Alternative Programming Languages

Mercury - JavaFX - Piet - Go

Serafín Vélez Barrera
serafin.velez.barrera@gmail.com

Universidad de Granada

11 de Enero de 2011

Index

What is a programming language

Definition

A programming language is an artificial language designed to communicate instructions to a machine

Types

1. General purpose: C, C++, Java, etc

Types

1. General purpose: C, C++, Java, etc
2. Specific language: VHDL

Types

1. General purpose: C, C++, Java, etc
2. Specific language: VHDL
3. Others: Artistic, etc

Mercury

Introduction

- Designed and implemented by a small group of researchers at the University of Melbourne (Australia) between this group there are: Fergus Henderson, Thomas Conway, etc.
- Based on the paradigm of purely declarative programming, and was designed to be useful for the development of large and robust "real-world" applications.
- It improves on existing logic programming languages by providing increased productivity, reliability and efficiency, and by avoiding the need for non-logical program constructs.

Introduction

This language differentiates of the rest by the concept to program.

Why?

Because is a logical programming and the syntax is similar to Prolog, with some additional declarations for types, modes, determinism, the module system, and pragmas, and with the distinction that function symbols may stand also for invocations of user- defined functions as well as for data constructors.

Syntax

Mercury have a syntax similar to Pascal but this language use terms like:

- Tokens
- Predicates

because is a logical programming language and is:

- A strongly typed language:

```
:- type list(T) ---> [] ; [T | list(T)].
```

- A strongly moded language:

```
:- mode append(in, in, out).
```

Developing an app

In other Workshop

JavaFX™

Hey guy, this rocks!!

Introduction

JavaFX is a language that is oriented to create RIA app's (RIA = Rich Internet Application) and this app's integrate the simplicity of a web app and the potentially of a desktop app.
JavaFX include various technologys like **JavaFX Script**, **JavaFX Mobile**, and **JavaFX TV**.
We are going to see **JavaFX Script**, apart of to be a scripting language is a language to desing User Interfaces, so have the powerfull of a script with the design of interfaces in graphic mode.

Developing an app

In other Workshop - Maybe later there will be a demo.

Piet

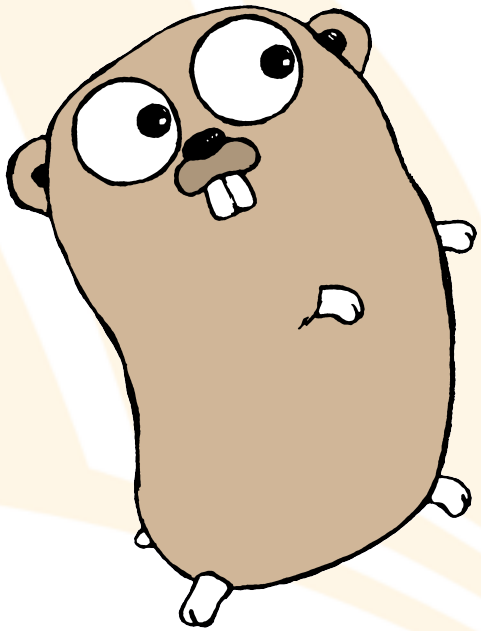
Introduction

Piet (named after painter Piet Mondrian) is one of the most known esoteric programming languages, which uses images as source code. The language uses 20 colors, and the commands are encoded as changes of color between adjacent pixels.
Piet was created by David Morgan-Mar, who aimed for a language which would have its code look like abstract art. The language hasn't spawned any dialects; there are several implementations, which differ slightly, mainly in the way the colors are processed.

Developing an app

In other Workshop - Maybe later there will be some examples.

GO



Introduction

GO was designed by Ken Thompson, Rob Pike, Robert Griesemer + Contributors + Open Source comunity.

Features:

- General purpose
- Concise syntax
- Expressive type system
- Concurrency
- Garbage collection
- Fast compilation
- Efficient execution

Introduction

Combines the powerfull and safety of a statically typed compiled language with the the expressiveness and convenience of a dynamically typed interpreted language.
Designed as a systems language (for develop app like: Web Servers, Browsers, etc).
Influenced by other languages: C, Modula, Pascal, Python, etc.

Syntax

- Comments

```
/* This is a comment; no nesting */
// So is this.
```

- Structs

```
type Point struct { x, y float }
func (p Point) Abs() float {
    return math.Sqrt(p.x*p.x + p.y*p.y)
}
```

Syntax

- Methods

```
type Vector []float
func (v Vector) Abs() float {
    sumOfSquares := 0.0
    for i := range v {
        sumOfSquares += v[i]*v[i]
    }
    return math.Sqrt(sumOfSquares)
}
```

Syntax

- Interfaces

```
type Abser interface {
    Abs() float
}
var a Abser

a = Point{3, 4}
print(a.Abs())

a = Vector{1, 2, 3, 4}
print(a.Abs())
```

Syntax

- Visibility

  ```
  ThisNameIsPublic
  thisOneIsNot
  ```

Developing an app

1st Step - Install the environment

Install C tools:

```
sudo apt-get install bison gawk gcc libc6-dev make
```

Install mercurial:

```
sudo easy_install mercurial
```

Fetch the repository:

```
hg clone -u release https://go.googlecode.com/hg/ go
```

Developing an app

1st Step - Install the environment

Install the environment:

```
sudo mv go/ /usr/local/
cd /usr/local/go/src
./all.bash
```

See bash and GO documentation to set up GO Path in the environment.

Developing an app

1st Step - Install the environment

Fetch last version:

```
cd go/src
hg pull
hg update release
./all.bash
```

Developing an app

2nd Step - Code

We can code the easiest example at last version: **“Hello world’ 2.0** :)

```
package main

import "fmt"

func main() {
    fmt.Printf("Hello world 2.0\n")
}
```

Developing an app

3rd Step - Compile and test

- Compile:

```
6g helloworld.go
```

- Link:

```
6l helloworld.6
```

- Test:

```
./6.out
```

Demos

DEMO!

Resources

Go

- Website Go Lang
- Youtube channel Google Developers

Mercury

- Website

JavaFX

- Website

Piet

- Website

License

This document have the following license:

Serafín Vélez Barrera – Oficina de Software Libre
serafin.velez.barrera@gmail.com