|  |  |
| --- | --- |
| |  | | --- | | **Multiple multiplayer objectives :: Bjarne Grönnevik** | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| This tutorial will combine the bomb and document objective tutorials into a map with multiple objectives of multiple types.  Oh yeah, you'll probably want to put the objective somewhere, so I recommend building a map, made up only of one small room ( To minimize compile times while you are learning ). All red words below are keywords that should not be changed, the rest you can change as much as you like.  Step 1) Do the bomb tutorial. Step 2) Do the document tutorial. Step 3) Get confused and wonder how to solve the win-conditions. Step 4) Smile and read on:  Use the exploder system's varaiable level.targets\_destroyed and set the variable level.targets\_to\_destroy to 2 ( one bomb and one document ). This way there is no need to change the allies win method from the bomb tutorial:   |  | | --- | | // Allied victory test  allies\_win\_bomb:  // While undestroyed objectives left  while(level.targets\_destroyed < level.targets\_to\_destroy)  // chill out  waitframe  // No objectives left allies win  teamwin allies  end // end allied victory test |   But what about the document? Well... that needs a change from the document tutorial. The **desk\_document\_check** method from the document tutorial will work if you just replace the **teamwin allies** line with a line that just adds 1 to the level.targets\_destroyed when a document is stolen:   |  | | --- | | // Document checks  desk\_document\_check:  while(1) { // forever  // Dont execute past this line  // until someone triggers the object  $documents\_trigger waittill trigger  // parm.other is the triggerer  if(parm.other.dmteam == allies) {  // Make the document graphix disappear  $documents hide  // Tell the win method that an  // objective has been completed  level.targets\_destroyed ++ // ++ adds 1  // break out of the while loop  break  }  // protection against making this  // thread use too much CPU  waitframe  }  end // end document checks |   OK, this will now work for any number of bomb objectives ( just add 1 to the level.targets\_to\_destroy for every bomb ) as the exploder system ( exploder.scr ) can handle this. But say you want to add another document theft objective, will that also work? Well, yes and no: Using the level.targets\_destroyed will work for any number of objectives, but in the **desk\_document\_check** method we use use the command **$documents hide** to remove the documents from the game, and this makes the method specific to exactly one document. So lets make the method a bit more geneal to make it work with any number of documents ( or any trigger that makes someting disappear, it does not have to be documents ):   |  | | --- | | document\_check:  while(1) { // forever  // Dont execute past this line  // until someone triggers the object  self waittill trigger  // parm.other is the triggerer  if(parm.other.dmteam == allies) {  // Make the document graphix disappear  self.target hide  // Tell the win method that an  // objective has been completed  level.targets\_destroyed ++ // ++ adds 1  break // out of while loop  }  waitframe // protection  }  end |   So?... Whats the difference? Well; not a lot... the line **$documents\_trigger waittill trigger** has been replaced with the line **self waittill trigger**, and the line **$documents hide** has been replaced with the line **self.target hide**.  So what is this **self** stuff? The self object can be set at the time the executing thread is created. Like this:   |  | | --- | | $documents\_trigger thread document\_check |   Creating a thread like this sets the self object to **$documents\_trigger**. This makes the trigger accessible in the thread without the thread knowing what name the object has ( $documents\_trigger ). So? Well: if you now add two more documents to the map, you just start a new thread for each of them, instead of writing a new method ( that will be almost identical ) for every document. Like this:   |  | | --- | | $button\_trigger thread document\_check  $horse\_trigger thread document\_check  $n00b\_trigger thread document\_check  $bad\_name\_trigger thread document\_check |   Just one more thing to make it work: the line **self.target hide** works because the trigger has been assigned a target that is the documents to hide when the trigger is activated ( if the object to hide has a targetname of **hi\_im\_bob**, then the trigger should have a target of **hi\_im\_bob** ).  Now you should have an idea of how to overrun your map with hords of the most cunning objectives. But dont do that. In a normal map, there should not be too many objectives ( unless you make up a new kind of objective where it does not hurt gameplay to acieve a lot of objectives ). |