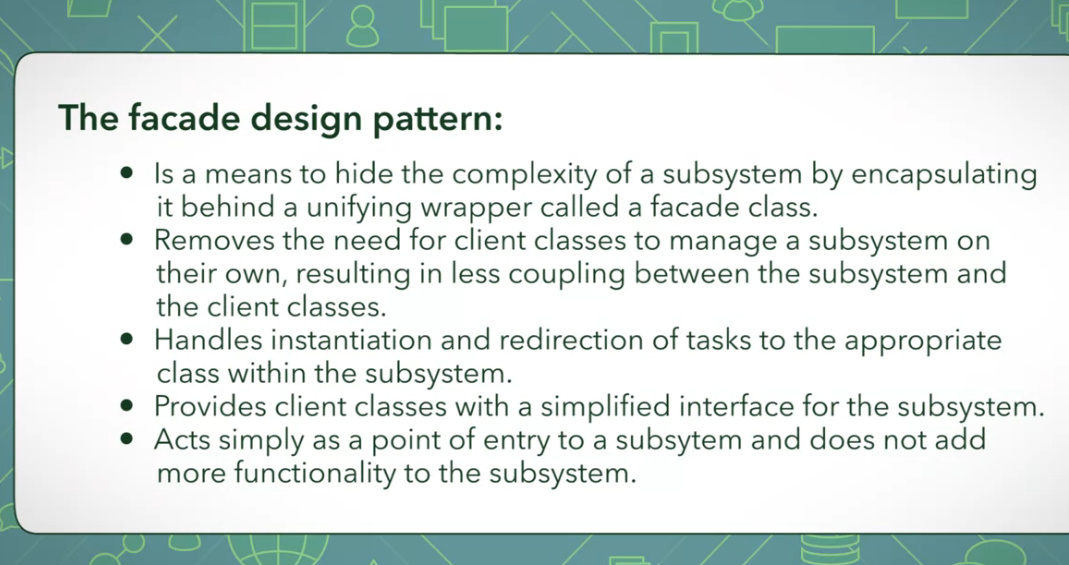
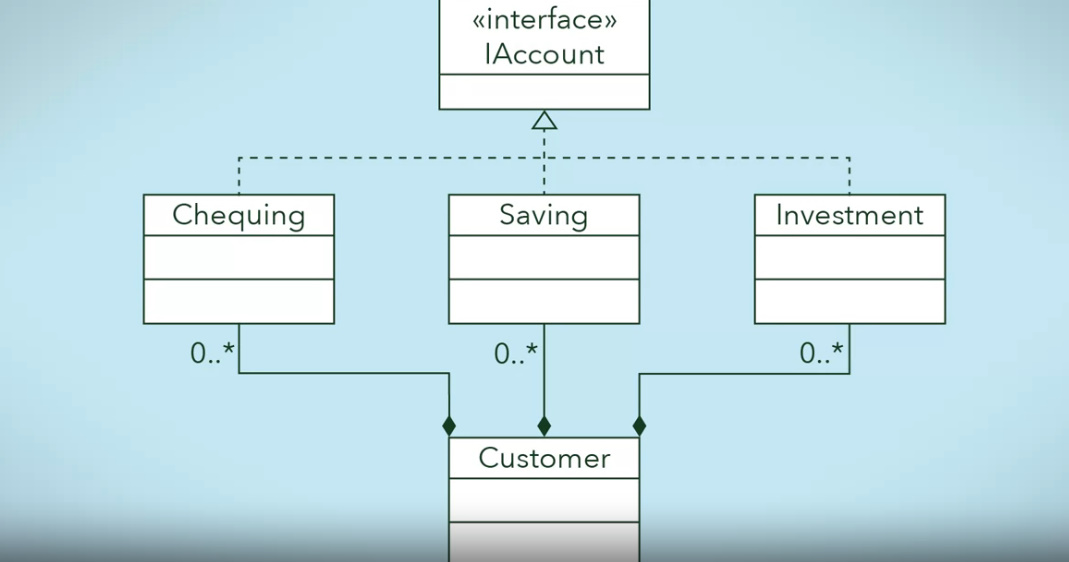
* Facade se traduce ca partea din fata a unei cladiri. Cand oamenii trec pe langa o cladire, ei vad doar partea ei in afara, doar exteriorul, nu vad cum e ea construita, nu vad pilonii,tevile etc. Facada ascunde doar detaliile complexe despre cladire si arata doar o parte friendly din ea.
* **Facade** – se bazeaza pe a ascunde complexitatea unui sistem si a oferi o interfata simpla clientului(altor clase) cu care poate accesa sistemul. **Facade ofera o interfata pentru a lucra cu un set de intefete, oferind metode mai simple de a lucra cu ele.**
* Facade mereu lucreaza cu una sau mai multe interfete, nu cu implementari. Ea lucreaza cu mai multe interfete pentru a scapa userul de a lucra cu ele, ci pentru a putea lucra cu una singura, ce deja lucreaza cu mai multe.
* Facade face ca interactiunea cu sistemul sa fie mai usoara, si atat, ascuzand complexitatea si deci implementarea sa. Nu are un model de code exact. Clasele viitoare for folosi facada, ca sa poata interactiona mai usor cu alte bucati din subsytem.

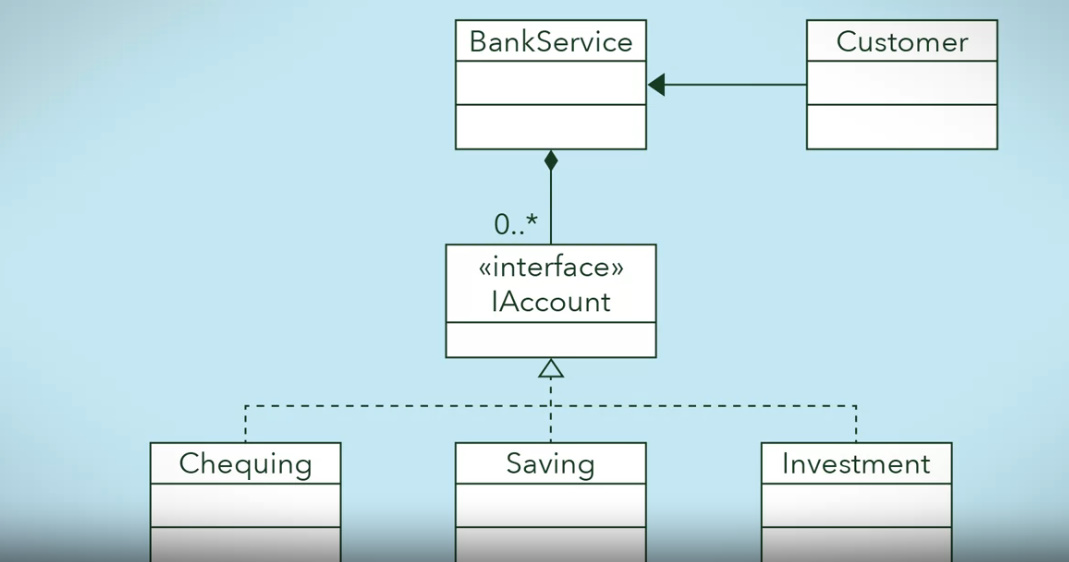


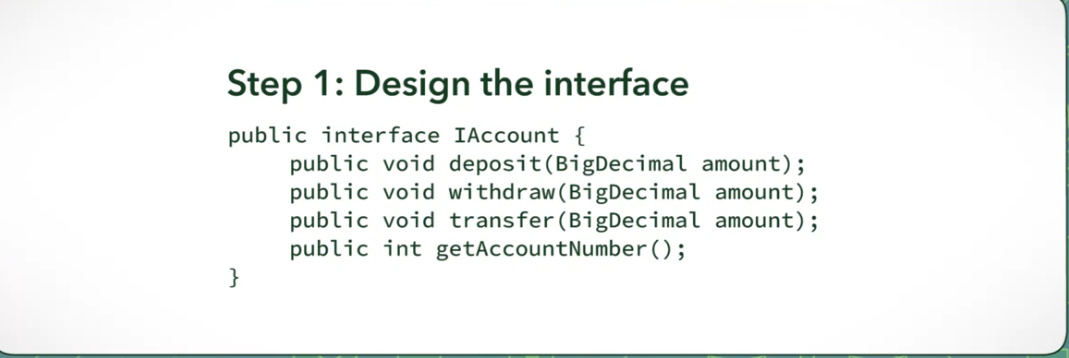
**Exemplu1**

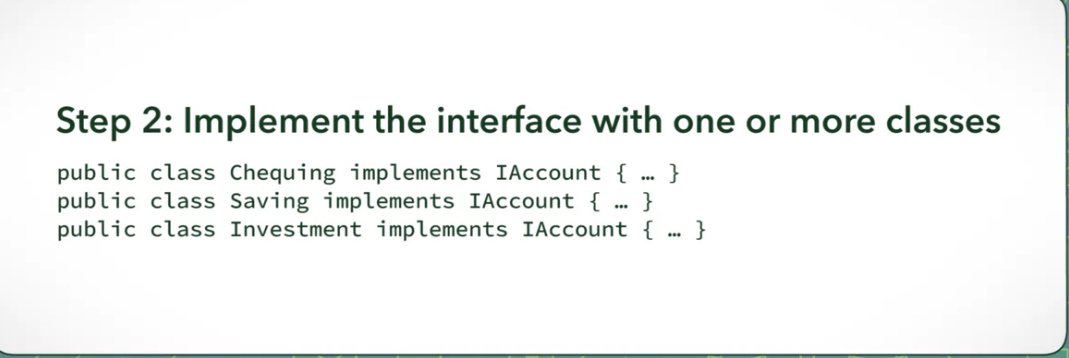


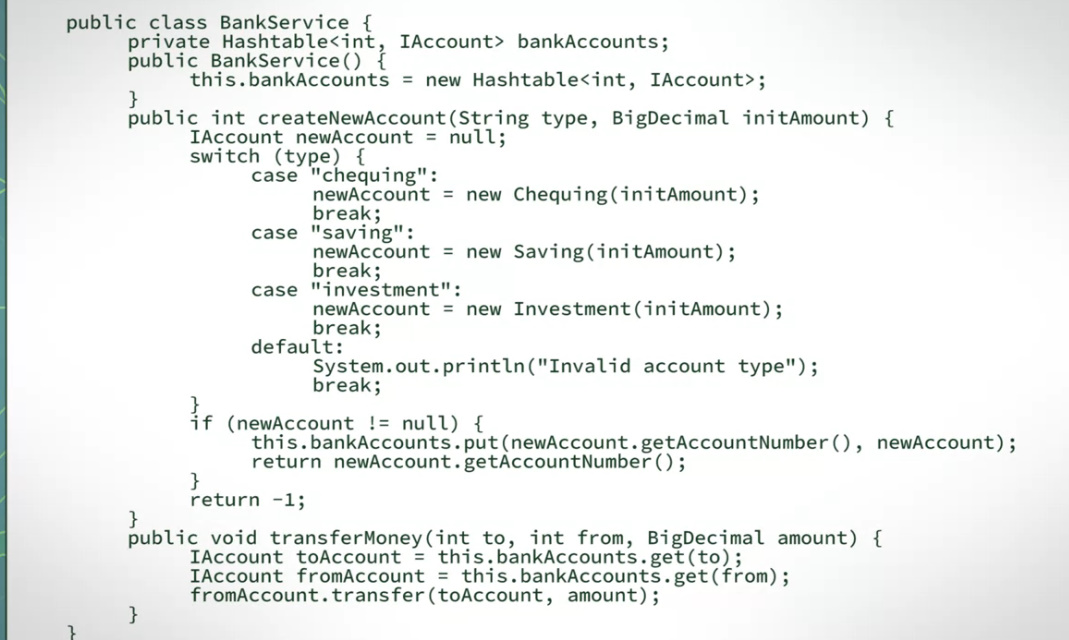
Daca nu avem Facada, ca in schema de mai sus, Customer clasa trebuie sa se ocupe de crearea si interactiunea directa cu clasele Chequing, Saving, Investment si asa are prea multe responsabilitati si grele. In plus, Customer mai stie despre toate metodele lor.

Dar, mai bine introducem un BankService, care e Facada, pentru a interactiona cu aceste clase, si Customer interactioneaza cu ele prin BankService

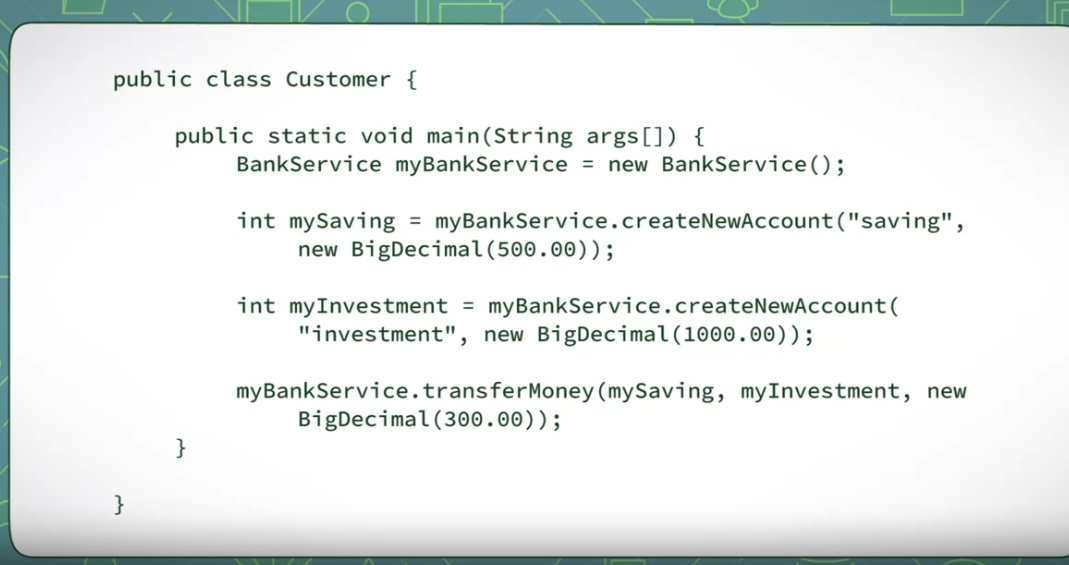








Aici Facada are o interfata, dar poate lucra si cu mai multe.



**EXEMPLU2**

public interface Shape {

void draw();

}

ublic class Rectangle implements Shape {

@Override

public void draw() {

System.out.println("Rectangle::draw()");

}

}

public class Square implements Shape {

@Override

public void draw() {

System.out.println("Square::draw()");

}

}

public class Circle implements Shape {

@Override

public void draw() {

System.out.println("Circle::draw()");

}

}

Avem o interfata si 3 implementari, desi am putea avea si mai multe interfete.

Anume aceasta e Facada:

public class ShapeMaker {

private Shape circle;

private Shape rectangle;

private Shape square;

public ShapeMaker() {

circle = new Circle();

rectangle = new Rectangle();

square = new Square();

}

public void drawCircle(){

circle.draw();

}

public void drawRectangle(){

rectangle.draw();

}

public void drawSquare(){

square.draw();

}

}

Lucram cu mai multe elemente din sistem, in cazul nostru cu mai multe clase, folosind o interfata, prin intermediul unei singure clase Facade, adica ShapeMaker

* Un alt exemplu, din lumea reala, e facada la un webstore. Noi facem comanda, dar nu interactionam direct cu sistemul. Facada o face, si noi o facem tot prin facada deci.