Objective:

- Create a Circuit with a Power Unit and an LED
- Call execute method of circuit to power the led

Classes:

- LED
- Switch
- Resistor
- Power Unit
- Circuit
- Parallel Connection
- Series Connection

Note: Red marked components are less priority for release version 1.0.0

Common Properties:

- In this ecosystem almost everything is a component.
- A component can hold reference to one or multiple components.
- Each component has at least 2 pins/legs (signed or unsigned i.e: positive, ground, neutral).
- Legs/Pins are not components.
- There will be no connectors. Components will be connected to each other via reference.
- Each component may/may not have a unique id.

Individual properties of different classes:

Component:

- It's an abstract class / interface. Parent class for all components.

  Properties:

  - Id [Unique id] (optional)

- Name/Tag [String] (Optional)
- Child [List of Components] (optional)

Circuit:

- A Circuit unit can hold one/more components.
- It has an execute method which can issue change of the states of the components. There might be a validate method which goes through all the components and is called by execute().
- If there are any connections that might cause the circuit to malfunction, the execute method throws an error. For example the positive and negative pins of a power unit might be connected without any other component in between. This will cause the execute() function to throw an error.
- On calling execute, the circuit will find all Power units and give them a command to emit a positive value from the positive pin and negative value from the negative pin.
- Each component can listen to these streams sent by the power source. The component can also emit the received stream to any listeners listening to it.
- If both pins of a component receive the values emitted from the positive and negative pins of the same power source, it can be defined as connected. There can be a status/Rx variable named **connected** which will be updated every time a component receives a stream. It can be listened to and/or checked by some public method to know the current state of a component.