# Тест 6

Шаблоны проектирования - это…

## Типовые решения типичных проблем при проектировании программного обеспечения

Шаблон проектирования, который позволяет объектам с несовместимыми интерфейсами взаимодействовать, - это…

## Adapter

Шаблон проектирования, который позволяет гарантировать, что класс имеет только один экземпляр, - это…

## Singleton

Шаблон проектирования, который позволяет создавать семейства связанных объектов без указания их конкретных классов, - это…

## Abstract factory

Шаблон проектирования, предоставляющий интерфейс для создания объектов в суперклассе, но позволяющий подклассам изменять тип создаваемых объектов, - это… **Factory method**

Шаблон проектирования, который позволяет предоставить замену или “плейсхолдер” для другого объекта, - это…

## Proxy

Шаблон проектирования, который позволяет создавать сложные объекты шаг за шагом, - это…

## Builder

Шаблон проектирования, который позволяет определить механизм подписки для уведомления нескольких объектов о любых событиях, происходящих с объектом, за которым они наблюдают, - это…

## Observer

Шаблон проектирования, который позволяет объединять объекты в древовидные структуры, а затем работать с этими структурами, как если бы они были отдельными объектами, - это…

## Composite

Шаблон проектирования, который позволяет перемещаться по элементам коллекции, не раскрывая лежащую в их основе структуру, - это…

## Iterator

Шаблон проектирования, который позволяет определить семейство алгоритмов, поместить каждый из них в отдельный класс и сделать их объекты взаимозаменяемыми, - это… **Strategy**

MVC(Model, View, Controller) - это шаблон проектирования, который…

## Подчеркивает разделение между бизнес-логикой и отображением

В MVC следующая часть управляет компоновкой и отображением:

## Вид

В MVC следующая часть управляет данными и бизнес-логикой:

## Модель

В MVC следующая часть направляет команды к модели и элементам представления:

## Контроллер

В MVC “контроллер” (controller)...

## Направляет команды к модели и элементам представления

В MVC “модель” (model)...

## Управляет данными и бизнес логикой

В MVC “представление” (view)...

## Управляет компоновкой и отображением

При использованиии шаблона проектирования singleton следующие методы класса НЕ ДОЛЖНЫ быть public:

**\_construct()**

**\_clone()**

**\_destruct()**

**В базе ДАННЫХ надо хранить ДАННЫЕ, а не файлы**

Стоит ли хранить CSS-файлы в базе данных?

## Нет, это снижает производительность

Стоит ли хранить JS-файлы в базе данных?

## Нет, это снижает производительность

Стоит ли хранить файлы изображений в базе данных?

## Нет, это снижает производительность

Стоит ли перемещать некоторые файлы веб-приложения за пределы DocumentRoot веб-сервера?

## Да, это повышает безопасность

Существует ли какой-то предопределенный синтаксис для шаблонов веб-страниц в PHP?

## Да, этот синтаксис описан в PSR

Выберите лучший подход к созданию веб-приложений:

## Использование шаблонизаторов

Стоит ли запрещать доступ к некоторым папкам веб-приложения соответствующими настройками веб-сервера?

## Да, это повышает безопасность

В чем главное преимущество подхода к шаблонизации, полностью исключающего использования PHP-кода внутри шаблона?

## Это упрощает реализацию

Почему размещение PHP-кода внутри шаблона веб-страницы - плохая идея?

## Это снижает поддерживаемость и сопровождаемость кода

Почему размещение PHP-кода внутри шаблона веб-страницы - плохая идея?

## Это может привести к рискам в области безопасности

Почему хранить учетные данные базы данных веб-приложения в php-файле лучше, чем хранить их в каком-то “неисполняемом” файле?

## В худшем случае (веб-доступ к этому файлу не запрещен) запрос к такому файлу выдаст просто пустую страницу

Существует ли какая-то предопределенная структура для веб-приложений PHP?

**Нет, но каждое приложения все равно должно иметь четкую и логичную структуру**