

# 13. Шаблон Фасада (Façade)

---

ЛЕКЦИОНЕН КУРС: ШАБЛОНИ ЗА ПРОЕКТИРАНЕ

ДОЦ. Д-Р ЕМИЛ ДОЙЧЕВ

# Общи сведения

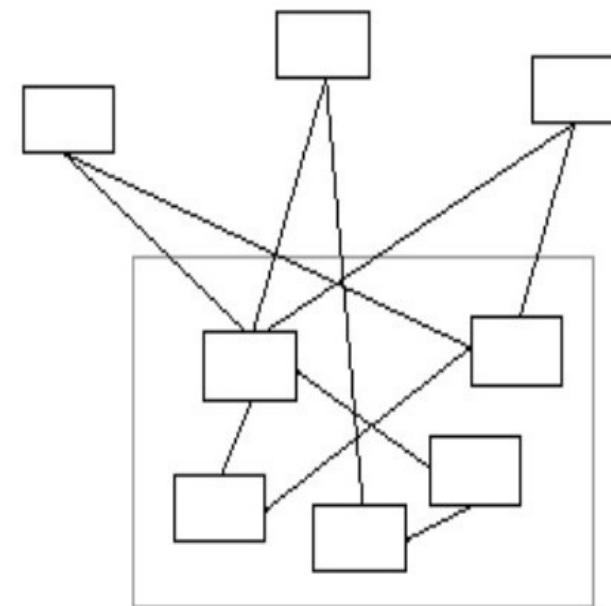
---

- ✓ **Вид:** Структурен за обекти
- ✓ **Цел:** Предоставя унифициран интерфейс към набор от интерфейси в една подсистема. Шаблонът Фасада дефинира интерфейс от високо ниво, който прави подсистемата лесна за използване.

# Мотивация

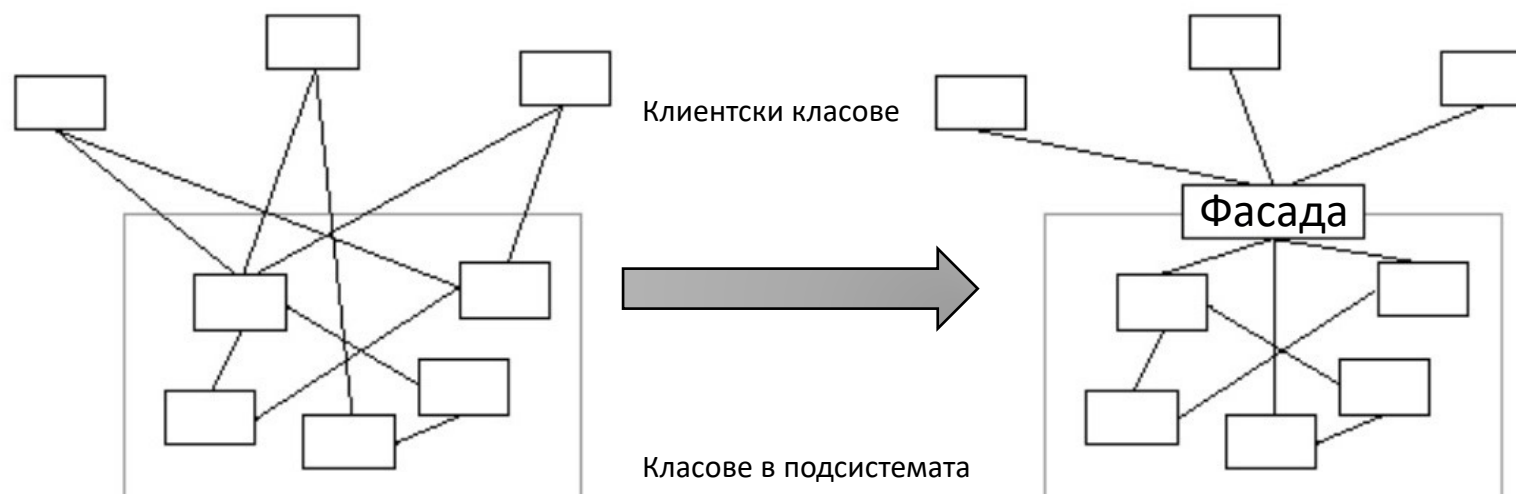
---

- ✓ Структурирането на една система в подсистеми спомага за намаляване на сложността ѝ.
- ✓ Подсистемите представляват набор от класове и евентуално други подсистеми.
- ✓ Интерфейсът на класовете, достъпни в подсистемата или набора от подсистеми може да стане прекалено сложен.



# Мотивация

- ✓ Един от начините за намаляване на комплексността е чрез въвеждане на фасаден обект, който предоставя един опростен интерфейс към функционалността на подсистемата.

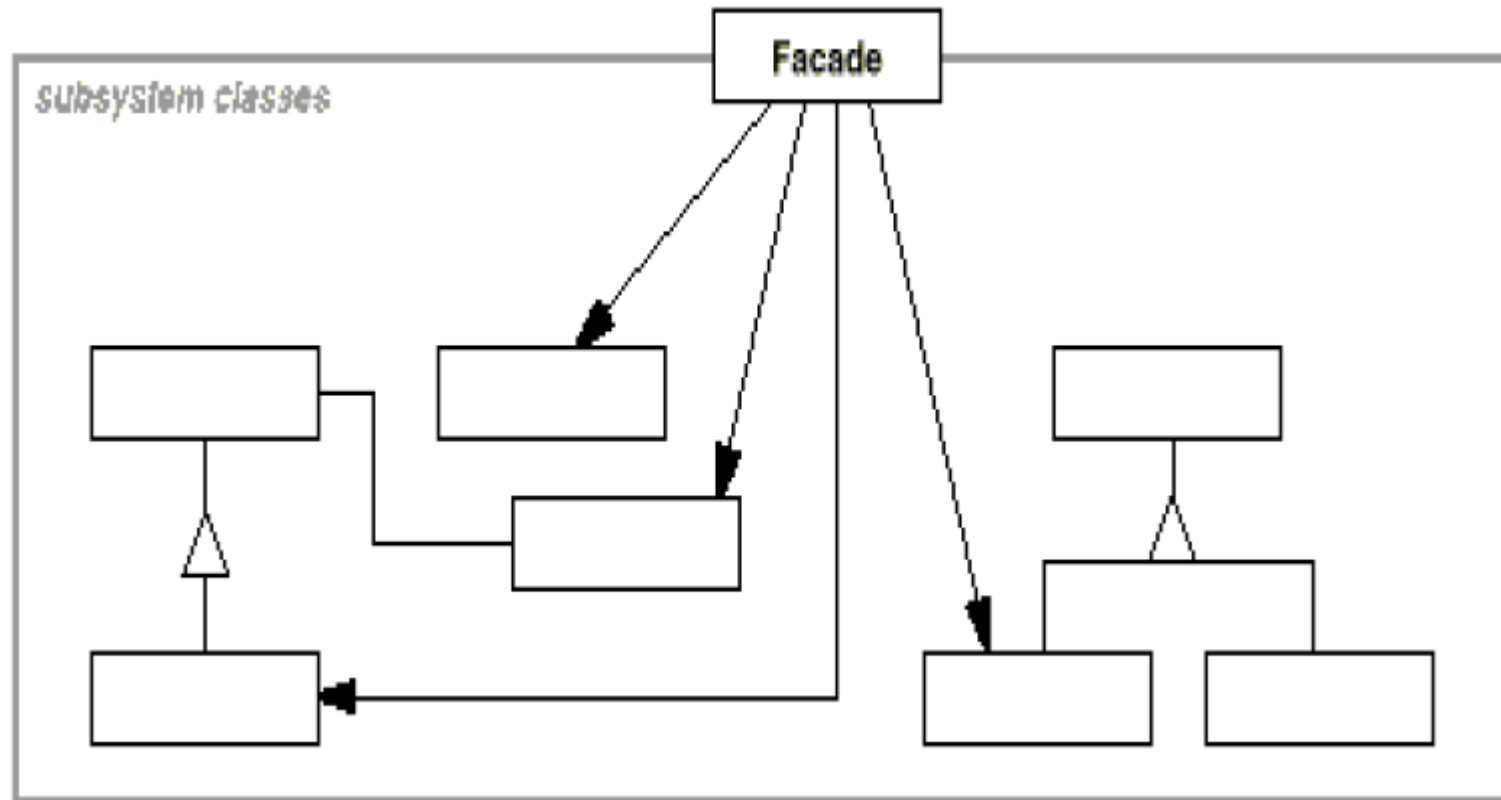


# Приложимост

---

- ✓ **Приложимост:** Шаблонът Фасада се използва в следните случаи:
  - Да се предостави прост интерфейс към сложна подсистема. Този интерфейс е достатъчен за повечето клиенти. Останалите клиенти, които имат нужда от по-сложна комуникация могат да заобиколят фасадата.
  - Да се отделят класовете на подсистемата от тези на клиентите и другите подсистеми. По този начин се постига независимост и преносимост на подсистемата.

# Структура



# Участници

---

## ✓ Фасада

- Знае кои класове на подсистемата отговарят за дадена заявка.
- Препраща клиентските заявки към съответните обекти на подсистемата.

## ✓ Класове на подсистемата

- Имплементират функционалността на подсистемата.
- Извършват работа, зададена от обекта фасада.
- Не знаят нищо за фасадата – т.е. не пазят референции към нея.

# Следствия

---

## ✓ Предимства

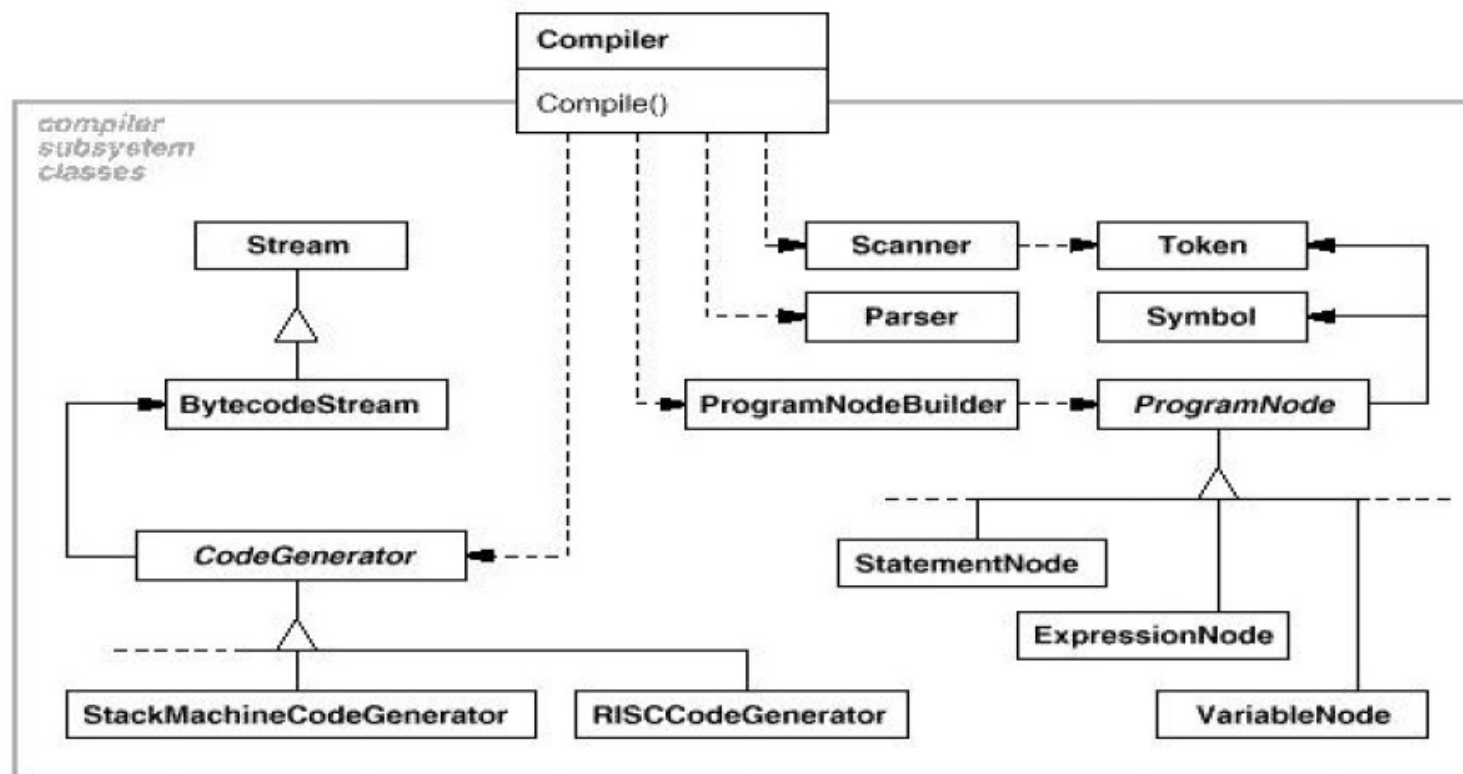
- Скрива имплементацията на подсистемата от клиентите, като така прави подсистемата по-лесна за използване.
- Постига „слабо свързване“ между подсистемата и клиентите. Това позволява промяна в подсистемата без да се засягат клиентите.
- Намалява компилационните зависимости в големи софтуерни системи.
- Не забранява на клиентите с по-сложни изисквания да достъпват директно класовете на подсистемата.
- Като забележка: фасадата не добавя функционалност – само опростява интерфейси.

## ✓ Недостатъци

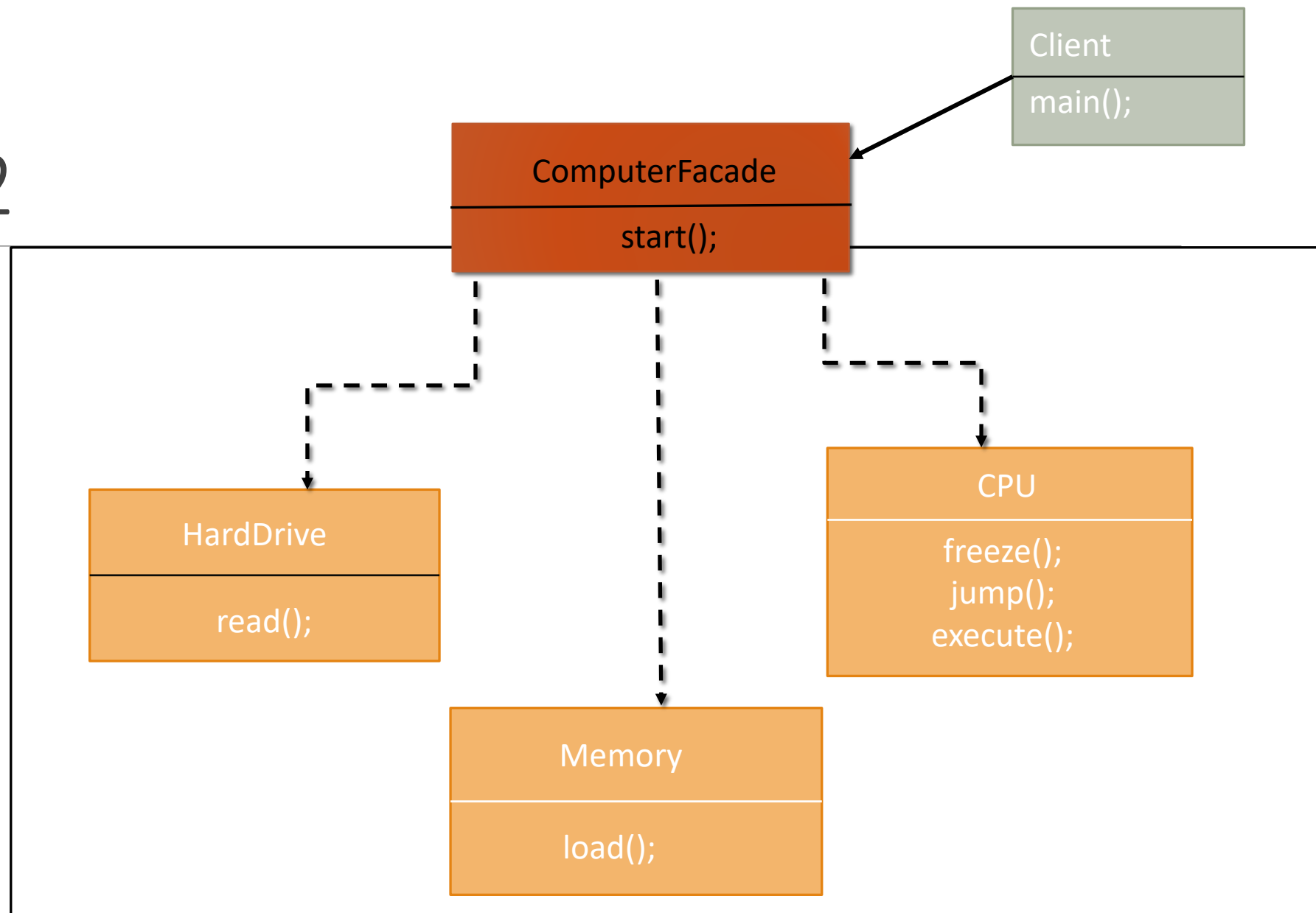
- Не забранява на клиентите да достъпват директно класовете на подсистемата.



# Пример 1



## Пример 2



# Пример 2

---

```
/* Сложни класове */  
  
class CPU {  
    public void freeze() { ... }  
    public void jump(long position) { ... }  
    public void execute() { ... }  
}  
  
class Memory {  
    public void load(long position, byte[] data) { ... }  
}  
  
class HardDrive {  
    public byte[] read(long lba, int size) { ... }  
}
```

# Пример 2

---

```
/* Фасада */
```

```
class ComputerFacade {  
    private CPU processor;  
    private Memory ram;  
    private HardDrive hd;  
  
    public ComputerFacade() {  
        this.processor = new CPU();  
        this.ram = new Memory();  
        this.hd = new HardDrive();  
    }  
}
```

```
    public void start() {  
        processor.freeze();  
  
        ram.load(BOOT_ADDRESS, hd.read(BOOT_SECTOR, SECTOR_SIZE));  
  
        processor.jump(BOOT_ADDRESS);  
  
        processor.execute();  
    }  
}
```

# Пример 2

---

```
/* Клиент */
```

```
class You {  
    public static void main(String[] args) {  
        ComputerFacade computer = new ComputerFacade();  
        computer.start();  
    }  
}
```

# Пример 3

---

- В съпътстващия проект

# Край: Шаблон Фасада

---

ЛЕКЦИОНЕН КУРС: ШАБЛОНИ ЗА ПРОЕКТИРАНЕ