ООП. Модуль 4.

Применим на практике функции и стандарты с которыми мы познакомились.

Домашнее задание состоит из нескольких небольших отдельных не связанных между собой задач (кроме CMS). Каждое задание делается в отдельном файле с уникальным пространством имен. Классы должны располагаться внутри папки, согласно стандарту psr-4.

1. Все на своих местах

a. Создайте класс Manager - он будет расставлять объекты по местам. класс содержит только один метод place($item).
   - Если в этот метод передан объект типа Бумажный, то метод должен вывести строку: 'Положил <Имя класса> на стол'
   - Если в этот метод передан объект типа Инструмент, то метод должен вывести строку: 'Убрал <Имя класса> внутрь стола'
   - Если в этот метод передано что-то другое, то метод должен вывести строку: 'Выкинул <Имя класса, если объект, либо переданное значение> в корзину'
   - внутри <> - нужно подставить то, что требуется, например: 'Положил Document на стол'

b. Создайте два абстрактных класса Papers (Бумажный) и Instrument (Инструмент)

c. Создайте Менеджера, и передайте ему не менее 5-ти различных $item'ов, так чтобы хотя-бы по одному разу было выведено каждое из сообщений.

2. Создатель

a. Создайте абстрактный класс Item, со свойством $name, которое устанавливается при создании объекта. Добавьте публичный метод show, который выводит строку 'Я ' . $this->name
b. Создайте class EmptyItem extends Item - В нем переопределен метод show, и он должен выводить: 'Класс ' . $this->name . ' не найден'
c. Создайте класс Creator (создатель) в нем должен быть один статический метод create($name), который создает новый класс, по переданному имени, если класс не найден - то должен быть возвращен новый объект EmptyItem. Пример: если передана строка cat - и есть класс Cat extends Item - то должен быть возвращен объект Cat.
d. Создайте классы наследуемые от Item, создайте массив из строк не менее 10 различных строк, почувствуйте себя создателем и в цикле создайте объекты, используя класс Creator, для каждой строки массива, и выполните метод show() у каждого из этих объектов.

3. Форматированный вывод

a. Создайте интерфейсы
    - Renderable с одним публичным методом render($string)
    - Formatter с одним публичным методом format($string) : string
b. Создайте класс Display, содержащий один метод: public static function show($formatter, $string)
c. Принцип работы этого метода
    - Если $formatter - объект реализующий интерфейс Renderable - то у этого объекта просто вызывается метод render

- Если $formatter - объект, реализующий интерфейс Formatter - то выводится результат вызова метода format
- Если $formatter - объект, не реализующий интерфейс Formatter, но имеет метод format - то выводится результат вызова метода format.
- В другом случае строка $string выводится как есть.

d. Придумайте свои реализации для каждого из случаев
e. Создайте массив из строк, создайте массив из этих объектов - и выведите каждую строку каждым из возможных форматов.