Programación orientada a objetos

Diagrama de clases

{desafío} latam_

Utilizar elementos de la programación orientada a objetos para la implementación de una pieza de software que da solución a un problema de baja complejidad

- Unidad 1: Flujo, ciclos y métodos
- Unidad 2: Arreglos y archivos
- Unidad 3: Programación orientada a objetos
- Unidad 4: Pruebas unitarias y TDD

Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Hacer uso de la herramienta StarUML para el diseño de diagramas UML.*
- *Reconocer una clase Java dentro de un diagrama de Clases para entender la lógica de la estructura de código.*

{desafío} latam_

¿En qué consiste un diagrama de clases?

{desafío} latam_

Herramienta StarUML

- Permite realizar todos los tipos de diagramas que se explicarán en esta unidad.
- Su uso es libre, aunque si lo pagamos tendremos acceso a más opciones (para los que deseen ahondar en el diseño con esta herramienta existen planes mensuales y anuales).

Enlace de descarga

/* Diagrama de clases */

Componentes de un diagrama de clase

Un diagrama de clases se compone de la siguiente estructura:

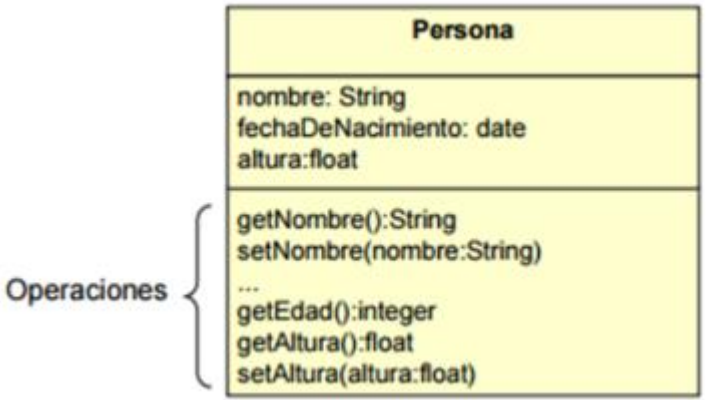
1. Nombre del Objeto a modelar.
2. Atributos o propiedades.
3. Tipo de dato del atributo.
4. Acciones u Operaciones.

Componentes de un diagrama de clase

Nombre de Objeto: Persona

Atributos: nombre: tipo de dato String
- fechaDeNacimiento: tipo de dato Date.
- altura: tipo de dato Float.

Acciones u Operaciones:
- getNombre(): String
- setNombre (String nombre)
- getEdad() : integer
- getAltura(): float
- setAltura (float altura)

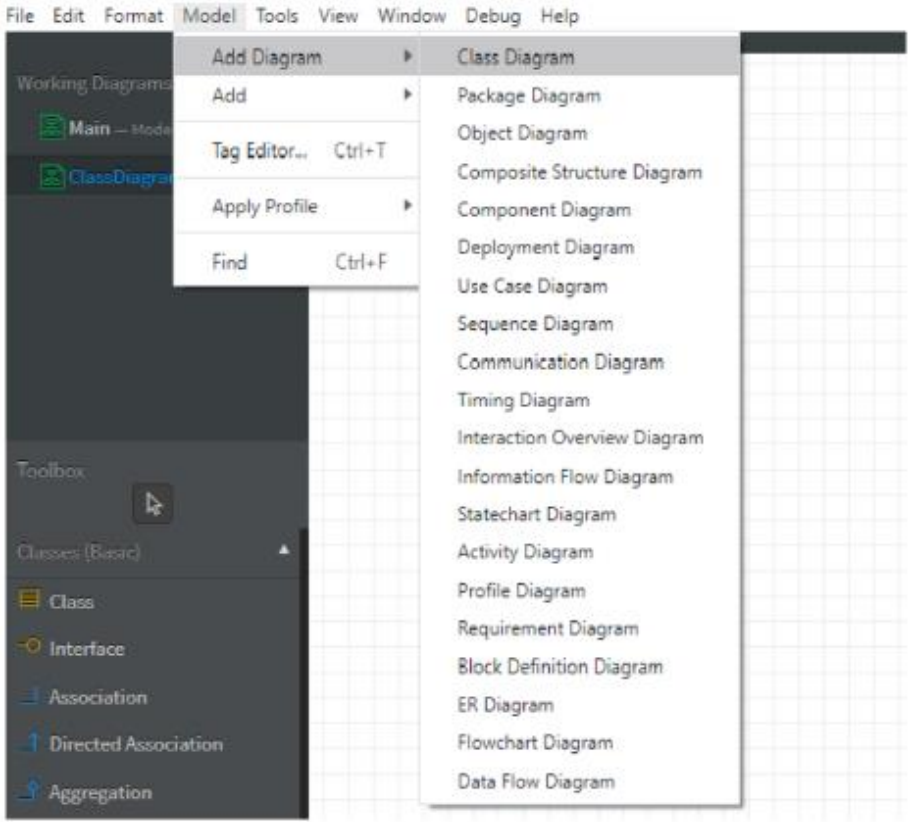


{desafío} latam_

Crear un diagrama de clases

En la herramienta StarUML

Model -> Add Diagram -> Class Diagram

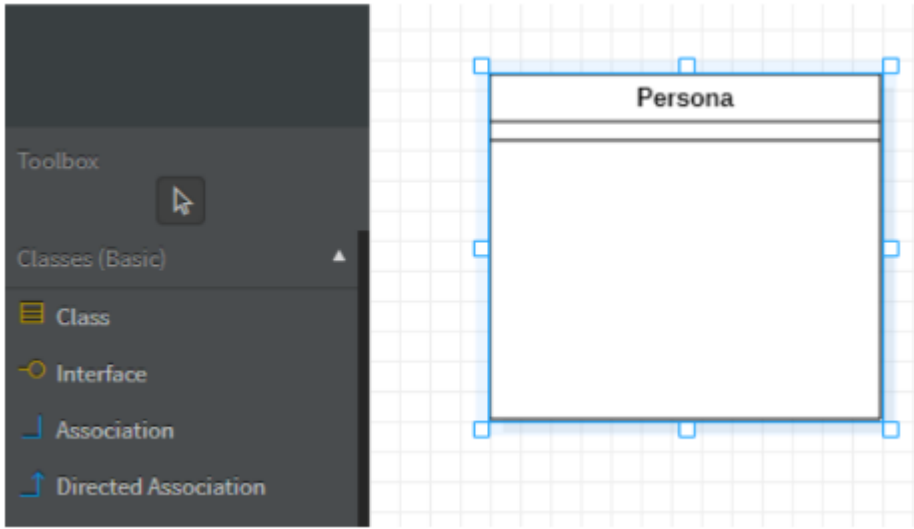


{desafío} latam_

Crear un diagrama de clases

En la herramienta StarUML

Para crear una Clase damos clic en “class” y dibujamos un cuadrado con el mouse en el espacio en blanco.

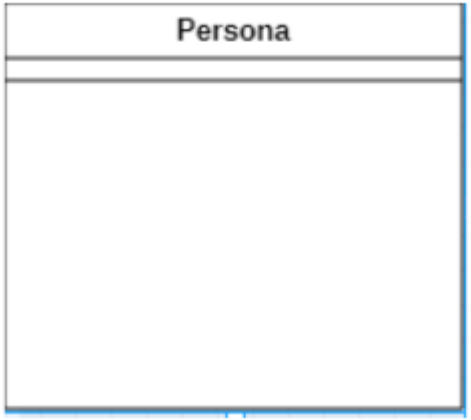


{desafío} latam_

Crear un diagrama de clases

En la herramienta StarUML

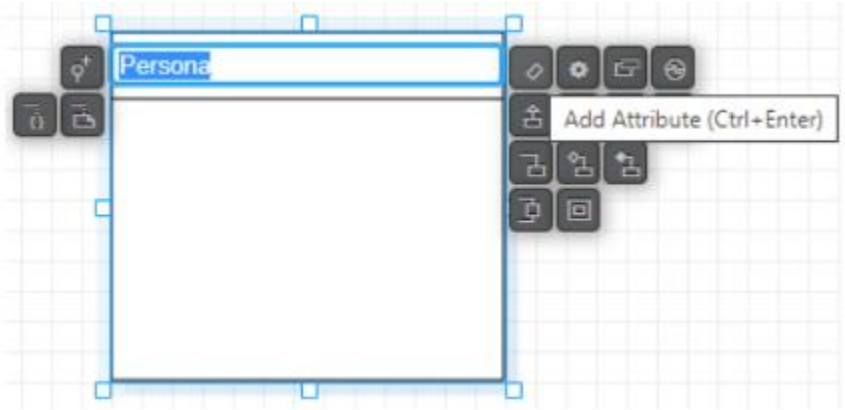
Para cambiar nombre de la clase solo debemos dar doble clic en la caja y cambiar el nombre, en el ejemplo es “Persona”.

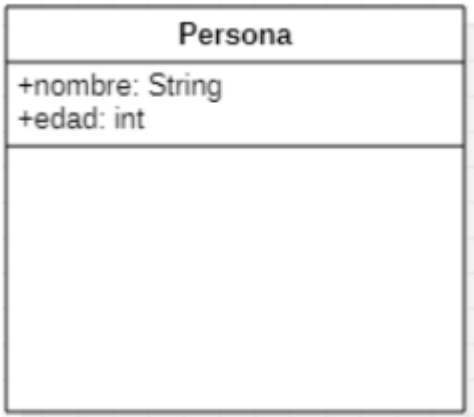


Crear un diagrama de clases

En la herramienta StarUML

Para agregar atributos a la clase, seleccionamos el icono y empezamos agregando el nombre de atributo y tipo de dato:

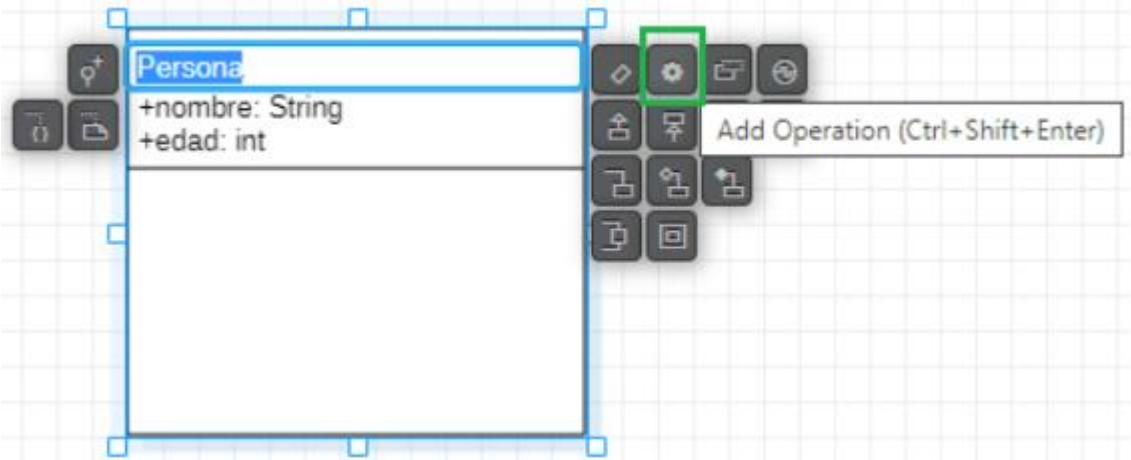


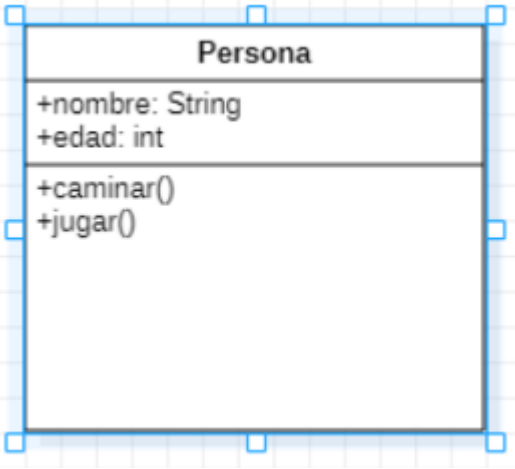


Crear un diagrama de clases

En la herramienta StarUML

Agregando acciones u operaciones:





Sigamos practicando en la herramienta StarUML

/* Reconocer una clase Java dentro de un diagrama de clase */

Antes de comenzar...

¿Qué es un objeto en Java?

- Un objeto es cualquier elemento de la vida real que sea tangible.
- Este elemento se compone por atributos o propiedades con su tipo de datos asociado.
- Los atributos siempre son privados, constructores, getter (accesadores), setter (mutadores) y eventos propios del objeto.

Cuando se programa un objeto en Java, este objeto pasa a llamarse **Clase**, ya que desde el modelo ideal del objeto se construye en Java para su ejecución.

En pocas palabras, un objeto es cualquier cosa tangible de la vida real y una clase es el objeto en ejecución con operaciones programables dentro del lenguaje Java.

Nomenclatura para la construcción de una Clase

Nombre de la clase

- Siempre debe comenzar con mayúscula
- Si el nombre de la clase es compuesta (más de una palabra), cada palabra debe comenzar con mayúscula.

Ejemplo:

- **Persona**: Clase con nombre simple.
- **PersonaJuridica**: Clase con nombre compuesto.

Nomenclatura para la construcción de una Clase

Atributos o Propiedades

El nombre debe ser lo más descriptivo posible por norma y por seguir buenas prácticas de programación. La regla de nomenclatura es la siguiente:

- Siempre se escribe la primera letra en minúscula para nombre simples
- Si es un nombre compuesto, el segundo nombre debe comenzar siempre con mayúscula, todo esto acompañado con el tipo de dato y su tipo de acceso.

Ejemplo:

- private - Tipo accesador.
- int - Tipo de dato.
- atributo - Nombre (simple) - fechaNacimiento (nombre compuesto).

```
private int nombre
private int fechaNacimiento
```

Nomenclatura para la construcción de una Clase

Constructores

- Siempre lleva el mismo nombre de la Clase, y puede recibir o no parámetros.

Más adelante veremos en profundidad este punto.

Nomenclatura para la construcción de una Clase

Operaciones

- Se forman por su tipo de acceso, público o privado.
- El método u operación puede o no recibir parámetros de entradas.
- El nombre del método siempre debe ser representado por el nombre de una acción.
    - Si el nombre simple del método siempre es con minúscula.
    - Si el nombre compuesto del método, la segunda palabra del nombre del método debe comenzar con su primera letra mayúscula
    - Si el nombre tiene dos o más palabras compuestas, cada palabra adicional debe comenzar con letra mayúscula.

Nomenclatura para la construcción de una Clase

Operaciones

Ejemplos:

- private - Tipo accesador.
- int - Retorno de tipo de dato entero.
- void: No retorna ningún valor.
- enviarMensajePersonal - Nombre de método compuesto.
- salir - Nombre de método simple.

Nomenclatura para la construcción de una Clase

Operaciones

- Método de nombre simple, sin parámetros y no retorna ningún valor:

```
public void imprimir() { }
```

- Método de nombre simple, con un parámetro y no retorna ningún valor:

```
public void imprimir(String input) { }
```

- Método con nombre simple, sin parámetros y que retorna un valor:

```
public int numero() { return 3 ; }
```

- Método con nombre simple, con parámetros y que retorna un valor:

```
public int numero(int valor ) { return valor ; }
```

Ejercicio guiado

Elementos de una clase

Crear un diagrama de clases para establecer elementos de una casa

Cocina

- altura : double
- cantidad hornillas : int
- prenderHorno() : String

Televisor

- marca : String
- definición : String
- apagarTelevisor()

Escritorio

- tamaño : double
- tipo : String

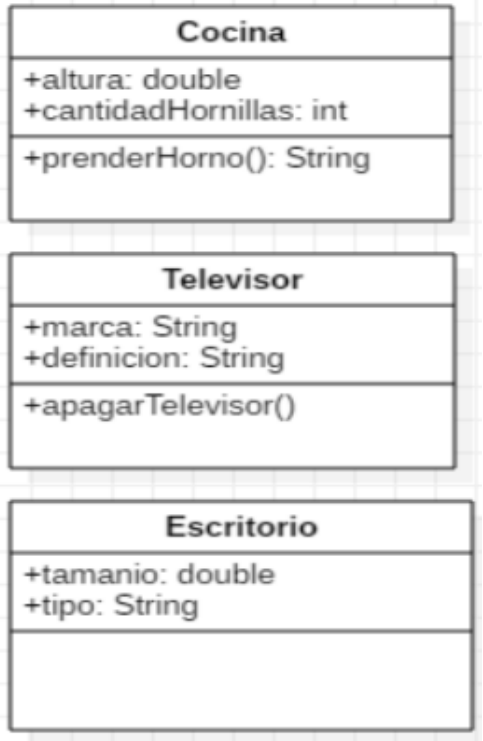
Elementos de una clase

Solución, utilizando StarUML

Crear cada clase con sus atributos y operaciones.

- Lo primero es identificar los objetos, sus atributos y sus operaciones. Posteriormente, se debe seguir las normas de la creación de un diagrama de Clases.

La siguiente imagen ilustra la solución con starUML:



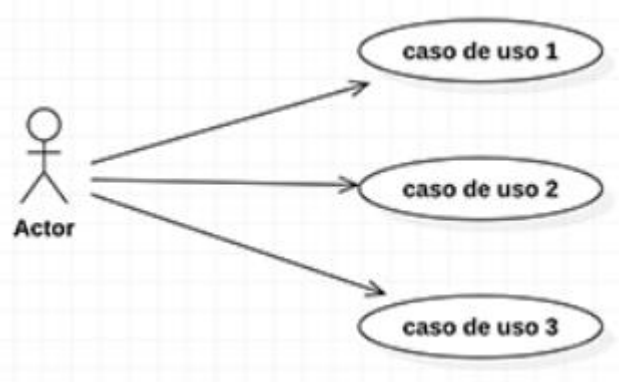
/* Diagrama de casos de uso */

Casos de uso

Diagrama

- Interacción típica entre un usuario y un sistema de software.
- Capta alguna función visible para el usuario.
- Puede ser pequeño o grande.
- Logra un objeto discreto para el usuario.

Una representación de un diagrama de casos de uso sería el siguiente:



Casos de uso

Diagrama

- Usualmente, el caso de uso se extrae de las interacciones que los potenciales usuarios del sistema tengan con la aplicación que se desee construir.
- Cada una de estas, se debe abordar de forma discreta, darle un nombre y escribir una breve descripción.
- No hay que detallar tan profundamente esta interacción, todo esto dependiendo de la cantidad de ramificaciones, de las que esté compuesto el caso de uso, se podrá más adelante, obtener mayores detalles que pueden resultar en nuevos casos de uso.
- Objetivos del usuario versus, interacciones con el sistema.

Casos de uso

Diagrama

Actores

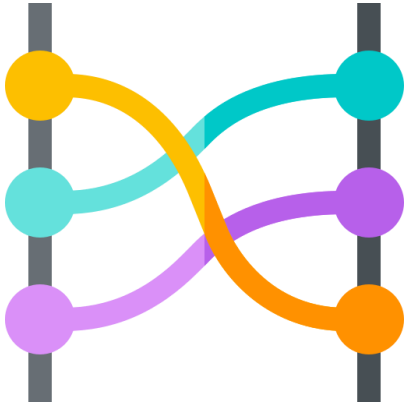
- Empleamos el término actor para llamar así al usuario, cuando desempeña ese papel con respecto al sistema.
- No es necesario que los actores sean seres humanos.

Casos de uso

Diagrama

Relaciones

- UML, define relaciones de estereotipos y generalización.
- Con estas relaciones, podemos ver gráficamente el cómo interactúan los casos de uso y los actores, para una mejor comprensión del escenario.

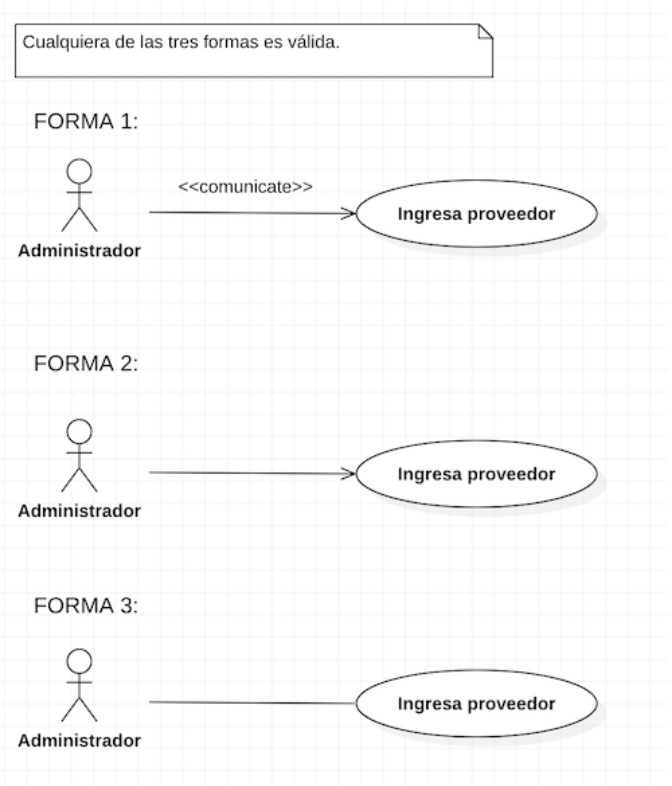


Casos de uso

Diagrama

Estereotipo: <<comunicate>>

- Esta relación es la que más veremos en los CU, como estereotipo se representa por <<communicate>>; pero generalmente este estereotipo no va escrito.
- Es una relación de asociación que nos muestra la interacción entre un actor y el caso de uso.

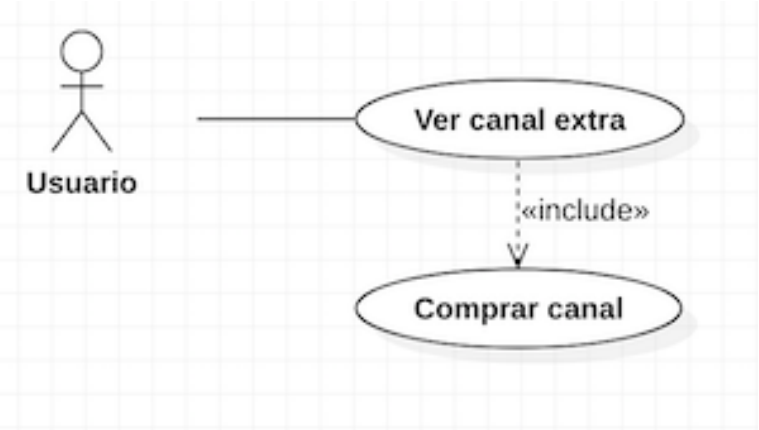


Casos de uso

Diagrama

Estereotipo: <<include>>

- En términos muy simples, cuando relacionamos dos casos de uso con un “include”, estamos diciendo que el primero (el caso de uso base) incluye al segundo (el caso de uso incluido).
- Es decir, el segundo es parte esencial del primero. Sin el segundo, el primero no podría funcionar bien; pues no podría cumplir su objetivo.

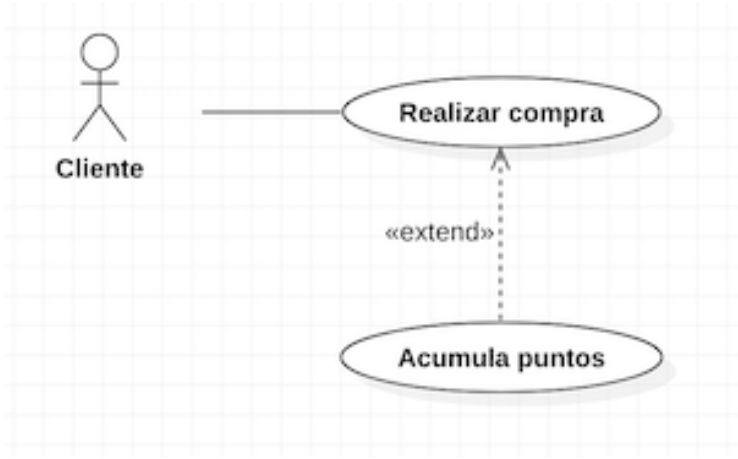


Casos de uso

Diagrama

Estereotipo: <<extend>>

- Un caso de uso puede tener una extensión que no sea indispensable, pero sería bueno para la comprensión de lo que se desea desarrollar, que esta extensión sea expresada en el diagrama, es en este caso (no abusar), en donde haremos esta relación.

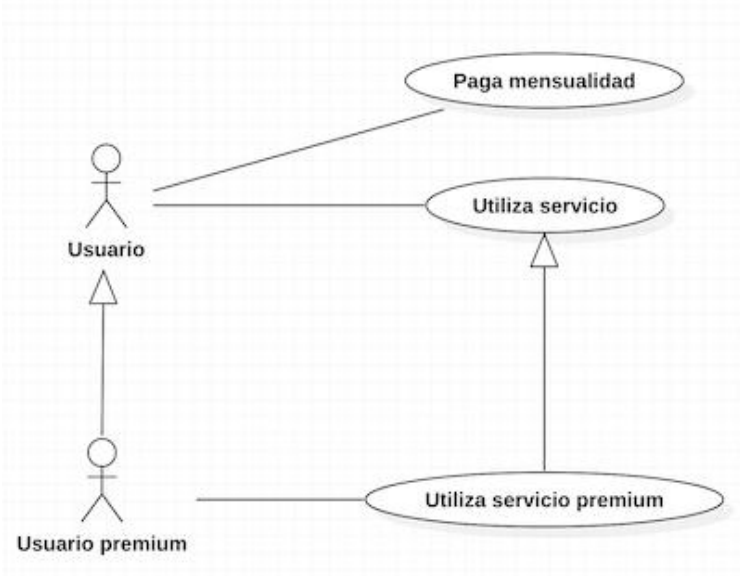


Casos de uso

Diagrama

Generalización

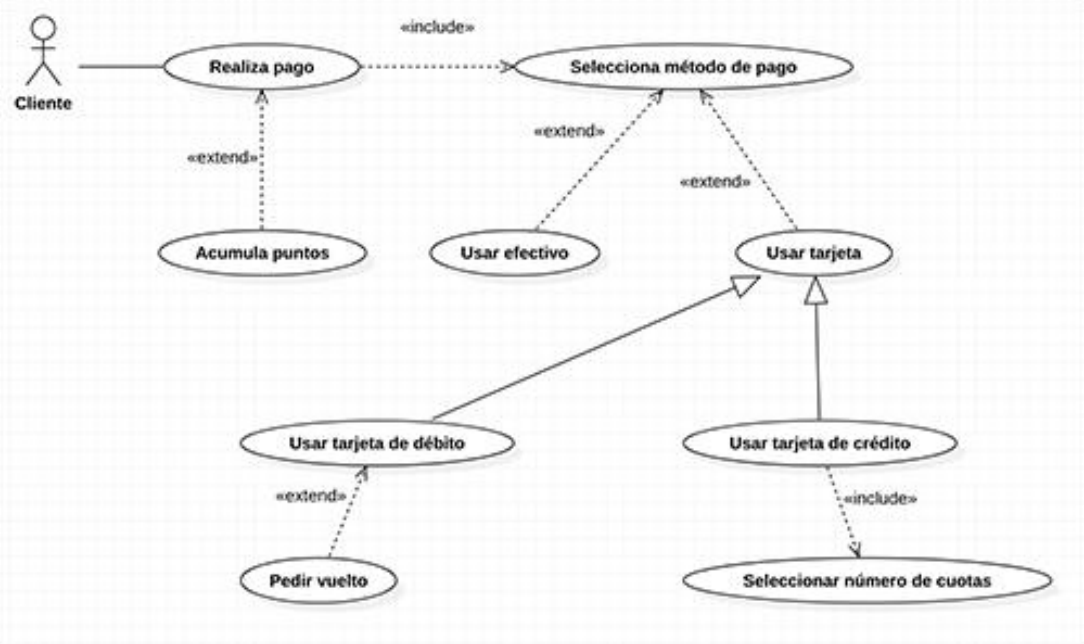
- Cuando hablamos de generalización, esta vez sí que nos estamos refiriendo a algo muy parecido de lo que hace la herencia en la orientación a objetos, pero esta vez en el contexto de comprender el escenario.



Casos de uso

Diagrama

Evitar abusos en la granularidad



Ejercicio

"Diseñar un diagrama de casos de uso, que exprese el escenario de una máquina expendedora de bebidas"

Ejercicio

"Diseñar un diagrama de casos de uso, que exprese el escenario que responde a un sistema de ventas de entradas online"

{desafío} latam_

¿Cuál es la diferencia entre los diagramas de clases y de casos de uso?

Próxima sesión...

- *Desafío guiado*

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales