Yii::$container->set('app\cart\storage\StorageInterface', 'app\cart\storage\CustomStorage', );

But our Sessionstorage class has non-standard constructor:

class SessionStorage implements StorageInterface {

public function construct(Session $session, $key) { ... }

}

Therefore we use an anonymous function to manually creatie the instance:

Yii::$container->set('app\cart\storage\StorageInterface', function() { return new SessionStorage(Yii::$app->session, 'primary-cart');

});

And after all we can retrieve the cart object from the container manually in our own controllers, widgets, and other places:

$cart = Yii::createObject('app\cart\ShoppingCart')

But every controller and other object will be created via the createObject method inside the framework. And we can use injection of cart via the controller constructor:

class CartController extends Controller {

private $cart;

public function construct($id, $module, ShoppingCart $cart, $config = [])

{

$this->cart = $cart;

parent:: construct($id, $module, $config);

}

// ...

}

Use this injected cart object:

public function actionDelete($id)

{

$this->cart->remove($id);

return $this->redirect(['index']);

}

See also

• For more information about DIP refer to [https://en.wikipedia.org/wiki/Dependencv inversion principle](https://en.wikipedia.org/wiki/Dependency_inversion_principle)

• In order to learn more about dependency injection container refer to [http://www.yiiframework.com/doc-2.0/gurde-concept-di-container.html](http://www.yiiframework.com/doc-2.0/guide-concept-di-container.html)