

M250 Overview

M250 Prsntn 2024J Overview

Phil Molyneux

13 October 2024

M250 Overview Tutorial

Agenda

- ▶ Introductions
- ▶ *Adobe Connect* — if you or I get cut off, wait till we reconnect (or send you an email)
- ▶ M250 overview
- ▶ Software resources & learning software packages
- ▶ Data types & variables
- ▶ Some useful Web & other references
- ▶ Time: about 1 hour
- ▶ Do ask questions or raise points.
- ▶ Slides/Notes [M250Tutorial20241013OverviewPrsntn2024J](#)

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Tutorial

Introductions — Phil

- ▶ *Name* Phil Molyneux
- ▶ *Background*
 - ▶ Undergraduate: Physics and Maths (Sussex)
 - ▶ Postgraduate: Physics (Sussex), Operational Research (Brunel), Computer Science (University College, London)
 - ▶ Worked in Operational Research, Business IT, Web technologies, Functional Programming
- ▶ *First programming languages* Fortran, BASIC, Pascal
- ▶ *Favourite Software*
 - ▶ Haskell — pure functional programming language
 - ▶ Text editors TextMate, Sublime Text — previously Emacs
 - ▶ Word processing in L^AT_EX — all these slides and notes
 - ▶ Mac OS X
- ▶ *Learning style* — I read the manual before using the software

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Tutorial

Introductions — You

- ▶ *Name ?*
- ▶ *Favourite software/Programming language ?*
- ▶ *Favourite **text editor** or **integrated development environment (IDE)***
- ▶ **List of text editors, Comparison of text editors and Comparison of integrated development environments**
- ▶ *Other OU courses ?*
- ▶ *Anything else ?*

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

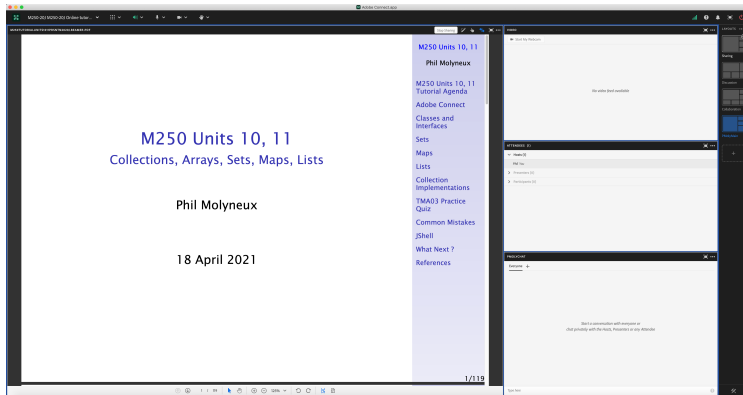
Software &
Programming

What Next ?

References

Adobe Connect

Interface — Host View



M250 Overview

Phil Molyneux

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented Programming Terminology

Using BlueJ

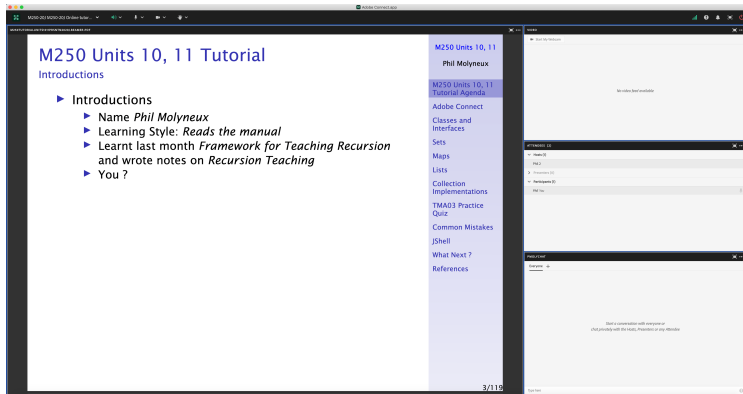
Software & Programming

What Next ?

References

Adobe Connect

Interface — Participant View



M250 Overview

Phil Molyneux

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented Programming Terminology

Using BlueJ

Software & Programming

What Next ?

References

Adobe Connect

Settings

- ▶ **Everybody** **Menu bar** **Meeting** **Speaker & Microphone Setup**
- ▶ **Menu bar** **Microphone** **Allow Participants to Use Microphone** ✓
- ▶ Check Participants see the entire slide **Workaround**
 - ▶ **Disable Draw** **Share pod** **Menu bar** **Draw icon**
 - ▶ **Fit Width** **Share pod** **Bottom bar** **Fit Width icon** ✓
- ▶ **Meeting** **Preferences** **General** **Host Cursor** **Show to all attendees**
- ▶ **Menu bar** **Video** **Enable Webcam for Participants** ✓
- ▶ Do not *Enable single speaker mode*
- ▶ Cancel hand tool
- ▶ Do not enable green pointer
- ▶ **Recording** **Meeting** **Record Session** ✓
- ▶ **Documents** Upload PDF with drag and drop to share pod
- ▶ Delete **Meeting** **Manage Meeting Information** **Uploaded Content**
and **check filename** **click on delete**

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented Programming Terminology

Using BlueJ

Software & Programming

What Next ?

References

► Tutor Access

TutorHome > M269 Website > Tutorials

Cluster Tutorials > M269 Online tutorial room

Tutor Groups > M269 Online tutor group room

Module-wide Tutorials > M269 Online module-wide room

► Attendance

TutorHome > Students > View your tutorial timetables

► Beamer Slide Scaling 440% (422 x 563 mm)

► Clear Everyone's Status

Attendee Pod > Menu > Clear Everyone's Status

► Grant Access and send link via email

Meeting > Manage Access & Entry > Invite Participants...

► Presenter Only Area

Meeting > Enable/Disable Presenter Only Area

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented

Programming

Terminology

Using BlueJ

Software &

Programming

What Next ?

References

Adobe Connect

Keystroke Shortcuts

- ▶ **Keyboard shortcuts in Adobe Connect**
- ▶ **Toggle Mic**  + **M** (Mac), **Ctrl** + **M** (Win) (On/Disconnect)
- ▶ **Toggle Raise-Hand status**  + **E**
- ▶ **Close dialog box**  (Mac), **Esc** (Win)
- ▶ **End meeting**  + ****

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented

Programming

Terminology

Using BlueJ

Software & Programming

What Next ?

References

Adobe Connect Interface

Sharing Screen & Applications

- ▶ **Share My Screen** > **Application tab** > **Terminal** for **Terminal**
- ▶ **Share menu** > **Change View** > **Zoom in** for mismatch of screen size/resolution (Participants)
- ▶ (Presenter) Change to 75% and back to 100% (solves participants with smaller screen image overlap)
- ▶ Leave the application on the original display
- ▶ Beware blue hatched rectangles — from other (hidden) windows or contextual menus
- ▶ Presenter screen pointer affects viewer display — beware of moving the pointer away from the application
- ▶ First time: **System Preferences** > **Security & Privacy** > **Privacy** > **Accessibility**

Agenda

Adobe Connect
Interface
Settings

Sharing Screen &
Applications

Ending a Meeting
Invite Attendees
Layouts
Chat Pods
Web Graphics
Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Adobe Connect

Ending a Meeting

- ▶ *Notes for the tutor only*
- ▶ **Student:** Meeting > Exit Adobe Connect
- ▶ **Tutor:**
- ▶ **Recording** Meeting > Stop Recording ✓
- ▶ **Remove Participants** Meeting > End Meeting... ✓
 - ▶ Dialog box allows for message with default message:
 - ▶ *The host has ended this meeting. Thank you for attending.*
- ▶ **Recording availability** *In course Web site for joining the room, click on the eye icon in the list of recordings under your recording* — edit description and name
- ▶ **Meeting Information** Meeting > Manage Meeting Information — can access a range of information in Web page.
- ▶ **Delete File Upload** Meeting > Manage Meeting Information > Uploaded Content tab select file(s) and click Delete
- ▶ **Attendance Report** see course Web site for joining room

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented Programming Terminology

Using BlueJ

Software & Programming

What Next ?

References

Adobe Connect

Invite Attendees

- ▶ **Provide Meeting URL** Menu Meeting Manage Access & Entry
Invite Participants...
- ▶ **Allow Access without Dialog** Menu Meeting
Manage Meeting Information provides new browser window with *Meeting Information* Tab bar Edit Information
- ▶ Check *Anyone who has the URL for the meeting can enter the room*
- ▶ Default *Only registered users and accepted guests may enter the room*
- ▶ **Reverts to default next session but URL is fixed**
- ▶ Guests have blue icon top, registered participants have yellow icon top — same icon if URL is open
- ▶ See [Start, attend, and manage Adobe Connect meetings and sessions](#)

Agenda

Adobe Connect

Interface

Settings

Sharing Screen & Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented Programming Terminology

Using BlueJ

Software & Programming

What Next ?

References

Adobe Connect

Entering a Room as a Guest (1)

- ▶ Click on the link sent in email from the Host
- ▶ Get the following on a Web page
- ▶ As *Guest* enter your name and click on **Enter Room**



Adobe Connect

M269-21J Online tutorial room
London/SE (1,13) CG [2311] (M269-21J)
(1)

Guest Registered User

Name

Guest Name

By entering a Name & clicking "Enter Room", you agree that you have read and accept the [Terms of Use](#) & [Privacy Policy](#).

Enter Room

M250 Overview

Phil Molyneux

Agenda

Adobe Connect

Interface

Settings

Sharing Screen &
Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Adobe Connect

Entering a Room as a Guest (2)

- ▶ See the *Waiting for Entry Access* for *Host* to give permission



Adobe Connect

Waiting for Entry Access

This is a private meeting. Your request to enter has been sent to the host. Please wait for a response.

[M250 Overview](#)

Phil Molyneux

Agenda

Adobe Connect

Interface

Settings

Sharing Screen &
Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

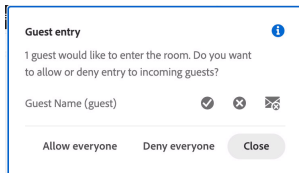
What Next ?

References

Adobe Connect

Entering a Room as a Guest (3)

- *Host* sees the following dialog in *Adobe Connect* and grants access



Agenda

Adobe Connect

[Interface](#)[Settings](#)[Sharing Screen & Applications](#)[Ending a Meeting](#)[Invite Attendees](#)[Layouts](#)[Chat Pods](#)[Web Graphics](#)[Recordings](#)

M250 Overview

[Object-oriented Programming Terminology](#)[Using BlueJ](#)[Software & Programming](#)[What Next ?](#)[References](#)

- ▶ **Creating new layouts** example *Sharing* layout
 - ▶ **Menu** ▶ **Layouts** ▶ **Create New Layout...** ▶ **Create a New Layout dialog**
▶ **Create a new blank layout** and name it *PMolyMain*
- ▶ New layout has no Pods but does have Layouts Bar open (see Layouts menu)
- ▶ **Pods**
 - ▶ **Menu** ▶ **Pods** ▶ **Share** ▶ **Add New Share** and resize/position — initial name is *Share n* — rename *PMolyShare*
 - ▶ **Rename Pod** **Menu** ▶ **Pods** ▶ **Manage Pods...** ▶ **Manage Pods**
▶ **Select** ▶ **Rename** or **Double-click & rename**
- ▶ Add Video pod and resize/reposition
- ▶ Add Attendance pod and resize/reposition
- ▶ Add Chat pod — rename it *PMolyChat* — and resize/reposition

Agenda

Adobe Connect

[Interface](#)[Settings](#)[Sharing Screen & Applications](#)[Ending a Meeting](#)[Invite Attendees](#)

Layouts

[Chat Pods](#)[Web Graphics](#)[Recordings](#)

M250 Overview

[Object-oriented Programming Terminology](#)[Using BlueJ](#)[Software & Programming](#)[What Next ?](#)[References](#)

- ▶ Dimensions of **Sharing** layout (on 27-inch iMac)
 - ▶ Width of Video, Attendees, Chat column 14 cm
 - ▶ Height of Video pod 9 cm
 - ▶ Height of Attendees pod 12 cm
 - ▶ Height of Chat pod 8 cm
- ▶ **Duplicating Layouts** does *not* give new instances of the Pods and is probably not a good idea (apart from local use to avoid delay in reloading Pods)
- ▶ **Auxiliary Layouts** name *PMolyAuxOn*
 - ▶ Create new Share pod
 - ▶ Use existing Chat pod
 - ▶ Use same Video and Attendance pods

Adobe Connect

Chat Pods

- ▶ **Format Chat text**

- ▶  menu icon 

- ▶ Choices: Red, Orange, Green, Brown, Purple, Pink, Blue, Black

- ▶ Note: Color reverts to Black if you switch layouts

- ▶  menu icon 

Agenda

Adobe Connect

Interface

Settings

Sharing Screen &
Applications

Ending a Meeting

Invite Attendees

Layouts

Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Graphics Conversion

PDF to PNG/JPG

- ▶ Conversion of the screen snaps for the installation of Anaconda on 1 May 2020
- ▶ Using GraphicConverter 1.1
- ▶ 
- ▶ Select files to convert and destination folder
- ▶ Click on  or 

Agenda

Adobe Connect

Interface
Settings
Sharing Screen & Applications
Ending a Meeting
Invite Attendees
Layouts
Chat Pods

Web Graphics

Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Adobe Connect Recordings

Exporting Recordings

- ▶ *Menu bar* > *Meeting* > *Preferences* > *Video*
- ▶ *Aspect ratio* > *Standard (4:3)* (not Wide screen (16:9) default)
- ▶ *Video quality* > *Full HD* (1080p not High default 480p)
- ▶ **Recording** *Menu bar* > *Meeting* > *Record Session* ✓
- ▶ **Export Recording**
- ▶ *Menu bar* > *Meeting* > *Manage Meeting Information*
- ▶ *New window* > *Recordings* > *check Tutorial* > *Access Type button*
- ▶ *check Public* > *check Allow viewers to download*
- ▶ **Download Recording**
- ▶ *New window* > *Recordings* > *check Tutorial* > *Actions* > *Download File*

Agenda

Adobe Connect

Interface
Settings
Sharing Screen & Applications
Ending a Meeting
Invite Attendees
Layouts
Chat Pods
Web Graphics
Recordings

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Overview

Module Aims and Structure

- ▶ **Object-oriented programming**
- ▶ Encapsulation enforces modularity — implementations of objects not available outside some *container*
- ▶ Inheritance encourages reuse of code and organises complexity
- ▶ Polymorphism enables a structured approach to different objects responding differently to the same messages
- ▶ **Java** is a class-based, object-oriented general-purpose programming language with strong type system
- ▶ **Type system** addresses the problem of ensuring that programs have meaning — we shall look at this later but consider
 - $1 + 2$ and $1 + '2'$ are they valid ?
- ▶ We are using Java to illustrate an object-oriented approach to programming
- ▶ We will not have complete coverage of every feature of Java

M250 Overview

Expression Quiz

- Evaluate `1 + 2`

```
jshell> 1 + 2  
$1 ==> 3
```

- Evaluate `1 + '2'`

```
jshell> 1 + '2'  
$2 ==> 51
```

- The evaluations are using **JShell** a Read-Eval-Print Loop (REPL) tool for exploring Java
- What is going on here? A character is a *numeric* type

```
jshell> (int) '2'  
$3 ==> 50  
jshell> (char) 50  
$4 ==> '2'
```

- The above are explicit *type conversions*

M250 Overview

Expression Quiz (2)

- Beware mixing characters and strings

```
jshell> '1' + '2'  
$5 ==> 99
```

```
jshell> "1" + "2"  
$6 ==> "12"
```

- The former does implicit type casting to `int`
- The latter is string concatenation
- The `(+)` operator is *overloaded*

M250 Overview

Chps 1 to 4

- ▶ Chp 1: Objects and classes
- ▶ Chp 2: Understanding class definitions
- ▶ Chp 3: Object interactions
- ▶ Chp 4: Grouping objects
- ▶ Chp 5: Functional processing of collections

M250 Overview

Phil Molyneux

Agenda

Adobe Connect

M250 Overview

Part 1

Part 2

Part 3

M250 Resources

M250 Chapter
Companions

M250 Assessment

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Overview

Chps 6 and 7

- ▶ Chp 6: More sophisticated behaviour
- ▶ Chp 7: Fixed-size collections — arrays
- ▶ Programming outside BlueJ

M250 Overview

Phil Molyneux

Agenda

Adobe Connect

M250 Overview

Part 1

Part 2

Part 3

M250 Resources

M250 Chapter
Companions

M250 Assessment

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Overview

Chps 9 to 12, and 14

- ▶ Chp 8: Designing classes and code conventions
- ▶ Chp 9: Well-behaved object and debugging
- ▶ Chp 10: Improving structure with inheritance
- ▶ Chp 11: More about inheritance
- ▶ Chp 12: Further abstraction techniques — Interfaces
- ▶ Chp 14: Handling errors

M250 Overview

Phil Molyneux

Agenda

Adobe Connect

M250 Overview

Part 1

Part 2

Part 3

M250 Resources

M250 Chapter
Companions

M250 Assessment

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

M250 Overview

M250 Resources

- ▶ [Module Guide \(online\)](#)
- ▶ [Software Guide \(online\)](#)
- ▶ [M250 BlueJ Activities](#)
- ▶ [M250 Java Reference](#)
- ▶ [Chapter companions — reading guides](#)
- ▶ [Exam Handbook \(print/online under Assessment\)](#)
- ▶ [Study calendar \(HTML/PDF\)](#)
- ▶ [Programming outside BlueJ \(optional\) \(online\)](#)

M250 Resources

M250 Exam Handbook

M250-21J Object-oriented Java programming



M250 Java Reference

This item contains selected online content. It is for use alongside, not as a replacement for the module website, which is the primary study format and contains activities and resources that cannot be replicated in the printed versions.
Copyright © 2021 The Open University

Contents

Introduction	2
1 Strings	3
2 Java collections framework	9
2.1 Collection interfaces	9
2.2 Collection implementation classes	17
2.3 Collection and other utility classes	19
3 Files and streams	24
3.1 Input- and output-related interfaces	24
3.2 Files and pathnames	25
3.3 Reading from character streams	26
3.4 Writing to character streams	29
4 Exceptions	33
4.1 Checked exceptions	34
4.2 Unchecked exceptions	35
5 Java operators used in M250	38
6 Java keywords	39

[M250 Overview](#)

[Phil Molyneux](#)

[Agenda](#)

[Adobe Connect](#)

[M250 Overview](#)

[Part 1](#)

[Part 2](#)

[Part 3](#)

[M250 Resources](#)

[M250 Chapter Companions](#)

[M250 Assessment](#)

[Object-oriented Programming Terminology](#)

[Using BlueJ](#)

[Software & Programming](#)

[What Next ?](#)

[References](#)

M250 Overview

M250 Chapter Companions — Extra Teaching

- ▶ Key areas going beyond the textbook:
- ▶ **Chp 1 Prequel** VMs; Java history; procedural/OO programming
- ▶ **Chp 3** Additional discussion on calling methods and bounding boxes of graphical objects
- ▶ **Chp 6** Iterating over a map; maps with sets as values
- ▶ **Chp 7** Arrays holding references to objects
- ▶ **Programming outside BlueJ** supplement
- ▶ **Chp 8** M250 Code conventions and design
- ▶ **Chp 9** More on static and dynamic code; different kinds of errors, debugging; formatted printing (printf)
- ▶ **Chp 10** Inheritance; constructors; members
- ▶ **Chp 11** More on hash codes, equals, and `Objects.hash` method; `getClass` compared to `instanceof`; immutability of keys
- ▶ **Chp 12** The `Comparable` interface

Agenda

[Adobe Connect](#)

[M250 Overview](#)

[Part 1](#)

[Part 2](#)

[Part 3](#)

[M250 Resources](#)

[M250 Chapter Companions](#)

[M250 Assessment](#)

[Object-oriented Programming Terminology](#)

[Using BlueJ](#)

[Software & Programming](#)

[What Next ?](#)

[References](#)

M250 Overview

M250 Assessment from 2021J

- ▶ 3 TMAs
 - ▶ TMA01 15% Chps 1 to 4
 - ▶ TMA02 15% Chps 6 and 7
 - ▶ TMA03 20% Chps 9 to 12, and 14
 - ▶ Exam 50%
- ▶ Final exam in June
- ▶ To pass, must achieve at least 40% in both OCAS (Overall Continuous Assessment Score) and 30% OES (Overall Examinable Score)

Object Concepts

Sample Code (1)

- ▶ From M250 version to 2020J Unit 2
- ▶ Example code 1

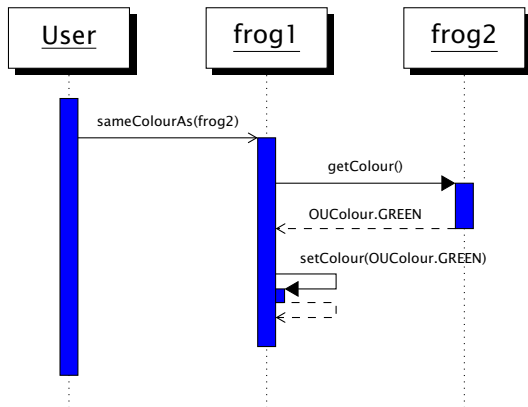
```
public void sameColourAs (Frog aFrog) {  
    this.setColour(aFrog.getColour()) ;  
}
```

```
jshell> frog1.sameColourAs(frog2)
```

- ▶ The **JShell** prompt is just indicative of code usage — usually would be graphical interface here

Unit 2: Object Concepts

Sequence Diagrams



- ▶ The diagrams are similar to the **UML (Unified Modelling Language)** diagrams
- ▶ See also **Unified Modelling Language**
- ▶ See Unit 2 Section 5.3 *Collaborating objects*, page 67

Object Concepts

Sample Code (2)

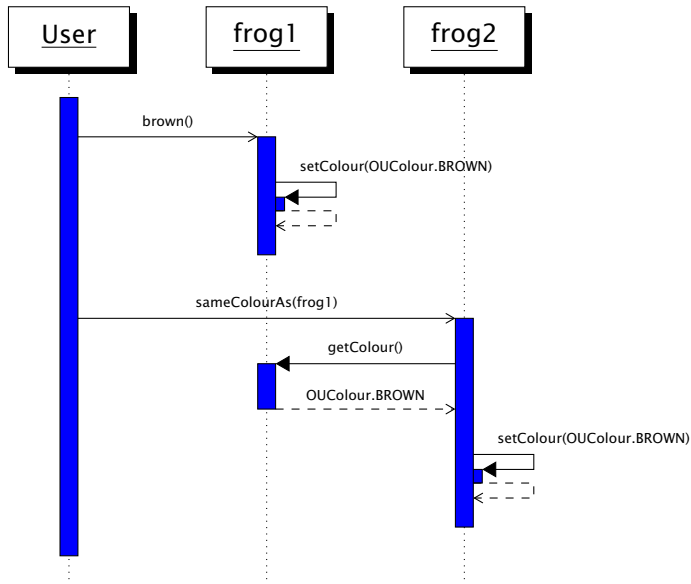
- ▶ From M250 version to 2020J Unit 2
- ▶ Example code (2)

```
public void sameColourAs (Frog aFrog) {  
    this.setColour(aFrog.getColour()) ;  
}  
  
public void brown() {  
    this.setColour(OUColour.BROWN) ;  
}
```

```
jshe11> frog1.brown()  
jshe11> frog2.sameColourAs(frog1)
```

Unit 2: Object Concepts

Sequence Diagrams (2)



Messages and Methods

Computation Model

- ▶ Object-oriented view: programs work using *objects* which communicate by sending *messages*

The diagram shows the expression `objectName.methodName(arg1, arg2, ...)`. Red curly braces and labels are used to identify its components: a brace above `objectName` is labeled "receiver"; a brace above `methodName` is labeled "message"; a brace below `methodName` is labeled "message name"; and a large brace below the entire expression is labeled "message-send".

- ▶ Objects have *state* (implemented by *attributes*) and *behaviour*, its response to messages (implemented by the *protocol*: methods implementing messages)
- ▶ A message must correspond to the signature of one of the methods in the protocol
- ▶ Object terminology is in *M250 Glossary* — this is an object-oriented view of programs
- ▶ Note: the terminology is not standard across all texts — see, for example, [Glossary of Java and Related Terms](#) from Barnes (2000) (author of [BlueJ](#) texts)

BlueJ

Installing & Using BlueJ — 2020J and before

- ▶ [M250 Website](#) > [Resources](#) > [Software resources](#) and follow the instructions
- ▶ Note: macOS users will have to disable *Gatekeeper* see [Disable Gatekeeper](#) — in Terminal say:

```
sudo spctl --master-disable
```

- ▶ On launch, BlueJ looks for **bluej.defs** for its settings — in macOS this should be at [~/M250/BlueJ/BlueJ.app/Contents/Resources/Java/bluej.defs](#)
- ▶ You can have spaces in file or folder names but that will make life painful, so don't
- ▶ See discussion forums for various problems and how to solve them

BlueJ

Installing & Using BlueJ — 2021J and later

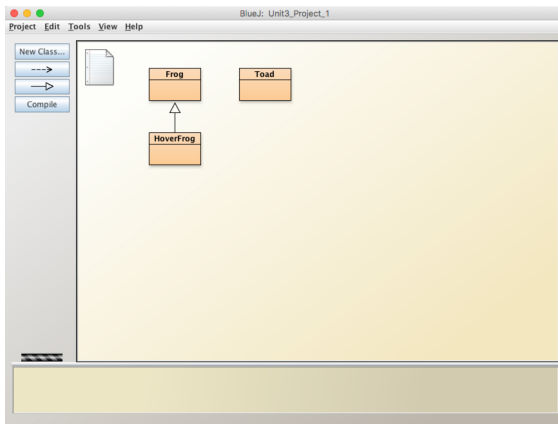
- ▶ [M250 Website](#) > [Resources](#) > [Software resources](#) and follow the instructions
- ▶ Note: macOS users will have to disable *Gatekeeper* see [Disable Gatekeeper](#) — in Terminal say:

```
sudo spctl --master-disable
```

- ▶ Install BlueJ version 4.1.4 from [BlueJ Version History](#) — (27 September 2021) site down!!
- ▶ You can have spaces in file or folder names but that will make life painful, so don't
- ▶ See discussion forums for various problems and how to solve them

BlueJ

BlueJ Interface



- ▶ BlueJ window for Unit 3 Project 1
- ▶ Name the parts ?

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Installing & Using BlueJ

BlueJ Interface

BlueJ Projects

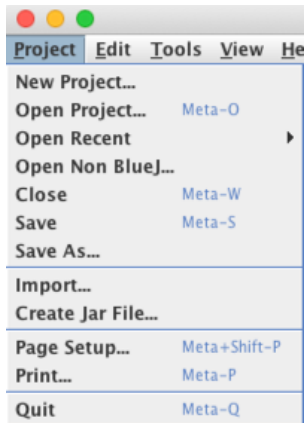
Software &
Programming

What Next ?

References

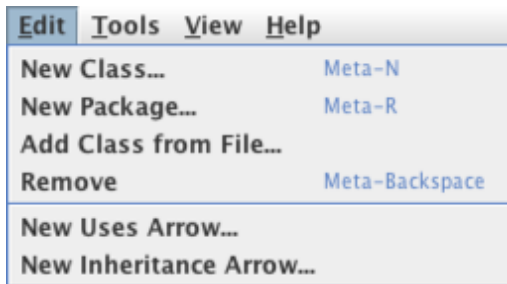
BlueJ

BlueJ Interface: Project Menu



BlueJ

BlueJ Interface: Edit Menu



BlueJ

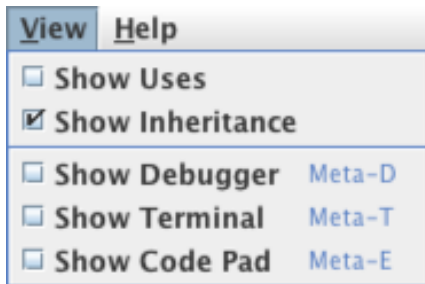
BlueJ Interface: Tools Menu

<u>T</u> ools	<u>V</u> iew	<u>H</u> elp
Compile		Meta-K
Compile Selected		Meta+Shift-K
Rebuild Package		
Reset Java Virtual Machine		Meta+Shift-R
Use Library Class...		Meta-L
Project Documentation		Meta-J
Preferences...		Meta-Comma
OUWorkspace		

- Note the **OUWorkspace**

BlueJ

BlueJ Interface: View Menu



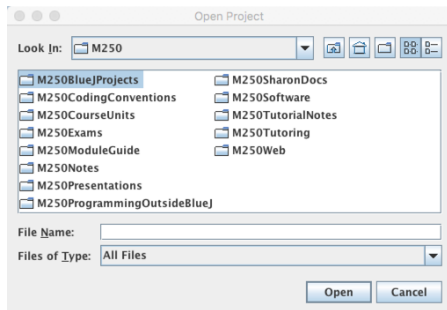
- The **Code Pad** takes part of the **Object Bench**

BlueJ

BlueJ Interface: Help Menu



- ▶ The **OU Class Library** will try to launch a Web browser
- ▶ The documentation is in [oudocs](#) *near* the BlueJ app
- ▶ In Phil's case: (but yours will be somewhere else)
- ▶ `file:///Users/molyneux/MyData/Documents/OU/Courses/Computing/M250/M250Software/M250Working/oudocs/index.html`



- ▶ **Open a project**
- ▶ To navigate into a folder
- ▶ Select **folder** and
- ▶ Double-click (Windows) or Return  or
- ▶ Enter  (Mac)
- ▶ Return  (Mac) means **Open**
- ▶ **New Project**
- ▶ Return  or Enter  (Mac) both mean **Create** (which is wrong)

Learning Software Packages

Key questions

1. Where is the package source ?
2. What version are you using ?
3. What documentation is available ?
4. What are the *names* for the parts of the interface ?
5. How do you leave the package ? How do you enter the package ?
6. Is there any on-line help and, if so, how is it used ?
7. Are there any initialisation files, configuration or preferences and how are they used ?
8. How do you import and export data from the package ?
9. *When all else fails*, how can you obtain advice ?

BlueJ — Exercise

Key Questions (1)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ Where is the package source ?
- ▶ What version are you using ?

BlueJ — Exercise

Key Questions (1a)

- ▶ Where is the package source ?

BlueJ [BlueJ Web site](#)

- ▶ What version are you using ?

4.1.4

- ▶ What version of Java SDK does that use ?

Java 8

BlueJ — Exercise

Key Questions (2)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What documentation is available ?

BlueJ — Exercise

Key Questions (2a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What documentation is available ?
- ▶ What Java documentation is available ?

[Java Documentation](#)

[JDK 17 Documentation](#)

[Java Platform, Standard Edition Documentation](#) — see
the documentation for JDK 8

- ▶ Which was the first version to have JShell ?
Java 9

BlueJ — Exercise

Key Questions (3)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What are the *names* for the parts of the interface ?

BlueJ — Exercise

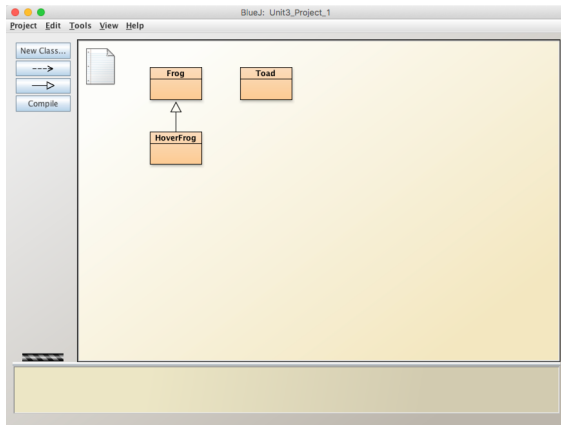
Key Questions (3a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What are the *names* for the parts of the interface ?
See diagram in BlueJ documentation
Main window
Object bench
- ▶ Where is the *Code Pad* ?
- ▶ How do you create new objects or inspect existing ones ?
- ▶ How do you execute methods of an object ?
- ▶ How do you edit the source code of a class ?

BlueJ — Exercise

Key Questions (3b)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What are the *names* for the parts of the interface ?



BlueJ — Exercise

Key Questions (3a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ What are the *names* for the parts of the interface ?
See diagram in BlueJ documentation
Main window
Object bench
- ▶ Where is the *Code Pad* ?  
- ▶ How do you create new objects or inspect existing ones ? *Right-Click on Class* in main window
- ▶ How do you execute methods of an object ? *Right-Click on Object* in Object bench
- ▶ How do you edit the source code of a class ? *Right-Click on Class* and 

BlueJ — Exercise

Key Questions (4)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ How do you leave the package ? How do you enter the package ?

BlueJ — Exercise

Key Questions (4a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ How do you leave the package ? How do you enter the package ?
- ▶ **Enter**
- ▶ Load a project  
- ▶ How do you use a project from the command line with JShell ?
- ▶ **Leave**

BlueJ — Exercise

Key Questions (5)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ Is there any on-line help and, if so, how is it used ?

BlueJ — Exercise

Key Questions (5a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ Is there any on-line help and, if so, how is it used ?

Help > BlueJ Tutorial

Help > BlueJ Web Site

Tools > Project Documentation

Help > Java Class Libraries to view Java API Specification

but you will have to read the documentation

BlueJ — Exercise

Key Questions (6)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ Are there any initialisation files, configuration or preferences and how are they used ?

BlueJ — Exercise

Key Questions (6a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ Are there any initialisation files, configuration or preferences and how are they used ?

Tools > Preferences...

BlueJ.defs

##	<USER_HOME>/. <code>bluej/bluej.properties</code>	(Unix)
##	C:\Winnt\profiles\ <code><USER_NAME>\Bluej\Bluej.properties</code>	(WinNT)
##	C:\<JDK_HOME>\Bluej\Bluej.properties	(Win9x)

- ▶ Notice [SaaSHub: Is BlueJ down ?](#)

BlueJ — Exercise

Key Questions (7)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ How do you import and export data from the package ?

BlueJ — Exercise

Key Questions (7a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ How do you import and export data from the package ?
- ▶ **Export**
- ▶ Using code outside BlueJ
- ▶ **Import**
- ▶ Using code developed in another editor or IDE

BlueJ — Exercise

Key Questions (8)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ *When all else fails*, how can you obtain advice ?

BlueJ — Exercise

Key Questions (8a)

- ▶ Answer the *Key Questions* for BlueJ
- ▶ *When all else fails*, how can you obtain advice ?

M250 Forums

BlueJ Web site

[StackOverflow: Questions tagged \[bluej\]](#)

Writing Programs & Thinking

The Steps

1. Invent a *name* for the program (or function)
2. What is the *type* of the function ? What sort of *input* does it take and what sort of *output* does it produce ? In Python a type is implicit; in other languages such as Haskell a type signature can be explicit.
3. Invent *names* for the input(s) to the function (*formal parameters*) — this can involve thinking about possible *patterns* or *data structures*
4. What restrictions are there on the input — state the preconditions.
5. What must be true of the output — state the postconditions.
6. *Think* of the definition of the function body.

Writing Programs & Thinking

The *Think* Step

► How to Think

1. Think of an example or two — what should the program/function do ?
2. Break the inputs into separate cases.
3. Deal with simple cases.
4. Think about the result — try your examples again.

► Thinking Strategies

1. Don't think too much at one go — break the problem down. Top down design, step-wise refinement.
2. What are the inputs — describe all the cases.
3. Investigate choices. What data structures ? What algorithms ?
4. Use common tools — bottom up synthesis.
5. Spot common function application patterns — generalise & then specialise.
6. Look for good *glue* — to combine functions together.

What Next ?

Programming, Debugging, Psychology

Although programming techniques have improved immensely since the early days, the process of finding and correcting errors in programming — known graphically if inelegantly as *debugging* — still remains a most difficult, confused and unsatisfactory operation. The chief impact of this state of affairs is psychological. Although we are happy to pay lip-service to the adage that to err is human, most of us like to make a small private reservation about our own performance on special occasions when we really try. It is somewhat deflating to be shown publicly and incontrovertibly by a machine that even when we do try, we in fact make just as many mistakes as other people. If your pride cannot recover from this blow, you will never make a programmer.

Christopher Strachey, Scientific American 1966 vol 215 (3) September pp112-124

What Next ?

To err is human ?

- ▶ To err is human, to really foul things up requires a computer.
- ▶ Attributed to [Paul R. Ehrlich](#) in [101 Great Programming Quotes](#)
- ▶ Attributed to [Bill Vaughn](#) in [Quote Investigator](#)
- ▶ Derived from [Alexander Pope](#) (1711, [An Essay on Criticism](#))
- ▶ *To Err is Humane; to Forgive, Divine*
- ▶ This also contains
 - A little learning is a dangerous thing;
Drink deep, or taste not the [Pierian Spring](#)*
- ▶ In programming, this means you have to *read the fabulous manual* ([RTFM](#))

What Next ?

Chps 1-4, TMA01

- ▶ Chps 1-3, Data types and expressions; Class definitions
- ▶ Tutorial 10:00 Sunday 12 November 2023 online
- ▶ Chps 4-5, Iteration, collections; Functional Java (optional)
- ▶ Tutorial 10:00 Sunday 19 November 2023 online
- ▶ TMA01 Thursday 14 December 2023

M250 Overview

Phil Molyneux

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

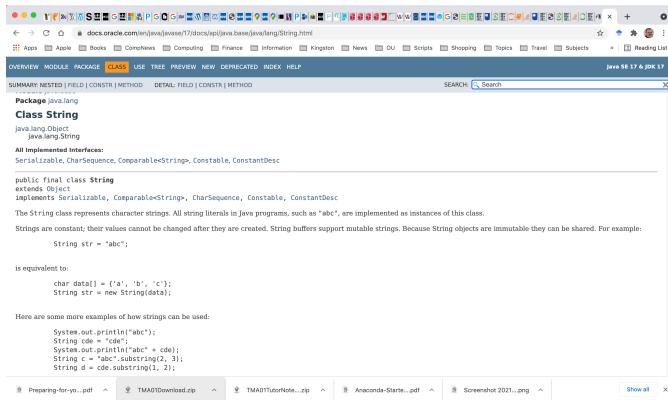
What Next ?

References

- ▶ [Java Documentation](#) — BlueJ has JDK 7 embedded, JDK 13 is current (2019)
- ▶ [JDK 13 Documentation](#)
- ▶ [Java Platform API Specification](#)
- ▶ [Java Language Specification](#)
- ▶ [JDK Documentation](#) > [API Documentation](#) > [java.base](#)
 - ▶ [java.lang](#) — fundamental classes for the Java programming language
 - ▶ [java.util](#) — Collections framework

Java

API Documentation (1)



- ▶ **Strings** are *immutable* objects
- ▶ See **`java.lang.StringBuilder`** for *mutable* strings
- ▶ In a *functional programming approach* everything is immutable — it makes life simpler (but at a cost)

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Java Documentation

Books Phil Likes

Java

API Documentation (2)

The screenshot shows the Java SE 17 API documentation. The top navigation bar includes links for OVERVIEW, MODULE, PACKAGE, CLASS (highlighted), USE, TREE, PREVIEW, NEW, DEPRECATED, INDEX, and HELP. The version is identified as Java SE 17 & JDK 17. A search bar is present on the right. The main content area is divided into two sections: **equals** and **contentEquals**.

equals

```
public boolean equals(Object anObject)
```

Compares this String to the specified object. The result is true if and only if the argument is not null and is a String object that represents the same sequence of characters as this object.

For finer-grained String comparison, refer to `Collator`.

Overrides:
`equals` in class `Object`

Parameters:
`anObject` - The object to compare this String against

Returns:
true if the given object represents a String equivalent to this string, false otherwise

See Also:
`compareTo(String)`, `equalsIgnoreCase(String)`

contentEquals

```
public boolean contentEquals(StringBuffer sb)
```

Compares this String to the specified StringBuffer. The result is true if and only if this String represents the same sequence of characters as the specified StringBuffer. This method synchronizes on the `StringBuffer`.

For finer-grained String comparison, refer to `Collator`.

Parameters:
`sb` - The StringBuffer to compare this String against

- Remember **(==)** tests for *identity* — what does this mean ?

Agenda

Adobe Connect

M250 Overview

Object-oriented
Programming
Terminology

Using BlueJ

Software &
Programming

What Next ?

References

Java Documentation

Books Phil Likes

M250

Books Phil Likes

- ▶ M250 is self contained — you do not need further books — but you might like to know about some:
- ▶ Sestoft (2016) *Java Precisely* — the best short reference
- ▶ Evans, Flanagan (2018) *Java in a Nutshell* — the best longer reference
- ▶ Barnes, Kölling (2016) *Objects First with Java* — the BlueJ book — see www.bluej.org for documentation and tutorial
- ▶ Bloch (2017) *Effective Java* — guide to best practice