一、蓝图函数的调用过程

默认的就是由UHT来生成Thunk调用。

Thunk：传值调用：函数被调用前参数的值已经被编译器计算好了，

        传名调用：只有当函数真正被调用时才计算参数的值（惰性求值）

        在此过程中编译器其实已经把惰性求值的过程包装成了一个Thunk辅助函数，先调这个辅助函数求出参数值，再进入函数主题。

.generated.h文件中有诸如DECLARE\_FUNCTION(execFUNCTION)这种代码

最终的函数签名为

void(\*)(UObject\*Context, FFrame& Stack,void\*const Z\_Param\_\_Result)

.gen.cpp文件中有相关的定义

DEFINE\_FUNCTION(UNavBar::execOnClickCallback)

{

    P\_GET\_PROPERTY(FStrProperty,Z\_Param\_key);

    P\_FINISH;

    P\_NATIVE\_BEGIN;

    P\_THIS->OnClickCallback(Z\_Param\_key);

    P\_NATIVE\_END;

}

通过P\_GET\_XXX宏从Stack中提取C++类型对象，最终再调用C++函数本身的实现

※关于虚拟机

程序虚拟机（如Java JVM）为运行单个计算机程序设计，这意謂它支持单个进程。虚拟机的一个本质特点是运行在虚拟机上的软件被局限在虚拟机提供的资源里——它不能超出虚拟世界。

虚拟机实现分为基于寄存器的虚拟机和基于栈的虚拟机。

二、蓝图虚拟机的实现

1，字节码：即指令

蓝图虚拟机的字节码在Script.h文件中，因为是专用的脚本语言，所以有一些特殊的字节码 ，如代理相关的代码（EX\_BindDelegate、EX\_AddMulticastDelegate）

2，栈帧

Stack.h中能找到FFrame的定义

// Variables

UFunction\* Node; //当前执行的脚本函数

UObject\* Object; //执行该脚本的UObject

uint8\* Code; //当前代码的执行位置

uint8\* Locals; //局部变量

UProperty\* MostRecentProperty; // 最近一次正在使用的uproperty

uint8\* MostRecentPropertyAddress; // 最近一次正在使用的uproperty地址

FlowStackType FlowStack;

FFrame\* PreviousFrame;//栈中的上一帧

FOutParmRec\* OutParms; //OutParam的信息

UField\* PropertyChainForCompiledIn;

UFunction\* CurrentNativeFunction; //当前的原生函数

bool bArrayContextFailed;

void FFrame::Step(UObject \*Context, RESULT\_DECL)

{

    int32 B = \*Code++;

    (Context->\*GNatives[B])(\*this,RESULT\_PARAM);

}

Step函数主要作用就是取出指令，然后在原生函数数组中找到对应的函数去执行

3，字节码对应函数

前面谈到的字节码都会有一个具体执行部分的代码，在ScriptCore.cpp中能找到定义

这里可以看到每个字节码对应的原生函数都在GNatives和GCasts里面

IMG_256IMG_257

这个宏会将原生函数放到GNatives数组中，定义了一个全局静态对象，这样就会在main函数执行前把函数放在数组中的位置。

4，执行流程

①C++调用蓝图函数（还需要详解）

ProcessEvent(UFunction\* Function, void\* Parms)

如果Function是FUNC\_Native，就是C++函数调用，如果不是FUNC\_Native，就是蓝图调用过程， 在ProcessScriptFunction中，先开辟一个新的栈帧NewStack，申请临时变量空间，函数返回值空间，获取函数参数，并记录在NewStack上

核心逻辑：

Step:

1，找到字节码，

2，执行相应函数execFinalFunction

3，执行CallFunction()，并且将FFrame传入

4，触发Function->InVoke()

5，执行原生函数

P\_GET\_OBJECT(ObjectType,ParamName) 取参数

P\_FINISH 取参数结束

P\_NATIVE\_BEGIN 原生函数调用开始

P\_THIS->XXX(ParamName) 传入参数并调用原生函数

P\_NATIVE\_END 原生函数调用结束

