



A Business Case:

~ Balloons On Sale ~



To illustrate concepts of Classes and Objects (using PHP)

LAU Yi Meng
IS113 (G10)

Business Scenario:

I am selling balloons.

The balloons are of different:

- Sizes (**S**, **M**)
- Colors (**red**, **gold**)

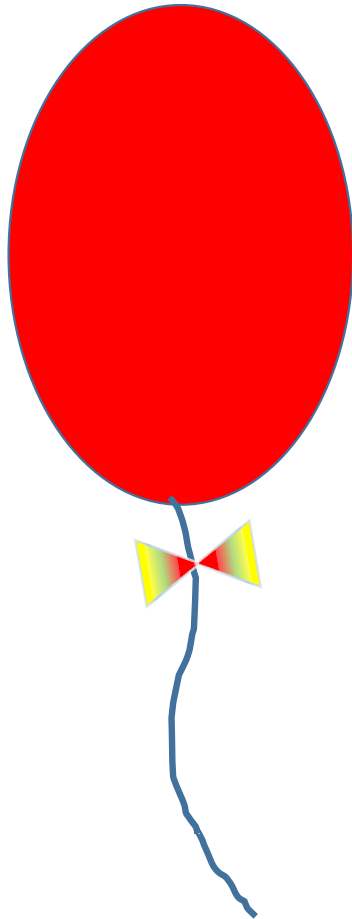
Each of them will be priced differently.

I need a simple web page to show these prices to my customers.

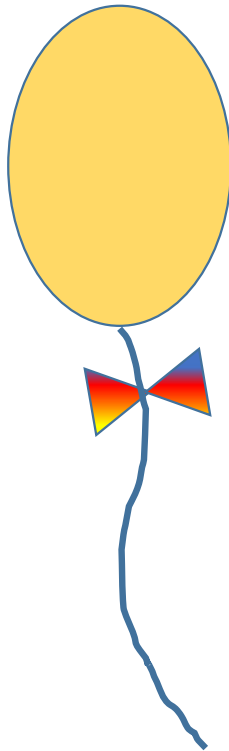


Number of Balloon types I am selling → 4

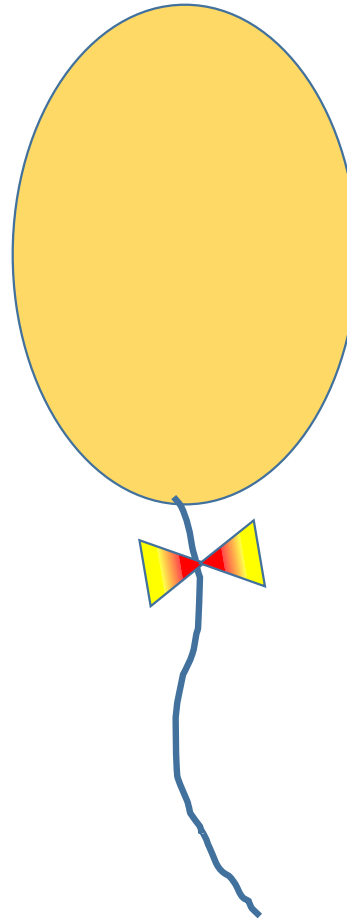
Color : Red
Size : Medium
Price : 4



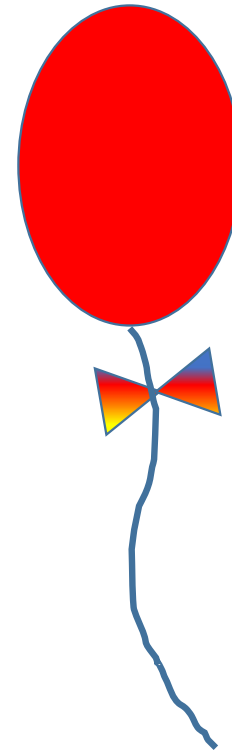
Color : Gold
Size : Small
Price : 3



Color : Gold
Size : Medium
Price : 5



Color : Red
Size : Small
Price : 2



How can I represent balloons in PHP?

There are a few ways...

Solution #1

<?php

```
$balloon1_color = 'Red';  
$balloon1_size  = 'S';  
$balloon1_price = 2;  
$balloon1_desc  = 'Red S';
```

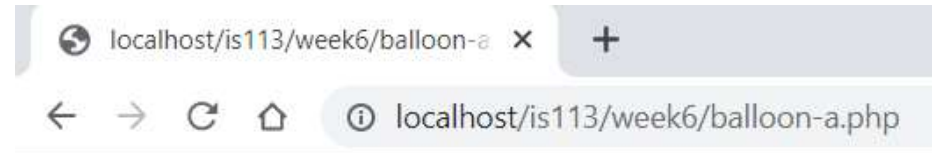
```
$balloon2_color = 'Red';  
$balloon2_size  = 'M';  
$balloon2_price = 4;  
$balloon2_desc  = 'Red M';
```

```
$balloon3_color = 'Gold';  
$balloon3_size  = 'S';  
$balloon3_price = 3;  
$balloon3_desc  = 'Gold S';
```

```
$balloon4_color = 'Gold';  
$balloon4_size  = 'M';  
$balloon4_price = 5;  
$balloon4_desc  = 'Gold M';
```

?>

Source: balloon-a.php



Balloons On Sale

Color	Size	Price (\$)	Desc
Red	S	2	Red S
Red	M	4	Red M
Gold	S	3	Gold S
Gold	M	5	Gold M

Works for small
data set

```
<?php

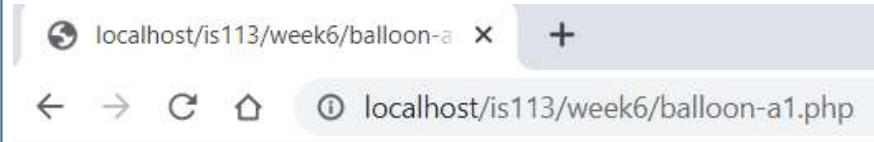
/* converting them into an array */

$balloons = [
    'balloon1' => [ 'Red' , 'S', 2, 'Red S' ],
    'balloon2' => [ 'Red' , 'M', 4, 'Red M' ],
    'balloon3' => [ 'Gold', 'S', 3, 'Gold S' ],
    'balloon4' => [ 'Gold', 'M', 5, 'Gold M' ]
];

?>
```

Source: balloon-a1.php

Using Associative Arrays



Balloons On Sale

Color	Size	Price (\$)	Desc
Red	S	2	Red S
Red	M	4	Red M
Gold	S	3	Gold S
Gold	M	5	Gold M

Business Scenario:

3 months later ...

My business is growing !!!

The balloons are now available in:

- Sizes (**S**, **M**, **L**)
- Colors (**red**, **gold**, **green**)
- Gas (**helium**, **oxygen**)

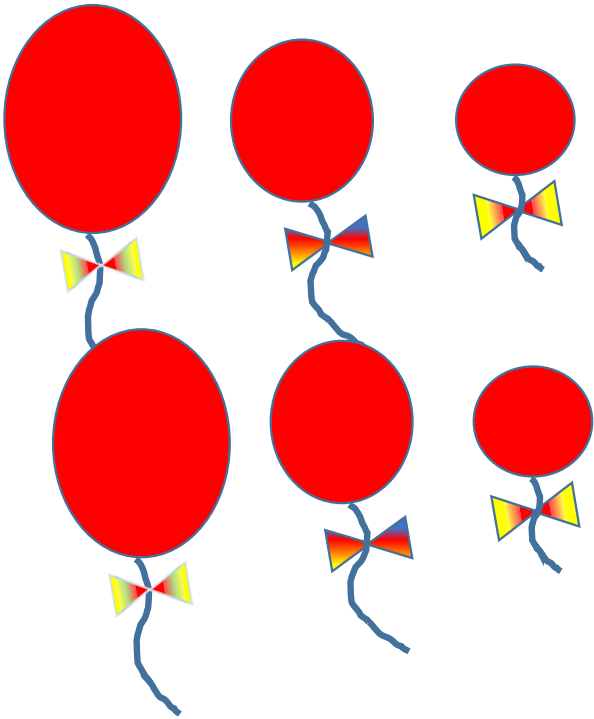
Each of them will be priced differently.

I need an **updated web page** to show these prices to my customers, and my business is growing.

Number of Balloons types I am selling → 18

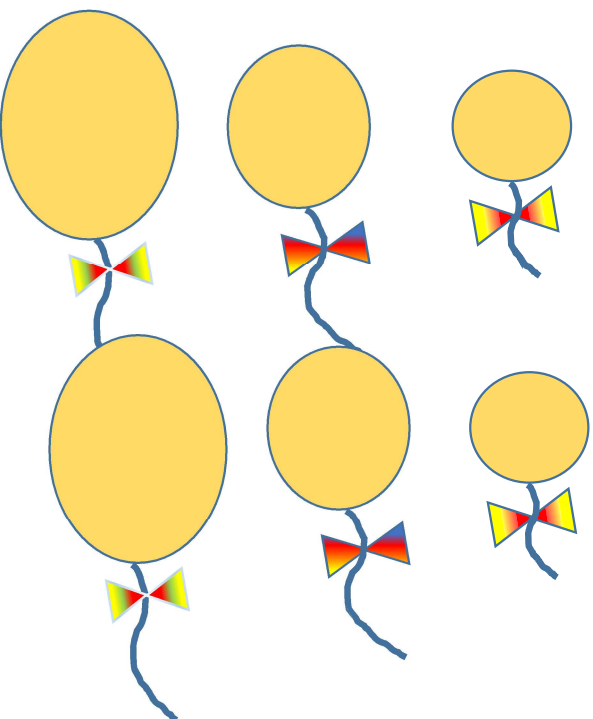


Color: Red Size : Large Gas : Oxygen Price : 6	Color: Red Size : Medium Gas : Oxygen Price : 4	Color: Red Size : Small Gas : Oxygen Price : 2
---	--	---



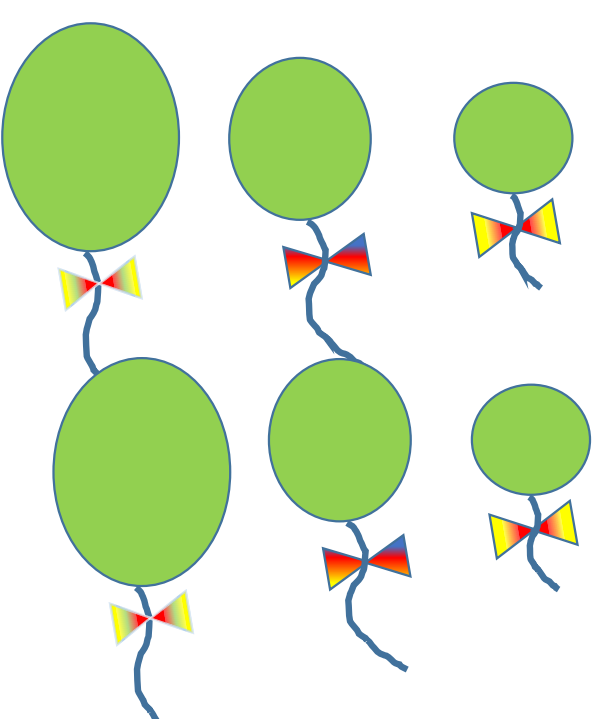
Color: Red Size : Large Gas : Helium Price : 7	Color: Red Size : Medium Gas : Helium Price : 5	Color: Red Size : Small Gas : Helium Price : 3
---	--	---

Color: Gold Size : Large Gas : Oxygen Price : 7	Color: Gold Size : Medium Gas : Oxygen Price : 5	Color: Gold Size : Small Gas : Oxygen Price : 3
--	---	--



Color: Gold Size : Large Gas : Helium Price : 8	Color: Gold Size : Medium Gas : Helium Price : 6	Color: Gold Size : Small Gas : Helium Price : 4
--	---	--

Color: Green Size : Large Gas : Oxygen Price : 8	Color: Green Size : Medium Gas : Oxygen Price : 6	Color: Green Size : Small Gas : Oxygen Price : 4
---	--	---



Color: Green Size : Large Gas : Helium Price : 9	Color: Green Size : Medium Gas : Helium Price : 7	Color: Green Size : Small Gas : Helium Price : 5
---	--	---

Solution #1

```

<?php

$balloon1_color = 'Red';
$balloon1_size  = 'S';
$balloon1_price = 2;
$balloon1_gas   = 'helium';
$balloon1_desc  = 'Red S Helium';

$balloon2_color = 'Red';
$balloon2_size  = 'S';
$balloon2_price = 3;
$balloon2_gas   = 'oxygen';
$balloon2_desc  = 'Red S Oxygen';

$balloon3_color = 'Red';
$balloon3_size  = 'M';
$balloon3_price = 4;
$balloon3_gas   = 'helium';
$balloon3_desc  = 'Red M Helium';

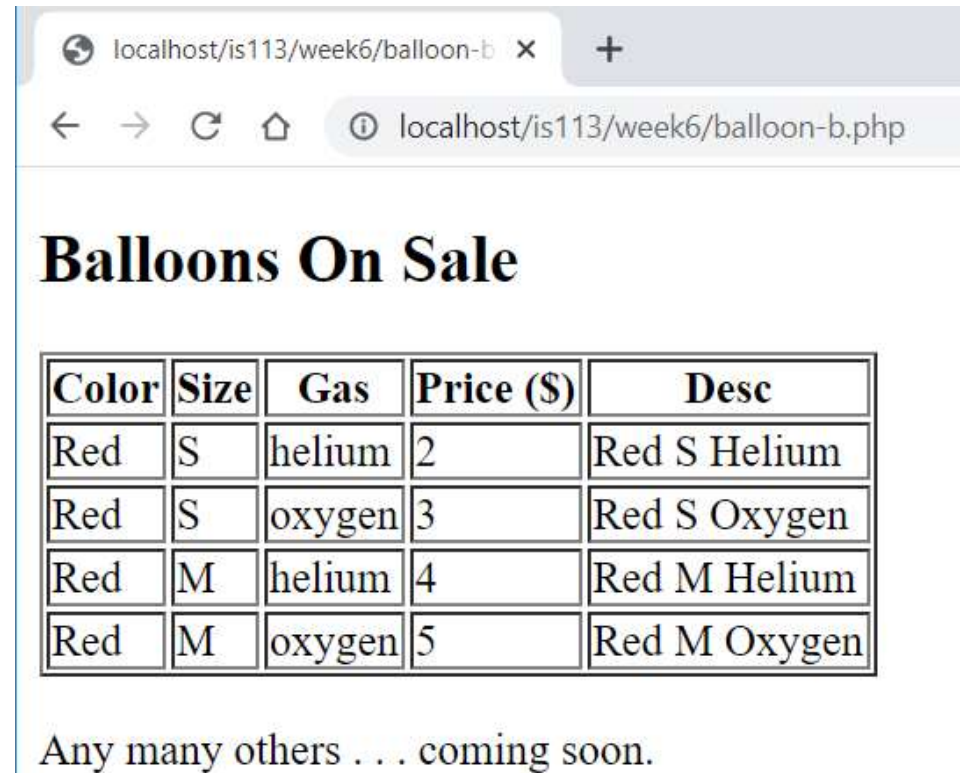
$balloon4_color = 'Red';
$balloon4_size  = 'M';
$balloon4_price = 5;
$balloon4_gas   = 'oxygen';
$balloon4_desc  = 'Red M Oxygen';

// . . .
// for the rest of the combination

?>

```

Source: balloon-b.php



The screenshot shows a web browser window with the address bar displaying 'localhost/is113/week6/balloon-b.php'. The page content includes a heading 'Balloons On Sale' and a table with the following data:

Color	Size	Gas	Price (\$)	Desc
Red	S	helium	2	Red S Helium
Red	S	oxygen	3	Red S Oxygen
Red	M	helium	4	Red M Helium
Red	M	oxygen	5	Red M Oxygen

Any many others . . . coming soon.

This method is no longer desirable.
So many variables....

```
<?php
$balloons = [
    'balloon1' => ['Red', 'S', 2, 'oxygen', 'Red S Oxygen'],
    'balloon2' => ['Red', 'S', 3, 'helium', 'Red S Helium'],
    'balloon3' => ['Red', 'M', 4, 'oxygen', 'Red M Oxygen'],
    'balloon4' => ['Red', 'M', 5, 'helium', 'Red M Helium'],
    'balloon5' => ['Red', 'L', 6, 'oxygen', 'Red L Oxygen'],
    'balloon6' => ['Red', 'L', 7, 'helium', 'Red L Helium'],
    'balloon7' => ['Gold', 'S', 3, 'oxygen', 'Gold S Oxygen'],
    'balloon8' => ['Gold', 'S', 4, 'helium', 'Gold S Helium'],
    'balloon9' => ['Gold', 'M', 5, 'oxygen', 'Gold M Oxygen'],
    'balloon10' => ['Gold', 'M', 6, 'helium', 'Gold M Helium'],
    'balloon11' => ['Gold', 'L', 7, 'oxygen', 'Gold L Oxygen'],
    'balloon12' => ['Gold', 'L', 8, 'helium', 'Gold L Helium'],
    'balloon13' => ['Green', 'S', 4, 'oxygen', 'Green S Oxygen'],
    'balloon14' => ['Green', 'S', 5, 'helium', 'Green S Helium'],
    'balloon15' => ['Green', 'M', 6, 'oxygen', 'Green M Oxygen'],
    'balloon16' => ['Green', 'M', 7, 'helium', 'Green M Helium'],
    'balloon17' => ['Green', 'L', 8, 'oxygen', 'Green L Oxygen'],
    'balloon18' => ['Green', 'L', 9, 'helium', 'Green L Helium']
];

?>
```

Source: balloon-b1.php

Using Associative
Arrays still works

localhost/is113/week6/balloon-c x +
localhost/is113/week6/balloon-c.php

Balloons On Sale

Color	Size	Gas	Price (\$)	Desc
Red	S	2	oxygen	Red S Oxygen
Red	S	3	helium	Red S Helium
Red	M	4	oxygen	Red M Oxygen
Red	M	5	helium	Red M Helium
Red	L	6	oxygen	Red L Oxygen
Red	L	7	helium	Red L Helium
Gold	S	3	oxygen	Gold S Oxygen
Gold	S	4	helium	Gold S Helium
Gold	M	5	oxygen	Gold M Oxygen
Gold	M	6	helium	Gold M Helium
Gold	L	7	oxygen	Gold L Oxygen
Gold	L	8	helium	Gold L Helium
Green	S	4	oxygen	Green S Oxygen
Green	S	5	helium	Green S Helium
Green	M	6	oxygen	Green M Oxygen
Green	M	7	helium	Green M Helium
Green	L	8	oxygen	Green L Oxygen
Green	L	9	helium	Green L Helium

Solution #2

<?php

```

$balloons = [
    'balloon1' => ['Red', 'S', '2', 'oxygen', 'Red S Oxygen'],
    'balloon2' => ['Red', 'S', '3', 'helium', 'Red S Helium'],
    'balloon3' => ['Red', 'M', '4', 'oxygen', 'Red M Oxygen'],
    'balloon4' => ['Red', 'M', '5', 'helium', 'Red M Helium'],
    'balloon5' => ['Red', 'L', '6', 'oxygen', 'Red L Oxygen'],
    'balloon6' => ['Red', 'L', '7', 'helium', 'Red L Helium'],
    'balloon7' => ['Gold', 'S', '3', 'oxygen', 'Gold S Oxygen'],
    'balloon8' => ['Gold', 'S', '4', 'helium', 'Gold S Helium'],
    'balloon9' => ['Gold', 'M', '5', 'oxygen', 'Gold M Oxygen'],
    'balloon10' => ['Gold', 'M', '6', 'helium', 'Gold M Helium'],
    'balloon11' => ['Gold', 'L', '7', 'oxygen', 'Gold L Oxygen'],
    'balloon12' => ['Gold', 'L', '8', 'helium', 'Gold L Helium'],
    'balloon13' => ['Green', 'S', '4', 'oxygen', 'Green S Oxygen'],
    'balloon14' => ['Green', 'S', '5', 'helium', 'Green S Helium'],
    'balloon15' => ['Green', 'M', '6', 'oxygen', 'Green M Oxygen'],
    'balloon16' => ['Green', 'M', '7', 'helium', 'Green M Helium'],
    'balloon17' => ['Green', 'L', '8', 'oxygen', 'Green L Oxygen'],
    'balloon18' => ['Green', 'L', '9', 'helium', 'Green L Helium']
];

```

?>

Source: balloon-b1.php

Similar attributes of
Balloons

- Color
- Size
- Price
- Gas
- Desc

Classes & Objects

Solution #3

<?php

```

$balloons = [
    'balloon1' => ['Red', 'S', '2', 'oxygen', 'Red S Oxygen'],
    'balloon2' => ['Red', 'S', '3', 'helium', 'Red S Helium'],
    'balloon3' => ['Red', 'M', '4', 'oxygen', 'Red M Oxygen'],
    'balloon4' => ['Red', 'M', '5', 'helium', 'Red M Helium'],
    'balloon5' => ['Red', 'L', '6', 'oxygen', 'Red L Oxygen'],
    'balloon6' => ['Red', 'L', '7', 'helium', 'Red L Helium'],
    'balloon7' => ['Gold', 'S', '3', 'oxygen', 'Gold S Oxygen'],
    'balloon8' => ['Gold', 'S', '4', 'helium', 'Gold S Helium'],
    'balloon9' => ['Gold', 'M', '5', 'oxygen', 'Gold M Oxygen'],
    'balloon10' => ['Gold', 'M', '6', 'helium', 'Gold M Helium'],
    'balloon11' => ['Gold', 'L', '7', 'oxygen', 'Gold L Oxygen'],
    'balloon12' => ['Gold', 'L', '8', 'helium', 'Gold L Helium'],
    'balloon13' => ['Green', 'S', '4', 'oxygen', 'Green S Oxygen'],
    'balloon14' => ['Green', 'S', '5', 'helium', 'Green S Helium'],
    'balloon15' => ['Green', 'M', '6', 'oxygen', 'Green M Oxygen'],
    'balloon16' => ['Green', 'M', '7', 'helium', 'Green M Helium'],
    'balloon17' => ['Green', 'L', '8', 'oxygen', 'Green L Oxygen'],
    'balloon18' => ['Green', 'L', '9', 'helium', 'Green L Helium']
];

```

?>

Source: balloon-b1.php

All **balloons** share
the **same set of**
attributes:

- Color
- Size
- Price
- Gas
- Desc



Class

Balloon

- color
- size
- price
- gas
- desc

+ getColor()
+ getSize()
+ getPrice()
+ getGas ()
+ getDesc()

Solution #3

Class

A **class** is a **template** from which individual **objects** can be created.

It **defines** what a **balloon**:

- **Should Be**
- **Can Do**

Balloon
- color
- size
- price
- gas
- desc
+ getColor()
+ getSize()
+ getPrice()
+ getGas ()
+ getDesc()

```
<?php
class Balloon{

    // Properties
    private $color;
    private $size;
    private $gas;
    private $price;
    private $desc;

    // Constructors
    public function __construct($color, $size, $gas, $price, $desc) {
        $this->color = $color;
        $this->size = $size;
        $this->gas = $gas;
        $this->price = $price;
        $this->desc = $desc;
    }

    // Getter methods
    public function getColor() {
        return $this->color;
    }

    public function getSize() {
        return $this->size;
    }

    public function getGas() {
        return $this->gas;
    }

    public function getPrice() {
        return $this->price;
    }

    public function getDesc() {
        return $this->desc;
    }
}
?>
```

Properties

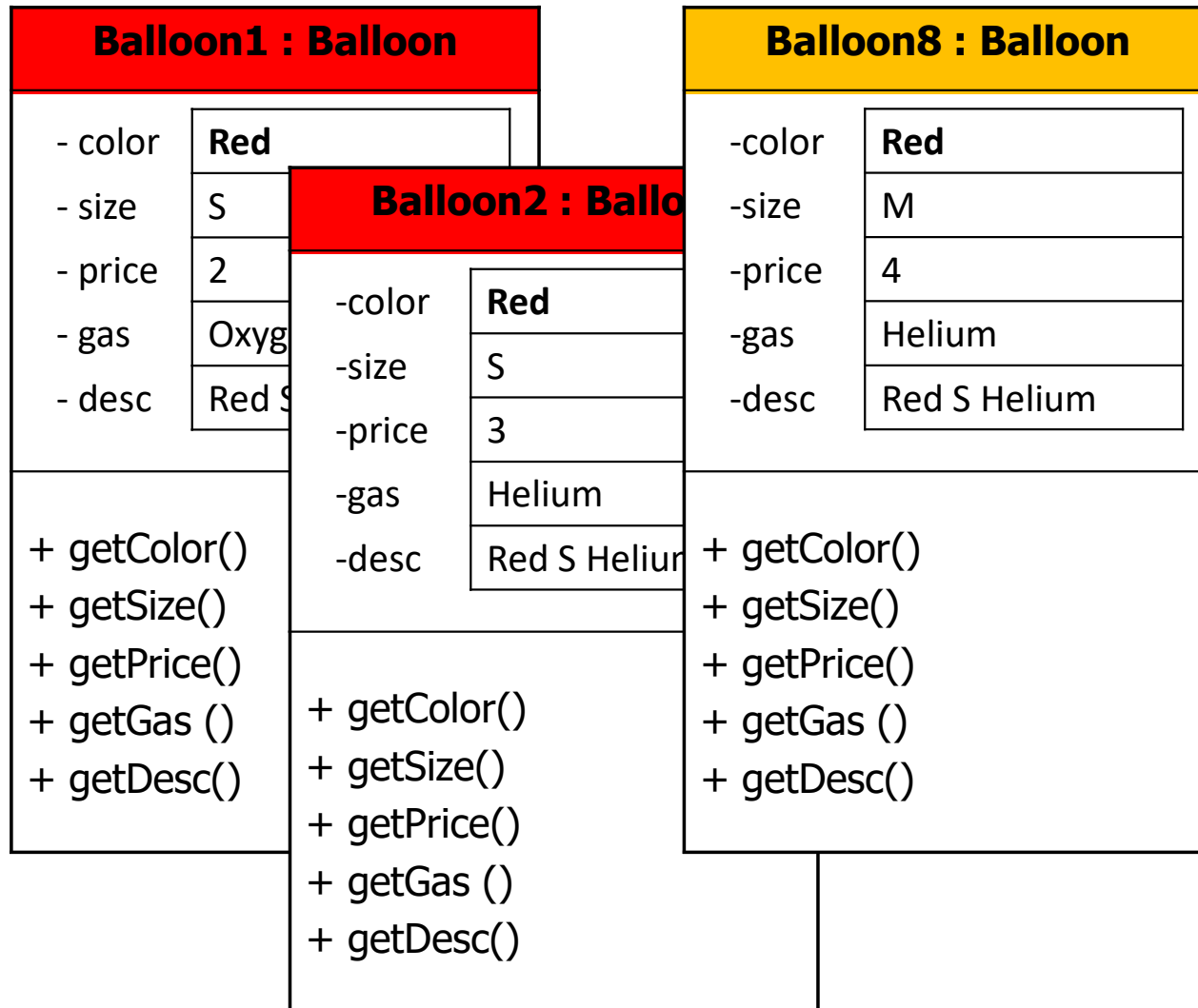
Constructor

- Creates an instance of a class

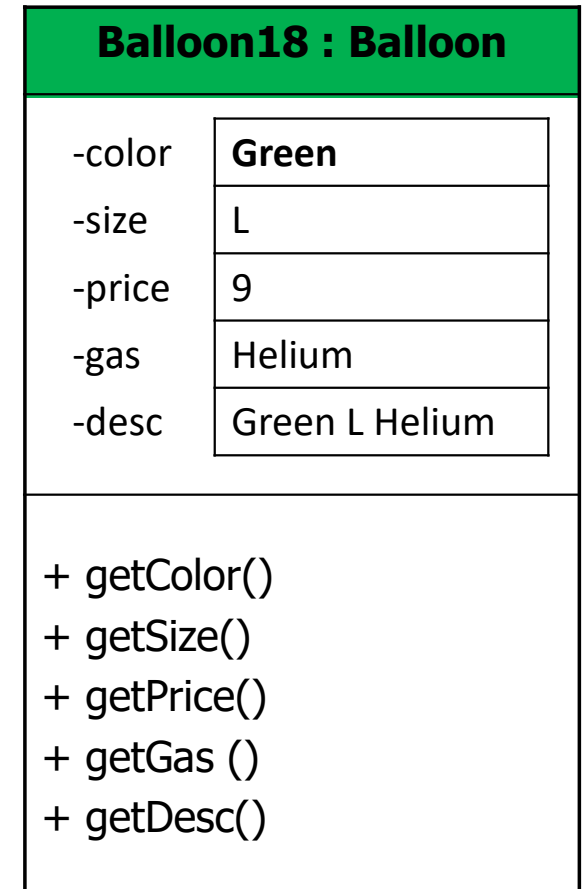
Methods

Source: Balloon.php

Objects

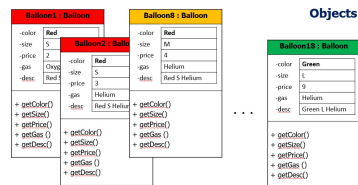


• • •



Object

An object is an **instance** of a class.



Objects



<?php

```
$balloon1 = new Balloon("Red", "S", 2, "oxygen", "Red S Oxygen");

$balloon2 = new Balloon("Red", "S", 3, "helium", "Red S Helium");

$balloon3 = new Balloon("Red", "M", 4, "oxygen", "Red M Oxygen");

...

$balloons_arr = [$balloon1, $balloon2, $balloon3, $balloon4,
$balloon5, $balloon6, $balloon7, $balloon8,
$balloon9, $balloon10, $balloon11, $balloon12,
$balloon13, $balloon14, $balloon15, $balloon16,
$balloon17, $balloon18];
```

Source: balloon-c.php

```
// Constructors

public function __construct($color, $size, $gas, $price, $desc) {
    $this->color = $color;
    $this->size = $size;
    $this->gas = $gas;
    $this->price = $price;
    $this->desc = $desc;
}
```

Constructor
- Creates an
instance of a class

Source: Balloon.php

Instantiation

- Create an object based on class template
- Class is **Balloon**

Solution #3

```

<!DOCTYPE html>
. . .
<table border='1'>
  <tr>
    <th> Color </th>
    <th> Size </th>
    <th> Gas </th>
    <th> Price ($) </th>
    <th> Desc </th>
  </tr>
  <?php
    foreach ($balloons_arr as $balloon)
      echo "
        <tr>
          <td> {$balloon->getColor()} </td>
          <td> {$balloon->getSize()} </td>
          <td> {$balloon->getGas()} </td>
          <td> {$balloon->getPrice()} </td>
          <td> {$balloon->getDesc()} </td>
        </tr>";
      ?>
    </table>
  . . .

```

```

// Getter methods

public function getColor() {
    return $this->color;
}

public function getSize() {
    return $this->size;
}

public function getGas() {
    return $this->gas;
}

public function getPrice() {
    return $this->price;
}

public function getDesc() {
    return $this->desc;
}

```

Methods

Source: Balloon.php

- Calling the methods to access an attribute.
- It can also be calling a method for some processing of the data.

Source: balloon-c.php

```
<?php

require_once "header.php";

$balloon1 = new Balloon("Red","S",2,"oxygen","Red S Oxygen");
$balloon2 = new Balloon("Red","S",3,"helium","Red S Helium");
$balloon3 = new Balloon("Red","M",4,"oxygen","Red M Oxygen");
$balloon4 = new Balloon("Red","M",5,"helium","Red M Helium");
$balloon5 = new Balloon("Red","L",6,"oxygen","Red L Oxygen");
$balloon6 = new Balloon("Red","L",7,"helium","Red L Helium");

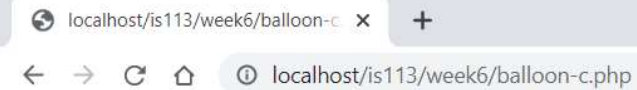
$balloon7 = new Balloon("Gold","S",3,"oxygen","Gold S Oxygen");
$balloon8 = new Balloon("Gold","S",4,"helium","Gold S Helium");
$balloon9 = new Balloon("Gold","M",5,"oxygen","Gold M Oxygen");
$balloon10 = new Balloon("Gold","M",6,"helium","Gold M Helium");
$balloon11 = new Balloon("Gold","L",7,"oxygen","Gold L Oxygen");
$balloon12 = new Balloon("Gold","L",8,"helium","Gold L Helium");

$balloon13 = new Balloon("Green","S",4,"oxygen","Green S Oxygen");
$balloon14 = new Balloon("Green","S",5,"helium","Green S Helium");
$balloon15 = new Balloon("Green","M",6,"oxygen","Green M Oxygen");
$balloon16 = new Balloon("Green","M",7,"helium","Green M Helium");
$balloon17 = new Balloon("Green","L",8,"oxygen","Green L Oxygen");
$balloon18 = new Balloon("Green","L",9,"helium","Green L Helium");

$balloons_arr = [$balloon1, $balloon2, $balloon3, $balloon4,
                 $balloon5, $balloon6, $balloon7, $balloon8,
                 $balloon9, $balloon10, $balloon11, $balloon12,
                 $balloon13, $balloon14, $balloon15, $balloon16,
                 $balloon17, $balloon18];

?>
```

Source: balloon-c.php



Balloons On Sale

Color	Size	Gas	Price (\$)	Desc
Red	S	2	oxygen	Red S Oxygen
Red	S	3	helium	Red S Helium
Red	M	4	oxygen	Red M Oxygen
Red	M	5	helium	Red M Helium
Red	L	6	oxygen	Red L Oxygen
Red	L	7	helium	Red L Helium
Gold	S	3	oxygen	Gold S Oxygen
Gold	S	4	helium	Gold S Helium
Gold	M	5	oxygen	Gold M Oxygen
Gold	M	6	helium	Gold M Helium
Gold	L	7	oxygen	Gold L Oxygen
Gold	L	8	helium	Gold L Helium
Green	S	4	oxygen	Green S Oxygen
Green	S	5	helium	Green S Helium
Green	M	6	oxygen	Green M Oxygen
Green	M	7	helium	Green M Helium
Green	L	8	oxygen	Green L Oxygen
Green	L	9	helium	Green L Helium

So far...

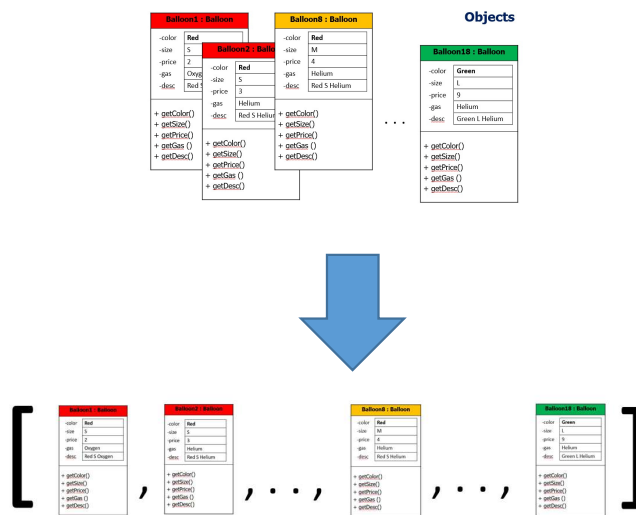
We have “hard-coded” our balloon data INSIDE our codes.

“I want the data file to be separated from my codes.”

Data Access Objects using **Classes**

Data Access Objects

Provide access to data stored in a file, database, etc.



```
<?php

// this class is a simulation of retrieval of records from some
// storage (e.g. database).

class BalloonDAO {

    private $balloons;

    public function __construct() {
        $this->balloons = [
            new Balloon("Red", "S", 2, "oxygen", "Red S Oxygen"),
            new Balloon("Red", "S", 3, "helium", "Red S Helium"),
            new Balloon("Red", "M", 4, "oxygen", "Red M Oxygen"),
            new Balloon("Red", "M", 5, "helium", "Red M Helium"),
            new Balloon("Red", "L", 6, "oxygen", "Red L Oxygen"),
            new Balloon("Red", "L", 7, "helium", "Red L Helium"),
            new Balloon("Gold", "S", 3, "oxygen", "Gold S Oxygen"),
            new Balloon("Gold", "S", 4, "helium", "Gold S Helium"),
            new Balloon("Gold", "M", 5, "oxygen", "Gold M Oxygen"),
            new Balloon("Gold", "M", 6, "helium", "Gold M Helium"),
            new Balloon("Gold", "L", 7, "oxygen", "Gold L Oxygen"),
            new Balloon("Gold", "L", 8, "helium", "Gold L Helium"),
            new Balloon("Green", "S", 4, "oxygen", "Green S Oxygen"),
            new Balloon("Green", "S", 5, "helium", "Green S Helium"),
            new Balloon("Green", "M", 6, "oxygen", "Green M Oxygen"),
            new Balloon("Green", "M", 7, "helium", "Green M Helium"),
            new Balloon("Green", "L", 8, "oxygen", "Green L Oxygen"),
            new Balloon("Green", "L", 9, "helium", "Green L Helium")
        ];
    }

    public function getBalloonsOnSale() {
        return $this->balloons;
    }
}

?>
```

Properties

Constructor
- Creates an instance of a class

Methods

Source: BalloonDAO.php

Solution #4

<?php

require_once "header.php";

\$dao = new BalloonDAO();

\$balloons_arr = \$dao->getBallonsOnSale();

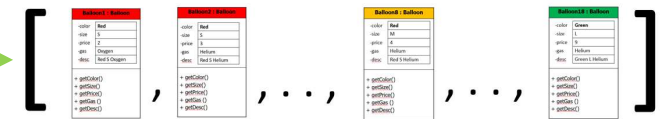
?>

```

public function __construct() {
    $this->balloons = [
        new Balloon("Red","S",2,"oxygen","Red S Oxygen"),
        new Balloon("Red","S",3,"helium","Red S Helium"),
        new Balloon("Red","M",4,"oxygen","Red M Oxygen"),
        new Balloon("Red","M",5,"helium","Red M Helium"),
        new Balloon("Red","L",6,"oxygen","Red L Oxygen"),
        new Balloon("Red","L",7,"helium","Red L Helium"),
        new Balloon("Gold","S",3,"oxygen","Gold S Oxygen"),
        new Balloon("Gold","S",4,"helium","Gold S Helium"),
        new Balloon("Gold","M",5,"oxygen","Gold M Oxygen"),
        new Balloon("Gold","M",6,"helium","Gold M Helium"),
        new Balloon("Gold","L",7,"oxygen","Gold L Oxygen"),
        new Balloon("Gold","L",8,"helium","Gold L Helium"),
        new Balloon("Green","S",4,"oxygen","Green S Oxygen"),
        new Balloon("Green","S",5,"helium","Green S Helium"),
        new Balloon("Green","M",6,"oxygen","Green M Oxygen"),
        new Balloon("Green","M",7,"helium","Green M Helium"),
        new Balloon("Green","L",8,"oxygen","Green L Oxygen"),
        new Balloon("Green","L",9,"helium","Green L Helium")
    ];
}

```

Constructor
- Creates an instance of a class

**Instantiation**

- Create an object based on class template
- Class is BalloonDAO

Source: BalloonDAO.php

Solution #4

(Same as Solution #3)

```

<!DOCTYPE html>
. . .
<table border="1">
  <tr>
    <th> Color </th>
    <th> Size </th>
    <th> Gas </th>
    <th> Price ($) </th>
    <th> Desc </th>
  </tr>
  <?php
    foreach ($balloons_arr as $balloon)
      echo "<tr>
        <td> {$balloon->getColor()} </td>
        <td> {$balloon->getSize()} </td>
        <td> {$balloon->getGas()} </td>
        <td> {$balloon->getPrice()} </td>
        <td> {$balloon->getDesc()} </td>
      </tr>";
    </tr>";
  ?>
</tr>
</table>
. . .

```

```

// Getter methods

public function getColor() {
    return $this->color;
}

public function getSize() {
    return $this->size;
}

public function getGas() {
    return $this->gas;
}

public function getPrice() {
    return $this->price;
}

public function getDesc() {
    return $this->desc;
}

```

Methods

Source: Balloon.php

- Calling the methods to access an attribute.
- It can also be calling a method for some processing of the data.

Source: balloon-d.php

I am selling EVEN MORE balloons now.

With new colors (Pink, Yellow, Blue, Purple, etc.) and sizes (XL, XXL).

The prices of my balloons change throughout the year.

For example:

- On Valentine's Day, the prices of balloons in Red, of size L, XL, XXL will increase by 50%.
- During the Christmas season, the prices of helium balloons will increase by 20%.

I need a more versatile way to support my business needs.

PHP Data Objects with MySQL database

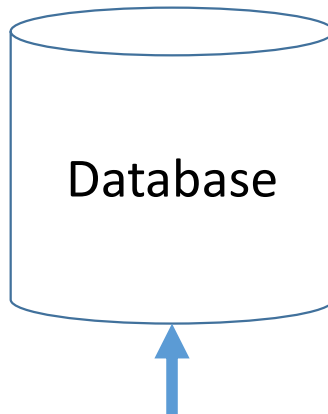


Table : Balloon

Attributes	Type
balloonID	String
color	String
size	String
price	Integer
gas	String
descr	String

balloonID	color	size	price	gas	descr
balloon1	Red	S	2	oxygen	Red S Oxygen
balloon2	Red	S	3	helium	Red S Helium
balloon3	Red	M	4	oxygen	Red M Oxygen
balloon4	Red	M	5	helium	Red M Helium
balloon5	Red	L	6	oxygen	Red L Oxygen
balloon6	Red	L	7	helium	Red L Helium
balloon7	Gold	S	3	oxygen	Gold S Oxygen
balloon8	Gold	S	4	helium	Gold S Helium
balloon9	Gold	M	5	oxygen	Gold M Oxygen
balloon10	Gold	M	6	helium	Gold M Helium
balloon11	Gold	L	7	oxygen	Gold L Oxygen
balloon12	Gold	L	8	helium	Gold L Helium
balloon13	Green	S	4	oxygen	Green S Oxygen
balloon14	Green	S	5	helium	Green S Helium
balloon15	Green	M	6	oxygen	Green M Oxygen
balloon16	Green	M	7	helium	Green M Helium
balloon17	Green	L	8	oxygen	Green L Oxygen
balloon18	Green	L	9	helium	Green L Helium

DAO using PHP Data Objects

PHP codes
"query" the
database via SQL.

It retrieves the
SQL query results.

```
<?php

// this class is a simulation of retrieval of records from some storage (e.g. database).
class BalloonDAO {

    public function getBallonsOnSale() {
        $sql = 'SELECT * FROM balloon';

        $connMgr = new ConnectionManager();
        $conn = $connMgr->getConnection();

        $stmt = $conn->prepare($sql);
        $stmt->execute();

        $result = [];

        $stmt->setFetchMode(PDO::FETCH_ASSOC);
        while($row = $stmt->fetch()) {
            //new Balloon("Red","S",2,"oxygen","Red S Oxygen"),
            $result[] = new Balloon($row['color'], $row['size'], $row['price'],
                                   $row['gas'], $row["descr"]);
        }

        $stmt->closeCursor();
        $conn = null;

        return $result;
    }
}
?>
```

Connect to database

Prepare and execute the SQL

Fetch results and return as an array of objects

Source: BalloonDAO.php

PHP Data Objects

```
<?php

// this class is a simulation of retrieval of records from some storage (e.g. database).
class BalloonDAO {

    public function getBallonsOnSale() {
        $sql = 'SELECT * FROM balloon';

        $connMgr = new ConnectionManager();
        $conn = $connMgr->getConnection();

        $stmt = $conn->prepare($sql);
        $stmt->execute();

        $result = [];

        $stmt->setFetchMode(PDO::FETCH_ASSOC);
        while($row = $stmt->fetch()) {
            //new Balloon("Red","S",2,"oxygen","Red S Oxygen"),
            $result[] = new Balloon($row['color'], $row['size'], $row['price'],
                                   $row['gas'], $row['descr']);
        }

        $stmt->closeCursor();
        $conn = null;

        return $result;
    }
}

?>
```

Connect to database

Prepare and execute the SQL

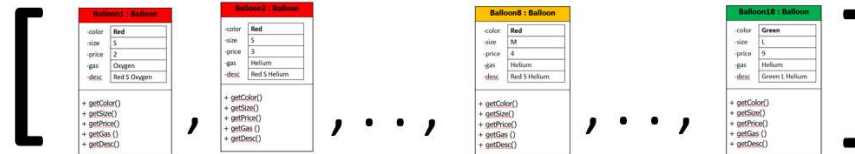
Fetch results and return as an array of objects

select * from balloon

Database:
bizballoon

+ Options

balloonID	color	size	price	gas	descr
balloon1	Red	S	2	oxygen	Red S oxygen
balloon2	Red	S	3	helium	Red S helium
balloon3	Red	M	4	oxygen	Red M oxygen
balloon4	Red	M	5	helium	Red M S helium
balloon5	Red	L	6	oxygen	Red L oxygen
balloon6	Red	L	7	helium	Red L helium
balloon7	Gold	S	3	oxygen	Gold S oxygen
balloon8	Gold	S	4	helium	Gold S helium
balloon9	Gold	M	5	oxygen	Gold M oxygen
balloon10	Gold	M	6	helium	Gold M helium
balloon11	Gold	L	7	oxygen	Gold L oxygen
balloon12	Gold	L	8	helium	Gold L helium
balloon13	Green	S	4	oxygen	Green S oxygen
balloon14	Green	S	5	helium	Green S helium
balloon15	Green	M	6	oxygen	Green M oxygen
balloon16	Green	M	7	helium	Green M helium
balloon17	Green	L	8	oxygen	Green L oxygen
balloon18	Green	L	9	helium	Green L helium



Source: BalloonDAO.php

Class vs. DAO vs. DAO with PDO

