

CocoMUD client - Bug #136

Problem with macros during multiplay

10/09/2018 09:48 PM - Moritz Wolfart

Status:	Closed	% Done:	100%
Priority:	Normal		
Assignee:	Vincent Le Goff		
Category:	Customization		
Sprint/Milestone:	16		
Description			
Hey hey! I encountered a strange behaviour while multiplaying. I am using Macros for movement. 8 for north, 2 for south and so on. And if i am playing two characters in the same world, this macros behave in a strange way. I am not sure to which playing window the macros are sended. I think to the first one i logged in to the world. Hope i could explain the problem correctly. Best wishes! Moritz:-)			

Associated revisions

Revision ee1d7ad7 - 12/22/2018 11:56 AM - Vincent Le Goff

Fix #136: remove the dependency of the sharp engine on world, add it to sessions

History

#1 - 10/10/2018 07:13 AM - Vincent Le Goff

- Tracker changed from Feature to Bug
- Category set to Customization
- Assignee set to Vincent Le Goff

Ok, let me see if I have understood and can reproduce it. I will comeback to you with more questions if I'm unable to see the behavior.

#2 - 10/10/2018 07:39 AM - Vincent Le Goff

- Sprint/Milestone set to 16

#3 - 12/20/2018 08:39 PM - Vincent Le Goff

Okay here's the thing. A lot of CocoMUD is still world-oriented, versus session-oriented. Worlds were here before characters or sessions. What it means is that a lot of feature are still centered on a world, and common to a world, instead of being spread on several clients as it should.

In our case, different clients are indeed created. However, only one sharp engine per world is created. Here's a very simple test:

1. Open CocoMUD.
2. Open a world (connect to it).
3. Open the same world in a different tab of CocoMUD.
4. Open the Python console in this tab (menu **tools** -> **open Python console...**).
5. Type in the following:

```
(panel.client, panel.client.factory.sharp_engine.client)
```

6. Compare both numbers. Repeat these steps in the other tab.

What happens is quite simple:

1. CocoMUD connects the first time to the world.
2. It creates a sharp engine for this world.
3. It then connects another client to the same world.
4. It already has a sharp engine for this world, so both clients share the same sharp engine.

This explains the situation. Now, a quick fix isn't a simple matter: the best option would be to remove the sharp engine from the world, and place it in the session (see <source:src/session.py>) which is incidentally ready for the change as you can see. A client equals a session, this is the closest analogy we have, so I would think the sharp engine should be bound here.

It's not a coincidence, however, if worlds are bound to a sharp engine: most methods used by the sharp engine and its functions work on the world. Take `#macro` for instance which creates a macro in the world. When the world first loads, it reads the content of `config.set` file which contains the configuration specific to this world. This configuration shouldn't be read several times, just the first time the world is loaded.

#4 - 12/20/2018 10:46 PM - Francisco Del Roio

Vincent Le Goff wrote:

Okay here's the thing. A lot of CocoMUD is still world-oriented, versus session-oriented. Worlds were here before characters or sessions. What it means is that a lot of feature are still centered on a world, and common to a world, instead of being spread on several clients as it should.

In our case, different clients are indeed created. However, only one sharp engine per world is created. Here's a very simple test:

1. Open CocoMUD.
2. Open a world (connect to it).
3. Open the same world in a different tab of CocoMUD.
4. Open the Python console in this tab (menu **tools** -> **open Python console...**).
5. Type in the following:
[...]
6. Compare both numbers. Repeat these steps in the other tab.

What happens is quite simple:

1. CocoMUD connects the first time to the world.
2. It creates a sharp engine for this world.
3. It then connects another client to the same world.
4. It already has a sharp engine for this world, so both clients share the same sharp engine.

This explains the situation. Now, a quick fix isn't a simple matter: the best option would be to remove the sharp engine from the world, and place it in the session (see <source:src/session.py>) which is incidentally ready for the change as you can see. A client equals a session, this is the closest analogy we have, so I would think the sharp engine should be bound here.

It's not a coincidence, however, if worlds are bound to a sharp engine: most methods used by the sharp engine and its functions work on the world. Take `#macro` for instance which creates a macro in the world. When the world first loads, it reads the content of `config.set` file which contains the configuration specific to this world. This configuration shouldn't be read several times, just the first time the world is loaded.

Architectural problems...

I see. It would be tricky to solve this.

Well, we must apply this fix for this version. For the next, I will try to rework these parts to decouple all Sharp objects, including the engine.

Cheers,

#5 - 12/21/2018 08:31 AM - Vincent Le Goff

Not too bad, but it might not be simple. I'll do it today or tomorrow if you want. There are only two open issues in sprint [16](#) so next built should be published tomorrow.

There might be some keys to check/translate into Spanish if you feel like it:

- The key "tools" in <source:src/translations/es/ui/menu.yml> . I translated it by "Instrumentos" but it might not be the best.
- Just below it, the key "sharp_script_console" might not be a good translation either.
- The keys in the source:src/translations/es/ui/dialog/sharp_script_console.yml are not translated at all.

Thanks again for the contributions for this build!

#6 - 12/22/2018 11:57 AM - Vincent Le Goff

- Status changed from Open to Closed

- % Done changed from 0 to 100

Fixed in commit [ee1d7ad73ee0c09742c26209b221c5d746bd43e5](#).