

Groovy!

Anderson Ledo

Departamento de Sistemas e Computação - DSC
Universidade Federal de Campina Grande - UFCG
<http://andersonledo.com>



Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Paradigma Orientado a Objetos, Script

Desde 2003

Surgida da Java Community Process, JSR-241

Desenvolvida por Guillaume Laforge

Tipagem Dinâmica, forte e duck

Influenciada por Java, Python, Ruby, Perl, Smalltalk, Objective-C

Sistema Operacional Multiplataforma

Licença Apache License v2.0

Website <http://groovy.codehaus.org>

Atualmente mantida por: SpringSource

TIOBE 44

Groovy?

Em poucas palavras ...

Resumindo... Groovy é uma linguagem dinamicamente tipada, que gera bytecode para ser executado em uma JVM. Ela integra conceitos de outras linguagens como Python, Ruby, Smalltalk e Perl, tornando-se de fácil aprendizado, poderosa e flexível. Migrar de outras linguagens pra Groovy é fácil. Principalmente de Java.

Groovy?

Em poucas linhas ...

Começando com o HelloWorld

- Python

```
import sys
print "Hello, " + sys.argv[1] + "!"
```

- Java

```
class HelloWorld
  static public void main( String args[] )
    System.out.println( "Hello World!" );
```

Começando com o HelloWorld

```
println "Hello, ${args[0]} !"
```

```
> groovy HelloWorld.groovy Pessoal  
> Hello, Pessoal !
```

Começando com o HelloWorld

```
println "Hello, ${args[0]} !"
```

```
> groovy HelloWorld.groovy Pessoa  
> Hello, Pessoa !
```

Começando com o HelloWorld

```
class HelloWorld{
  def static main(def args){
    println "Hello, ${args[0]} !"
  }
}
```

```
class HelloWorld{
  public static void main(String[] args){
    println "Hello, ${args[0]} !"
  }
}
```

Começando com o HelloWorld

```
class HelloWorld{
  def static main(def args){
    println "Hello, ${args[0]} !"
  }
}
```

```
class HelloWorld{
  public static void main(String[] args){
    println "Hello, ${args[0]} !"
  }
}
```

Começando com o HelloWorld

```
class HelloWorld{
    public static void main(String[] args){
        System.out.println("Hello, ${args[0]} !");
    }
}
```

```
class HelloWorld{
    public static void main(String[] args){
        System.out.println("Hello, " + args[0] + "!");
    }
}
```

Começando com o HelloWorld

```
class HelloWorld{
    public static void main(String[] args){
        System.out.println("Hello, ${args[0]} !");
    }
}
```

```
class HelloWorld{
    public static void main(String[] args){
        System.out.println("Hello, " + args[0] + "!");
    }
}
```

Começando com o HelloWorld

```
class HelloWorld{
    public static void main(String[] args){
        StringBuffer sb = new StringBuffer("Hello, ")
        sb.append(args[0])
        sb.append(" !")
        System.out.println(sb.toString())
    }
}
```

Pode ser simples ou complexo

Depende do que você precisa...

Começando com o HelloWorld

```
class HelloWorld{
    public static void main(String[] args){
        StringBuffer sb = new StringBuffer("Hello, ")
        sb.append(args[0])
        sb.append(" !")
        System.out.println(sb.toString())
    }
}
```

Pode ser simples ou complexo

Depende do que você precisa...

Começando com o HelloWorld

```
class HelloWorld{
    public static void main(String[] args){
        StringBuffer sb = new StringBuffer("Hello, ")
        sb.append(args[0])
        sb.append(" !")
        System.out.println(sb.toString())
    }
}
```

Pode ser simples ou complexo

Depende do que você precisa...

Mais exemplos... (Closures)

```
square = { print ((it * it) + " ,") }  
[ 1, 2, 3, 4 ].collect(square)
```

Saída: 1, 4, 9, 16

Mais exemplos... (Closures)

```
//multiline string...  
def meuNome = "Anderson"  
def myString = ""  
Esta é a nossa apresentação sobre Groovy.  
Meu nome é ${meuNome}.  
;  
""  
println(meuNome[0..-1])
```

Saída: Anderson

Mais exemplos... (Closures)

```
//sobrecarga de operadores
class MinhaClasse{
String nome
def plus(MinhaClasse mc){ mc.nome + this.nome }
}
def mc1 = new MinhaClasse(nome:"João ")
def mc2 = new MinhaClasse(nome:"Da Silva")
println mc1 + mc2
println mc2 + mc1
```

Saída: "João Da Silva" "Da Silva João"

Mais exemples...

```
def text = "nice cheese gromit!"
```

```
def sub = text[5..10]  
assert sub == 'cheese'
```

```
def map = [name:"Gromit", likes:"cheese", id:1234]
```

```
assert map["name"] == "Gromit"  
assert map.name == "Gromit"
```

```
def list = [10, 11, 12]  
def answer = list[2]  
assert answer == 12
```

Versátil...

- GroovyServerPages(GSP)
- GroovyScience
- Google Data Support
- Groosh
- GSQL
- GParS
- E muitos outros módulos.

Frameworks

- Grails
 - Criar aplicações rapidamente (protótipos em minutos...).
 - Filosofia **Convention over Configuration** do Rails
 - Munda Java!
 - Provedores de persistência, plugins (segurança, visual, conteúdo, serviços etc.)
 - Comunidade cresce mais a cada dia...
- Gaelyk
 - Aplicações 'mais leves' para a Google Application Engine
 - Bem recente...
 - Promissor!
- Griffon
 - Segue a mesma filosofia do Grails, mas para aplicações Desktop.

Frameworks

- Grails
 - Criar aplicações rapidamente (protótipos em minutos. . .).
 - Filosofia **Convention over Configuration** do Rails
 - Munda Java!
 - Provedores de persistência, plugins (segurança, visual, conteúdo, serviços etc.)
 - Comunidade cresce mais a cada dia. . .
- Gaelyk
 - Aplicações 'mais leves' para a Google Application Engine
 - Bem recente. . .
 - Promissor!
- Griffon
 - Segue a mesma filosofia do Grails, mas para aplicações Desktop.

Frameworks

- Grails
 - Criar aplicações rapidamente (protótipos em minutos. . .).
 - Filosofia **Convention over Configuration** do Rails
 - Munda Java!
 - Provedores de persistência, plugins (segurança, visual, conteúdo, serviços etc.)
 - Comunidade cresce mais a cada dia. . .
- Gaelyk
 - Aplicações 'mais leves' para a Google Application Engine
 - Bem recente. . .
 - Promissor!
- Griffon
 - Segue a mesma filosofia do Grails, mas para aplicações Desktop.

Frameworks

- Grails
 - Criar aplicações rapidamente (protótipos em minutos. . .).
 - Filosofia **Convention over Configuration** do Rails
 - Munda Java!
 - Provedores de persistência, plugins (segurança, visual, conteúdo, serviços etc.)
 - Comunidade cresce mais a cada dia. . .
- Gaelyk
 - Aplicações 'mais leves' para a Google Application Engine
 - Bem recente. . .
 - Promissor!
- Griffon
 - Segue a mesma filosofia do Grails, mas para aplicações Desktop.

Mais considerações. . .

- Se você quer que as coisas sejam mais Groovy-like ou mais Java-like só depende de você.
- É uma linguagem fácil de se aprender. . .
- Mas, trabalhar com linguagens de script, como Groovy, requer uma certa experiência do programador.

Obrigado!

