Test de infraestructura “Teuton”

Agradecimientos

AgileCanarias, comunidad Teuton, Software Libre.

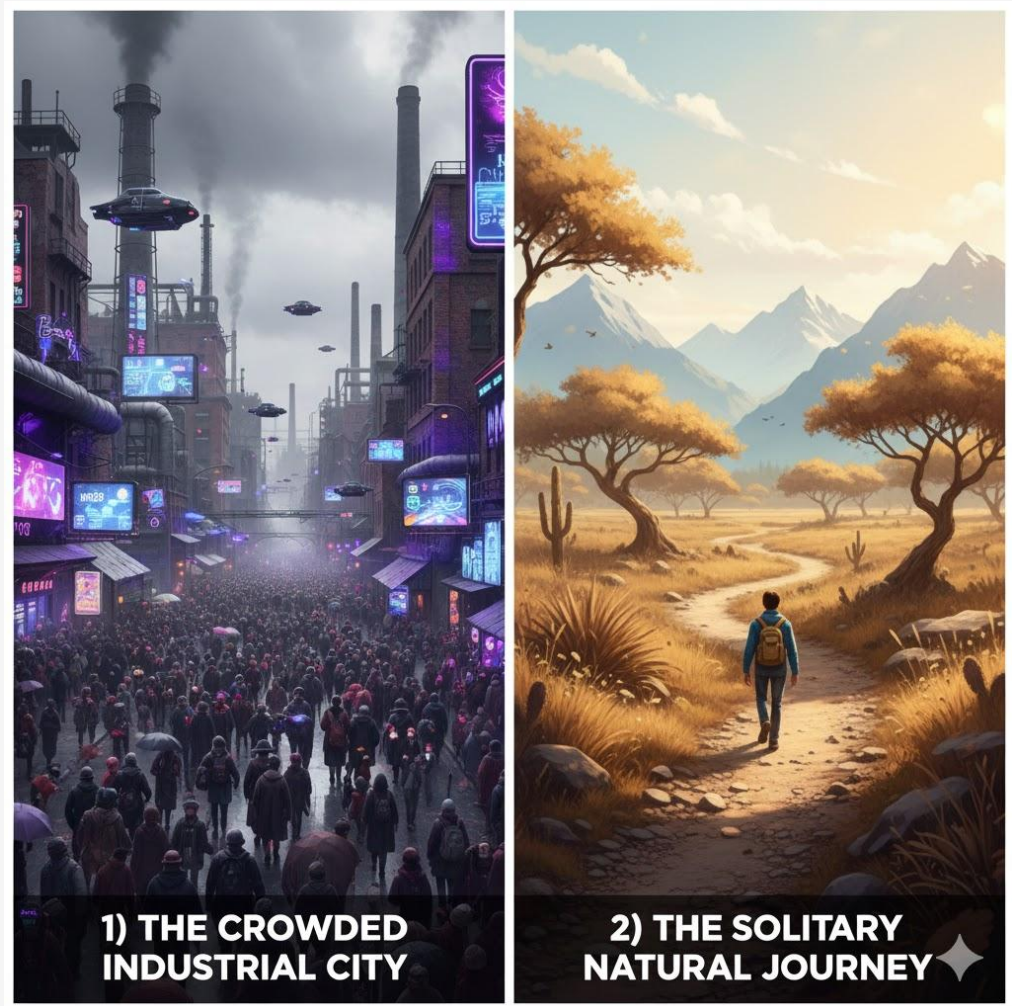
agilecanarias

Presentación

David Vargas (@dvarrui)

Ruby, OpenSUSE, software libre y Starwars

El camino



agilecanarias

Ruby
TDD
Teuton

agilecanarias

OpenSUSE

- GNU/Linux OpenSUSE, y la comunidad del Software Libre
- Ruby para scripting, YAST, etc.
- **Si te encuentras un problema y lo puedes solucionar... ¡adelante!**

Concurso de programación - PROGRAMAME

agilecanarias

Correcciones automáticas basadas en tests

🚦 TDD: Desarrollo guiado por pruebas de software.

🚦 Lograr código limpio que funcione.

🚦 Garantizar que el software cumple con los requisitos que se han establecido.

“Clean Code” (Robert C. Martin)

“Código limpio” (Carlos Ble)

Code testing - Example 1 - Recursive

```ruby
def factorial(n)
 if (n == 0 || n == 1)
   return 1
 else
   return n * factorial(n - 1)
 end
end
```

```ruby
require 'test/unit'
require_relative 'factorial_recursive'

class TestFactorial < Test::Unit::TestCase
 def test_base_case
   assert_equal(1, factorial(0))
   assert_equal(1, factorial(1))
 end

 def test_2_3_4
   assert_equal(2, factorial(2))
   assert_equal(6, factorial(3))
   assert_equal(24, factorial(4))
 end
end
```

Code testing - Example 1 - Recursive

- Los paréntesis son opcionales cuando son redundantes.
- El if admite la forma de lenguaje natural.
- Toda sentencia es una expresión. Esto es, devuelven algo. El último return sobra.

```
def factorial(n)
 return 1 if n == 0 || n == 1
 n * factorial(n - 1)
end
```

```
require 'test/unit'
require_relative 'factorial_recursive'

class TestFactorial < Test::Unit::TestCase
 def test_base_case
   assert_equal(1, factorial(0))
   assert_equal(1, factorial(1))
 end

 def test_2_3_4
   assert_equal(2, factorial(2))
   assert_equal(6, factorial(3))
   assert_equal(24, factorial(4))
 end
end
```

agilecanarias

RubyLover

- Tipado dinámico
- OO real (todo es un objeto) y de primera clase
- Metaprogramación
- “Parece” lenguaje natural (inglés).
- Legible y fácil de escribir.

Code testing - Example 2 - Object methods

```ruby
def factorial(n) = (1..n).to_a.reduce(:*)
```

- Una función de una sentencia se puede poner en una línea.
- (1..n) es azúcar sintáctico de Range.new(1,n). Es un objeto.
- Los métodos de los objetos se pueden encadenar. El método to_a devuelve un objeto Array
- El método reduce del Array reduce la lista usando el método indicado.
- Los simbolos existen en muchos lenguajes.

agilecanarias

```ruby
require 'test/unit'
require_relative 'factorial_recursive'

class TestFactorial < Test::Unit::TestCase
 def test_base_case
   # assert_equal(1, factorial(0))
   assert_equal(1, factorial(1))
 end

 def test_2_3_4
   assert_equal(2, factorial(2))
   assert_equal(6, factorial(3))
   assert_equal(24, factorial(4))
 end
end
```

Code testing - Example 2 - Object methods

- Esto es lo mismo de antes pero poniendo los métodos de forma más “bonita” o “estilo funcional”.
- reduce devuelve un valor que es el return de la funciñon.

```
def factorial(n)
  (1..n)
    .to_a
    .reduce(:*)
end
```

```
require 'test/unit'
require_relative 'factorial_recursive'

class TestFactorial < Test::Unit::TestCase
 def test_base_case
   # assert_equal(1, factorial(0))
   assert_equal(1, factorial(1))
 end

 def test_2_3_4
   assert_equal(2, factorial(2))
   assert_equal(6, factorial(3))
   assert_equal(24, factorial(4))
 end
end
```

agilecanarias

EL CICLO DE LA EVALUACIÓN

FIN. HASTA LA PRÓXIMA.

agilecanarias

Problema

- Reducir las tareas repetitivas y aburridas
- DevOps: Procesos reproducibles y automatizados
- Jugar es divertido (Gamificación)

agilecanarias

Idea

- Ruby: programar un nuevo proyecto
- Aplicar buenas prácticas al proceso
- Herramienta para facilitar las tareas
- ¡¡¡Premio asegurado!!!

agilecanarias

Principios

- TDD (Test-Driven Development)
- DRY (Don’t repeat yourself)
- DevOps (Automatización de procesos)
- RubyLover!

Teuton: Caso de uso

✅ ¿Qué es Teuton?

- Programa multiplataforma.
- Licencia Software libre.
- Test de infraestructura.

🔔 ¿Qué resuelve?

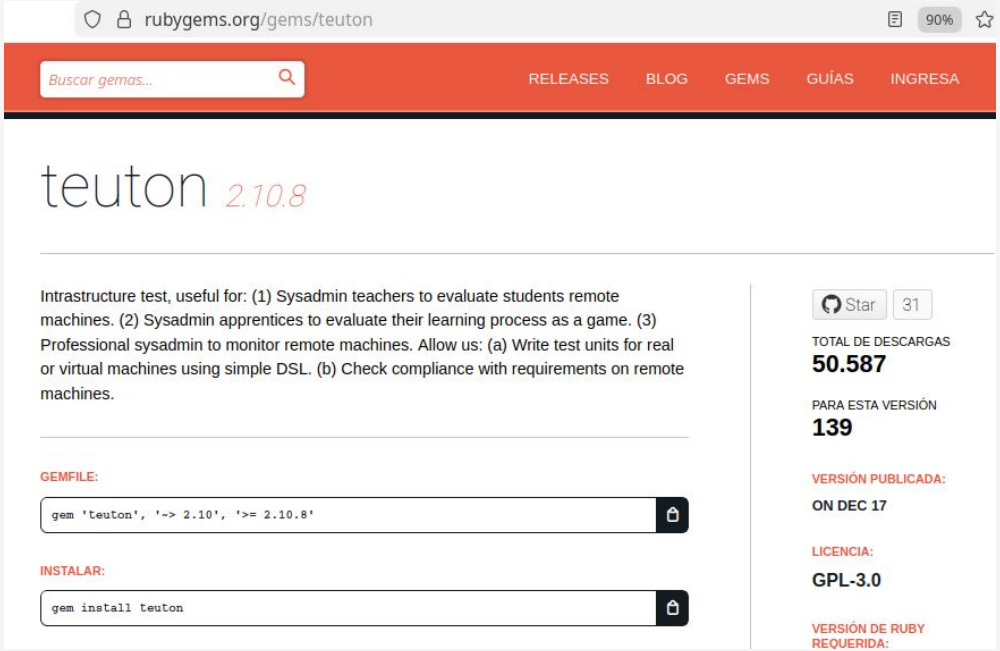
1. Revisar nuestra infraestructura como si fuera código.
2. Automatizar las correcciones de las MV remotas de los alumnos.

RubyGems - “gem install teuton”

Aplicar tests para mantener la calidad de...

- El código.
- ¿La infraestructura?
- ¿Los scripts?
- ¿Configuraciones?
- ¿Entornos?

¡Cualquier entorno CLI!



agilecanarias

Instalación - CLI

- Instalar Ruby
- gem install teuton, instalar la gema

```
$ teuton help
Commands:
  teuton [run] [OPTIONS] DIRECTORY   # Run test from directory
  teuton check [OPTIONS] DIRECTORY   # Check test and config file content
  teuton config [OPTIONS] DIRECTORY  # Suggest configuration.
  teuton help [COMMAND]              # Describe available commands or one specific command
  teuton new DIRECTORY               # Create skeleton for a new project
  teuton readme DIRECTORY            # Show README extracted from test contents
  teuton version                     # Show the program version
```

EVALUACIÓN AUTOMÁTICA CON TEUTON

1. ¡A PROGRAMAR!

2. ¡AYUDANDO EN CLASE!

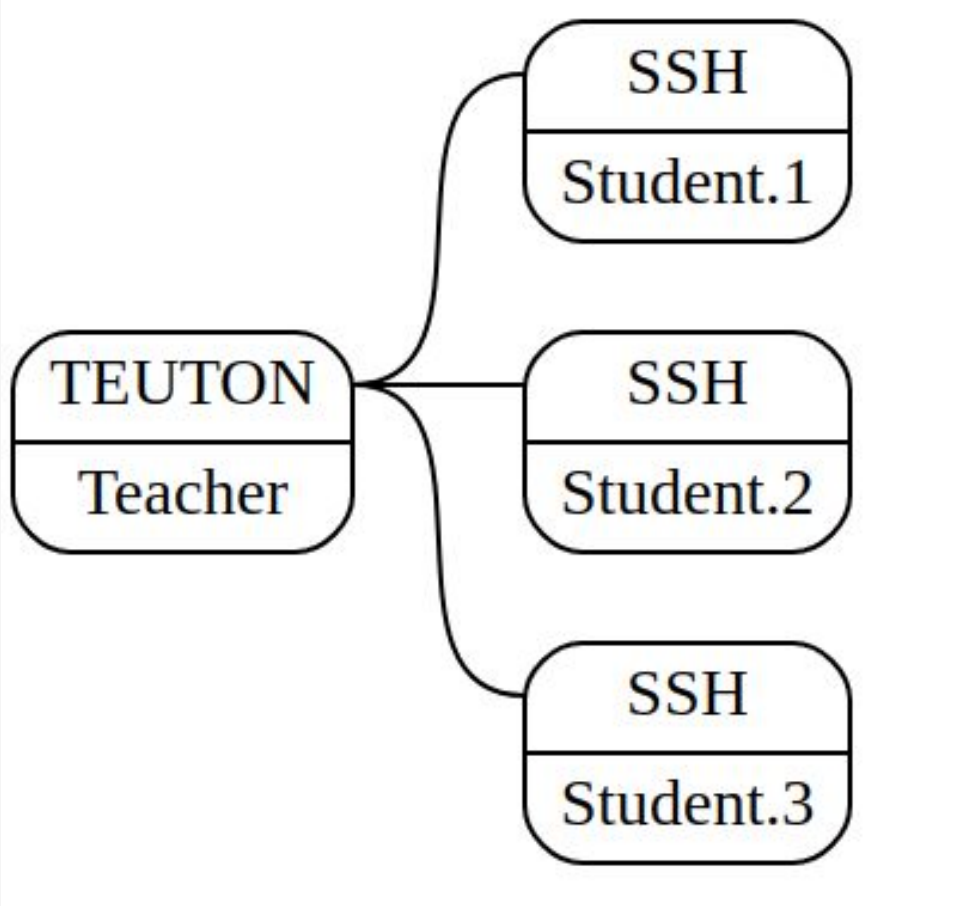
3. ¡TEUTON AL RESCATE!

¡EVALUACIÓN RÁPIDA Y JUSTA!

Entorno de trabajo

Un test consiste de:

- **config.yaml**: Fichero de configuración.
- **start.rb**: Test (target, run ,expect, etc.)



agilecanarias

agilecanarias

RubyLover

- Tipado dinámico
- POO (todo es un objeto). “Ciudadanos de primera clase”
- Metaprogramación
- Escribir de forma “natural” y leer de forma “natural”.
- ¡Programar poesía!

agilecanarias

DSL (Domain Specific Language)

Beneficios:

- Creado para resolver problemas en un campo concreto.
- Permite a los expertos del dominio crear y entender código (modelar problemas del dominio) más fácilmente
- Ejemplos: SQL, CSS, el álgebra en matemáticas. Mayor expresividad y concisión para su dominio específico.

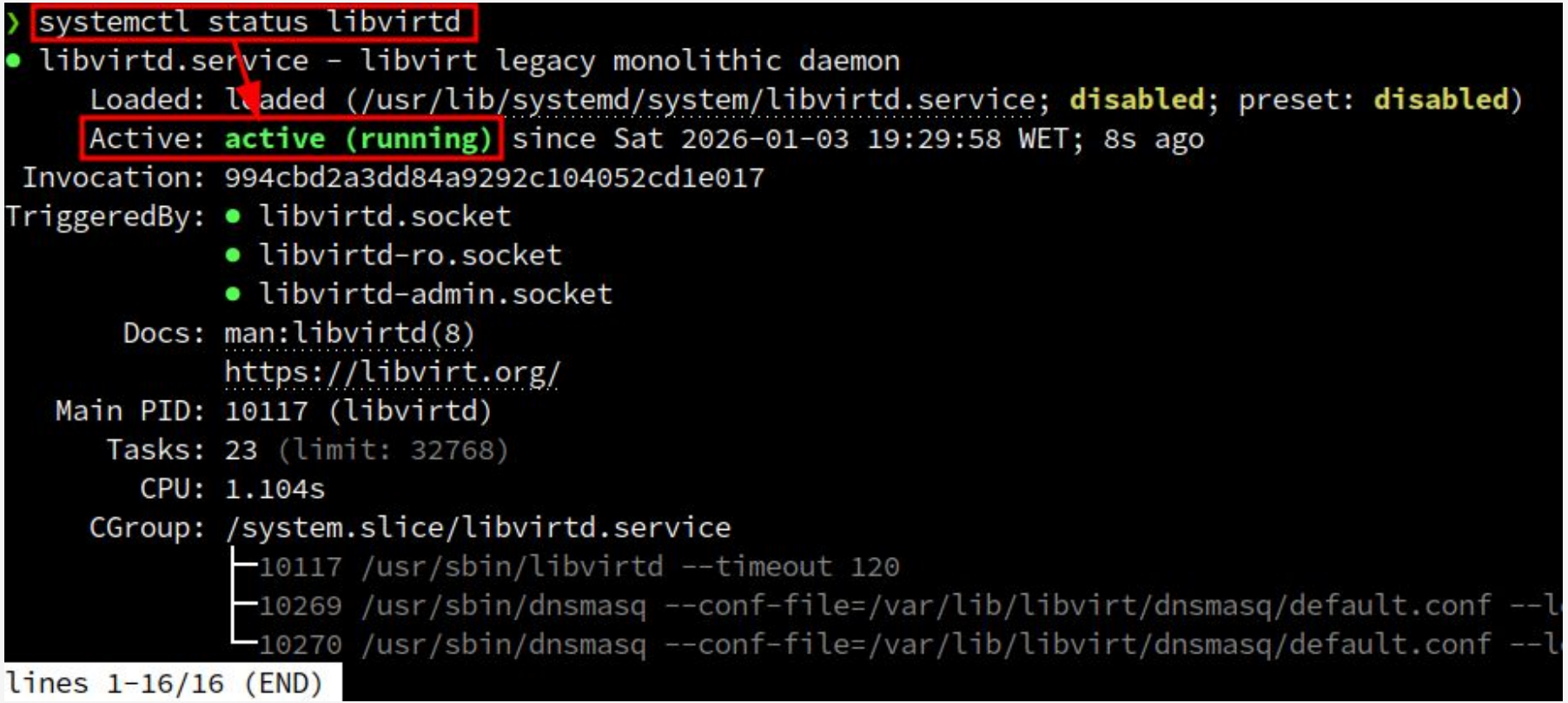
¿Y si creamos un DSL para resolver nuestro problema?

DSL de Teuton

- **target**: Descripción del objetivo
- **run**: Comando que ejecutamos para comprobar.
- **expect:** Lo que esperamos “ver” en la salida del comando anterior.

```
group "Learn about targets" do
 target "Create user obiwan"
 run "id obiwan"
 expect ["uid=", "(obiwan)", "gid="]

 target "Delete user vader"
 run "id vader"
 expect_fail
end
```



1. Si tienes el comando para comprobar el target
2. Y sabes lo que se debe "mirar" en la salida. **¡Ya tienes el test!**

Ejemplos de tests

Repos de ejemplos:

Ejemplos documentados:

- https://github.com/teuton-software/teuton/examples

Repo de ejemplos:

- https://github.com/dvarrui/teuton-tests

Tutorial: Crear un test para una práctica Nginx apoyándonos en la IA.

- https://github.com/dvarrui/charlas/blob/main/teuton/nginx/docs/index.md

```yaml
# File: config.yaml
# Desc: configutarion file
---
global:
 host1_username: root
cases:
- tt_members: student_1
 host1_ip: 192.168.1.201
 host1_password: secret_1
- tt_members: student_2
 host1_ip: 192.168.1.202
 host1_password: secret_2
- tt_members: student_3
 host1_ip: 127.0.0.1
 host1_password: secret_3
```

```ruby
# File: start.rb
# Desc: test definitions
group "Remote host" do
 target "Create user root"
 run "id root", on: :host1
 expect ["uid=", "(root)", "gid="]

 target "Delete user vader"
 run "id vader", on: :host1
 expect_fail
end

play do
 show
 export
end
```

```
expect ["obiwan", "kenobi"]
expect_first "episode"
expect_last "the end"
expect_one "rogue"
expect_one ["obiwan","kenobi"]
expect /Obiwan|obi-wan/

expect_none "vader"
expect_none ["darth", "vader"]
expect_none
expect_nothing

expect_ok
expect_fail
expect_exit NUMBER
expect_exit MIN..MAX
```

```
expect_sequence do
  find "A"
  find "B"
  find "C"
end

expect_sequence do
  find "A"
  next_to "B"
  next_to "C"
end

expect_sequence do
  find "A"
  next_to "B"
  ignore 2
  next_to "C"
end
```

DEMO!

nginx/v03.custom

Future roadmap

1. Documentación
2. Difusión
3. Feedback
4. teuton-server
5. Orientar a monitorización

Proyectos y Repositorios

github.com/teuton-software/teuton
github.com/teuton-software/teuton-get
github.com/dvarrui/teuton-tests

Ahora, pregunto yo! ;-)

github: @dvarrui
telegram: @dvarrui
email: dvarrui@proton.me

agilecanarias

¡Muchas gracias!