Название папок.Для названия папок использовать только строчные буквы. Для разделения слов использовать символ “\_”.

В папке **resources**. Нужно хранить элементы, которые в дальнейшем могут быть подгружены через код(пока там все элементы)

Каждую группу элементов нужно хранить в своей папке. Группы элементов:

- *animation*(анимации)

- *audio* (звуки)

-*images* (изображения)

-*prefabs* (префабы) сцены

В папке **scenes** нужно хранить только сцены. Файлы с расширением .scene

Папка **scripts** для скриптов .

Game – точка входа. Все работает через него.

Есть категории:  
 - *animation*

-*bonus*

-*fruits*

-*movement*

-*player* (нет)

-*ui* (нет)

-*sound*

-*utils* (нет)

В почти в каждой категории есть интерфейс, (нет) – означает отсутствие. Его и нужно реализовывать для расширения функционала. Интерфейс нужно хранить в отдельной папке, чтобы он не «сливался с другими скриптами»

Если интерфейс в папке один, то его реализации вместе, если их 2 и более, как в папке **bonus**, тогда нужно создавать для каждой группы папку.

Скрипты, которые будут навешиваться на ноды, должны в названии иметь иметь *Component*.

Для объектов, которые будут часто добавляться и уничтожать лучше использовать встроенные **NodePool**

В коде имена переменных начинать с “\_”, свойства (**@property**) как обычно. Имена переменных с маленькой буквы. Имена методов *camelCase*.

Не забывает логические блоки отделять пробелами, чтобы не было полотенца из кода.

Важно очищать ресурсы. Создавайте для этого метод clear, в нем отпишитесь от всех событий. Обновите содержимое ссылочных типов: массивов, объектов, классов и т.п.

Не забывайте про модификаторы доступа. Они нужны для того, чтобы проще ориентироваться по классу. В каждом языке программирования свои нюансы. C# нет модификатора доступа (private по умолчанию). TS нет модификатора доступа (public по умолчанию)   
 Класс:  
 - приватные поля

- конструктор

- свойства

- публичные методы

- приватные методы

- метод clear() (публичный. Но он должен быть в конце)