Paradigmas de Lenguajes de Programación

Paradigma de Objetos: Taller con Smalltalk

Departamento de Computación, FCEyN, UBA

Segundo Cuatrimestre 2017

Objetivos del día

1. Conocer el entorno Pharo
2. Reforzar lo que sabemos de objetos
   - envío de mensajes
   - mensajes de instancia vs. mensajes de clase
   - polimorfismo
3. Metodología para progamar usando tests
4. Algunas sugerencias de diseño

Machete

Sintaxis

- "Comentarios"
- | var1 var2 ...|
- [:arg1 :arg2 | | var1 var2 | expresión]
- expresión1. expresión2. expresión3
- objeto mensaje
- objeto msj1; msj2
- var := expresión
- ^ expresión

Literales

- 123
- 123.4
- $c
- 'palabra'
- #símbolo
- #(123 123.3 $a 'abc' #abc)

Palabras Reservadas

- self
- super
- nil
- true
- false
- thisContext

Conociendo Pharo

Algunas herramientas básicas:

- **Playground/Workspace** para interactuar con el sistema
- **Transcript** para registrar lo que pasa
- **System browser** para navegar las clases definidas
- **Inspector** para inspeccionar un objeto
- **Debugger** para ver en qué le pifiamos

Conociendo Pharo

Algunas herramientas básicas:

- **Playground/Workspace** para interactuar con el sistema
- **Transcript** para registrar lo que pasa
- **System browser** para navegar las clases definidas
- **Inspector** para inspeccionar un objeto
- **Debugger** para ver en qué le pifiamos

Tip: Con `shift+enter` pueden navegar más rápido

Precalentando...

1. Definir (en el playground) un closure que reciba un parámetro n y nos diga la fecha dentro de n días.

Precalentando...

1. Definir (en el playground) un closure que reciba un parámetro *n* y nos diga la fecha dentro de *n* días.
2. Implementar una forma sencilla de verificar si un día cualquiera es fin de semana. Para ello, un objeto de la clase Date deberá responder al mensaje "esFinde" que devuelve true o false.

Cada loco con su Pharo

Pharo es un sistema base que nosotros modificamos

- No hay diferencia entre el lenguaje y nuestro código
- Podemos modificar cualquier clase del sistema
- La imagen contiene el estado completo del sistema en un instante

Cada loco con su Pharo

Pharo es un sistema base que nosotros modificamos

- No hay diferencia entre el lenguaje y nuestro código
- Podemos modificar cualquier clase del sistema
- La imagen contiene el estado completo del sistema en un instante

Este estado incluye todos los objetos existentes, incluyendo clases y métodos que hayamos definido... ¡guarden la imagen!

Tip: Tengan a mano el `alt+.` si se les cuelga la imagen

Ejercicio: Polinomios

Modelar polinomios de una variable en Pharo.

Ejercicio: Polinomios

Modelar polinomios de una variable en Pharo.

Para atacar la hoja en blanco vamos a hacer una versión light de **TDD**.

Ejercicio: Polinomios

Modelar polinomios de una variable en Pharo.

Para atacar la hoja en blanco vamos a hacer una versión light de **TDD**.

Las ideas principales:

1. Pensar una funcionalidad chica que nos esté faltando
2. Escribir un test que describa la funcionalidad
3. Hacer pasar el test
4. Refactorizar si hace falta
5. Volver al paso 1

Ejercicio: Polinomios

Modelar polinomios de una variable en Pharo.

Para atacar la hoja en blanco vamos a hacer una versión light de **TDD**.

Las ideas principales:

1. Pensar una funcionalidad chica que nos esté faltando
2. Escribir un test que describa la funcionalidad
3. Hacer pasar el test
4. Refactorizar si hace falta
5. Volver al paso 1

¿Cómo llamo a los Test?

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Hacerse las siguientes preguntas puede ayudar a decidir:

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Hacerse las siguientes preguntas puede ayudar a decidir:

- ¿En mi cabeza representan formas más concretas de un concepto más general?

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Hacerse las siguientes preguntas puede ayudar a decidir:

- ¿En mi cabeza representan formas más concretas de un concepto más general?
- ¿Se los usa de forma intercambiable en alguna situación?

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Hacerse las siguientes preguntas puede ayudar a decidir:

- ¿En mi cabeza representan formas más concretas de un concepto más general?
- ¿Se los usa de forma intercambiable en alguna situación?
- ¿Me restringe mucho hacerlo?

Subclasificación

Variable, **Constante**, **Suma** y **Producto** tienen pinta parecida... ¿tiene sentido que tengan una superclase en común?

Hacerse las siguientes preguntas puede ayudar a decidir:

- ¿En mi cabeza representan formas más concretas de un concepto más general?
- ¿Se los usa de forma intercambiable en alguna situación?
- ¿Me restringe mucho hacerlo?
- ¿Es necesario?

Versionado

¡Funciona! ¿Cómo compartimos esta maravilla?

Hay varias formas:

Versionado

¡Funciona! ¿Cómo compartimos esta maravilla?

Hay varias formas:

- Exportar el código y mandar el `.st` por mail

Versionado

¡Funciona! ¿Cómo compartimos esta maravilla?

Hay varias formas:

- Exportar el código y mandar el `.st` por mail
- Commitear los `.st` en un SCM tradicional (git, svn, etc.)

Versionado

¡Funciona! ¿Cómo compartimos esta maravilla?

Hay varias formas:

- Exportar el código y mandar el `.st` por mail
- Commitear los `.st` en un SCM tradicional (git, svn, etc.)
- Control de versiones objetoso: **Monticello**

¿Cómo seguimos?

Nos gustaría que se puedan mostrar los polinomios

¿Cómo seguimos?

Nos gustaría que se puedan mostrar los polinomios

Pensar... ¿Hay algo repetido entre la implementación de `evaluarCon:` y `mostrar`?

¿Cómo seguimos?

Nos gustaría que se puedan mostrar los polinomios

Pensar... ¿Hay algo repetido entre la implementación de `evaluarCon:` y `mostrar`?

¡La forma de recorrer la estructura!

- Desacoplar estas responsabilidades es un típico problema de diseño.
- Muestra gratis de cómo se tratan estas cosas en Ingeniería 2
- Si quieren la posta, cursen las optativas de objetos.

FIN