Subject: Renegade API C++ help needed
Posted by reborn on Wed, 07 Jan 2009 13:16:45 GMT
View Forum Message <> Reply to Message

Hey guys, I cannot remember if I shared this or not now, I don't think I did, but I meant to, I wrote something a few months that allowed me to have my map rotation as random.
There have been a couple of changes to the code since I first wrote it, I found that the function Get_Random_Int was not actually very random at all, infact it became really predictable and players moaned allot about the rotation wasn't as random as it might seem in the short run (however over a long period it was).
I made a simple random number generator to replace that function and it seems better now.
My function "requestrandommap" works, I call it on the level_load event so each time the map loads, the next one is random.
This however poses a problem, the first time the server starts, it uses the first map in the rotation list, all the maps after are random, but it always has to use the first map in the list because I call my function on level_load.

Can anyone think of a way to make the first map when the server loads random? Or a place where I can call my function when the server first loads? I thought about doing a gameover console command after the server has first loaded, but I am unsure of the best way to tell when the server has first loaded.

If you can help and suggest something, I would apreciate it. My server solution is based on SSGM.

For anyone reading this thread, but doesn't know how to hel me, but is interested in there own server having a random map rotation, check out the source code:

You will need to include these libraries:

```
#include <math.h>
#include <iostream>
#include <time.h>
#include <stdlib.h>
```

you will need to use namespacestd:

```
using namespace std;
```

Here is the function for getting a random integer value between 0 and max that I use. It isn't clever, but it seems to work better for this situation than Get_Random_Int, and I do not need a min value either, so it's fine for this use.

```
int Get_Random_Int_Not_Crap(int n){
return rand() % n;
}
```

Most bots have a groovey little function for announcing the next map and responding to chat hooks like !nextmap. I have hard coded this into the scripts.dll file, so I guess you would need to remove it from your bots if you want to use it.

You need to declare the global variable at the top of your .cpp file, I am using gmmain.cpp myself...

```
char *mapname;
```

Here is the map announce function that gets called quite a bit:

```
//This just announces what the next map will be. The global variable "mapname" gets set by the request random map code.
void mapnameannounce(){
Console_Input(StrFormat("msg The next map will be %s",mapname).c_str());
}
```

Here is the actual function for requesting a random map:

```
//This is the function that makes the next map a random one from the current list of maps on your server.
//You can call it at any time with "requestrandommap();".
void requestrandommap(){
//I get the name of the current map here
char *currmapname = The_Game()->MapName;
//I initialise and delcare the variable "numberofmaps here"
int numberofmaps = 0;
//Many thanks to Roshambo for this nice little "for" loop
//The loop is basically responsible for getting the amount of maps in the rotation
for(;*The_Game()->MapList[numberofmaps] != 0; numberofmaps++);
//I get a random number between 0 (maps use 0 based indexing) and the amount of maps in rotation (hence the need to know the amount of maps).
int RandomNum = Get_Random_Int_Not_Crap(numberofmaps);
//This code here makes the server think that the current map is a different one, so it logically will play the map next in the list to the one it thinks is currently being played
The_Game()->MapNumber = RandomNum;
//Therefore the next map that will get played is the one after the one that the server thinks is
```

```
playing right now (but isn't), so this is how I get the name of the next map
mapname = The_Game()->MapList[RandomNum +1];
if (RandomNum + 1 > numberofmaps - 1){
mapname = The_Game()->MapList[0];
}
//Code to make sure the next map will never be the same one as the current map, you need to be running more then one map for this to work. I will always use more then one map so I never bothered to account for this.
if ((strcmp(currmapname,mapname))== 0){
 Console_Output("Had to request a respawn, just saved you playing the same map twice...\n");
requestrandommap();
}
else {
//Just log the next map on the console
Console_Output("The next map will be: %s\n",mapname);
//Console_Output("The next map number is: %i\n",RandomNum + 1);
//Call the function that announces the next map to be played in-game.
mapnameannounce();
}
}
```

I slip the "requestrandommap" function in the "level_load" event of SSGM here:

```
void Level_Loaded() {
//This sets the seed for random number generation
srand(time(NULL));
//This means that each time the map loads, another random map is set after it
requestrandommap();
```

Here are a bunch of chat hooks I use so that players can use in-game to see what the next map to be played will be:

```
class mapChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<mapChatCommand>
mapChatCommandReg("!nextmap",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map2ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
```

```
mapnameannounce();
 }
};
ChatCommandRegistrant<map2ChatCommand>
map2ChatCommandReg("!next",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map3ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<map3ChatCommand>
map3ChatCommandReg("!n",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map4ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<map4ChatCommand>
map4ChatCommandReg("!N",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map5ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<map5ChatCommand>
map5ChatCommandReg("!NEXT",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map6ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<map6ChatCommand>
map6ChatCommandReg("!NEXTMAP",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map7ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
mapnameannounce();
 }
};
ChatCommandRegistrant<map7ChatCommand>
map7ChatCommandReg("!Nextmap",CHATTYPE_ALL,0,GAMEMODE_ALL);

class map8ChatCommand : public ChatCommandClass {
 void Triggered(int ID,const TokenClass &Text,int ChatType) {
```

```
mapnameannounce();
 }
};
ChatCommandRegistrant<map8ChatCommand>
map8ChatCommandReg("!Next",CHATTYPE_ALL,0,GAMEMODE_ALL);
```

So, if you feel like stretching your brain a little bit, then give me a hand.

I have to mention a thanks to Roshambo, he wrote the for loop that counts the amount of maps the server has in it's list. An awesome guy if ever there was one.
