OOP

Object-oriented programming

Data modeling

- Relational databases: tables, columns and rows
- Procedural programming: variables and functions
- OOP: objects containing data and behavior

Relational databases

- Entities are represented as rows in a table, and their properties as fields
- Functionality is provided by the database and used via queries

Procedural

```
$ball1_position_x = 15;
$ball1_position_y = 10;
$ball1_movement_x = 0;
$ball1_movement_y = 0;

$ball2_position_x = 20;
$ball2_position_y = 10;
$ball2_movement_x = 0;
$ball2_movement_y = 0;

$ball3_position_x = 25;
$ball3_position_y = 10;
$ball3_movement_x = 0;
$ball3_movement_y = 0;

        ...

function collision ($firstBallX, $secondBallX, $firstBallY, $secondBallY, $firstBallMovX, $secondBallMovX, $firstBallMovY,
$secondBallMovY)
    {
        ...
    }
```

OOP

```
class BilliardBall {
    public $x, $y, $movementX, $movementY;

    public function __construct($positionX = 0, $positionY = 0)
    {
        $this->x = $positionX;
        $this->y = $positionY;
         $this->movementX = 0;
         $this->movementY = 0;
    }

    public function collision($otherBall)
    {
        ...
    }
}

$ball1 = new BilliardBall(15, 10);
$ball2 = new BilliardBall(20, 10);
$ball3 = new BilliardBall(25, 10);
```

So what is OOP?

- Object-oriented programming is a programming paradigm based on working with arbitrary data structures we call objects.
- To a computer “objects” mean nothing - OOP is meant to make understanding easier for humans.

Class and Object

Objects?

- Entities in programming that are representations of real-world entities
- Can represent physical entities like cars, computers etc., or abstract ones like files/folders, HTTP requests or documents.
- Variables of a custom type, which we define ourselves
- That custom type is called *Class*

How do we create an object?

- We define a class, which is a “blueprint” for the object
- We **instantiate** the class using the **new** keyword

```
class Person {}

$marko = new Person(“Marko”, “Markovic”);

 var_dump($marko);
```

What is a class?

- A custom type that we define for our needs
- It defines the data and behavior of objects
- It defines the way objects can interact with each other
- Everyday speech - class means kind, sort, category
- A template for creating new objects
- Objects of the same class have something in common

Exercise I - difference between objects and classes

What can be considered as a class and what as an object?

- Person
- Vivify Ideas
- City
- Marko Marković
- Novi Sad
- Company

Write down your answers.

Exercise I - difference between objects and classes

```
$markoMarkovic = new Person('Marko', 'Markovic');
$noviSad = new City('Novi Sad', 21000);
$vivifyIdeas = new Company('VivifyIdeas');


$markoMarkovic->setCity($noviSad);
$markoMarkovic->setCompany($vivifyIdeas);
```

Exercise II - create class

1. Create a script “vezba.php”
2. Create class **City**
3. Create the object **$noviSad** from the class **City**

Check slide 9 for help!

Exercise III - class identification

Let’s now identify classes and objects in this classroom.

Write down your thoughts!

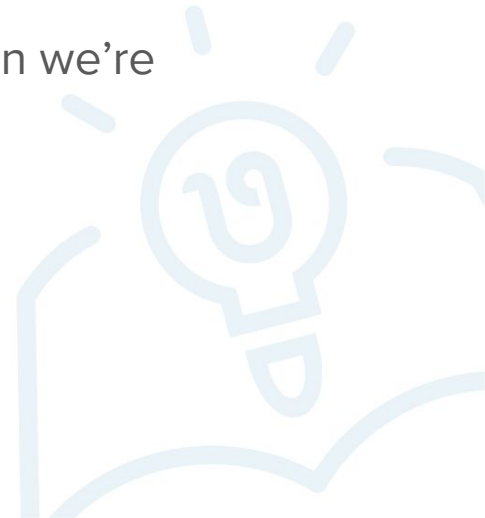
Attributes

Attributes

```
class Person {
        public $firstName;
        public $lastName;
        ...
}
```

Attributes

- Also called properties, fields etc.
- They contain the object data
- They contain the current state of the object
- They can be simple data types like numbers or strings, or they can be objects themselves
- The data we choose to represent depends on the application we're developing



Exercise IV - adding attributes

1. Add attributes **name** and **population** to class **City**

Check previous slides for help

Assign value to attributes

```
class Person {
        public $name;
        public $age;
}

$joe = new Person();
$joe->name = ‘Joe’; // Setting the property “name” on object “$joe”
$joe->age = 30; // Setting the property “age” on object “$joe”

$bob = new Person();
$bob->name = ‘Bob’;
$bob->age = 35;

var_dump($joe);
var_dump($bob);
```

Exercise V - assign value to attributes

1. Assign values “Novi Sad” and 200000 to attributes **name** and **population** in object **$noviSad**

Check previous slides for help

Methods

Methods

```
class Person {
        public $firstName;
        public $lastName;
        public $attribute3;
        public $attribute4;

        public function walk() {
                ...
        }

        public function speak() {
                ...
        }

}
```

Methods

- The other important part of OOP
- Methods are functions bound to an object of a certain class
- They are used to change the state of an object.
- Objects would be only data structures without methods

Calling methods

```
class Person {
	public $name;
	public function sayHello() {
		echo 'Hello!<br>';
	}
}

$joe = new Person();
$joe->sayHello(); // Calling the method “setName” on object “joe”
```

Exercise VI - methods

1. Add method **getName** to class **City.** It should just echo string *“Method getName is called”*
2. Call that method on object **$noviSad**

Check previous slides for help

Access modifiers

Public vs private

- Access modifiers
- If a property is public, it’s accessible from anywhere
- Private properties are accessible only from inside the class. They are “hidden” from the outside
- This is not a security measure - it exists to help prevent errors when programming.

Public vs private

```
class Person {
	public $firstName;
	public $lastName;

	private $id;

	public function setId ($newId) {
		$this->id = $newId;
	}
}
```

Public vs private

```
class Person {
	private $name;
	public $age;

	public function getName() {
		echo 'Calling getName method<br>';
	}

}

$joe = new Person();
$joe->age = 30;
$jos->getName();
$joe->name = ‘Joe’; // Trying to access the private attribute “name” on object “$joe”

var_dump($joe);
```

Exercise VII - access modifiers

1. Set **private** access modifier to attribute **name**
2. Try to assign value to attribute **name** of the object **$noviSad**. You should get an **error**!
3. Change the access modifier to **public**
4. Try to execute the script again
5. Set **public** access modifier to method **getName**
6. Call that method on object **$noviSad**

Check previous slides for help

Why use private?

- To prevent accidental changes of data that needs to be changed in a controlled way
- When creating a class for others to use, there’s no need to “expose” internal methods and properties
- Both properties and methods can be private
- We don’t access these properties or methods directly, we use public properties and methods to change the state in a controlled way

$this

The special variable $this

- Only available inside methods
- Refers to the current object, i.e. the object that the method was called on

Using $this

```
class Person {
        private $name;
        public $age;

        public function setName($text) {
                $this->name = $text;
                echo 'Name: ' . $this->name;
        }
}

$joe = new Person();
$joe->setName("Joe");

var_dump($joe);
```

Exercise VIII - using $this

1. Inside the method **getName** implement returning of the attribute **name**
2. Call the method on object **$noviSad** and display returned value
3. Add method **setName** that will accept one parameter **$newName**
4. Inside the method **setName** implement assigning of the parameter **$newName** to the attribute **name**

Check previous slides for help

The special variable $this

- $this is the object that the method is called on, i.e. it’s the object left of the arrow ->
- Otherwise than having $this, methods are like ordinary functions, and you can do anything inside them, including creating other objects, of other classes or even of that same class
- $this is an object like any other, it can be passed to functions and returned from a function, but nothing can be assigned to directly ($this = 'x' is not allowed!)

Constructor

The __construct method

- A constructor is a special method, called when creating an object
- Used to initialize an object, especially in cases where uninitialized objects can be problematic
- In other languages they are usually named the same as the class

Using __construct

```
class Person {
	public $name;
	public $age;

	public function __construct($name, $age) {
		$this->name = $name;
		$this->age = $age;
	}
}

$obj = new Person(“Marko”, 35);

var_dump($obj->name, $obj->age);
```

Exercise IX - using __construct

1. Add constructor to class **City**. Add two parameters in constructor: **name** and **population**
2. Change the creation of the **$noviSad** object to use two constructor parameters
3. In the constructor, implement assigning values from provided parameters **name** and **population** to attributes **name** and **population**
4. Remove assigning value to attributes **name** and **population** outside of the class
5. Call **getName** function and echo the return value

Check previous slides for help