## Orientacion a Objetos 1

- **Profesores:**

**Dra. Alicia Diaz**
**Dra. Roxana Giandini**
**Dr. Alejandro Fernandez**
**Dr. Gustavo Rossi**

email:

[alicia, giandini, gustavo, alejandro.fernandez]@lifia.info.unlp.edu.ar

Lifia

## Comienzo de las practicas

- Ya los matriculamos en la plataforma catedras.info.unlp.edu.ar
  - Si no pueden entrar, escriben a objetos.uno@lifia.info.unlp.edu.ar
- En la plataforma ya publicamos:
  - Los criterios de aprobación de la materia
  - Parte del material que utilizaremos (tps, ambiente, etc)
  - Las consultas de práctica inician la semana que viene (semana del 26).
  - Esta semana (semana del 19) les pasaremos un link para que elijan turnos. Esten atentos
- Todos estos temas se presentarán también en la explicación de esta semana (miércoles) y luego se comunicarán por la plataforma

Lifia

## Contenidos del Curso

- Como “pensar” software usando objetos
- Diseño de aplicaciones complejas usando objetos
- Introduccion a la Modelizacion
- Introduccion a la construccion de interfases graficas

**QUE SUPONEMOS QUE SABEN**

- Elementos de algorítmica
- Conceptos básicos de objetos (objeto, mensaje, método, clase, herencia)

Lifia

## Contexto

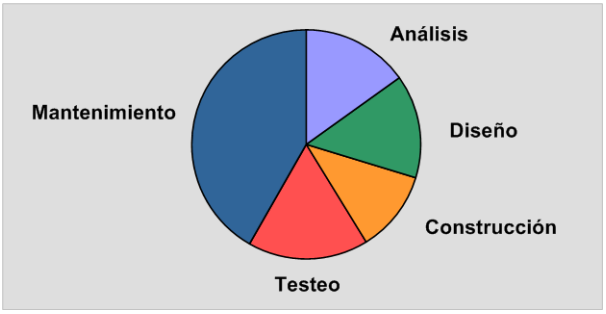
- En los 90: Computacion interactiva y Web Estatica
- En los 2000: Evolucion de la Web. Aparicion de la telefonia movil
- En los 2010: Internet Movil, Internet de las cosas, Computacion en la Nube, Big Data, Deep Learning

En Software: Metodos Agiles vs Unificados, Desarrollo conducido por modelos, Software mas “volatil”, Requerimientos cambiantes permanentemente. Clientes y usuarios mejor formados y piden funcionalidad mas sofisticada

Lifia

## Motivación

- Manejar complejidad
- Flexibilidad al cambio
- Mejorar la reusabilidad
- Minimizar costos de mantenimiento

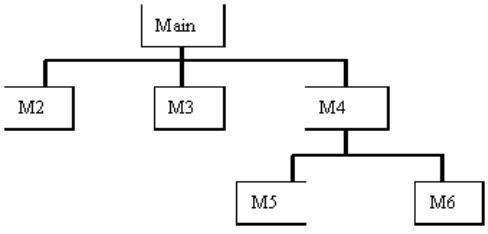



**Costo asociado a cada etapa del desarrollo**

Lifia

## Programación Estructurada

- Sistemas contienen datos y programas.
- Los programas manipulan los datos.
- Los programas están organizados por:
  - Descomposición funcional.
  - Flujo de Datos.
  - Módulos.




- Asignación, secuencia, iteración, condicionales

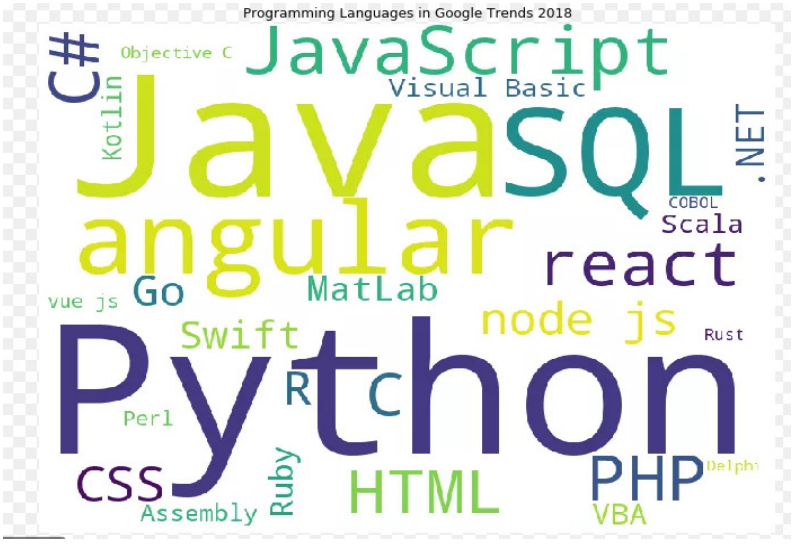
Lifia

## Que tipo de aplicaciones construimos hoy?

- Distribuidas (combinando hard/soft, personas…)
- Basadas en “servicios” provistos por terceros
- Que combinan “partes” de otras aplicaciones
- Que pueden crecer en forma completamente ininmaginable..
- Con operaciones que no son atomicas (ej. Invitacion de amistad en facebook)
- Que se componen y descomponen y cuyas partes son usadas por otros….

Lifia

## Aplicaciones complejas…




Lifia

## Ejemplos

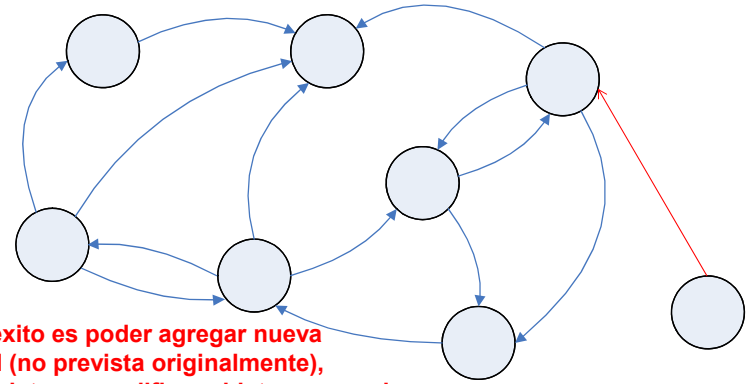
- Facebook y su evolucion
- Netflix (software reemplazando a medios de comunicacion tradicionales)
- Nuevos negocios: Uber, Airbnb, Glovo, etc., todos basados en software

Lifia

## Programa Orientado a Objetos

- ¿Qué es un programa OO?

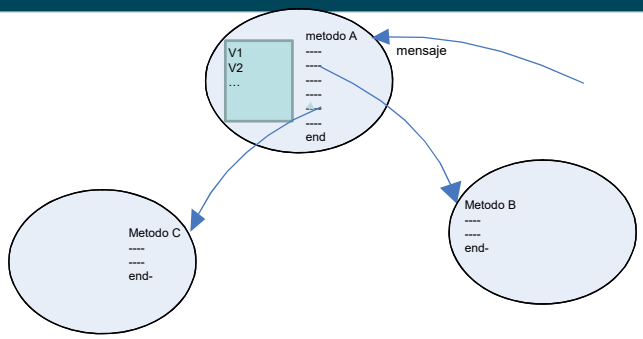
Un conjunto de *objetos* que *colaboran* enviándose *mensajes.* Todo computo ocurre "dentro" de los objetos



**La clave del exito es poder agregar nueva funcionalidad (no prevista originalmente), reemplazar objetos o modificar objetos y que el sistema "no se entere", ni se rompa.**
**E.g. integracion Whatsapp y messenger Facebook**

Lifia

## Más en detalle




- Los sistemas están compuestos (solamente) por un conjunto de **objetos** que **colaboran** para llevar a cabo sus responsabilidades.
- Los objetos son **responsables** de:
  - conocer sus propiedades,
  - conocer otros objetos (con los que colaboran) y
  - llevar a cabo ciertas acciones.

Lifia

## Aspectos de interés en esta definición

- No hay un objeto "main"
- Cuando codificamos, describimos (programamos) clases
- Una jerarquia de clases no indica lo mismo que la jerarquia top-down
- Cuando se ejecuta el programa lo que tenemos son objetos que cooperan y que se crean dinámicamente durante la ejecución del programa

Lifia

## Impacto en como “pensamos” el software

- La estructura general cambia: en vez de una jerarquía: Main/procedures/sub-procedures tenemos una red de “cosas” que se comunican
- Pensamos en que “cosas” hay en nuestro software (los objetos) y como se comunican entre sí.
- Hay un “shift” mental crítico en forma en la cual pensamos el software como objetos

Lifia

## ¿Qué es un objeto?

- Es una *abstracción* de una *entidad* del *dominio del problema.* Ejemplos: Persona, Producto, Cuenta Bancaria, Auto, Plan de Estudios,….
- *Puede representar tambien conceptos del espacio de la solucion (estructuras de datos, tipos “basicos”, archivos, ventanas, iconos..)*

Lifia

## Características de los Objetos

- Un objeto tiene:
  - *Identidad.*
    - para distinguir un objeto de otro
  - *Conocimiento.*
    - En base a sus relaciones con otros objetos y su estado interno
  - *Comportamiento*.
    - Conjunto de mensajes que un objeto sabe responder

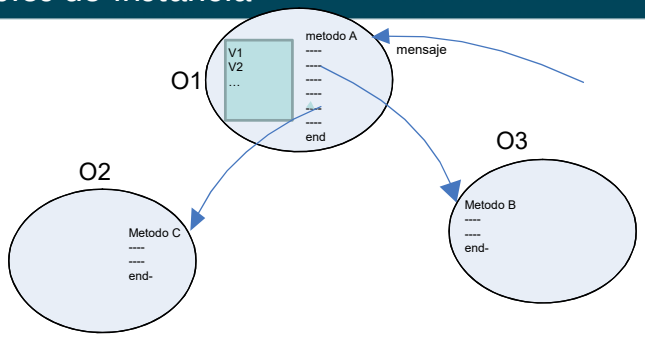
Lifia

## El estado interno

- El estado interno de un objeto determina su *conocimiento*.
- El estado interno está dado por:
  - Propiedades basicas (intrinsecas) del objeto.
  - Otros objetos con los cuales colabora para llevar a cabo sus responsabilidades.
- El estado interno se mantiene en las *variables de instancia (v.i.)* del objeto.
- Es **privado** del objeto. Ningún otro objeto puede accederlo. (Cual es el impacto de esto?)

Lifia

## Variables de instancia




- En general las variables son REFERENCIAS (punteros) a otros objetos con los cuales el objeto colabora.
- Algunas pueden ser atributos basicos
- En el grafico O1 puede mandarle mensajes a O2 y O3 porque "los conoce", o sea hay una variable en O1 que APUNTA a O2 y otra a O3 (o la misma variable que cambia de valor en diferentes momentos)

Lifia

## Ejemplos

- Objeto Alumno:

  v.i: nombre, dni, fecha nac,

  carrera, legajo

-Las 2 primeras pueden ser tipos "básicos": string, numero (dependiendo del lenguaje)

-Las otras 3 son referencias a objetos de la Clase Fecha, Carrera y Legajo

- Objeto Carrera:

  v.i: nombre, año, materias

Materias es una colección (array?) de objetos Materia

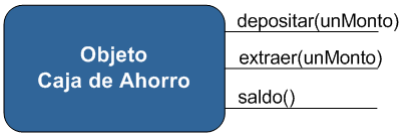
- Objeto Materia:

  v.i: nombre, año, semestre, correlativas

Lifia

## Comportamiento

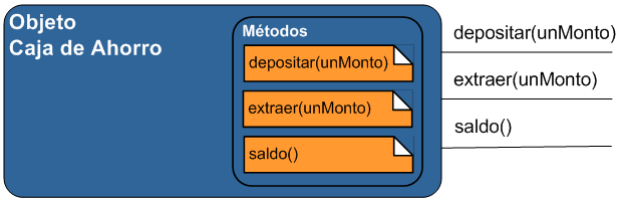
- Un objeto se define en términos de su comportamiento.
- El comportamiento indica qué sabe hacer el objeto. Cuáles son sus *responsabilidades*.
- Se especifica a través del conjunto de *mensajes* que el objeto sabe responder: *protocolo*.
- Ejemplo:




Lifia

## Comportamiento – implementación

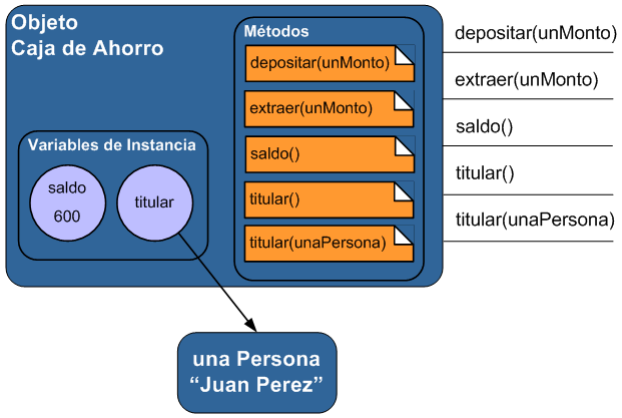
- La **realizacion** de cada mensaje (es decir, la manera en que un objeto responde a un mensaje) se especifica a través de un *método.*
- Cuando un **objeto** recibe un **mensaje** responde activando el *método* asociado.
- El que envía el mensaje *delega* en el receptor la manera de resolverlo, que es *privada* del objeto.




**Impacto: Localizacion de funcionalidad**

Lifia

## Ejemplo Caja de Ahorro




Observese la variable “titular” apuntando a un objeto Persona

Lifia

## Envío de un mensaje

- Para poder enviarle un mensaje a un objeto, **hay que conocerlo.**
- Al enviarle un mensaje a un objeto, éste responde activando el método asociado a ese mensaje (siempre y cuando exista).
- Como resultado del envío de un mensaje puede retornarse un objeto.

Lifia

## Especificación de un Mensaje

- ¿Cómo se especifica un mensaje?
  - *Nombre*: correspondiente al protocolo del objeto receptor.
  - *Parámetros*: información necesaria para resolver el mensaje.
- Cada lenguaje de programación propone una sintaxis particular para indicar el envío de un mensaje.

*Ejemplo:* *cuenta depositar (cantidad)*
*figura dibujar*
*figura rotar*
*producto precio*

- Tanto figura como producto son variables que apuntan a un objeto que entiende el mensaje correspondiente

Lifia

## Métodos

- ¿Qué es un método?
  - Es la contraparte funcional del mensaje.
  - Expresa la forma de llevar a cabo la semántica propia de un mensaje particular (el *cómo*).
- Un método puede realizar básicamente 3 cosas:
  - Modificar el estado interno del objeto.
  - Colaborar con otros objetos (enviándoles mensajes).
  - Retornar y terminar.

**Y la entrada/salida de informacion?**

Lifia

## Ejemplo – Depositar en Cuenta Bancaria

**depositar(unMonto)**

*"Agrega unMonto al saldo actual de la cuenta"*

saldo ← saldo + unMonto

Lifia

## Formas de Conocimiento

- Para que un objeto conozca a otro lo debe poder "nombrar". Decimos que se establece una ligadura (binding) entre un nombre y un objeto.
- Podemos identificar tres formas de conocimiento o tipos de relaciones entre objetos.
  - Conocimiento Interno: Variables de instancia.
  - Conocimiento Externo: Parámetros.
  - Conocimiento Temporal: Variables temporales.
- Además existe una cuarta forma de conocimiento especial: las pseudo-variables (como "this" o "self"

Lifia

## Encapsulamiento

*"Es la cualidad de los objetos de ocultar los detalles de implementación y su estado interno del mundo exterior"*

- Características:
  - Esconde detalles de implementación.
  - Protege el estado interno de los objetos.
  - Un objeto sólo muestra su "cara visible" por medio de su protocolo.
  - Los métodos y su estado quedan escondidos para cualquier otro objeto. Es el objeto quien decide *qué* se publica.
  - Facilita modularidad y reutilización.

Lifia

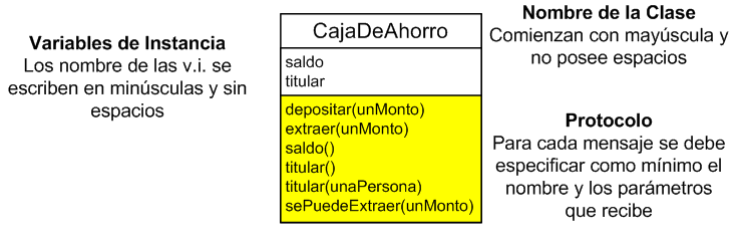
## Clases e instancias

- Una clase es una descripción abstracta de un conjunto de objetos.
- Las clases cumplen tres roles:
  - Agrupan el comportamiento común a sus instancias.
  - Definen la *forma* de sus instancias.
  - *Crean objetos que son instancia de ellas*
- En consecuencia todas las instancias de una clase se comportan de la misma manera.
- Cada instancia mantendrá su propio estado interno.

Lifia

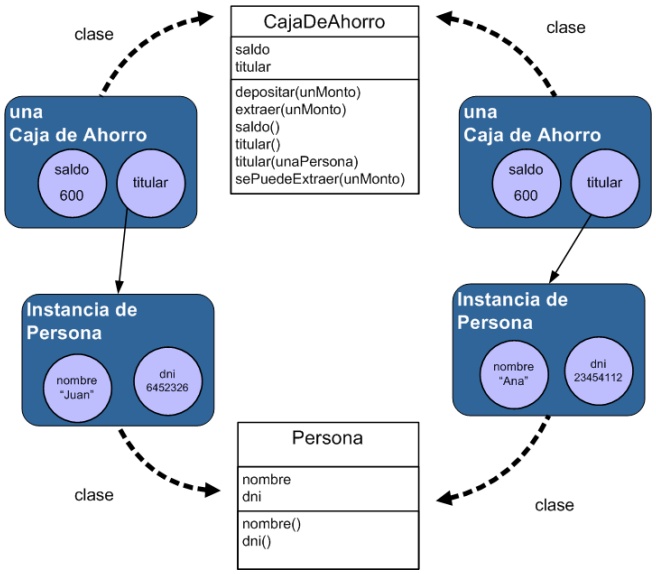
## Especificación de Clases

- Las clases se especifican por medio de un nombre, el estado o estructura interna que tendrán sus instancias y los métodos asociados que definen el comportamiento
- Gráficamente:



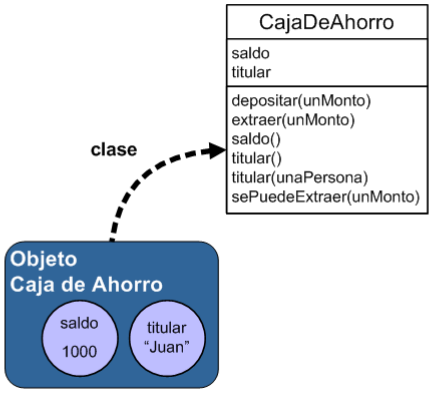

Lifia

## Ejemplo de clases e instancias




Lifia

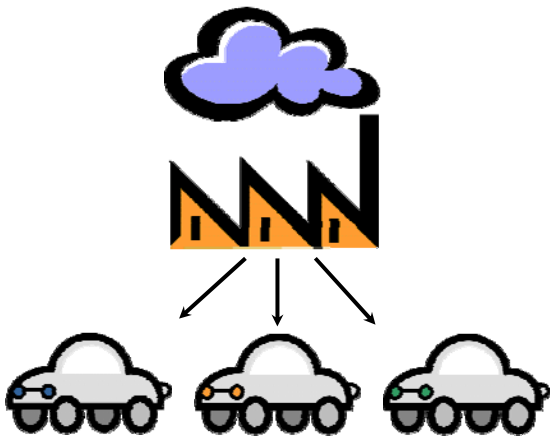
## Envío de mensajes con clases




Lifia

## Creación de Objetos

- ¿Cómo creamos nuevos objetos?
- Instanciación



Lifia

## Creación de Objetos

- Comúnmente se utiliza la palabra reservada *new* para instanciar nuevos objetos.
- Según el lenguaje
  - **new** es un mensaje que se envía a la clase.
  - **new** es una operación especial.
- Nosotros vamos a usar al new como mensaje de clase (las clases son objetos).
- En este caso, la ejecución del método retorna una nueva instancia de la clase a la que se le envía el mensaje.
- Quien crea objetos? Cuando los crea?

Lifia

## Instanciación

- Es el mecanismo de creación de objetos.
- Los objetos se *instancian* a partir de un molde.
- La **clase** funciona como molde.
- Un nuevo objeto es una *instancia* de una clase.
- Todas las instancias de una misma clase
  - Tendrán la misma estructura interna.
  - Responderán al mismo protocolo (los mismos mensajes) de la misma manera (los mismos métodos).

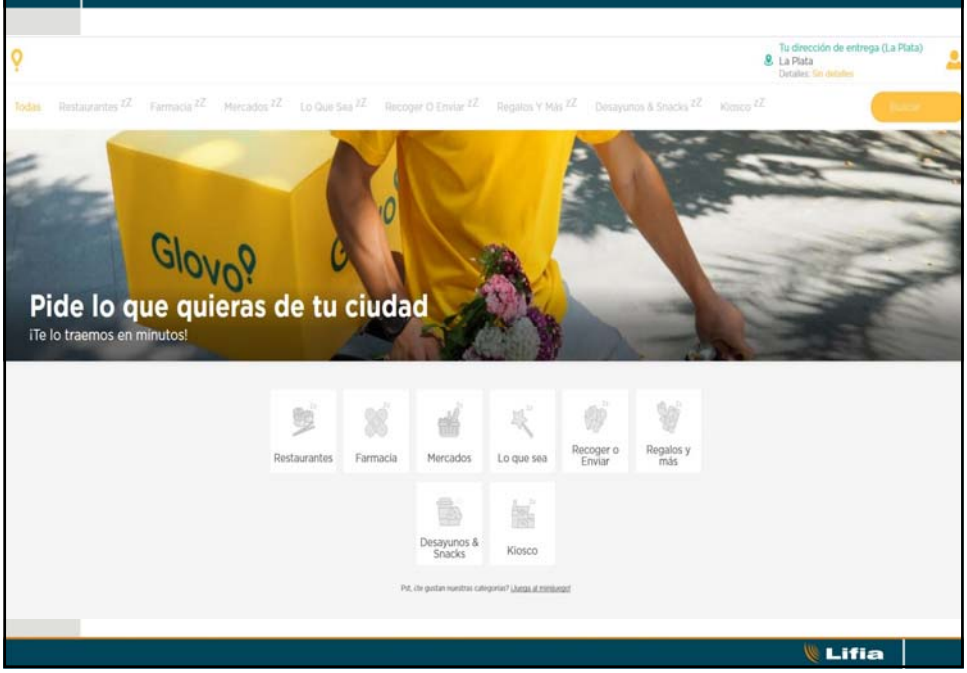
Lifia

## Inicialización

- Para que un objeto esté listo para llevar a cabo sus responsabilidades hace falta *inicializarlo.*
- Inicializar un objeto significa darle valor a sus variables.
- ¿De dónde sacamos esos valores iniciales?
- A veces son valores “obvios”. Otras veces necesitamos indicarlos en la instanciacion

Lifia

## Ejemplo: Gloovo




Lifia

## Objetivos del Ejemplo

- Repasar los conceptos basicos sobre un diseño realista
- Vamos a analizar el diseño sin preocuparnos por COMO llegamos a el (por ahora).

Se trata de describir con objetos la funcionalidad mas importante de un sistema del tipo Glovo o PedidosYa, que permite que cualquier usuario registrado (cliente) pida un producto en un comercio registrado y alguien (otros usuarios especiales que se postulan como “Gloovers") le lleve el producto a la casa.

Con un conjunto de opciones desplegables el "cliente" elige producto (o producto y negocio en el que lo quiere comprar) y registra un pedido. En ese momento sabe el precio que pagará tanto por el producto como por el envío. El sistema todavía no puede estimar un tiempo de entrega. El “sistema” recibe el pedido y lo postea entre los usuarios “Gloovers". Cuando alguno de ellos lo “toma”, el pedido se asigna al "acarreador“ (un gloover) y se comunica al negocio que provee el producto para que lo prepare. El Gloover va al negocio y retira el producto. Utiliza la aplicación para confirmar que retiro el producto. Se dirige a la dirección de entrega y entrega el producto. Utiliza la aplicación para confirmar que lo entregó.

Cuando el cliente recibe el pedido, utiliza la aplicación para confirmar la recepción. En ese momento, se carga el costo a su medio de pago, se para al negocio, y se paga al "acarreador". Adicionalmente el cliente puede calificar al gloover con un puntaje y un comentario

Lifia