XCOM EU/EW UPK file format

Disclaimer:

All trademarks mentioned herein belong to their respective owners.

All information on package format was found by people like Gildor, Antonio Cordero Balcazar, Eliot van Uytfanghe and the others. I just gathered it and verified for XCOM EU/EW packages in hope that it will help modders in better game understanding and creation of new modding tools.

This document can be considered a beta version. I'm still working on improving it and adding missing information.

04/14/2015

Wasteland Ghost aka wghost81 (wghost81@gmail.com).

Contents

Links and Acknowledgments

XCOM Mod Talk

XCOM Wiki on Nexus Wiki

Gildor's Forums, Links and Tools

UE Explorer

Unreal Developer Network

Unreal Wiki

Unreal Tournament Package Delphi Library

UPK Info

Unreal Package file format by Epic Games

Unreal Tournament Public Source Code

On Reflection and Serialization

Unreal Packages

Links

Unreal Packages

Content Cooking

Content Streaming

General information

Unreal Engine packages (UPK files) are used to store various game assets, like scripts, textures and sounds.

Unreal packages are modeled closely after DLLs for their dependencies. A package always contains the following in this order:

- package file summary
- name table
- import table
- export table

Package file summary contains some basic information and offset/size of various tables stored in the package. Name table contains all the serialized names. Export table contains all the objects serialized into the package, while import table contains all external dependencies.

Each export entry inside the export table contains an offset into the file where its data is located. Export objects are saved into the package in an order that allows them to be created/loaded in a linear (seekfree) fashion. This means that an object's class or outer is always sorted before the object itself.

Unreal Engine serialization basics

Serialization is the ability to write objects to and read objects from byte streams. In game terms, serialization encompasses export/import of static state, saving/loading of dynamic state, and transmitting/receiving of state information over a network.

All Unreal objects inherit from base serializable UObject class. UObject and all its child classes are able to serialize themselves. While serializing, parent object data are stored first, then child object data follows.

For example, UState inherits from UStrust, which inherits from UField, which inherits from UObject. So, State objects are serialized in this order: UObject → UField → UStrust → UState.

Header

Package header consist of summary, array of compressed chunks, Name, Import and Export tables and Depends Table.

FPackageFileSummary

Unreal package summary, stored at the top of the file.

Size	Type	Name	Notes
4 bytes	Signature	Signature	Package signature: 0x9E2A83C1
4 bytes	int32	Version/LicenseeVersion	int16 Version: lower 2 bytes int16 LicenseeVersion: higher 2 bytes XCOM packages Version = 845
4 bytes	int32	HeaderSize	Size of the header, including Name, Export and Import Tables and Depends Table
variable	FString	PackageGroup	"None" for XCOM packages
4 bytes	int32	PackageFlags	
4 bytes	int32	NameCount	
4 bytes	int32	NameOffset	
4 bytes	int32	ExportCount	
4 bytes	int32	ExportOffset	
4 bytes	int32	ImportCount	
4 bytes	int32	ImportOffset	
4 bytes	int32	DependsOffset	
4 bytes	int32	SerializedOffset	Equal to HeaderSize
4 bytes	int32	Unknown2	
4 bytes	int32	Unknown3	
4 bytes	int32	Unknown4	
16 bytes	FGuid	GUID	Package Global Unique Identifier
variable	TArray<FGenerationInfo>	Generations	GenerationsCount entries of FGenerationInfo type

4 bytes	int32	EngineVersion	
4 bytes	int32	CookerVersion	
4 bytes	int32	CompressionFlags	
variable	TArray<FCompressedChunk>	CompressedChunks	NumCompressedChunks entries of FCompressedChunk type

Name Table

Contains NameCount elements of FNameEntry type.

FNameEntry

A global name.

Size	**Type**	**Name**	**Notes**
variable	FString	Name	
8 bytes	int64	NameFlags	

Import Table

Contains ImportCount elements of FObjectImport type.

FObjectImport

Import object data.

Size	**Type**	**Name**	**Notes**
8 bytes	UNameIndex	PackageIdx	Name of the package this object resides in
8 bytes	UNameIndex	TypeIdx	This object type
4 bytes	UObjectReference	OwnerRef	Owner reference
8 bytes	UNameIndex	NameIdx	This object name

Export Table

Contains ExportCount elements of FObjectExport type.

FObjectExport

Export object data.

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	TypeRef	This object type (reference to inner or outer class)
4 bytes	UObjectReference	ParentClassRef	Parent class reference
4 bytes	UObjectReference	OwnerRef	Owner reference
8 bytes	UNameIndex	NameIdx	This object name
4 bytes	UObjectReference	ArchetypeRef	Archetype object reference

Size	Type	Name	Notes
4 bytes	int32	ObjectFlagsH	
4 bytes	int32	ObjectFlagsL	
4 bytes	int32	SerialSize	Serialized data size
4 bytes	int32	SerialOffset	Serialized data file offset
4 bytes	int32	ExportFlags	
4 bytes	int32	NetObjectCount	
16 bytes	FGuid	GUID	Net related: zero if NetObjectCount == 0, non-zero if NetObjectCount > 0
4 bytes	int32	Unknown	
variable	Unknown	Unknown	4 x NetObjectCount bytes of data

Depends Table

Starts with DependsOffset (right after ExportTable), ends with HeaderSize offset.

Size	Type	Name	Notes
variable	unknown	Depends	(HeaderSize-DependsOffset) bytes of unknown data (zeros for uncompressed XCOM upk)

Core objects

Core objects inherit from UObject native base class.

Hierarchy

```
class   UObject;
class       UField;
class           UConst;
class           UEnum;
class           UProperty;
class               UByteProperty;
class               UIntProperty;
class               UBoolProperty;
class               UFloatProperty;
class               UObjectProperty;
class                   UClassProperty;
class                   UComponentProperty;
class               UNameProperty;
class               UStructProperty;
class               UStrProperty;
class                   UFixedArrayProperty;
class                   UArrayProperty;
class                   UMapProperty;
class                   UDelegateProperty;
class                   UInterfaceProperty;
class               UStruct;
class                   UScriptStruct;
class                   UFunction;
class                   UState;
class                       UClass;
class           UTextBuffer;
class           UComponent;
```

UObject

The base class of all objects.

For all object types:

Size	**Type**	**Name**	**Notes**
4 bytes	int32	NetIndex	Assigned by Linker, used for Net Replication

If HasStack flag is set only:

Size	**Type**	**Name**	**Notes**
22 bytes	Unknown	Unknown	Unknown data

For non-'Class' objects only:

Size	**Type**	**Name**	**Notes**
variable	UDefaultPropertyList	DefaultProperties	List of default values for object's variables

UField

Base class of UnrealScript objects.

For all object types:

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	NextRef	Next field

For 'Struct' objects only:

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	ParentRef	Super (parent) field reference

UStruct

An UnrealScript structure.

Size	Type	Name	Notes
variable	UObjectReference	ScriptTextRef	Source text?
4 bytes	UObjectReference	FirstChildRef	Reference to first child object
variable	UObjectReference	CppTextRef	Source text?
4 bytes	int32	Line	Compiler info
4 bytes	int32	TextPos	Compiler info
4 bytes	int32	ScriptMemorySize	Script memory size
4 bytes	int32	ScriptSerialSize	Script serialized data size
variable	ByteCode	DataScript	ScriptSerialSize bytes

UStruct Children

UStruct children objects are chained together through ChildrenRef.NextRef. Here's an example pseudo-code to iterate through all children:

```
nextChild = FirstChild;
while (nextChild != 0)
    nextChild = nextChild.NextRef;
```

All function parameters, locals and return value are function's children. And all functions are children of corresponding class.

Child objects are stored in Export Table in reverse order: ChildN, …, Child2, Child1, Owner. So next child index is always lesser than previous child index. But it is possible to break default order and link any child object to UStruct.

UScriptStruct

Size	Type	Name	Notes
4 bytes	int32	StructFlags	
variable	UDefaultPropertyList	StructDefaultProperties	List of default values for structure variables

UFunction

An UnrealScript function.

Size	Type	Name	Notes
2 bytes	int16	NativeToken	
1 byte	int8	OperPrecedence	Pre or post operator
4 bytes	int32	FunctionFlags	
2 bytes	int16	RepOffset	For Net functions only (if FunctionFlags.Net is set), omitted for the rest.
8 bytes	UNameIndex	NameIdx	Function name

UState

An UnrealScript state.

Size	Type	Name	Notes
4 bytes	int32	ProbeMask	
2 bytes	int16	LabelTableOffset	Memory offset of LabelTable inside DataScript
2 bytes	int32	StateFlags	
variable	TMap<UNameIndex, UObjectReference>	StateMap	<State name, State object reference> 4 bytes of StateMapSize + 12 x MapSize bytes

LabelTable

LabelTable is stored as a part of a script and located at the end of DataScript. LabelTable begins with 0x0C token. Last LabelTable entry contains NameTableIdx to "None" with LabelMemoryPos = 0x0000FFFF.

FLabelEntry

An entry in LabelTable.

Size	Type	Name	Notes
8 bytes	UNameIndex	NameTableIdx	Label name
4 bytes	int32	LabelMemoryPos	Memory label position inside DataScript

UClass

An object class.

Size	Type	Name	Notes
4 bytes	int32	ClassFlags	
4 bytes	UObjectReference	WithinRef	
8 bytes	UNameIndex	ConfigNameIdx	
variable	TMap<UNameIndex, int32>	ComponentsMap	NumComponents elements of <ComponentNameIdx, Unknown> pairs, 4 + 12 x NumComponents bytes total
variable	TMap <UObjectReference, int32>	Interfaces	NumInterfaces elements of <InterfaceObj, Unknown> pairs, 4 + 8 x NumInterfaces bytes total
variable	TArray<UNameIndex>	DontSortCategories	Editor related
variable	TArray<UNameIndex>	HideCategories	Editor related
variable	TArray<UNameIndex>	AutoExpandCategories	Editor related
variable	TArray<UNameIndex>	AutoCollapseCategories	Editor related

Size	Type	Name	Notes
4 bytes	int32	ForceScriptOrder	Editor related
variable	TArray<UNameIndex>	ClassGroups	Editor related
variable	FString	NativeClassName	
8 bytes	UNameIndex	DLLBindName	
4 bytes	UObjectReference	DefaultRef	DefaultProperties object reference

UConst

An UnrealScript constant.

Size	Type	Name	Notes
variable	FString	Value	

UEnum

An enumeration, a list of names usable by UnrealScript.

Size	Type	Name	Notes
variable	TArray<UNameIndex>	Names	Enum names

UComponent

Structure unknown.

UProperty

An UnrealScript variable.

Size	Type	Name	Notes
4 bytes	int32	ArraySize	lower 2 bytes: ArrayDim higher 2 bytes: ElementSize
8 bytes	int64	PropertyFlags	
8 bytes	UNameIndex	CategoryIndex	
4 bytes	UObjectReference	ArrayEnumRef	
2 bytes	int16	RepOffset	For Net properties only (if PropertyFlags.Net is set), omitted for the rest!

UByteProperty

An unsigned byte value or 255-value enumeration variable.

Size	Type	Name	Notes
4 bytes	UObjectReference	EnumObjRef	Enum type

UIntProperty

A 32-bit signed integer variable. Does not have type-specific persistent members.

UBoolProperty

A single bit flag variable residing in a 32-bit unsigned double word. Does not have type-specific persistent members.

UFloatProperty

An IEEE 32-bit floating point variable. Does not have type-specific persistent members.

UObjectProperty

A reference variable to another object which may be nil.

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	OtherObjRef	Object type

UClassProperty

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	ClassObjRef	Object class

UComponentProperty

Does not have type-specific persistent members.

UNameProperty

A name variable pointing into the global name table. Does not have type-specific persistent members.

UStructProperty

A structure variable embedded in (as opposed to referenced by) an object.

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	StructObjRef	Struct type

UStrProperty

A dynamic string variable. Does not have type-specific persistent members.

UFixedArrayProperty

A fixed length array.

Size	**Type**	**Name**	**Notes**
4 bytes	UObjectReference	InnerObjRef	Inner object type

Size	Type	Name	Notes
4 bytes	int32	Count	

It seems, that there are no FixedArrayProperties in XCOM. Fixed arrays are defined via ArrayDim field of UProperty object. So, array of 15 integers, for example, will have type "IntProperty" in Export Table and ArrayDim = 15 in UProperty serialized data.

UArrayProperty

A dynamic array.

Size	Type	Name	Notes
4 bytes	UObjectReference	InnerObjRef	Inner object type

UMapProperty

A dynamic map.

Size	Type	Name	Notes
4 bytes	UObjectReference	KeyObjRef	
4 bytes	UObjectReference	ValueObjRef	

UDelegateProperty

Size	Type	Name	Notes
4 bytes	UObjectReference	FunctionObjRef	
4 bytes	UObjectReference	DelegateObjRef	

UInterfaceProperty

Size	Type	Name	Notes
4 bytes	UObjectReference	InterfaceObjRef	

UTextBuffer

An object that holds a bunch of text. The text is contiguous and, if of nonzero length, is terminated by a NULL at the very last position.

Size	Type	Name	Notes
4 bytes	int32	Top	
4 bytes	int32	Pos	
variable	FString	Text	

Core types

UObjectReference

Size	Type	Name	Notes

4 bytes	int32	ObjRef	

Zero value represents null-object. Positive value indicates an index to Export Table, negative— an index to Import Table.

UNameIndex

Size	Type	Name	Notes
4 bytes	int32	NameTableIdx	Index to Name Table
4 bytes	int32	Numeric	

If Numeric is zero, name string is equal to Name Table name, extracted by NameTableIdx. Otherwise, numeric is added to the name string, equal to Numeric-1. Example: “SomeObjectName_0”, “SomeObjectName_1”, …

FGenerationInfo

Size	Type	Name	Notes
4 bytes	int32	ExportCount	
4 bytes	int32	NameCount	
4 bytes	int32	NetObjectCount	

FCompressedChunk

Size	Type	Name	Notes
4 bytes	int32	UncompressedOffset	
4 bytes	int32	UncompressedSize	
4 bytes	int32	CompressedOffset	
4 bytes	int32	CompressedSize	

FGuid

Globally unique identifier. sprintf format: "%08X%08X%08X%08X".

Size	Type	Name	Notes
4 bytes	uint32	GUID_A	
4 bytes	uint32	GUID_B	
4 bytes	uint32	GUID_C	
4 bytes	uint32	GUID_D	

FString

Size	Type	Name	Notes
4 bytes	int32	StringLength	Length of a string
variable	variable	String	ASCII or Unicode string

If StringLength > 0, String is an ASCII null-terminated string. If StringLength < 0, String is an

Unicode string.

Core templates

TArray

template<class T> class TArray;

Templated dynamic array.

Size	Type	Name	Notes
4 bytes	int32	ArrSize	Array size
T dependent	T	TArrElement	sizeof(T) x ArrSize bytes of data

TMap

template<class TK, class TI> class TMap;

Maps unique keys to values. TMap data are stored in TArray<TPair>. TMap is serialized as TArray of type T=TPair.

TPair

Size	Type	Name	Notes
TK dependent	TK	Key	
TI dependent	TI	Value	

Default properties

Default properties are packed as list variable names and their identifiers. Default properties by themselves aren't objects, they are just text, written as a sequence of name table indexes followed by values.

UDefaultPropertyList

Default properties list contains a list of UDefaultProperty entries. List ends with UNameIndex to “None”.

UDefaultProperty

For all objects:

Size	Type	Name	Notes
8 bytes	UNameIndex	NameIdx	Property name
8 bytes	UNameIndex	TypeIdx	Property type name
4 bytes	int32	PropertySize	Property size in bytes
4 bytes	int32	ArrayIdx	For fixed arrays: array element index (counts from 1), for other types— zero.

For BoolProperty only (omitted for the rest):

Size	Type	Name	Notes
1 byte	int8	BoolValue	0 (false) or 1 (true)

For StructProperty only (omitted for the rest):

Size	Type	Name	Notes
8 bytes	UNameIndex	StructNameIdx	Structure name, defines the internal data type

For ByteProperty only (omitted for the rest):

Size	Type	Name	Notes
8 bytes	UNameIndex	EnumNameIdx	Enum object name, defines the type of internal enum variable

For all objects:

Size	Type	Name	Notes
Size bytes	variable	PropertyValue	Size bytes of property value, type of value is determined by property type and internal variable type (for structure properties)

PropertyValue is determined by property type: it can be int or float 4-byte value, UNameIndex or UObjectReference, etc. For arrays the first 4 bytes of PropertyValue determine the number of array elements. To properly deserialize array elements, one needs to determine elements type first. It can be done by searching for this array object in Export Table, deserializing it do determine Inner Object Reference and finally retrieving Inner Object Type by its reference. If array is native or imported, there will be no Export Table entry or serialized property data, so its type will remain unknown. But one may try to guess the type from an object's name: “SequenceObjects” array most probably holds “ObjectProperty” data, etc.

Property Value

Property value defines a default variable's value. Its size in bytes is determined by PropertySize field. For structure and array properties this value may be quite complex and is often contains an internal default properties list.

IntProperty

PropertySize = 4

Size	Type	Name	Notes
4 bytes	int32	Value	integer variable

FloatProperty

PropertySize = 4

Size	Type	Name	Notes
4 bytes	float	Value	floating-point variable

ObjectProperty, InterfaceProperty, ComponentProperty, ClassProperty

PropertySize = 4

Size	Type	Name	Notes
4 bytes	UObjectReference	Value	Object reference

Guid

PropertySize = 16

Size	Type	Name	Notes
16 bytes	FGuid	Value	GUID

NameProperty, ByteProperty

PropertySize = 8

Size	Type	Name	Notes
8 bytes	UNameIndex	Value	Name table index: name for NameProperty and Enum value name for ByteProperty

StrProperty

PropertySize— variable

Size	Type	Name	Notes
variable	FString	Value	Text

StructProperty

PropertySize— variable. Depends on StructNameIdx.

ArrayProperty

Dynamic array property. PropertySize— variable.

Size	Type	Name	Notes
4 bytes	int32	NumElements	Number of array elements
variable	variable	InternalData	NumElements x ElementSize bytes of data

ElementSize is determined by array's data type. To determine the type, one needs to find this array as an object inside an Export or Import Table. If the type equals to, say, IntProperty, then array will hold NumElements x 4 bytes of integers. In general, proper deserialization of default properties requires reconstruction of all import and export objects, which is a complex task.

Core structure properties

Vector, Rotator, Vector2D and the others, defined in Core package:

- Vector {float X, float Y, float Z}
- Vector2D {float X, float Y}
- Rotator {int Pitch, int Yaw, int Roll}
- Color { byte R, byte G, byte B, byte A }

- LinearColor { float R, float G, float B, float A }

Flags

PackageFlags

Flag	Name	Description
0x00000001	AllowDownload	Allow downloading package
0x00000002	ClientOptional	Purely optional for clients
0x00000004	ServerSideOnly	Only needed on the server side
0x00000008	BrokenLinks or Cooked	Loaded from linker with broken import links
0x00000010	Unsecure	Not trusted
0x00000020	Encrypted	
0x00008000	Need	Client needs to download this package
0x00020000	Map	
0x00200000	Script	
0x00400000	Debug	
0x00800000	Imports	
0x02000000	Compressed	
0x04000000	FullyCompressed	
0x20000000	NoExportsData	
0x40000000	Stripped	
0x80000000	Protected	

CompressionFlags

Flag	Name	Description
0x00000001	ZLIB	
0x00000002	LZO	
0x00000004	LZX	

NameFlags

Unknown

ObjectFlags

ObjectFlagsL:

Flag	Name	Description
0x00000001	Transactional	Object is transactional

0x00000002	Unreachable	Object is not reachable on the object graph
0x00000004	Public	Object is visible outside its package
0x00000080	Private	
0x00000100	Automated	
0x00004000	Transient	Don't save object
0x00008000	Preloading	Data is being preloaded from file
0x00010000	LoadForClient	In-file load for client
0x00020000	LoadForServer	In-file load for server
0x00040000	LoadForEdit	In-file load for editor
0x00080000	Standalone	Keep object around for editing even if unreferenced
0x00100000	NotForClient	Don't load this object for the game client
0x00200000	NotForServer	Don't load this object for the game server
0x00400000	NotForEdit	Don't load this object for the editor
0x01000000	NeedPostLoad	Object needs to be postloaded
0x02000000	HasStack	Has execution stack
0x04000000	Native	Native (UClass only?)
0x08000000	Marked	Marked (for debugging)

ObjectFlagsH:

Flag	**Name**	**Description**
0x00000020	Obsolete	
0x00000080	Final	
0x00000100	PerObjectLocalized	
0x00000200	PropertiesObject	
0x00000400	ArchetypeObject	
0x00000800	RemappedName	Name is remapped

ExportFlags

Flag	**Name**	**Description**
0x00000001	ForcedExport	

FunctionFlags

Flag	**Name**	**Description**
0x00000001	Final	Function is final (prebindable, non-overridable function)
0x00000002	Defined	Function has been defined (not just declared)
0x00000004	Iterator	Function is an iterator
0x00000008	Latent	Function is a latent state function

0x00000010	PreOperator	Unary operator is a prefix operator
0x00000020	Singular	Function cannot be reentered
0x00000040	Net	Function is network-replicated
0x00000080	NetReliable	Function should be sent reliably on the network
0x00000100	Simulated	Function executed on the client side
0x00000200	Exec	Executable from command line
0x00000400	Native	Native function
0x00000800	Event	Event function
0x00001000	Operator	Operator function
0x00002000	Static	Static function
0x00004000	NoExport or OptionalParameters	
0x00008000	Const	Function doesn't modify this object
0x00010000	Invariant	Return value is purely dependent on parameters; no state dependencies or internal state changes
0x00020000	Public	
0x00040000	Private	
0x00080000	Protected	
0x00100000	Delegate	
0x00200000	NetServer	
0x01000000	NetClient	
0x02000000	DLLImport	

StructFlags

Flag	Name	Description
0x00000001	Native	
0x00000002	Export	
0x00000004	Long or HasComponents	
0x00000008	Init or Transient	
0x00000010	Atomic	
0x00000020	Immutable	
0x00000040	StrictConfig	
0x00000080	ImmutableWhenCooked	
0x00000100	AtomicWhenCooked	

ClassFlags

Flag	Name	Description
0x00000001	Abstract	Class is abstract and can't be instantiated directly
0x00000002	Compiled	Script has been compiled successfully
0x00000004	Config	Load object configuration at construction time
0x00000008	Transient	This object type can't be saved; null it out at save time
0x00000010	Parsed	Successfully parsed
0x00000020	Localized	Class contains localized text
0x00000040	SafeReplace	Objects of this class can be safely replaced with default or NULL
0x00000080	RuntimeStatic	Objects of this class are static during gameplay
0x00000100	NoExport	Don't export to C++ header
0x00000200	Placeable	
0x00000400	PerObjectConfig	Handle object configuration on a per-object basis, rather than per-class
0x00000800	NativeReplication	Replication handled in C++
0x00001000	EditInlineNew	
0x00002000	CollapseCategories	
0x00004000	ExportStructs	
0x00200000	Instanced	
0x00400000	HideDropDown or HasComponents	
0x00800000	CacheExempt or Hidden	
0x01000000	ParseConfig or Deprecated	
0x02000000	HideDropDown2	
0x04000000	Exported	
0x20000000	NativeOnly	

StateFlags

Flag	Name	Description
0x00000001	Editable	State should be user-selectable in UnrealEd
0x00000002	Auto	State is automatic (the default state)
0x00000004	Simulated	State executes on client side

PropertyFlags

PropertyFlagsL:

Flag	**Name**	**Description**
0x00000001	Editable	
0x00000002	Const	
0x00000004	Input	
0x00000008	ExportObject	
0x00000010	OptionalParm	
0x00000020	Net	
0x00000040	EditConstArray or EditFixedSize	
0x00000080	Parm	
0x00000100	OutParm	
0x00000200	SkipParm	
0x00000400	ReturnParm	
0x00000800	CoerceParm	
0x00001000	Native	
0x00002000	Transient	
0x00004000	Config	
0x00008000	Localized	
0x00010000	Travel	
0x00020000	EditConst	
0x00040000	GlobalConfig	
0x00080000	Component	
0x00100000	OnDemand	
OnDemand	Init	
0x00200000	New or DuplicateTransient	
0x00400000	NeedCtorLink	
0x00800000	NoExport	
0x01000000	Cache or NoImport	
0x02000000	EditorData or NoClear	
0x04000000	EditInline	
0x08000000	EdFindable	
0x10000000	EditInlineUse	
0x20000000	Deprecated	
0x40000000	EditInlineNotify or DataBinding	

0x80000000	SerializeText or Automated	

PropertyFlagsH:

Flag	Name	Description
0x00000001	RepNotify	
0x00000002	Interp	
0x00000004	NonTransactional	
0x00000008	EditorOnly	
0x00000010	NotForConsole	
0x00000020	RepRetry	
0x00000040	PrivateWrite	
0x00000080	ProtectedWrite	
0x00000100	Archetype	
0x00000200	EditHide	
0x00000400	EditTextBox	
0x00001000	CrossLevelPassive	
0x00002000	CrossLevelActive	