**Strategy Design Pattern**

**What is Strategy Pattern?**

Strategy design pattern is one of the behavioral design patterns. It defines a set of algorithms, provides an interface for accessing these algorithms, and designs them as interchangeable while defining algorithms. In addition, the client can choose any algorithm at runtime.

**When we should use**

It is used when we have more than one algorithm(methods) for a particular job.

**How to use?**

First, Strategy an algorithm interface is created. Then Concrete algorithm classes are created to implement the created interface. The Context class is created. This class contains a method for setting algorithms of the algorithm interface type. With this method, the Context class holds the algorithm object. Finally, a Client class is created. This class produces objects from the Context class and algorithm classes, allowing operations to be performed.

**What are the Benefits?**

At runtime, we can choose any algorithms defined for a particular operation.

**Sample Application**

*Scenario:* Let's make an application that helps us to communicate. Let there be two different types of communication in it;

* SMS
* E-Mail

**/\*\***

**\* Strategy (algorithm) interface**

**\*/**

**public interface Communicate {**

**public void send();**

**}**

**/\*\***

**\* Then Concrete algorithm class**

**\*/**

**public class Email implements Communicate {**

**@Override**

**public void send() {**

**System.out.println("Email is send");**

**}**

**}**

**/\*\***

**\* Then Concrete algorithm class**

**\*/**

**public class Sms implements Communicate {**

**@Override**

**public void send() {**

**System.out.println("Sms is send");**

**}**

**}**

**/\*\***

**\* Context class**

**\*/**

**public class Context {**

**private Communicate communicate;**

**public void setCommunicate(Communicate communicate) {**

**this.communicate = communicate;**

**}**

**public void sendInformation(){**

**communicate.send();**

**}**

**}**

**/\*\***

**\* Client class**

**\*/**

**public class Test {**

**public static void main(String[] args) {**

**Context context=new Context();**

**context.setCommunicate(new Email());**

**context.sendInformation();**

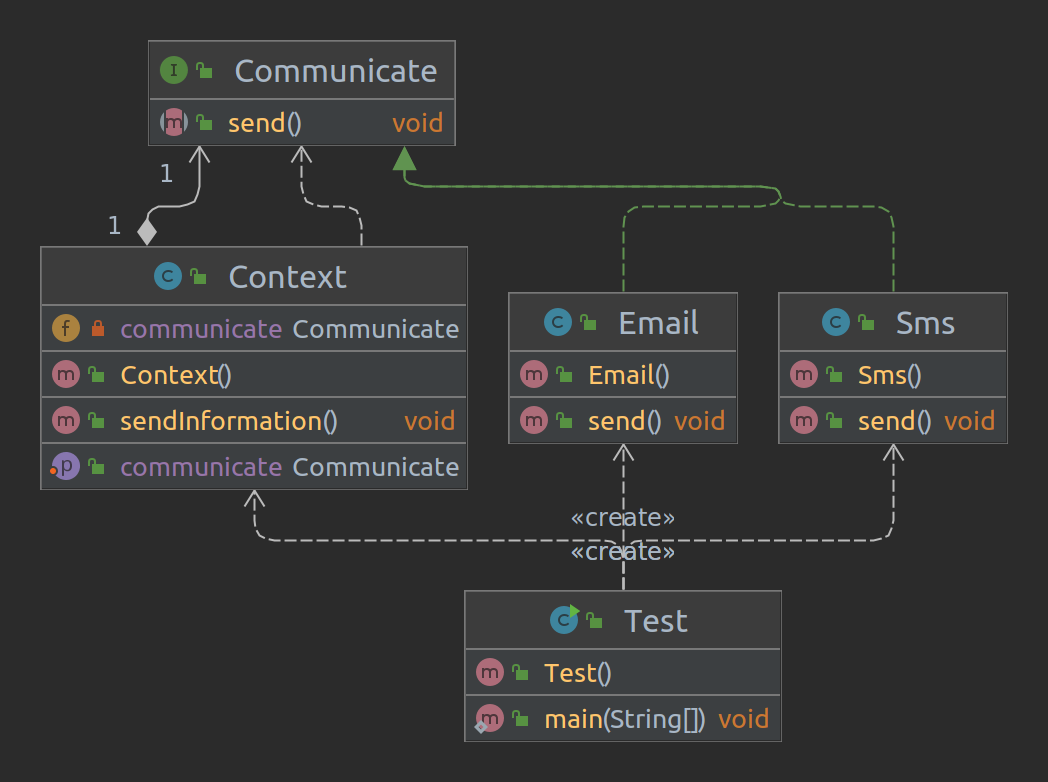
**context.setCommunicate(new Sms());**

**context.sendInformation();**

**}**

**}**

**UML Diagram of Our App**

****