Анализ реализации задания для Matryoshka Games

1. Customers, у всех в методе Awake идет одинаковая инициализация, у всех есть static Instance, все три наследуются от монобеха. Это явное дублирование кода, вместо этого реализуется

public class Manager<T> : MonoBehaviour where T : Manager<T> {

от которого наследуются все подобные менеджеры

Внутрь ложатся все методы и поля Instance, Инициализация и тд

Так же добавил бы метод Exist для осуществления проверок на наличие менеджеров, так как обращение к не существующему менеджеру создало бы копию

public static bool Exist {

get { return (\_instance != null); }

}

Ну и раз код расширяется – можно ввести переменную

virtual protected bool DontDOnLoad { get { return false; } } , которую будут перегружать для того, чтобы некоторые менеджеры могли существовать после смены сцен

(в инициализации дописываем)

if (DontDOnLoad) {

GameObject.DontDestroyOnLoad(this.gameObject);

}

1. Нет использования #region внутри классов, сейчас это не важно, но когда классы разрастутся – будет гораздо проще, когда они будут внутри полей Public, Private, Fields и тд
2. В огромном количестве мест допущено некорректное использование модификаторов доступа – зачастую нет необходимости модифиактора public, на его месте должно быть

[SerializeField] protected/private

1. CustomersController const string CUSTOMER\_PREFABS\_PATH = "Prefabs/Customer";

несмотря на то, что данный путь используется исключительно в данном классе – на мой взгляд более понятном подходом будет иметь единый статик класс в котором будут размещаться все необходимые пути для объектов

1. Общее замечание, в некоторых методах есть обращение к статическим экземплярам классов(контроллерам), которое осуществляется по типу

var name = Contorller.Instance;

name.DoSomeTh();

На мой взгляд это излишество, можно просто обращаться к методу через экземпляр

Contorller.Instance.DoSomeTh();

1. Класс OrdersController

Метод Init

var ordersConfig = Resources.Load<TextAsset>("Configs/Orders");

Текстовый путь должен быть константой и вынесен в статик-класс констант(или хотя бы в переменную константу)

1. LoseWindow WinWindow – дублирование кода, оба должны быть унаследованы от общего класса (условная Panel) с виртуальными методами инициализации, показа и скрытия

Возможно это должна быть вообще единая панель, в которой в зависимости от приходящего значения активируются необходимые элементы и надписи.

1. Customer – константа. П.4
2. FoodPlacer –очень не нравится реализация когда поля просто вписаны в объекты на сцене(даже не префабы), а по ним и сравнение идет(в FoodPresenter) и перебор по именам, в общем если кто-то поменяет букву в объекте на сцене все сломается. Поэтому имена выносятся в enum и в объекте выбирается
3. Дублирование спрайтов, cancel off можно добиться просто небольшим затемнением cancel on
4. pan\_sosige\_3\_raw, отличается размерами от двух других сосисок.
5. Очень некомфортная система работы с Config-ами, на мой взгляд простой ScriptableObject решил бы эту проблему проще и с ним было бы удобнее работать и разработчиками и геймдизайнерам.