Heartwave Design Patterns

State Pattern

In our HeartWave device, the behavior of the user's interaction changes based on the state of the device. To ensure our codebase was maintainable, we decided to implement the State pattern. As the State pattern suggests, we created an enum called DeviceState that represented the state of the device. The state of the device is stored within the Device class instance. The MainWindow class reads the state of the device to determine which screen should be displayed and updates the state of the device based on the actions the user performs. Additionally, we designed the device states to ensure that each state is only reachable from a single other state. This made implementing the logic for the back and menu buttons much easier as it was not necessary to maintain a history of the users navigation.

Singleton Pattern

The Heart Wave Device itself is a single entity. For that reason, the class responsible for encapsulating our device state (the Device class) is a singleton. The Device class has a single global point of access on MainWindow and acts as the only instance of the device throughout our system.

Observer Pattern

Qt's signals and slots are an implementation of the Observer pattern. In our application, the elements of the user interface act as subjects, and our MainWindow class acts as the observer.