Internet Applications

JAMOUM UNIVERSITY COLLEGE – COMPUTER SCIENCE DEPARTMENT

UMM AL-QURA UNIVERSITY

I. AMAL ALSHOMRANI

PHP Classes and Objects

CHAPTER 10

Object-Oriented Overview

SECTION 1 OF 3

Overview

Object-Oriented Overview

PHP is a full-fledged object-oriented language with many of the syntactic constructs popularized in languages like Java and C++.

Earlier versions of PHP do not support all of these object-oriented features,

- PHP versions after 5.0 do

Terminology

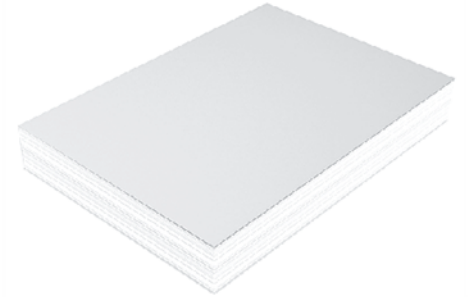
Object-Oriented Terminology

The notion of programming with objects allows the developer to think about an item with particular **properties** (also called attributes or **data members**) and methods (functions).

The structure of these **objects** is defined by **classes**, which outline the properties and methods like a blueprint.

Each variable created from a class is called an object or **instance**, and each object maintains its own set of variables, and behaves (largely) independently from the class once created.

Relationship between Class and Objects



Book class

Defines properties such as:
title, author, and number of pages

Objects (or instances of the Book class)

Each instance has its own title, author, and number of pages property values

UML
The Unified Modelling Language

The standard diagramming notation for object-oriented design is **UML (Unified Modeling Language).**

Class diagrams and object diagrams, in particular, are useful to us when describing the properties, methods, and relationships between classes and objects.

For a complete definition of UML modeling syntax, look at the Object Modeling Group's living specification

UML Class diagram

By example

Every Artist has a

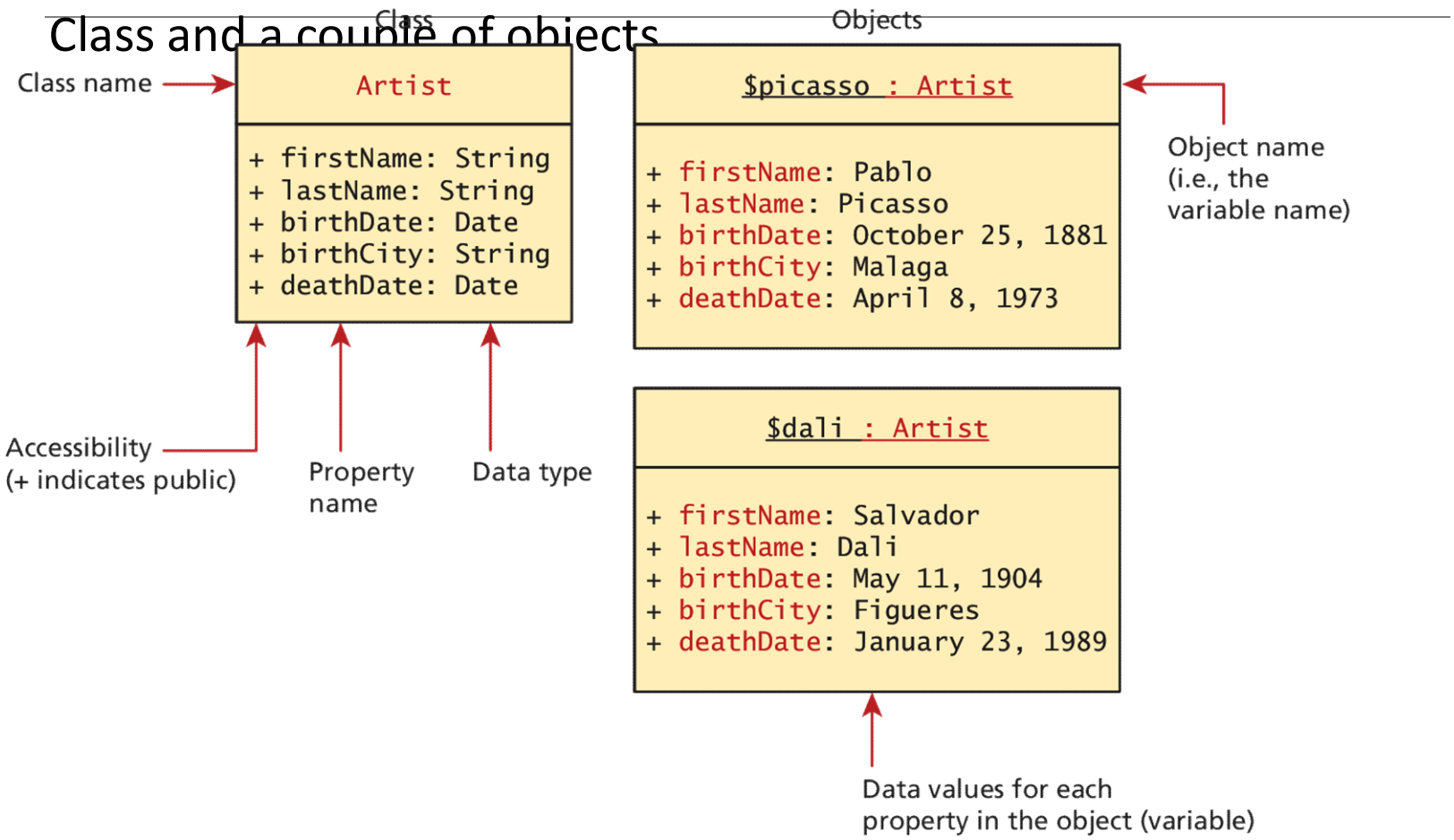
- first name,
- last name,
- birth date,
- birth city, and
- death date.

Using objects we can encapsulate those properties together into a class definition for an Artist.

UML articulates that design

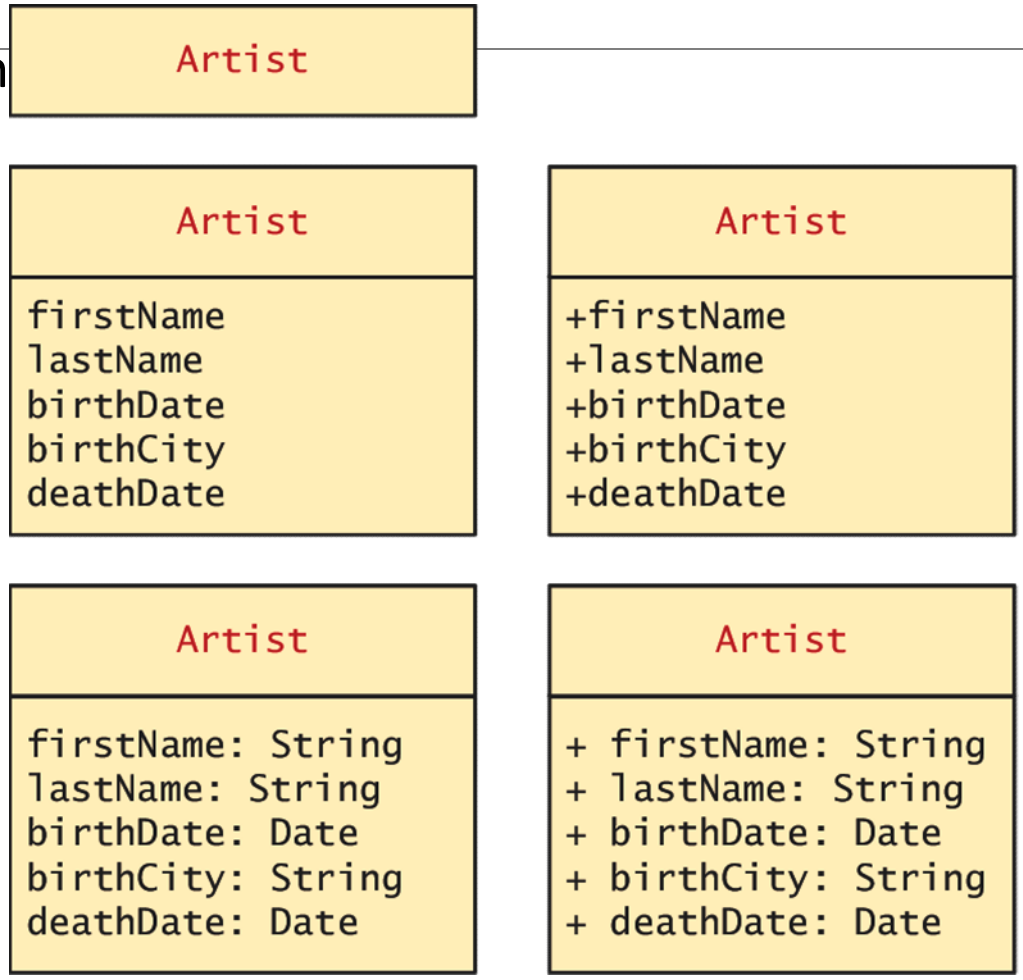
UML Class diagram

Class and a couple of objects



UML Class diagram

Differen



Server and Desktop Objects

Not the same

One important distinction between web programming and desktop application programming is that the objects you create (normally) only exist until a web script is terminated. While desktop software can load an object into memory and make use of it for several user interactions, a PHP object is loaded into memory only for the life of that HTTP request.

We must use classes differently than in the desktop world, since the object must be recreated and loaded into memory

Unlike a desktop, there are potentially many thousands of users making requests at once, so not only are objects destroyed upon responding to each request, but memory must be shared between many simultaneous requests, each of which may load objects into memory or each request that requires it

Objects and Classes in PHP

SECTION 2 OF 3

Defining Classes

In PHP

The PHP syntax for defining a class uses the class keyword followed by the class name and { } braces

```
class Artist {
   public   $firstName;
   public   $lastName;
   public   $birthDate;
   public   $birthCity;
   public   $deathDate;
}
```

LISTING 10.1 A simple Artist class

Instantiating Objects

In PHP

Defining a class is not the same as using it. To make use of a class, one must **instantiate** (create) objects from its definition using the *new* keyword.

$picasso = new Artist();

$dali = new Artist();

Properties

The things in the objects

Once you have instances of an object, you can access and modify the properties of each one separately using the variable name and an arrow (->).

```
$picasso = new Artist();
$dali = new Artist();
$picasso->firstName = "Pablo";
$picasso->lastName = "Picasso";
$picasso->birthCity = "Malaga";
$picasso->birthDate = "October 25 1881";
$picasso->deathDate = "April 8 1973";
```

LISTING 10.2 Instantiating two Artist objects and setting one of those object's properties

Constructors

A Better way to build

Constructors let you specify parameters during instantiation to initialize the properties within a class right away.

In PHP, constructors are defined as functions (as you shall see, all methods use the function keyword) with the name **__construct().** (Note: there are two underscores _ before the word construct.)

Notice that in the constructor each parameter is assigned to an internal class variable using the $this-> syntax. you **must** always use the $this syntax to reference all properties and methods associated with this particular instance of a class.

Constructors

An Example

```
class Artist {
   // variables from previous listing still go here
   ...

   function __construct($firstName, $lastName, $city, $birth,
                        $death=null) {
      $this->firstName = $firstName;
      $this->lastName = $lastName;
      $this->birthCity = $city;
      $this->birthDate = $birth;
      $this->deathDate = $death;
   }
}
```

LISTING 10.3 A constructor added to the class definition

Notice as well that the $death parameter in the constructor is initialized to null; the rationale for this is that this parameter might be omitted in situations where the specified artist is still alive.

Constructors

Using the constructor

```
$picasso = new Artist("Pablo","Picasso","Malaga","Oct 25,1881","Apr 8,1973");

$dali = new Artist("Salvador","Dali","Figures","May 11 1904", "Jan 23 1989");
```

Methods

Functions In a class

Methods are like functions, except they are associated with a class.

They define the tasks each instance of a class can perform and are useful since they associate behavior with objects.

To output the artist, you can use the reference and method name as follows:

```
$picasso = new Artist( . . . )

echo $picasso->outputAsTable();
```

Methods

The example definition

For our artist example one could write a method to convert the artist's details into a string of formatted HTML

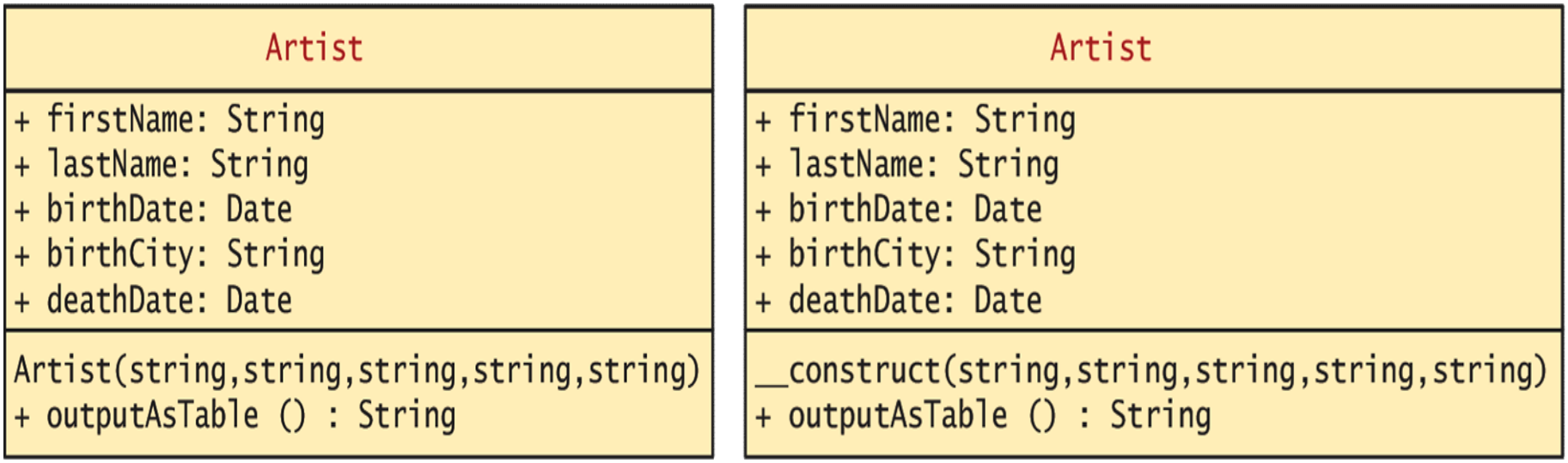
```
class Artist {
   ...
   public function outputAsTable() {
     $table = "<table>";
     $table .= "<tr><th colspan='2'>";
     $table .= $this->firstName . " " . $this->lastName;
     $table .= "</th></tr>";
     $table .= "<tr><td>Birth:</td>";
     $table .= "<td>" . $this->birthDate;
     $table .= "(" . $this->birthCity . ")</td></tr>";
     $table .= "<tr><td>Death:</td>";
     $table .= "<td>" . $this->deathDate . "</td></tr>";
     $table .= "</table>";
     return $table;
   }
}
```

LISTING 10.4 Method definition

Methods

UML class diagrams adding the method

Notice that two versions of the class are shown in this Figure, to illustrate that there are different ways to indicate a PHP constructor in UML.



Methods

NOTE

Many languages support the concept of overloading a method so that two methods can share the same name, but have different parameters. While PHP has the ability to define default parameters, no method, including the constructor, can be overloaded!

Visibility

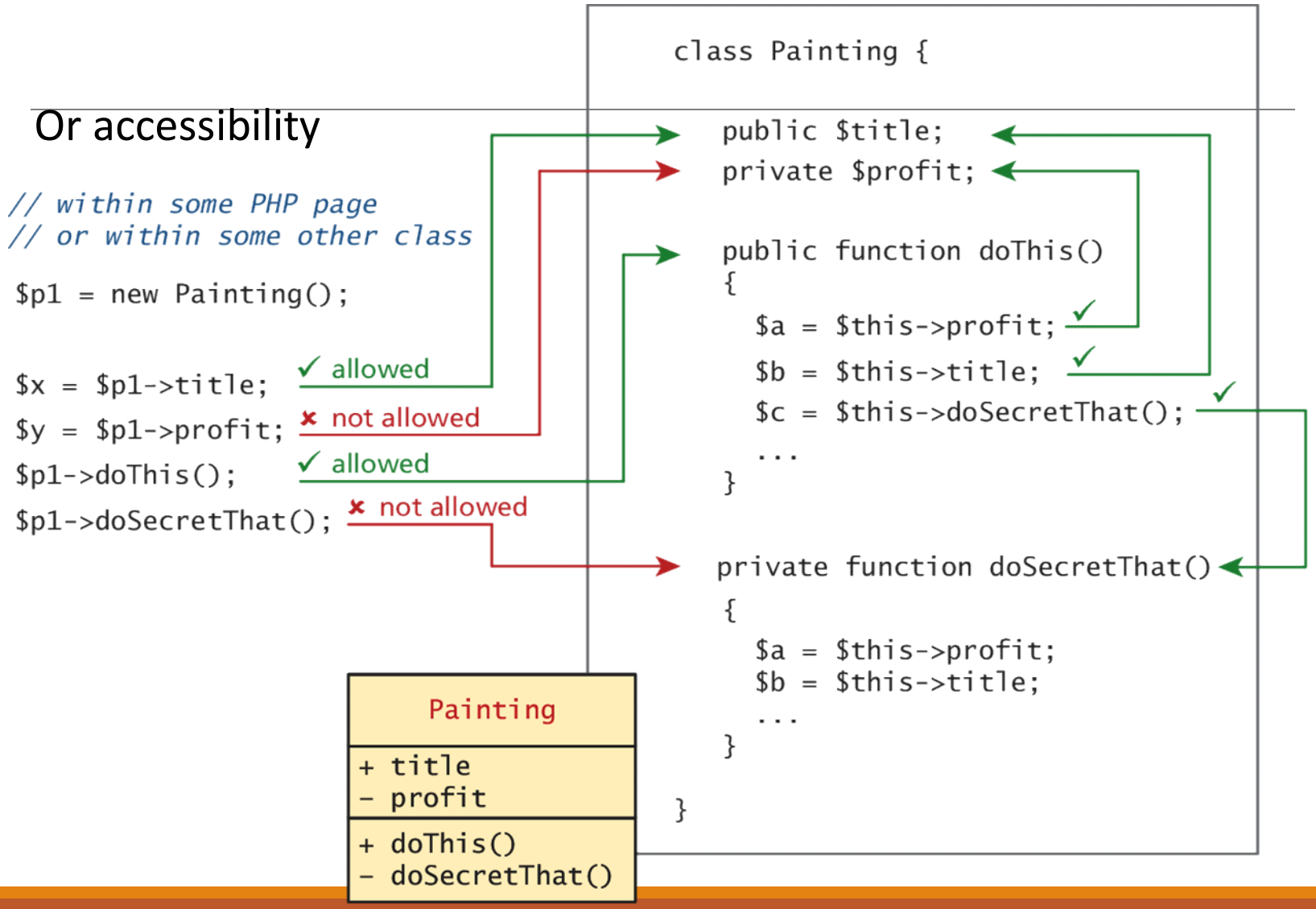
Or accessibility

The **visibility** of a property or method determines the accessibility of a **class member** and can be set to:

- **Public** the property or method is accessible to any code that has a reference to the object
- **Private** sets a method or variable to only be accessible from within the class
- **Protected** is related to inheritance…
- In UML, the "+" symbol is used to denote public properties and methods, the "–" symbol for private ones, and the "#" symbol for protected ones.

Visibility

Or accessibility



Static Members

A **static** member is a property or method that all instances of a class share.

Unlike an instance property, where each object gets its own value for that property, there is only one value for a class's static property.

Static members use the self:: syntax and are not associated with one object

They can be accessed without any instance of an Artist object by using the class name, that is, via **Artist::$artistCount.**

Static Members

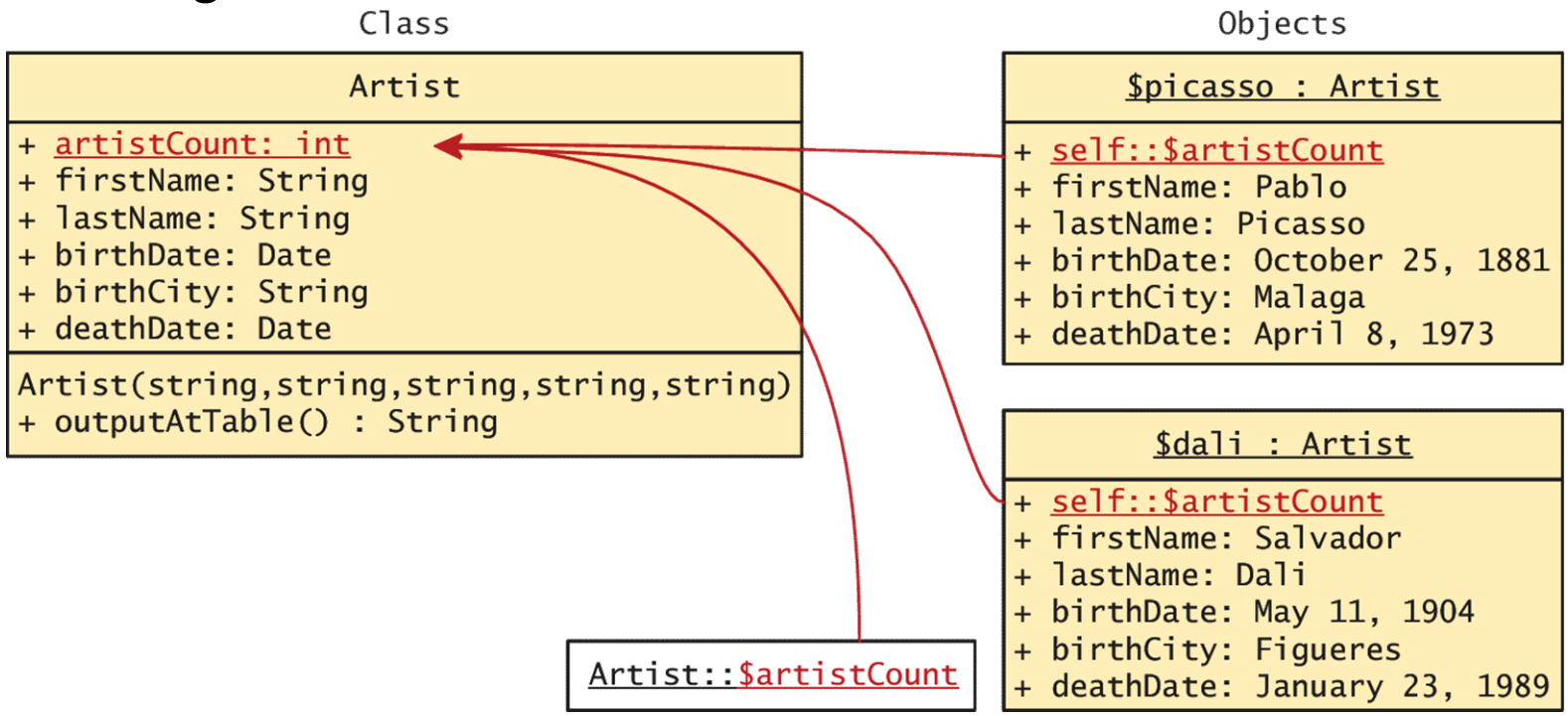
```
class Artist {
    public static $artistCount = 0;
    public   $firstName;
    public   $lastName;
    public   $birthDate;
    public   $birthCity;
    public   $deathDate;

    function __construct($firstName, $lastName, $city, $birth,
                         $death=null) {
       $this->firstName = $firstName;
       $this->lastName = $lastName;
       $this->birthCity = $city;
       $this->birthDate = $birth;
       $this->deathDate = $death;
       self::$artistCount++;
    }
 }
```

LISTING 10.5 Class definition modified with static members

- To illustrate how a static member is shared between instances of a class, we will add the static property artistCount to our Artist class, and use it to keep a count of how many Artist objects are currently instantiated.
- Notice that you do not reference a static property using the $this-> syntax, but rather it has its own self:: syntax.

Static Members

Uml again



Class constants

Never changes

Constant values can be stored more efficiently as class constants so long as they are not calculated or updated

They are added to a class using the **const** keyword.

```
const EARLIEST_DATE = 'January 1, 1200';
```

Unlike all other variables, constants don't use the $ symbol when declaring or using them.

Accessed both inside and outside the class using

- **self::EARLIEST_DATE** in the class and
- **classReference::EARLIEST_DATE** outside.