Контейнер внедрения зависимостей

Dependency Inversion Principle (DIP) предположим, что мы создаем модульный код с низкой связью с помощью извлечения четких подсистем абстракции.

Например, если вы хотите упростить большой класс, вы можете разделить его на несколько кусков рутинного кода и извлечь каждый кусок в новый простой разделенный класс.

Принцип говорит о том, что низкоуровневые фрагменты должны реализовывать достаточную и четкую абстракцию, а высокоуровневый код должен работать только с этой абстракцией, а не с низкоуровневой реализацией.

Когда мы разделяем большой многозадачный класс на небольшие специализированные классы, мы сталкиваемся с проблемой создания зависимых объектов и внедрения их друг в друга.

Если раньше мы могли создать один экземпляр:

$service = new MyGiantSuperService();

А после разделения мы создадим и получим все зависимые предметы и построим наш сервис:

$service = new MyService(

new Repository(new PDO('dsn', 'username', 'password')), new Session(),

new Mailer(new SmtpMailerTransport('username', 'password', host')),

new Cache(new FileSystem('/tmp/cache')),

);

Контейнер для инъекций зависимостей-это фабрика, которая позволяет нам не заботиться о создании наших объектов. В Yii2 мы можем настроить контейнер только один раз и использовать его для получения нашего сервиса, как это:

$service = Yii::$container->get('app\services\MyService')

Мы можем также использовать это:

$service = Yii::createObject('app\services\MyService')

Или мы просим контейнер внедрить его как зависимость в конструкторе другой службы:

use app\services\MyService;

class OtherService {

public function construct(MyService $myService) { ... }

}

Когда мы получим экземпляр OtherService:

$otherService = Yii::createObject('app\services\OtherService')

Во всех случаях контейнер будет разрешать все зависимости и внедрять зависимые объекты друг в друга.

В рецепте мы создаем корзину с подсистемой хранения и автоматически вставляем корзину в контроллер.