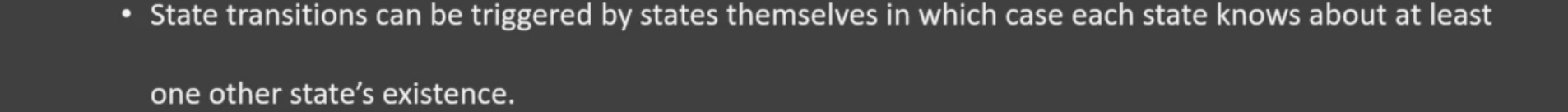
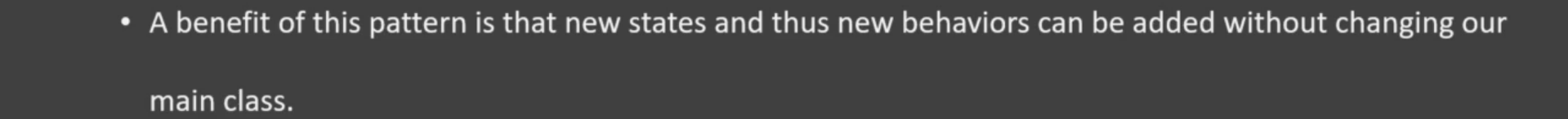
1. Graphical user interface, application

   Description automatically generated
2. This is very interesting design pattern.   
   Once you learn, you will have completely new way of extending your existing implementation.
3. Graphical user interface, text, website

   Description automatically generated
   1. You may be wondering that aren’t this very generic statement for all object as we know object are supposed to behave as per their internal state.
   2. But the difference in this pattern is that 🡺 In our regular class, we have if-else or switch statements to check the internal state of an object and object behaves accordingly.  
      But we’re going to implement that state specific behavior in a separate class.
4. **So, how we’re going to implement this?**
   1. Suppose we have a requirement that our state has two values A, B.
   2. Then we’re going to create two different class one for value A, and another for value B.
   3. Then we will have one object for each of these two classes in our original object.
   4. 
5. **State Transition**:
   1. How we should move from state one to state two, that is called **State Transition.**Itcan be triggered by the state themselves.  
      So a states will decide based on some input that they get when an operation is called on them, that it is time to switch to a different state.
   2. 
6. **Benefits**:
   1. 
   2. We can add new state thus new behavior for an object without changing our main class.  
      So, we will be creating new classes which will represent behavior specific to a new state.
7. Text

   Description automatically generated
8. **Let’s see the UML of this design pattern**.
9. A screenshot of a computer

   Description automatically generated with medium confidence
   1. **Class Context**:
      1. This is the class which a client interacts with.  
         This class provides the functionalities which a client expects for.
      2. Let’s take an object of Context Class for understanding purpose called context object.
      3. This object contains **state object** & the state object defines the actual behavior when Context.**operation()** is called.
   2. The concrete classes of State represent values that our state can take.