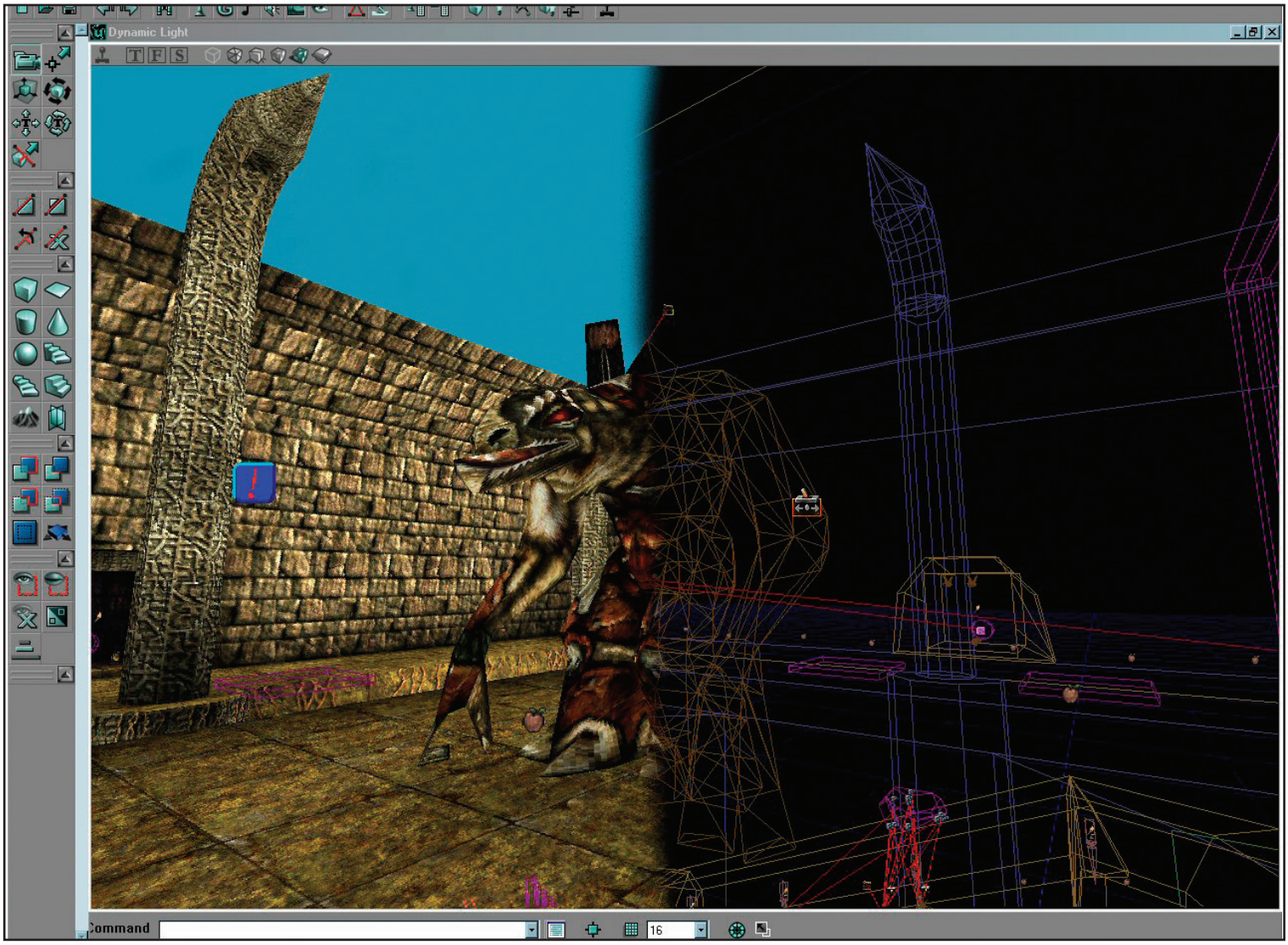
Planet Monster-Hunt MonsterHunt Mapping Guide



V-0.8

Table Of Contents

Version History

05/05/05 - V0.8: The first release of this guide. It looks a bit ugly and contains many spelling and grammar errors. Several sections have yet to be completed, including: Checkpoints, Bot Support, Advanced Trigger Systems, and Special Effects.

Preface

Thank you for taking the time to read this. This guide is intended to provide an in-depth explanation of mapping for the MonsterHunt (MH) mod for Unreal Tournament.

Before we begin, there are a few things I would like to clarify. This guide will not teach you how to use UnrealEd. By that I mean, before you begin with this, you must already know the basics of map creation within the Unreal Engine. I am not going to waste time re-inventing the wheel. There are already hundreds of other tutorials out there that can teach you how to map.

In other words; any emails I get asking how to add actors, or subtract a room, or anything else that is essential to making a map, will be promptly IGNORED AND DELETED.

Now, if you are completely new to mapping for Unreal, and would still like to make maps for MH, don't despair. You can still do so; you will just have to learn a few things first. I recommend visiting a few of these sites:

http://www.utmaniac.com/: This site contains literally 1000's of tutorials for UT, UT2003, and UT2004. I recommend looking through their beginner's tutorial section as a start.

http://www.leveldesigner.com/: Excellent Unreal mapping site.

http://www.3dbuzz.com/: Home of the Video Training Module. They have many forums dedicated to all types of 3d editing, including Unreal.

http://wiki.beyondunreal.com/: The Unreal Wiki. This site details practically every facet of the Unreal Engine.

They also have mapping topics for beginners and experts alike. I would visit this site last, as it is more technical and they expect you to know a little bit about Unreal.

Also, if you happen to have a copy of the Unreal Tournament 2004 Editors Choice DVD lying around, have a look on disc 2. There are hours of video tutorials on how to use UnrealED. Keep in mind however, those tutorials are for UT2004, so several topics they cover are not applicable to UT99. However, the same fundamentals still apply, and they are definitely beneficial.

Now then, if you feel you are competent with UnrealED, feel free to read on.

Introduction

So you have decided you would like to try your hand at making maps for MonsterHunt, but you aren't sure of where to begin. Or you have been doing it for a while but would like to learn some new techniques or brush up on some old ones. Either way, you have come to the right place. I wanted this to be an all-inclusive guide for making MonsterHunt maps, as well as a place where mappers can share various secrets and techniques that they have.

This guide is divided into 3 main parts. The Beginner section contains all of the necessary information to get you started on your first MH map. It will explain step-by-step how to set up your map to work perfectly with MonsterHunt. It also contains some general thoughts on MH mapping, which should be a good read for experts and newbies alike.

The Intermediate section details various techniques of MonsterHunt that are commonly used in maps. Things that are not necessary, but can add a lot of variation and gameplay to your maps.

Finally, the advanced section will explain some complicated techniques that can mean the difference between a good map, and an amazing one.

I hope that this guide will constantly evolve and change and I encourage others to send me comments to add. Anything you write will be credited, and it's a chance to share your knowledge and expand the community. If you have any comments/questions/criticisms/etc. then please email them to me at timmypowergamer@shaw.ca or, post them on the message boards at http://www.planetmonsterhunt.com/

I hope you enjoy reading this as much as I enjoyed writing it.

Beginners

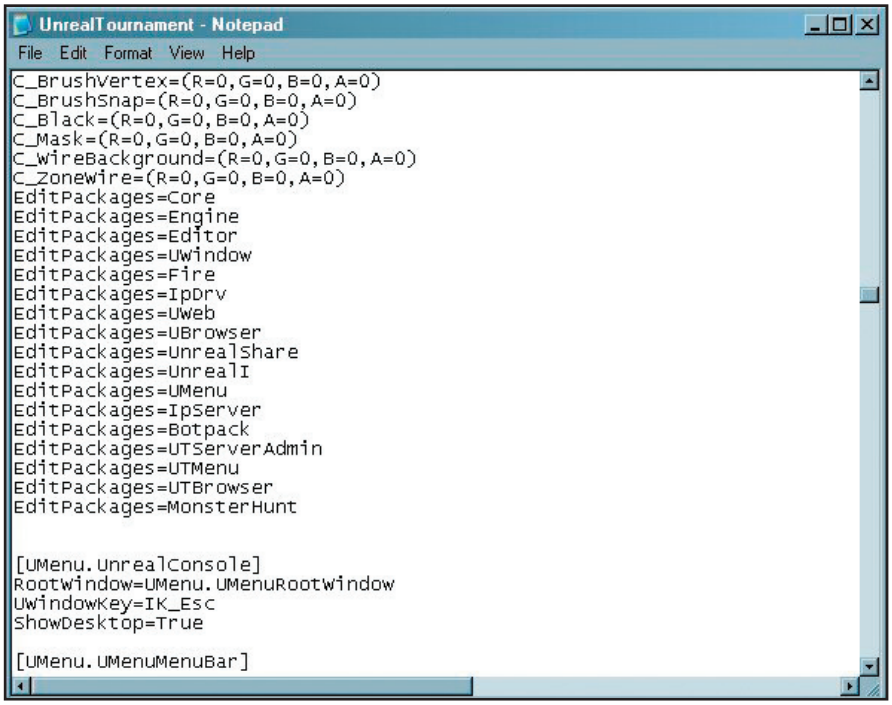
Installing MH v503

The first thing you will need to do is make sure you have the latest version of MonsterHunt installed in your Unreal Tournament directory. Start by visiting the download section at http://www,planetmonsterhunt.com/. You will need to be registered there first, however if you are reading this guide, there's a good chance you already are.

In the downloads section, click "MonsterHunt Files" then download "MonsterHunt503-Full.zip". When its done downloading, unzip it, and then run the .umod file inside. Tell it where your UnrealTournament directory is (usually c:\UnrealTournament). Then it will do the rest. You will now have all the necessary files to make a MH map.

Setting Up UnrealED For MH

Before you begin constructing your level, it will help to have all the MonsterHunt classes already loaded into the editor.



Make sure that all copies of Unreal and UnrealED are closed, then browse to your "*UnrealTournament**System*" directory, and look for a file called UnrealTournament.ini. Open it with the text editor of your choice (notepad will work just fine). Press ctrl-f to open a search window, and search for "editpackages". The first one it finds should say "editpackages=core". After that there will be a list of other editpackages. At the end of that list, add another line that reads "EditPackages=MonsterHunt" (without the quotes). Save the file and exit. Now when you open UnrealED the next time, all of the actors for MonsterHunt will already be loaded. This is mostly just for convenience, in case you begin working on a map and forget where all of your actors are.

Basic MH Mapping

Now you are ready to begin your map. I'm assuming you already have an idea for a map, or possibly it's already constructed. If so, great! If not, as you read this, feel free to create a small MH map just to test some of these topics out. It doesn't have to be pretty, just a few rooms and some hallways. This section of the guide will attempt to explain a few of the basic rules for MH mapping.

In order for your map to be playable and enjoyable in MonsterHunt, there are several things you will need to have. These include:

-PlayerStarts

-Monsters

-Weapons

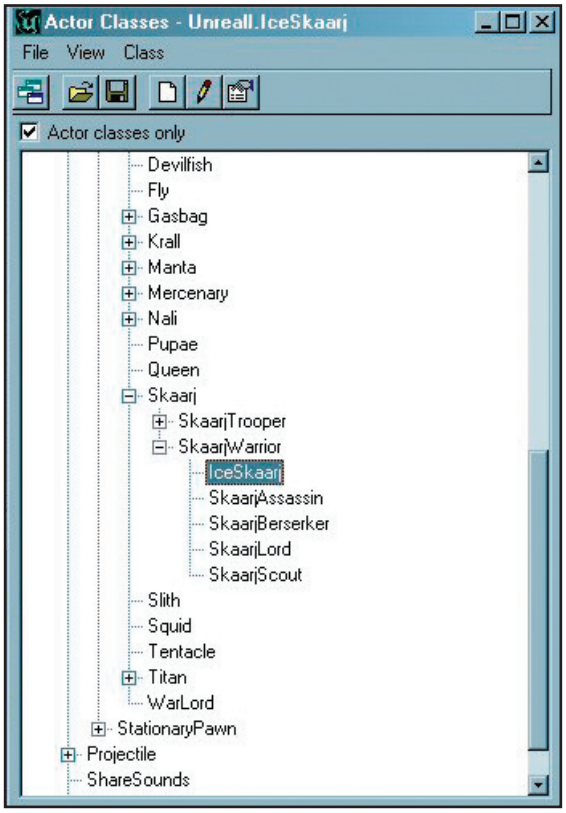
-A MonsterEnd Trigger

-Set the gametype to MonsterHunt

PlayerStarts

MonsterHunt is a team based game that allows up to 32 players to play at the same time. However, it is rare that this many players will ever join a game at the same time. However, certain maps and servers can get really busy at times, and it's not impossible to have 16 players at the same time. If your map only has 8 spawn points, half of those players will be killed instantly upon spawning. Needless to say, this is very frustrating for players and it will be looked upon as a bug. To avoid this, make sure you have at least 12 PlayerStarts at the beginning of your map, with ample room for players to spawn and move around.

Monsters



What would a MonsterHunt map be without monsters? You will want to make sure you have plenty of monsters in your map for the players to duke it out with. Typically, this means you will start the map fighting fairly weak enemies, and the difficulty should ramp up as you get closer to the end. Most maps end with a type of "boss fight" which is typically a Titan, Warlord, Queen, or other type of exceptionally strong monster.

To add a monster to your map, simply open the Actor Class Browser, and navigate to "Pawn > ScriptedPawn", and select a monster from the list. Several of the monsters have subclasses that can be added as well, such as the RockTitan, and IceSkarrj.

Then, right-click in the Perspective viewport where you would like your monster, and click "add (monstername)".

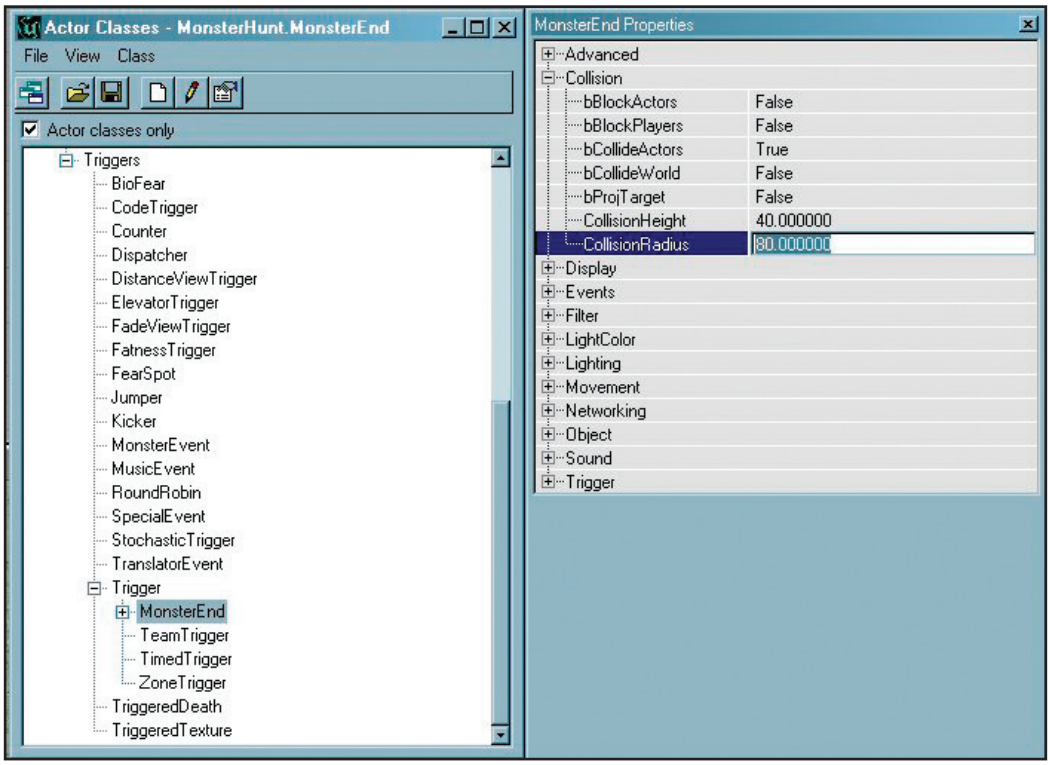
Weapons And Ammo

Your map will have to have sufficient amounts of pickups so that it the players aren't frustrated because they have nothing to kill that Titan with. At the same time, if there is too much ammo or too powerful weapons, the players will blow through your map in no-time, which is no fun either. Most likely, you will never have to worry about placing too much ammo, because it MonsterHunt, it always goes fast, especially with a lot of players. So make sure that you will always have enough for all the players. Also, try to spread out ammo of the same type, so that no one player can reach an area and take all the ammo. If you spread it out, that ensures that everyone will have a fair shot at getting some of it, without having to wait around for it to respawn.

Weapon placement is also VERY important when you are designing your level. Try to start the players off with weaker weapons, then add stronger and stronger ones as the map progresses. If everybody starts the map with a rocket launcher and a flak cannon, people will find it too easy. Not to mention that they will have nothing to work for. Usually, maps begin with weapons like biorifles, stingers, and sometimes shockrifles. Then toward the middle of the map give the player more powerful weapons such as miniguns,

rippers, pulseguns, and sometimes sniper rifles. Then finish with flak cannons and rocket launchers. Redeemers can be a nice addition on occasion, just so long as they are either hidden really well, or in a hard to reach place. NEVER NEVER NEVER, put in a supershockrifle (ESR). They totally ruin most MH maps, as they make it waayyyy too easy to complete.

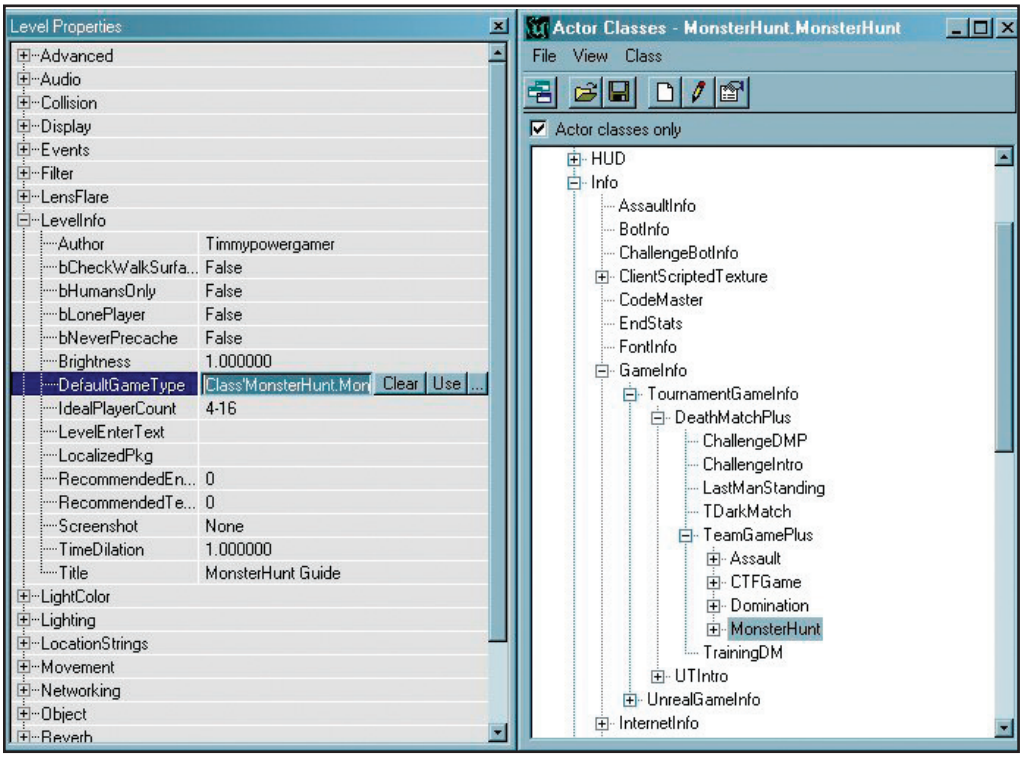
MonsterEnd Trigger



The MonsterEnd trigger is what tells the game when your map has been completed. THIS MUST BE LOCATED AT THE END OF YOUR MAP. Place one in an area that the player can run to once they have defeated the boss, or finished the last objective. It acts just like a normal trigger, in that it will go off when you enter its radius. It is located under “triggers > trigger” in the Actor Browser. After you place it, you will want to increase its collision radius so that players are guaranteed to hit it when they reach the exit. To do this, open the MonsterEnd’s properties (select it and press F4), then expand Collision, and set it’s CollisionRadius to a higher number. You can place as many of them as you want.

Setting The Gametype

The last thing you will need to set before your map is fully playable with MH, is to set the default gametype. This way, the game will know that when this map is started, it should be played as a MonsterHunt map. To do this, open the LevelProperties (press F6) and expand LevelInfo. For a start, fill out the fields Author, Ideal Player Count, and Title with whatever you would like. Then click the field that says DefaultGameType and click the ellipsis button next to it (…). This will open the Actor Browser. Expand Info > GameInfo > TournamentGameInfo > DeathmatchPlus > TeamgamePlus, then select MonsterHunt. Back in the LevelProperties, next to DefaultGameType, click use. Now your map is ready to be played in MonsterHunt.



Map Flow

I feel I should take a bit to talk about what makes a good MonsterHunt map. There are far too many poorly done MH maps out there that suffer from bad design, lack of testing, and just plain bad ideas.

There are several different “styles” of MonsterHunt map that you typically see when playing on a server. The 3 main types that come to mind are Godz-style maps, Single Player maps, and Spam maps. Those are just names that I have given them for the purpose of explaination. If you can think of a better example or name then feel free to email me. Also, the opinions below are my own. If you would like to add anything to what I have then please do email me, and I will add it and credit you.

Godz Style maps are basically clones of the popular MH-Godz map. The premise of these maps is that there are several different “courses” that must be completed. Each course is based on a different type of monster, where you must battle through hoards of that monster to face a super-strong version of it at the end (called the god). Once each god has been killed, players are transported back to a “loadout room” where they can fill up on weapons and ammo. In order to unlock new types of weapons, there are often “jumping courses” that must be completed. These consist of a series of narrow platforms that must be traversed to the end, at which point a new powerful weapon is unlocked in the loadout room. For a good example, check out MH-godz (duh…). There are a lot of others, but most of them suck, and I cant think of any names right now.

These types of map can be very fun if certain pitfalls are avoided. Often they are very ugly, with poor texturing and lighting. Always make sure that your map looks good, because people get sick of looking at the same boring grey texture for half an hour. It also shows that you don’t care enough to put any effort into your map. Also, the architecture of these maps is usually pretty basic with square rooms all over the place. Try to throw in some variation by changing up the shape of different areas, rather than square rooms and hallways. Finally, if you decide to add jumping courses (not necessary, but for some reason some people like them (im not a huge fan)) make sure that they are not too hard for beginner players. If all of the good weapons are hidden at the end of a ridiculously hard jumping course, then a lot of people will get frustrated real quick and wont want to play your map. Some maps also have jumping courses as part of the main path trough the level (i.e. MH-Herebeke). Its not recommended, but if for some reason you do it then provide a way to bypass the jumping course once its been done. Players get annoyed when they have to do the same jumping course over and over again each time they die. If you avoid doing these things, if can make for an extremely fun and playable level.

Single Player style maps are more along the lines of the maps that come with MonsterHunt. They are not intended for single player play, typically they have a linear progression, and sometimes a sort of story and objectives associated with them (much like you might find in a single player map). These maps often look the best; they have good texturing and a lot of attention to detail. However, quite a few seem to have problems with lighting. It seems that people try to make their maps feel really moody and scary, so they make the lighting very dim (or nonexistent) and justify it by giving you a flashlight. The problem is FIGHTING ENEMYS IN THE DARK IS NOT FUN. It’s annoying and frustrating, avoid it if you can. Dark sections can be useful tools if used sparingly, just don’t place a lot of strong or fast enemies in them.

Other than that, just make sure that your map has good objectives and that it looks good. These types of maps are usually my favorite, because it really feels like you are playing a co-operative single-player game, which is what MonsterHunt was intended to be. For some excellent examples, check out the maps that come with MH, most of Derdak2rot’s maps, and I like to think that MH-MayanFuryV2 is good too (but that’s my own map so I’m just shamelessly self-plugging here.)

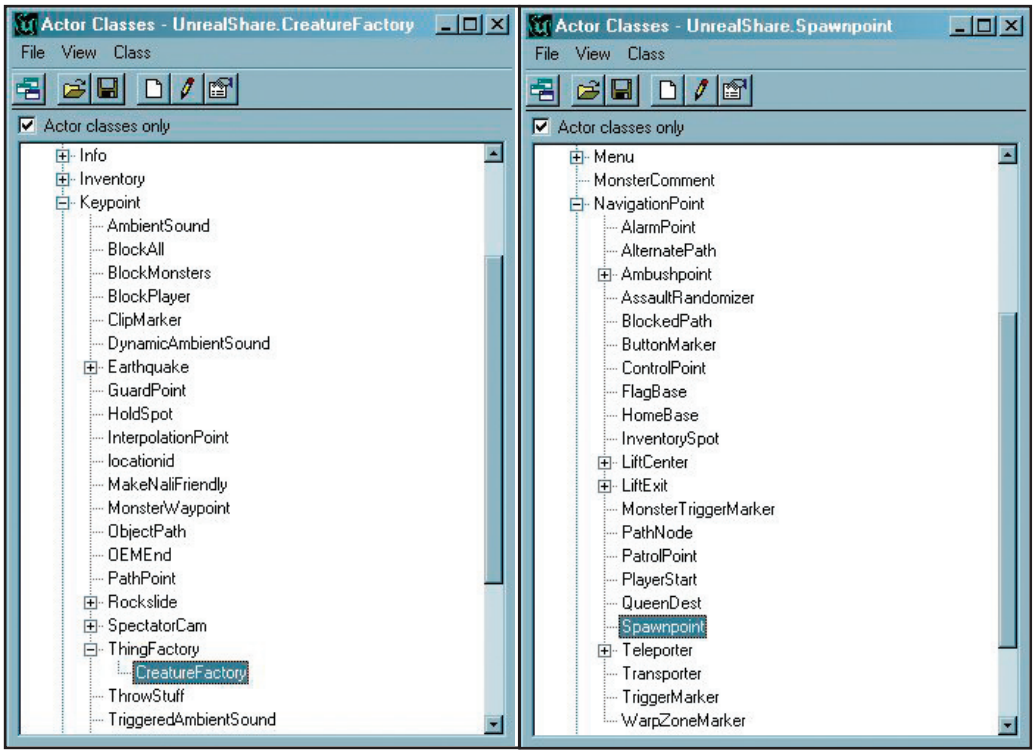
Finally there are Spam maps. Personally, I find these types of maps the most annoying. They usually consist of waves after waves of mindless slaughtering (it sounds better than it is). They are usually bland and unimaginative, and get boring very quickly. The premise of a spam map is that you flood or “spam” the map with monsters from a creaturefactory. It’s not uncommon to fight hundreds of monsters at the same time, for an extended length of time. Also, the monsters on these maps are often so difficult that the only way to complete them is using UTJMH (a mod that makes you extremely powerful). The only tip I can give here is to stay away from this type of map. That’s not saying that they can never be fun, it’s just difficult to make a good spam map that people will want to play. Good examples of spam maps (well, ones that I enjoy anyway) are MH-Brutality and MH-MonstersOfSpam.

There are other ways to do MH maps as well, but these are the type that I see the most. The best tip I can give is to properly test your level. Its very easy to just turn on god mode while you are testing and

assume that your level will be playable, but unless you actually take the map online and test it with a few people, you will never know exactly how it will play. Finally, just make sure that it looks good. Even if you are not a skillful mapper, there is no excuse for bad lighting and texturing. Make it look like you care, and your map will be 200% better.

Intermediate

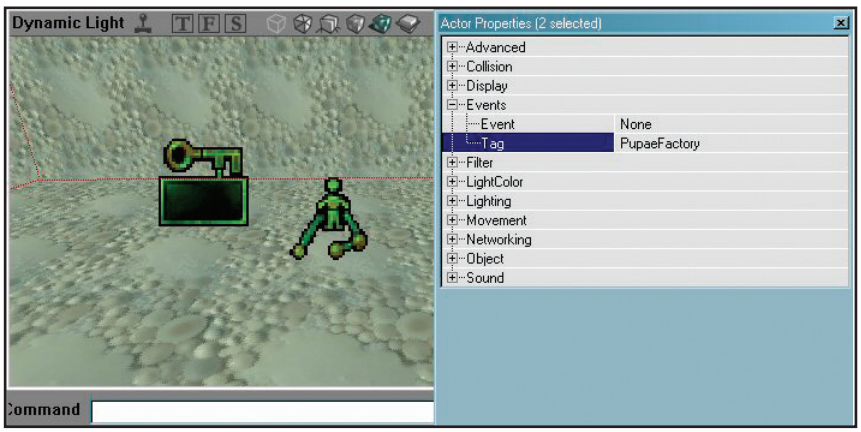
Creature Factories



Creature Factories are an extremely useful tool for an MH mapper. They are actors you place in your map that when triggered will begin spawning monsters at predetermined spawn points. This can be a huge time saver as it keeps you from having to place hundreds of monsters. Also, because all of those monsters don't exist within the game until you actually need them, it improves the speed of the map because the engine doesn't have to keep track of all of them all the time. They can be somewhat confusing to set up at first, but I will attempt to make it simple as possible for you.

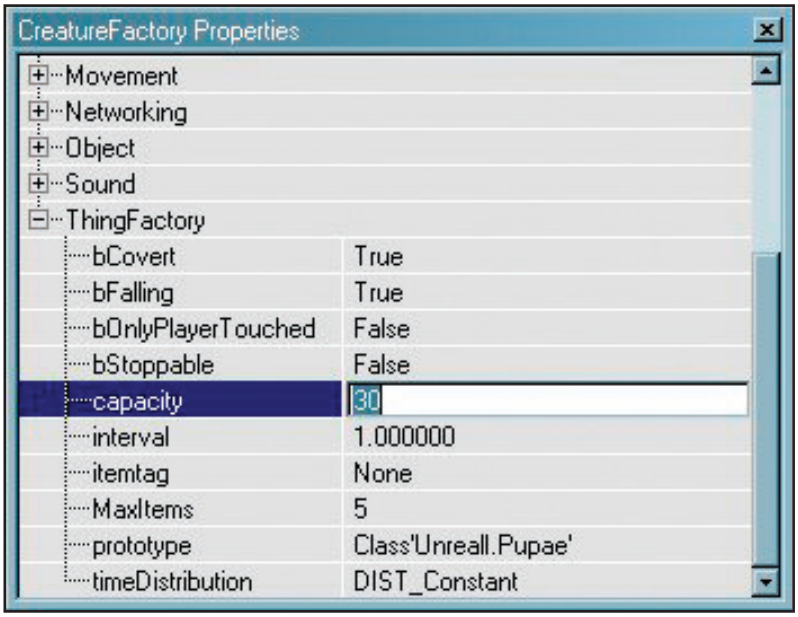
The first step is to place a CreatureFactory within your map. It doesn't matter where, however you should keep it in the same area as where you would like your monsters to spawn, just for easy access. The CreatureFactory is located in the Actor browser under "keypoints > ThingFactory". I should note here that a ThingFactory will do the same thing as a CreatureFactory, however a CreatureFactory has some extra options and settings that make it easier to use for spawning monsters. So you might as well just use a CreatureFactory, they are there for a reason. I say that because I have seen lots of people using ThingFactorys to spawn monsters, and they wonder why it doesn't work perfectly all the time.

Ok, so now you have a CreatureFactory (I'm going to just refer to it as a "CF" from now on cause I'm tired of typing CreatureFactory) but you still need to add some spawnpoints for the monsters. The Spawnpoint is located under NavigationPoints in the Actor browser. Simply add them wherever you would like the monsters to spawn, but you can only place up to 32 for each CF. Often it is wise to place them out of view of the player, so that they can't see the monsters spawning, but its not critical. When you are done, select all the Spawnpoints you placed and the CF, open their properties (F4) and give all of them a unique tag. So for example, if the monster you want to spawn are pupae, call all of the Spawnpoints and the CF "pupaefactory". This is a little different than most other actors, in that you are linking several actors by using the same tag, instead of tag and event.



Next you need to set up the CF to spawn the right monster. Select it and open its properties, then expand "ThingFactory". There are several options here you can change. I will explain which ones you need to change, then I will explain what the others do. The first field you need to set is the "proto-

type" field. Highlight it and hit the browse button, then navigate in the actor list to whatever monster you would like to spawn. So using my pupae example again, browse to pawn > ScriptedPawn > Pupae and select it. Then click "use" in the CF's properties. Next, you need to tell it how many to spawn. There are 2 fields that dictate this: MaxItems and Capacity. Capacity is the total number of monsters you want to spawn from the factory before it shuts down. MaxItems is the maximum number of monsters from that factory that can be in the game at one time. For example, if I wanted to spawn a total of 50 pupae, I would set the capacity to 50. However, 50 pupae on screen at the same time is a lot, so I would set the MaxItems to 15. That way, there is only ever at most 15 pupae on the screen at a time. Each time one is killed, it will be replaced by the CF until all 50 have been spawned, then it will never spawn any more. If you set the capacity to –1, it will continue spawning monsters forever. The last thing you need to do is set something else to trigger it. So in another actor (for example a trigger) set its "event" field to the tag of the CF. You should see a sort of spiderweb of red lines joining the trigger, all your spawnpoints and the CF. When the trigger is activated, the CF will begin spawning monsters.



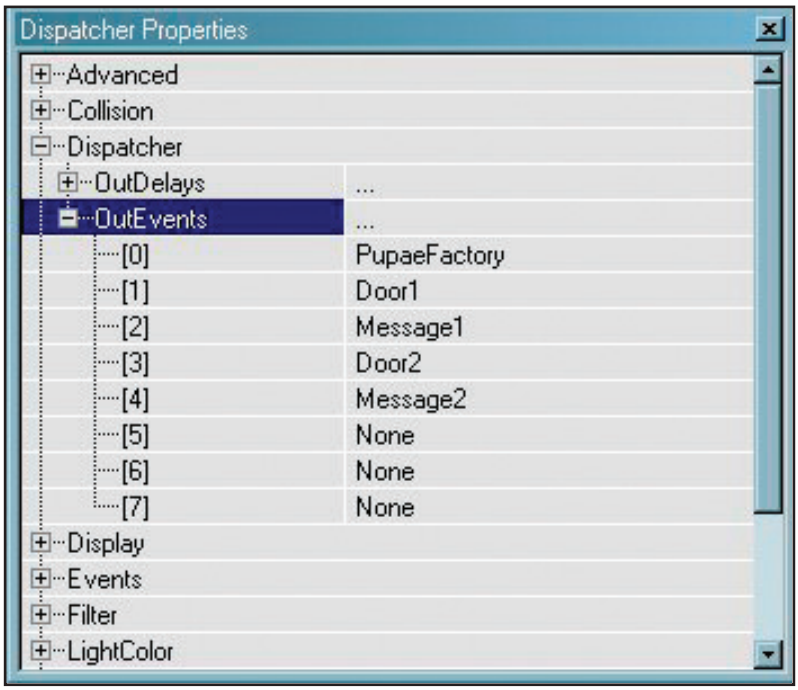
Here are some other properties you might want to change:

Interval: Sets the amount of time in seconds between each spawn. If you wanted all of the monsters to suddenly appear at the same time, set this really low (like 0.01). If you wanted them to gradually appear set it higher.

Distribution: Changes the time betweens spawns, by changing the distribution pattern. You probably wont need to change this, as it doesn't become noticeable unless the Interval is very high.

bCovert: When set to True, this makes it so that monsters will only spawn at spawnpoints that nobody is looking at. If False, you will be able to see the monsters spawn. This is set to True by default on the CF.

bStoppable: If set to true, then you can stop this CF from spawning its monsters by triggering it once again after its already been triggered.

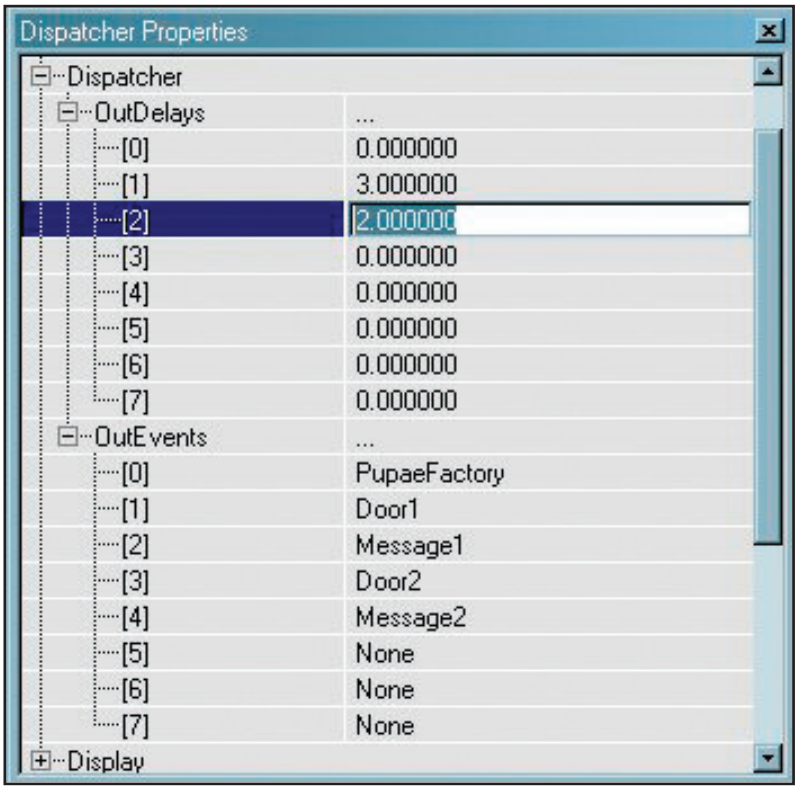


bFalling: When true, anything spawned from this factory will automatically fall to the ground. Change this to false if you are spawning flying monsters such as Manta

Dispatchers

In MonsterHunt, there may be a lot of times when you need to trigger a bunch of events in a row, or at the same time. You may also need to put a time delay between a few events. Instead of having a big mess of red lines all over the place linking each event to the next, use a dispatcher. This handy little tool is incredibly easy to use. Simply add one to your map in any location. It's located (oddly enough) under triggers in the Actor browser. Then open its properties and give

it a unique tag. Then expand “Dispatcher > OutEvents”. There are 8 Event fields. Simply add any events you want to trigger to those fields in the order you would like them to trigger.
Example:



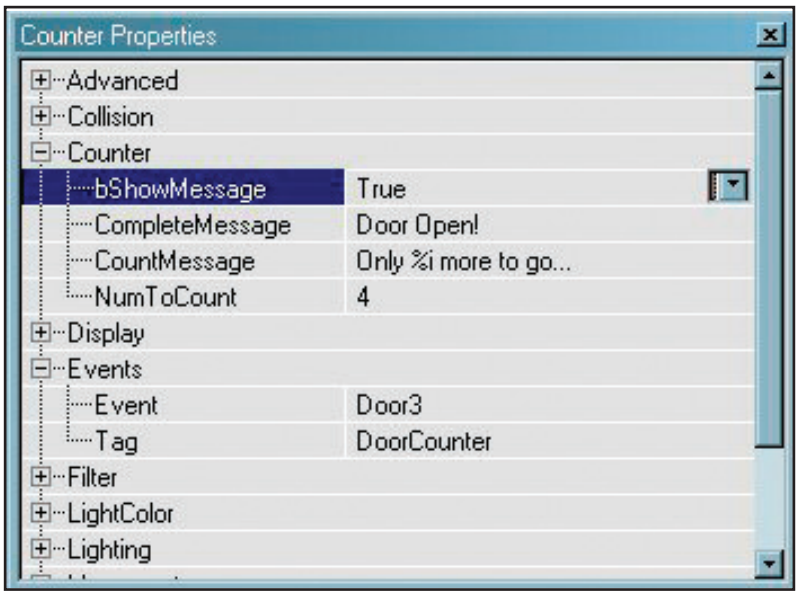
Now if you are trying to make all of them trigger at the same time, then you are done. Since you have not set any delays, when the dispatcher is triggered, every event in the OutEvents list will be instantly triggered. However, if you need to wait a few seconds before one of those events triggers, then you need to set up some delays. Expand “OutDelay”, you will notice it has 8 number fields, each one corresponding to the matching OutEvent. You can set the time in seconds for it to wait between each event. Keep in mind however that delays are cumulative. This means that if you set a 3 second delay on event 1, and a 2 second delay on event 2, then event 2 will trigger 5 seconds after the dispatcher is triggered. It will immediately trigger event 0, then wait 3 seconds and trigger #1, then wait an additional 2 before it triggers #2.

That’s all there is to dispatchers, if for some reason you need to trigger more than 8 events at a time, simply set the #8 OutEvent to the tag of another dispatcher and keep going. Dispatchers are much cleaner and more convenient that linking a string of events together.

Counters

Counters are fairly essential to a good MH map, and are even easier to use than dispatchers. Basically a Counter will (oddly enough) count the number of times its been triggered, then when a certain number has been reached it triggers another event. This can be very useful in cases where you may need the player to complete multiple events before proceeding. For example, you may need them to hit 4 switches to open a door. Or kill a certain number of monsters. One of the more difficult things that can be accomplished with counters is to ensure that all creatures from a CreatureFactory have been killed before proceeding. I will explain how to do that in a later section (it’s a little more complicated than you might think).

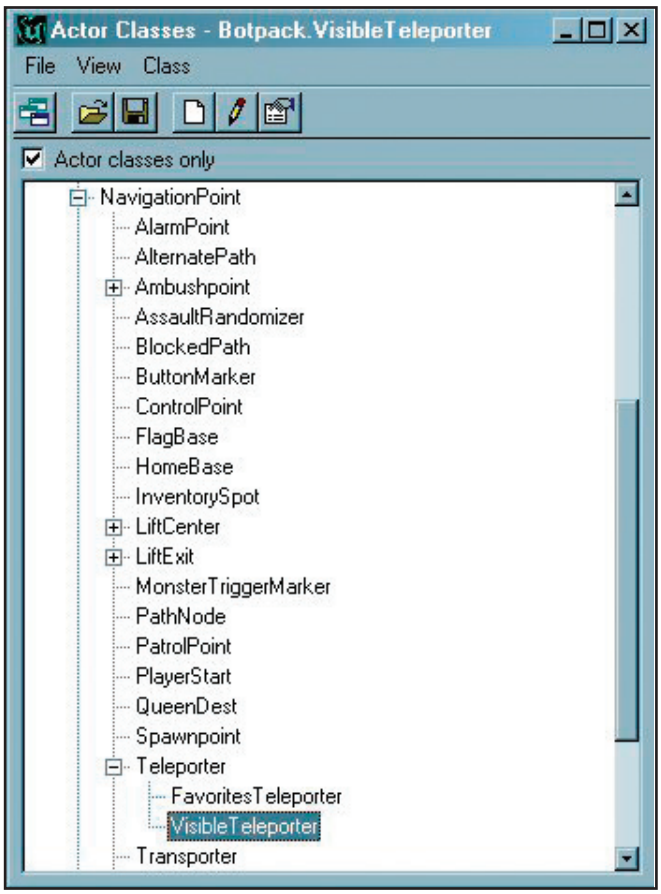
Anyways, to set up a counter you must first place one in the map. Again, location doesn’t matter, but keep it close to the area where its used for simplicity’s sake. The counter is found under “Triggers” in the Actor Class browser. Open its properties and give it a unique tag, under “Events”. Also set its Event to whatever you would like to have triggered. Then expand the “counter” tag and set NumToCount to however many times you would like it to be triggered. You can also set messages that will be displayed in the chatbox when it is triggered. The CountMessage displays every time the counter is triggered. By default it says “Only %i more to go...”. The %i is the number of times it still has to be triggered, the counter keeps track of that automatically. You can also set the completed message which will display when the counter has reached its limit. The messages will only display if you have bShowMessage set to True. By default it is false so you will not see anything as it gets triggered. Setting messages are useful when you want to inform the player that they should be finding more switches/monsters to kill, or if something about your counter is not working correctly and you



need to debug it.

That's about it for Counters. Like I said, they are simple. There are some interesting things that can be done with them, and they are used all the time in MH mapping.

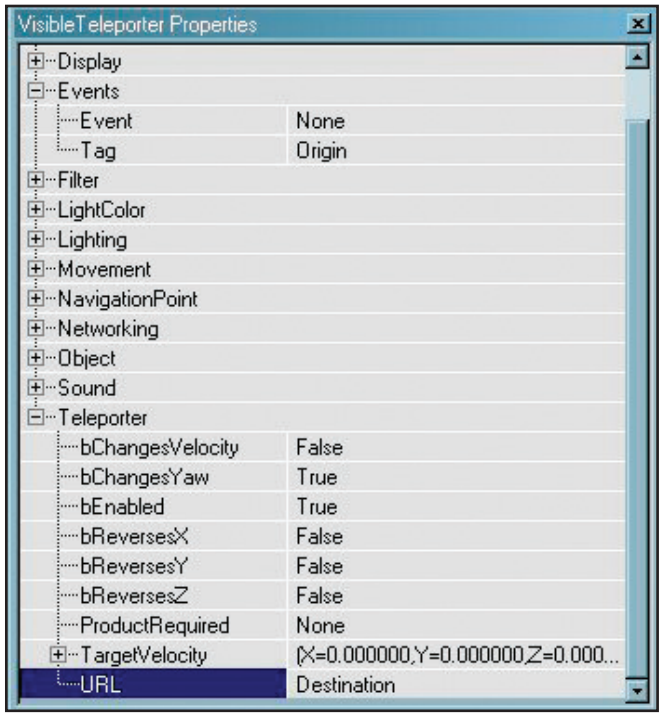
Teleporters



Most people that have done any other type of mapping know how to use teleporters, as they are used extensively in DM, CTF, and just about every other gametype out there. MonsterHunt is no different either. However, there are certain things that you may want to do with teleporters that aren't typically done in other gametypes, such as locking them, or making one-way teleports. Ill go over how to do each of those here, as well as explaining how to set up a basic teleport for anyone that doesn't know how.

To set up a basic teleport system, start by adding 2 Teleporter actors to your map, in the positions you would like the player to teleport from/to. They are located under NavigationPoint in the Actor Class browser. There are 3 types of teleporter. The basic "Teleporter" actor can not be seen in game. It is used when you don't want the player to know there is a teleport spot there (useful for 1-ways) or if you have made your own teleport effect. If you expand the "Teleporter" you will see 2 others: the VisibleTeleporter and the FavoritesTeleporter. The VisibleTeleporter is what you will probably use most often, as it shows up as several spinning blue rings in game. They are used all the time in other gametypes because they are quick and easy to use. Im not sure what the FavoritesTeleporter does, and the Unreal Wiki doesn't even have a description for it. So just ignore it, because it probably isn't useful to you at all.

Once you have your teleporters in position, you will need to tell them where they are teleporting to. Select one and open its properties, and set its tag to something unique. For this example we will call it "teleport1". Then expand "Teleporter" and set the URL field to the tag of the second teleporter, which we will call "teleport2". Then repeat the process for the second teleporter; Give it the tag "teleport2" and the URL "teleport1". That's all there is to it, you should now have a working 2-way teleporter.



In the case where you need your teleporter to only go in one direction, you will need to set bEnabled to False on the destination teleporter. I have seen many maps where to achieve this the mapper simply leaves the URL field of the destination teleport blank. This causes a small error in Unreal and it will show up in the textbox as "destination does not exist" or something similar. So make sure you set bEnabled to false if you are doing a 1-way. Also, I recommend making the destination Teleport a regular "Teleporter" instead of a VisibleTeleporter. That way the player will not mistakenly run into it trying to escape a monster, only to find that nothing happens.

Lastly, if you have a Teleport with bEnabled set to false, and you trigger it using another actor (like a trigger or a monster or something like that), then it will become active again. This is used when you have a teleport to be "locked: at first, and then become active when a certain event has been triggered (like pressing a switch).

Checkpoints

This section is to be completed at a later date. It will explain how to set up checkpoints in your map, so that once a certain point has been reached the player will spawn there instead of at the beginning. Useful if you have a very long map. Expect this on the next update.

Advanced Trigger Systems

Here I will explain some ways of combining some of the above actors into more complicated, but useful systems. All I'm talking about is combining the effects of various triggers to create a new effect. This way, mappers can create a sort of "Custom Trigger" without knowing any sort of coding or UnrealScript. It requires a bit of ingenuity and imagination, but there are some rather complicated effects that can be pulled off without writing a single bit of code. I will give an example of this that seems to pop up quite frequently on forums, involving counters and creaturefactories. I may also add some other good ones that I think of or are submitted to me. If you have a good example or trick that you think should be in here, please email me or post it on the PlanetMH forums. Now on to business.

A frequent problem that I've heard is how to get a counter to trigger after every monster from a CreatureFactory is killed. It seems like it would be easy to do, but because of the way the CreatureFactory works, it's actually somewhat complicated. A coder would simply re-write to CreatureFactory to support this, but most mappers don't have that luxury. So here is an effective way to solve that.

The first thing to do is set up your CreatureFactory. There should be a CreatureFactory actor, and several Spawnpoints that all have the same tag (as described in the above section). Once you have your CreatureFac working properly, its time to add a counter. Set the Tag of the counter to the same thing as the CF and Spawnpoints. So if the CF is named "pupae", the counter should be named "pupae" as well. This is somewhat contrary to what most people would think, but trust me, it works. Now you will have to set the NumToCount on the counter to the same thing as the Capacity of the CF, only +1. So if the Capacity of your creatureFactory is 30, set the NumToCount on the counter to 31. Then just set the Event of the counter to whatever you want (a door for example). When you run the map, the door will only open once every monster from the factory is dead.

The reason why this works, is because of the way the CF is coded. You see, every time a monster is spawned from the factory, that monsters Event is set to the Tag of the CF. When the monster dies it triggers that event, this tells the CF to reactivate (unless it has reached its capacity already). So by setting the tag of the counter to the same thing, it in turn ALSO gets triggered each time a monster dies. The reason you have to add 1 to the NumToCount is to make up for the initial trigger, that activated the CF. Without it, the door would open after 29 monsters have been killed, because it had been triggered 29 times, plus the time that started the whole process. That leaves one monster still running around.

Another thing you could do with this is have it so that monsters from 2 different factories must be killed before proceeding (say if you want 2 types of monsters attacking at the same time). This can be done by repeating the process with the second factory, only instead of having the counters trigger a door, have them trigger another counter set to 2. That way, once both Factories have been exhausted and the monsters killed, the door will open.

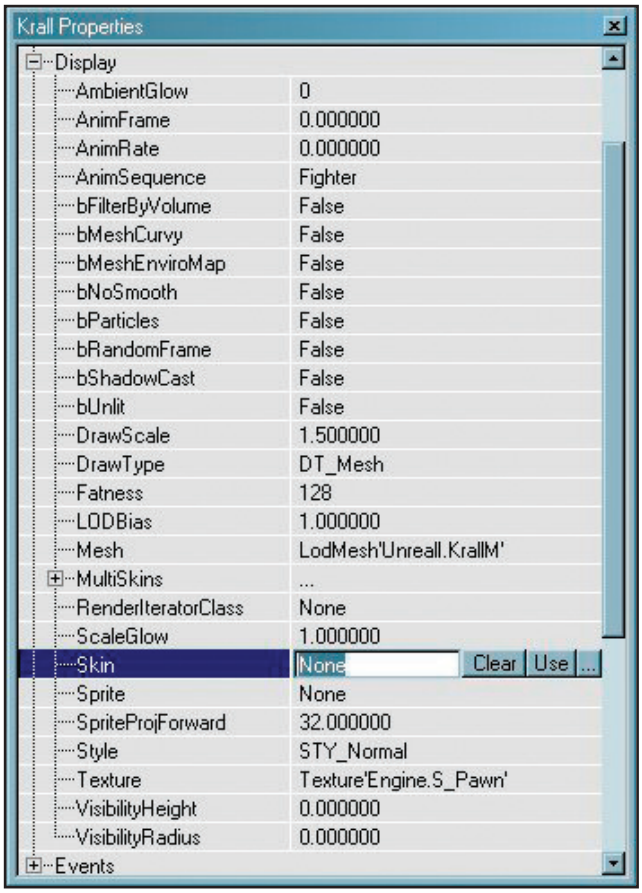
Like I said, you kind of have to use your imagination for these things. It also takes quite a bit of patience and determination. If you don't understand any of this, or have something to add, please feel free to let me know. I wasn't even sure if I should talk about this or not (or what section it should go in) so please also let me know if it's helpful.

Ok, that's all for the Intermediate section of this guide (for now anyway). Next we will move on to some of the more technical and complicated things that can be done to your maps.

Advanced Bot Support

This will be completed at a later date. Bot support for MH is fairly tricky and requires some explanation. Also, since not many people play MH with bots, im sure nobody will mind if this is left out for now. However, I will say that adding bot support is not only a great way to test out your level for gameplay, but it really shows you went the extra mile for your map. Anyways, more to come at a later update.

Modifying Monsters

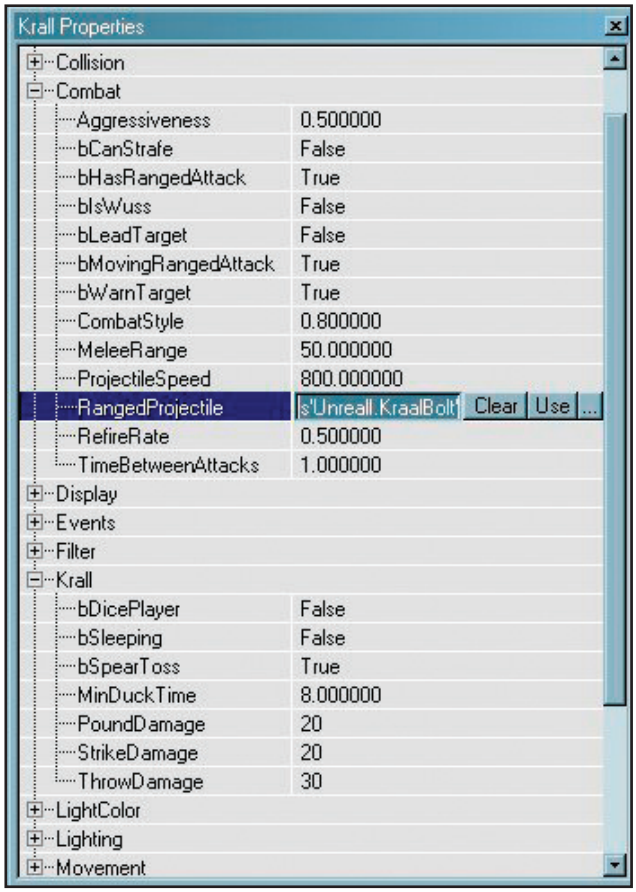


This is a great way to expand on the gameplay of MH, and anybody can do it with a little knowledge. I will explain here how to change the attributes of monsters, such as making them larger and sronger, changing their skins and projectiles, and sometimes even their behavior. None of this requires any knowledge of coding or Uscript. The process of editing monsters is really very simple, but to do it well you need to know what everything does, and what you can and can't change. First of all, to modify a monster, it is as simple as adding one to your map and opening its properties. There are many, many different properties you can change for each monster, I will go through as many as I can.

The 3 main changes most people make to their monsters are size, health, and damage. To alter the size of a monster you will need to expand the "display" tab in its properties. In Display there are 3 fields you might want to change. The drawscale field will alter the size of the monster. Changing it to 2 will double the size of the monster, and 0.5 will half the size. You can also change the skin that the monster uses. Simply select a texture in the texture browser and click use. Most people will do things like making a Water Titan by applying a water texture and setting style to Translucent. Its difficult to make a proper skin for a monster, so unless you know how to skin i wouldnt bother trying anything complicated. Those 3 settings are the only ones you will most likely need to change in the display tab.

To change the health of the monster, simply expand the "pawn" tab and change the value for Health. Keep in mind that the drawscale effects this value when you are deciding its health.

Lastly, to change the damage a monster does, look for a tab named after the monster you are working on. So if you were modifying a krall you would expand the "krall" tab. Inside there are modifiers for various melee attacks the monster has. You can also change the projectile the monster uses. Just expand the Combat tab and change RangedProjectile to another projectile. They can be found (oddly enough) under "projectiles" in the actor browser. You can also change other attributes such as RefireRate, ProjectileSpeed, and Aggressiveness. Most of the properties are fairly self-explanatory, however if you are ever unsure you can look it up in the Unreal Creatures Guide. Just about every attribute of Scripted Pawns and unreal AI is described in it, so it's your best resource if you are doing anything difficult.



There are limitations to this method of modification. You can't modify monsters spawned from a creature factory, or change the damage or skin of a projectile. So here is a simple way to get around all that:

The first thing you need to do is create a new class. To do that, just right-click the monster you want to edit in the Actor browser and select New... Then in the window that opens, enter "MyLevel" into Package, and give it a unique name, then click OK. A blue window will open with your new script in it. Just close it, you don't have to do any scripting for this. Go back into the Actor browser and have a look at the monster you are editing. Your monster will be a subclass of it. Just right click-it and select Default Properties. Now you can

modify its properties just like you would a normal monster. When you are done, just place the monster somewhere in your map (or use it in a CreatureFactory) and then save your map. Because you saved it to the package MyLevel, you will not have to distribute any extra packages with your map. You can also use this same technique to modify Projectiles, Weapons, and Pickups.

Compressing Maps

Most server operators will thank you if you include a compressed version of your map when you release it. It saves them time and speeds up download times when playing online. There is a simple way to compress your map, using a program that is included with UT called UCC. To compress a map using UCC, you must open up a command prompt in windows, and navigate to your UnrealTournament/System directory. Then type “ucc compress” and then the path to the map you want to compress. It would look something like “ucc compress c:/unrealtournament/maps/mapname.unr”.

Now if you think that sounds like a lot of work, then you are right. Fortunately, there is a much easier way. Simply download this batch file and place it in your UnrealTournament/System directory. Then create a new folder on the root of your C drive called “compress” (without quotes). When you are ready to compress your map(s) simply copy them to the compress directory and run compress.bat (the batch file you placed in your UT/System folder). You should also include any music, sound, texture, or .u files that your map uses. When its done, there will be new files in the compress folder with the .uz extension. Simply include those with the rest of your map files when you distribute it.

If you created the Compress folder somewhere other than C:\compress, then you will have to edit the batch file. Simply right-click it and click edit, then change everything that says “c:/compress/” to the correct path.

Skillz writes:

For linux and most likely mac's as well, the command parameters are:

/ucc-bin compress [pathtofile/filename.extension] -nohomedir

Just make sure that ucc-bin has write privileges, and the directory it's going to be copied to has write privileges as well. Any user using Linux should know directory/file permissions. Wild cards are accepted as well. So for example you can:

cd /System
/ucc-bin compress ../Maps/.unr -nohomedir*

It will compress every single map in the maps directory.

So there you go. If you are trying to compress on Linux or a Mac then that would be the process. Of course, UnrealEd will not work on Linux so unless you are running a server then there is no reason to use Linux for that.

Contact Information

Thank you again for reading this guide. If you have any comments or suggestions (or spelling errors to report ;)) then feel free to contact me at timmypowergamer@shaw.ca, or message me on the Planet MonsterHunt message boards: http://www.planetmonsterhunt.com/ .

As I said at the beginning of this guide, I will only answer questions directly regarding the information in this guide. Questions about using UnrealEd, UnrealTournament or anything that you WOULD ALREADY KNOW IF YOU TOOK THE TIME TO SEARCH FOR IT will be ignored. Spam or flame attacks will not be tolerated either and will result in a ban from PlanetMH or (in the case of an email attack) legal action.

Credits

I would like to thank everyone at Planet MonsterHunt for patiently waiting for this guide. Thanks to Shrimp and the entire original MonsterHunt team for this excellent Modification Special thanks go out to Skillz for hosting this and for his excellent site. Also, I would like to thank everyone who has ever written a tutorial for UnrealEd, for without them I would never have learned all this in the first place. If you feel that your name should be on this list and I forgot you, feel free to contact me.

Legal Information