



لغة البرمجة العربية الجميلة

# لماذا البرمجة بكلمات؟

- لكي لا تظل المصطلحات عباراتٍ صمّاء:
- Class صار "فصيلة"، object صار "كائن"، ...
- لكي يتعلم الأطفال البرمجة، ويشارك المجتمع كله في النهضة العلمية.
- لأنها قوية كلغة وليس فقط لأنها مبنية على العربية.
- لأنها جميلة!

# إنها تكبر!

ملف تحرير برنامج اختبار رمز خاص مساعدة

جديد فتح حفظ تنسيق تنفيذ

هذه هي النسخة الأولى لشهر إبريل 2011. حمل أحدث نسخة من [www.kalimat-lang.com](http://www.kalimat-lang.com)

gui\_calculator.k

النافذة = النافذة. الحالية ()  
أزرار الأرقام = مصغوفة (10)  
أحد أزرار الأرقام = قناة ()  
زر. عملية = قناة ()

لكل أ من 1 إلى 9:  
أزرار الأرقام [أ] = زر جديد  
أزرار الأرقام [أ]: حدد المكان (40+ باقي. قسمة (أ-1, 3) × 30 + 10 × (-4)  
تقريب ((أ-1) ÷ 3))  
أزرار الأرقام [أ]: حدد الحجم (30, 30)  
أزرار الأرقام [أ]: حدد النص (كنص (أ))  
النافذة: اصف (أزرار الأرقام [أ])

الرسائل المتغيرات

ابريل 2011

ملف تحرير برنامج اختبار مساعدة

جديد فتح حفظ تنفيذ تحقق

بدون عنوان

اطبع "ما اسمك؟"  
اقرأ م  
اطبع "وماذا تريد؟"  
اقرأ ن  
اطبع "حسنًا يا م، م، إليك ب"، ن

تنفيذ البرنامج

ما اسمك؟  
امينة  
وماذا تريد؟  
حصان صغير  
حسنًا يا أمينة إليك بحصان صغير

مايو 2010

# هيا نتعلم كلمات!

- نظرة عامة
- أوامر الإدخال والإخراج
- أوامر التحكم
- المصفوفات والقواميس
- الإجراءات والدوال
- الرسم والأطراف
- الكائنات والفصائل
- البرمجة المتوازية
- واجهات الاستخدام الرسومية (GUI)

# نظرة عامة

- البرنامج مكون من أوامر، كل أمر على سطر، ويكتب مباشرةً بدون الحاجة لتعريف شيء مثل main (كما في الـ C مثلاً)
- لا يوجد type declarations، والمتغير يأخذ أي نوع في أي وقت، لكن لا بد من تعريف المتغير قبل استخدامه
- يمكن تعريف الإجراءات والدوال بأي ترتيب، في أول البرنامج أو آخره
- كلمات تميّز بين الإجراءات والدالة. الإجراء مثل void function في اللغات الشهيرة، وهذا التمييز لأهداف تعليمية

# الإدخال والإخراج

اطبع "مرحبا"

اطبع "النتيجة هي"،  $13+12$

اطبع "بوجي و..."

اطبع "طمطم"

اطبع بعرض (4) س، بعرض (5) ص

- أمر **اطبع** يطبع تعبيراً أو أكثر على نفس السطر، وينتقل للسطر التالي بعدها.
- لو أردت منع الانتقال للسطر التالي ضع ثلاث نقاط ... في آخر الأمر
- لو أردت طباعة رقم بحيث يضيف مسافات لو كان الرقم اصغر من عدد معين من الخانات، استخدم **بعرض**

# الإدخال والإخراج

اقرأ م

اقرأ م ، #ن

اقرأ "ما سنك؟" ، السن

اقرأ "ما سنك و عنوانك؟" ، #السن ، العنوان

- يقرأ نصاً من المستخدم مثل cin أو ReadLine
- يأتي بعده متغير أو أكثر
- اختياريًا يمكن بدءه برسالة نصية تظهر قبل أن يُدخل المستخدم القيم المطلوبة
- لو سبقت اسم المتغير بعلامة # فإنه يقرأ قيمةً عدديةً بدلاً من قيمةٍ نصيةٍ

# التحكم

اقرأ " ادخل رقمين " ، # أ ، # ب

إذا أ < ب :

اطبع أ ، " هو الأكبر "

وإلا :

اطبع ب ، " هو الأكبر "

ثم

- لاحظ أنه في أوامر إذا، وإلا، الألف تحتها همزة!  
(مفتاح shift + غ).

# التحكم

العمليات المنطقية في لغة كلمات دائماً تعود بالقيمة  
صحيح أو خطأ

- الروابط المنطقية هي التالي
- أ وأيضا ب (يمكن استخدام أكثر من معامِل أي أ وأيضا ب وأيضا ج ...الخ)
- أ أو ب ، أ أو ب أو ج
- ليس أ
- ليس أ ولا ب، ليس أ ولا ب ولا ج
- Evaluation is short-circuited, like e.g C
- بالمناسبة، علامة لا يساوي في كلمات هي <>

# التحكم

اقرأ " ادخل قيمتين: " ، # أ ، # ب

إذا أ < ب:

اطبع أ ، " هو الأكبر "

وإلا إذا أ > ب:

اطبع ب ، " هو الأكبر "

وإلا:

اطبع " إنهما متساويان ! "

ثم

- لا يوجد ما يوازي switch في كلمات، مثلها في ذلك لغة Python

# التحكم

س = 1

كرر مادام س  $\Rightarrow$  10 :

اطبع س

س = س + 1

تابع

كرر :

اطبع " هذه حلقة لانها ئية "

تابع

- لا يوجد حالياً ما يوازي do/while لكن هذه مشكلة ننوي حلها (:)

# التحكم

لكل أ من 1 إلى 10 :

اطبع أ

تابع

- في هذا المثال الحلقات تصاعدية، والزيادة دائماً بواحد.

- أنواع أخرى من الحلقات:

لكل أ من 1 إلى 100 بخطوة 5 :

لكل أ من 10 نزولاً إلى 1:

لكل أ من 100 نزولاً إلى 1 بخطوة -5 :

# التحكم

علامة أ

اطبع "عاوز المصروف يا بابي"

اذهب إلى أ

- نعم، كلمات بها أمر اذهب إلى
- هذا لأهداف تعليمية، أنظر مثلاً تنظيم كتاب "تحقيق الذات في كتابة البرمجيات"
- المعلم، أو المبرمج، لديه الحرية في أن يستخدمها أو يكتفي بالطرق الهيكلية
- الحلقات في كلمات ليس فيها ما يوازي ، break ، continue - وهذا متعمد للتبسيط - فهنا قد يفيد اذهب إلى

# المصفوفات

س = مصفوفة ( 10 )

لكل أ من 1 إلى 10 :

اقرأ #س [ أ ]

تابع

- يمكن للمصفوفة أن تحتوي عناصر من أنواع مختلفة
- الترقيم يبدأ من الواحد
- المصفوفة هي reference type ، أي أن تخصيص مصفوفة س = ص يجعلهما يشيران لنفس المصفوفة ولا ينسخ س في مصفوفة جديدة

# المصفوفات

س = [ 5 ، 12 ، 38 ، 40 ]

ص = [ [ "ملك" ، "فريدة" ] ، [ "احمد" ] ،  
[ [ "سلمى" ] ] ]

- هذه تسمى مصفوفات حرفية array literals

# المصفوفات

س = مصفوفة .متعددة ( [ 4 ، 5 ، 2 ] )

س [ 4 ، 3 ، 1 ] = 100

- كلمات تدعم المصفوفات متعددة الأبعاد، مثلها مثل C# لكن ليس مثل (على سبيل المثال) Java

- هنا اصطلاح خاص: نحن نقصد معنى مختلف عن array of array، الذي هو موجود بالتأكيد في Java وغيرها.

في كلمات:

- س [ 2 ] [ 1 ] array of array

- س [ 2 ، 1 ] multi-dimensional array

# القواميس

السن = { "أحمد" <= 10 ، "مريم" <= 9 ، سلمى <= 3 }

اطبع السن ["أحمد"]

السن ["مريم"] = السن ["مريم"] + 1

- القواميس مبنية داخلياً على الفصيلة QMap وهي جزء من مكتبة QT، تستخدم الـ skip list
- {} تأتي بقاموس فارغ
- الأنواع المسموح بها كمفاتيح هي: الأعداد الصحيحة، النصوص، مصفوفات من الأنواع المسموح بها (المقارنة هنا بالقيمة وليس الـ reference).
- في المستقبل ننوي اضافة طريقة لدعم custom data types كقيم مفتاحية

# الإجراءات والدوال

إجراء قدم . التحية ( الشخص ) :  
اطبع "مرحبا يا " ، الشخص  
اطبع " أهلا يا " ، الشخص  
اطبع "welcome" ، الشخص

نهاية

قدم . التحية ("مجي")

قدم . التحية ("هشام")

- لاحظ أن كلمة إجراء بها همزة تحت الألف
- مثل اللغات شبيهة الC، لا بد من كتابة القوسين أثناء تعريف الإجراءات واستدعائه، حتى لو لم يكن هناك قائمة عوامل parameters.

# الدوال

دالة اسم .جد ( الاسم . الكامل ) :  
م = تفصيل ( الاسم . الكامل ، " " )  
ارجع ب: م [3]

نهاية

اطبع اسم .جد ( "محمد سامي علي" )

- العرف هو تسمية الدالة باسم القيمة المرجوعة لتكون قراءة البرنامج طبيعية، يعني س=منتصف(ص)، (ع) وليس س=احسب.المنتصف(ص، ع)
- لنفس السبب عادةً ما تكون العوامل معرفة لا نكرة

# ملاحظات على المتغيرات

- نطاق المتغير دائما محليّ local scope ما لم يعلن العكس
- حتى المتغيرات في البرنامج الرئيسي - خارج كل الإجراءات والدوال - هي خاصة بالبرنامج الرئيسي فقط
- لجعل النطاق global استخدم أمر مشترك هكذا  
س مشترك
- وهذا ينفع في البرنامج الرئيسي فقط

## ملاحظات على المتغيرات 2

- لا يمكنك استخدام المتغير قبل تعريفه، ولا يوجد type declarations؛ ما الذي يعرف المتغير إذن؟
  - تخصيص قيمة له
  - كونه عامل parameter لإجراء أو دالة
  - تعريفه بكلمة **مشترك**
  - كونه عداد counter في أمر **لكل/تابع**

# بعض الدوال الجاهزة

## • حسابية

جا ، جتا ، ظا ، معكوس.جا ، معكوس.جتا ، معكوس.ظا  
جذر، لو ، لو.ه (لو = log ، لو.ه = ln) ، تقريب(عدد)،  
باقي.قسمة(عدد1، عدد2)

## • نصية

طول(نص)، أول(نص، عدد)، آخر(نص، عدد)، وسط(نص، بداية، طول)،  
يبدأ(نص1، نص2)، ينتهي(نص1، نص2)، يحتوي(نص1، نص2)،  
تفصيل(النص، الفاصل)، تقليم(النص)

## • مصفوفات

مصفوفة(الطول)، مصفوفة.متعددة(الأبعاد)، طول(مصفوفة)

## • عام

عشوائي(عدد) تعيد عدداً صحيحاً يبدأ من صفر وأقل من القيمة  
المقدمة

• قائمة كاملة في <http://www.kalimat-lang.com/wiki/builtins>

# بعض الدولب والإجراءات الجاهزة

- **دوال حسابية 2**

تقريب(عدد)، باقي.قسمة(عدد1،عدد2)

- **دوال نصية 2**

رقم(نص)، حرف(نص)

- **دوال تحويلية**

كعدد(نص)، كنص(عدد)

- **دوال متعلقة بأنواع البيانات**

نوع(قيمة)

- **إجراءات تعامل مع الشاشة والبرنامج**

انتظر(ملليثانية)، امسح.الشاشة( )، امسح.الكتابة( )،  
حدد.مكان.المؤشر(صف، عمود)

- **الثابت(نص) :** هذه الدالة تأخذ نصاً وتعود بثابت في نظام كلمات نفسه.

حالياً لو أخذت النص "سطر.جديد" تعود ب \n

# Tail call elimination

دالة مضروب (أ، ن) :

إذا  $n = 0$  :

ارجع ب أ

وإلا :

وكل إلى مضروب (أ×ن، ن-1)

نهاية

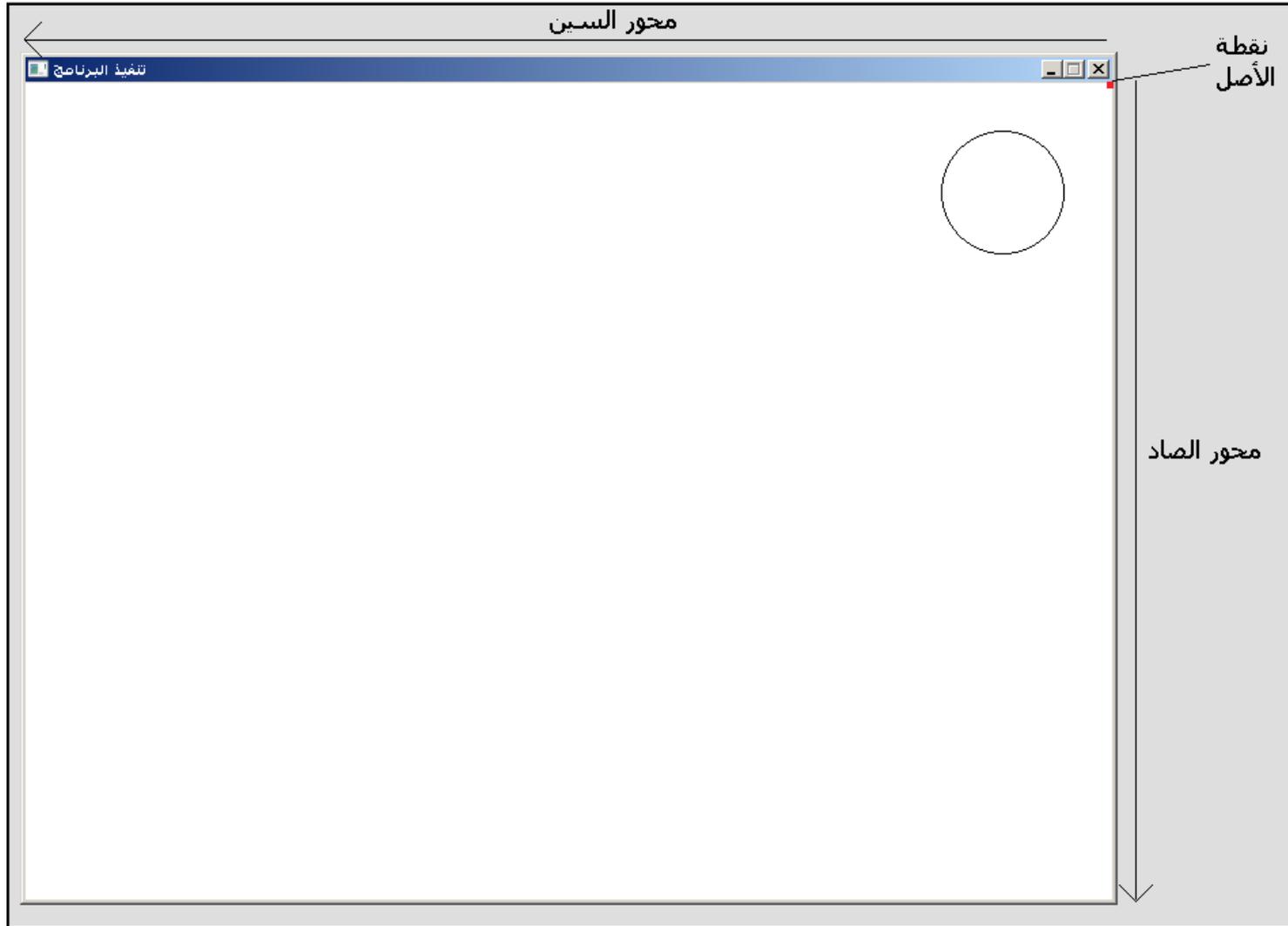
اطبع مضروب (1، 20)

- وكل إلى تنفع مع الإجراءات والدوال وليس الدوال فقط

- Can your C++, Java, C# or Python do this?

Haha

# نظام الإحداثيات



- الدقة 600×800
- بدء العد من الصفر

# أوامر الرسم

ارسم .نقطة (س، ص) ، اللون

ارسم .خط (س1 ، ص1) - (س2، ص2) ، اللون

ارسم .مستطيل (س1، ص1) - (س2، ص2) ، اللون ، ملء

ارسم .دائرة (س.مركز ، ص.مركز) ، اللون ، ملء

• اللون قيمة من 0 إلى 15

• ملء هو قيمة منطقية (صحيح/خطأ)

• اللون والملء دائما اختياريين

• في حالة إدخال الملء وعدم إدخال اللون اترك فاصلة، مثلاً  
ارسم.دائرة (40،50) ، 10 ، صحيح

# الأطيفاف sprites

- صورة متحركة على الشاشة تستخدم عادةً في الألعاب
- لا تدمر خلفية الشاشة تحتها
- خلفية الطيف شفافة
- لو رسمته في مكان أ ثم رسمته في مكان ب يختفي تلقائياً من أ.
- مأخوذ (مع سائر أوامر الرسم) من مكتبة g4c المأخوذة بدورها من كمبيوتر صخر
- كلمات، في تمنينا، هي الوريث الفكري لصخر!

# الأطياتف

ط = حمل .طيف ("abc .bmp")

ارسم .طيف ط في (12، 13)

اخف .طيف (ط)

اظهر .طيف (ط)

- لا بد أن يكون ملف الصورة في نفس المكان المخزن فيه البرنامج
- بالتبعية، لا بد من تخزين البرنامج الذي يستخدم أطياتاً قبل تنفيذه
- كلمات ستعطيك رسالة خطأ لو لم تفعل ذلك

# الأطياف

- دوال وإجراءات متعلقة بالأطياف:

## إجراءات

اظهر.طيف(الطيف)، اخف.طيف(الطيف)

## دوال

يمين.الطيف(ط)، يسار.الطيف(ط)،

عرض.الطيف(ط)، ارتفاع.الطيف(ط)،

قمة.الطيف(ط)، قاع.الطيف(ط)

- ربما كان الأفضل عمل هذه الدوال بطريقة Object oriented، لكن فضلنا الطريقة العادية لعدم إجبار المعلم على تدريس ال OOP قبل الألعاب...

# الكائنات والفصائل

- Kalimat shares many features with its sister dynamic languages like e.g Python:
  - Dynamic typing/duck typing
  - Object reference model, all classes are reference types
  - Classes are also objects (first-class values)

# تعريف فصيلة جديدة

فصيلة شخص:

له اسم ، سن

له عنوان

نهاية

م = شخص جديد

اسم م = "تامر"

سن م = 12

عنوان م = "شارع الورد"

- لاحظ طريقة الوصول للبيانات: بيان الكائن مثل obj.field في اللغات الأخرى

# الفصائل

- هذا يجعل قراءة البرامج سلسلة جداً:  
اسم أخ خال مدير م = "ميشو"
- لكنه يؤدي إلى التباس أحياناً، مثلاً لو كان التعبير بين قوسين، سيعتبر المترجم هذا استدعاء دالة وليس قراءة بيان:  
اطبع اسم (س)
- في تلك الحالات الخاصة يمكن استخدام رمز \$  
اطبع اسم \$ (س)

# Methods

- كما نعرف، تنقسم الـ functions في كلمات إلى **إجراءات** لا تعود بقيمة و**دوال** تعود بقيمة
- نفس النظام بالنسبة للـ member functions، هي إما **استجابة** لا تعود بقيمة أو **رد** يعود بقيمة. مثلاً:

ق : اصف(12)

- هنا ارسلنا إلى ق رسالة اصف(12) واستجاب لها

اطبع ق : عدده()

هنا ارسلنا إلى ق رسالة عدده() ورد عليها بقيمة

# Methods

- في لغات البرمجة المعروفة، الكائن متلقي الرسالة يكون له اسم ثابت مثل **this**
- ويجوز عدم ذكر ذلك الاسم في حالات كثيرة، بحيث نقول `x` بدلاً من `this.x`
- لكننا نرى أن هذا له مشاكل تعليمية؛ وقت كثير يضيع في معنى كلمة `this` أو في شرح من أين جاءت `x`، بينما المفاهيم نفسها ليست صعبة
- لذلك في لغة كلمات لا بد دائماً من تقديم اسم للكائن المرسل له الرسالة عند تعريف `method`

# تعريف استجابة

فصيحة شخص:

له اسم ، سن

يستجيب ل : ا عرض ( )

نهاية

استجابة شخص ع ل : ا عرض ( ) :

اطبع " الاسم هو " ، اسم ع ، " والسن

هو " ، سن ع

نهاية

# تعريف رد

فصيحة شخص:

له اسم ، سن

يرد على اكبر . من ( أ )

نهاية

رد شخص ش على اكبر . من ( الآخر ) :

ارجع ب: سن ش < سن الآخر

نهاية

# الوراثة، تعدد الصور، الربط الحيّ

فصيلة مركبة:

له وزن، سرعة

نهاية

فصيلة سيارة:

مبني على مركبة

له رقم.شاسيه

نهاية

- تعدد الصور يحدث اوتوماتيكياً بسبب duck typing
- نفس الشيء بالنسبة للربط الديناميكي؛ كأن كل الاستجابات virtual
- لعمل شيء مثل abstract classes عرف استجابة ولا تكتب تفاصيلها
- يمكن استخدام الأمر **ناد.الاستجابة.السابقة** في تعريف استجابة بالفصيلة الوارثة للاستفادة من الاستجابة الموروثة، مثل super في لغة Java

# الحيوانات

فصيلة حيوان:

له وزن

يستجيب ل: صوت ( )

يستجيب ل: عدد. الأقدام ( )

نهاية

فصيلة قط:

مبني على حيوان

نهاية

فصيلة كلب:

مبني على حيوان

نهاية

# الحيوانات

استجابة حيوان ح ل: عدد.الأقدام():

اطبع ٤

نهاية

• هكذا عدد.الأقدام سيرثها القط والكلب

استجابة قط ق ل: صوت():

اطبع "ناو"

نهاية

استجابة كلب ك ل: صوت():

اطبع "هاو"

نهاية

• لكن كل منهما له صوت مختلف

# الحيوانات

تونة = قط جديد

عضمة = كلب جديد

تونة: صوت ()

تونة: عدد. الأقدام ()

عضمة: صوت ()

عضمة: عدد. الأقدام ()

- للأسف كلمات ليس فيها حالياً ما يوازي ال constructors
- لحل هذه المشكلة حالياً يمكن عمل دالة صنع. كذا/ () تجهّز الكائن المطلوب
- اضافتها سهلة، لكني لا أريدها مثل اللغات المعروفة؛ ما زلت أفكر في طريقة أفضل...
- التعبير المنطقي **هو ب** يرجع صحيح إذا كان أ من الفصيلة ب أو فصيلة أعلى منها، مثل instanceof في Java. يعمل أيضاً مع الأنواع مثل integer، string

# التعامل مع الحوادث event handling

هذه العبارة تصلح في أي مكان في البرنامج:  
**عند حادثة** <اسم الحادثة> **نفذ** <اسم إجراء>

• أسماء الحوادث المعروفة:

- ضغط.زر.ماوس
- رفع.زر.ماوس
- تحريك.ماوس
- ضغط.مفتاح
- رفع.مفتاح
- ادخال.حرف
- تصادم (أي تصادم طيفين)

# التعامل مع الحوادث

- إجراء التعامل مع الماوس يأخذ دائماً قيمتيّ س، ص
- إجراءات لوحة المفاتيح تأخذ عدد ونص يمثلان (أ) رقم كودي للزر (مستقل عن اللغة، يعني كود ح هو نفسه كود p) والأكواد أيضاً للمفاتيح الخاصة مثل الأسهم. (ب) نص يحوي الحرف الذي يعبر عنه الزر، وهذا يعتمد على اللغة
- يمكن كتابة برنامج قصير يدلك على الكود الذي تريده
- إجراءات الأظياف تأخذ الطيفين الذان اصطدا

# لوحة المفاتيح العربية

× = خ + shift •

÷ = هـ + shift •

] = ب + shift •

[ = ي + shift •

< = ر + shift •

> = ج + shift •

} = ز + shift •

{ = وء + shift •

# البرمجة المتوازية

## شغل <استدعاء إجراء>

- تقوم بتشغيل الإجراء على التوازي مع باقي البرنامج.  
starts the procedure asynchronously
- حالياً ليس توازياً حقيقياً لكن ننوي جعله كذلك
- صدق أو لا تصدق، هذا يجعل البرمجة أسهل وقد وضعناه لأسباب تعليمية:
- تسهيل ال GUI programming
- تسهيل البرامج ذات الرسوم المتحركة
- لأننا داخلون عصر ال parallel processing

# القنوات

- A method of synchronization and data sharing between processes
- Based on the CSP model (concurrent sequential processes), used e.g in Google's Go language
- Based on message-passing instead of shared state
- We use it also for GUI event handling

# القنوات

إنشاء قناة

ق = قناة ( )

الإرسال

ارسل 12 إلى ق

الاستقبال

تسلم س من ق

- الإرسال والاستقبال دائماً متزامنان synchronous، أي أن المرسل ينتظر حتى يتسلم أحد رسالته، والمستقبل ينتظر حتى تأتي الرسالة
- لو لم يهم قيمة الرسالة وكان فقط يهم عملية إرسالها، يمكن استخدام الأوامر

ارسل إشارة إلى ق

تسلم إشارة من ق

# القنوات

تخير :

ارسل 12 إلى ق1:

اطبع "لقد ارسلت إلى ق1"

أو تسلم س من ق2:

اطبع "لقد تسلمت من ق2"

أو ارسل "مرحبا" إلى ق3:

اطبع "لقد ارسلت إلى ق3"

تم

- أول عملية اتصال تصلح من هؤلاء هي التي ستتم (ويتم تنفيذ الكود المرتبطة بها)

- لو كان أكثر من قناة جاهزة في نفس اللحظة سيتم الاختيار من بينهم عشوائياً

# GUI

ن = النافذة . الحالية ( )

ز = زر جديد

ز : حدد . المكان (30 ، 30)

ز : حدد . الحجم (30 ، 70)

ز : حدد . النص ( " اضغط هنا " )

ن : اصف (ز)

كرر :

تسلم إشارة من ضغط زر 1

اطبع " شاطر "

تابع

# GUI

• ماذا لو أردنا التعامل مع أكثر من زر؟

كرر :

تخير :

تسلم إشارة من ضغط زر 1 :

....افعل كذا وكذا

أو تسلم إشارة من ضغط زر 2 :

....افعل كذا وكذا

ثم

تابع

# GUI

- لقد غيرنا طريقة التعامل مع الـ GUI
- ماذا لو كان ينبغي على البرنامج أن يضغط زر، ثم يختار شيئاً، ثم يضغط زر آخر؟
  - يكفي ان نقول **تسلم/من** ... بالترتيب المطلوب!
- ماذا لو أردنا أن يضغط المستخدم على زر ثلاث مرات؟
  - نضع **تسلم/من** في حلقة تكرارية!
- يمكننا أن نشغل كذا إجراء على التوازي، كل منهم يتعامل مع جزء مستقل من الـ GUI
- صار التفكير في الـ GUI مثل التفكير في الـ console

# التحفة الندية في فصائل الواجرات الرسومية

## • نافذة

الاستجابات: كبر ( ) ، تحرك إلى (س، ص)،  
اضف (كائن)، حدد الحجم (عرض، ارتفاع)،  
حدد العنوان (نص)

## • زر

الاستجابات: حدد المكان (س، ص)،  
حدد الحجم (عرض، ارتفاع)، حدد النص (نص)  
الردود: نصه ( )  
القنوات: ضغط

# التحفة الندية في فصائل الواجرات الرسومية

## • صندوق نصي

الاستجابات: حدد المكان (س، ص)،  
حدد الحجم (عرض، ارتفاع)، حدد النص (نص)،  
الحق نص (نص)

الردود: نصه ( )

القنوات: تغير

## • سطر نصي

- تماماً مثل صندوق نصي لكن يقبل سطرًا واحدًا.

# التحفة الندية في فصائل الواجرات الرسومية

## • صندوق.سرد listbox

الاستجابات: حدد.المكان(س، ص)،  
حدد.الحجم(عرض، ارتفاع)، **اضف**(كائن)،  
**اضف.في**(كائن، عدد)

الردود: **عنصر.رقم**(عدد)

القنوات: **تغير.اختيار**

# التحفة الندية في فصائل الواجرات الرسومية

- صندوق. مركب combo box
- الاستجابات: حدد. المكان (س، ص)،  
حدد. الحجم (عرض، ارتفاع)، **حدد. النص** (نص)،  
**اضف** (كائن)، **اضف. في** (كائن، عدد)،  
**حدد. أبحرر** (منطقي)
- الردود: **عنصر. رقم** (عدد)، **نصه** ( )
- القنوات: **تغير. اختيار**، **تغير. نص**

# التحفة الندية في فصائل الواجهات الرسومية

## • علامة نصية label

- الاستجابات: حدد.المكان (س، ص)، حدد.الحجم (عرض، ارتفاع)، حدد.النص (نص)
- الردود: نصه ( )

## • صندوق استبيان check box

- الاستجابات: حدد.المكان (س، ص)، حدد.الحجم (عرض، ارتفاع)، حدد.النص (نص)، **حدد.القيمة** (عدد)
- الردود: نصه ( )، **قيمته** ( )
- القنوات: تغير.قيمة
- ملاحظات: القيمة 0 = غير مختار، 1 = اختيار جزئي، 2 = مختار

# التحفة الندية في فصائل الواجرات الرسومية

- صندوق. اختيار option button
- الاستجابات: حدد. المكان (س، ص)،  
حدد. الحجم (عرض، ارتفاع)، حدد. النص (نص)،  
**حدد. القيمة** (منطقي)
- الردود: نصه ( )، **قيمه** ( )
- القنوات: اختيار

# التحفة الندية في فصائل الواجرات الرسومية

## • مجموعة.اختيارات option group

- الردود: اضف (صندوق.اختيار)، الزر.الموسوم(الوسم)
- ملاحظات: تستخدم الفصيلة مجموعة.اختيارات في جميع صناديق الاختيار بحيث تكون في مجموعات مستقلة. اضف الاختيارات كلها للمجموعة ثم اضف المجموعة للنافذة.
- ماذا لو أردت التعامل مع صندوق اختيار معين؟ القيمة الراجعة من **اضف(ص)** هي عدد صحيح اسمه الوسم يدل على الصندوق الذي تم اضافته، ويمكن استرجاع الكائن الذي يعبر عن ذلك الصندوق عن طريق الدالة **الزر.الموسوم(الوسم)**

# ما الذي لم نقله؟

- القراءة والكتابة في الملفات  
<http://www.kalimat-lang.com/wiki/Byexample#Files>
- الوحدات

test.k

باستخدام "welcome.k"  
رحب ("سمعان")

welcome.k

وحدة الترحيب  
إجراء رحب (شخص):  
اطبع "مرحباً يا "، شخص  
نهاية

تمت

حمان صغير  
حسنا يا أمينة إليك بحمان صغير

<http://www.kalimat-lang.com>  
<http://code.google.com/p/kalimat>  
<http://iamsamy.blogspot.com>