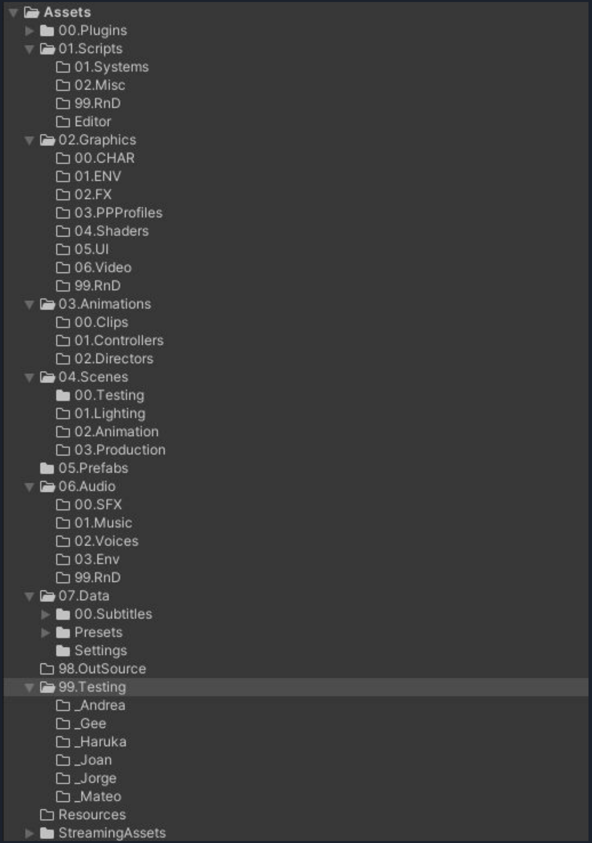
Development Team Conventions

Project Folder Structure



```
Assets
  00.Plugins
  01.Scripts
    01.Systems
    02.Misc
    99.RnD
    Editor
  02.Graphics
    00.CHAR
    01.ENV
    02.FX
    03.PPProfiles
    04.Shaders
    05.UI
    06.Video
    99.RnD
  03.Animations
    00.Clips
    01.Controllers
    02.Directors
  04.Scenes
    00.Testing
    01.Lighting
    02.Animation
    03.Production
  05.Prefabs
  06.Audio
    00.SFX
    01.Music
    02.Voices
    03.Env
    99.RnD
  07.Data
    00.Subtitles
    Presets
    Settings
  98.OutSource
  99.Testing
    _Andrea
    _Gee
    _Haruka
    _Joan
    _Jorge
    _Mateo
  Resources
  StreamingAssets
```

all The Testing Should be Contained within The Folder “99.Testing” and within it A folder called _DeveloperName

If The Developer is Creating a Bigger System, Animation, Shader,etc that is not ready for production it should be placed in the subfolder 99.RnD of the Corresponding Folder.
e.g.:

Assets/01.Scripts/99.RnD/Subtitle System

Asset Files Naming Convention

Scenes	SXX_Name (XX increase by 10 unless its an intermediary scene)
Materials	MAT_Name
Shader	SH_Name
Font	FONT_Name
RenderTextures	RT_Name
SkeletalMesh	SKM_Name
StaticMesh	SM_Name
Visual Effect	VFX_Name
Animator Controller	AController_Name
AnimationClip	AClip_Name
AvatarMask	Mask_Name
Pipeline Asset	URP_Name

Asset Files Naming Convention

SFX File	SFX_Name /SFX_Name_3D
Voice File	V_Name/ V_Name_3D
Music File	M_Name/M_Name_3D
Environment Sound	ENV_Name/ ENV_Name_3D
Video File	VID_Name
Timeline Asset	DIR_Name
Timeline Signal	SIG_Name
Subtitle File	SUB_Name
Post Process Profile File	PP_Name
Prefab	PF_Name
Textures	T_Name

Asset Files Naming Convention

Textures Color	T_C_Name
Textures Normal	T_N_Name
Textures Emmisive	T_E_Name
Textures Mask	T_M_Name
Textures Roughness	T_R_Name
Textures Metallic	T_MT_Name
Textures Specular	T_S_Name
Textures Displacement	T_D_Name
Textures Ambient	T_AO_Name
Textures Height	T_H_Name
Textures Flow	T_F_Name
Textures Liight map	T_L_Name

Build Naming Conventions

1. Build Naming Rule(ビルドネーミング規定）

①EXE

- 3-6 digit capital arphabets + _VerXX+.exe (アルファベット大文字６文字まで＋_Ver＋２桁番号+.exe)
- Arphabets is short name of Projecttitle (アルファベットはタイトルの略称)

ex: Thank you for sharing your world→TYFSYW_Ver01.exe

②Compressed file(圧縮ファイル名）

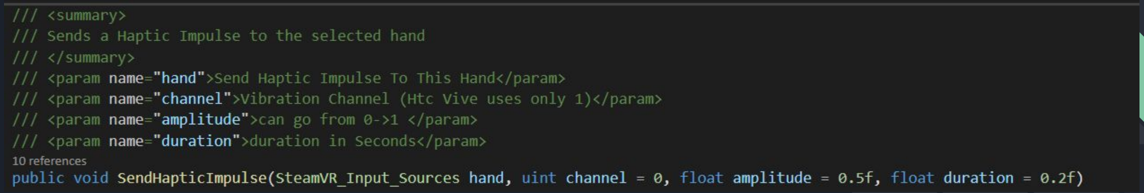
- 3-6 digit capital arphabets + _VerXX+.rar (アルファベット大文字６文字まで＋_Ver＋２桁番号+.rar)

ex: Thank you for sharing your world→TYFSYW_Ver01.rar

注意: exeとBuild名は変えない

Scripting Documentation

- All The Code should be Commented as much as possible
  - Every Class Should have a short description at the beginning of the class for future reference
  - every Exposed Function should have a VS Summary



```
/// <summary>
/// Sends a Haptic Impulse to the selected hand
/// </summary>
/// <param name="hand">Send Haptic Impulse To This Hand</param>
/// <param name="channel">Vibration Channel (Htc Vive uses only 1)</param>
/// <param name="amplitude">can go from 0->1 </param>
/// <param name="duration">duration in Seconds</param>
10 references
public void SendHapticImpulse(SteamVR_Input_Sources hand, uint channel = 0, float amplitude = 0.5f, float duration = 0.2f)
```

  - Whenever system that will be operated by the Design or Producers Team is created, there should be a Documentation on Confluence/GDrive for its correct operation.

Coding Convention

NameSpaces

Namespaces should use PascalCase except for Acronyms which should be all UpperCase

e.g.:

```
namespace KVRL.Utilities.Subtitles
{

}
```

The Namespace of any KVRL Project should be

```
namespace KVRL.ProjectName
{

}
```

Classes and Interfaces Should use PascalCase

e.g:

```
class MyClass
interface IPausable
```

Methods should use PascalCase

Coding Convention

All non Static Fields should use camelCase and avoid the use of underscores

```
        public int publicField
        private int privateField
        int privateField
        protected protectedField
```

Static Fields should use PascalCase

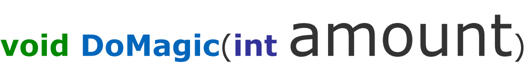
```
        public static int StaticField;
```

Properties should use PascalCase

```
public bool CanBePaused
   {
       get { return !isPaused; }
       set { }
   }
```

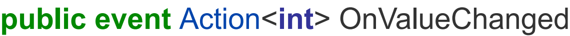
Coding Convention

Parameters should use camelCase



```
void DoMagic(int amount)
```

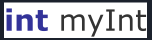
Actions Should use PascalCase



```
public event Action<int> OnValueChanged
```

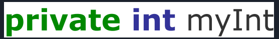
Access Level Modifiers should be explicitly written down

Avoid:



```
int myInt
```

Prefer



```
private int myInt
```

All Braces should get their own Line(C# convention)

Always Include Braces to conditional Statements irrespective of the number of lines required

Version Control

For Version Control we will use Github:

There should always be 3 main Branches

1. Master
   a. main Branch
2. Dev
   a. Development Branch
   b. Each developer should create their own branch if they´re doing long term work.
3. Stable
   a. Use this to hold the latest version of the stable build

Commit Messages

THERE SHOULD ALWAYS BE A DESCRIPTIVE MESSAGE WITH EACH COMMIT

Message Prefixes:

- ADD-AssetType-Message
  - Use this when your´e adding a new Feature, assets, Etc
- FIX-AssetType-Message
  - Use This for a specific Error Fix
- DEV-AssetType-Message
  - Use This When You're doing a change to an existing System/Asset

- REMOVE-AssetType-Message
  - use This When your´e removing an Asset
- TEST-AssetType-Message
  - use this when you're sharing a test

[Optional]

Replace Asset Type for the type of asset when the Commit only includes Change to an specific asset/asset group

[Mandatory]

Replace Message for a descriptive Message