



## جامعة طرابلس - كلية تقنية المعلومات



مقدمة في هندسة البرمجيات

Introduction to software Engineering

**ITGS-213**

المحاضرة السابعة - أدوات البرمجة  
Programming Tools

خريف 2020



## مواضيع المحاضرة



Programming Tools

Utility Tools

إرشادات في كتابة شفرة الجيدة

هندسة البرمجيات بمساعدة الحاسوب

**Aided Software Engineering (CASE)**



## البرمجة

□ البرمجة Programming : هي عبارة عن كتابة مجموعة من الأوامر (مجموعة قواعد أو إرشادات) أو ما يُعرَف بالشفيرة Code أو الكود، وهذه الأوامر هي التي تُشكِّل ما يُعرَف بالبرنامج Program

□ لغات البرمجة Programming language هي مجموعة أوامر مكتوبة على شكل رموز تستند إلى قواعد معينة يفهمها جهاز الحاسوب ويقوم بتنفيذها، وتُمرُّ لغات البرمجة بمجموعة من الخطوات والمراحل قبل أن يتم تنفيذها.



## أنواع طرق البرمجة

### (1) البرمجة الهيكلية Structured Programming

- تعتمد البرمجة الهيكلية على تجزئة البرنامج إلى عدة برامج جزئية أو فرعية حيث يتم الربط بين هذه البرامج الفرعية لتشكيل البرنامج العام وتظهر فاعليته في حالة المسائل متوسطة الحجم كما يسهل اكتشاف الأخطاء بهذا الأسلوب..
- يطلق عليها أيضا تسمية البرمجة الاجرائية **Procedural Programming** وذلك بتجزئة البرنامج الى وحدات تسمى اجراءات لتسهيل القراءة واعادة الاستخدام تسمى هذه الاجزاء بـ **Procedures - functions** :أسماء.



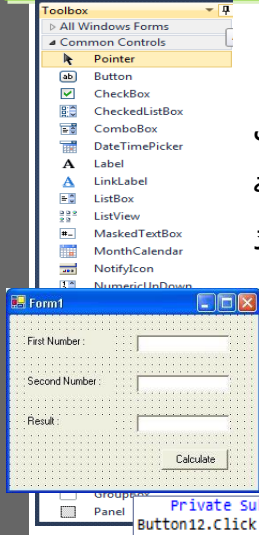
## أنواع طرق البرمجة

### (2) البرمجة بالحدث والمرئية Visual Programming & Events

ويسمى هذا النوع بالبرمجة بالحدث والمرئية لانه يتم تنفيذ مجموعة من الافعال التي نقوم بتعيينها نتيجة لحدث قام المستخدم بإجراؤه على احد عناصر واجهة البرنامج مثل Button command و سيطرة أكثر للمستخدم على تسلسل تنفيذ العمليات.

يتم تنفيذ البرنامج بحسب رغبة المستخدم.(Event-driven languages)

يتمتع هذا النوع من لغة البرمجة بوجود Tools Box الذي يمكن المبرمج من اعداد واجهة رسومية للمستخدم.



```
Private Sub Button12_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button12.Click
```

## أنواع طرق البرمجة

### (3) البرمجة الشيئية Object-Oriented Programming

وتسمى أيضا البرمجة الكائنية أو غرضية التوجيه. وهي تعتبر أن الاجسام من حولنا هي كائنات تتفاعل مع بعضها بغض النظر أكانت حية أو جامدة.

يتم فيها تقسيم البرنامج إلى وحدات تسمى الكائنات Objects ، كل كائن عبارة عن حزمة من البيانات والمتغيرات والثوابت والدوال. وهي كائنات قابلة للاستخدام. Reuse

ويتم بناء البرنامج بواسطة استخدام الكائنات وربطها مع بعضها البعض.

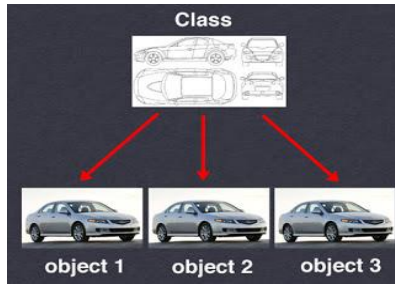
أصبحت لغة شائعة في التسعينات ولا زالت .

هذا المفهوم نتاج طبيعي لما هو موجود في حياتنا اليومية.

اسم الكائن
بيانات (خصائص)
طرق

## البرمجة الشيئية Object-Oriented Programming

□ فمثلاً إذا كان عندنا كائن سيارة فإن للسيارة متغيرات خاصة بها ودوال خاصة بها ، مثلاً من متغيرات السيارة، كمية الوقود المتوفرة، وسرعة السيارة الحالية، ودرجة حرارة المحرك، ومن دوال السيارة دالة لزيادة السرعة، ودالة المكابح، ودالة تشغيل أضواء السيارة، تستطيع بهذه الطريقة أن تفهم أجزاء البرنامج أكثر، ويصبح المبرمج منظماً أكثر.



## البرمجة الشيئية Object-Oriented Programming

- الكائن (**object**): رزمة برمجية تحتوي على البيانات والعمليات المنوطة بها وهي حالة خاصة للفصيلة وينتمي الكائن الى فصيلة معينة. **Classes**
- الطرق أو الطريقة (**Method**): هي عبارة عن خدمات يؤديها الكائن لتلبية المتطلبات الوظيفية التي تؤثر على البيانات داخل الكائن والتي تعد غالباً مخفية عن الكائنات الأخرى.
- الفصيلة : **Class** تصف فئة من الأشياء لها نفس الخصائص **attributes** والعمليات **operations** والعلاقات **relationships** هو حالة عامة من الكائن.
- فصيلة فرعية : **Subclass** عبارة عن عائلة متفرعة من الفصيلة الام وترث عنها البيانات والطرق.



## البرمجة الشيئية Object-Oriented Programming

- الخصائص: (**Attributes**) صفات الكائن أو الفصيلة.
- التصرف: (**behaviour**) هو عمل أو تصرف يقوم به الكائن عند تمرير رسالة (**Message**) أو اجابة رسالة (**Response**)
- **Message**: هي آلية التواصل بين الكائنات عبر الرسائل حيث تحمل بيانات أو تعليمات.
- **Encapsulation**: هو إخفاء المعلومات بين الفصائل ويتم تبادل البيانات عن طريق الرسائل بين الكائنات وإخفاء البيانات الداخلية.
- **Inheritance**: تعني الفصيلة الفرعية ترث كل الخصائص والطرق للفصيلة الام.



## مزايا البرمجة الشيئية

- خصائص البرمجيات المطورة باستخدام البرمجة الشيئية: **OOP**
    1. البرمجة بلغة طبيعية: (Natural language)  
استخدام مصطلحات طبيعية يفهمها المستخدم.
    2. موثوقية البرمجيات الجاهزة: (Reliability)  
برمجيات خالية من الاخطاء.
    3. اعادة الاستخدام: (Reuse)
- إذا تم اختبار ومصادقة اي جزء برمجي ، من الممكن اعادة استخدامه بكل ثقة.



## مزايا البرمجة الشيئية

4. سهولة الصيانة: (Maintainability)  
احد اهم اهداف البرمجة الشيئية التقليل من اعمال الصيانة الى الحد الادنى.
5. سهولة التمدد: (Extendable)  
السهولة في اضافة وظائف جديدة للمنظومة.
6. التقليل من مدة اعداد البرمجيات:  
وفرت التقنية الشيئية برمجيات جاهزة مما ادى الى تقليل زمن تطوير اي برمجيات الى اشهر او حتى اسابيع.



## لغات البرمجة Programming Languages

□ تعتبر لغات البرمجة من أهم أدوات البرمجة المستخدمة ومن امثلة لغات البرمجة:

**C**

هي لغة برمجة متعدّدة الاستخدامات، ظهرت في أوائل سبعينات القرن الماضي، وهي أقدم لغة برمجة والأكثر استخداماً.

**C++**

تُعدّ هذه اللغة تطويراً للغة C، وقد أُضيفت إليها خصائص جعلت منها لغةً كائنيّة التوجّه، وتُستخدَم هذه اللغة في تطوير البرمجيات المختلفة والألعاب

**Java**

هي لغة كائنيّة التوجّه



**Python**  
لغة البايثون من بين لغات البرمجة الأساسية والاكثر استخدام في السنوات الاخيرة



**JavaScript & PHP.**  
لغات تطبيقات الانترنت

**Prolog**  
احدى لغات الذكاء الاصطناعي

**SQL**  
هي لغة مختصة ببرمجة قواعد البيانات



## Top 10 In-Demand programming languages to learn in 2021

Rank	Language	Type	Score
1	Python	☉ 📱 📺	100.0
2	Java	☉ 📱 📺	95.4
3	C	📱 📺	94.7
4	C++	📱 📺	92.4
6	JavaScript	☉	88.1
6	C#	☉ 📱 📺	82.4
7	R	📱	81.7
8	Go	☉ 📱	77.7
9	HTML	☉	75.4
10	Swift	📱 📺	70.4

IEEE Spectrum

Rank	Language	Type	Score
1	Python	☉ 📱 📺	100.0
2	Java	☉ 📱 📺	92.4
3	C	📱 📺	91.6
4	JavaScript	☉	89.5
5	C++	📱 📺	87.3
6	Go	☉ 📱	79.9
7	Swift	📱 📺	73.4
8	HTML	☉	72.9
9	Dart	☉ 📱	71.6
10	Rust	☉ 📱 📺	70.3

Trending

Rank	Language	Type	Score
1	Python	☉ 📱 📺	100.0
2	C	📱 📺	96.0
3	Java	☉ 📱 📺	95.9
4	JavaScript	☉	89.6
6	C++	📱 📺	88.3
6	Go	☉ 📱	87.3
7	R	📱	85.7
8	HTML	☉	81.3
9	C#	☉ 📱 📺	79.8
10	SQL	📱	71.9

Jobs

Web



Enterprise



Mobile



Embedded



Source: IEEE Spectrum Interactive Ranking

## أدوات مساعدة Utility Tools

- هي مجموعة أدوات تساعد في اعداد وتعديل وتصحيح وادارة البرمجيات.
- المصححات: **Debugger** هي اداة تستخدم لاكتشاف الاخطاء في البرامج.
- **المحررات والمتصفحات: Editors/Browsers**

المحررات تساعد على التركيز على جزء من الشفرة المراد تعديلها، وتشمل

```

div-css.html [x]
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4 <html xmlns="http://www.w3.org/1999/xhtml" >
5   <head>
6     <title>CSS - div </title>
7     <style>
8       .div {
9         background-color: lightblue;
10        width: 200px;
11        padding: 25px;
12        border: 25px solid navy;
13        margin: 25px;
14      }

```

المحررات التالي:

1. تلوين النص
  2. القص واللصق بين النوافذ والصفحات
  3. إلغاء التعديل وإعادةه
- أما المتصفحات فتستخدم في لعرض الصفحات.

## أدوات مساعدة Utility Tools

- **مكونات الشفرة: Code Generators**

توفر وقت لكتابة شفرة معينة بحيث تكون جاهزة للاستعمال حيث تمكن المستخدم من تعريف الواجهة.

- **المكتبة: Libraries**

تحتوي على العديد من الدوال والاجراءات او الفصائل التي تساعد على اعداد

برمجيات ذات جودة في

```

Form1.vb [x] Form1 (Design)
Button1 Click
Public Class Form1
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
Button1.Text = "Click"
End Sub
End Class

```



## إرشادات في كتابة شفرة الجيدة

□ يعد الأسلوب الجيد أمرا مهم في أساليب البرمجة والتي تتضمن الشكل العام للشفرة المصدرية ومن هذه الإرشادات:

(1) المسافة البادئة: إن المسافات البادئة الصحيحة والمنطقية تجعل الشفرة أكثر قابلية للقراءة ولا تؤثر في الوظيفة.

<pre>if (hours &lt; 24 &amp;&amp; minutes &lt; 60 &amp;&amp; seconds &lt; 60) {     return true; } else {     return false; }</pre>	<pre>if ( hours&lt; 24 &amp;&amp; minutes&lt; 60 &amp;&amp; seconds&lt; 60) {return true ;} else {return false ;}</pre>
<pre>int i; for(i=0;i&lt;10;++i){     printf("%d",i*i+i); }</pre>	<pre>int i; for (i = 0; i &lt; 10; ++i) {     printf ("%d", i * i + i); }</pre>



## إرشادات في كتابة شفرة الجيدة

(2) التعليقات: التي تشير إلى بعض المسائل الخاصة.

<pre>int ix; // Index to scan array long sum; // Accumulator for sum</pre>	<pre>class MyClass { int foobar(int qux, // first parameter int quux); // second parameter int foobar2(int qux, // first parameter int quux, // second parameter int quuux); // third parameter };</pre>
--	--

(2) استعمال اقل ما يمكن من تراكيب التحكم مثل (If-Then-Else) حتي يصبح البرنامج سهل القراءة والفهم والصيانة.

(3) التنسيق: 

```
int n = Integer.parseInt(args[0]); // size of population
int trials = Integer.parseInt(args[1]); // number of trials
```

(4) التقليل من استعمال الحلقات المتداخلة ويجب إلا يزيد عمقها التداخل عن خمس تداخلات.

(5) عدم او قلة استخدام جملة Go To.

(6) يجب ألا يزيد البرنامج الفرعي عن 50 جملة برمجية.



## ادوات كيس CASE TOOLS

- ❑ تشبه أدوات كيس البرمجية برنامج AUTO-CAD الذي يستخدم لمساعدة المهندسين في رسم الخرائط المعمارية.
- ❑ لأداء اي منتج برمجي نحتاج الي أدوات مساعدة , تسمى هذه الادوات (CASE) بـ هندسة البرمجيات بمساعدة الحاسوب *Computer Aided Software Engineering* (**CASE**). اي اعداد المنظومات بمساعدة الحاسوب.
- ❑ ومن امثلة أدوات كيس التي تستخدم لرسم مخطط حالة الاستخدام ما يسمى برنامج VISIO من اعداد شركة ميكروسوفت.



19

## ادوات كيس *Computer Aided Software Engineering (CASE) TOOLS*

- ❑ هي برامج جاهزة على الحاسب تساعد محلي النظم ومهندسي البرمجيات في دعم وأتمتة الأعمال التي تتم خلال دورة حياة تطوير النظام **Software Development Life Cycle SDLC** .
- ❑ الغرض منها تسهيل عملية توحيد فلسفة التصميم داخل المؤسسة.



20

## الهدف من ادوات كيس Objectives of CASE TOOLS

1. زيادة سرعة التطوير والتصميم.
2. تسهيل وتحسين إجراءات الاختبار من خلال التدقيق الآلى.
3. يساعد على توحيد إجراءات تطوير النظم بناءا على منهجية.
4. تحسين عملية إدارة المشروع.
5. صيانة أسهل للبرامج.
6. تشجع على إعادة الاستخدام والاستفادة.
7. تساعد على زيادة الإنتاجية.



21

## أدوات الكيس CASE

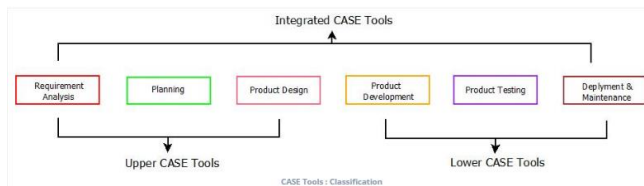
تنقسم ادوات CASE الى قسمين:

**الأدوات الكبيرة: Upper-CASE**

• وهي أدوات لدعم نشاطات العمليات المبكرة من المتطلبات و التصميم.

**الأدوات الصغيرة: Lower-CASE**

• وهي أدوات لدعم النشاطات التالية مثل البرمجة و تصحيح الأخطاء و الاختبار والتنفيذ (البرمجة) والصيانة و دعم أعمال التطوير المختلفة عبر المراحل المختلفة مثل التوثيق.



22

## مكونات أدوات كيس Components CASE TOOLS

- كانت أدوات الكيس في البداية في السبعينات عبارة عن معالج الكلمات يستعمل لتوثيق البرمجيات. ثم طورت واصبحت تشتمل على الآتي:
1. أدوات رسم المخططات. Diagramming tools.
  2. مكونات الشاشات ومولدات التقارير. Screen/report generators.
  3. مولد البرامج والتطبيقات. Code generators.
  4. مولد الوثائق. Documentation generators.
  5. أدوات الاختبار. Testing tools.



23

## مزايا أدوات كيس CASE TOOLS

1. تبسيط عملية إعداد وصيانة البرمجيات.
2. تسرع عملية إعداد البرمجيات.
3. تزيد من إنتاجية المبرمج ومحلل النظم.
4. توفر تواصل أفضل مع المستخدم بسبب استخدام الرسوم التي جانب النص.



24

## CASE tools examples أمثلة أدوات كيس

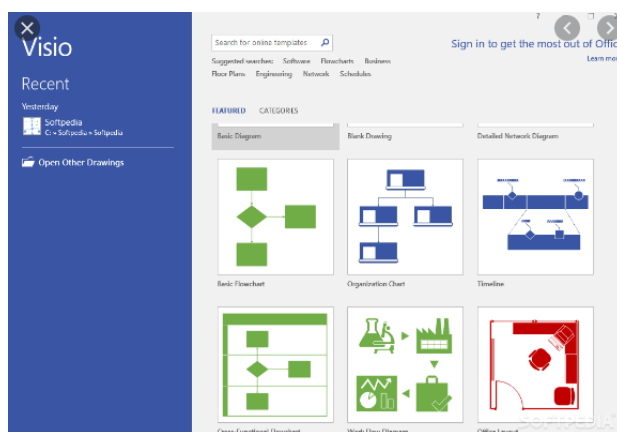
### □ CASE Tools examples:

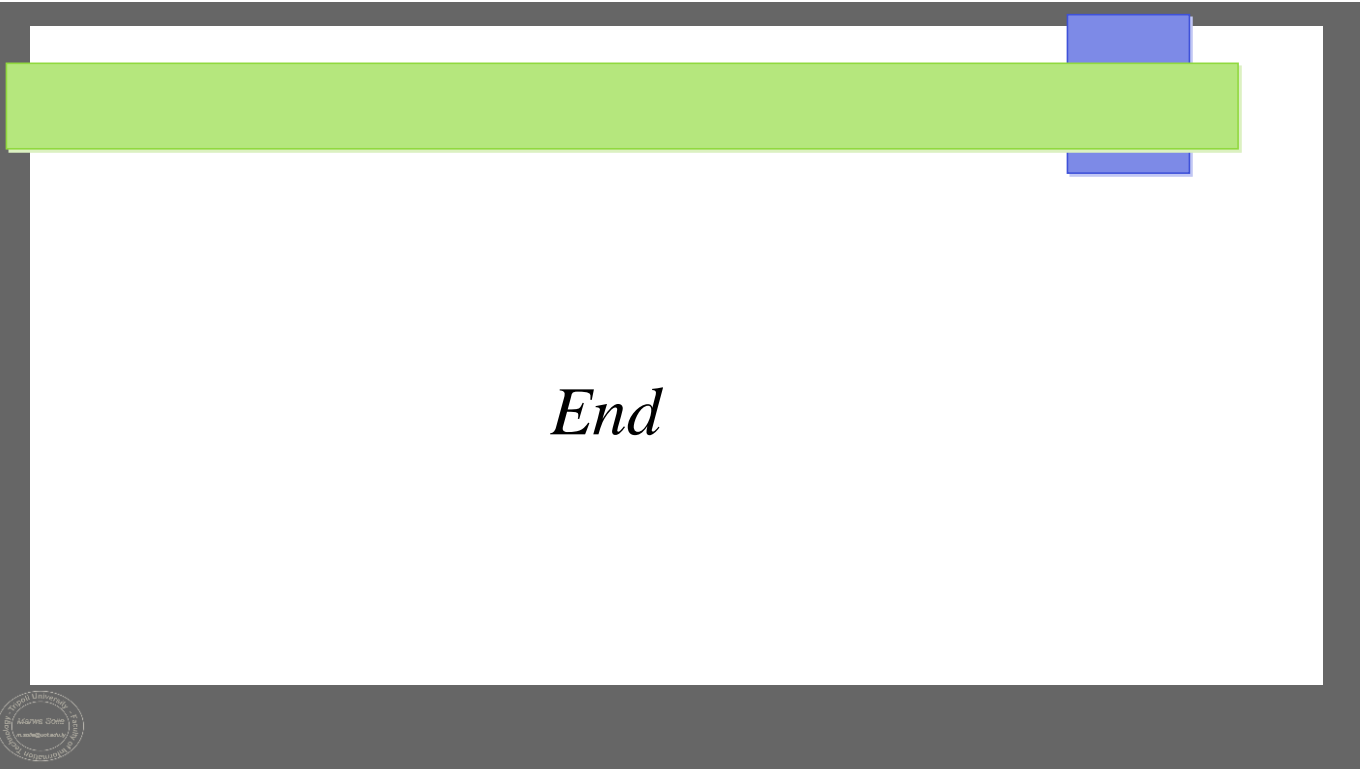
Accept 360,  
CaseComplete for requirement analysis.  
Dashboards, Project Scheduling  
Microsoft visio  
Dreamweaver,  
JMeter  
AppWatch



25

## Microsoft Visio 2019





*End*

