UML Y Diagramas de Clases (Parte I)

Introducción a UML

Competencias

- Comprender qué es UML y para qué nos sirve.
- Reconocer los distintos tipos de diagramas en UML para analizar los requerimientos e implementar las posibles soluciones con rapidez.

Introducción

Dentro del mundo de la programación, sea cual sea su lenguaje, UML toma un rol importante a la hora de desarrollar un programa, software o aplicación.

En esta unidad nos embarcamos a conocer el mundo de los diagramas y la importancia que tiene en la programación. Un diagrama bien desarrollado puede ahorrar mucho tiempo si entendemos el comportamiento de sus componentes, relaciones y desarrollo.

Los diagramas son utilizados para representar :

- El funcionamiento de un Software a gran escala.
- Comportamientos de acciones.
- Comunicación entre distintos componentes del diagrama a modelar.

Cuando hablamos de UML podemos identificar de forma inmediata de qué manera se explicará un diagrama, sea cual sea el caso expuesto.

Qué es UML y su importancia

UML, Unified Modeling Language (o Lenguaje Unificado de Modelado en español), es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. No es un lenguaje de programación, como piensa mucha gente, sino que es una serie de normas y estándares gráficos respecto a cómo se debe representar los esquemas relativos al software. En otras palabras, es un estándar de cómo se debe representar algo.

En este capítulo el enfoque será reconocer un diagrama de clases orientado a objetos en Java.

UML del mundo real a Diagrama

Antes de comenzar a diagramar se debe tener pleno conocimiento de todos los requerimientos del software. La importancia de esto es clave. Si realizamos una analogía en la vida real, un diagrama sería el plano de un arquitecto para construir una casa; con esto nos referimos a que un buen diagrama debe explicar por sí solo un software sin siquiera utilizarlo o programarlo. Esa es la importancia de entender bien el negocio: saber crear un diagrama y plasmar lo importante del software.

El siguiente ejemplo muestra un diagrama de clases de empleados en una empresa:

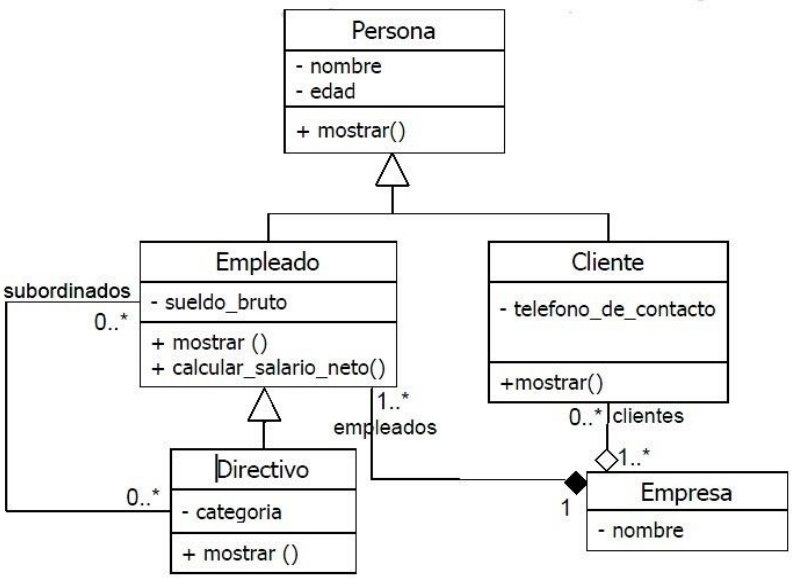

Imagen 1. Ejemplo de Diagrama de Clases.
Fuente: Desafío Latam.

Tipos de Diagramas

Debemos considerar que en el mundo de UML existen varios tipos de diagramas. Cada diagrama está enfocado en un tipo de detalle específico. Por ejemplo, existen diagramas de flujo, clases, secuencia, entre otros. Por concepto de sentido común aprenderemos a conocer a nivel de lectura algunos tipos de diagramas:

- **Diagramas de casos de uso:** Representan a los actores y casos de uso (procesos principales) que intervienen en un desarrollo de software.

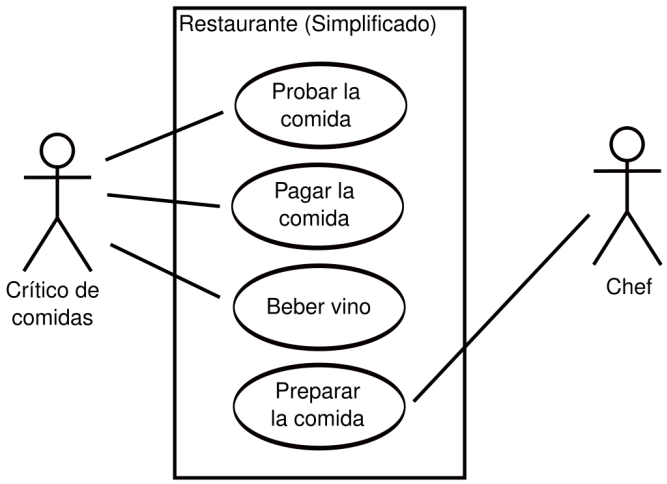

Imagen 2. Ejemplo diagrama de caso de Uso.
Fuente: Desafío Latam.

- **Diagramas de clase:** El diagrama de clases recoge las clases de objetos y sus asociaciones. En este diagrama se representa la estructura y el comportamiento de cada uno de los objetos del sistema y sus relaciones con los demás objetos, pero no muestra información temporal, sino que muestra el comportamiento de los eventos en los flujos de conexión entre clases.

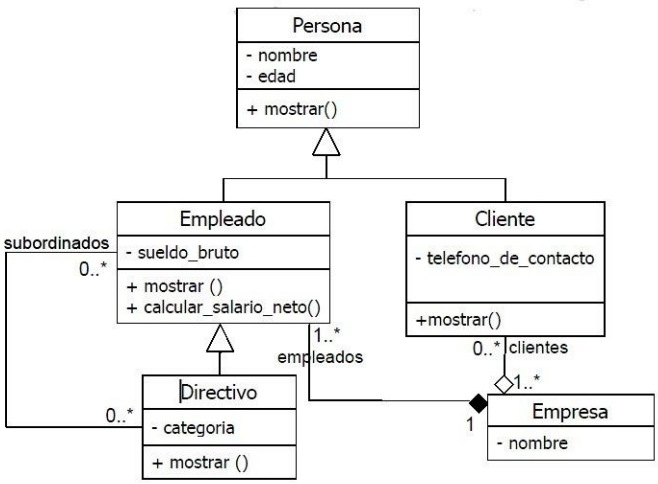

Imagen 3. Ejemplo diagrama de Clases.

- **Diagrama de secuencia:** Representan objetos software y el intercambio de mensajes entre ellos. El diagrama siempre llevará como normativa crear objetos de izquierda a derecha, estos objetos representan el flujo del diagrama, se pueden ocupar en una o varias llamadas del flujo según sea el caso.

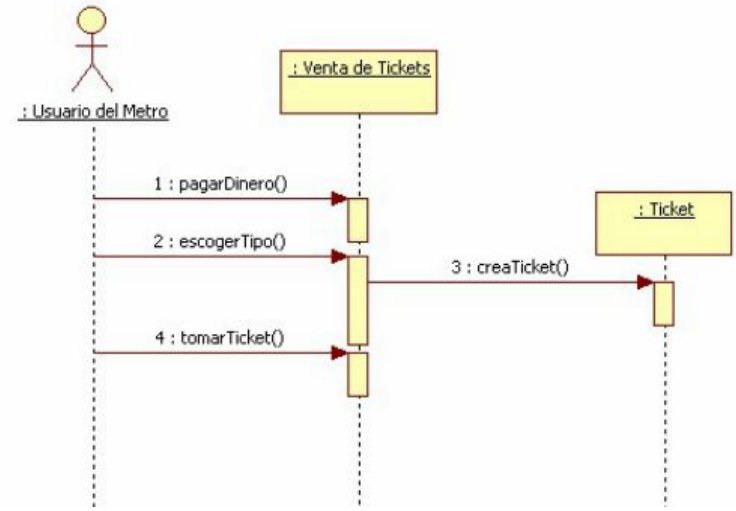

Imagen 4. Ejemplo diagrama de secuencia.
Fuente: Desafío Latam.

- **Diagrama de colaboración:** Representan objetos o clases y la forma en que se transmiten mensajes; colaboran entre ellos para cumplir un objetivo.

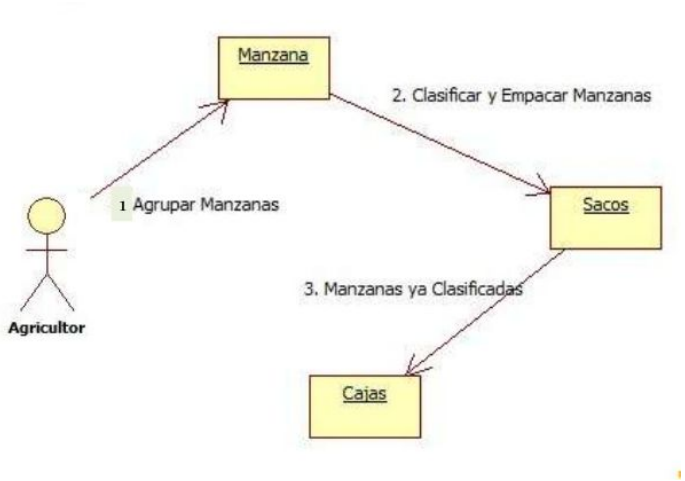

Imagen 5. Ejemplo diagrama de colaboración.
Fuente: Desafío Latam.

- **Diagrama de estados:** Representan cómo evoluciona un sistema, es decir, cómo va cambiando de estado a medida que se producen determinados eventos.

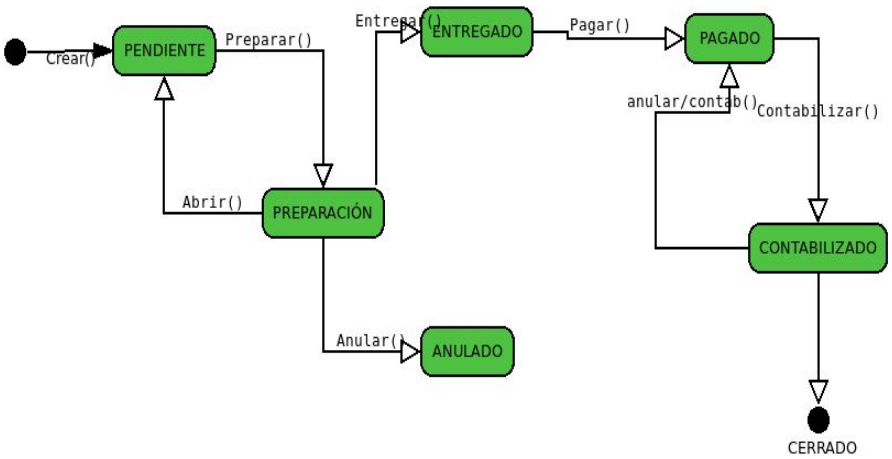

Imagen 6. Ejemplo diagrama de estados.
Fuente: Desafío Latam.

- **Otros diagramas:**
    - Diagramas de actividad,
    - diagramas de paquetes,
    - diagramas de arquitectura software, entre otros.

Ejercicio guiado - Analizar el diagrama a utilizar

La funcionalidad “Venta con Débito” realiza ventas desde puntos Transbank. Comienza cuando el vendedor ingresa el monto en el punto de venta. Posteriormente, se le entrega la máquina al cliente el cual valida e ingresa su clave. Finalmente devuelve el punto al vendedor el cual le entrega el comprobante de boleta.

- Identificar el diagrama correspondiente al caso expuesto.

Solución guiada

El diagrama a utilizar para este ejercicio es un Diagrama de Secuencia. A continuación se explica algunas técnicas de análisis para una mayor rapidez en la elección.

Lo que se debe analizar para crear el diagrama correcto es lo siguiente:

- La forma de explicar el caso es contando una historia con palabras claves como esto “Comienza”, “posteriormente” y “Finalmente”, indicando secuencia en el caso.
- Los objetos nombrados: Punto, Vendedor, Cliente, Boleta.

Con esta información realizaremos un diagrama de Secuencia ya que están los elementos, mensajes y secuencia del caso.

Ejercicio propuesto (1)

Análisis de un Diagrama

Reconocer los componentes necesarios para realizar el diagrama de un software que lleve a cabo una venta de distintos tipos de casa a un cliente. Identificar qué tipo de diagrama se debe implementar.

El vendedor se identifica con:

- Nombre.
- Rut.
- Sucursal.
- Código vendedor.

Las casas pueden ser de dos tipos y tienen:

- Color.
- Metros cuadrados.
- Altura.

Y el cliente tiene:

- Nombre.
- Rut.
- País.
- Fecha de nacimiento.

Diagrama de Clases

Competencias

- 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.

Introducción

Conoceremos en profundidad lo que es un diagrama de clases y su mundo en la programación orientada a objetos, además, conoceremos una herramienta llamada StarUML que nos servirá para construir diagramas de clases.

Nos enfocaremos en describir una clase con sus atributos y operaciones para así tener una perspectiva descriptiva de lo que es una clase en diagrama de clases como en Java.

Existen otras herramientas para realizar diagramas como por ejemplo:

- Lucidchart
- diagrams.net
- argUML
- Visio, entre otras.

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.

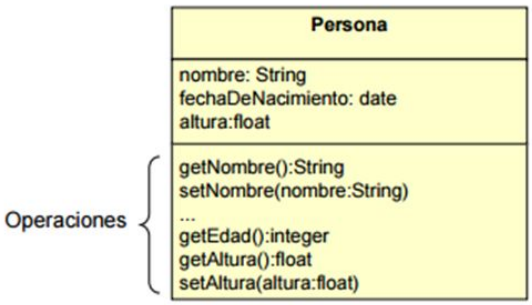

Imagen 7. Ejemplo diagrama de clases.
Fuente: Desafío Latam.

- **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)

Uso de la herramienta StarUML

Esta herramienta nos 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). Para efectos de uso en la unidad, la versión libre nos será suficiente.

Para instalar la herramienta, debemos ir a la siguiente URL: http://staruml.io/download

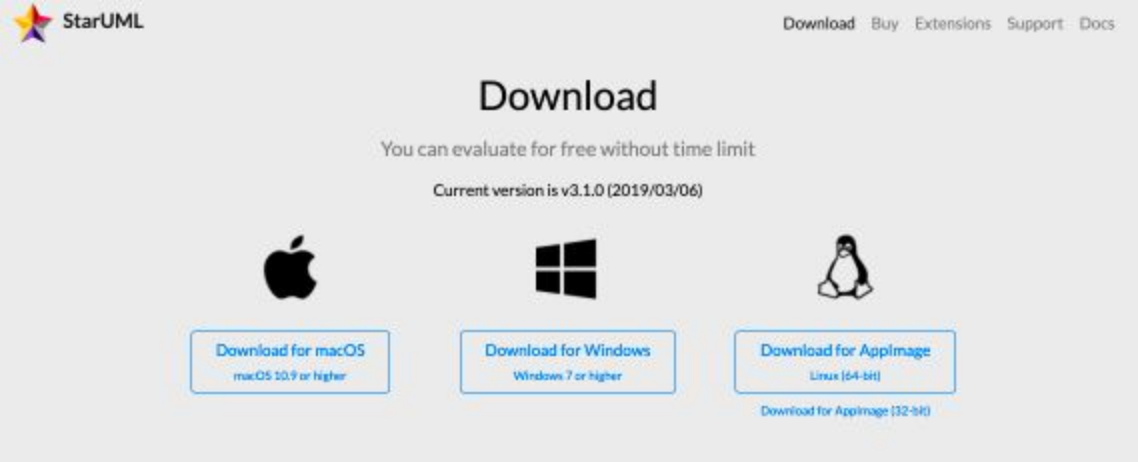

Imagen 8. Página de descarga StarUML.
Fuente: Desafío Latam.

Para crear un diagrama de clases realizaremos la siguiente acción:

Model -> Add Diagram -> Class Diagram

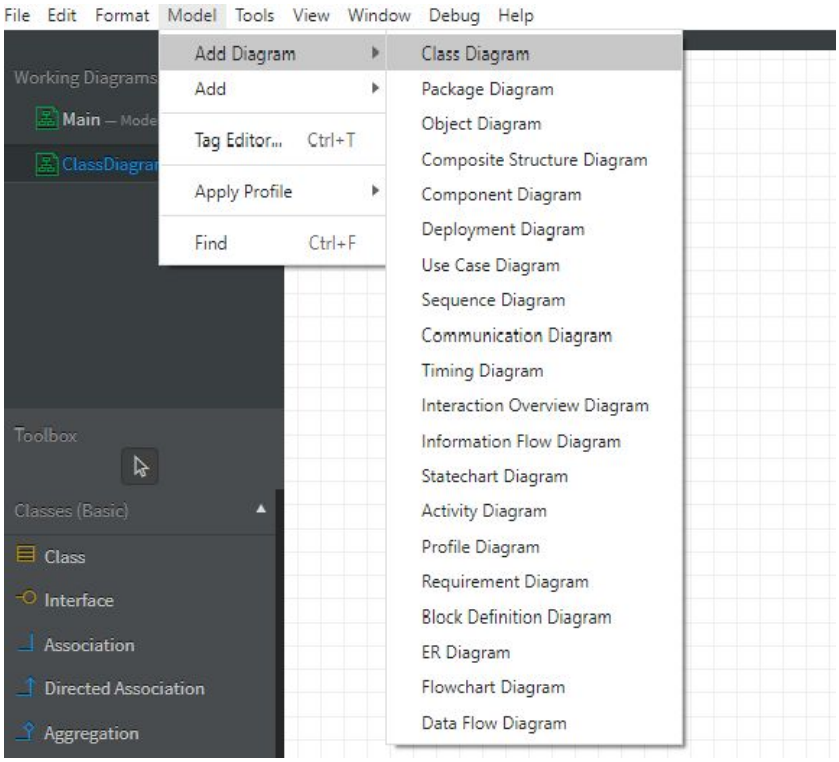

Imagen 9. Crear Diagrama de clases StarUML.
Fuente: Desafío Latam.

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

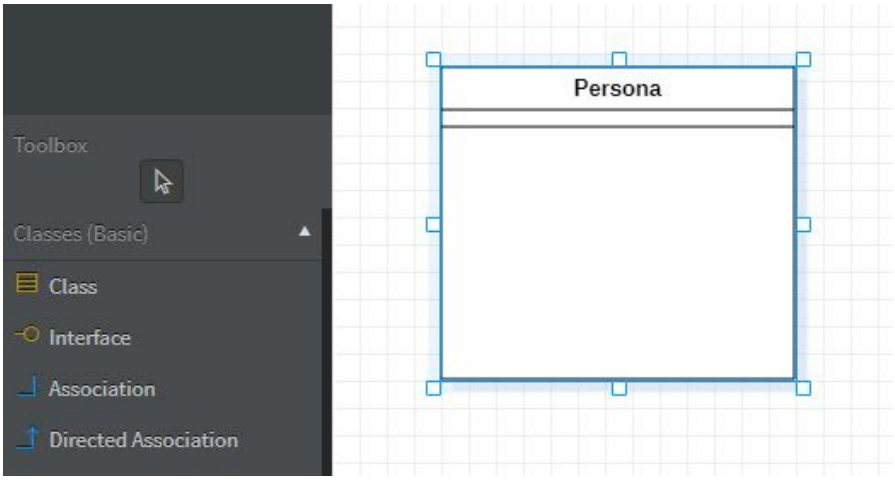

Imagen 10. Creación de clase.
Fuente: Desafío Latam.

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

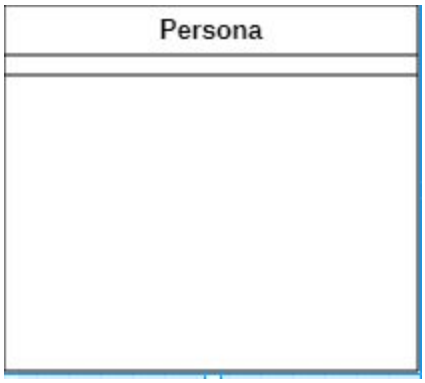

Imagen 11. Cambiar nombre de clase.
Fuente: Desafío Latam.

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

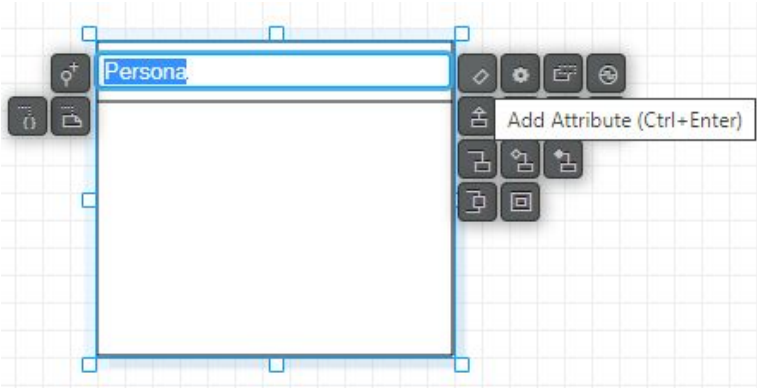

Imagen 12. Agregar Atributos.
Fuente: Desafío Latam.

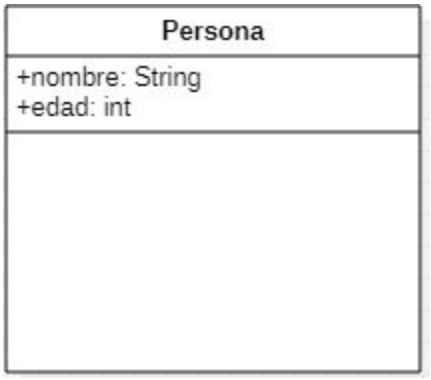

Imagen 13. Agregando atributos y tipo de dato.
Fuente: Desafío Latam.

Agregando acciones u operaciones:

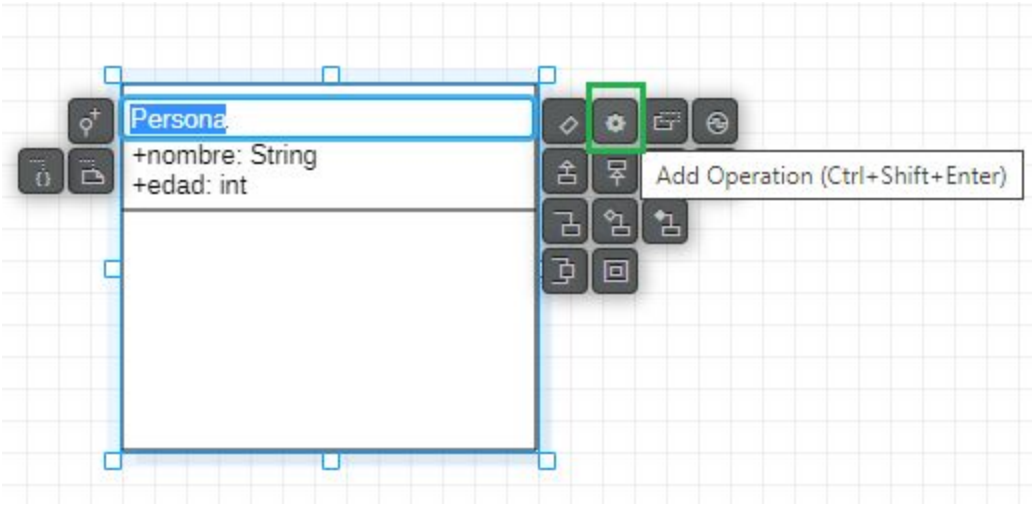

Imagen 14. Botón que permite agregar Operaciones o Acciones.
Fuente: Desafío Latam.

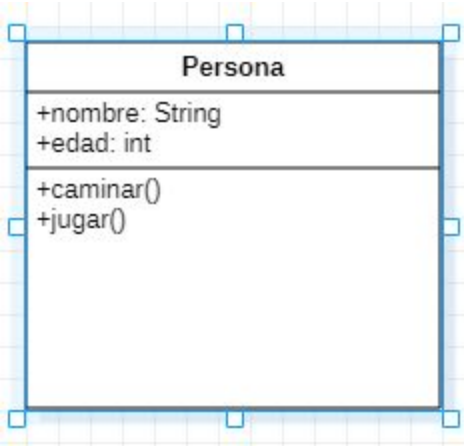

Imagen 15. Operaciones agregadas.
Fuente: Desafío Latam.

Reconocer una clase Java dentro de un diagrama de clase

Antes de comenzar a explicar qué es una clase en Java, primero explicaremos qué es un objeto en Java.

Un objeto en Java 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: El 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.

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 y, 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) - fecha**N**acimiento (nombre compuesto).

private int nombre
private int fechaNacimiento

Constructores: Siempre lleva el mismo nombre de la Clase, y puede recibir o no parámetros. Esto lo veremos en detalle en la unidad "Orientación a Objetos".

Operaciones: Las operaciones se forman por su tipo de acceso, público o privado, además sabremos si este retorna o no retorna algún tipo de dato.

El nombre del método siempre debe ser representado por el nombre de una acción, la nomenclatura del nombre del método siempre es con minúscula si este es un nombre simple; si el nombre del método es compuesto la segunda palabra del nombre del método debe comenzar con su primera letra mayúscula; si el nombre del método tiene dos o más palabras compuestas en su nombre cada palabra adicional debe comenzar con letra mayúscula.

El método u operación puede o no recibir parámetros de entradas.

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.

- 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  ;  }
```

Tomando el ejemplo anterior veremos una clase Java orientada a Diagrama de Clase.

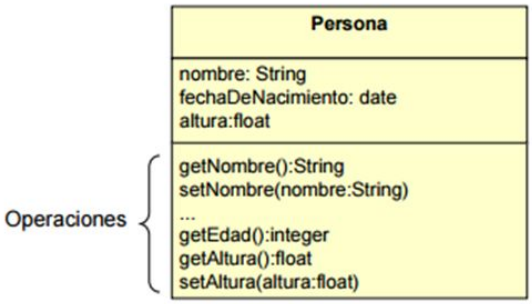

Imagen 16. Diagrama de clases orientado a objetos en Java.
Fuente: Desafío Latam.

Ejercicio guiado: Elementos de una clase

Crear un diagrama de clases para establecer elementos de una casa, la información que se tiene es la siguiente:

- Cocina
  - altura : double
  - cantidad hornillas : int
  - prenderHorno() : String
- Televisor
  - marca : String
  - definición : String
  - apagarTelevisor()
- Escritorio
  - tamaño : double
  - tipo : String

Solución ejercicio guiado

Para esto utilizaremos 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:

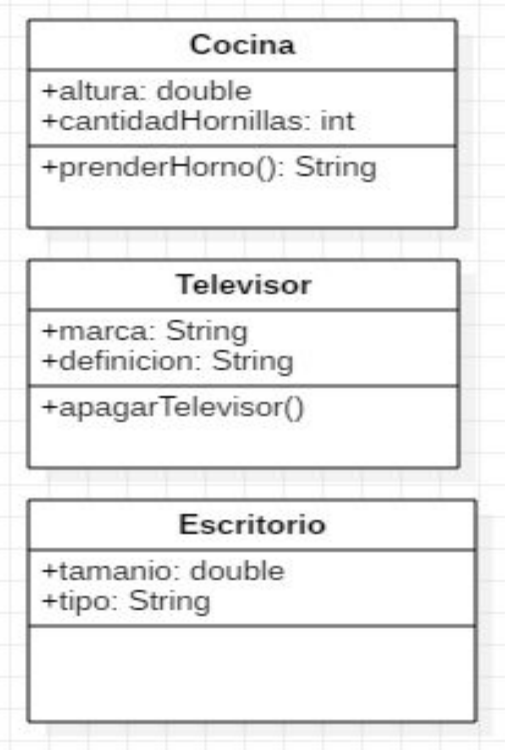

imagen 17 : Solución Ejercicio Guiado.
Fuente: Desafío Latam.

Ejercicio propuesto (2)

Crear un diagrama clases para identificar los distintos tipos de instrumentos de una banda de rock, la información que se tiene es la siguiente:

- **Guitarra**
  - tipo mástil: String
  - marca: String
- **Batería**
  - cantidad platillos: int
  - tipo pedal: int
  - cantidad Cajas: int
- **Parlantes**
  - tamaño: int
  - altura: double
  - marca: String

Soluciones Ejercicios propuestos

Solución ejercicio propuesto (1)

El diagrama a implementar es un **Diagrama de Clases** y para llegar a esta conclusión se necesitan las siguientes observaciones:

- Existen elementos con características.
- Se necesita realizar un software.
- No hay explicación de la secuencia de la venta por los actores o elementos.
- Las características están explícitas en el caso, por ejemplo: nombre, rut.

Solución ejercicio propuesto (2)

El modelo que propone el ejercicio quedaría de la siguiente manera:

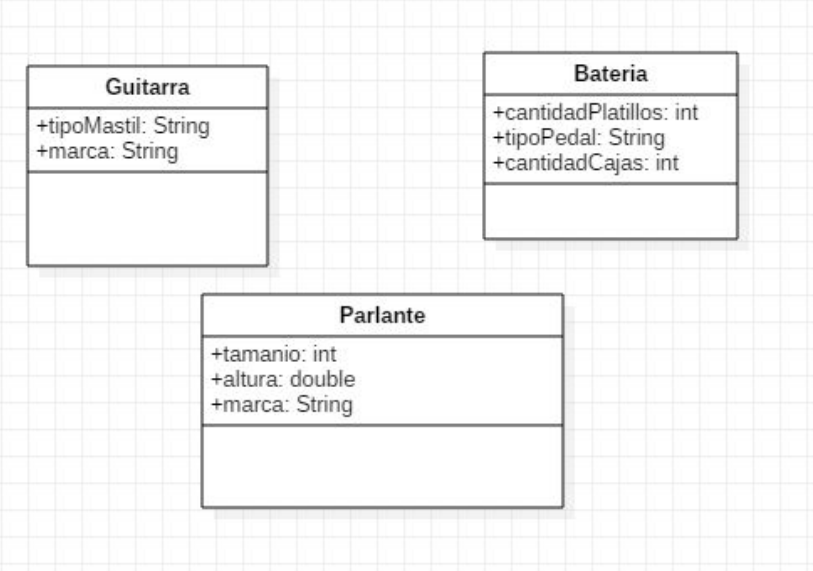

Imagen 18 : Solución Ejercicio Propuesto (2)
Fuente: Desafío Latam.