Configuration

This tutorial covers the different basic configuration files and how you can finetune your RCSkills installation to your liking.
If you want to configure skills and effects or create professions and classes you should take a look at the respective configuration tutorials. But keep in mind that you should first configure this part of your RCSkills installation because it will affect how your skills, effects and professions behave.

Table Of Contents

Folder Layout

To keep things simple and in an ordered manner most of the configurations are located in sub folders. For the skills, effects and profession configs you can create your own subfolder trees and RCSkills will automatically scan thru the subfolder tree and detect all configuration files in it.
The folder layout below is based of the created plugin folder inside your plugins base folder.

`alias-configs/`
Contains variations of the available base skills created by you. See the Skills and Effects Tutorial for more information.
Supports recursive subfolder search.

`effect-configs/`
Contains all programmed effects and their default config. See the Skills and Effects Tutorial for more information.
You must not create your own configs in this folder!

`languages/`
Contains the different language files. You can create your own language file and it will take effect based on the chosen language in the Minecraft client.

`professions/`
Here you can create your different classes and professions. See the Professions and Classes Tutorial for more information.
Supports recursive subfolder search.

`skill-configs/`
Contains all programmed skills and their default config. See the Skills and Effects Tutorial for more information.
You must not create your own configs in this folder!

`skills-and-effects/`
Contains your skills and effect packages in a `.jar` format. You can get these packages from skill developers. You need to restart your server for the skills and effects to load.

Main Configuration Files

config.yml

This file contains a lot of important settings that touch the very core of RCSkills. Be careful when you change them and always test the changes before doing them on a live server if you dont know what exactly you are changing!
Each property is explained after the pound (#) sign.

Source Code

1. # Setting this to true will disable skills automatically if they run into an error while loading
2. disable-error-skills: false
3. # same as above but for abilities
4. disable-error-abilities: false
5. # after how many seconds should a player be removed from the cache

6. hero-cache-timeout: 300
7. # should players that logout be cached - setting this to false will ignore the setting above
8. cache-offline-players: false
9. # when firing a range projectile it queues a callback task. This defines after how many ticks this task will get cancelled.
10. # Settings this lower may cause ranged attacks to fail. A higher value may cause lags when using a lot projectiles, but may help when very long range attacks do not trigger skills.
11. callback-purge-ticks: 1200
12. # This defines the max amount of displayed hearts in the players GUI.
13. health-scale: 20.0
14. defaults:
15. # If you are using high health and damage values this should be set to true.
16. # For example setting this to true and defining the fall damage as 0.05 will always deal 5% of the players max health as fall damage.
17. environment-damage-in-percent: true
18. # do not change this unless you know exactly what you are doing!
19. effect-priority: 1.0
20. # Defines how long a player needs to wait in seconds in order to cast the next skill. This is not affected by the actual skill cooldown value.
21. global-cooldown: 1.5
22. # defines a default permission group (skill). You will need to set this when using RCSkills as a permissions handler.
23. permission-group: guest
24. # how much damage does a fist attack do?
25. fist-damage: 1.0
26. # defines the chance a sekelton has to knockback a player
27. skeleton-knockback: 0.75
28. # how long until a party inventation expires in seconds
29. party-timeout: 30.0
30. # Defines how long a player has to wait between two fist attacks in seconds
31. swing-time: 1.0
32. # defines how long players need to wait after a combat action until they can switch their pvp status back
33. pvp-combat-timeout: 300.0
34. # the radius in blocks for which party members gain exp
35. party-exp-range: 100
36. # how long until players can switch their pvp status when doing pvp actions
37. pvp-toggle-delay: 300.0
38. # how often should the server check for valid player entities
39. # dont change this unless you know what it does
40. character-invalidation-interval: 100
41. profession:
42. # the base cost for changing a profession
43. change-cost: 100
44. # multiplier for the profession level the player changes to.
45. # base cost + level-modifier * level = profession change cost
46. change-level-modifier: 25.0
47. interface:
48. # defines how often the interfaces is refreshed. Important changes are directly updated so chaning this shouldn't do much.
49. updateinterval: 20
50. hero:
51. # defines the maximal hero level a player can get
52. max-level: 100
53. # config paths can be adjusted here
54. # if you change these paths please keep them in mind for the other tutorials
55. paths:
56. skill-configs: skill-configs/
57. alias-configs: alias-configs/
58. skill-jars: skills-and-effects/

59. effect-configs: effect-configs/
60. effect-jars: skills-and-effects/
61. profession-configs: professions/
62. abilities-jars: abilities/
63. # if you have a multiworld or bungeecord environment (with the same db) you can ignore these worlds
64. # players will be able to use everything they have but their progress wont be saved
65. ignored-worlds:
66. - freebuild
67. - lobby
68. - event
69. # define what professions dont get maxed out when using the /rcs maxout <player> command
70. excluded-max-out-professions:
71. - virtual
72. # defines what effect types are displayed in the sidebar scoreboard
73. sidebar-effect-types:
74. - SYSTEM
75. - HARMFUL
76. - HELPFUL
77. - DAMAGING
78. - BUFF
79. - DEBUFF

Display All

damages.yml

In this file you define all of the different default custom damage values of mobs and entities as well as environment damage. All of the values are flat and must be specified as doubles. Please note that if you set `environment-damage-in-percent: true` in die `config.yml` that the last section will be percentage values. Setting a value of 1.0 there will always kill the player instantly.

experience.yml

In this config you specify the different experience values mobs and blocks give when killed or harvested. This will overriden by custom mobs that have a level. Also each skill has its own exp section and can be configured separatly from this file.

```
exp-rate: 1.0
skill-prof-exp-rate: 1.0
prof-hero-exp-rate: 1.0
```

The three values above can be adjusted if you want players to level faster.

levels.yml

Here you define different argument values for the programmed formulas and create new formula types that are useable in the profession and classes. You will need to play around a bit with these values when you start a new RCSkills server. But you shouldn't change existing values or else the level calculations and experience of existing players will be fucked up madly.
If you need to adjust them anyways you need to lock down your server, change the values and then run the converter script from the RCSkills - Tools section to recalculate the needed EXP and Levels for your existing players.

Formula Types

There are currently four different formula types, but it is planned to allow dynamic addition of new formulas. You can create as many forumla aliases from these types as you want and use them in skills and profession configs. The formulas always calculated the needed EXP for the given level.

- Linear Formula - `type: linear`
  `neededExp = x * level`

- Base Formula - `type: base`
  `neededExp = base + x * level`
- Static Formula - `type: static`
  `neededExp = amount`
- WoW Formula - `type: wow`
  `neededExp = (-0.4 * level^2) + (modifier * level^2)`

In the config you can define all non level parameters as config values and modify them. There are also three different sections you can define formulas in and every section has a default formula. See this `levels.yml` example config:

Source Code

```
professions:
default:
type: wow
modifier: 100
primary:
type: wow
modifier: 100
secundary:
type: wow
modifier: 220
beruf:
type: wow
modifier: 46.7
skills:
default:
type: mcmmo
base: 200
x: 225
heroes:
default:
type: wow
modifier: 146.5
```

Display All

paths.yml

In the `paths.yml` config you define different paths your players can take. For every path a player can only have one class and walking the tree up to the parent class. This means if you take the class path for example and you player chose the rogue, specialising into the assassin the player now has the rogue and assassin class. If he switches the class inside the path he will drop his current class and choose the new one. But the player can also have a profession from the second path.
By defining requirements in the different profession/class configurations (see Professions and Classes) you can create very complex path scenarios for your players.

Source Code

```
# unique path identifier
class:
# readable name for the path - is displayed in commands, ui, etc
name: Klassen
# the priority defines the calculations of health and attributes
priority: 10
# identifier of classes the player can start at
parents:
- warrior
- rogue
```

```
- mage
- priest
# defines if exp gained in combat go to this class
# also defines if ui elements are displayed/switched when entering/leaving combat
select-in-combat: true
select-out-of-combat: false
prof:
name: Beruf
priority: 5
parents:
- bergarbeiter
- bauer
- alchemist
select-in-combat: false
select-out-of-combat: true
```

Display All

Requirement Configs

When you configure Skills and Effects or Professions and Classes you may come across requirements. These requirements are used to fine tune what skills are unlocked when and what professions have what requirements. With this approach it is possible to create the craziest class and skill unlock scenarios. For example you could have a class that only gets unlocked once a certain Quest has been completed or an other class has reached the defined level and the player has a certain skill level.

To be continued...