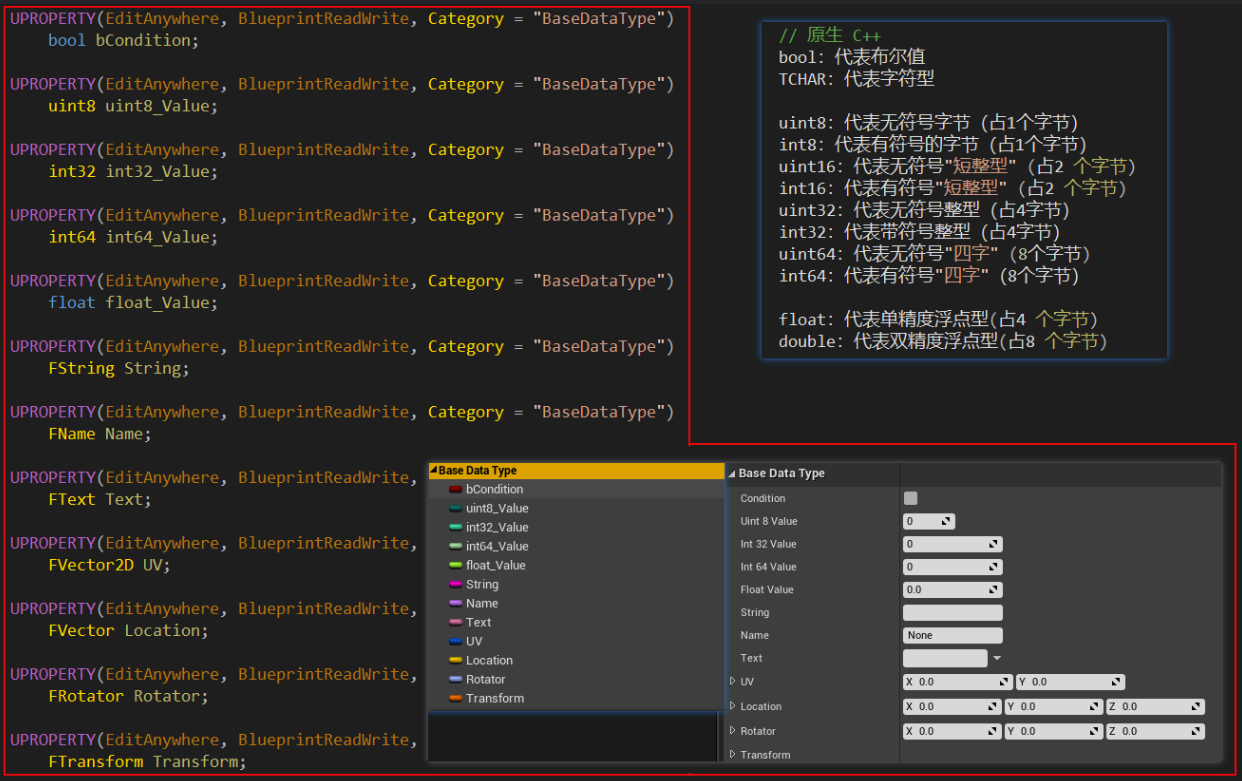
UnrealEditor案例集

# 基本数据类型

[字符&字符串](https://www.cnblogs.com/KillerAery/p/14385574.html#%E8%BD%AC%E6%8D%A2%E5%AD%97%E7%AC%A6%E7%BC%96%E7%A0%81)

<https://blog.csdn.net/Jason6620/article/details/126503682>



## 类型转换

### Lex -> FString (LexToString)

FString aa = LexToString(\*phComBox->GetSelectedItem());

### int转string的多种方式

1. 1.
2. int a = 10;
3. char \*intStr = itoa(a);
4. string str = string(intStr);
5. 2.
6. int a = 10;
7. stringstream ss;
8. ss << a;
9. string str = ss.str();
10. 3. C++11  （推荐，如果支持的话）
11. #include <string>
12. std::string s = std::to\_string(42);
13. 4. C++ 98即可   （不支持C++11就用这个或者2，233 ）
14. #include <sstream>
15. #define SSTR( x ) static\_cast< std::ostringstream & >( \
16. ( std::ostringstream() << std::dec << x ) ).str()
17. 5. Boost
18. #include <boost/lexical\_cast.hpp>
19. int num = 4;

## TCHAR：UE4通过对char和wchar\_t的封装

TCHAR就是UE4通过对char和wchar\_t的封装

char ANSI编码

wchar\_t 宽字符的Unicode编码

使用 TEXT() 宏包裹作为字面值

TCHAR\* TCharString = TEXT("Hello, World!");  
  
// 引擎字符串(TCHAR\*) -> ANSI字符串(char\*)  
char\* CharString = TCHAR\_TO\_ANSI(TCharString);  
  
// 引擎字符串(TCHAR\*) -> std::string  
std::string str = TCHAR\_TO\_UTF8(TCharString);  
  
// 引擎字符串(TCHAR\*) -> wchar\_t\*  
wchar\_t\* WCharString = TCHAR\_TO\_WCHAR(TCharString);  
  
// ANSI字符串(char\*) -> 引擎字符串(TCHAR\*)  
TCHAR\* TCharString1 = ANSI\_TO\_TCHAR(CharString);  
  
// wchar\_t\* -> 引擎字符串(TCHAR\*)  
TCHAR\* TCharString2 = UTF8\_TO\_TCHAR(WCharString);  
TCHAR\* TCharString3 = WCHAR\_TO\_TCHAR(WCharString);

## FNames：常用作标识符

常用作标识符等不变的字符串（如：资源路径/资源文件类型/骨骼名称/表格行名等）

比较字符串操作非常快

即使多个相同的字符串，也只在内存存储一份副本，避免了冗余的内存分配操作

（不区分大小写）

### 使用

* 初始化

FName TestHUDName = FName(TEXT("ThisIsMyTestFName"));

* 比较

CompareFloat = TestFName.Compare(OtherFName);

* 搜索判断

if( FName(TEXT("pelvis"), FNAME\_Find) != NAME\_None )**{}**

### FString <-> FName 转换

FString TestHUDString;  
TestHUDName = FName(\*TestHUDString);  
TestHUDString = TestHUDName.ToString();

### Int<->FText

#### Int 转为 FText

通过FString::FromInt(int)转换成FString，再转换成SEditableTextBox可编辑文本框能接收的FText类型。

int vvv = pow(8, setMapSize->GetValue());  
SizeValue->SetText(FText::*FromString*(FString::*FromInt*(vvv)));

#### FText/FString 转为 Int

这里调用控件传入的SEditableTextBox文本变量，

注意：Atoi调用变量前面要加\*指针（SlateMain.cpp）

MatObj->MaxTextureSize = FCString::*Atoi*(\*SizeValue->GetText().ToString());

## FText：静态字符串，处理文本本地化、UI 文本显示等

FText 是一种静态字符串，在UE4中可以负责处理文本本地化、UI 文本显示等

支持文本本地化

提高文本渲染的性能

较快的copy操作

### FString <-> FText

TestHUDText = FText::*FromString*(TestHUDString);  
TestHUDString = TestHUDText.ToString();

### new FString使用中文

equalSel.Add(MakeShareable(new FString ("text"))); //只能用英文

equalSel.Add(MakeShareable(new FString(LOCTEXT("InKey", "大于").ToString())));

### FName -> FText

TestHUDText = FText::*FromName*(TestHUDName);

### FText 使用中文方法

SAssignNew(equalText,SEditableTextBox).Text(**FText**::FromString(TEXT("大于等于"))).MinDesiredWidth(10)

#### ★闭坑-字符对比判断失败

两个字符做比较时需要确定字符类型一至,如下判断字符中文,需要对中文字符用TEXT()封装

void SSlateMain::comp\_equalText(TSharedPtr<FString> Item, ESelectInfo::Type SelectInfo)  
{  
 //该函数用于SComboBox下拉列表修改事件,在这里判断对比字符是需要TEXT("")包装的  
 if (\*Item == TEXT("等于"))  
 **{** equalText->SetText(FText::*FromString*(TEXT("等于")));  
 **}**

}

### 比较判断



## FString

### 初始化

FString str1("Hello!");  
FString str2 = TEXT("Hello World!");  
FString str3 = FString::*Printf*(TEXT("I am %d years old"), 18);

### 传入整数 %i

int cou = 7;

FString::*Printf*(TEXT("共处理 %i 个贴图文件"), cou);

传入控件变量文本需要ToString转换,并在变量前面加指针解引

FString::Printf(TEXT("Load Form:( %s )successful."),\*(matFile->GetText().ToString()))

### 比较判断

if(MyFString.Equals(OtherFString, ESearchCase::CaseSensitive)){...} // 大小写敏感  
if(MyFString.Equals(OtherFString, ESearchCase::IgnoreCase)){...}  
if(MyFString == OtherFString)**{**...**}**

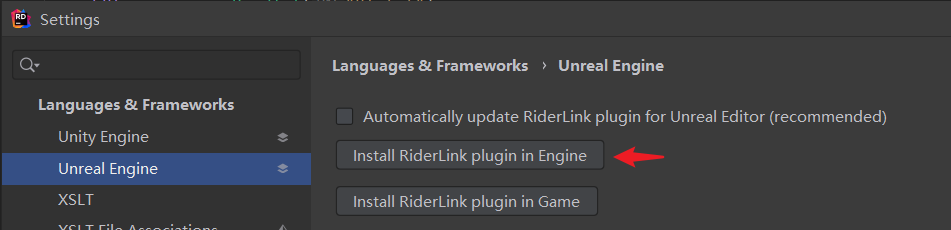
## 字符串操作

# 插件

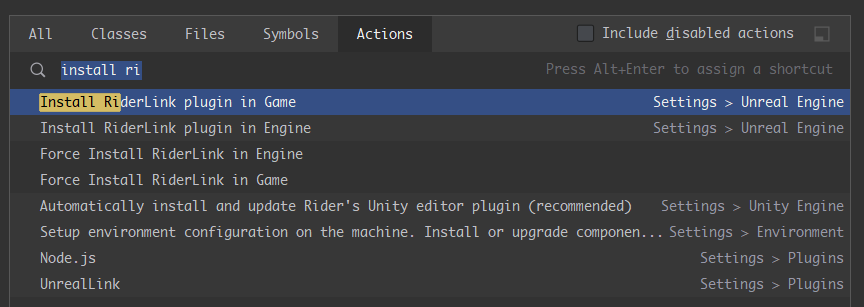
## UE4插件移植到UE5

UE5编译出现‘RD’报错时需要安装RiderLink in Game

安装RiderLink plugin in Engine

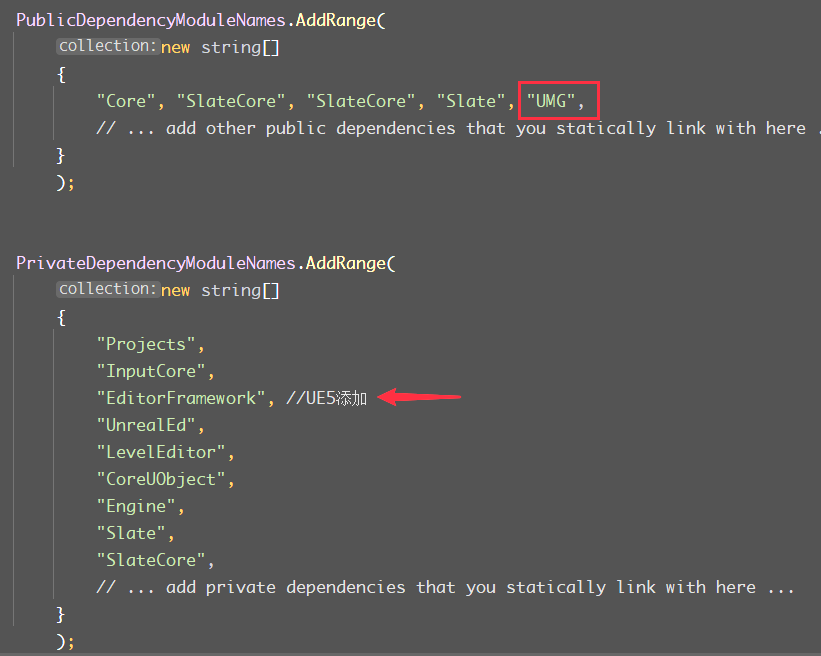


也可以(ctrl shift A)打开命令输入窗口来快速搜索



UE4插件移植到UE5出现的问题

SceneTools\_W\_P.Build.cs文件添加内容，"UMG"可以不用添加



函数报错，InvokeTab此函数在UE5弃用，改用TryInvokeTab

SceneTools\_W\_P.cpp(92): [C2039] 'InvokeTab': is not a member of 'FGlobalTabmanager'

在SceneTools\_W\_P.cpp添加包含

#include "Framework/Docking/TabManager.h"

并修改报错的InvokeTab为TryInvokeTab

FGlobalTabmanager::Get()->TryInvokeTab(SceneTools\_W\_PTabName);

## Slate UI

### 中英文文本

SNew(STextBlock).Text(FText::FromString("Align:"))

SNew(STextBlock).Text(LOCTEXT("key", "层级与对齐：")) //支持中文

SNew(STextBlock).Text(FText::FromString(TEXT("层级与对齐："))) //支持中文

### FLinearColor(Rf,Gf,Bf,Af)

设置控件颜色写法

SAssignNew(equalText,SEditableTextBox).Text(FText::*FromString*(TEXT("大于等于"))).MinDesiredWidth(10)  
.BackgroundColor(FLinearColor(0.1f,0.1f,0.1f,0.0f))  
.ForegroundColor**(**FLinearColor(0.9f,1.0f,0.8f,1.0f)**)**

### SSpinBox 滑动整数控件

.cpp中创建传递数值

+ SHorizontalBox::*Slot*().VAlign(VAlign\_Center)  
 .HAlign(HAlign\_Left).AutoWidth().Padding(2.0f)  
**[** //像该案例用数值拖动事件传入val值可以不需要用SAssignNew再获取值，直接用SNew即可//SAssignNew(searchSize, SSpinBox<int>).MaxValue(4096).MinValue(32).Value(1024).ToolTipText(SNew(SSpinBox<int>).MaxValue(4096).MinValue(32).Value(1024).ToolTipText(  
 LOCTEXT("resetseachVal", "输入查找贴图尺寸"))

.OnValueChanged**(**this, &SSlateMain::**SetSizeValue)** //数值变化事件绑定函数  
**]**

SetSizeValue函数,需要传入一个int32类型作为控件值传入才可以匹配OnValueChanged

void SSlateMain::SetSizeValue(FPlatformTypes::int32 val)  
**{** int vvv = pow(2,val);   
 SizeValue->SetText(FText::*FromString*(FString::*FromInt*(vvv)));  
**}**

.h中定义变量承接数值，及事件函数

//定义函数

public:

void SetSizeValue(FPlatformTypes::int32 val); //设置尺寸值次方计算，val是传入的本控件整数值

protected:

//定义SSpinBox传入变量  
 TSharedPtr<SSpinBox<int>> searchSize; //重置贴图大小查找尺寸

TSharedPtr<class SEditableTextBox> SizeValue; //贴图大小次方计算

调用变量数值

int Rows = searchSize->GetValue();

### SComboBox

#### .cpp控件主体,

在控件主体前添加固定列表成员

compOpations.Add(MakeShareable(new FString("Default")));  
compOpations.Add(MakeShareable(new FString("NormalMap")));

紫色字段为承接返回值变量

+ SHorizontalBox::*Slot*().Padding(3.0f)  
 .HAlign(HAlign\_Left).VAlign(VAlign\_Center).AutoWidth()  
**[** SAssignNew(compSettingComBox, SComboBox<TSharedPtr<FString>>) //用于成员添加即时刷新列表,如果是固定成员无需传递变量,用SNew即可  
 .OptionsSource(&compOpations) //存放下拉列表成員  
 .OnGenerateWidget(this, &SSlateMain::comp\_ComboItem)  
 .OnSelectionChanged(this, &SSlateMain::comp\_ComboChanged)  
 .IsFocusable(true)[  
 ◎ SAssignNew(compComBoxCont, SBox) //如果需要每个成员传递变量,用于获取成员信息,使用SAssignNew指定变量  
 ● SNew(SBox) //如果不需要获取成员信息只要用SNew()  
 ]  
**]**

用于显示下拉菜单内容,所有SComboBox的.OnGenerateWidget(...)共用该函数

TSharedRef<class SWidget> SSlateMain::comp\_ComboItem(TSharedPtr<FString> InItem)  
**{** return SNew(STextBlock).Text(FText::*FromString*(\*InItem));  
**}**

#### SComboBox控件返回成员名

SAssignNew**(**phComBox, SComboBox<TSharedPtr<FString>>**)**

FString aa = LexToString(\*phComBox->GetSelectedItem());  
GEngine->AddOnScreenDebugMessage(-1, 3.f, FColor::Green, aa);

phComBox->RefreshOptions(); //刷新下拉列表

#### .h定义变量及事件函数

protected:   
 TSharedPtr<SComboBox<TSharedPtr<FString>>> compSettingComBox; //下拉列表,调用->RefreshOptions()可用于刷新列表  
 TArray<TSharedPtr<FString>> compOpations; //ComBox存放下拉列表成員数组类型  
 TSharedPtr<SBox> compComBoxCont; //传递SBox成员变量

### SEditbleTexBox可编辑文本框

获取文本时需要ToString()转换

if (texPath->GetText().ToString() == "Default")

#### .cpp实现

SAssignNew(texPath, SEditableTextBox).Text(FText::*FromString*(\*pppp)).MinDesiredWidth(150)  
.BackgroundColor(FLinearColor(0.98f,0.87f,0.65f,0.65f))  
.OnTextChanged(this, &SSlateMain::OnPathText)

.OnTextChanged()文本框修改事件函数,(const FText& Text)传入文本内容

void SSlateMain::OnPathText(const FText& Text)  
**{** if(FPaths::*DirectoryExists*(\*(FPaths::*ProjectContentDir*() + Text.ToString())))  
 CFN.WriteIni(FString("SceneTools"), FString("Path"), Text.ToString(), IniPath);  
 else  
 GEngine->AddOnScreenDebugMessage(-1, 6.f, FColor::*Red*, FString::*Printf*(TEXT(" %s 目录不存在!"), \*Text.ToString()));  
**}**

#### .h定义变量及函数

#include "Widgets/Input/SEditableTextBox.h"

class SSlateMain :public SCompoundWidget {

public:  
 void OnPathText(const FText& Text);

TSharedPtr<class SEditableTextBox> texPath;

}

## 参考资源链接

#### [UE4 Slate 编程入门一之Slate控件基本的创建](https://blog.csdn.net/qq_29523119/article/details/98475938)

## 批量处理贴图資源

批量修改UE引擎中指定目錄的法線圖壓縮方式（CompressionSettings）

### 获取目录资源,修改参数

获取指定类型文件，搜索子文件夾([参考链接](https://www.myredstone.top/archives/1115))

FReply SSlateMain::OnGoButtonClicked() {

TArray<FAssetData> AssetDatas;//聲明數組，用於存放收集的資源

FAssetRegistryModule& AssetRegistryModule = FModuleManager::LoadModuleChecked<FAssetRegistryModule>("AssetRegistry");

FARFilter Filter; //創建過濾屬性設置

Filter.PackagePaths.Add(\*FString::Printf(TEXT("/Game/%s"), \*(texPath->GetText().ToString())));//需要FName類型字符 \*FString 前面加星號解引出來使用

Filter.ClassNames.Add(UTexture::StaticClass()->GetFName());//資源類型名

if (RPath->IsChecked())

{ //遞歸搜尋路徑，選項勾選查找子目錄

Filter.bRecursivePaths = true;

}else{

Filter.bRecursivePaths = false;

}

Filter.bRecursiveClasses = true;//必須為true才可以獲取到文件

AssetRegistryModule.Get().GetAssets(Filter, AssetDatas);//傳入過濾，存放資源的數組

FString mess = FString::Printf(TEXT("共搜索 %i 个贴图文件"), AssetDatas.Num());

GEngine->AddOnScreenDebugMessage(-1, 10.f, FColor::Green, \*mess);

UE\_LOG(LogTemp, Log, TEXT("共搜索 %i 个贴图文件"), AssetDatas.Num());

int cou;

cou = 0;

for (const FAssetData& AssetData : AssetDatas)

{

UTexture\* MatObj = Cast<UTexture>(StaticLoadObject(UTexture::StaticClass(), nullptr, (TEXT("Texture'%s'"), \*AssetData.ObjectPath.ToString())));

//GEngine->AddOnScreenDebugMessage(-1, 10.f, FColor::White, FString::FromInt(MatObj->CompressionSettings.GetValue()));

if (MatObj->CompressionSettings.GetValue() == 1) {

MatObj->CompressionSettings = TC\_Default;//進入屬性查看參數類型（Alt+G進入TC\_Default聲明，查看其他枚舉）

//對修改的資源標記未保存狀態

MatObj->AddToRoot();

MatObj->UpdateResource();

MatObj->MarkPackageDirty();

cou += 1;

UE\_LOG(LogTemp, Log, TEXT("%s 【已处理】"), \*AssetData.AssetName.ToString());

}

}

if (cou > 0) {

UE\_LOG(LogTemp, Log, TEXT("共处理 %i 个法线贴图文件"), cou);

GEngine->AddOnScreenDebugMessage(-1, 10.f, FColor::Green, FString::Printf(TEXT("共处理 %i 个法线贴图文件"), cou));

}

return FReply::Handled();

}

### 加载资源 StaticMesh

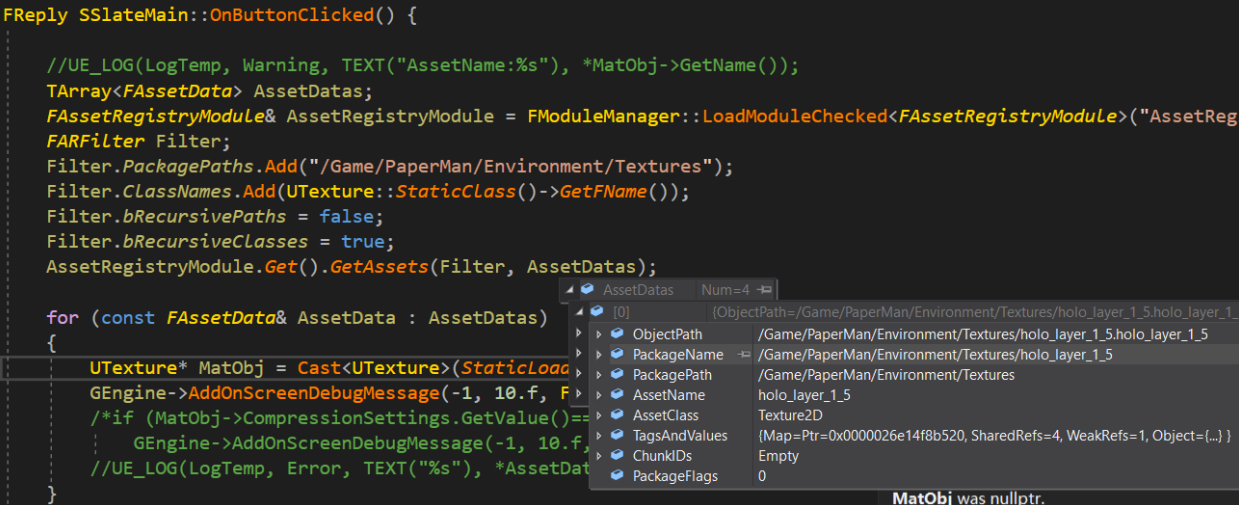
UStaticMesh\* mesh = LoadObject<UStaticMesh>(nullptr,TEXT("/Game/StarterContent/Props/SM\_Chair.SM\_Chair"));

if(mesh)

{

}

指定路径中获取 Texture 类资源放入 AssetDatas 数组，其中包含的属性与值如下图：



## 获取资源管理器(Asset)中贴图尺寸GitImportedSize()

需要用UTexture2D才能获取到贴图的导入尺寸，包括LOD Bias等参数

UTexture2D\* MatObj = Cast<UTexture2D>(StaticLoadObject(UTexture2D::*StaticClass*(), nullptr, (TEXT("Texture'%s'"), \*AssetData.ObjectPath.ToString())));

if ( MatObj->GetImportedSize().Y >= 1024) {  
 MatObj->MaxTextureSize = 512;  
 //標記未保存星號  
 MatObj->AddToRoot();  
 MatObj->UpdateResource();  
 MatObj->MarkPackageDirty();  
 cou += 1;  
 //文件路径写入文本  
 FString MatPath = \*MatObj->GetPathName();  
 FString MatLeft;  
 MatPath.Split(".", &MatLeft, nullptr);  
 Content += FString::*Printf*(TEXT("%s\n"), \*MatLeft);   
 }  
**}**

## 获取目录下的所有资源

### 硬获取

UObjectLibrary\* ObjectLibrary = UObjectLibrary::CreateLibrary(UUserDefinedStruct::StaticClass(), false, GIsEditor);

ObjectLibrary->LoadAssetDataFromPath(TEXT("/Game"));

TArray<FAssetData> AssetDatas;

ObjectLibrary->GetAssetDataList(AssetDatas);

for (const FAssetData& AssetData : AssetDatas)

{

UE\_LOG(LogTemp, Error, TEXT("%s"), \*AssetData.GetFullName());

}

### 软获取

TArray<FAssetData> AssetDatas;

FAssetRegistryModule& AssetRegistryModule = FModuleManager::GetModuleChecked<FAssetRegistryModule>("AssetRegistry");

FARFilter Filter;

Filter.PackagePaths.Add("/Game");

Filter.ClassNames.Add(UUserDefinedStruct::StaticClass()->GetFName());

Filter.bRecursivePaths = true;

Filter.bRecursiveClasses = true;

AssetRegistryModule.Get().GetAssets(Filter, AssetDatas);

for (const FAssetData& AssetData : AssetDatas)

{

UE\_LOG(LogTemp, Error, TEXT("%s"), \*AssetData.GetFullName());

}

## 获取项目路径-FPaths

获取插件文件夹,返回结尾有带斜杠

\*(FPaths::ProjectPluginsDir() + "SceneTools\_W\_P/setings.ini")

FPaths::ProjectContentDir() //可以拿到Content目录的绝对路径

# 方法集

## 开次方函数pow

//对2进行开方，后面为开方次数

int vvv = pow(2,setMapSize->GetValue());

//转换整数为FText字符,运用到滑动数值框SSpinBox中  
SizeValue->SetValue (FText::FromString(FString::FromInt(vvv)));

## RW读写文本（FFileHelper）

### RW-1读写函数

示例1

//.h 聲明在公共區

public:

FString ReadTxt(FString filepath);

bool WriteTxt(FString savestring, FString filepath);

========================================================================

//.cpp 讀取

FString SSlateMain::ReadTxt(FString filepath)

{

FString readString;

//路徑ProjectPluginsDir() = "../../../PM/Plugins/"

FFileHelper::LoadFileToString(readString, \*(FPaths::ProjectPluginsDir() + "SceneTools\_W\_P/" + filepath));

return readString;

}

//寫入，如文件不存在會創建

bool SSlateMain::WriteTxt(FString savestring, FString filepath)

{

bool result;

result = FFileHelper::SaveStringToFile(savestring, \*(FPaths::ProjectPluginsDir() + "SceneTools\_W\_P/" + filepath));

return true;

}

示例2

//读取文件函数  
FString SSlateMain::ReadTxt(FString PN)  
{  
 FString readString;  
 //路徑ProjectPluginsDir() = "../../../PM/Plugins/"  
 FFileHelper::*LoadFileToString*(readString, \*(FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/" + PN));  
 return readString;  
}  
//写入文件函数  
bool SSlateMain::WriteTxt(FString savestring, FString PN)  
**{** bool result;  
 result = FFileHelper::*SaveStringToFile*(savestring, \*(FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/" + PN));  
 return true;   
**}**

示例2-读取函数运用

FString aaa = **ReadTxt**("ST.txt");  
TArray<FString> arr;  
aaa.ParseIntoArray(arr, TEXT("\n"), false); //以换行符劈开文本字符

示例2-写入函数运用

bool SSlateMain::DiffPathText(FString path)  
**{** int con = 0;  
 FString aaa = ReadTxt("ST.txt"); //先读取记录作为重复查找使用  
 TArray<FString> arr;  
 aaa.ParseIntoArray(arr, TEXT("\n"), false); //将读取的文本以换行符拆分到数组arr中  
 for (const FString hist : arr)  
 {  
 if (hist == path) //查找到相同的 con+1  
 con += 1;  
 }  
 if (con == 0) //如果没有找到重复开始写入新内容  
 {  
 aaa += (path + "\n"); //在原来的内容上新增新记录  
 **WriteTxt**(aaa, "ST.txt"); //运用写入函数写入更新内容  
 return true;  
 } else {  
 return false;  
 }  
**}**

#### RW-1-1使用std打开文件

using namespace std;

//定义std可执行std字符串变量,转换FString变量加入std字符串

std::string const& test = std::string("start ") + std::string(TCHAR\_TO\_UTF8(\*FileName));

//调用执行

system(test.c\_str());

#### RW-1-2调用system打开文件

通过 system("start D:\\filename.txt") 方法打开文件

//获取文件路径FString变量,使用指针

FString FP = FString::Printf(TEXT("start %s"), \*FileName);

system(TCHAR\_TO\_UTF8(\*FP)); //字符转为UTF8

#### RW-1-3读取多行文档

分割每一行存入數組，將讀取的多行字符串以換行符分割

TArray<FString> arr;

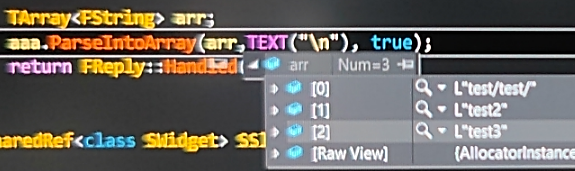
aaa.ParseIntoArray(arr,TEXT("\n"), false);//傳入（收集成員的數組，分割字符，是否包含空字符）

for (const FString line : arr) {

if (hist == path)

con += 1;

}



#### RW-1-4读取 DataTable

// .h

public:

FString GetUIName(int Index);

// .cpp

FString SSlateMain::GetUIName(int Index) {

UDataTable\* UIDataTable = LoadObject<UDataTable>(NULL, \*ST\_UITEXTFile);

TArray<FName> UIRow = UIDataTable->GetRowNames();

FChinaUIData\* uiRow = UIDataTable->FindRow<FChinaUIData>(UIRow[Index], ST\_ContextString);

FString UIT = \*uiRow->Info;

return UIT;

}

//调用

GetUIName(0);

## 打印文本到3D视窗

GEngine->AddOnScreenDebugMessage(-1, 6.f, FColor::Green, FString::Printf(TEXT("共处理 %i 个贴图文件"), cou));

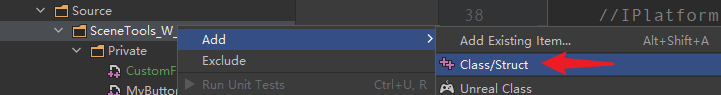
传入变量记得变量前有时需要加指针\*,如下在缺少指针时编译出错

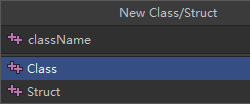
void SSlateMain::OnPathText(const FText& **Text**)  
**{** if(FPaths::*DirectoryExists*(\*(FPaths::*ProjectContentDir*() + Text.ToString())))  
 CFN.WriteIni(FString("SceneTools"), FString("Path"), **Text**.ToString(), IniPath);  
 else  
 GEngine->**AddOnScreenDebugMessage**(-1, 6.f, FColor::*Red*, FString::*Printf*(TEXT("目录 %s 不存在!"), \***Text**.ToString()));  
**}**

# 其他知识

## 自定义类函数及调用

创建类,cpp与h文件父文件夹右键添加 类/结构体 菜单



选择类,输入类名称

### .h头文件

#pragma once  
  
class **CustomFN**  
{  
public: //可在外部文件中调用  
 FString ReadIni(FString PN); //定义函数**★**  
};

### .cpp中实现

#include "CustomFN.h"  
#include "Misc/FileHelper.h"  
  
FString **CustomFN**::ReadIni(FString PN)  
**{** FString readString;   
 FFileHelper::*LoadFileToString*(readString, \*(FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/" + PN));  
 return readString;  
**}**

### 其他文件中调用

#include "CustomFN.h" //包含头文件**★**

void SSlateMain::Construct(const FArguments& InArgs)  
{  
 **CustomFN** CFN; //定义自定类**★**  
  
 if (FPaths::*FileExists*(\*(FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/setings.ini")))  
 **{** CFN.ReadIni("settings.ini"); //再用定义的类变量获取函数**★**  
 **}**

## UEC++读写INI文件

[参考网址](https://blog.csdn.net/weixin_46840974/article/details/126182948)

### .h创建类

#pragma once  
  
class CustomFN  
{  
public:  
 void ReadIniValue(FString Section, FString inKey, FString& outValue, FString IniFile);  
 bool WriteIni(FString newSection, FString newKey, FString newValue, FString IniFile);  
};

### .cpp写实现

#include "CustomFN.h"  
#include "Misc/FileHelper.h"  
  
void CustomFN::ReadIniValue(FString Section, FString inKey, FString& outValue, FString IniFile)  
{  
 if (!GConfig)  
 return;  
 GConfig->GetString(\*Section, \*inKey, outValue, IniFile);  
}  
bool CustomFN::WriteIni(FString newSection, FString newKey, FString newValue, FString IniFile)  
**{** if(!GConfig)  
 return false;  
 GConfig->SetString(\*newSection, \*newKey, \*newValue, IniFile);  
 GConfig->Flush(true); //即时刷新写入文件  
 return true;  
**}**

### 其他文件中调用函数

ini文件路径可以统一使用全局变量

FString IniPath = (FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/settings.ini");

创建一个继承自定义类的全局类名CFN来调用里面的函数

CustomFN CFN;

写入ini

CFN.WriteIni(FString("SceneTools"), FString("Path"), \*(texPath->GetText().ToString()), \*IniPath);

读取ini到变量运用

FString **rval**;

CFN.ReadIniValue(FString("SecneTools"), FString("Path"), **rval**, \*IniPath);

suffixSeach->SetText(FText::*FromString*(**rval**));

### ★INI记录读写示例:

定义全局变量接收读取值,用到UI的初始值设置

FString s\_equal="";  
int s\_searchV=1024;

定义INI文件变量,读写函数中传入

FString IniPath☆ = (FPaths::ProjectPluginsDir() + "SceneTools\_W\_P/settings.ini");

记录读取给全局变量赋值,整数类型需要通过FCString::Atoi将整数转换为FString

if (FPaths::*FileExists*(\*(FPaths::*ProjectPluginsDir*() + "SceneTools\_W\_P/settings.ini")))  
**{**  
 ★CFN.ReadIniValue(FString("SceneTools"), FString("Equal"), s\_equal★, \*IniPath☆);   
 FString s\_SV;  
 ★CFN.ReadIniValue(FString("SceneTools"), FString("SearchV"), s\_SV, \*IniPath);  
 s\_searchV★ = FCString::*Atoi*(\*s\_SV);   
**}**

Slate控件赋值

字符串变量需要加\*指针

SAssignNew(equalText,SEditableTextBox).Text(FText::*FromString*(\*s\_**equal★**)).MinDesiredWidth**(**10**)**

整数变量不用加\*指针

SAssignNew(searchSize, SSpinBox<int>).MaxValue(4096).MinValue(32).Value(s\_**searchV★**).ToolTipText(FText::FromString(TEXT("拖动")))

.OnEndSliderMovement(this,&SSlateMain::SearchSizeValueSet)

控件事件函数写入INI

void SSlateMain::SearchSizeValueSet(FPlatformTypes::int32 val)  
{  
 for (int i=5; i <= 11;i++)  
 **{** int sss =pow(2,i);  
 if (val >= sss && val < pow(2,(i+1)))  
 {  
 searchSize->SetValue(sss);  
 ★CFN.WriteIni(FString("SceneTools"), FString("SearchV"), FString::*FromInt*(sss), IniPath☆);  
 }  
 **}**

}

## 元编程传递

//在头文件中的 public: 下的BEGIN与END之间

SLATE\_BEGIN\_ARGS(SSlateMain) {}//一个不完全的结构体

float \_A;//声明变量

WidgetArgsType& Hello(float InArg)

{

\_A = InArg;

return this->Me();

}

SLATE\_END\_ARGS()//带};的结构体，是上面的Begin的结尾

//在cpp中调用\_A

void SSlateMain::Construct(const FArguments& InArgs)

{

InArgs.\_A;

}

## 循环

### 数组内循环

FString newP = "";

TArray<FString> SArr;   
for (const FString sss : SArr)  
**{** newP += sss + "/";  
**}**

### switch(exp){case val:}

一个switch语句允许测试一个变量等于多个值时的情况。每个值称为一个case，且被测试的变量会对每个 switch case进行检查。

语法格式：

switch(expression){

case 常量表达式1:

语句1;

...

语句n;

break; // 可选

case 常量表达式2:

语句1;

...

语句n;

break; // 可选

...

case 常量表达式n:

语句1;

...

语句n;

break; // 可选

default: // 可选

语句1;

...

语句n;

break; // 可选

}

## 拆分字符串到数组(ParseIntoArray)

FString aaa = ReadTxt("ST.txt"); //读取多行文本  
TArray<FString> arr;  
aaa.ParseIntoArray(arr, TEXT("\n"), false); //将读取的文本以换行符为拆分条件,分割字符串到数组arr中

## 强制转换

a = (int \*)b;

例：

*#include <stdio.h>*

int main (){

int a= 10;

void \*b = &a;

printf("int a = %d\n",a);

printf("void (int \*)b =%d \n",\*(int \*)b);

printf("void (double \*)b =%d \n",\*(double\*)b); //编译器并不会报错但是其结果却有点出人意料

return 0;

}

返回：

int a = 10  
void (int \*)b =10  
void (double \*)b =0