

Perl in Bioinformatics

Gang Chen
chengang@bgitecholutions.com

October 24, 2015

Outline

- 1 Objec Oriented Prorgamming
- 2 OOP in Perl
- 3 Bioinformatics Projects in Perl

Next

- 1 **Objec Oriented Prorgamming**
 - Basic Ideas
 - Concepts
- 2 OOP in Perl
- 3 Bioinformatics Projects in Perl

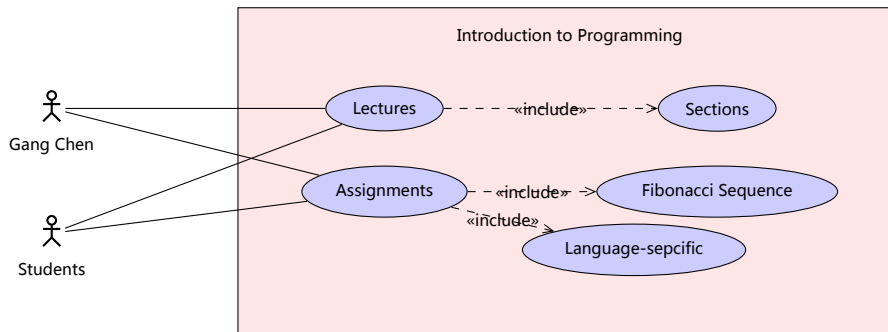
Next

- 1 Object Oriented Programming
 - Basic Ideas
 - Concepts
- 2 OOP in Perl
- 3 Bioinformatics Projects in Perl
 - Annovar
 - Cirocs
 - BioPerl

Object-oriented Programming (OOP)

- OO is a language-independent concept
- in principle not limited to programming (OO design)
 - ⇒ databases, business plans
- improves reusability and exchangability of code
- separation of partial problems
- "real world" modelling
- representation in **Universal Markup Language (UML)**

Object-oriented Programming (OOP)



Basic ideas

- **Everything is an object**
Gang, students, lecture
project, world, spy, information
- **Objects interact by sending/receiving messages**
Gang → students: object orientation is a concept
world → map: what is the object at position X?
- **An object consists of objects**
a course consists of lectures
a world consists of land or water fields

Basic ideas

- Every object has a type
Gang is a teacher
the map is a rectangle of land / water fields
- All objects of the same type understand the same messages
all students hear the lecture
all spies can retrieve information

Next

1 Object Oriented Programming

- Basic Ideas
- Concepts

2 OOP in Perl

3 Bioinformatics Projects in Perl

- Annovar
- Cirocs
- BioPerl

Classes, Interfaces and Methodse

World
spies : ListOfSpies
setSpyCount(count: int) getSpyCount() : int getMap() : Map

class

attributes

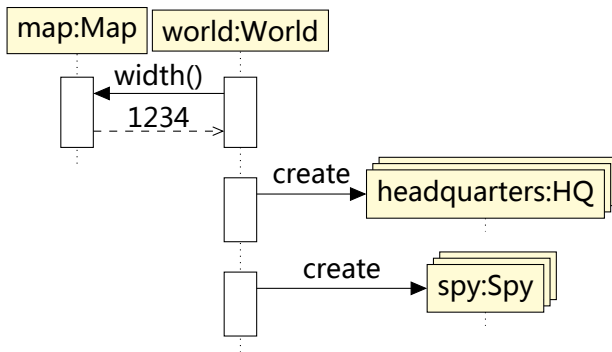
methods

Map
tiles: VectorOfTiles
getWidth() : int getHeight() : int at(x: int, y:int) : Tile

Classes, Interfaces and Methods

- **classes** describe the type of objects (define their **interface**)
- the interface consists of **methods** and **attributes** / properties
- methods
 - are functions that operate on objects of this class
 - can take extra arguments of arbitrary types
 - can return values of arbitrary types
- attributes are objects of arbitrary other types

Objects/Instances



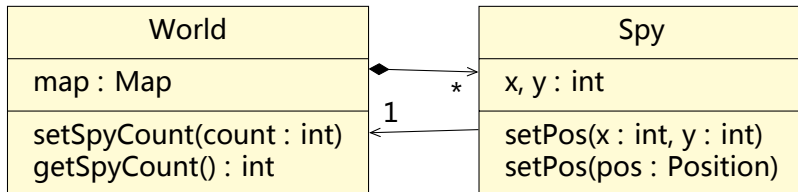
- every object has an immutable class assigned when it is created
- objects communicate via their class interfaces
- classes can communicate via static member functions

Overloading and signature

Spy
setPos(x : int, y : int) setPos(pos: Position)

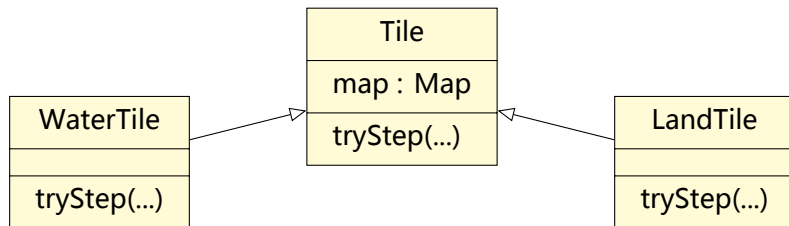
- a method is described by name and **signature**
- signature is formed by the types of all taken arguments
 - setPos(x : int, y : int)
 - setPos(pos : Position)
- methods with identical names but different arguments can exist in one class -- **overloading**
- return type is not part of the signature -- cannot always resolve overload at compile time

Composing classes



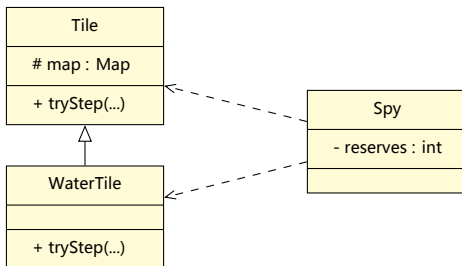
- objects are made of objects (**attributes**) --- classes declare the types of these objects
- simple attributes appear below class name
- complex classes shown as **composition**
- "has-a" or "has-many" relations:
 - a world hosts many spies,
 - a spy belongs to one world

Inheritance and class hierarchy



- subclasses of classes -- **class hierarchy**
- subclasses inherit methods and attributes of all superclasses
- no need to duplicate code
- .. but methods might behave differently (**polymorphism**)
- **abstract classes** implement only parts of the interface

Implementation hiding



- **public** (`+') elements are visible to all
- **protected** (`#') elements are only visible to derived classes
- **private** (`-') attributes or methods are not visible to other objects
- `map` is protected \Rightarrow visible to `WaterTile`, not to `Spy`
- `Spy` can access `tryStep` of all tiles

Next

- 1 Objec Oriented Prorgamming
- 2 OOP in Perl
- 3 Bioinformatics Projects in Perl

OOP in Perl

- A class is a package.
- An object is a reference that knows its class.
- A method is a subroutine.

OOP in Perl: Example

- snp.pm
- runoop.pl

Other OOP Systems in Perl

- Moose

Next

- 1 Objec Oriented Prorgamming
- 2 OOP in Perl
- 3 Bioinformatics Projects in Perl**
 - Annovar
 - Cirops
 - BioPerl

Next

1 Objec Oriented Prorgamming

- Basic Ideas
- Concepts

2 OOP in Perl

3 Bioinformatics Projects in Perl

- **Annovar**
- Cirocs
- BioPerl

Annovar

Annovar

ANNOVAR: Functional annotation of genetic variants from high-throughput sequencing data

<http://www.openbioinformatics.org/annovar/>

Next

1 Objec Oriented Prorgamming

- Basic Ideas
- Concepts

2 OOP in Perl

3 Bioinformatics Projects in Perl

- Annovar
- **Cirocs**
- BioPerl

Circos

Circos

Circos is a software package for visualizing data and information. It visualizes data in a circular layout — this makes Circos ideal for exploring relationships between objects or positions.

<http://circos.ca>



Next

1 Objec Oriented Prorgamming

- Basic Ideas
- Concepts

2 OOP in Perl

3 Bioinformatics Projects in Perl

- Annovar
- Cirops
- **BioPerl**

BioPerl

<http://bioperl.org>

Installation

- from source
- cpan (recommended)

Thanks!