Observer pattern

Introduction

This document is part of a series of documents I will create as part of my studies of Design patterns.
In these documents I will touch the topic of what I learned and I will further put it to practice giving by an as relevant possible example.
Furthermore at the end of my Semester, February 2018, I might revisit some of these documents to shed light on patterns that I was able to use during my projects.

What I learned

With the observer pattern I learned a way to loosely couple (observer) code to data(an observable) in a way so that we can decide at run time if we want to stay in the know of an observable or not.
This is done through the use of a simple interface, assigned with composition.
This allows us to subscribe and unsubscribe as many observers as we want. I further learned that whilst implementing the observer pattern pub/sub style is not hard, Java and other libraries offer an already pre-made implementation of the pattern.
However, using the standard java library has a significant downside because it uses inheritance rather than composition.

An example

An example where one could use the observer pattern would be a project of mine called 'Trees of life'. (Glow)
In trees of life a tree would have 12 'veins'(Led-strips) and 6 touchable hands, whenever one hand is touched the 2 connected veins should start pulsing.
The solution I applied back then was by hard coding 2 veins to 1 hand after which the hand would tell the vein to make a pulse because it is touched.
By applying the observer pattern and making each hand an observable and the veins an observer we could have applied much better OOP principles allowing for better modification in the future.

The code

https://github.com/RickVM/Design-patterns/tree/master/Observer%20pattern