Clase grabada

Recibe una cálida:

¡Bienvenida!

Te estábamos esperando 😁

English Always | alkemy

Clase grabada

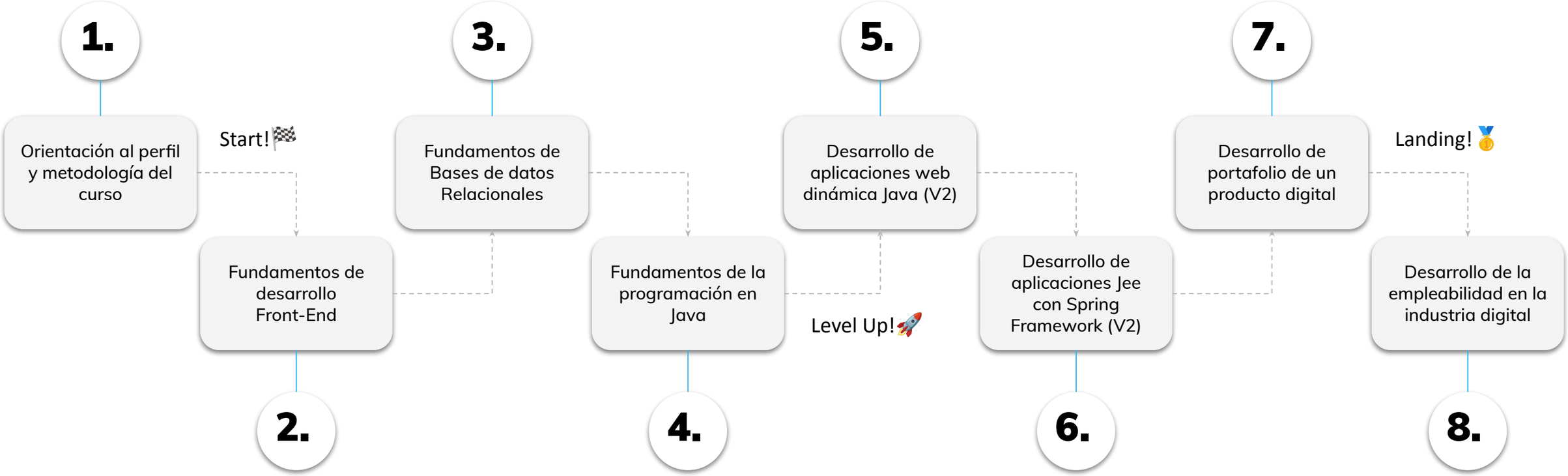
El Paradigma de Orientación a Objetos

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0

English Always | alkemy

HOJA DE RUTA

¿Cuáles **skill** conforman el programa?



English Always | alkemy

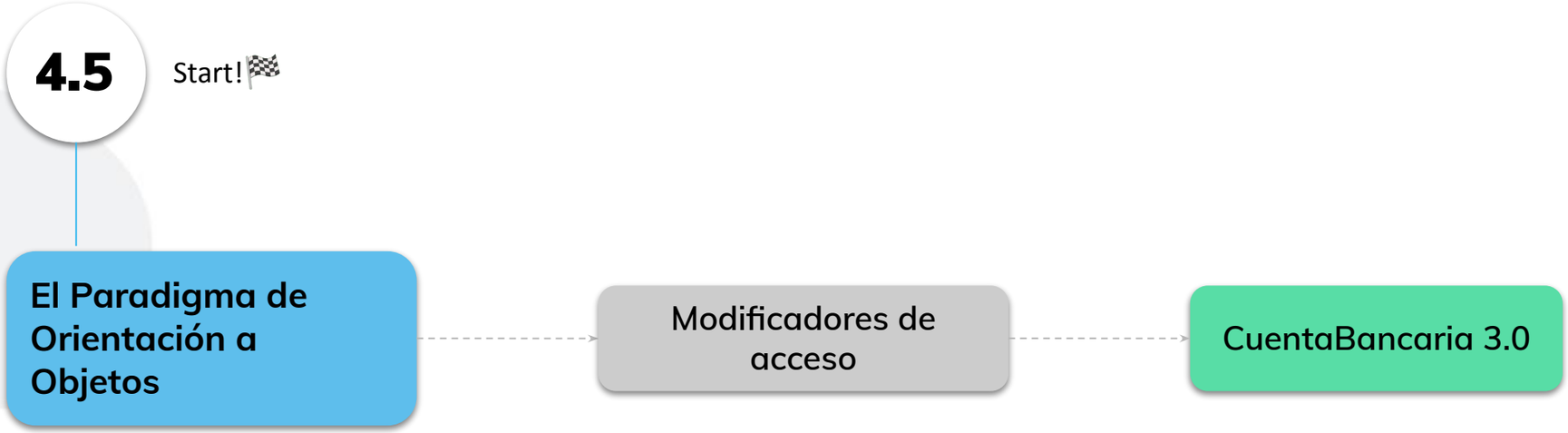
REPASO CLASE ANTERIOR

En la clase anterior trabajamos 📚:

- ✓ *Métodos constructores e instanciación de objetos*

English Always | alkemy

LEARNING PATHWAY



English Always | alkemy

OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?

- *Comprender el concepto de encapsulamiento*
- *Aprender la utilización de accesadores y mutadores*

Rompehielo🥶

¿Esto es una receta?: 🙌

Tenemos en nuestras manos una receta de cocina super especial! Lo que la hace tan especial, es que es **secreta**. Pero, podemos darte algunas pistas...
La receta lleva los siguientes ingredientes:

- Harina
- Huevos
- Leche
- Esencia sabor vainilla

¿Pueden adivinar qué vamos a cocinar? Levanten la mano o respondan en el chat!

Así como la receta completa está oculta, en POO podemos decidir qué información de un objeto mostrar y qué mantener oculta mediante el **encapsulamiento y los modificadores de acceso**.

Encapsulamiento

Encapsulamiento

¿Qué es?

El encapsulamiento **oculta** lo que hace un objeto de lo que hacen otros objetos y del mundo exterior, por lo que se denomina también **ocultación de datos**.

El encapsulamiento busca de alguna forma **controlar el acceso a los datos que conforman un objeto** o instancia, de este modo podríamos decir que una clase y por ende sus objetos que hacen uso de **modificadores de acceso** (especialmente privados) **son objetos encapsulados**.

Ocultar el estado interno y hacer que toda interacción sea a través de los métodos del objeto es un mecanismo conocido como **encapsulamiento de datos**.

Modificadores de acceso

¿Para qué se utilizan?

Permiten dar un **nivel de seguridad mayor** a nuestras aplicaciones **restringiendo el acceso** a diferentes atributos, métodos, constructores asegurándonos que el usuario deba seguir una "ruta" especificada por nosotros para acceder a la información.

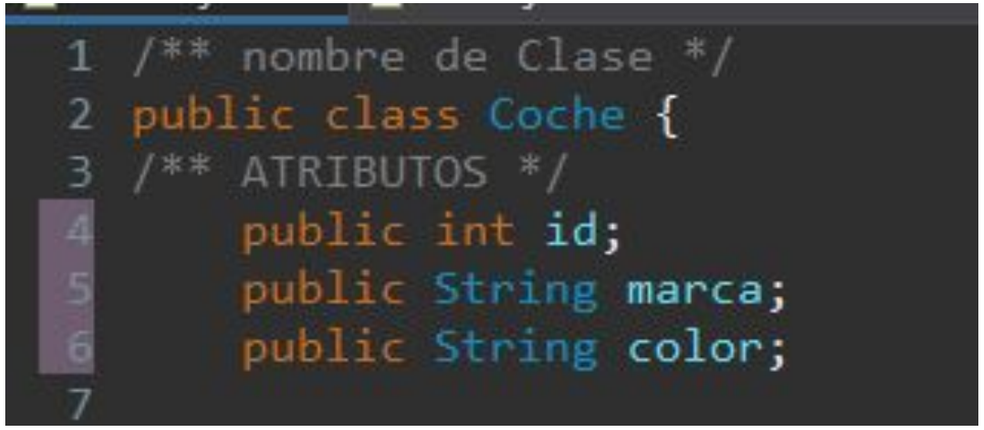
Todas las clases poseen diferentes niveles de acceso en función del modificador de acceso (**visibilidad**):

- **public**
- **private**
- **protected**

Modificadores de acceso

Modificador public:

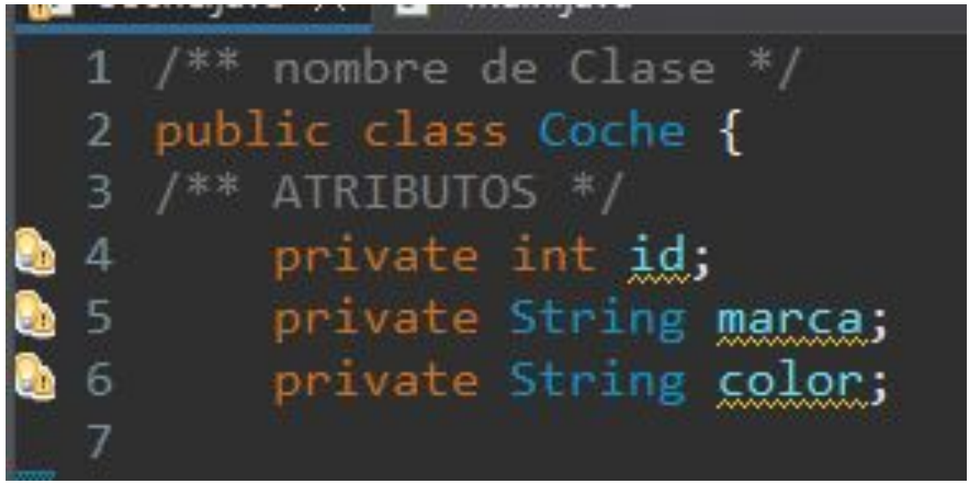
El modificador de acceso public es **el más permisivo** de todos, esto quiere decir que si un componente de una clase es public, tendremos **acceso a él desde cualquier clase** o instancia sin importar el paquete o procedencia de ésta.



Modificadores de acceso

Modificador private:

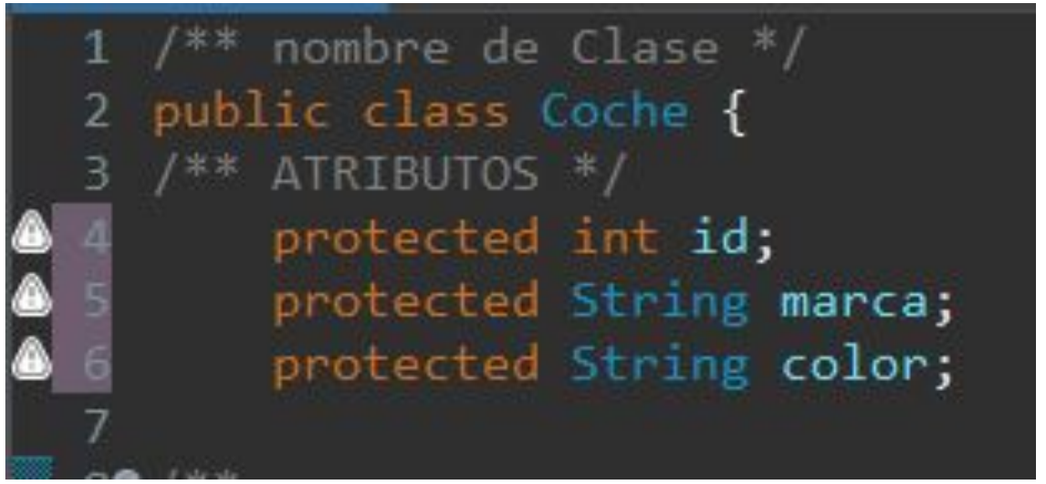
En Java **es el más restrictivo de todos**, básicamente cualquier elemento de una clase que sea privado puede ser accedido **únicamente por la misma clase**. Es decir, si por ejemplo, si un atributo es privado solo puede ser accedido por lo métodos o constructores que se encuentren en la misma clase. Ninguna otra clase, sin importar la relación que tengan, podrá tener acceso a ellos.



Modificadores de acceso

Modificador protected:

El modificador de acceso protected nos **permite acceso** a los componentes con dicho modificador desde la misma clase, clases **del mismo paquete y clases que hereden de ella** (incluso en diferentes paquetes).



Modificadores de acceso

No son únicamente para los atributos, sino también para los métodos y procedimientos específicos de cada clase. El comportamiento de los componentes será el mismo que explicamos anteriormente.

Nota: Siempre se recomienda que los atributos de una clase sean **privados** y por tanto cada atributo debe tener sus propios métodos de acceso y consulta (los veremos en la próxima clase).

Nota 2: Siempre que se use una clase de otro paquete, se debe importar usando **import**. Cuando dos clases se encuentran en el mismo paquete no es necesario hacer el import pero esto no significa que se pueda acceder a sus componentes directamente.

LIVE CODING

Ejemplo en vivo

Billetera Virtual:

En esta breve ejemplo, vamos a colocar los modificadores de acceso necesarios a cada atributo y método de la clase Cuenta, y vamos a ver cómo se comporta.

1. *Agregar modificador public y ejecutar el programa*
2. *Agregar modificador private y ejecutar el programa*
3. *Agregar modificador protected y ejecutar el programa*

Tiempo:

Evaluación Integradora ✨

¿Listos para un nuevo desafío? En esta clase comenzamos a construir nuestro....

Trabajo Integrador del Módulo 💪🏽

Iremos completándolo progresivamente clase a clase.

English Always | alkemy

Ejercicio N°

CuentaBancaria 3.0

CuentaBancaria 3.0

Manos a la obra: 🙌

Ahora vamos a poner en práctica lo aprendido modificando la visibilidad de los atributos y métodos de la clase CuentaBancaria con la que venimos trabajando.

Consigna: ✍️

- Colocar modificador private a los atributos
- Colocar modificador public a los métodos
- Crear un método público ‘ingresarDinero’ que permita ingresar un monto de dinero y sumarlo al atributo saldoActual.
- Imprimir los valores por pantalla

Tiempo🕐: *30 minutos*

¿Alguna consulta?

RESUMEN

¿Qué logramos en esta clase?

- ✓ ***Reconocer la utilidad e implementación de los modificadores de acceso***

English Always | alkemy

#WorkingTime

Continuemos ejercitando

¡Antes de cerrar la clase! Te invitamos a: 👇👇👇

1. Repasar nuevamente la grabación de esta clase
2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
   a. Material 1 *(Foro)*
   b. *Lectura Módulo 4, Lección 5: páginas 6 - 7*
3. Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.

¡Muchas Gracias!

Nos vemos en la próxima clase 👋🏻

English Always | alkemy

Momento:

Time-out!

⏰5 min.