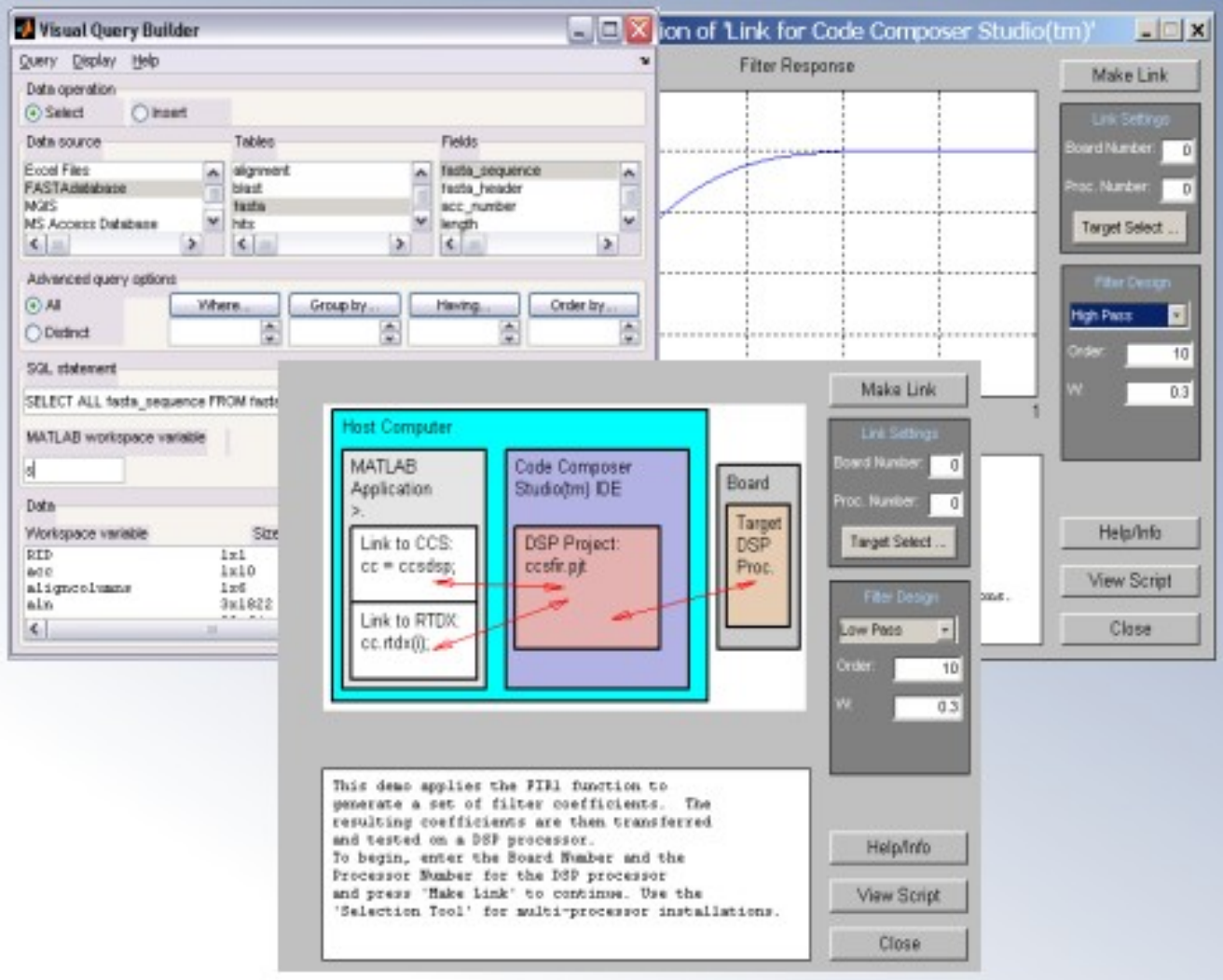
تصميم الواجهات الرسومية بالـ Matlab



بن العبد

بســـم الله الـرحمـــن الرحيـــم

"و لا تحسبن الله غافلا عما يعمل الظالمون إنما يؤخرهم ليوم تشخص فيه الأبصـــار."

صدق الله العظيم

إبراهيم 42

في "أمتنا" شعب في التراب و شعوب في السحاب, شعب فلسطيني يستحلب الصخر و شعوب خليجية تتمرغ و تتمرغ النعمة في أعتابهم و يأبون إلا أن يدوسوها بأقدامهم.

الفهرس

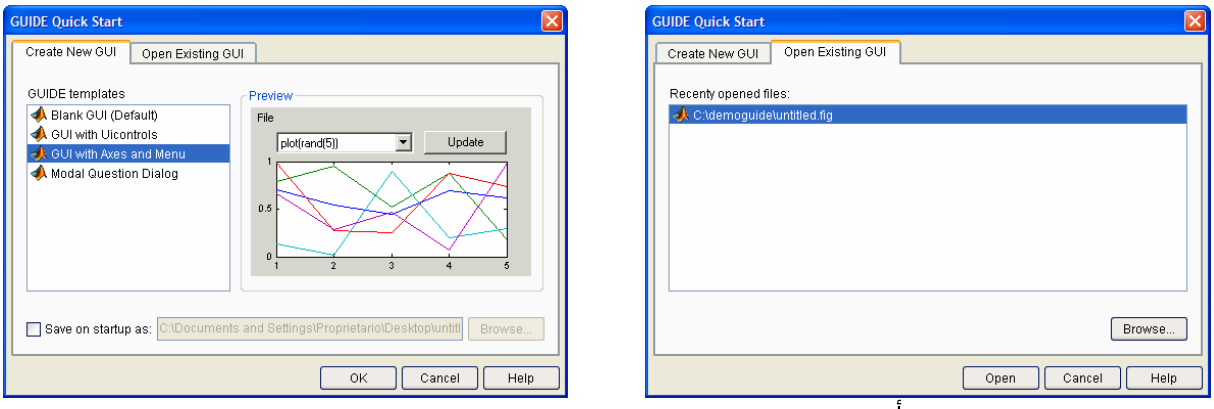
1. مقدمة

هذا الدرس موجه لمن لهم خلفية في البرمجة بالـ Matlab و يهدف للمساعدة في تطوير برامج متكاملة ذات واجهات رسومية تفاعلية. و ينقسم إلى جزئين منفصلين. يخص الجزء الأول التصميم بإستخدام الأداة guide في حين أن الجزء الثاني يهتم بالتصميم البرمجي للواجهات الرسومية.

2. الجزء الأول

في هذا الجزء سأحاول تقديم شرح مختصر لتصميم الواجهات بإستعمال guide الذي يوفره الـ Matlab. الـ guide عبارة عن أداة تحتوي على عناصر جاهزة توفر مجموعة من المكونات واوجهة رسومية جاهزة تضاف لها هذه العناصر عبر السحب و الإفلات[1].

عند تنفيد التعليمة guide نتحصل على الواجهة المجسدة في الصورة رقم 1. تحتوي هذه الواجهة على قائمة من أنواع الواجهات الرسومية التي يمكن تصميمها. كما توفر إمكانية فتح تصميم سابق.

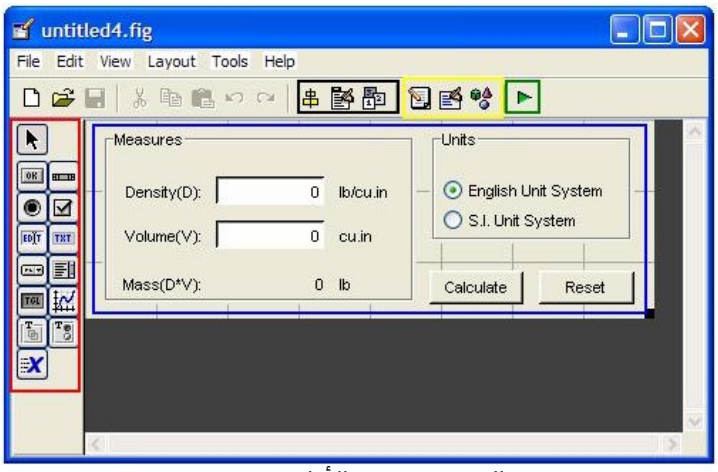


أ ب

الصورة رقم 1: إنشاء و فتح مشروع

عند إختيار تصميم واجهة رسومية جديدة من نوع Blank GUI نتحصل على ما هو مجسد في الصورة رقم 2. الواجهة تضم المكونات الرئيسية و الأكثر إستخداما و هو ما ييسر للمبرمج تعديلها والإستفادة من الشفرة الأولية المولدة تلقائيا. و بالإضافة للمكونات المتواجدة في الواجهة الرسومية، توجد عدة مكونات أخرى في علبة الأدوات[2] مما يتيح للمبرمج تصميم واجهة أكثر تناسبا مع احتياجاته.

[1] السحب و الإفلات: Drag and drop
[2] علبة الأدوات: ToolBox



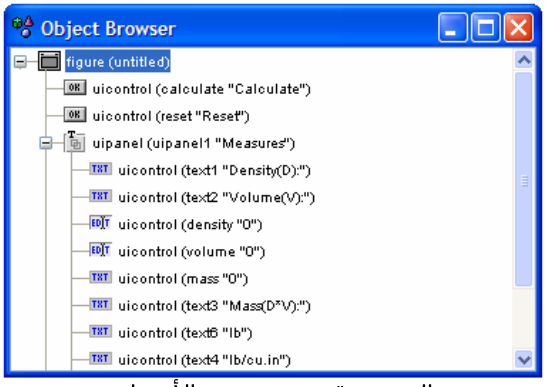
الصورة رقم 2: الأداة guide.

قبل الشروع في التصميم و تفاصيله نتوقف لفهم محتويات الواجهة الأولية لتصميم الواجهات الرسومية و عناصرها.

- ✓ يجسد المستطيل الأزرق و الأكثر أهمية بالنسبة لنا مجال التصميم و ما سيظهر لاحقا في الواجهة النهائية, و تهدف كافة العناصر المحيطة يه للتحكم في كل ما نضيفه للواجهة.
- ✓ يجمع المستطيل الأحمر المكونات التي يمكن إضافتها للواجهة الرسومية.
- ✓ المستطيل الأخضر و بكل بساطة زر الترجمة.
- ✓ يجمع المستطيل الأصفر مجموعة من العناصر الهامة في تصميم المكونات و تعديل خصائصها و وضائفها و سنفهمها سويا إن شاء الله إبتداء من اليمين إلى اليسار:

- و ضيفة العنصر الأول و هو متصفح الكائنات[3], الإطلاع على قائمة المكونات المدرجة في الواجهة بترتيب. و أعني هنا بعبارة "ترتيب" هو إبراز المكونات بكيفية مستقلة إن كانت كذلك في الواجهة. أما إن كانت متواجدة داخل حاوية[4] فإنها لن تظهر إلا إذا تصفحنا محتويات هذه الأخيرة.
- تجسد الصورة رقم 3 متصفح الكائنات حيث تحتوي أربعة عناصر رئيسية. الزران والحاويتان, و في كل حاوية مجموعة العناصر التابعة لها. يمكن التعديل على هذه العناصر عبر النقر المزدوج فيتم فتح واجهة جديدة ألا و هي مراقب الخصائص[5] (الصورة رقم 4).

[3] متصفح الكائنات: Object Browser.
[4] حاوية: Uipanel.
[5] مراقب الخصائص: Property Inspector.



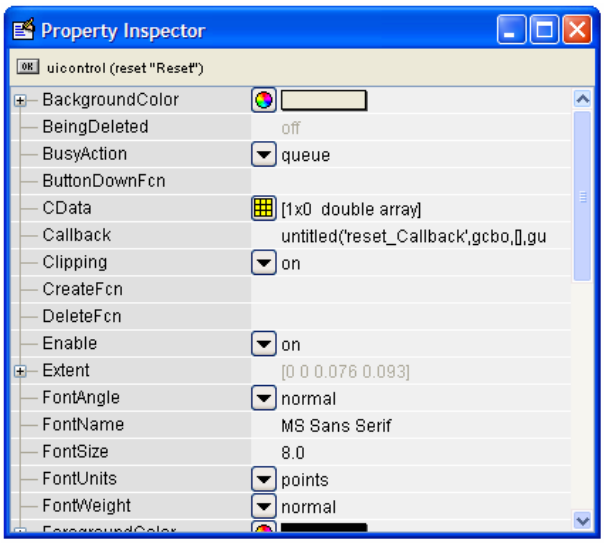
الصورة رقم 3: متصفح الأشياء.

- كل ما قد تحتاجه من تعديل في تصميم المكونات و وظائفها متوفر في مراقب الخصائص.
- يحملنا العنصر الثالث في المربع الأصفر إلى الشفرة الأولية للواجهة حيث يمكن تعديلها وفق متطلباتنا. في ما يلي جزء من الشفرة الأولية للزرين الرئيسين في الواجهة حيث يمكن تعديلهما سوى مباشرة أو عبر إستدعاء دوال أخرى كم هو الحال في إستدعاء الدالة initialize_gui.

```
% --- Executes on button press in calculate.
function calculate_Callback(hObject, eventdata, handles)
% ...
mass = handles.metricdata.density * handles.metricdata.volume;
set(handles.mass, 'String', mass);

% --- Executes on button press in reset.
function reset_Callback(hObject, eventdata, handles)
% ...
initialize_gui(gcbf, handles, true);
```

- كل ما قد تحتاجه من تعديل في تصميم المكونات و وظائفها متوفر في مراقب الخصائص.



الصورة رقم 4: مراقب الخصائص.

- ✓ يضم المستطيل الأسود و المتبقي بعض الوضائف الثانوية التي قد نحتاجها في التصميم:
  - مهمة العنصر الأول في القائمة ترتيب العناصر, حيث إذا تنقل المستخدم بين مكونات الواجهة عبر الزر Tab في لوحة المفاتيح, يكون هناك توافق بين أماكنها و ترتيبها.
  - نلاحظ في الصورة رقم 1 غياب قائمة الإختيارات أو ما يسمى باللغة الإنجليزية Menu. هذا لا يعني إستحالة إضافة هذا العنصر للواجهة, إنما ذلك ممكن عبر تفعيل الإختيار الثاني في المستطيل المعني. بذلك نجد كل اللوازم لتصميم قائمة الإختيارات و القوائم المسندلة[6] أيضا.
  - وضيفة العنصر الأخير في المستطيل الأسود في الصورة رقم 2 تحديد توزع العناصر في الواجهة خاصة عند تغير حجمها.

قبل أن أختم هذا الجزء أذكر بإمكانية الوصول لأغلب الإختيارات المذكورة أنفا عبر النقر بيمين الفأرة على المكون الذي نود تعديل خصائصه.

يولد حفظ الواجهة المتحصل عليها ملفين بإمتدادين مختلفين وهما .m و fig. يتكون الأول من شفرة الدوال التي سيتم إستدعائها لاحقا، في حين أن الثاني يضم شفرة تصميم الواجهة الرسومية, لا يمكن تعديل نصه يدويا و لابد من إعادة فتحه في برنامج التصميم guide كلما إقتضت الضرورة تعديله.

خاتمة

بعد هذه الجولة السريعة و الموجزة لكيفية إستخدام guide ننتقل إلى الطريقة الثانية, التي قد تكون أصعب نظرا لتصميم الواجهة برمجيا من الألف إلى الياء دون إستخدام واجهات رسومية أو أدوات جاهزة. و لكن إستعمال العبارة "قد" يفيد الإحتمال و الفرضية و لا يمكن التأكد إلا في نهاية الجزء الثاني.

[6] القوائم المسدلة: ContextMenu.

3. الجزء الثاني

سنتدرج في هذا الجزء في التصميم البرمجي للواجهات إنطلاقا من تصميم النافذة وصولا إلى إدراج نصوص المساعدة. علما أنني إستخدمت في هذا الدرس الإصدار 7.1 للـ matlab

و ربما تكون هناك بعض الإختلافات عند تجربة الشفرات المدرجة في هذا الدرس في إصدارات أخرى من البرنامج.

1.3. تصميم الواجهة

سنبدأ تصميم واجهة رسومية انطلاقا من بعض التعليمات البسيطة التي ربما نعلمها جميعنا. أول هاته التعليمات:

```
F_MainFigure=figure
```

هاته التعليمة ستولد واجهة رسومية فارغة و سيتم إسناد قيمة 1 كمخرج عند نجاح العملية.

إستعملنا هنا الدالة بدون أي مدخلات, و السؤال الذي يطرح نفسه هنا هو هل لهذه الدالة مدخلات؟ فربما إذا عرفنا هاته المدخلات ستصبح المهمة أسهل. لذلك سنبحث قليلا في نصوص المساعدة بإستعمال:

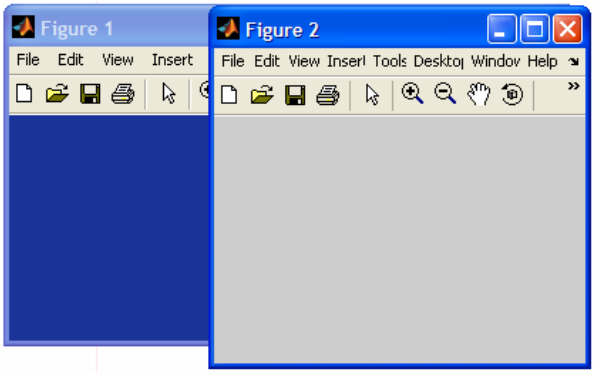
```
help figure
```

للأسف أن نص المساعدة لا يكفي لمعرفة طرق تصميم واجهة رسومية تماثل تلك التي توفرها لغات البرمجة. إذا كل ما عليك هو الإنتظار حتى نهاية الدرس.

أول خطوة في تغير الواجهة الرسومية الأصلية التي يوفرها الـ matlab تتمثل في تغير لون خلفية, جرب تنفيذ التعليمات التالية مباشرة في نافذة التعليمات:

```
F_MainFigure = figure('Color',[0.1 0.2 0.6]), figure
```

النتيجة مجسدة في الصورة رقم 3, تغير لون خلفية الواجهة من اللون الأصلي المسند لها إلى اللون الأزرق. نلاحظ في التعليمة التي ولدت هذا التغير أننا أضفنا مدخلين للدالة figure; في الحقيقة هما ليس متغيرين إنما متغير و القيمة المسندة له.

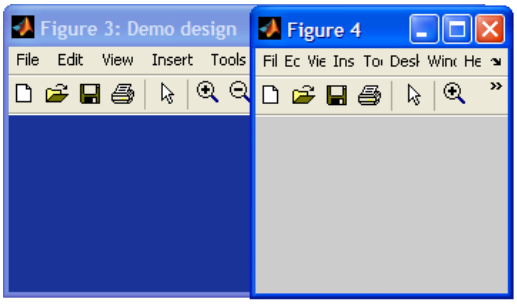


صورة رقم 1: تغير لون خلفية الواجهة

القيمة المسندة لمتغير تغير الخلفية تتمثل في مصفوفة أحادية الأبعاد متكونة من ثلاثة أرقام متراوحة بين 0 و 1. الأرقام الثلاثة تمثل الألوان الأحمر, الأخضر و الأزرق بالترتيب لذلك إذا أردت أن يطغى لون من هذه الألوان على البقية يجب أن تكون القيمة المسندة له الأكبر. نلاحظ هنا أن الـ matlab يعتمد نظام RGB[7] في التعامل مع الألوان.

نعيد الآن النظر مجددا للصورة رقم 1. نلاحظ أن الـ matlab أسند تلقائيا إسم لكل نافذة; ماذا لو أردنا وضع أسماء نختارها نحن لتكون مناسبة أكثر لمحتوى الواجهة الرسومية أو لتحمل إسم التطبيق أو غير ذلك من الإعتبارات؟ الإجابة تختصرها التعليمات التالية:

```
F_MainFigure = figure('Color',[0.1 0.2 0.6], 'Name', 'Demo design'), figure
```

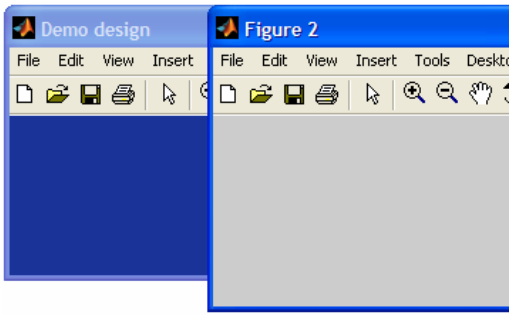


صورة رقم 2: تغير إسم الواجهة

في التعليمة السابقة و كما تلاحظون قمت بإضافة مدخلين للدالة figure, وهما متغير إسم الواجهة والقيمة المسندة له. لكن المشكل هنا هو أننا لم نتخلص بعد من تلك التسمية التلقائية التي يسندها الـ matlab و كل ما قمنا به هو إضافة تسمية أخرى للموجودة. لذلك فالخطوة الموالية في تصميمنا للواجهة الرسومية ستتمثل في التخلص كليا من التسمية التلقائية التي يسندها الـ matlab. كل ما عليك الآن هو تجربة هاته التعليمات:

```
F_MainFigure = figure('Color',[0.1 0.2 0.6], 'Name', 'Demo
design','NumberTitle', 'off'), figure
```

تم تعديل إسم الواجهة بالكيفية التي يريدها. كما يوحي له السابق, لم أعدل مجددا التسمية إنما عمت فقط بتعطيل خاصية الإضافة التلقائية لرقم الواجهة.



الصورة رقم 3: حذف ترقيم الواجهة

[7] Red, Green Blue:RGB

نتوقف مع الشفرة السابقة التي بدأت تطول و لا تزال قائمة مدخلات الدالة figure تطول, لأذكر بإمكانية إستعمال الرمز "..." كدلالة على إمتداد التعليمات على أكثر من سطر.

سنقوم هذه المرة بتعطيل صلاحية تغير حجم الشاشة. للقيام بذلك يكفي إضافة متغير Resize لقائمة مدخلات الدالة figure وإسناد القيمة off لها, كما يلي:

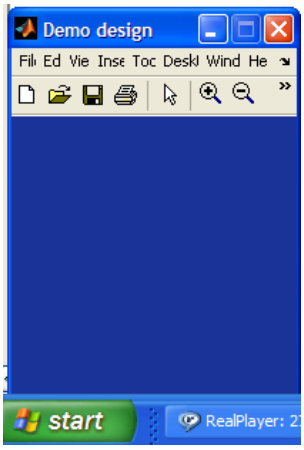
```
F_MainFigure = figure('Color',[0.1 0.2 0.6], ...
                      'Name', 'Demo design', ...
                      'Resize', 'off',...
                      'NumberTitle', 'off')
 figure
```

الغاية من حذف صلاحيات تغير حجم الواجهة الحفاظ على توزع العناصر المكونة للواجهة و عدم إنحصارها في ركن واحد عند تكبيرها. يمكن أن تحدث هذه المشاكل عندما تكون الإحداثيات المسندة للمكونات ثابتة و مستقلة عن حجم الواجهة الذي تحدده قيمة المتغير ScreenSize.

نواصل تصميمنا للواجهة بإضافة متغير جديد وهام وهو Position. هذا المدخل سيمنحنا حرية تحديد مكان ظهور الواجهة و حجمها لذلك فإن القيمة المسندة لها من نوع مصفوفة أحادية الأبعاد متكونة من أربعة عناصر; الأول و الثاني لتحديد الإحداثيتين الأفقية و العمودية و الثالث و الرابع لتحديد الطول و العرض. إذا الكود السابق يصبح كما يلي:

```
F_MainFigure = figure('Color',[0.1 0.2 0.6], ...
                      'Name', 'Demo design', ...
                      'Position', [10 60 200 200], ...
                      'Resize', 'off',...
                      'NumberTitle', 'off')
```

حددت قيمة المتغير Position بكيفية تجعل الواجهة تظهر فوق شريط المهام في الركن الأيسر للشاشة كما هو مبرز في الصورة رقم 5.



الصورة رقم 5: تحديد مكان و حجم الواجهة.

كان التعريف المسبق لقيمة المتغير Position بطريقة "بدائية". لأنه لم يكن هناك أخذ بعين الإعتبار حجم الشاشة المتغير من حاسوب لآخر, و لتجاوز هذا الإشكال يمكننا إستغلال المتغير ScreenSize

```
SCREENSIZE = get(0,'ScreenSize')
```

الذي يوفره الـ matlab و الذي يرجع حجم الشاشة في شكل مصفوفة عند إستدعائه بالكيفية التالية:

النتيجة حسب مقاييس حاسوبي:

```
SCREENSIZE =
           1           1        1440         900
```

جرب الآن الكود التالي. كيف صار تصميم الواجهة؟

```
SCREENSIZE = get(0,'ScreenSize');
F_MainFigure = figure('Color',[0.1 0.2 0.6], ...
 'Name', 'Demo design', ...
 'Position', [SCREENSIZE(1)  SCREENSIZE(2) SCREENSIZE(3) SCREENSIZE(4)], ...
 'Resize', 'off',...
 'NumberTitle', 'off')
```

ملأت الشاشة!!! هذه معلومة جديدة ربما تحتاجها عند تصميم لعبة أو غير ذلك من التطبيقات. نعود الآن لجعل الواجهة تظهر في نفس المكان الذي إخترناه سابقا ولكن هذه المرة وفق مقاييس الشاشة و بكيفية أكثر إحترافية:

```
SCREENSIZE = get(0,'ScreenSize');
FigureSize= [SCREENSIZE(1)  60*SCREENSIZE(2) round(SCREENSIZE(3)/5)
round(SCREENSIZE(4)/5)];

F_MainFigure = figure('Color',[0.1 0.2 0.6], ...
  'Name', 'Demo design', ...
  'Position',FigureSize, ...
  'Resize', 'off',...
  'NumberTitle', 'off')
```

إستعملت الدالة round لتحويل ناتج القسمة لعدد طبيعي و يمكن أيضا إستعمال الدالة floor. هاتان الدالتان غير متماثلتين و تختلف النتيجة حسب ما بعد الفاصلة. لمعرفة الفرق بينهما جرب مثلا التعليمات التالية:

```
floor(5.93), round(5.93)
```

يمكن للمطور إضافة عدة تعديلات أخرى على الواجهة الرسومية عبر إدراج خاصيات أخرى. يبرز الجدول رقم 1 قائمة بهاته الخصائص و وضيفة كل منها.

الخاصية	الوضيفة
Color	تعديل لون الواجهة
Menubar	إبقاء أو حذف شريط المهام
Name	تسمية الواجهة الرسومية
Numbertitle	تعديل ترقيم الواجهة
Parent	تحديد الواجهة الأم
Position	تحديد المكان و المقاييس
Resize	صـــلاحية تعديل حجم الشاشة
Tag	تحديد المؤشر
Toolbar	إبقاء أو حذف شريط الأدوات
Userdata	بيانات المستخدم
Visibile	إظهار أو إخفاء الواجهة

الجدول رقم 1: خصائص الواجهة.

بعد التعرَف على جملة من الإختيارات التي تمكننا من تصميم واجهة رسومية في الـ matlab بالمواصفات التي نريدها, سننتقل إن شاء الله إلى الجزء الخاص بإضافة الأزرار و الإطارات و غيرها من المكونات التي نعرفها في باقي لغات البرمجة.

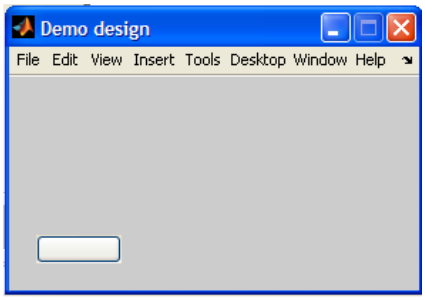
2.3. إدراج المكونات

الدالة الرئيسية في الجزء الموالي هي الدالة uicontrol و التي سيعتمد عليها باقي تصميمنا. سنبدأ إذا التصميم بإستدعاء هذه الدالة في نافذة الأوامر كانت النتيجة واجهة رسومية تحتوي زر دون نص في ركنها الأسفل الأيسر. لكن مجددا نص المساعدة حول كيفية إستعمال هذه الدالة غير كاف. إذا سنبدأ في فهم كيفية إستعمالها لجعل هذا الزر ذى وضيفة.

في باقي لغات البرمجة مثل C# و Delphi و غيرها نعرّف الواجهة ثم نضيف لها الأزرار و باقي المكونات. بحيث أن كل مكون ينتمي لواجهة دون غيرها عبر تعريفه داخل القسم المعرف للواجهة أو تحديد الواجهة التي ينتمي لها. يرث الـ matlab أيضا أحد هذه المبادئ و يفرض تحديد الواجهة المنتمي لها المكون. لذلك نعود إلى الشفرة التي كتبناها سابقا لتصميم الواجهة و نضيف لها الجزء المتعلق بالزر كما يلي:

```
SCREENSIZE = get(0,'ScreenSize');
FigureSize= [5*SCREENSIZE(1)  70*SCREENSIZE(2) round(SCREENSIZE(3)/5)
round(SCREENSIZE(4)/6)];
F_MainFigure = figure('Color',[0.8 0.8 0.8], ...
                      'Name', 'Demo design', ...
                      'Position',FigureSize, ...
                      'Resize', 'off',...
                      'NumberTitle', 'off')
b = uicontrol('Parent', F_MainFigure)
```

و هكذا تمت إضافة الزر إلى الواجهة التي عرفناها مسبقا كما هو مجسد في الصورة رقم 6.



الصورة رقم 6: إضافة عنصر للواجهة.

ملاحظة: في الصورة السابقة إستعملت عبارة "مكون" عوض عن "زر" لأن الدالة uicontrol لا تخص فقط الأزرار إنما باقي المكونات أيضا, و المكون التلقائي الذي تضيفه هو زر. سنتعرف لاحقا على كيفية إستعمال هذه الدالة في إضافة بقية أنواع المكونات.

نضيف مجموعة من الإختيارات الإضافية لتعريف شفرة الزر لتصبح كما يلي:

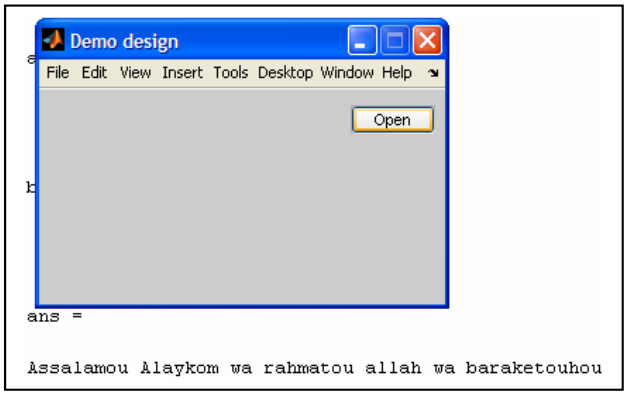
```
b = uicontrol('Parent', F_MainFigure,...
  'BackgroundColor',[0.701961 0.701961 0.701961], ...
  'Callback','Function1', ...
  'Interruptible', 'off', ...
  'Position',[222 120 60 20], ...
  'String','Open')
```

تمكن الخاصية Callback من تحديد إسم الدالة التي نريد إستدعائها. كتبت Function1 كمثال بحيث أنه سبق تعريف هذه الدالة. و يمكن أيضا كتابة التعليمات مباشرة كقيمة للخاصية Callback إذا كان عدد التعليمات محدودا, بالطريقة التالية:

```
'Callback','sprintf(''Assalamou Alaykom wa rahmatou ALLAH wa
baraketouhou'')', ...
```

نلاحظ هنا تكرار الرمز ' و ذلك لأنه محجوز في الـ matlab لتحديد النصوص. لذلك إذا اردنا إستعماله داخل نص فلابد من سبقه بمثيله.

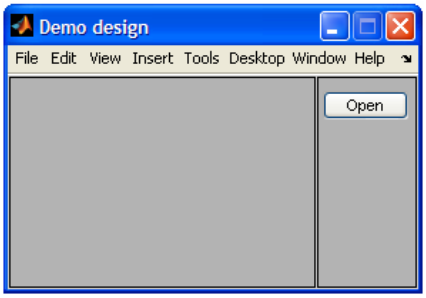
الخاصية الثانية التي ربما تحتاج الشرح هي Interruptible; في توفر إمكانية تعطيل الوضيفة المسندة للمكون, كما يوحي إسمها. يعد هذا المتغير ضروري و مستعمل في محله عند تنفيذ عمليات تطلب مدة زمنية كبيرة أو لا متناهية. إثر التعديل الأخير يصبح شكل الواجهة التي صممناها ومخرجاتها كما هو مبرز في الصورة رقم 7.



نواصل تصميمنا بإضافة إطار أو ما يسمى Frame في بعض لغات البرمجة للواجهة التي صممناها. الدالة التي سنستعملها هنا هي ذاتها الدالة السابقة و لكن بتعديل مختلف لمدخلاتها. إذا صارت الشفرة الجديدة التي سنضيفها كما يلي:

```
h_f_background = uicontrol('Parent', F_MainFigure, ...
  'BackgroundColor',[0.701961 0.701961 0.701961], ...
  'Position',[2 2 215 148], ...
  'Style','frame')
```

تعريف مشابه لشفرة الزر و لكن بإدراج خاصية جديدة و هي Style. تمكننا هذه الأخيرة من تحديد نوع المكون وفق القيمة المسندة له و التي تنتمي لمجموعة من القيم المعرفة مسبقا في الـ matlab (الجدول رقم 3). أذكر هنا أنه لابد من تعريف الإطار قبل تعريف باقي المكونات و بعد كتابة الشفرة الخاصة بالواجهة. بهذه الكيفية سنحدد الواجهة المنتمي لها الإطار و سنتجنب حجب باقي المكونات من قبل هذا الأخير. قمت بإضافة شفرة الإطار و آخر مماثل لها مع تعديل الإحداثيات و الحجم و كانت النتيجة الصورة رقم 8.

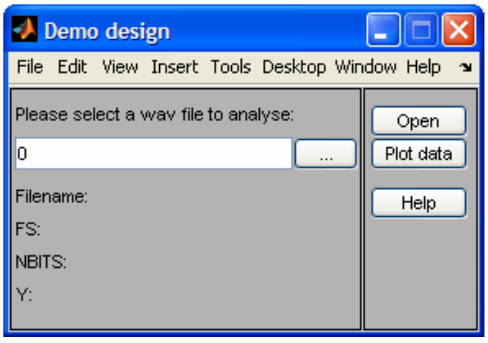


الصورة رقم 8: إضافة إطارات للواجهة

واصلت تصميم الواجهة لأحصل في نهاية المطاف ماهو مجسد في الصورة رقم 9. الجديد في هذه الأخيرة تواجد مكونات في الواجهة الرسومية دون تقديم مسبق. إضافة نص أو حقل نص أو غير ذلك يكون عبر تغير الخاصية style للدالة uicontrol المعرفة للمكونات. مثال:

```
h_f_background = uicontrol('Parent', F_MainFigure, ...
  'BackgroundColor',[1 1 1], ...
  'Position',[2 2 215 148], ...
  'Style','edit', …
  'Callback','Method_To_Call', ...
  'String','0', ...)
```

يمكننا إضافة عدة مكونات أخرى، تعادل ما يوفره guide على سبيل الذكر edit, popupmenu, text.



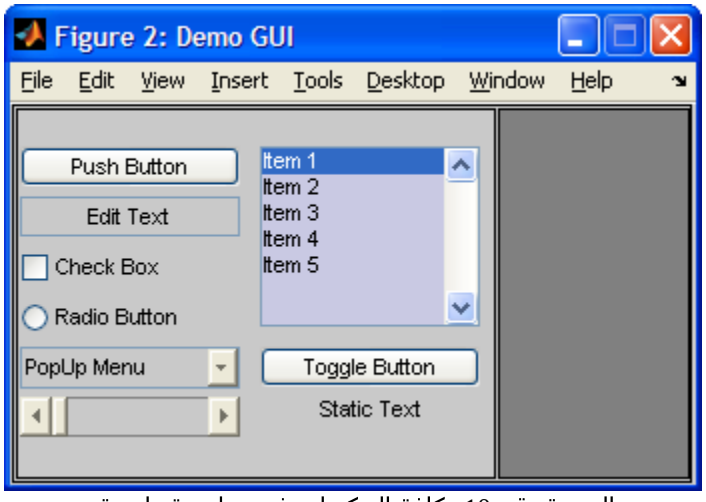
الصورة رقم 9: الواجهة النهائية.

تتشارك أغلب المكونات في الخصائص و تتماثل من حيث التصميم مع إختلافات طفيفة في الخصائص. يحوصل الجدول رقم 2 أغلب الخصائص التي يمكن إستغلالها في تصميم المكونات وسنرى في الفقرة التالية كيفية الحصول على قيمة مكون ما.

الخاصية	الوضيفة
Parent	تحديد الواجهة الأم
BackgroundColor	تحديد لون الخلفية
Position	تحديد المكان و المقاييس
Style	تحديد المكون
Tag	تحديد المؤشر
String	تحديد النص
FontWeight	تحديد نوع الكتابة
Callback	الإستدعاء
Interruptible	التعطيل
Value	تحديد القيمة

الجدول رقم 2: خصائص المكونات.

لا أنصح بإدراج كافة المكونات في واجهة واحدة إلا عند الضرورة القصوى, لأن ذلك يؤثر في إخراج الواجهة و يضفي عليها تعقيد نحن في غنى عنه. تمثل الصورة رقم 10 إستثناء لأن الغاية منها إبراز جميع المكونات التي يمكن إدراجها في الواجهة. علما أن المكون الذي يمثل الخلفيات هو المكون Frame.



الصورة رقم 10: كافة المكونات في و اجهة واحدة.

لإدراج هذه المكونات يكفي إسناد الأسماء التالية للخاصية style لكل مكون:

Push Button	Radio Button	Slider	Check Box	Frame
Static Text	PopUp Menu	ListBox	Edit Text	Toggle Button

الجدول رقم 3: أسماءالمكونات.

3.3. التفاعل بين المكونات

كل ما سبق ذكره في تصميم الواجهات الرسومية و المكونات كاف لتصميم واجهات غير تفاعلية, ونادرا ما نحتاج لذلك كمشروع مستقل. لذلك وجب الحديث عن كيفية تمرير القيم بين المكونات والربط فيما بينها لتوفير أكثر صلاحيات للمستخدم تتجاوز مشاهدة نتيجة العمليات الرياضية.

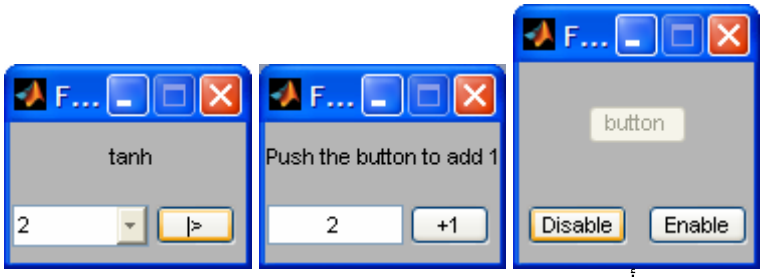
نبدأ بمثال بسيط يتمثل في إظهار صورة على الواجهة حسب إختيار المستخدم. واجهة التطبيق مجسدة في الصورة رقم 11, و لتصميم تلك الواجهة يكفي تكرار ما تم ذكره سابقا مرار. علما أن الجديد هنا يكمن في كيفية إظهار الصورة عبر النقر على الزر المسمى بإسم الصورة.

```
function ShowImage(ImageName)
switch ImageName
    case 'cameraman'              % 'ShowImage cameraman'
     imshow('cameraman.tif');
    case 'peppers'                % 'ShowImage peppers'
     imshow('peppers.PNG');
    case 'saturn'                 % 'ShowImage saturn'
     imshow('saturn.png');
end
```

تمثل التعاليق المضافة في الشفرة السابقة القيمة التي ينبغي إسنادها للخاصية Callback للأزرار الثلاثة, كما يجبذ تعريف الدالة ShowImage في ملف مستقل محفوظ في نفس مسار تواجد شفرة تصميم الواجهة.

الصورة رقم 11: التفاعل بين المكونات

يهدف المثالي التالي لإبراز كيفية التحكم في خصائص مكون في الواجهة من قبل مكون آخر كالتفعيل و التعطيل. هذه العمليات تكون عن طريق الخاصية Tag, و هي عبارة عن مؤشر للمكون يمكن التحكم فيه من خلالها كما هو مجسد في الصورة 11.أ.



ج ب أ

الصورة رقم 12: التفاعل بين المكونات

نلاحظ غياب لائحة الإختيارات من الواجهة و ذلك يعود لإدراج الخاصية MenuBar عند تصميم الواجهة.

```
Mainfigure=figure('Color',Bckgrnd, ...
                  'MenuBar', menubar,...
                  'PaperType','a4letter', ...
                  'Resize', 'Off',...
                  'Position',[150,150, 100,70],...
                  'Name', 'Demo GUI');
```

تحتوي الواجهة 12.أ ثلاثة أزرار, إثنان منها لتفعيل و تعطيل الزر الثالث الذي أسندت القيمة ControlTagللخاصية Tag التابعة له.

```
function Disable_Enable(action)
switch action
    case 'Disable'
     set(findobj('Tag','ControlTag'),'Enable','off');
    case 'Enable'
     set(findobj('Tag','ControlTag'),'Enable','on');
    case 'Show'
        imshow('cameraman.tif');
end
```

المثال الموالي (الصورة 12. ب) تجسيد لكيفية قراءة و تعديل نص مكون ما.

كلما نقر المستخدم على الزر يتم قراءة نص خانة الكتابة, تحويل نصها لرقم, إضافة 1 له ثم تعديل نصها, علما أن مؤشر خانة النص هو t_2.

في المثال السابق إستخدمت أمرين في أمر واحد, و هما البحث عن مؤشر المكون و تعديل خاصيته. في هذا المثال جزأت الأوامر أكثر لتيسر فهمها. كخطوة أولى, ينبغي البحث ما إن وجد المؤشر t_2. إذا تكللت عملية البحث بالنجاح و كانت قيمة المؤشر غير فارغة, نتمكن عندها من قراءة أو تعديل إحدى خصائصه عبر إستخدام الدالتان get و set.

```
function Increment
    handle = findobj('Tag','t_2');
    if ~isempty(handle)
       CurrentVal = str2num(get(handle, 'String'));
       CurrentVal=CurrentVal+1;
       set(handle,'String',int2str(CurrentVal));
    end
end
```

في المثال المجسد في الصورة 12.ج إستخدمت المكون popup-menu. وهو أصعب من الأزرار وحقول النصوص من ناحية التعامل لكونه يضم بيانات في شكل مصفوفات, واحدة ظاهرة للمستخدم و الثانية تضم القيم المقابلة لكل نص. لتيسير التعامل مع هذا المكون يمكن إستخدام متغيرات عامة مشتركة بين شفرة الواجهة و الشفرة النواة التي تضم الوضائف الرئيسية و الدوال. بهذه الكيفية يتحقق التوافق بين الإختيار الذي فعله المستخدم و القيمة المقابلة له.

```
function PopuModifier
% Shared parameters
global popup_strD;
global popup_strV;
global popupValue;
global p_popup;

    handle = findobj('Tag','popup_tag');
    if ~isempty(handle)
     eval(['g_str = popup_strV'])
     old_g = deblank(g_str(popupValue,:))
     SelectedIndex = get(p_popup, 'Value')
     set(findobj('Tag','t_text'),'String', deblank(g_str(SelectedIndex,:)));
    end
```

في ما سبق إعتمدنا الدالة الرئيسية uicontrol. و إسمها إختصار للنص User Interface Control. لهذه الدالة عدة "أخوات" مفيدة في تصميم الواجهات الرسومية التفاعلية, جمعت جلها في الجدول التالي:

>>ls C:\MATLAB71\toolbox\matlab\uitools\ui*			
uitoolbar.m	uisetcolor.m	uigridcontainer.m	uibuttongroup.m
uitoolfactory.m	uisetfont.m	uiload.m	uiclearmode.m
uitree.m	uisetpref.m	uiopen.m	uicontainer.m
uitreenode.m	uistack.m	uipanel.m	uiflowcontainer.m
uiundo.m	uisuspend.m	uipushtool.m	uigetdir.m
uiwait.m	uitab.m	uiputfile.m	uigetfile.m
uitoggletool.m	uitabgroup.m	uirestore.m	uigetpref.m
uisave.m	uitable.m	uiresume.m	uigettool.m
uigettoolbar.m			

كما نلاحظ, هذه الدوال مجموعة في مجلد واحد وهو مجلد الأدوات لتصميم الواجهات التابع للـ matlab. أترك لكم إكتشاف وضائف هذه الدوال.

4.3. نصوص المساعدة

ليكون التطبيق متكامل و إحترافي فلا بد من إضافة نصوص مساعدة و معلومات حول المطور, الإصدار و غير ذلك من البيانات التي قد نحتاج لإبرازها لمستخدمي التطبيق. يمكننا إضافة نصوص المساعدة بعدة طرق سوى بتصميم واجهات رسومية مخصصة لنصوص المساعدة أو بكل بساطة نستغل ما يوفره الـ matlab من أدوات مخصص لهذا الغرض. لكن قبل المرور لصلب الموضوع أذكر بإمكانية إدراج نصوص مساعدة داخل أي شفرة نكتبها و يظهرها البرنامج بنفس كيفية نصوص المساعدة التابعة لدواله.للقيام بذلك لا بد من كتابة نص المساعدة مباشرة بعد تعريف الدالة شرط أن لا يحتوي سطر فارغ. و إذا إحتجت لإدراج سطر فارغ للفصل بين فقرات نص المساعدة فلابد من إدراج سطر فارغ في شكل تعليق.

```
function [out1, out2]=DemoHelp(varargin)
% This is a the help text of the method DemoHelp...
%
%   See also ShowHelp

% 14.7.2009
% IBen Laiid

if nargin < 1
MsgBox('Invalid Input Arguments.','Demo GUI Design');
end
% Continue your code here
```

في المثال أعلاه تجسيد لما سبق ذكره خاصة في ما يخص ترك سطر فارغ داخل نص المساعدة. كما أدرجت سطر فارغ كليا و ليس في شكل تعليق و ذلك لإدراج معلومات حول كاتب الشفرة و تاريخ كتابتها دون أن تظهر هذه المعلومات في نص المساعدة عند القيام بذلك في نافذة الأوامر.

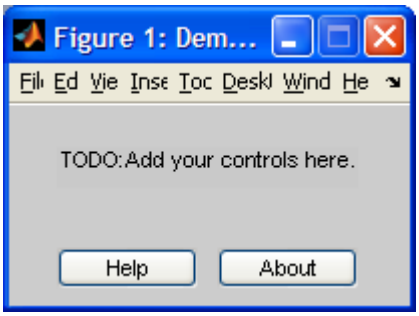
```
>> help DemoHelp
This is a the help text of the method DemoHelp

See also ShowHelp
```

نعود لتصميم الواجهات الرسومية و إدراج نصوص المساعدة بطريقة الـ matlab, أعني الواجهة التي تظهر عند النقر على الزر F1 في أي واجهة للبرنامج. بإيجاز تمكن الدالة helpwin إمكانية إظهار نص كما لو أنه تابع لنصوص المساعدة للـ matlab.

تعتمد هذه الدالة على مدخلين رئيسين و هما نص المساعدة و عنوانه, شرط أن تكون أسطر النص محررة بصفة مستقلة بعضها عن بعض و لها نفس الطول لأن الدالة تتعامل مع النص على أنه مصفوفة.

تصور الواجهة الرسومية رقم 13 مثال لإدراج نص مساعدة و نص لإظهار معلومات حول التطبيق. شفرة التطبيق مقسمة إلى ملفين أحداهما مخصص لتصميم الواجهة و إستدعاء الدالة المعرفة في الملف الثاني.



الصورة رقم 13: تطبيق نصوص المساعدة.

لا أرى في هذا المثال ما يستوجب الشرح خاصة أنه سبق شرح مختلف أجزاءه سابقا.

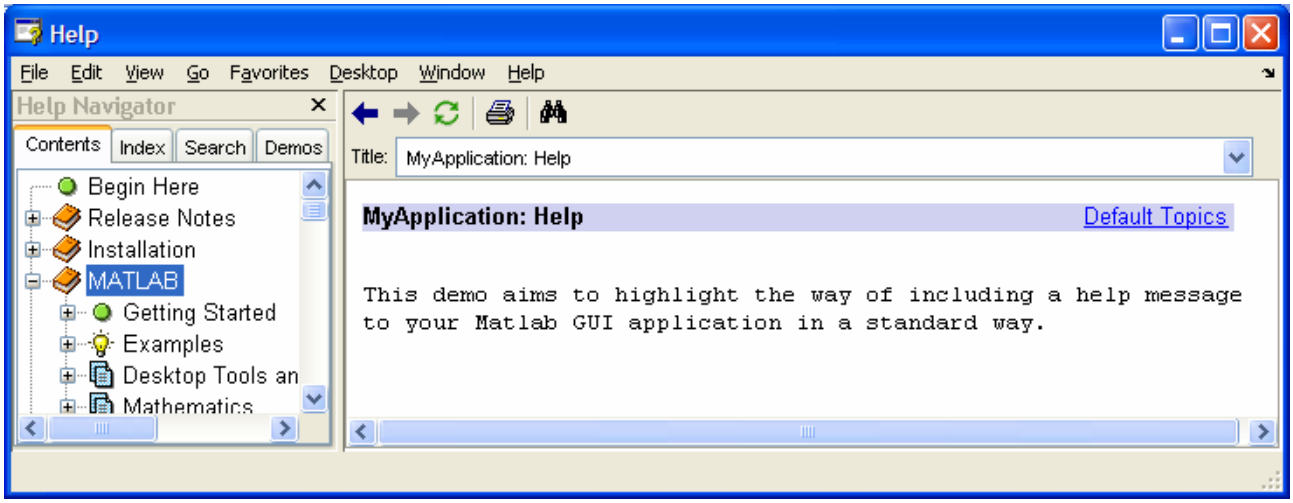
```
function InterfaceManager
%=======================================
Y_Pos=10; X_Pos=25; Width=70; Bckgrnd=[0.789 0.789 0.789];
%=======================================
Mainfigure=figure('Color',[0.8 0.8 0.8], ...
                  'PaperType','a4letter', ...
                  'Resize', 'Off',...
                  'Position',[150,150, 200,100],...
                  'Name', 'Demo GUI');
Text   =uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',Bckgrnd, ...
                  'String', 'TODO:Add your controls here.',...
                  'Position', [X_Pos Y_Pos+50 Width+80 20],...
                  'Style', 'text');
%=======================================
button1=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',Bckgrnd, ...
                  'String', 'Help',...
                  'Position', [X_Pos Y_Pos Width 20],...
                  'Style', 'Push Button',...
                  'Callback', 'ShowHelp Help ');
button2=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',Bckgrnd, ...
                  'String', 'About',...
                  'Position', [X_Pos+80 Y_Pos Width 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'ShowHelp About');
end
```

يمكن جمع مختلف نصوص المساعدة في شفرة واحدة يتم التميز بينها عبر إستخدام الجملة الشرطية . و يكفي إستدعاء الدالة و تحديد عنوان نص المساعدة لتقوم الدالة المعرفة في آخر الشفرة باللازم.

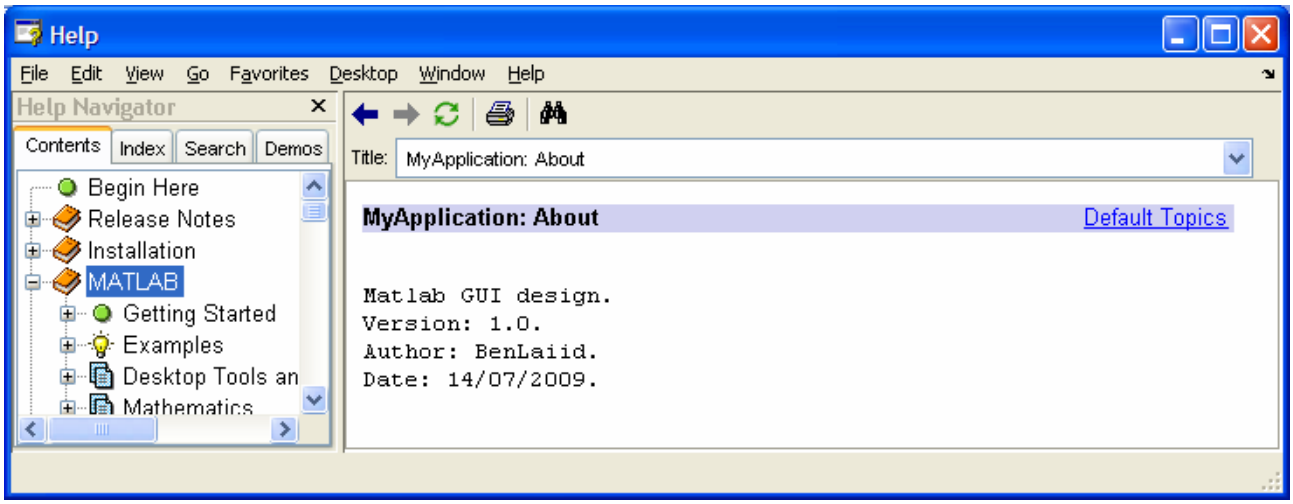
```
function ShowHelp(which_help)
 switch which_help
 case 'Help'
  helptitle = 'MyApplication: Help';
  helptext=[ ...
    'This demo aims to highlight the way of including a help message    '
    'to your Matlab GUI application in a standard way.                  '];

 case 'About'
    helptitle = 'MyApplication: About';
    helptext=[ ...
    'Matlab GUI design.                                                 '
    'Version: 1.0.                                                      '
    'Author: BenLaiid.                                                  '
    'Date: 14/07/2009.                                                  '];
end
helpwin(helptext, helptitle);
```

أخيرا نتحصل على واجهات مثل الصورة 14 و 15. مثل هذه الواجهات تفتح الشهية لمزيد تعلم الـ matlab والسعي لتطوير تطبيقات أكثر فأكثر تعقيدا و إحترافية طبعا.



الصورة رقم 14: واجهة نصوص المساعدة.



الصورة رقم 15: الواجهة "حول البرنامج".

المفيد

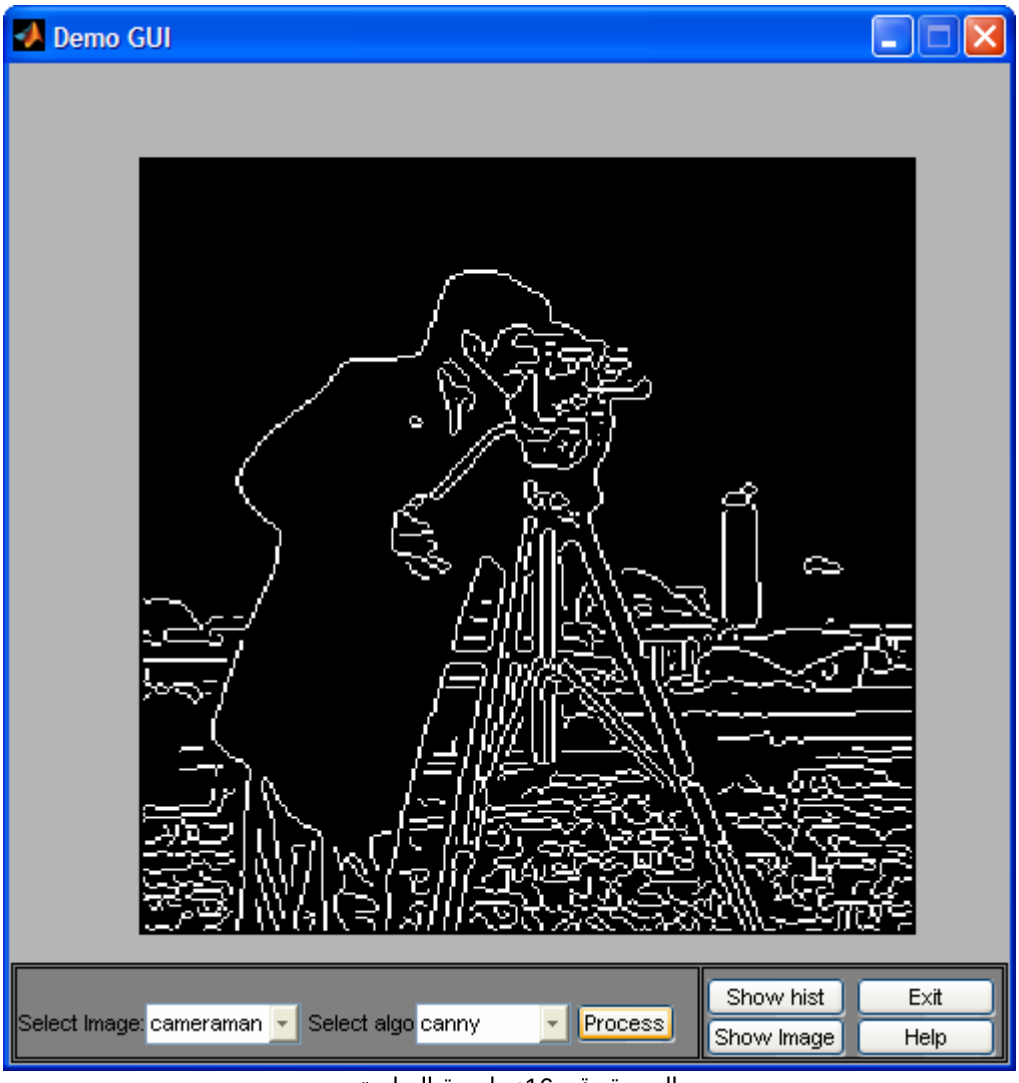
أعلم أنني ثرثرت كثيرا في ما سبق في هذا الدرس. سندخل إذا في صلب الموضوع مع تطبيقات متكاملة لتجسيد ما سبق ذكره.

التطبيق: معالج الصور.

الغاية من هذا التطبيق

- المقارنة بين أربعة خوارزميات مختلفة Canny, Sobel, Roberts, Prewitt عبر تطبيقها على أربعة صور مختلفة تابعة للـ matlab.
- كيفية تمرير البيانات بين عدة واجهات رسومية عبر تعريف المتغيرات من نوع global.
- إستدعاء واجهة رسومية ثانية مصممة بطريقتنا و تظهر نص المساعدة عوض عن إستخدام الأداة helpwin التابعة للـ matlab.

و العديد من النقاط الأخرى التي أترك لكم الفرصة لإكتشافها.



الصورة رقم 16: واجهة التطبيق.

في ما يلي شفرة التطبيق, علما أنها تمتد على ملفين منفصلين يحملان نفس إسم الدالة الرئيسية.

```
function InterfaceManager
% Shared parameters
global popup_strD;
global popup_strV;
global popupValue;
global p_popup;
global p_popupM;
global popup_ModeValue;
global popup_ModestrD;
global popup_Mode_strV;
% Initialize the default value of the popup
popupValue = 1;
popup_ModeValue=1;
% Popup-menu items texts
popup_strD = 'cameraman|circuit|kids|trees';
popup_ModestrD='prewitt|sobel|roberts|canny|all';
% Popup-menu items values
popup_strV = [ 'cameraman'; 'circuit  '; 'kids     '; 'trees    '];
popup_Mode_strV=['prewitt';'sobel  ';'roberts';'canny  ';'all    '];
%=======================================
Y_Pos=10; X_Pos=4; Width=50;
Figure_Width=500; Figure_Height=500;
Bckgrnd=[0.71 0.71 0.71];
%=======================================
Mainfigure=figure('Color',Bckgrnd, ...
                  'MenuBar', menubar,...    % remove the menubar
                  'Numbertitle','off',...
                  'PaperType','a4letter', ...
                  'Position',[150,150, Figure_Width,Figure_Height],...
                  'Resize', 'off',...
                  'Name', 'Demo GUI',...
                  'Tag', 'f_Mainfigure');
MainFrame=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'Position', [2 2 Figure_Width-2 Figure_Height/10],...
                  'Style', 'frame',...
                  'Tag','popup_tag');
SideFrame=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'Position', [347 4 151 46],...
                  'Style', 'frame',...
                  'Tag','popup_tag');
SideFrame=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'Position', [4 4 342 46],...
                  'Style', 'frame',...
                  'Tag','popup_tag');
%=======================================
txt_box=uicontrol('Parent', Mainfigure,...
                'BackgroundColor',[0.5 0.5 0.5], ...
                 'String', 'Select Image:
Select algo:',...
                 'Position', [X_Pos+1 Y_Pos+3 Width+150 15],...
                 'Style', 'text',...
                 'Tag','t_text');
%=======================================
%...
```

```
%=======================================
p_popup=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[1 1 1], ...
                  'Position', [X_Pos+65 Y_Pos+2 Width+28 20],...
                  'Style', 'popupmenu',...
                  'String',popup_strD ,...
                  'Value',popupValue,...
                  'Tag','popup_tag');
p_popupM=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[1 1 1], ...
                  'Position', [X_Pos+200 Y_Pos+2 Width+28 20],...
                  'Style', 'popupmenu',...
                  'String',popup_ModestrD ,...
                  'Value',popup_ModeValue,...
                  'Tag','popup_Mode_tag');
%=======================================
btn_Proc=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Process',...
                  'Position', [X_Pos+280 Y_Pos+2 Width 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'kernel Process');
btn_Orig=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Show Image',...
                  'Position', [X_Pos+345 Y_Pos-5 Width+20 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'kernel show');
btn_hist=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Show hist',...
                  'Position', [X_Pos+345 Y_Pos+15 Width+20 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'kernel hist');
btn_Help=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Help',...
                  'Position', [X_Pos+420 Y_Pos-5 Width+20 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'kernel Help');
btn_exit=uicontrol('Parent', Mainfigure,...
                  'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Exit',...
                  'Position', [X_Pos+420 Y_Pos+15 Width+20 20],...
                  'Style', 'pushbutton',...
                  'Callback', 'kernel Exit');
%=======================================
% Show default image
imshow('cameraman.tif')
```

هذه شفرة الملف الثاني و هي النواة التي تحتوي الوضائف الرئيسية التي يتم إستدعائها من قبل مكونات الواجهة.

```
function kernel(action)
global I; % Read The image and store it in the global variable I
switch action
    case 'Process'
        Process()
    case 'show'
        imshow(I)
    case 'hist'
        figure, imhist(I);
    case 'Help'
       Help();
    case 'Exit'
       App_Exit();
end
%=======================================
function ProcessImage(I, Mode)
Per = edge(I,'prewitt');
Sob = edge(I,'sobel');
Rob = edge(I,'roberts');
Can = edge(I,'canny');
switch Mode
    case 1 % 'Per'
     imshow(Per);
    case 2 % 'Sobel'
     imshow(Sob);
    case 3 %'Roberts'
     imshow(Rob);
    case 4 %'Canny'
     imshow(Can);
    case 5  % 'All'
     figure
     subplot(2,2,1), subimage(Per), title('prewitt')
     subplot(2,2,2), subimage(Sob), title('sobel')
     subplot(2,2,3), subimage(Rob), title('roberts')
     subplot(2,2,4), subimage(Can), title('canny')
end
%=======================================
function Process()
% Shared parameters
global popup_strD;
global popup_strV;
global popupValue;
global p_popup;
global p_popupM;
global popup_ModeValue;
global popup_ModestrD;
global popup_Mode_strV;
global I;
% Find if there is an object having the tag popup_tag
handle = findobj('Tag','popup_tag');
if ~isempty(handle)
      eval(['g_str = popup_strV;']);
      SelectedIndex = get(p_popup, 'Value');
      ImageName= deblank(g_str(SelectedIndex,:));

  switch ImageName
    case 'kids'
     I=imread('kids.tif');
    case 'cameraman'
     I=imread('cameraman.tif');
%...
```

```
    case 'trees'
     I=imread('trees.tif');
    case 'circuit'
     I=imread('circuit.tif');
  end
end % end if

handle1=findobj('Tag','popup_Mode_tag');
if ~isempty(handle1)
      eval(['g_str = popup_Mode_strV;']);
      ProcessImage(I, get(p_popupM, 'Value'));
end % end if
%=======================================
function Help
HelpFigure=figure('Color',[0.71 0.71 0.71], ...
                  'MenuBar', menubar,...           % remove the menubar
                  'PaperType','a4letter', ...      %'Resize', 'Off',...
% disable resize property
                  'Position',[150,150, 200,200],...
                  'Resize', 'off',...
                  'Numbertitle','off',...
                  'Name', 'Demo GUI: Help',...
                  'Tag','f_HelpFigure');
SideFrame=uicontrol('Parent', HelpFigure,...
                    'BackgroundColor',[0.5 0.5 0.5], ...
                    'Position', [2 2 198 198],...
                    'Style', 'frame',...
                    'Tag','popup_tag');
txt_box=uicontrol('Parent', HelpFigure,...
                 'BackgroundColor',[0.5 0.5 0.5], ...
                  'String', 'Write some blabla here ...',...
                  'Position', [4 20 150 150],...
                  'Style', 'text',...
                  'Tag','t_text');
%=======================================
function App_Exit
Tags = ['f_HelpFigure'
        'f_Mainfigure'];
 for i=1:size(Tags,1)
   handle = findobj('Tag', deblank(Tags(i,:)));
   if ~isempty(handle)
     close(handle);
   end
 end
```

4. خاتمة

و صلنا إلى ختام هذا الدرس و قد إطلعنا فيه على أساسيات تصميم الواجهات الرسومية في الـ matlab بطريقتين مختلفتين. الخلاصة أن الأداة يسيرة الإستخدام و لكنها لا توفر الحرية الكافية و المرونة التي قد نجدها عند التصميم البرمجي. يبقى الكثير من التفاصيل و المعلومات حول كيفية تصميم الواجهات و لم أذكر في هذا الدرس إلا الأساسيات التي قد نحتاجها لتعلم هذا النوع من البرمجة المثير و الممتع في الآن ذاته و للمزيد من التفاصيل يمكن مراجعة الرابط [1].

أخيرا أسئل الله عز و جل أن يجعل هذا العمل خالصا لوجهه الكريم و أن يكون حافزا لكل ذا علم و إن كان متواضع ليعمل على نشر علمه و مساعدة غيره لإنشاء جيل جديد لا يحني الجبين إلا لله عز وجل. فلقد ورثنا الذل عن آبائنا و لا داعي لنورِّثه لأبنائنا.

المراجع

[1]
www.mathworks.com/access/helpdesk/help/techdoc/index.html?/access/helpdesk/help/techdoc/ref/figure_props.html

لكل الإستفسارات, الإقتراحات و التصويب المتعلق بمحتويات هذا الكتاب بالإمكان الإتصال بي عبر البريد الإلكتروني التالي:

Ben_Laiid@laposte.net