Práctica 1 b: Intro al paradigma, el lenguaje y sus herramientas

Tomás Felipe Melli

April 4, 2025

Índice

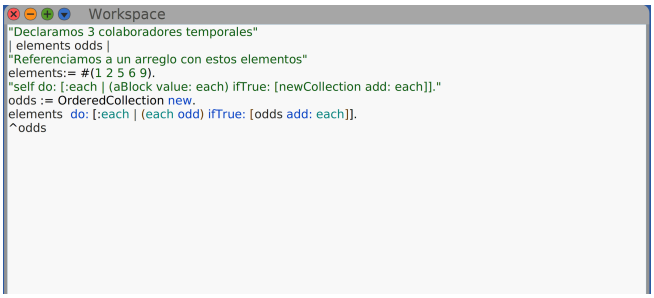
1 Ejercicio 0 : Debugger

2 Ejercicio 1 : Colecciones

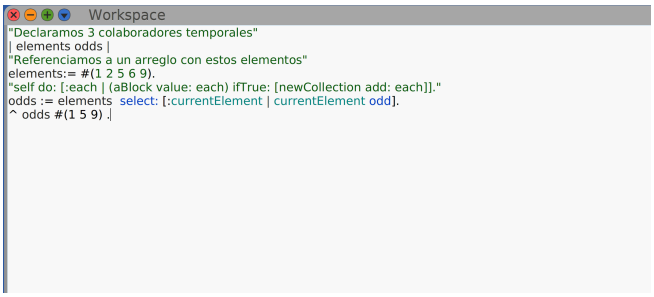
```
"Un array tiene fixed length. Los podemos definir con :"
x := Array with: 5 with: 4 with: 3 with: 2. #(5 4 3 2) .

"Tambie n se puede utilizar la sintaxis reducida"
y := #(5 4 3 2). #(5 4 3 2) .

"Cambiar el elemento en la primera pos por el 42"
x at: 1 put: 42. x #(42 4 3 2) .

"que  pasa si queremos poner un elemento en una posicio n que no existe ?"
x at: 5 put: 4.

"Ordered Collections"
z := OrderedCollection with:4 with:3 with:2 with:1. z add: 42. z add: 2. z an OrderedCollection(4 3 2 1 42
    2).
z size 6.

"y _ x occurrencesOf:                          number of times object in collection"
z occurrencesOf: 2 2 .

"Sets"
m := Set with: 4 with: 3 with: 2 with: 1.  a Set(4 3 2 1) .

"x add: 4; add: 3; add: 1; add: 2; yourself.     add element to collection"
m add: 42 42 .
m add: 2 2 .
m size  5 .
m occurrencesOf: 2 1 .
"En el set los ordena de mayor a menor"
m  a Set(42 4 3 2 1) .

"Dictionary"
d := Dictionary new a Dictionary() .

"x add: #a->4; add: #b->3; add: #c->1; add: #d->2; yourself.     add element to collection"
d add: #a->4; add: #b->3; add: #c->1; add: #d->2; yourself  a Dictionary(#b->3 #a->4 #d->2 #c->1 ) .

d add: #e->42 #e -> 42 .
d a Dictionary(#b->3 #a->4 #e->42 #d->2 #c->1 ) .

d size  5 .
d keys  #(#b #a #e #d #c) .
d values  #(3 4 42 2 1) .
d at: #a 4 .
d at: #z ifAbsent: [24] 24 .
d at: #a ifAbsent: ['No existe']  4 .

"Conversio n a colecciones"
x asOrderedCollection  an OrderedCollection(42 4 3 2) .

x asSet  a Set(42 4 3 2) .
```

2.1 1.3

2.2 1.8

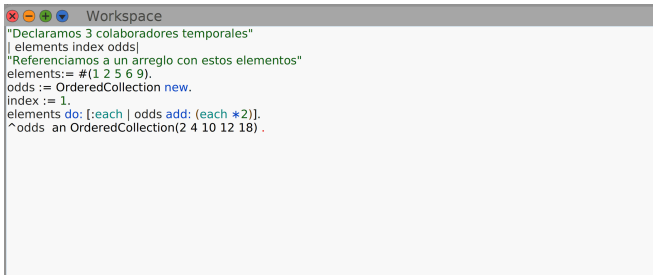


2.3 1.9

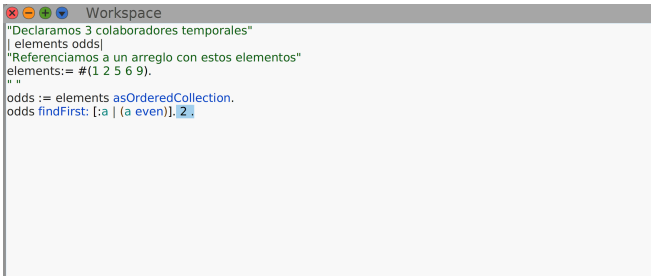


2.4 1.10

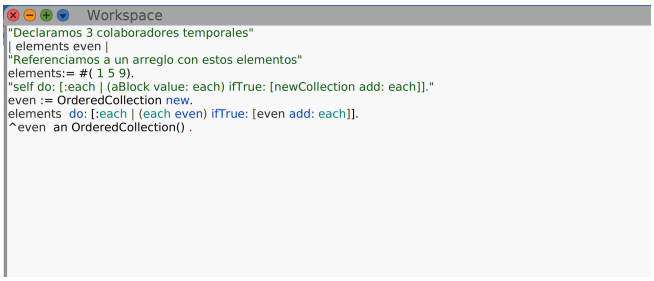
2.5 1.11



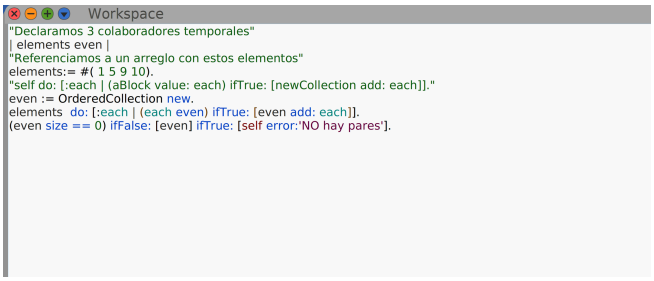
2.6 1.13



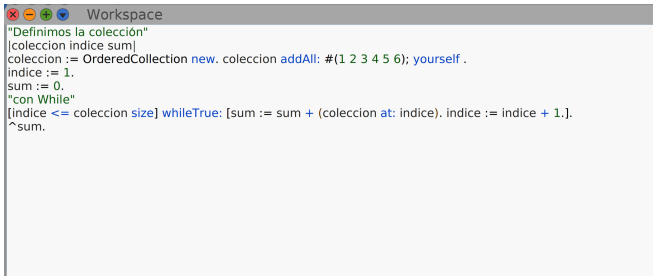
2.7 1.14



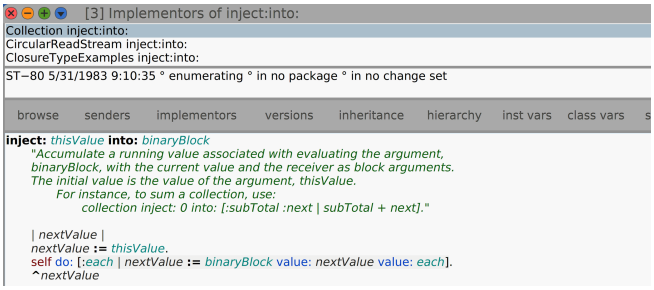
2.8 1.15



2.9 1.16



2.10 1.17

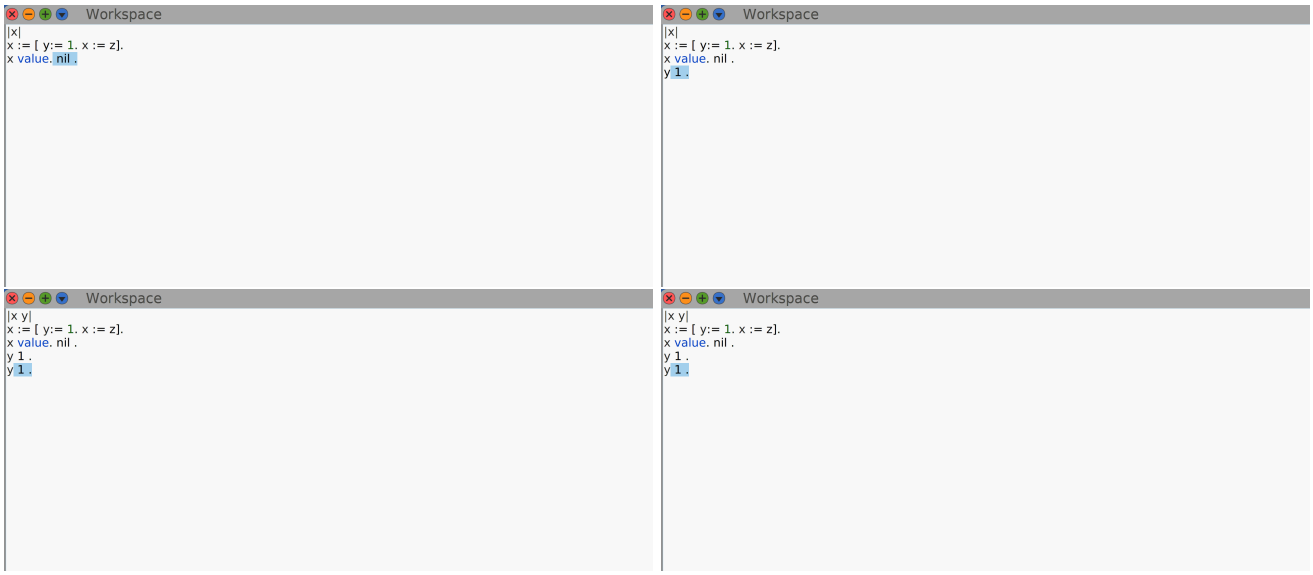


2.11 1.18

3 Ejercicio 2 : Bloques

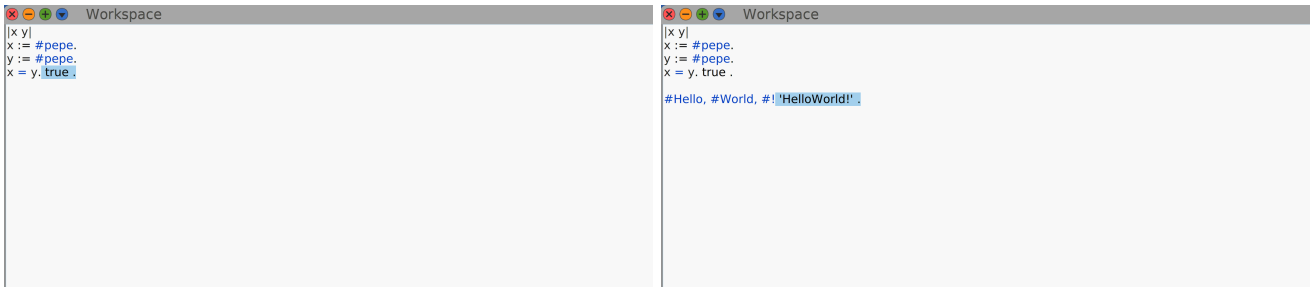
Un **Block** en Smalltalk es una forma de agrupar y encapsular una serie de instrucciones que pueden ser evaluadas o ejecutadas en un contexto específico.
Un block es similar a lo que en otros lenguajes de programación se podría llamar una "función anónima" o "closure". Es una secuencia de código que no se ejecuta inmediatamente, sino que se puede almacenar y luego ejecutar en otro contexto cuando sea necesario. Los blocks pueden tomar argumentos, y pueden ser asignados a variables, pasados como parámetros a otros métodos, o devueltos como valores de un método.
Ejemplo :

```
1 [ :x | x * 2 ]
```



4 Ejercicio 3 : Símbolos

Un **symbol** en Smalltalk es un objeto que consta de un nombre que se utiliza principalmente como identificador en el código, y es inmutable. Es utilizado frecuentemente en situaciones donde se necesita un nombre único para referirse a algo, como en la representación de nombres de variables, métodos o claves en colecciones.
A diferencia de las cadenas de texto, los symbols son únicos e inmutables. Esto significa que dos símbolos con el mismo nombre son en realidad el mismo objeto en memoria. En cambio, las cadenas de texto, aunque puedan tener el mismo contenido, son objetos diferentes.



5 Ejercicio 4 : Medidas

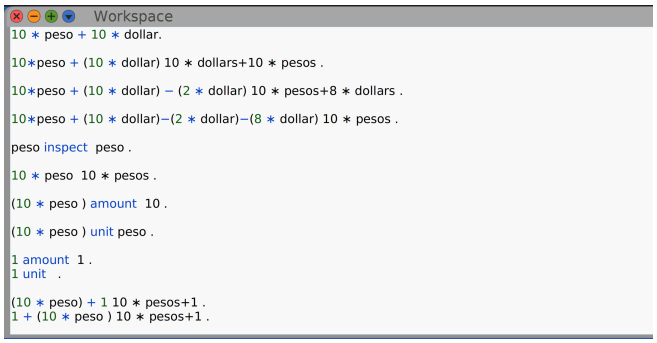
La relación entre estos tres eventos radica en que todos ellos fueron el resultado de errores en el manejo de unidades de medida y conversiones. A continuación, algunas similitudes clave:

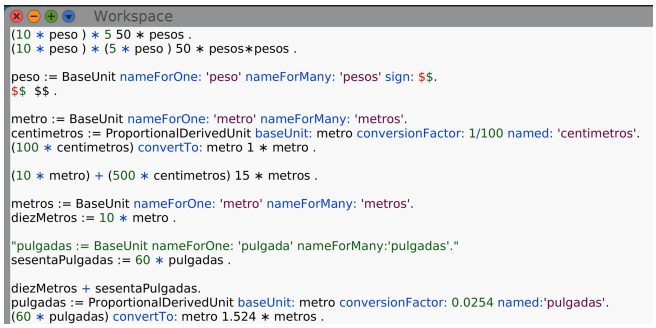
- Errores en la conversión de unidades o en el cálculo de medidas
  - En el Mars Orbiter, hubo un error al convertir entre el sistema métrico y el imperial.
  - En el Gimli Glider, el error se debió a una conversión incorrecta entre libras y kilogramos.

  - En el Ariane 5, el problema estuvo relacionado con la precisión de los cálculos y el manejo de valores numéricos, aunque no directamente con un error de conversión de unidades, fue un fallo en cómo se trataron las medidas en el software.

- Impacto de no verificar adecuadamente los sistemas y cálculos : En todos estos casos, hubo una falta de validación adecuada de los cálculos y conversiones de unidades antes de ejecutar las operaciones críticas. Si se hubiera revisado de manera más exhaustiva, los errores podrían haberse evitado.

- Consecuencias graves debido a la falta de atención al detalle: Los tres eventos muestran cómo pequeños detalles, como un error en la conversión de unidades o una falla en el manejo de los cálculos, pueden tener consecuencias desastrosas para misiones importantes y sistemas complejos.






```
diezMetros + sesentaPulgadas   60 * pulgadas+10 * metros .

"definir los bitcoins"
bitcoin := BaseUnit nameForOne: 'bitcoin' nameForMany:'bitcoins'.


"kelvin y farenheit"
```

6 Ejercicio 5 : Fechas