Your first program!

Display a map of Paris
Spotlight position of Louvre museum
Highlight line 8 of the metro
Show buildings and trams

Filling in the feature body

class PREVIEW
  inherit TOURISM
  feature explore_on_click is
    -- Show city info and route.
    do
      Paris, display
      Louvre, lighten
      Line8, spotlight
      Paris, build
      Paris, equip
    end
end

Program formatting

Between adjacent elements:
  * break: one or more spaces, "tabs", "carriage returns"

All kinds of break are equivalent

Typographical variations (boldface, italics, colors) do not affect meaning (semantics) of program
Style rules

For indentation, use tabs, not spaces
Use this property to highlight the structure of the program, particularly through indentation

Feature call

The fundamental mechanism of program execution: apply a "feature" to an "object"
Basic form: your_object.your_feature

- Object - (target of the call)
- Feature of the call

Predefined objects

Paris, Louvre, Metro, and Line8 are names of predefined objects
Defined in class TOURISM from which PREVIEW inherits.
display, lighten, spotlight, and animate are features, applicable to these objects

More style rules

- Class name: all upper-case
- Period in feature call: no space before or after
- Names of predefined objects: start with upper-case letters
- New names (for objects you define) start with lower-case letters
Object technology

We work with objects
Our style of programming: Object-Oriented programming
Abbreviation: O-O
More generally, "Object Technology": includes O-O databases, O-O analysis, O-O design...

Software execution is made of operations on objects — feature calls

A distinct mode of expression

Paris, display
next_message, send
computer, shut_down
telephone, ring

Every operation applies to an object
(the target of the call)

What's an object?

It's a software notion: machine known through the operations applicable to it.

Three kinds of object:
- Some reflect material objects of the outside world: the Louvre, Paris, a metro car.
- Some correspond to abstract notions from the outside world: a line, a route...
- Some express purely software notions ("data structures")

A key attraction of object technology is its modeling power: connect software objects to objects of the problem domains
You should not, however, confuse them
In this course, "object" by default means software object

Features, commands and queries

Feature: an operation available on a certain class of objects

Three kinds:
- Command
- Query
- Creation procedure (seen later)

Queries

Goal: obtain properties of objects

Should not modify the object, or any other

Examples, for "route" objects:
- What is the origin (first station) of Line8?
- What is the end point of Line8?
- How many stations does Line8 have?
- Which stations does Line8 traverse?

Commands

Goal: produce a change on an object, or several

Examples, for "route" objects:
- Animate Line8
- Append (add at the end) a station to Line8.
- Prepend (add at the beginning) a station to Line8
A command

Intro. to Programming, lecture 2: Dealing with objects I

A query

Intro. to Programming, lecture 2: Dealing with objects I

Command-query separation principle

Asking a question shouldn't change the answer

An object is a machine

An executing program is a machine
It's made of smaller machines: objects

During execution there may be many objects (e.g. millions)

Two views of objects

An object has data, stored in memory.

An object is a machine offering queries and commands.

The connection:

The operations that the machine provides (2) access and modify the object's data (1).

An object is a machine

A machine, hardware or software, is characterized by the operations ("features") users may apply
Objects: a definition

An object is a software machine allowing programs to access and modify a collection of data.

Defining and classifying features

A feature is an operation that programs may apply to certain classes of objects.

- A feature that accesses an object is a query
- A feature that may modify an object is a command

Using queries

Queries are as important as commands

Queries don’t “do” anything, but yield a value, e.g., Line8.origin yields the starting station of Line8

You may work with the return values of queries, e.g., highlight the starting station on the screen

Features may have arguments

Task:
> Show starting point of Line8 on "console" window

You need:
> Predefined object Console.
> Feature show applicable to Console.
> The object Line8
> Feature origin returning starting point and applicable to Line8

The new feature call:
> Console.show(Line8.origin)

Extending the feature body

```pascal
class PREVIEW
  inherit TOUR
feature explore_on_click is
  -- Show city info, route, and the route's origin.
  do
    Paris.display
    Louvre.lighten
    Line8.spotlight
    Paris.build
    Paris.equip
    Console.show(Line8.origin)
  end
end
```

Features with arguments

```
your_object.your_feature(some_argument)
```

some_argument is a value that your_feature needs

Example: feature show must know what to show.

Some concept as function arguments in maths:
```
cos(x)
```

Features may have several arguments:
```
x.f(a, b, c, d) -- Separated by commas
```

In well written O-O software, most have 0 or 1 argument
A distinct mode of expression

Paris, display
next_message, send
computer, shut_down
telephone, ring

Every operation applies to an object

A distinct mode of expression

Paris, display
next_message, send_to (recipient)
computer, shut_down_after (3)
telephone, ring_several (10, Loud)

Every operation applies to an object and may take arguments

Scaling up

One of the toughest issues in learning software is to find solutions that work well both “in the small” and “in the large”.

That’s the goal for the techniques we teach in this course.

An object has an interface

An object has an implementation

Information hiding
What we have seen so far

- Classes (a first view)
- Basic program text structure
- Objects
- Features
- Feature call
- Command/query distinction
- Feature arguments
- Information hiding
- Basic ideas of object technology

Seen so far

End of lecture 2