

Simplicity, Consistency, Memorability & Power in the Microworlds of M206

Dramatizing Object Creation & Destruction

**Simon Holland
Department of Computing
Faculty of Mathematics and Computing
Open University**

Background: two paradigm shifts

- **Objects First**
- **Integrating (HCI) Human Computer Interaction with Programming**

Selection of basic object concepts to be taught

object	object creation
message	object destruction
class	polymorphism
instance	separable user interface
argument	method
subclass	keyword
superclass	object dictionary
assignment	object reference
garbage collection	variable
message reply	type (or lack of type) of variable

Objects

Each object is *something like an application program* in that

- it can *respond to commands*
(exactly which commands will depend on the kind of object)
- it may be able to *store information*
(the kind of information will depend on the kind of object) and
- it can generally *give you back information or do work in response to commands*
(of what kind will depend on the kind of object)

- Every part of a piece of software built using objects *is itself an object*
- The only way to get an object to do something is to send it a message
- Some message require one or more items of extra information - called arguments
- Every message has a message reply - only some are useful

Introduction to Frog microworld

- **Sending messages to objects via the GUI interface**
- **Sending message to objects via the programming language**
- **Inspecting objects**
- **Using hci concepts to help teach programming concepts**
- **Message replies**

Summary: visualising objects, state, behaviour and messages

Chp13/Behaviours

Practicals Class-browser Amphib World Workspace Notes

Graphical interface for sending messages

abby
freda
harriet
tom

Inspect

left right up down
green brown jump home
hover... colour...
sameColourAs...

Go to guide Help Detach

Chp13/Behaviours/Workspace

Variables specific to world in this section are shown here

Evaluation pane

```
tom colour: Red
tom left

abby := tom
tom := freda
freda := abby

tom := 2
tom + tom
fergie := Frog new
```

Variables

```
abby
freda
harriet
tom
```

Evaluate selection Cut Copy Paste Inspect Delete

Display pane

```
An instance of class Toad (position 11, colour Brown)
An instance of class Toad (position 9, colour Brown)
An instance of class Toad (position 9, colour Yellow)
An instance of class Toad (position 11, colour Brown)
An instance of class Toad (position 9, colour Yellow)
?
```

show message answers hide message answers

Inspect last message answer

Clear Copy

Chp13/Behaviours/Class browser

Class:	Instance variables:	Methods:
Frog	colour	left
HoverFrog	position	position:
Toad		release
		right
		sameColourAs:

New Remove
Look

sameColourAs: anAmphibian
"make myself the same colour as another amphibian"
self colour: anAmphibian colour

Relating names to existence: object creation and destruction

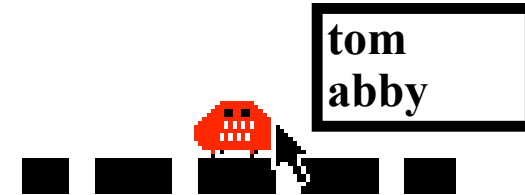
Is the name of an object part of that object? Can names be changed?

- Can a single object be given more than one name?
- Can the names (of say a frog and a toad) be swapped?
- Can a name be used to refer to an object of a completely different class?
- Can objects be destroyed? If so, how?
- Can objects be created? If so, how do you make them stick around?

Concepts of variable, object dictionary, object reference, garbage collection, object creation, object destruction

- **Initial situation in microworld**

freda is a frog
harriet is a hoverfrog
tom is a toad



- **Can a single object be given more than one name?**

abby := tom

- **Can the names (of say a frog and a toad) be swapped?**

tom := freda
freda := abby

- **Can a name be used to refer to an object of a completely different class?**

tom := 2

- **Can objects be destroyed?
If so, how?**

!!!!

- **Can objects be created?**

Frog new

- **If so, how do you make them stick around?**

fergie := Frog new

Visualising execution model for objects

Simpler problem of visualising message precedence

Uniform extension of precedence tool gives full execution model

Full generalisation only became apparent by refining simpler tool until uniform and consistent

Pedagogical Constraints on Microworlds

A given Microworld should be

**Simple
Uniform
Consistent
Memorable
Powerful**

All key concepts should be perceptualised or dramatized in a consistent manner in a given microworld

The microworld should allow students to deal with key concepts interactively and concretely