<http://market/?second=Alex>

12.3

**Метод Public ( public function )** – данная функция доступна везде и всем.

**Метод Protected** **( public function )** -данная функция будет доступна только в текущем классе и классе «наследнике» - тот класс, который будет наследовать эту функцию.

**Метод Private ( public function ) –** только в текущем классе.

<?php

include "./avto/bmw.php";

include "./avto/mercedes.php";

// Проверка - если параметр авто существует, то мы исполняем те условия, которые

if(isset($\_GET['avto'])) {

    switch ($\_GET['avto']) {

        case "bmw":{

            echo "Пользователь ищет запчасти по BMW";

        }

        break;

        case "mercedes":{

        }

             echo "Пользователь ищет запчасти по mercedes";

        break;

        case "audi":{

        }

             echo "Пользователь ищет запчасти по mercedes";

        break;

1. Инициализируем наш класс BMW с помощью - **$newClass() = new Bmw();** Теперь мы имеем доступ к нашим методам. Что происходит при инициализации? При инициализации вызывается метод ( один раз)
2. Public function\_constract() {} – служит для определения методов или параметров, которые необходимы для класса, например: переменные, свойства ( инициализация классов в методе constract (конструктор) делать каки-то операции с методами ). То есть мы можем их передать при инициализации в наш класс. Например, давайте передадим (**$name**) – public function\_construct($name) в файле BMW.PHP.
3. Нам нужно передать название автомобиля – мы передадим «BMW» в файле index.PHP - $newClass = new Bmw(‘’’BMW’’);
4. У **класса** могут быть какие то свои свойства – какие то переменные, которые определяются внутри класса, и доступ к которым устанавливается, согласно уровням доступа «**Public, Protected, Private**». В данном случае у нас есть переменная «NAME», которой был установлен уровень доступа «**Private**».
5. Для указания переменной в классе «Privete $name» используем метод **«This»** в контексте текущего класса в файле BMW.PHP. Таким образом, при инициализации нашего класса BMW, мы будем передавать название «name», название «name» будем присваивать свойству класса «$name». Также можно не указывать текущие название «BMW», тогда его необходимо указать в качестве параметров по умолчанию public function\_construct ($name = «Название не указано»). в файле BMW.PHP.
6. Добавляем еще один класс $newClass() = new Bmw("") в файле index.PHP

Добавляем функцию Public function getName()

{

return

}

 //\*\* Доступен везде и всем \*/

    //\*\* Метод для получения названия авто \*/

    //\*\* @return string - названия авто \*/

    Public function getName()

    {

        return $this->name;

    }

1. Вызов метода нашего класса.

Вызываем класс: $newClass() = new Bmw("BMW") – 1 класс, где $newClass() - класс = new Bmw – имя класса; При этом хотим вызвать название нашего авто);

$newNewer() = new ("") – 2 класс (без имени, ничего не инициализируем, при инициализации нашего класса не передаём никакие параметры в наш класс. При этом хотим вызвать название нашего авто);

 $newClass = new Bmw("BMW");

            $newNever = new ();

            $newClass ->getName;

            $newNever ->getName;

Для вывоза используем запрос: http://market/?avto=bmw

Ответ: Пользователь ищет запчасти по BMW

Index.php

?php

include "./auto/bmw.php";

include "./auto/mercedes.php";

// проверка

if(isset($\_GET['auto'])){

    switch ($\_GET['auto']) {

        case "bmw":{

            echo "Пользователь ищет запчасти по BMW";

            $newClass = new Bmw("bmw");

            $newNever = new Bmw();

           // $newAuto ->newtClass ->getName();

            $nameAuto = $newNever ->getName();

            echo $nameAuto;

        }

        break;

BMW.PHP

<?php

// класс для работы с авто BMW.

class bmw

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    //\*\* Доступен везде и всем \*/

    //\*\* Метод для получения названия авто \*/

    //\*\* return string - названия авто \*/

    Public function getName()

    {

        return $this->name;

Для того чтобы что-то отобразить пользователю

1. Давайте положим результат выполнения нашего метода в какую-то переменную $nameAuto, а теперь выведем это всё пользователю.
2. $nameAuto = $newNever ->getName();
3. echo $nameAuto;

Итого, что мы сделали?

Мы вывели «newNewer» без названия, присвоили значение «name» по умолчанию «название не указано», потом присвоили нашему классу свойство «name = название не указано», и вернули через «getName», присвоив значение переменной «Newname» и вывели в браузере «echo $nameAuto»

Index.php

<?php

include "./auto/bmw.php";

include "./auto/mercedes.php";

// проверка

if(isset($\_GET['auto'])){

    switch ($\_GET['auto']) {

        case "bmw":{

            echo "Пользователь ищет запчасти по BMW";

            $newClass = new Bmw("bmw");

            $newNever = new Bmw();

           // $newAuto ->newtClass ->getName();

            $nameAuto = $newNever ->getName();

            echo $nameAuto;

        }

        break;

bmw.php

<?php

// класс для работы с авто BMW.

class bmw

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    //\*\* Доступен везде и всем \*/

    //\*\* Метод для получения названия авто \*/

    //\*\* return string - названия авто \*/

    Public function getName()

    {

        return $this->name;

Вызовем метод «Newclass», где мы указываем или передаём название через компьютер.

Сначала мы вызываем функцию «Getname» = bmw, а затем отображаем пользователю наш результат = bmw.

Idex.php

<?php

include "./auto/bmw.php";

include "./auto/mercedes.php";

// проверка

if(isset($\_GET['auto'])){

    switch ($\_GET['auto']) {

        case "bmw":{

            // echo "Пользователь ищет запчасти по BMW";

            $newClass = new Bmw("bmw");

            $newNever = new Bmw();

            $nameAuto = $newClass ->getName();

            // $nameAuto = $newNever ->getName();

            echo $nameAuto;

        }

        break;

        case "mercedes":{

            echo "Пользователь ищет запчасти по mercedes";

        }

        break;

        case "mercedes":{

            echo "Пользователь ищет запчасти по mercedes";

        }

        break;

bmw.php

<?php

// класс для работы с авто BMW.

class bmw

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    /\*\* Доступен везде и всем

    \* Метод для получения названия авто

    \* @return string - названия авто

    \*/

    Public function getName()

    {

        return $this->name;

    }

}

Ответ: bmw

Будем вызывать метод в зависимости от того какой класс мы инициализируем, передаём на наш сервер (один из двух, bmw или mercedes).

Для этого необходимо всё скопировать из класса bmw в класс Mercedes (данные из файла).

Наш класс будет иметь теже самые методы и теже самые свойства. Мы инициализируем класс в зависимости от того какой параметр пользователь вызвал (bmw или mercedes), и вызовем наше название.

Index.PHP

<?php

/\*\* класс для работы с авто Mercedes

\*

\*/

class Mercedes

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    /\*\*

     \* Доступен везде и всем.

     \* Метод для получения названия авто.

     \* @return string - название авто (что возвращать)

     \*/

    public function getName()

    {

        return $this->name;

    }

}

BMW.PHP

<?php

// класс для работы с авто BMW.

class bmw

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    /\*\*

     \* Доступен везде и всем.

     \* Метод для получения названия авто.

     \* @return string - название авто (что возвращать)

     \*/

    Public function getName()

    {

        return $this->name;

    }

}

Ответ: Пользователь выбрал автомобиль: bmw

Mercedes.PHP

<?php

/\*\* класс для работы с авто Mercedes

\*

\*/

class Mercedes

{

    private $name;

    public function \_\_construct($name = "Название не указано")

    {

        $this -> name = $name;

    }

    /\*\*

     \* Доступен везде и всем.

     \* Метод для получения названия авто.

     \* @return string - название авто (что возвращать)

     \*/

    public function getName()

    {

        return $this->name;

    }

}

Ответ: Пользователь выбрал автомобиль: Mercedes