TU Bergakademie Freiberg
Arbeitsgruppe Softwareentwicklung und Robotik
Prof. Dr. Sebastian Zug
M. Sc. Georg Jäger

Softwaretechnologie-Projekt (Prototyp)

Aufgabenblatt 2

Vorbereitungsaufgaben

1. C++ Syntax

Informiert Sie sich über die folgenden Elemente in C++:

- *static* Functions/Member Variablen
- *public, protected, private* Member einer Klasse
- Funktionszeiger und der Container *std::function<...>*
- Smart Pointer (*std::shared_ptr<...>, std::unique_ptr<...>, std::weak_ptr<...>*)

2. Design-Patterns

Welche Design-Pattern kennt ihr? Wie können diese beispielhaft in C++ umgesetzt werden? Informiert euch im speziellen zu folgenden Design-Pattern:

- Singleton:
  - Welche Vor- und Nachteile haben diese im Besonderen auf die Lebensdauer des Objektes?
- Publish/Subscribe:
  - Welche Möglichkeiten der Implementierung im Hinblick auf die benötigten Klassen gibt es in C++?
- Factory

3. Anti-Patterns

Informieren Sie sich über *Anti-Patterns*. Was ist ihre Bedeutung und wie können sie verhindert werden? Was besagt das *Golden Hammer*-Pattern?

Übungsaufgabe

In dieser Übung sollen die besprochenen Design-Pattern zur Strukturierung des Codes angewandt werden. Das Ziel bilden zwei Szenarien:

1. **Hello World!** - Implementiert einen Publisher, der den Text „Hello World!“ mit einer anschließenden Zahl (1, 2, 3, 4, ...) auf ein Topic schreibt und eine Subscriber, der den erhaltenen Text auf der Konsole ausgibt.

2. **Ping/Pong** - Implementiert ein Ping-Pong Szenario, bei dem ein Objekt immer dann „Ping“ auf einem Topic publiziert, wenn es auf einem anderen Topic „Pong“ erhält und umgekeht. Das Objekt, das „Ping“ publiziert, soll beginnen.

Implementieren Sie das *Publish/Subscribe* Design-Pattern. Definieren Sie dazu die Klassen des *Topic*, *Publishers* und *Subscribers*. Nutzen Sie eine weitere Klasse *Broker*, die als Singleton implementiert werden soll, als Vermittler zwischen *Publisher* und *Subscriber*.

Implementieren Sie zur vereinfachten Generierung von *Publisher* und *Subscriber* Objekten entsprechende Factory-Methoden innerhalb des Broker.

Hinweis: Nutzen Sie Templates und Vererbung um möglichst wenig redundanten Code zu produzieren.