6. Multiplayer

- Server/Client-Bedingungen abfragen
- ExecVM
- RemoteExec
- Tasks erstellen (JIP-proof)
- ! RESPAWNS !

Server/Client-Bedingungen abfragen

Eine Liste von "machine types" (singleplayer, server, client) und wie sie jeweils abgefragt werden können. Um an allen Ecken und Enden **Performance zu sparen**, macht es Sinn, möglichst viele Prozesse **nur dort berechnen zu lassen, wo sie wirklich laufen müssen**. Der Dedicated Server hat z.B. kein Interface, aber Achtung: Auch ein Client kann Server sein, hat dann also auch ein Interface! Will man zum Beispiel erfragen, ob die Recheneinheit ein Multiplayer-Host ist, aber KEIN Dedicated Server, muss man abfragen:

```
if (hasInterface) then {
```

und

```
if (isServer) then {...};
```

Zur Abfrage eines reinen Servers also:

```
if (isServer) then {...};
```

Beispiel:

```
if (isServer) then {v1 animate ["terc",1]};
```

Normalerweise muss man nur ein oder maximal zwei Rechnertypen abfragen. Als Missionsbauer ohne permanenten Zugriff auf einen Dedicated Server macht es beispielsweise keinen Sinn,

```
if (isDedicated) then {...};
```

abzufragen, auch wenn die Mission letztlich auf einem DS läuft. Es genügt eine reine Serverabfrage (siehe oben), damit der Prozess nur beim Host (ob Dedicated oder nicht) berechnet wird. Alles andere erschwert nur das das Debugging.

Link (siehe auch Kommentar von Killzone Kid!):
https://community.bistudio.com/wiki/isServer

Hier eine Liste für die Ausgabewerte aller drei Varianten:

EDITOR PREVIEW / SINGLEPLAYER:
isMultiplayer returns false
isServer returns true
isDedicated returns false

MULTIPLAYER (NON-DEDICATED) HOST SERVER

- isMultiplayer returns true
- isServer returns true
- isDedicated returns false

MULTIPLAYER DEDICATED SERVER

- isMultiplayer returns true
- isServer returns true
- isDedicated returns true

MULTIPLAYER CLIENT

- isMultiplayer returns true
- isServer returns false
- isDedicated returns false

ExecVM

ExecVM führt ein Script in gezeiteter Umgebung aus. Dies bedeutet dass das Script durch sleep etc. unterbrochen werden kann. Der Rückgabewert ist ein "handle" zum ausgeführten Script womit getestet werden kann ob es noch läuft oder schon durch ist.

```
_handle = execVM "scripts\test.sqf";
```

RemoteExec

RemoteExec

RemoteExec ist das mit Abstand beste Werkzeug, um sich durch den Arma-Multiplayer-Lokalitäten-Dschungel zu kämpfen.

Syntax (remoteExec):

```
params remoteExec [functionName, targets, JIP]
```

Um uns das Ganze von Anfang an ein wenig einfacher zu machen, nehmen wir den optionalen Parameter JIP gleich mal raus. Dieser ist Standardmäßig false und solltet ihr nicht genau wissen, was ihr tut, solltet ihr es auch dabei belassen. Daher:

```
params remoteExec [functionName, targets]
```

params: Bei Params handelt es sich um die Parameter des Befehls den ihr ausführen wollt. Bei Befehlen die selbst nur einen Parameter haben (z.B. “hint”) kommt hier einfach dieser Parameter hin. Bei Befehlen mit 2 Parametern (z.B. “addAction”) entsteht ein Array, wobei der Parameter, der vor dem Befehl steht, zuerst kommt und dann der Parameter, der hinter dem Befehl steht.

Beispiele:

1. hint “Das ist nicht ganz einfach”;

-> “Das ist nicht ganz einfach” remoteExec [“hint”];

1. KI addAction ["Sag Hallo", "scriptHallo.sqf"];

-> [KI, ["Sag Hallo", "scriptHallo.sqf"]] remoteExec [“addAction”];

Hierbei könnt ihr einfach Copy&Paste nutzen. Alles vor dem Befehl vor das Komma im Array, alles dahinter, hinter das Komma.

functionName: Dabei handelt es sich schlicht und ergreifend um den Befehl, den ihr ausführen wollt. Einzig anzumerken ist, dass er in Anführungszeichen stehen muss, da es sich um einen String handeln muss.

targets (optional): Mit targets beginnt die Magie dieses Befehls. Es handelt sich dabei um einen optionalen Parameter des remoteExec-Befehls, er ist daher nicht zwingend nötig. Sollte er fehlen, wird der Befehl schlicht global ausgeführt (sollte nur für Befehle mit lokalem Effekt ausgeführt werden).

Als target kann man ein einzelnes Objekt, eine Gruppe von Objekten, den Server (target = “2”) oder auch alle außer den Server wählen (target = “-2”). Das Hervorragende daran ist, dass dieses command genau da ausgeführt wird, wo das target lokal ist. Solltet ihr also beispielsweise einen LKW manipulieren wollen, der aber in der Mission von einem x-beliebigen Spieler gefahren wird (und daher nicht mehr lokal zum Server ist), könnt ihr als target diesen LKW angeben und der Befehl wird automatisch auf dem Client des Spielers ausgeführt zu dem der LKW lokal ist.

Was man bei target angeben will, hängt natürlich hauptsächlich davon ab, ob man einen Befehl mit lokalem Argument oder mit lokalem Effekt benutzt. Beim lokalen Argument wird das target meist das zu manipulierende Objekt sein. Bei lokalen Effekt kann man das target darauf zuschneiden, für wen der Effekt gelten soll.

Beispiele:

1. “Info an die ganze Gruppe” remoteExec [“hint”, (group S1)];

Zeigt diesen Hint für die ganze Gruppe des Spielers S1 an.

1. [LKW, 1] remoteExec [“setFuel”, LKW];

Tankt den LKW komplett voll.

1. "Hallo Freunde" remoteExec ["hint", (nearestObjects [Zivi, ["Man"], 50])];

Zeigt die Nachricht “Hallo Freunde” für jeden Spieler in einem 50m Radius um das Objekt “Zivi” an.

Tasks erstellen (JIP-proof)

https://forums.bohemia.net/forums/topic/167564-how-to-arma-3-editor-creating-a-jip-proof-coop-mp-mission-task-setup/

https://www.youtube.com/watch?v=Lv7f0NZ5esE

! RESPAWNS !

Kurzes Vorwort:

Respawns in Arma können auf unterschiedliche Arten und Weisen gesetzt werden, ich zeige euch hier meine nach dem "old-fashioned-way" in 3 Schritten. Das hat immer funktioniert und ermöglicht euch auch einen kleinen Einblick in das Scripting und dem Erstellen von zusätzlichen Dateien neben der mission.sqm. Ich werde die Schritte kurz fassen und am Ende des Eintrags eine detailliertere Erklärung der Auswirkungen schreiben, für die die es interessiert.

Damit das alles klappt, müsst ihr 1 Schritt im Windows Explorer und 2 Schritte im Eden Editor erledigen. Am besten macht ihr das parallel zueinander, speichert eure Mission nach den Änderungen im Explorer und startet mal durch um das zu testen.

!WICHTIG!

Ihr macht euch euer Leben um einiges leichter, wenn ihr euch von vorn herein Notepad++ runterladet. Dieses kann jede Arma 3 Datei bearbeiten, die ihr im Scripting braucht und verwendet!

Schritt 1: Mission erstellen und Respawn-Modul in der Liste rechts auswählen

- Lokalen MP-Server hosten: Arma 3 Hauptmenü->Multiplayer->Server hosten
- Einen Soldaten platzieren (für das Beispiel nehmen wir einen BLUFOR-Soldaten)
- Auf "Module" in der rechten Liste wechseln und unter "Multiplayer" die "Respawnposition" auswählen
- Setzt das Modul auf den Punkt, wo die Spieler respawnen sollen und macht einen doppelten Linksklick auf das Modul
- "Führende Seite" stellt ihr auf die Seite um, der eure Spieler angehören (hier ist es BLUFOR) und klickt auf OK
- Fertig

Schritt 2: Attribute anpassen

- In der oberen Leiste des Editors auf "Attribute" klicken und "Multiplayer" auswählen
- Im Reiter "Respawns" den Respawn von "ausgeschaltet" auf "An benutzerdefinierter Position" verändern
- Respawnverzögerung auf 10 Sekunden setzen (Unbedingt darauf achten, das Sekunden ganz rechts angezeigt werden! 10 Stunden Respawns sind nich so nice)

- In der Tabelle den Haken bei "Benutzerdefinierte Position" setzen
- Fenster schließen und Mission abspeichern
- Fertig

Schritt 3: Die description.ext

- Windows Explorer öffnen
- Dokumente->Arma 3 Other *Profiles->Name eures Arma 3 Profils->*mpmissions-> *Name eurer Mission*
- Im Ordner eurer Mission ein neues Textdokument erstellen und in **description.txt** abspeichern
- In euerer description.txt folgende Zeilen eingeben:

```
respawn = BASE;
respawn = BLUFOR;
respawnDelay = 10;
respawnOnStart = 0;
```

- Abspeichern und umbennen in **description.ext**
- **Hinweis:** Mit dem Texteditor oder dem WordPad lassen sich .ext, .sqf, .sqm usw. nicht mehr öffnen. Mit Notepad++ schon.

- Kehrt in eure Mission zurück, klickt auf Speichern innerhalb des Editors, damit die Änderungen wirksam werden und startet durch.

Erklärung:

respawn = BASE;

Es gibt mehrere Möglichkeiten die ihr eintragen könnt:

BASE -> benötigt ihr für benutzerdefinierte Respawns wie in unserem Beispiel

INSTANT-> Respawnt euch an Todesort

GROUP-> Habt ihr KI in eurem Trupp, respawned ihr als einer von denen

SIDE-> Gibt es KI auf der Seite des Spielers? Glückwunsch, das bist jetzt du!

BIRD-> Die über alles geliebte Seemöwe für die RICHTIGE Vogelperspektive

respawnDelay = 7;

Legt die Verzögerung als Timer fest, wenn respawned wurde. 10 Sekunden heißt 10 Sekunden warten bevor man wieder rein kann.

respawn = BLUFOR;

Ist eine Vorlage für festgelegte Respawns bestimmter Seiten. Alternativen sind OPFOR und INDFOR, je nachdem welcher Seite eure Spieler angehören, müsst ihr diese eintragen.

respawnOnStart =0;

Es gibt 2 Möglichkeiten:

1 -> Ihr werdet bei Missionsstart gezwungener Maßen respawned und könnt direkt einen Respawn auswählen

0 oder -1 -> Ihr startet wie gewohnt als das was ihr sein sollt an dem Ort an dem ihr sein sollt und respawned erst, wenn ihr sterbt oder den Button dafür drückt

Wer experimentieren möchte und sich mehr in die Materie einlesen will, kann den offiziellen Anleitungen von Bohemia Interactive hier folgen.