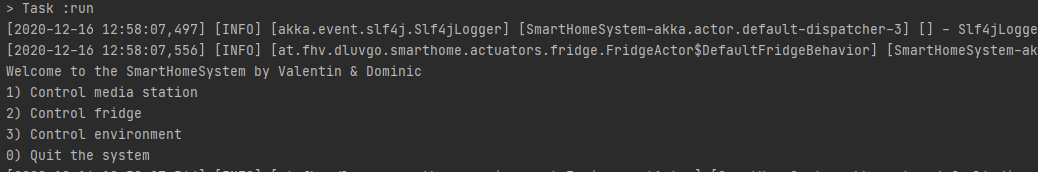
Dokumentation Smart Home

# Verwendung

gradle run

Man kann das Smart Home mit einer CLI bedienen:



Die verfügbaren Optionen werden im Log angezeigt und man kann direkt in die Konsole schreiben und ENTER drücken.

Es gibt verschiedene Submenüs, je nach gewählter Option.

# Design Entscheidungen

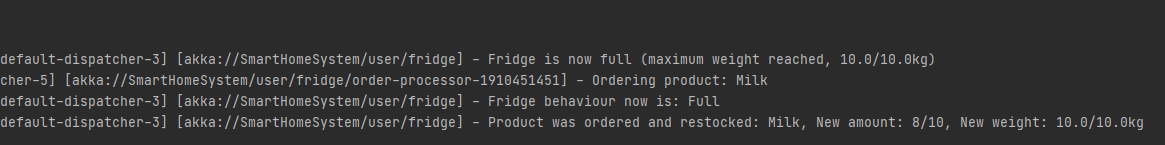
## Fridge

Wir haben uns bewusst gegen die Verwendung separater Aktoren für Anzahl/Gewicht im Fridge entschieden, da dies unserer Meinung nach nur wenige Vorteile bringen würde. Das hat folgende Gründe:

* Der Kühlschrank muss bei jeder Änderung beide Aktoren notifizieren (z.B. bei consume)
* Springender Punkt: Bei der nächsten Operation (z.B. bei order) braucht der Kühlschrank nun wieder die Sensordaten von beiden Aktoren -> mit request/response oder ask
  + Der Kühlschrank muss nun aber auf die Antwort beider Aktoren warten -> es muss synchronisiert werden, und eine Lösung mit while(true) warten ist für uns auch keine schöne Lösung
* Wir wissen grundsätzlich, wie wir weitere Aktoren spawnen können und Nachrichten austauschen, das sieht man anderen Aktoren

Dafür haben wir beim Fridge unterschiedliche Behaviors implementiert: DefaultFridgeBehavior und FullFridgeBehavior. Beide Behaviors unterstützen unterschiedliche Nachrichten, das bedeutet, ein FullFridgeBehavior kann z.B. keine weiteren Produkte bestellen.

Der Wechsel erfolgt, wenn der Kühlschrank voll ist, also entweder max\_weight oder max\_items erreicht ist. Es erfolgt eine Logausgabe („Fridge behavior now is: {}“):



Ansonsten haben wir überall sonst auf separate Aktoren gesetzt, das sieht man z.B. beim Environment und den Temperatur-/Wettersensoren gut.