Software Project Planning & Management

Lenka Pitonakova

Student Conference in Complexity Science, SCCS 2014

nag

Welcome! Today you will learn how to:

- Make code that lasts
- Plan your code
- Make your code happen
- Share your code

Planning your code: Language

It’s both about personal taste
and about raw facts

Planning your code: Language

Compiled	**vs.**	**Interpreted**
C++, Java		Python, JavaScript
Fast to run		Fast to execute

Planning your code: Language

Object-oriented	**vs.**	**Procedural**
Classes, instances		Modules
Encapsulation, inheritance		Procedures, local variables
E.g. a game		E.g. a numerical library

Planning your code: Language

Many more programming styles:
Imperative, Functional, Event-Driven, …

Think what you need it to do and how YOU like to think about problems

Planning your code: Structure

Once decided what style and language to use, you need to plan how you will implement your ideas.

-> Abstract from details and visualise
-> How will data be obtained and manipulated?
-> What will generate output?

Planning your code: Structure

Call graphs (procedural programming)

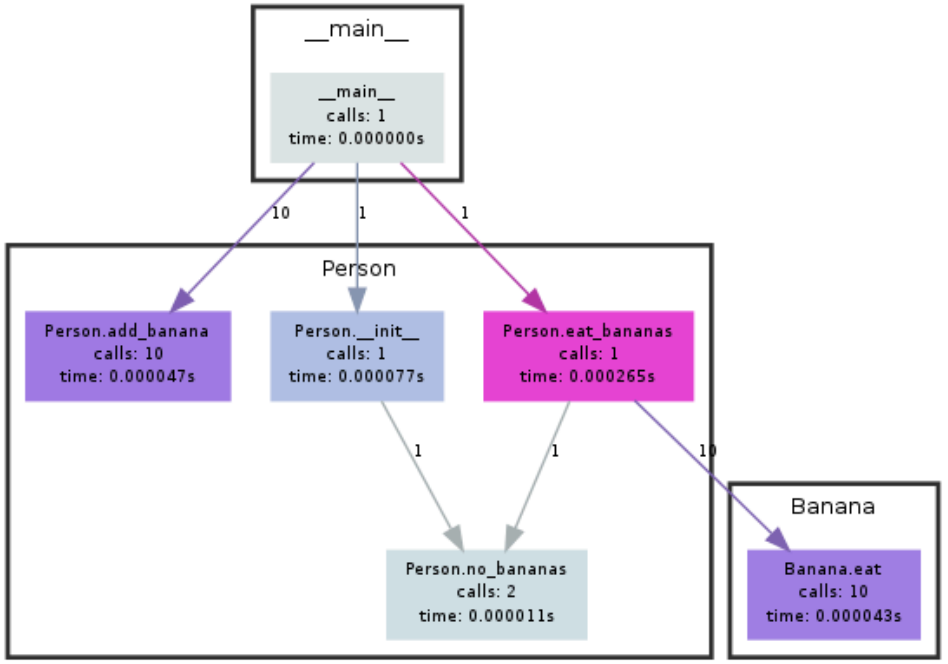
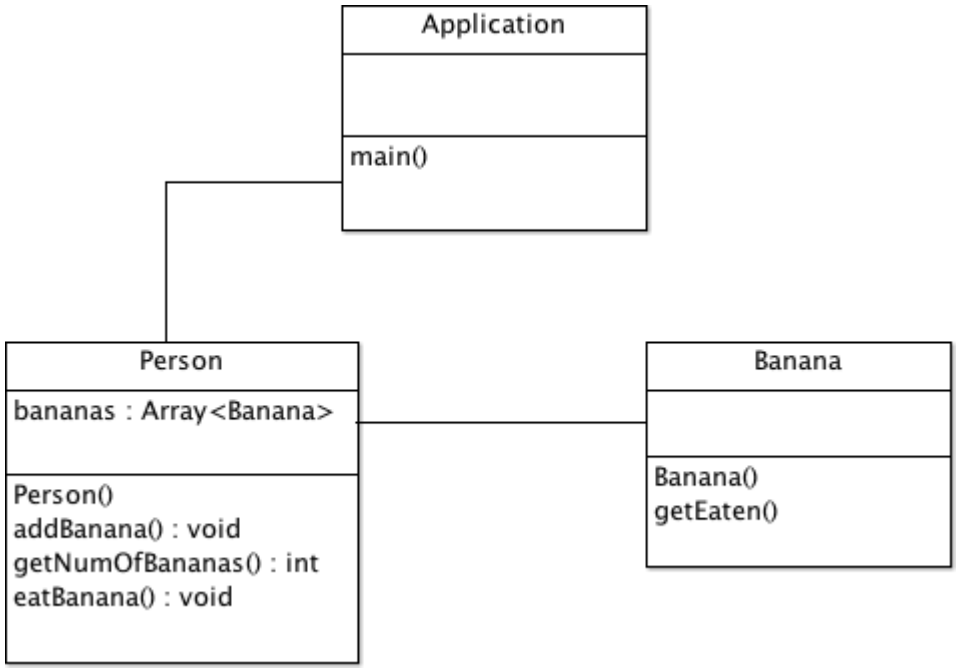


Image source: http://en.wikipedia.org/wiki/Call_graph

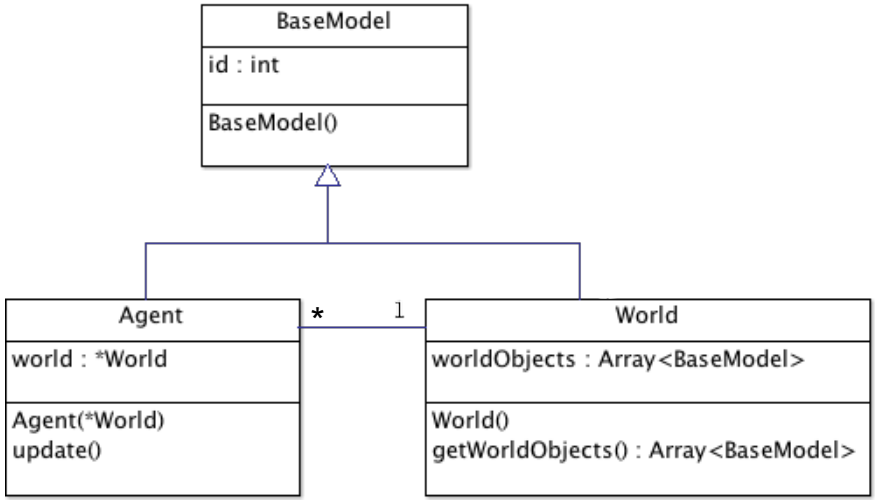
Planning your code: Structure

Class diagrams
(object-oriented programming)



Planning your code: Structure

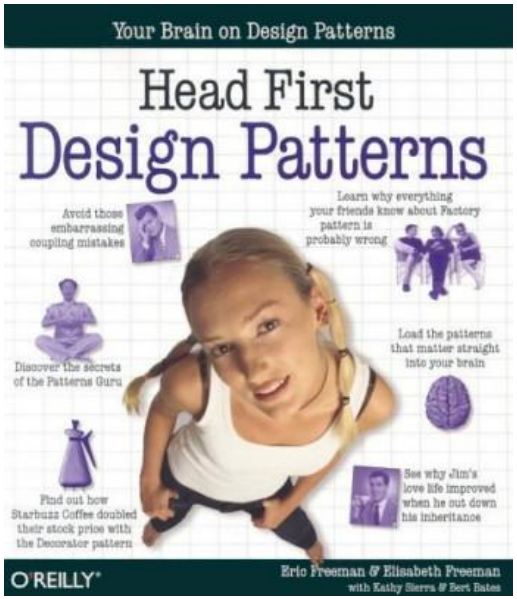
Example: Model-View-Controller



Planning your code: Structure

MVC is a **design pattern** often used for games or web apps

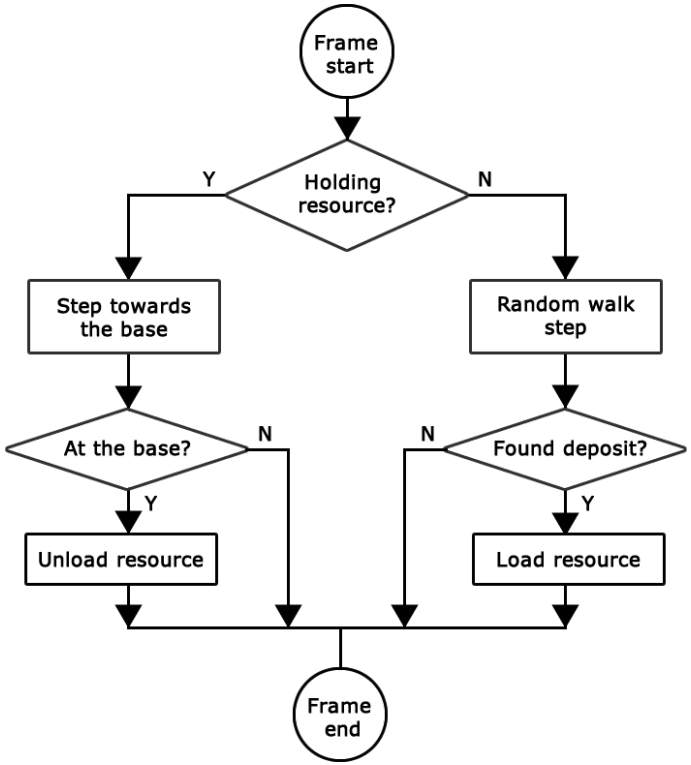
Many other design patterns: Singleton, Blackboard, Factory, …



Planning your code: Algorithm

Flow charts

- Quick overview
- Easier to change compared to a program



The importance of visualisation

- **Before** you code
  - Plan before creating a mess
- **During** coding
  - Complex changes
- **After** coding:
  - Keep graphs up to date, they will be useful later