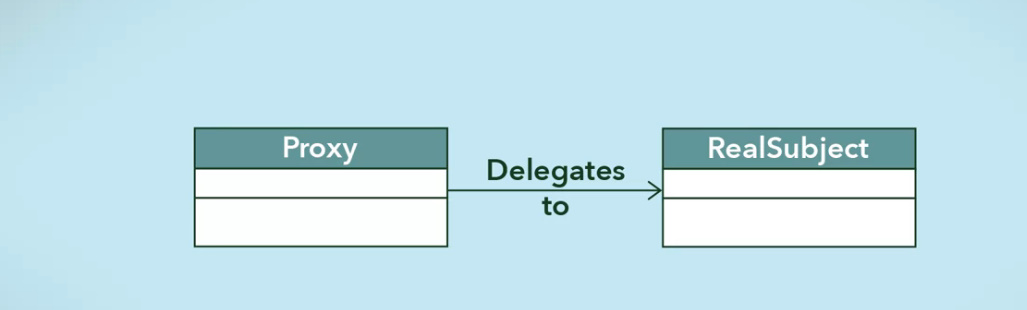
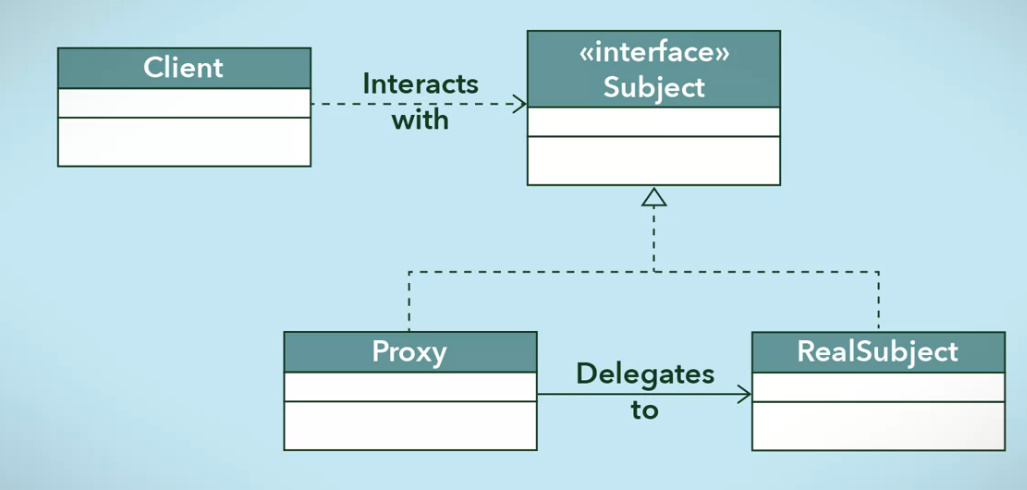
* In lumea reala, uneori e mai bine si usor sa folosim inlocuitori sau substituienti. De ex, cand se fac simulari cu accidente, se folosesc roboti in loc de oameni. Cand o firma are vreo prezentare, se foloseste vreun speaker in loc de seful firmei, ca reprezentant
* Toti acesti inlocuitori se numesc Proxy
* **Proxy** – se bazeaza pe a crea un obiect ce il controleaza pe altul si actioneaza in locul lui, putand si sa il foloseasca.
* Proxy este o versiune simplificata a obiectului original. Un Proxy object e capabil sa faca acelasi task ca obiectul original, si poate chiar sa il foloseasca pentru a le face.
* In Software, uneori e mai bine sa folosim un Proxy pentru un obiect decat obiectul original.
* Proxy mereu are in el o referinta la un obiect concret pe care il reprezinta.
* E mai securizat sa lucram cu un proxy decat direct cu obiectul original
* proxy poate securiza obiectul orginal, prin a face verificari inainte de a accesa ceva din el.
* Scenarii in care folosim un Proxy:

1. Pentru a se comporta ca un **virtual Proxy**. Instantierea unor obiecte poate fi costisitoare, si utilizarea la proxy in locul lor poate fi mai comoda. De ex, la utilizarea editoarelor de imagini, incarcarea la toate imaginile deodata ar face ca PC sa dea freez. Proxy va crea obiectul in el cand va fi necesar.
2. Pentru a se comporta ca un **proection Proxy** – pentru a limita si controla accesul la obiectul original.
3. Pentru a se comporta ca un **remote proxy** – cand clasa proxy e locala si clasa obiectului original e remote

**Structura la Proxy class**



Un proxy, adesea, foloseste o instanta unui obiect original, desi nu mereu. Uneori pur si simplu ii inlocuieste tot lucrul si nici nu face vreo referinta la el.

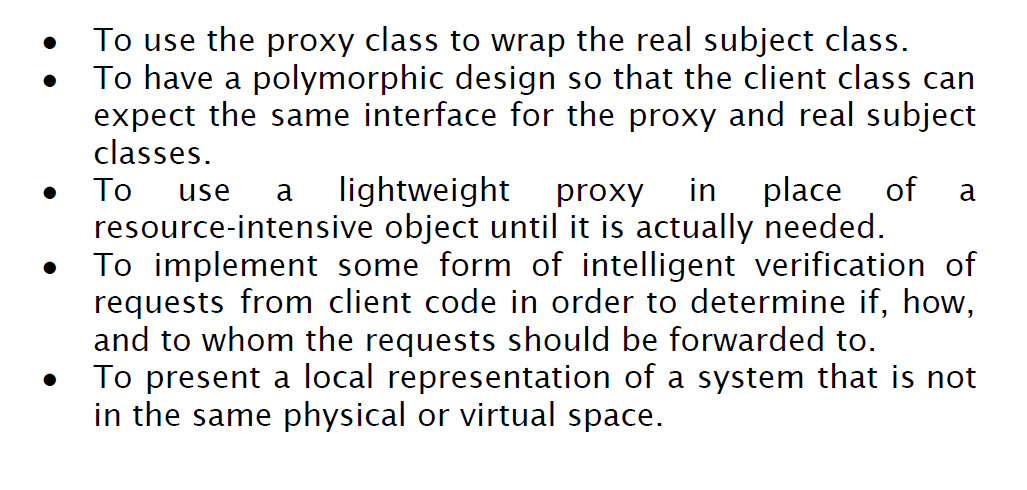


Proxy trebuie neaparat sa ofere aceleasi metode ca obiectul original.

Deci, proxy implementeaza interfata obiectului original si are o instanta de a lui ca field.

Proxy trebuie sa se comporte ca obiectul original, asa ca clientul nici sa nu stie ca asta e proxy.

**Rolul la Proxy**



* **When the original object is present in different address space, and we want to represent it locally**. We can create a proxy which does all the necessary boilerplate stuff like creating and maintaining the connection, encoding, decoding, etc., while the client accesses it as it was present in their local address space. This is called the Remote Proxy

**Exemplu**

public interface CommandExecutor {

public void runCommand(String cmd) throws Exception;

}

public class CommandExecutorImpl implements CommandExecutor {

@Override

public void runCommand(String cmd) throws IOException {

//some heavy implementation

Runtime.getRuntime().exec(cmd);

System.out.println("'" + cmd + "' command executed.");

}

}

public class CommandExecutorProxy implements CommandExecutor {

private boolean isAdmin;

private CommandExecutor executor;

public CommandExecutorProxy(String user, String pwd){

if("Pankaj".equals(user) && "J@urnalD$v".equals(pwd)) isAdmin=true;

executor = new CommandExecutorImpl();

}

@Override

public void runCommand(String cmd) throws Exception {

if(isAdmin){

executor.runCommand(cmd);

}else{

if(cmd.trim().startsWith("rm")){

throw new Exception("rm command is not allowed for non-admin users.");

}else{

executor.runCommand(cmd);

}

}

}

}