UERJ

CLOJURE

TAREFA-02

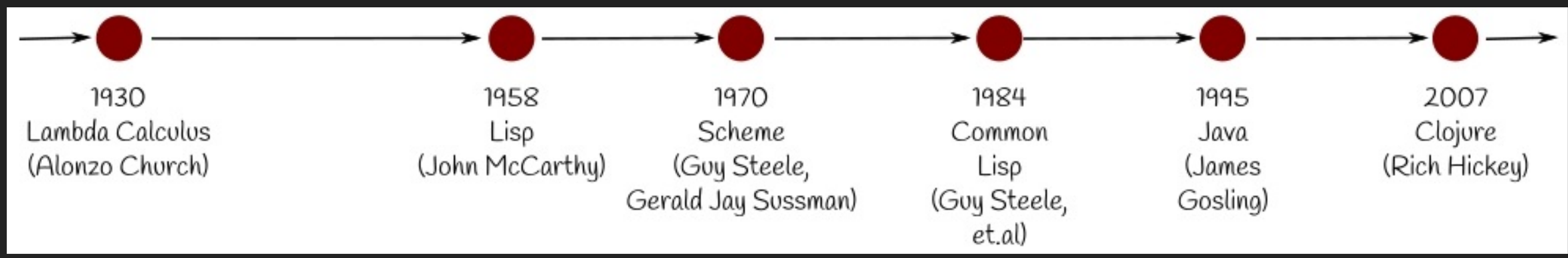
AGENDA

- INTRODUÇÃO
- CARACTERISTICAS E VANTAGENS
- HELLO WORLD
- EXEMPLOS
- COMPARAÇÃO COM PHP

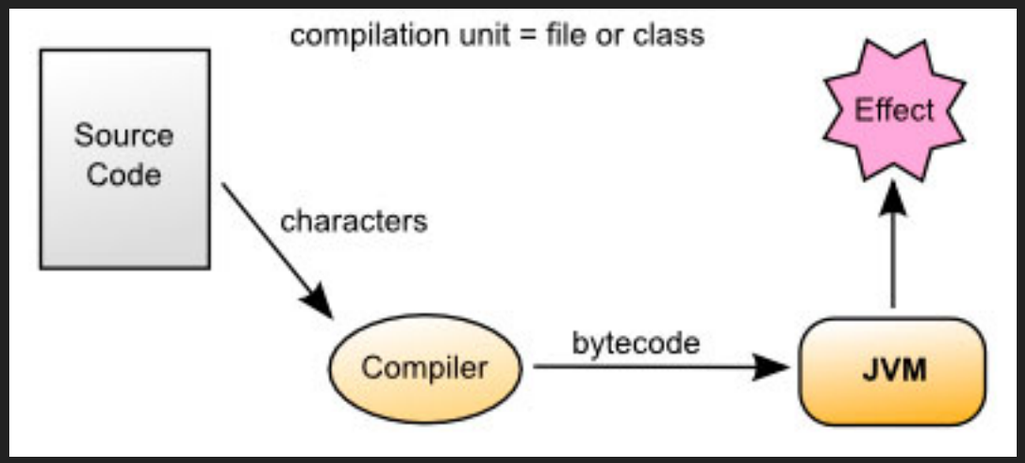
CLOJURE

- Clojure é uma linguagem baseada em programação funcional (LISP) que possui integração com boa parte das linguagens atuais , além de ser bastante utilizado no mercado profissional.

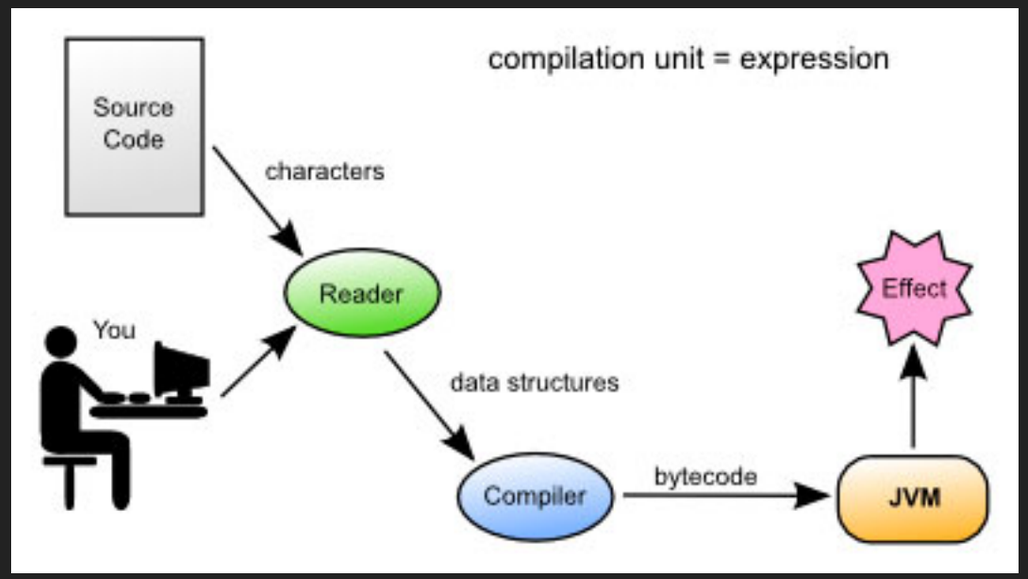
- A linguagem Clojure foi criada por Rich Hickey criada em 2007.Foi criado por que era necessária uma linguagem parecida com Lisp, que funcionasse como Programação Funcional, com uma plataforma estabelecida e Concorrência.



CARACTERISTICAS E VANTAGENS



- Compilação em JAVA



- Compilação em Clojure

- Clojure é uma linguagem compilada. Ela compila para bytecode JVM da mesma maneira que o Java faz.É baseada em recursão e consegue escrever código Java em Clojure.

EXEMPLO DO HELLO WORLD EM RUNTIME

```
Clojure 1.10.0
user=> (println "Olá Mundo!")
Olá Mundo!
nil
user=> (defn raiz-quadrada [x] (* x x))
#'user/raiz-quadrada
user=> (raiz-quadrada 4)
16
user=>
```

- MAP , FILTER E REDUCE

```
user=> (compare 10 20)
-1
user=> (map + [1 2 3] [4 5 6])
(5 7 9)
user=> (filter even? (range 10))
(0 2 4 6 8)
user=> (reduce + [1 2 3 4 5])
15
_
```

- OPÇÃO GRÁFICA - JAVA


```
user=> (javax.swing.JOptionPane/showMessageDialog nil "Olá Mundo" )
```

EM CLOJURE

- MACRO

```
Last login: Sun Apr 14 21:49:51 on ttys002
MacBook-Pro:~ BIRDRED$ clj
Clojure 1.10.0
(defn Example []
   (defmacro Simple [arg]
      (list 2 arg))
   (println (macroexpand '(Simple 2))))
#'user/Example
user=> (Example)
(2 2)
nil
user=>
```

EM CLOJURE

- MACRO x FUNÇÃO

```
user=> (defmacro twice [e] `(do ~e ~e))
#'user/twice
user=> (twice (println "Testando"))
Testando
Testando
nil
user=>
```

- MACRO

```
user=> (defn twice [e] `(do ~e ~e))
#'user/twice
user=> (twice (println "Testando"))
Testando
(do nil nil)
user=>
```

- FUNÇÃO

EM PHP

- MACRO
  - O mais próximo que PHP consegue utilizar com macro é a utilização do define
  - define (string $name , mixed $value [, bool $case_insensitive = FALSE]) : bool

EM CLOJURE

- PROTOCOLS

```
Clojure 1.10.0
(defprotocol P
  (foo [x])
  (bar-me [x] [x y]))
P
(deftype Foo [a b c]
  P
  (foo [x] a)
  (bar-me [x] b)
  (bar-me [x y] (+ c y)))
user.Foo
user=> (bar-me (Foo. 1 2 3) 42)
45
user=>
```

EM PHP

- PROTOCOLS

```php
<?php

// Declara a interface 'iTemplate'
interface iTemplate
{
    public function setVariable($name, $var);
    public function getHtml($template);
}

// Isso NÃO funcionará
// Fatal error: Class BadTemplate contains 1 abstract methods
// and must therefore be declared abstract (iTemplate::getHtml)
class BadTemplate implements iTemplate
{
    private $vars = array();

    public function setVariable($name, $var)
    {
        $this->vars[$name] = $var;
    }
}
?>

```

BIBLIOGRAFIA

- Clojure : [clojure.org]()
- Wikipedia Clojure: [https://pt.wikipedia.org/wiki/Clojure#Hist.C3.B3ria_e_processo_de_desenvolvimento]()
- Clojure Docs : [https://clojuredocs.org]()
- Artigo 'Devemos usar Clojure?' : [http://imasters.com.br/artigo/25335/linguagens/devemos-usar-clojure?trace=1519021197&source=single]()
- GrokPodCast 141 Clojure : [http://www.grokpodcast.com/2015/07/16/episodio-141-clojure/]()
- GrokPodCast 142 Clojure : [http://www.grokpodcast.com/2015/07/23/episodio-142-clojure/]()
- GrokPodCast 143 Clojure : [http://www.grokpodcast.com/2015/07/30/episodio-143-clojure/]()
- HipsterChat : [http://hipsters.tech/tecnologias-no-nubank-hipsters-01/]()
- Implementação : [https://www.php.net/manual/pt_BR/language.oop5.interfaces.php]()
- Exemplos de Macro (Macro x Função) : [http://blog.klipse.tech/clojure/2016/05/01/macro-tutorial-1.html]()
- Mais Exemplos de Macro : [http://clojure-doc.org/articles/language/macros.html]()
- Outros Exemplos de Macro :[https://aphyr.com/posts/305-clojure-from-the-ground-up-macros]()
- Exemplo das Funções :[https://stackoverflow.com/questions/3667403/what-is-the-difference-between-defn-and-defmacro]()

UERJ