Dokumentation zu TickTimerBsp

- bestehend aus den Klassen ServerBsp, Client1, Client2
- die Voraussetzungen waren: dass es 20 Ticks pro Sekunde gibt und vom Server Events ausgelöst werden können (verstehe ich als Senden, Empfangen und Aktualisieren der Spielsituation)

Dokumentation zur Klasse ServerBsp

- soll nicht die Funktion eines Servers, sondern die Funktion des TickTimers innerhalb, des Servers beispielhaft aufzeigen
- Attribute
  - tick: ist die Zählvariable der Ticks (20 Ticks pro sec) und wird von 0-20 gezählt wobei 0=20 wegen der Periodizität
- Methode main()
  - Attribute
    - delay: gibt die einmalige Verzögerung an mit der der Task, welcher periodisch ausgeführt wird, startet; bei delay=0 wird ohne Vezögerung gestartet
    - period: gibt die Periode an mit der der Task (zum Task gehört auch die Tickinkrementierung) ausgeführt wird; ist auf 50ms gesetzt, weil sich alle auf 20 Ticks/sec geeinigt haben.
    - timer: Der Timer der mit dem Server startet und so lange läuft wie der Server
  - Methoden:
    - scheduleAtFixedRate(): Methode der Klasse timer die einen TimerTask(), beschrieben durch die Funktion run(), alle 50ms (bzw. 1/20sec) startet/ausführt
    - run(): hier wird als erstes der Tick inkrementiert, danach wird eine beispielhafte Funktionalität aufgezeigt; die Anweisungen können hier auch durch „echte“ Funktionen, wie senden und empfangen ersetzt werden; mit jedem 2. tick wird überprüft ob bei den Klassen/Clienten 1 & 2 das Attribut boniDoppelteGeschwindigkeit auf true gesetzt ist (wird ja iwie vom Spielserver verwaltet); ist es der Fall wird sofort gesendet oder empfangen, sonst wird es alle vier ticks gemacht;
    - Die Funktionen Client1.hallo() sollen nur bespielhaft aufzeigen, dass es möglich ist denn TimerTask mit beliebig vielen Funktionen aus verschieden Klassen auszubauen um so alle notwendigen Schritte in der gegebenen Zeit zu durchlaufen;
  - Anmerkungen:
    - Lässt man das Programm laufen sind 15 Iterationen pro Sekunde erkennbar, wobei Client1 mit doppelter Geschwindigkeit abgearbeitet wird, weil sein Attribut boniDoppelteGeschwindigkeit auf True gesetzt ist;
    - lässt man den Teil zwischen Tickinkrementierung und dem Auskommenttieren Code raus und verwendent stattdessen den auskommenttieren Code, kann der Effekt der zufälligen Setzung von boniDoppelteGeschwindigkeit und die entsprechende Ausführungsgeschwindigkeit beobachtet werden
    - es kann alles angepasst werden
    - Anzahl der Clienten 2-4 kann separat implemetiert werden
    - lässt sich auch auf den Verwaltungsserver anwenden mit 1-n Spielservern
    - in der Funktion run(), oder einer dazu Äquivaltenten, sollten evtl. alle vier Clienten gehalten werden, auch wenn der 3.-4. evtl. nicht existieren, man

könnte dann ein if-statement einführen, dass erstmal prüft ob eine Klasse existiert und dann den Rest durchführen, falls es möglich ist