Ràpid com C, elegant com Ruby

Hash: 4165

Història de *Crystal*

Història de *Crystal*

2011 - Ary Borenszweig, Juan Wajnerman i Brian Cardi crean *Crystal*

Història de *Crystal*

2011 - Ary Borenszweig, Juan Wajnerman i Brian Cardi crean *Crystal*

2013 - *Crystal* esdevé *self-hosting*

Història de *Crystal*

2011 - Ary Borenszweig, Juan Wajnerman i Brian Cardi crean *Crystal*

2013 - *Crystal* esdevé *self-hosting*

Juny 2014 - Llençament oficial de *Crystal*

Història de *Crystal*

2011 - Ary Borenszweig, Juan Wajnerman i Brian Cardi crean *Crystal*

2013 - *Crystal* esdevé *self-hosting*

Juny 2014 - Llençament oficial de *Crystal*

Juny 2016 - *Crystal* apareix al TIOBE Index

Propòsits del llenguatge

Propòsits del llenguatge

Sintaxi similar a *Ruby*

Propòsits del llenguatge

Sintaxi similar a *Ruby*

Rapidesa i eficiència com la de *C*

Propòsits del llenguatge

Sintaxi similar a *Ruby*

Rapidesa i eficiència com la de *C*

Orientació a objectes total

Propòsits del llenguatge

Sintaxi similar a *Ruby*

Rapidesa i eficiència com la de *C*

Orientació a objectes total

Tipat estàtic

Propòsits del llenguatge

Sintaxi similar a *Ruby*

Rapidesa i eficiència com la de *C*

Orientació a objectes total

Tipat estàtic

Compilació a codi natiu

Paradigmes de programació

Programació orientada a objectes

Polimorfisme

Herència

Classes poden ser *abstract*

Tres niveles de visibilitat:

- *Public*, per defecte
- *Private*
- *Protected*

Atributs i mètodes de classe

Paradigmes de programació

Programació orientada a objectes

Polimorfisme

Herència

Classes poden ser *abstract*

Tres niveles de visibilitat:

- *Public*, per defecte
- *Private*
- *Protected*

Atributs i mètodes de classe

Exemple:

```
1  abstract class Animal
2    abstract def talk
3  end
4
5  class Dog < Animal
6    def talk
7      "Woof!"
8    end
9  end
10
11 class Cat < Animal
12   def talk
13     "Miau"
14   end
15 end
16
17 class Person
18   getter pet
19
20   def initialize(@name : String, @pet : Animal)
21   end
22 end
23
24 john = Person.new "John", Dog.new
25 peter = Person.new "Peter", Cat.new
```

Paradigmes de programació

Imperatiu

Noció d’estat

Canvis d’estat

Paradigmes de programació

Imperatiu

Noció d'estat

Canvis d'estat

Funcional

Closures

Paradigmes de programació

Imperatiu

Noció d'estat

Canvis d'estat

Funcional

Closures

Exemple:

```ruby
def counter
  x = 0
  ->{ x += 1; x }
end

proc = counter
proc.call #=> 1
proc.call #=> 2
```

Sistema de tipus

Inferència de tipus

Permet no especificar el tipus

Tres regles principals:

- Assignació d'un tipus literal
- Assignació del resultat de crear una instancia d'una classe
- Assignació d'una variable que és argument d'un mètode amb un tipus definit

Sistema de tipus

Inferència de tipus

Permet no especificar el tipus

Tres regles principals:

- Assignació d'un tipus literal
- Assignació del resultat de crear una instancia d'una classe
- Assignació d'una variable que és argument d'un mètode amb un tipus definit

Union type

Una variable pot venir ser de diversos tipus

Sistema de tipus

Inferència de tipus

Permet no especificar el tipus

Tres regles principals:

- Assignació d'un tipus literal
- Assignació del resultat de crear una instancia d'una classe
- Assignació d'una variable que és argument d'un mètode amb un tipus definit

Union type

Una variable pot venir ser de diversos tipus

```
1  if 1 + 2 == 3
2    a = 1
3  else
4    a = "hello"
5  end
```

Sistema d’execució

Self-hosting

Compila a codi natiu

Sistema d’execució

Self-hosting

Compila a codi natiu

crystal run

```
$ crystal run hello_world.cr

Hello world!
```

Sistema d'execució

Self-hosting

Compila a codi natiu

crystal run

```
$ crystal run hello_world.cr

Hello world!
```

crystal build

```
$ crystal build hello_world.cr

$ ./hello_world

Hello world!
```

Principals aplicacions

NeuraLegion

Level UP Solutions

diploid

NIKOLA™ MOTOR COMPANY

blockvue

LPs similars