PACKAGER 64

A ServerActor addressing XC_Engine
based servers for game UT'99
updated the predecessor with 32 games
into a 64 games one

Description: Like its predecessor this is a ServerActor which I did mainly for a multi-server, more exactly an UT'99 server running multiple game-types (custom recommended) and having as purpose less memory consumed with all sort of packages loaded because are needed by each of those game-types. There is no problem with game-types and their packages where they are resident because a game-type doesn't need declaration in ServerPackages - IT IS MAPPED automatically as long as Server will speak with player what is running and definitely player will be served with game. The problem comes with mutators or tools specific for each game-type which is not needed by the rest but... they are loaded with no purpose making por player to wait a lot of downloads. By using this tool, your server will need only default Skins or even Nothing mentioned if you put them in this <packager> task VIA ini file called ServerPackager.ini created at **first run** or... accessing **preferences** **ConsoleCommand** in your checker version of UT or working version not that used On-Line which is recommended to stay clean. Version for 64 games was requested for some Multi-Server having... whatever MANY gaming options... Whatever errors from these games crashing it, it's not my problem.

Operation: Default plain UT I'm not so convinced as having capability of setting up packages but with XC_Engine this is a matter of calling a simple function. If this toy is loaded in a plain server, a crash will show up immediately because this actor will trigger a function which defaut UT doesn't have anywhere, it's a new thing. As result, this Actor will work only in presence of XC_Engine - it's definitely a ServerActor.

Packager will scan ini file for 64 game-types possible configured and for each game-type can map 32 packages in particular. If you need more for that game (lol) you will need to drop them as permanent packages in default Server.ini or whatever ini used by your server. I think checking more arrays is not healthy for iterations operated at start which have a default load. So to speak sampling:

- DM using some new monsters are in packages only here;
- CTF using Team Type monsters are in other packages loaded only here;
- MonsterHunt with specific tools has its own packages.

With other words not all DM CTF MH is being loaded triggering new player to download one week everything for playing 10 minutes - they are leaving if are killed a few times I'm speaking from a few real experiences.

Player will have separate Packages to download and... if wants to stay and switch game-type VIA some MapVote new packages are triggered for the new voted map.

Configuration: This **Packager_64.U** file has to be dropped usually in System folder at once with INT if desired, or folder with paths to <**U**> files in your server if you have other deals...

Next stage is opening ini file used by server and completing it with a line resident around [XC_Engine.XC_GameEngine] section:

```
ServerActors=Packager_64.ServPackager
```

And this was the primary setup.

Start server and stop it after a few seconds letting actor to show up in Server.log or such – warning, not all rental servers services are flushing files properly (write cache crap) perhaps you need to do this at home first without other sh!t ”process managers” killing threads and not allowing flushing write buffers correctly. After this first run INI file should be created. Once completed, in next sessions is no longer saved but just read.

Next stage is setting up INI accordingly:

```
GmType[0] is first game-type using SrvPacks0[0] to SrvPacks0[31]
GmType[1] is the second game-type using SrvPacks1[0] to SrvPacks1[31]
...
GmType[63] is the last game-type using SrvPacks63[0] to SrvPacks63[31]
```

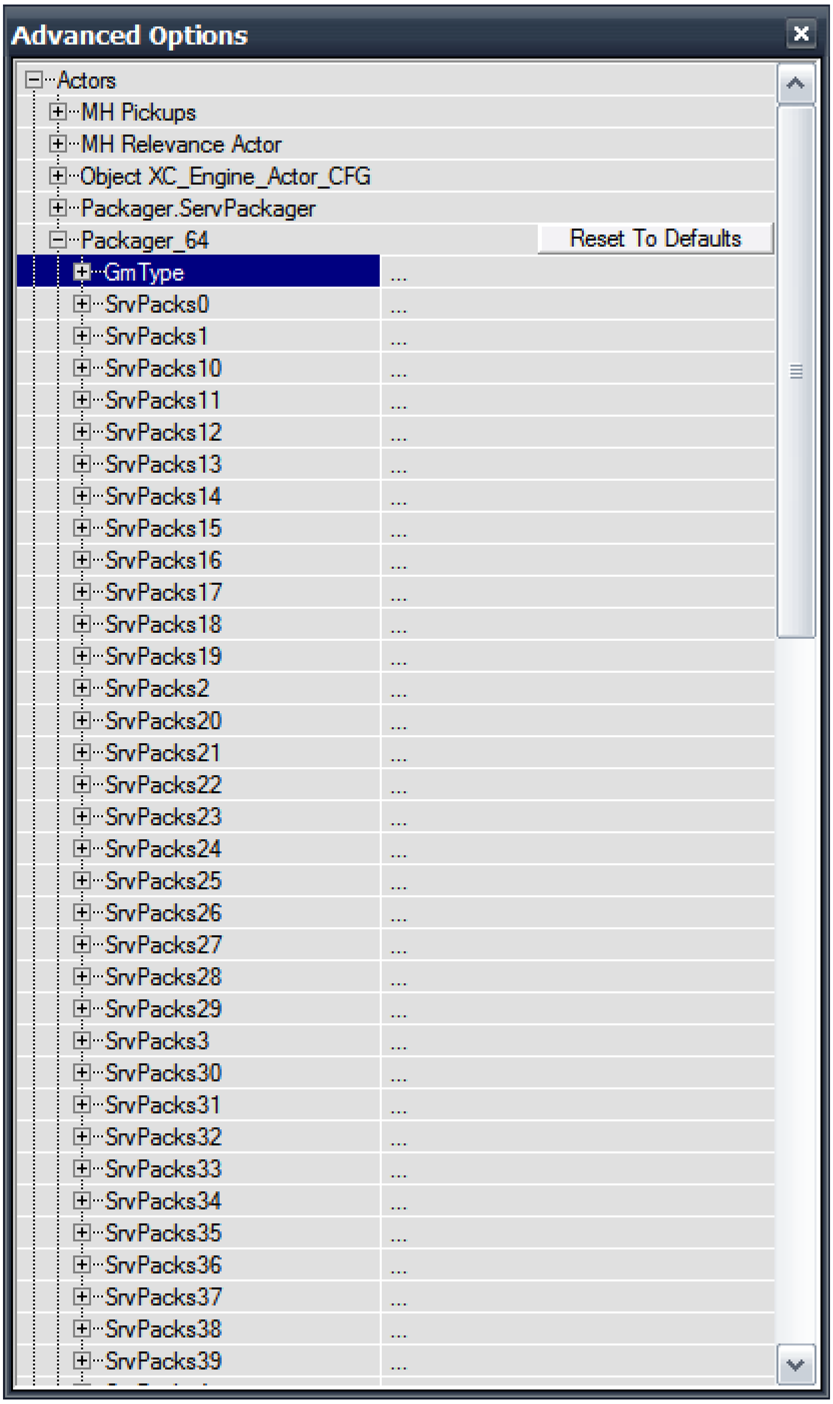
Ini file having fore mentioned name do looks like in the sample shown below:

```
[Packager_64.ServPackager]
GmType[0]=MonsterHunt
GmType[1]=DeathMatchPlus
GmType[2]=TeamGamePlus
GmType[3]=CTFGame
GmType[4]=
...
SrvPacks0[0]=Ns_SpawnPoint2
SrvPacks0[1]=
...
SrvPacks1[0]=UTToolbox
SrvPacks1[1]=NsMonster
SrvPacks1[2]=Cacher
SrvPacks1[3]=UTrooper
SrvPacks1[4]=UTrooper2
SrvPacks1[5]=
...
SrvPacks2[0]=UTMHPack
SrvPacks2[1]=UTrooper
SrvPacks2[2]=UTrooper2
SrvPacks2[3]=UTToolbox
SrvPacks2[4]=
...
SrvPacks3[0]=NsMonster
SrvPacks3[1]=Cacher
SrvPacks3[2]=UTToolbox
SrvPacks3[3]=XCTFAnnouncer_ST
SrvPacks3[4]=
...
```

Update Note: This ini might have the old entry if fore mentioned <packager> has been used, both of them are using the same ini file for preventing a lot of files created. Old entries can be deleted if previous Actor is no longer used.

Alternate method for creating INI file.

You can see some table opening a configuration menu in <preferences> ConsoleCommand and writing stuff over there - available if you have INT file provided with Actor else you cannot access Actor's properties from Advanced Menu.



Advanced Options

Actors
MH Pickups
MH Relevance Actor
Object XC_Engine_Actor_CFG
Packager.ServPackager
Packager_64 Reset To Defaults
GmType ...
SrvPacks0 ...
SrvPacks1 ...
SrvPacks10 ...
SrvPacks11 ...
SrvPacks12 ...
SrvPacks13 ...
SrvPacks14 ...
SrvPacks15 ...
SrvPacks16 ...
SrvPacks17 ...
SrvPacks18 ...
SrvPacks19 ...
SrvPacks2 ...
SrvPacks20 ...
SrvPacks21 ...
SrvPacks22 ...
SrvPacks23 ...
SrvPacks24 ...
SrvPacks25 ...
SrvPacks26 ...
SrvPacks27 ...
SrvPacks28 ...
SrvPacks29 ...
SrvPacks3 ...
SrvPacks30 ...
SrvPacks31 ...
SrvPacks32 ...
SrvPacks33 ...
SrvPacks34 ...
SrvPacks35 ...
SrvPacks36 ...
SrvPacks37 ...
SrvPacks38 ...
SrvPacks39 ...

By closing that window, an INI will be saved according to your options. This INI must go in Server's System nearby U file, or... in path declared with other files than U. In multi-server configured by myself, U files are in other Folder than INT and INI. For a Default XC Server all goes to System. Paths deal is another subject for debates not for current document.

In all cases you will want to write the game class properly and name of package which has to be mapped for that game-type. It's not hard at all.

For any problem encountered you can look at Sample INI provided in order to figure these declarations.

ONCE AGAIN - this tool is crashing a server without XC_Engine. Actor will log Packages mapped, so you can figure if you did a wrong ”packaging” task. This version for 64 game-types has a clocking system from XC_Core implemented, you can compare various servers in performance terms how fast do they move and then you can pick the right suitable service for you. Speaking about time and clocking, multi-server where I have dropped this tool has the newer XCServerCommandlet which Higor did and default UCC.EXE provided by NfoServers being a Windows based server with a very stable tick-rate.

Here is a log file fragment to confirm that this ServerActors does its job:

```
Log: Resolving unreal.epicgames.com...
Log: Resolving master.333networks.com...
ServPackager_64: >> Actor Loaded and Started.
ServPackager_64: Found game-type DeathMatchPlus
ServPackager_64: ServerPackages mapping for DeathMatchPlus is operating as follows:
ServPackager_64: Adding UTToolbox in ServerPackages.
ServPackager_64: Adding NsMonster in ServerPackages.
ServPackager_64: Adding Cacher in ServerPackages.
ServPackager_64: Adding UTrooper in ServerPackages.
ServPackager_64: Adding UTrooper2 in ServerPackages.
ServPackager_64: ServerPackages mapping for DeathMatchPlus is ended.
ServPackager_64: Mapping Packages took around 8 MilliSeconds.
```

Other coding notes:

Source-code for Actor is being added in current archive for whoever wants to modify this tool as he/she wants.

Still Credits Problems??:

Higor and Epic - Uscript is developed by Epic and new natives are developed by Higor...

And probably those people with server services rental, you cannot run an UT server in air, you need a machine.

We can keep going with credits toward our parents, they are guilty for our existence, etc. I'm affraid that credits can go in X pages written, so I'll stop here...