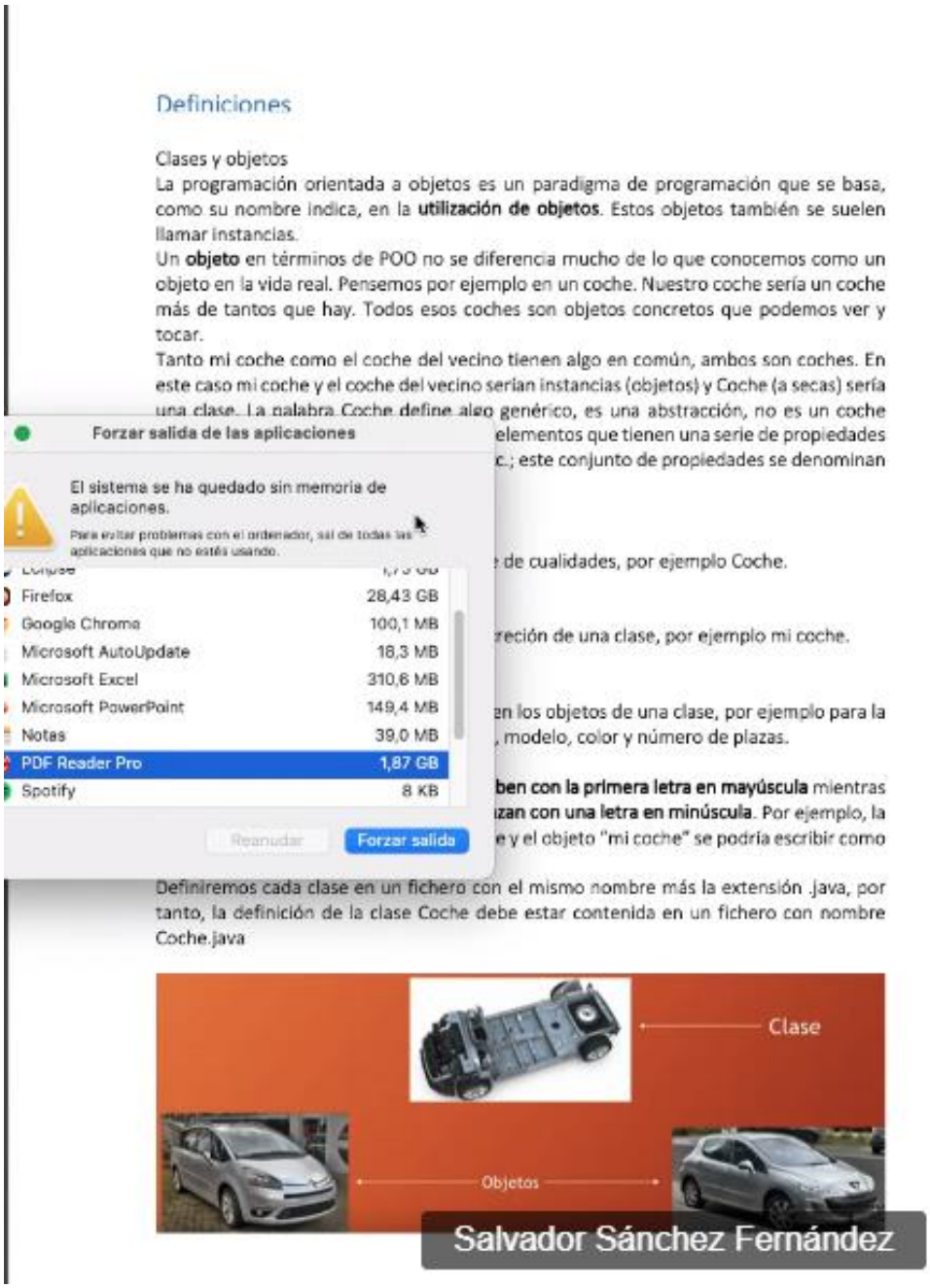


- **OBJETO** = INSTANCIA (minúsculas)
- **Plantilla** = en mayúscula (la primera)
- Propiedades = **atributos o variables de instancia**
- No tiene main.
- El main es otra clase distinta
- New class.java

Clase
Concepto abstracto que denota una serie de cualidades, por ejemplo Coche.

Instancia
Objeto palpable, que se deriva de la concreción de una clase, por ejemplo mi coche.

Atributos
Conjunto de características que comparten los objetos de una clase, por ejemplo para la clase coche tendríamos matrícula, marca, modelo, color y número de plazas.

En Java, los nombres de **las clases se escriben con la primera letra en mayúscula** mientras que **los nombres de las instancias comienzan con una letra en minúscula**. Por ejemplo, la clase coche se escribe en Java como Coche y el objeto "mi coche" se podría escribir como

En Java, los nombres de **las clases se escriben con la primera letra en mayúscula** mientras que **los nombres de las instancias comienzan con una letra en minúscula**. Por ejemplo, la clase coche se escribe en Java como Coche y el objeto "mi coche" se podría escribir como miCoche.
Definiremos cada clase en un fichero con el mismo nombre más la extensión .java, por tanto, la definición de la clase Coche debe estar contenida en un fichero con nombre Coche.java

EJEMPLO

Vamos a definir a continuación la clase Libro con los atributos isbn, autor, titulo y numeroPaginas.

```
/*Definición de la clase Libro*/
public class Libro {
// atributos
String isbn;
String titulo;
String autor;
int numeroDePaginas;
 }
```

A continuación creamos varios objetos de esta clase.

```
/*Programa que prueba la clase Libro*/
public class PruebaLibro {
        public static void main(String[] args) {
        Libro lib = new Libro();
        Libro miLibrito = new Libro();
        Libro quijote = new Libro();
        }
}
```

Hemos creado tres instancias de la clase libro: lib, miLibrito y quijote.
Las variables de instancia (atributos) determinan las cualidades de los objetos.

(protegido) y private (privado).
En la siguiente tabla se muestra desde dónde es visible/accesible un elemento (atributo o método) según el modificador que lleve.

	En la misma clase	En el mismo paquete	En una subclase	Fuera del paquete
private	✔	✖	✖	✖
protected	✔	✔	✔	✖
public	✔	✔	✔	✔
sin especificar	✔	✔	✖	✖

Figura Ámbito de los elementos de una clase.

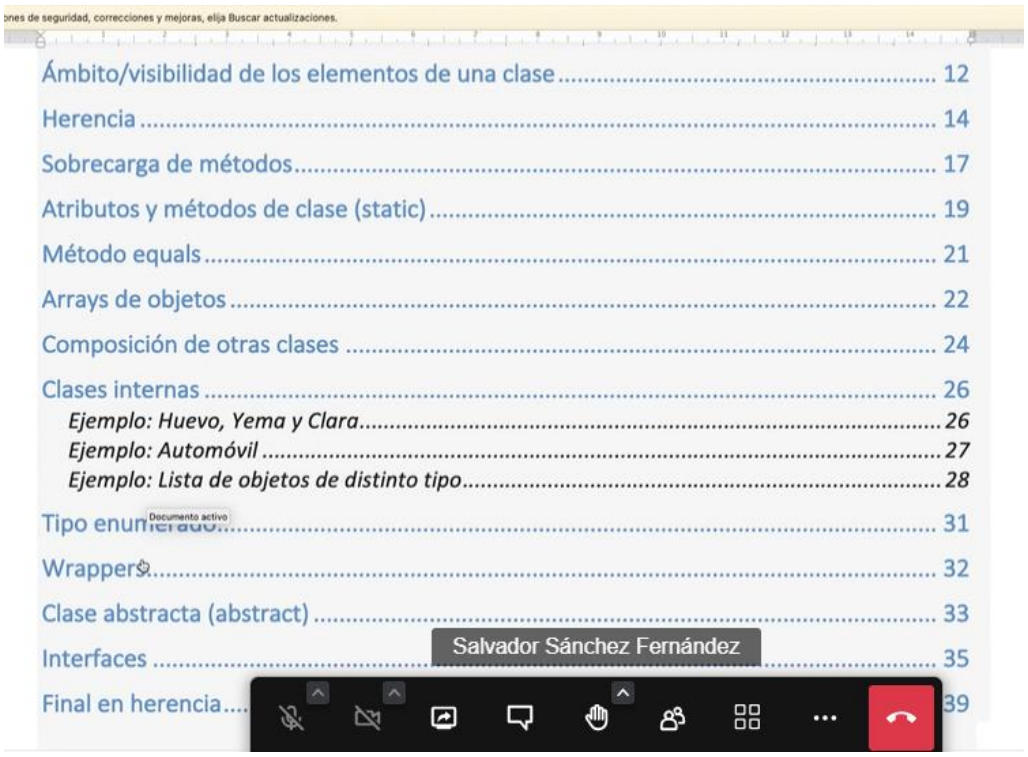
Por ejemplo, un atributo private solamente es accesible desde la misma clase, sin embargo, a un método protected se podrá acceder desde la misma clase donde esté definido, desde otro fichero dentro del mismo paquete o desde una subclase.
Como regla general, se suelen declarar private los atributos o variables de instancia y public los métodos.

```
/*Definición de la clase Animal */
public class Animal {

private String tipo;  //doméstico,salvaje
private String sexo;  //macho,hembra

public Animal (String tipo, String sexo) {
      this.tipo = tipo;
      this.sexo = sexo;
}

public Animal () {
      this.tipo = "salvaje";
      this.sexo = "macho";
}
```


ones de seguridad, correcciones y mejoras, elija Buscar actualizaciones.
Ámbito/visibilidad de los elementos de una clase ... 12
Herencia ... 14
Sobrecarga de métodos ... 17
Atributos y métodos de clase (static) ... 19
Método equals ... 21
Arrays de objetos ... 22
Composición de otras clases ... 24
Clases internas ... 26
Ejemplo: Huevo, Yema y Clara ... 26
Ejemplo: Automóvil ... 27
Ejemplo: Lista de objetos de distinto tipo ... 28
Documento activo
Tipo enumerado ... 31
Wrappers ... 32
Clase abstracta (abstract) ... 33
Salvador Sánchez Fernández
Interfaces ... 35
Final en herencia ... 39