# Treball Dirigit LP: Crystal

Hash: 36367

# Índex

- Descripció i principals aplicacions
- Història i relació amb altres LPs
- Propòsit del llenguatge
- Paradigmes de programació que admet Crystal
- Sistema d’execució
- Sistema de tipus
- Altres característiques particulars
- Exemples de codi il·lustratius
- Visió personal i crítica de Crystal

# Descripció de Crystal

- Llenguatge de codi obert
- Llicència Apache 2.0
- Es pot utilitzar en diferents sistemes operatius: Linux, macOS, FreeBSD i OpenBSD.
- Es pot executar a les plataformes IA-32 (i386), AArch64 i X86-64.

# Aplicacions de Crystal

- Programació de baix nivell i enllaços C (té moltes funcions que el fan útil per a aplicacions incrustades i IoT)
- Pot funcionar amb un nombre creixent de bases de dades SQL i NoSQL, té un mòdul de bases de dades propi en un paquet anomenat crystal-db
- Creació d’aplicacions web

# Història de Crystal

- Va ser anomenat originalment Joy
- Creat el 2011, 1a versió llençat el 2014 i l'última versió (1.4.1), el 22 d'abril de 2022
- Dissenyat pels informàtics Ary Borenszweig, Juan Wajnerman i Brian Cardiff i desenvolupat per l'empresa Manas Technology Solutions.
- 2 objectius originals:
    - "Combinar l'eficiència de Ruby per escriure codi i l'eficiència de C per a executar codi".
    - "Que el compilador comprengui el que volem dir sense haver d'especificar el tipus per a tot arreu".

# Relació amb LPs similars

- Com s'ha mencionat anteriorment, és un llenguatge inspirat en Ruby i C, però també en menor part, pels llenguatges de programació Python, Rust i Go
- La seva sintaxis s'assembla molt a la de Ruby, però compila un codi natiu molt més eficient usant un backend LLVM, a costa d'excloure aspectes dinàmics

# Propòsit del llenguatge

- Crystal és un llenguatge de programació de **propòsit general**, és a dir, un llenguatge informàtic que s'aplica àmpliament a tots els dominis d'aplicació i no té característiques especialitzades per a un domini determinat, i els procediments, instruccions i estructures de dades dels quals estan dissenyats per a resoldre qualsevol tipus de problema.

# Paradigmes de programació que admet Crystal

- Crystal és un llenguatge de paradigma múltiple
- Paradigmes que admet:
  - Paradigma concurrent
  - Paradigma orientat a objecte (Object Oriented Programming, OOP)

# Sistema d'execució

- Crystal és un llenguatge de programació compilat
- Diferencia les etapes de compilació i execució

# Sistema de tipus

- Crystal és un LP de tipat fort

```
error in line 1
Error: no overload matches 'Int32#+' with type Char
```

- La comprovació de tipus de Crystal és mixt, si s'especifica el tipus, es coneix el tipus de variables en temps de compilació, en cas contrari, la inferència de tipus permet al programador no declarar el tipus i es coneixerà durant el temps d'execució

```ruby
class Person
  def initialize(@name)
    @age = 0
  end
end
```

# Opció 1: Amb restriccions de tipus

```
class Person
  @name : String
  @age : Int32
  def initialize(@name)
    @age = 0
  end
end
```

# Opció 2: Sense restriccions de tipus

```
class Person
  def initialize()
    @name = "John"
    @age = 0
  end
end
```

# Altres característiques particulars

- Crystal permet la utilització de macros

```
macro define_method(name, content)
  def {{name}}
    {{content}}
  end
end

# This generates:
#
#     def foo
#       1
#     end
define_method foo, puts 1

foo # => 1

define_method foo2, puts "HOLA"

foo2 # => "HOLA"
```

# Altres característiques particulars

- El nom d'una classe ha de començar per una lletra majúscula, sinó dona error de compilació

```
class Fibonacci
  def calculo_valor(n)
    if (n == 0)
      return 0
    elsif (n == 1)
      return 1
    else
      return calculo_valor(n - 1) + calculo_valor(n - 2)
    end
  end
end
```

# Exemples de codis il·lustratius

- Exemple 1: Hello world

```
class Greeter
  def initialize(@name : String )
  end

  def salute
    puts "Hello #{@name}!"
  end
end

g = Greeter.new("world")
g.salute
```

- Exemple 2: càlcul del nombre de Fibonacci

```ruby
class Fibonacci
  def calculo_valor(n)
    if (n == 0)
      return 0
    elsif (n == 1)
      return 1
    else
      return calculo_valor(n - 1) + calculo_valor(n - 2)
    end
  end
end

f = Fibonacci.new()
puts f.calculo_valor(2)  #1
puts f.calculo_valor(3)  #2
puts f.calculo_valor(4)  #3
puts f.calculo_valor(5)  #5
puts f.calculo_valor(6)  #8
```

- Exemple 3: valor absolut

```ruby
class AbsValue
  def calculo_valor(n)
    if (n > 0)
      return n
    else
      return n * -1
    end
  end
end

f = AbsValue.new()
puts f.calculo_valor(2)      #2
puts f.calculo_valor(-2)     #2
```

# Visió personal i crítica de Crystal

- Al ser un llenguatge compilar estàticament, és un llenguatge ràpid, que pot assolir velocitats d'execució properes a les de C.
- Respecte a C, té l'avantatge de poder utilitzar macros.

| Language | Version | Result(s) |
|---|---|---|
| C | gcc 4.9.2 | 3.575 |
| Crystal | v0.20.1 | 6.981 |
| Rust | 1.14.0 | 7.808 |
| Julia | v0.5.0 | 9.657 |
| LuaJIT | 2.1.0-beta2 | 10.150 |
| Go | go1.7.4 | 10.249 |
| Node.js | v7.3 | 15.122 |