1. Использовать версию Unity указанную в вашем трелло.
2. Название ветки формируется ТипРаботы/имя-перваяБукваФамилии/название-задачи
3. Все имена переменных, классов и т.д. должны использовать американскую версию английского языка.
4. Использовать пробелы, вместо табуляции (4 пробела) (В MVS гор. клав. ctrl R + W)
5. Названия приватных полей класса (в том числе **readonly**) должны использовать **camelCase** и впереди ставить нижнее подчеркивание. (\_**camelCase** )
6. Названия методов и свойств класса должны использовать **PascalCase** и начинаться с заглавной буквы и не должны совпадать с зарезервированными именами Unity и .net.
7. Все константы должны быть названы с использованием **snake\_case** полностью в **ВЕРХНЕМ\_РЕГИСТРЕ** и сгруппированы в специально отведенном для них объекте.
8. Названия интерфейсов должны использовать **PascalCase**, иметь префиксную букву “I” и быть либо существительными, либо прилагательными с постфиксом ...able. (**IEnumarable**)
9. Названия типов (в том числе классов и перечислений) должны использовать **PascalCase** .
10. Каждый отдельный объект(класс, интерфейс и т.д.) должны быть в отдельном файле (скрипте)
11. Названия параметров функции и локальные переменные должны использовать **camelCase** и начинаться с маленькой буквы.
12. Имена булевских переменных должны начинаться с “is” (или соответствующих форм глагола “to be”), “has”, “can”, “should”. Остальным типам запрещено использовать слова “is”, “has”, “can”, “should” в качестве префикса в имени.
13. Запрещено использовать любые виды сокращений в именах классов, методов и переменных (за исключением стандартных и устоявшихся сокращений вроде id, email, min, max).
14. Запрещено в именах переменных, методов, свойств и типов использовать специальные символы, отличные от строчных и прописных английских букв в т.ч цифры.
15. Основная идея в том, что папки и все “Unity-объекты” (материалы, префабы, скрипты и т.д.) должны использовать **PascalCase,** с заглавной буквы, а все “сторонние” объекты (файлы текстур, аудио файлы и т.д.) - **snake\_case**.
16. Запрещено использовать пробелы в именах папок и любых файлов внутри проекта.
17. Названия .cs файлов должны использовать **PascalCase**, а также полностью совпадать с именем главного типа в этом файле. Запрещено в названии файла скрипта использовать слово Script.
18. Все регионы пишутся в формате **PascalCase.**
19. Код необходимо “оборачивать в регионы (#region)” и в том порядке, который указан ниже
20. Приватные типы данных (struct, enum и т.д.), которые используются только в этом классе. “PrivateData”
21. Для полей “Fields”
22. Для свойств и индексаторов “Properties”
23. Для методов связанных с жизненным циклом класса (конструкторы, деструкторы) “ClassLifeCycles”
24. Для Unity методов (Start, Awake, OnTrigger) “UnityMethods”
25. Для методов “Methods”
26. Для реализации каждого интерфейса отдельный регион с названием данного интерфейса для группы его методов
27. Порядок расположения пустых строк внутри файла
28. Между регионами - 2 пустые строки.
29. Внутри региона отступ от директив “region” - 1 пустая строка.
30. Между подключенными пространствами имен и объявлением класса - 2 пустые строки.
31. Между методами (свойствами, классами) внутри региона - 1 пустая строка.
32. Между смысловыми “блоками” кода внутри метода - 1 пустая строка.
33. В конструкции “if else” пишем else на новой строке без пропуска на пустую.
34. Группа атрибутов относящейся к одному объекту должна указываться в одних квадратных скобках через запятую.
35. Длинные строки кода, более 120 символов, необходимо разносить на несколько строк. При это необходимо придерживаться следующих общих правил:
36. перенос строки должен быть *после* оператора.
37. перенос строки должен быть *после* запятой.
38. каждая строка, начиная со второй, должна находиться на следующем уровне табуляции по сравнению с первой строкой.
39. Каждая фигурная скобка должна находиться на отдельной линии, причём соответствующие фигурные скобки должны быть выровнены по вертикали.
40. Обязательно наличие ровно одного пробела в следующих случаях:
41. с обеих сторон бинарного оператора, кроме оператора “.”.
42. с обеих сторон “?” и “:” тернарного оператора.
43. с обеих сторон “=>” в случае lambda expressions и expression body.
44. с обеих сторон “{” и “}”.
45. с обеих сторон “:” при наследовании, реализации интерфейсов и указании ограничений параметров generic.
46. после операторов if, while, for, foreach, switch, using.
47. между каждым из атрибутов для члена класса.
48. после “:” при использовании именованного аргумента метода.
49. после запятой при перечислении параметров метода.
50. после // и началом текста комментария.
51. Запрещено в одной строке объявлять более одной переменной.
52. При написании дробного числа необходимо целиком писать и целую, и дробную части, а также использовать соответствующую букву на конце в нижнем регистре.
53. Оператор switch обязательно должен содержать default case, который заканчивается break и расположен в самом низу switch.
54. Любое перечисление должно содержать None. Если это int перечисление, то None должен соответствовать 0.
55. Для всех объектов перечисления необходимо выставить значения и выравнивать по знаку равенства
56. Метод или конструктор не должен принимать более 6 параметров
57. Комментарии, как и весь код, должны быть написаны исключительно на английском языке (американской версии).
58. В проекте обязательно использование namespace верхнего уровня, сформированного на основании имени проекта.
59. Каждый член типа должен иметь модификатор доступа с минимально возможной видимостью
60. Запрещено использование любых “магических” чисел и строк в коде.
61. Запрещено в игровой логике использовать механизм перехвата исключений (try-catch), однако разрешается бросать исключения.
62. Запрещено использование операторов “??” и “?.” для наследников класса UnityEngine.Object.
63. По возможности минимизировать проверку на null для наследников класса UnityEngine.Object.
64. Импортируйте все пространства имён, которые вы используете в коде.
65. Подписка/отписка на ивенты должна располагаться в методе OnEnable/OnDisable.
66. Запрещено подписываться на ивенты неявно (например, указанием вызываемого метода в компоненте внутри префаба, как это возможно для Unity компоненты Button).
67. Каждому подписанию на ивент должна соответствовать отписка от ивента.
68. Нежелательно использовать в циклах и методах операторы break, return и continue.
69. При объявлении локальной переменной, предпочтение отдавать ключевому слову var, нежели явному указанию типа данных
70. Отдавать предпочтение интерполяции строк нежели string.Format
71. Перед вызовом события всегда проверять на null, если не был применен паттерн Null-object
72. По умолчанию при объявлении класса выставлять ключевое слово sealed
73. Запрещается для события создавать свой собственный делегат.
74. Запрещается использовать ключевое слово this в явном обращении к объекту
75. По возможности минимизировать сериализацию сложных объектов через инспектор
76. Структура папок проекта:
77. Fonts
78. Materials
79. Prefabs
80. Resources
81. Scenes
82. Scripts
83. Textures
84. Все объекты находящиеся в папках проекта должны быть сгруппированы в подпапки
85. Необходимо удалять неиспользуемые ресурсы из проекта
86. Запрещается в репозиторий помещать отладочный и закомментированный или не рабочий код
87. По возможности минимизировать наследование от класса MonoBehaviour
88. Предпочтительно использовать expression body вместо однострочных тел
89. Запрещается оставлять пустые тела методов
90. При явном переопределении метода всегда вначале вызывается базовый функционал, за редким исключением базовый функционал вызывается в конце метода
91. Запрещается использовать авто свойства, если они не скрывают один из аксессоров доступа
92. Запрещается работать с префабами в режиме AutoSave.
93. Модификатор override ставится после модификатора доступа.
94. Поля которые инициализируются в методе, должны быть в том же порядке, как они объявлены в классе / структуре.
95. Классы для методов расширения должны иметь модификатор **partial**
96. Объявление перечислимого типа, предназначенного для идентификации битовых флагов, должно содержать атрибут [Flags].
97. При использовании битовых флагов для проверки содержит ли переменная необходимую маску запрещено использовать Enum.HasFlag. Вместо этого нужно использовать “==” (если нужно совпадение с каждым битом из маски) или “!= Enum.None” (если нужно совпадение с любым битом из маски). Метод Enum.HasFlag выполняет упаковку (boxing) входного значения
98. Запрещено использовать ключевое слово unsafe. Использование этого ключевого слова снижает производительность, а также позволяет намного легче “выстрелить себе в ногу”.
99. При написании юнит тестов необходимо создавать папку Test рядом с тем функционалом, который тестируется и создавать для него Assembly Definitions
100. Запрещено использовать expression-body для реализации методов, если получившаяся строка кода занимает более одной строки (более 120 символов).
101. Следует отдавать предпочтение перегрузке метода или конструктора для аргументов метода, а не использованию параметров по умолчанию.
102. Перегружать методы необходимо лишь тогда, когда они совпадают по назначению и различаются лишь входными параметрами.
103. Запрещено использование логики процентов внутри кода проекта (в том числе, в [SerializeField]). Вместо процентов везде должны использоваться дробные числа (например, вместо интервала 0% - 100% должны использоваться дробные числа из интервала [0, 1]). Дробные числа существенно удобнее для работы программиста. Также это позволяет добиться однообразия подхода к подобным задачам на разных проектах.
104. Следует избегать значения float переменных на неравенство. При необходимости необходимо использовать функцию Mathf.Approximately() (оператор “!=” использовать запрещено).
105. Нужно избегать разделения объявления и инициализации локальных переменных.
106. Локальные методы всегда должны быть объявлены внизу метода, для читабельности

## Ссылка на Example Code

<https://github.com/FitiLsan/Code-Convention-Example>