Clean Code

What are the clean code rules?

- **Design rules**

1. Keep configurable data at high levels.
2. Prefer polymorphism to if/else or switch/case.
3. Separate multi-threading code.
4. Prevent over-configurability.
5. Use dependency injection.
6. Follow Law of Demeter. A class should know only its direct dependencies

- **Names rules**

1. Choose descriptive and unambiguous names.
2. Make meaningful distinction.
3. Use pronounceable names.
4. Use searchable names.
5. Replace magic numbers with named constants.
6. Avoid encodings. Don't append prefixes or type information.

Example:

```
const n = 'John'
const l = 'Doe'
```

```
const lastNamePhysician = 'Doe'
const firstNamePhysician = 'John'
```

- **Functions rules**

1. Small.
2. Do one thing.
3. Use descriptive names.
4. Prefer fewer arguments.
5. Have no side effects.
6. Don't use flag arguments. Split method into several independent methods that can be called from the client without the flag.

Example:

```
function createAppointment(patientName, physicianName, date, time, location) {

}
```

```
function createAppointment (appointment) {

}
```

```
function createAppointment(appointment) {
  const date = appointment.date
  const patientList = getPatientList()
  const currentPatient = patientList.find(patient => {
    patient.id = appointment.patient.id
  })
  const currentPhysician = physicianList.find(physician => {
    physician.id = appointment.physician.id
  })
  const currentLocation = locationList.find(location => {
    location.id = appointment.location.id
  })
  sendToCreateAppointment(currentPatient, currentPhysician, currentLocation)
}
```

```
function getPatientInformation(patientId) {
  const patient = patientList.find(patient => {
    patientId = patient.id
  })

  return (patient)
}
```

- **Comments rules**

1. Always try to explain yourself in code.
2. Don't be redundant.
3. Don't add obvious noise.
4. Don't use closing brace comments.
5. Don't comment out code. Just remove.
6. Use as explanation of intent.
7. Use as clarification of code.

8. Use as warning of consequences.