fiuba

algo3

Resolución de problemas con objetos

Pablo Suárez
psuarez@fi.uba.ar

Resolución de problemas con objetos

Temario

Objetos

Mensajes

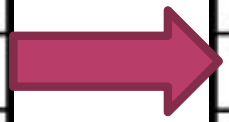
Delegación

Comportamiento

Estado y encapsulamiento

Un problema

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9



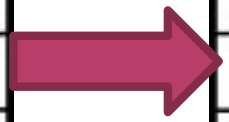
5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

fiuba algo3

Su turno

¡Acotar el problema a un escenario único!

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9



5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

¿Puedo colocar el 7 en la fila 2, columna 3?

fiuba algo3

POO vs. Estructurada

- Tanto la programación estructurada como la POO atacan el problema por partes
- Programación estructurada busca las partes en la propia solución del problema
  - Funciones, procedimientos: acciones que el programa debe realizar
- POO busca las partes en las entidades que surgen del dominio del problema en sí mismo
  - Entidades = Objetos

Resolviendo con objetos

- Encontrar entidades del dominio del problema
  - Van a ser nuestros “objetos”
- Descubrir / establecer cómo interactúan esas entidades para resolver el problema
  - Buscamos los “mensajes” que los objetos se envían, en qué orden y bajo qué condiciones lo hacen
- Determinar cómo hacen esos objetos para responder a los mensajes
  - “Comportamiento” de los objetos

Escenario: ver si podemos colocar un número en una celda

Encontrar objetos

Determinar cómo deben interactuar los objetos

Implementar el comportamiento de los objetos

Encontrar objetos

"establecer si el número 7 se puede colocar en la celda ubicada en la fila 2 y la columna 3"

Objetos "de dominio":

- El número 7
- La celda 2 3
- La fila 2
- La columna 3
- La caja que contiene la celda 2 3

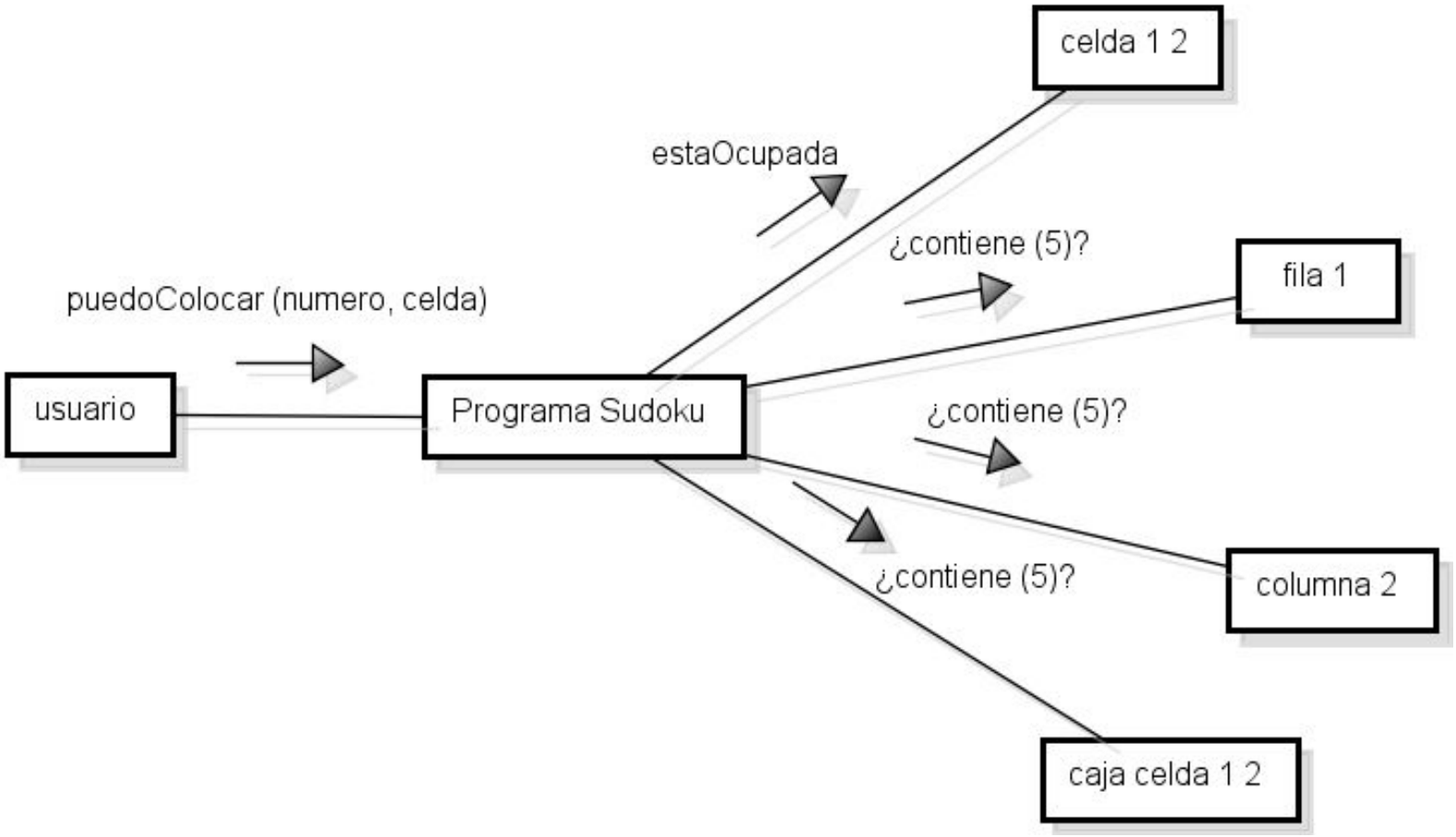
Determinar cómo deben interactuar los objetos (1)

Fijarse

- Si ya hay un **número** en la **celda** (si está ocupada)
- Si el **número** ya está en la **fila** en la que está la **celda** en cuestión
- Si el **número** ya está en la **columna** en la que está la **celda** en cuestión
- Si el **número** ya está en la **caja** en la que está la **celda** en cuestión

Si no está en ninguno de las tres, podemos colocarlo en la celda. Si no, no.

Determinar cómo deben interactuar los objetos (2)



Determinar cómo deben interactuar los objetos (3)

- Resolución: objeto “usuario” le envía un mensaje al programa
  - puedo colocar (7, celda23)?
- El objeto “Programa Sudoku” envía cuatro mensajes a cuatro objetos diferentes
  - Uno para ver si la celda está libre o no
  - Otros tres para saber si el número 7 ya se encuentra en la fila, columna o caja correspondiente
- Ahora, el objeto “Programa Sudoku” puede responderle al objeto “usuario” si se puede o no colocar el número 7 allí

Definiciones preliminares

Objeto

- Una entidad que existe en tiempo de ejecución y que puede enviar y recibir mensajes

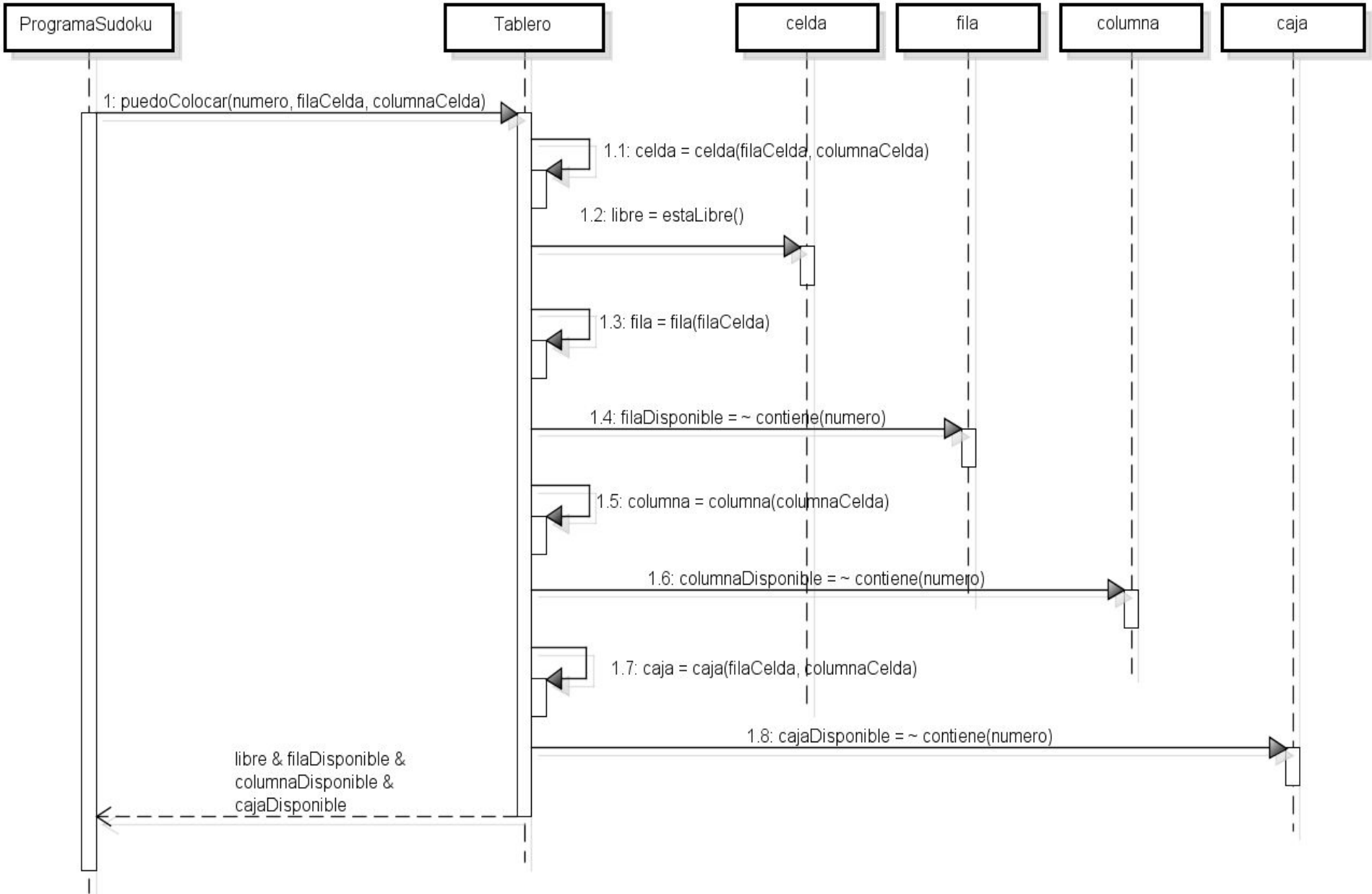
Delegación

- Cuando un objeto, para responder un mensaje, envía mensajes a otros objetos, decimos que delega ese comportamiento en otros objetos

Comportamiento

- Las posibles respuestas a los mensajes recibidos por un objeto

Delegación y diagrama de secuencia



Primeros usos de UML: diagramas de interacción

- Diagrama de comunicación
  - Muestra mejor la colaboración de los objetos
  - Mayor libertad para pensar opciones
- Diagrama de secuencia
  - Muestra el orden en que se deben pasar los mensajes
  - Tiende a que pensemos secuencialmente
- ¿Cuál usamos?
  - Secuencia, por habitualidad

Programa OO

Conjunto de objetos enviando mensajes a otros objetos

Los objetos receptores reciben los mensajes y reaccionan

- Haciendo algo (comportamiento)
- Devolviendo un valor (que depende de su estado)

Los mensajes pueden implicar la creación de nuevos objetos

El comportamiento puede delegarse a su vez en otro objeto

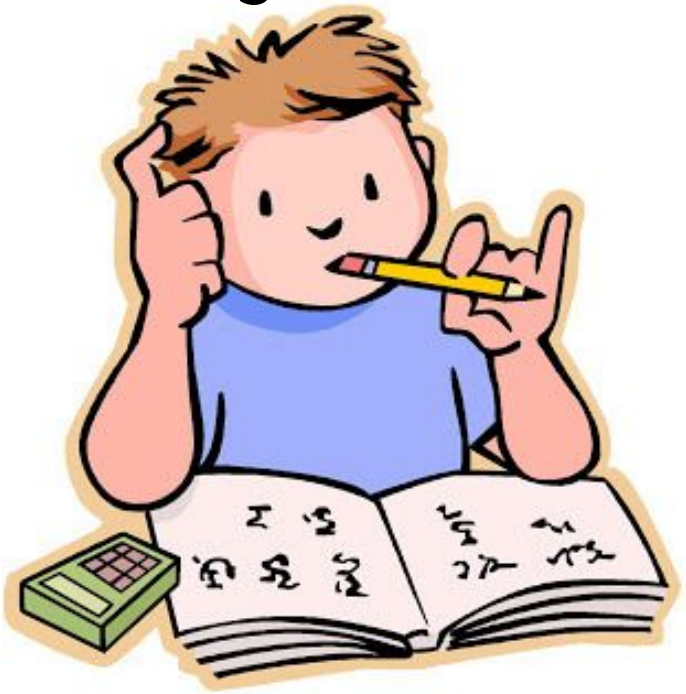
Recapitulación

Recapitulación: preguntas

¿Qué es un objeto?

¿A qué llamamos comportamiento de un objeto?

¿A qué llamamos delegación?



Implementar el comportamiento de los objetos

Para asegurar que los objetos que reciben los mensajes van a entenderlos y hacer algo como respuesta

- Si una celda debe responder si está libre, tendremos que implementarla a más bajo nivel

En términos de POO: implementar un **método** para la celda cuyo nombre es *estaLibre*

Ya vamos a llegar…

Más definiciones: método

Llamamos método a la implementación de la respuesta de un objeto a un mensaje

En términos de implementación, se asemeja a funciones o procedimientos de programación en otros paradigmas

Algunos métodos

```
tablero >> puedoColocar (numero, filaCelda,
  columnaCelda)
tablero >> celda (filaCelda, columnaCelda)
celda >> estaLibre
tablero >> fila (filaCelda)
fila >> esta (numero)
```

Comportamiento es central

Objeto: una entidad que existe en tiempo de ejecución y que puede enviar y recibir mensajes

¡Y actuar!

=>

Objeto: una entidad que existe en tiempo de ejecución y que tiene comportamiento

Objeto != Variable estructurada

Los objetos tienen identidad

- La identidad es lo que distingue a un objeto de otro
  - Sin importar lo parecidas que sean en cuanto a comportamiento, la celda 2 3 y la celda 5 7 del Sudoku son dos objetos diferentes
  - La identidad de un objeto lo acompaña desde su creación hasta su muerte

Los objetos tienen estado

El estado tiene que ver con algo que cambia a través del tiempo

- La celda 2 3 de nuestro programa de Sudoku se encuentra libre al inicio del juego
- A medida que éste avanza puede cambiar esta situación: al final queda allí el número 2
- Decimos que el objeto celda 2 3 ha cambiado de estado, pasando de no tener un valor a tener el valor 2

Los objetos tienen comportamiento

Conjunto de posibles respuestas de un objeto ante los mensajes que recibe

Pueden provocar:

- Un cambio de estado en el objeto receptor del mensaje
- La devolución del estado de un objeto, en su totalidad o parcialmente
- El envío de un mensaje desde el objeto receptor a otro objeto (delegación)

Los objetos tienen responsabilidades

Una manera de hablar de la interacción entre comportamiento y estado

- Comportamiento activo (actuar ante la llegada de mensajes) +
- Mantenimiento de un estado interno

Cambios de estado

Para cambiar el estado de un objeto, tiene que ocurrir *algo* que provoque ese cambio

Ese “algo” suele ser un mensaje recibido por el objeto

Primer vínculo entre estado y comportamiento: uno de los comportamientos posibles de un objeto es el cambiar de estado al recibir un mensaje

Estado ¿visible?

Habitual: estado privado

Sirve como soporte interno para brindar determinado comportamiento

Si fuese necesario accederemos a él mediante mensajes (también comportamiento)

Consulta del estado

- Otro vínculo: hay ocasiones en las que el objeto exhibe su estado a través del comportamiento
  - Probablemente necesitemos preguntarle a una celda si contiene o no un número
  - Si bien éste es un estado del objeto, recurriremos a un mensaje (comportamiento) para que el objeto nos dé información sobre el estado

Encapsulamiento

Cada objeto es responsable de responder a los mensajes que recibe, sin que quien le envía el mensaje tenga que saber cómo lo hace

Razones:

- Puede haber implementaciones alternativas para una misma operación
- En el futuro, podemos cambiar una implementación por otra, ambas correctas, sin afectar al cliente que utiliza el servicio

Encapsulamiento: “Tell, don’t ask”

- Los objetos deben manejar su propio comportamiento, sin que manipulemos su estado desde afuera
  - Si implementamos la celda haciendo que en una celda desocupada haya un 0
  - Preguntar si está libre sería: `celda >> contiene (0)`
  - Pero el cliente conocería cuestiones de implementación interna del objeto
  - Más razonable: `celda >> estaLibre`
  - Ahora la implementación interna es desconocida para el cliente

Polimorfismo

Capacidad de respuesta que tienen distintos objetos de responder de maneras diferentes a un mismo mensaje

Ejemplos:

```
fila >> contiene (7)
columna >> contiene(7)
caja >> contiene(7)
celda23 >> contiene(7)
```

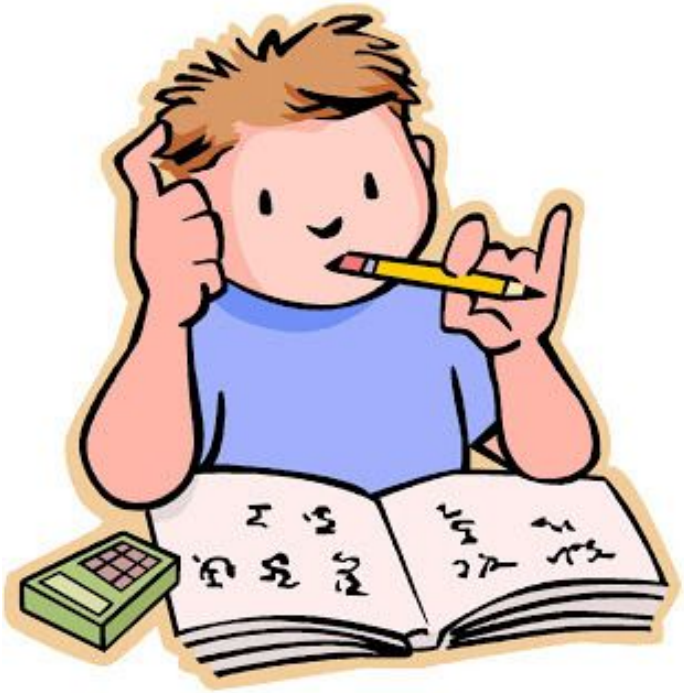
Ya volveremos: tema central de POO

Recapitulación

Recapitulación: preguntas

¿A qué llamamos responsabilidades de un objeto?

¿Qué diferencia a los lenguajes basados en clases y los basados en prototipos?



Implementación de objetos y lenguajes

Basados en clases o basados en prototipos

(según cómo implementan el comportamiento)

Creación de objetos en tiempo de ejecución en el área de memoria dinámica o en tiempo de compilación en la pila

Con verificación de tipos en tiempo de compilación o en tiempo de ejecución

Implementación de objetos y lenguajes: ejemplos

Lenguaje	Implementación del comportamiento	Creación de objetos	Comprobación de tipos
Smalltalk	Clases	Dinámica	T de ejecución
Java	Clases	Dinámica	T de compilación
JavaScript	Prototipos	Dinámica	No hay
Python	Clases	Dinámica	T de ejecución
C++	Clases	Estática	T de compilación
C#	Clases	Dinámica	T de compilación
Ruby	Clases	Dinámica	T de ejecución
Self	Prototipos	Dinámica	No hay

Claves

Se trabaja con objetos y mensajes

Para resolver problemas debemos encontrar

- Objetos
- Secuencia de mensajes
- Implementar comportamiento

El comportamiento es esencial a los objetos

El paradigma se basa en objetos: las clases son una decisión de implementación

POO modulariza en base a las entidades del dominio del problema

Lectura obligatoria

Principios de diseño de Smalltalk, de Daniel H. H. Ingalls.

Lo tienen en:

http://www.smalltalking.net/Papers/stDesign/stDesign.htm

Qué sigue

- Diseño por contrato
  - Como modelo de implementación de objetos
- Colaboraciones de objetos y separación de responsabilidades
  - Incluye delegación, herencia y cuestiones estructurales
- Polimorfismo