Scheduling Task

To submit custom repeating, delayed, async tasks WDPE has implemented `WaterdogScheduler`.

Accessing the scheduler

Scheduler can be accessed from proxy instance using `ProxyServer.getScheduler()` or using `WaterdogScheduler.getInstance()`.

Scheduling new task

You have several options to execute task. Here are all methods:

- `scheduleAsync(Runnable task)`: Submit the task to task pool and execute it async (using executor service).
- `scheduleTask(Runnable task, boolean async)`: Submit the task and execute it. Executes task asynchronously if `async = true`.
- `scheduleDelayed(Runnable task, int delay, boolean async)`: Submit the task to queue and execute it after specified delay (in ticks, 1 tick = 50ms). If `async = true` task will be executed asynchronously.
- `scheduleDelayed(Runnable task, int delay)`: This is alias to `scheduleDelayed(Runnable task, int delay, boolean async)` where `async = false`.
- `scheduleRepeating(Runnable task, int period, boolean async)`: Schedule repeatiting task with given period (in ticks). If `async = true` task will be executed every time asynchronously. **Note that task may mot be executed on same thread!**
- `scheduleRepeating(Runnable task, int period)`: Alias to `scheduleRepeating(Runnable task, int period, boolean async)` where `async = false`.
- `scheduleDelayedRepeating(Runnable task, int delay, int period, boolean async)`: Schedule task which will start first time after given delay. Task will repeate in given period. If `async = true` task will be executed every time asynchronously.
- `scheduleDelayedRepeating(Runnable task, int delay, int period)`: Alias to `scheduleDelayedRepeating(Runnable task, int delay, int period, boolean async)` where `async = false`.

Every method return `TaskHandler` which can be used to change executing behaviour. To cancel the task, you can use `TaskHandler.cancel()`. You can also pass `Task` class which implements `Runnable` and contains `onRun(int currentTick)`, `onCancel()` methods. Function `onCancel()` will be

called once the task is completed or canceled.

When should I use async task?

We recommend to use asynchronous tasks when your task does not require to be completed after previously submited task. Multi threaded model comes with several benefits such as non-blocking executing. WaterdogPE fully supports multi threaded features and even uses it.
Even is executing task using non-async task will not be executed on main thread. We use single threaded executor which ticks and executes non-async tasks. Asynchronous tasks are executed using `ThreadPoolExecutor` which can execute tasks on any of the threas from the pool.

Revision #2

Created Fri, Jan 15, 2021 10:18 AM by Alemiz

Updated Sun, Oct 30, 2022 7:52 AM by Alemiz