Нейминг и управление ресурсами в Unity

Пример создания папок в проекте:

```
Assets/
├── 2D/
│   ├── Sprites/
│   ├── Textures/
│   └── Materials/
├── Audio/
│   ├── Music/
│   └── SFX/
├── Prefabs/
├── Scenes/
│   ├── Core/
│   ├── Levels/
│   └── UI/
├── Scripts/
│   ├── Core/
│   ├── Gameplay/
│   ├── UI/
│   └── Utils/
├── UI/
│   ├── Sprites/
│   ├── Fonts/
│   └── Icons/
└── Resources/
```

Правила нейминга:

1. Сцены (Scenes):

// ПЛОХО
level1.unity
scene2.unity
menu.unity

// ХОРОШО
SC_MainMenu.unity
SC_Level_01_Forest.unity
SC_Level_02_Cave.unity
SC_Cutscene_Intro.unity

Правила:

Префикс SC_ для идентификации сцен
Уровни: SC_Level_[Номер]_[НазваниеЛокации]
Короткое и четкое описание локации.

2. Скрипты и классы:

// ПЛОХО
playerScript.cs
gameManager.cs
button1.cs

// ХОРОШО
PlayerController.cs
GameManager.cs
UI_MainMenuButton.cs
Enemy_Orc_Warrior.cs

Правила:

PascalCase для имен классов
Описательные названия
Префиксы для категорий: UI_, AI_, System_

3. Префабы (Prefabs):

// ПЛОХО
player.prefab
enemy.prefab
item.prefab

// ХОРОШО
PF_Player_Hero.prefab
PF_Enemy_Orc_Archer.prefab
PF_Item_HealthPotion.prefab
PF_UI_Button_Start.prefab

Правила:

Префикс PF_ для префабов
Структура: PF_[Тип]_[Подтип]_[КонкретноеИмя]
Группировка по функциональности

4. Материалы (Materials):

// ПЛОХО
mat1.mat
red.mat
stone.mat

// ХОРОШО
M_SpriteLit_Stone_Granite_01.mat
M_SpriteLit_Metal_Iron_Rusty.mat
M_UI_Button_Normal.mat
M_SpriteUnlit_Character_Skin_Human.mat

Правила:

Префикс M_ для материалов
Структура: M_[ТипМатериала]_[КонкретноеИмя]_[Подтип]
Описание текстуры/поверхности
Нумерация для вариаций

5. Текстуры и спрайты:

// ПЛОХО
image1.png
sprite.png
texture.jpg

// ХОРОШО
T_Stone_Wall_Diffuse_1024.png
T_Character_Hero_Albedo.png
S_UI_Icon_Health.png
S_Environment_Tree_01.png

Правила:

T_ для текстур, S_ для спрайтов
Указание разрешения для текстур
Четкое описание содержимого

6. Аудио файлы

// ПЛОХО
sound1.wav
music.mp3
effect.ogg

// ХОРОШО
A_Music_MainTheme.wav
A_SFX_Player_Jump.wav
A_SFX_Enemy_Death_01.wav
A_UI_Button_Click.wav

Правила:

A_ для аудио файлов
Категоризация: Music, SFX, UI
Описание источника звука

Нейминг и управление коммитами в Fork

Шаг 1: Выберите тип изменения
В начале сообщения укажите тип, который описывает суть изменений:

feat: Добавление новой функциональности.

fix: Исправление ошибки.

docs: Изменение в документации.

style: Правки по стилю (форматирование, пробелы, запятые и т. д.).

refactor: Рефакторинг кода без исправления ошибок или добавления нового функционала.

perf: Оптимизация производительности.

test: Добавление или обновление тестов.

build: Изменения, касающиеся сборки проекта.

ci: Настройка или изменение CI/CD.

chore: Прочие задачи (например, изменения в .gitignore).

revert: Откат предыдущего коммита.

Пример: `fix(art): fix sprite validation issue`

Шаг 2: Укажите область изменений (scope, опционально)

Scope помогает уточнить, к какой части проекта относится изменение. Используйте универсальные области, чтобы сразу было понятно, что именно затронуто:

art: Работа над визуальным контентом (спрайты, материалы, текстуры, 3д модели тд)

ui: Пользовательский интерфейс.

core: Основной геймплейные системы

code: Работа над скриптами

config: Файлы конфигурации (настройка проекта)

docs: Документация.

db: Изменения структуры файлов внутри проекта

build: Скрипты сборки или сборочные файлы.

asset: Добавление сторонних ассетов (например 2d controller из Unity asset store)

Пример: `feat(code): add 2d controller for character`

Шаг 3: Напишите краткое описание
Описание должно быть коротким и ясным.

```
feat(code): add 2d controller for character
fix(art): fix sprite validation issue
perf(art): removal of excess materials
```