Dynamic Programming Languages

Carl Friedrich Bolz, David Schneider
Dynamische Programmiersprachen
Heinrich-Heine-Universität Düsseldorf
Sommersemester 2010

What this course is about

Getting a feel for dynamic languages:

- What a dynamic language is
- What distinguishes different dynamic languages
- Learn Python and see other languages (Smalltalk, Ruby, JavaScript, SELF)
- Typical implementation techniques

Fahrplan

- Introduction & Python
- Object Models
  - Smalltalk: message-based
  - Python: attribute-based
  - prototypes
  - multiple inheritance
  - duck typing

Fahrplan

- Introduction & Python
- Object Models
  - Smalltalk: message-based
  - Python: attribute-based
  - prototypes
  - multiple inheritance
  - duck typing
- Implementation
  - Interpreters
  - implementation of object models
  - bytecode-based implementations
  - PyPy

Fahrplan

- Introduction & Python
- Object Models
  - Smalltalk: message-based
  - Python: attribute-based
  - prototypes
  - multiple inheritance
  - duck typing
- Implementation
  - Interpreters
  - implementation of object models
  - bytecode-based implementations
  - PyPy
- Bonus:
  - other execution models
  - partial evalution
  - ...

Properties of Dynamic Languages

A dynamic language is a language with:

- Dynamic typing
- Most things changeable at run-time
- Reflection
- “Late-Bound Everything”

Properties of Dynamic Languages (2)

In practice, dynamic languages are often:

- Interactive
- Garbage-Collected
- Interpreted (and slower)
- "Everything is an Object"

Interactivity

Interactivity:

- Type and immediately execute parts of programs

Dynamic Typing

Dynamic Typing:

- Types not declared in the source code
- Types attached to values at run-time

Garbage-Collected

Garbage-Collected:

- No manual memory management

“Late-Bound Everything”

Late Binding:

- The object that a variable name references can only be determined at run-time

“Everything is an Object”

Everything is an Object:

- Numbers, Functions, Classes, their Instances, Lists, Modules, etc.
- All objects are manipulated in the same way

Changeable at run-time

Most things changeable at run-time, e.g.:

- The bindings
- All objects (classes, methods, modules...)

Reflection

Reflection:

- The way in which we can inspect (and change) the running program from within itself
- Non-modifying reflection is sometimes called **introspection**

Pros and Cons

Pros and Cons of Dynamic versus non-Dynamic languages:

Pros and Cons

...No kind of programming language is inherently better or worse

- It's about the "right tool for the right job"
- We will come back to Pros and Cons

Examples

Dynamic languages (exercice: why?):

- Python
- Ruby, Perl, Smalltalk, Lisp, Scheme, JavaScript, Lua, Self...
- Prolog

Anti-Examples

Non-dynamic languages (exercice: why?):

- C
- Java (but getting there)
- Assembler
- Haskell, C++, Ada, ML, Pascal, Fortran, Cobol...

End