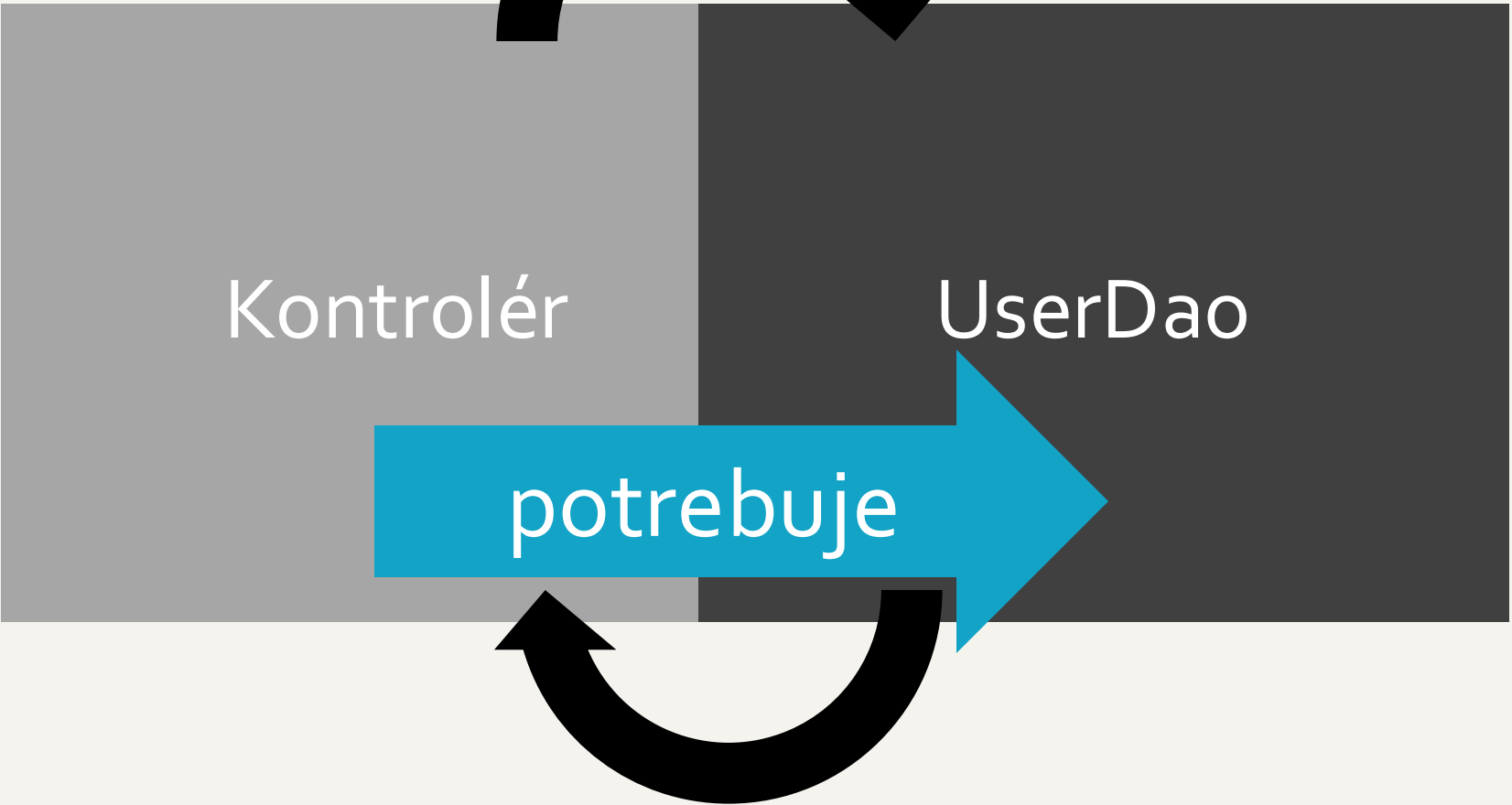


Kontrakty,
Továrne,
Jedináčikovia,
Vymenované typy,
Docker

UINF/PAZ_{1c}
4. prednáška



prepojenie
GUI a DAO



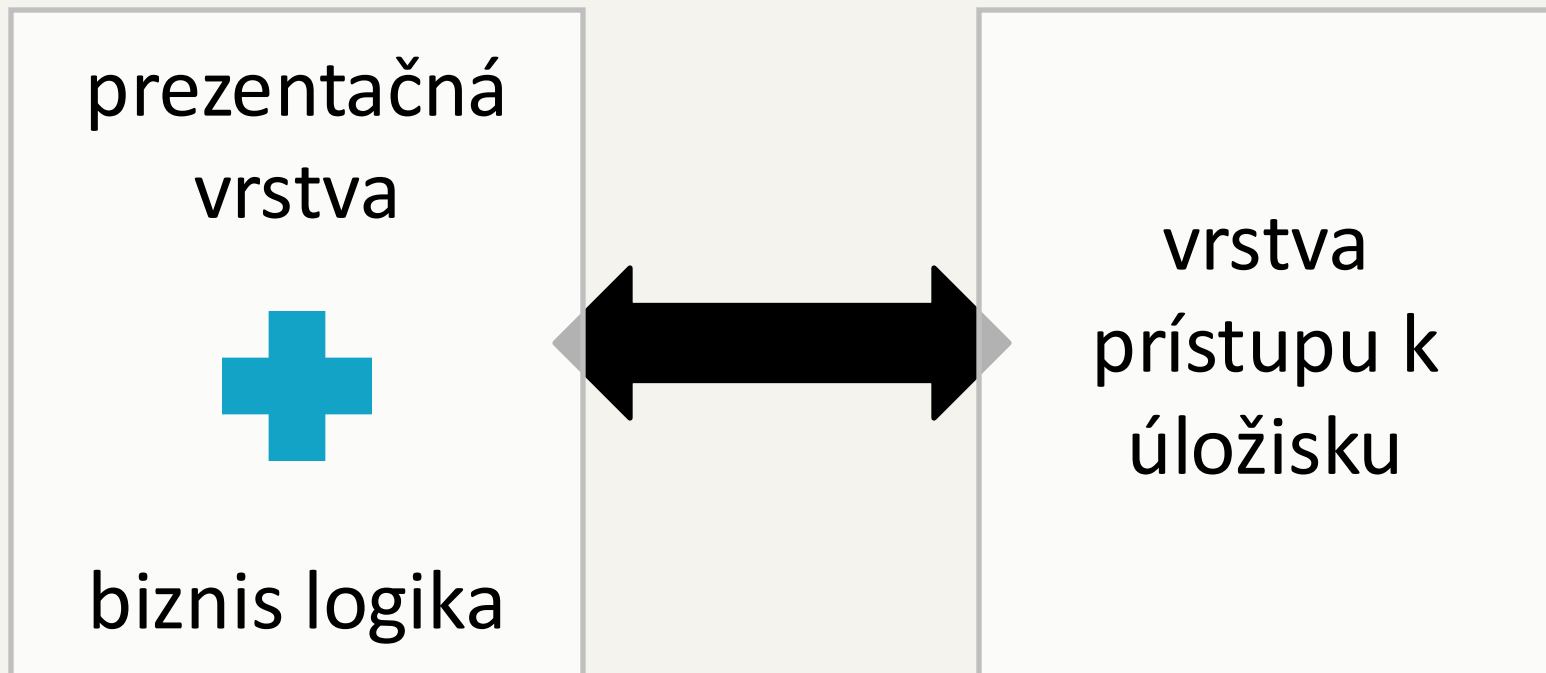
dependency

objekt vyžaduje pre fungovanie iný objekt

kolaborácia

závislosť

2vrstvové aplikácie



kód:
inštančná
premenná

```
class EntranceMainController {  
    private UserDao userDao;  
    ...  
}
```

```
class UserDao {  
    ...  
}
```

kód

```
class EntranceMainController {  
    private UserDao userDao = new UserDao();  
    ...  
}
```

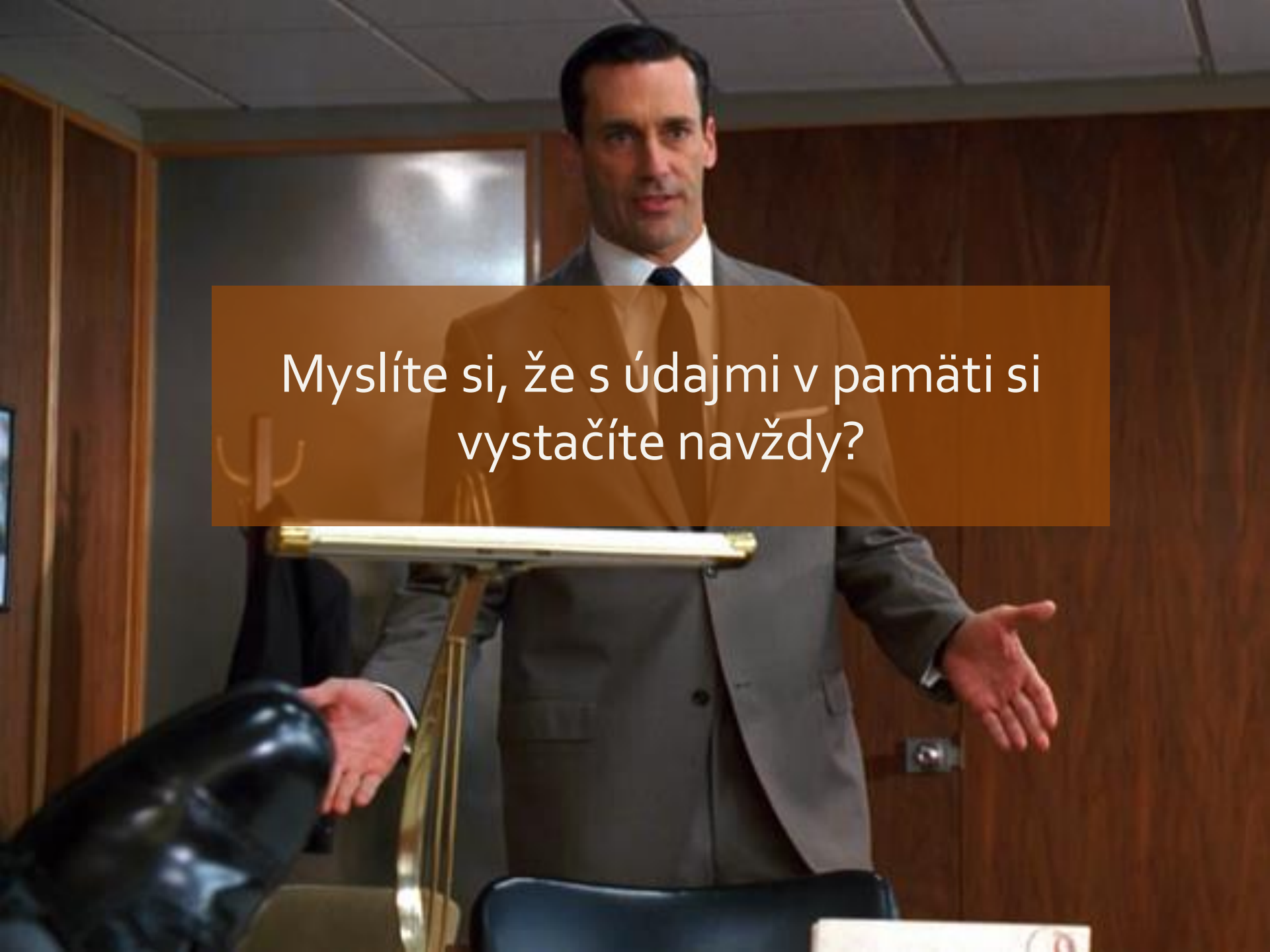
trieda si vytvorí závislosť sama

„jeden modul programu vyrába iný“



trieda sa zadrôtuje s implementáciou

čo ak chceme
implementáciu meniť?

A man in a grey suit and dark tie stands behind a gold-colored podium in a wood-paneled room. He is gesturing with both hands, palms up, as if speaking or presenting. A semi-transparent orange rectangle is overlaid on the image, containing white text.

Myslíte si, že s údajmi v pamäti si
vystačíte navždy?



migrujme
použivateľov
na disk / do databázy



vymieňame
perzistentnú vrstvu!

refaktor





zahodíme UserDao nad pamäťou?

A man in a dark grey suit, white shirt, and striped tie stands in front of a window. The window has horizontal blinds on the left and dark red curtains on the right. Sunlight filters through the blinds, creating a striped pattern on the man's suit. The man has a serious expression and is looking directly at the camera. His hands are clasped in front of him, and a bright pink scribble obscures the lower part of his hands.

uvažujme
úplne inak



trieda je čierna skrinka

čo, nie ako

pri návrhu triedy začíname
od schopností

implementácia až v druhom
rade

spomeňme na TDD – najprv testy
nad prázdnyimi metódami

CONTRACT

vonkajší pohľad na triedu

The diagram consists of three main elements arranged horizontally. On the left is a blue arrow pointing to the right, containing the text 'klient' and 'slušné správanie'. In the center is a red square containing the text 'kontrakt'. On the right is an orange arrow pointing to the left, containing the text 'trieda', 'prísľuby', 'zodpovednosti', and 'garancie'. The blue and orange arrows overlap the central red square, with their points of contact creating darker shades of blue and orange respectively.

klient

slušné správanie

kontrakt

trieda

prísľuby
zodpovednosti
garancie

interfejs

sada operácií, ktoré objekt
poskytuje navonok

= rozhranie
= protokol
= kontrakt

```
public interface UserDao {  
    User findById(Integer id);  
    List<User> findByName(String name);  
    User findById(String chipId);  
    List<User> activeUsers();  
    void save(User user);  
    void delete(User user);  
}
```

ktokoľvek vie
naplniť kontrakt

```
public class MemoryUserDao implements UserDao
{
    private List<User> users = new ArrayList<>();

    public User findById(Integer id) {
        for(int i = 0; i < users.size(); i++)
            if (users.get(i).getId()==id)
                return users.get(i);
        return null;
    }
    /* implementácie ostatných metód */
}
```

ktokoľvek vie
naplniť kontrakt

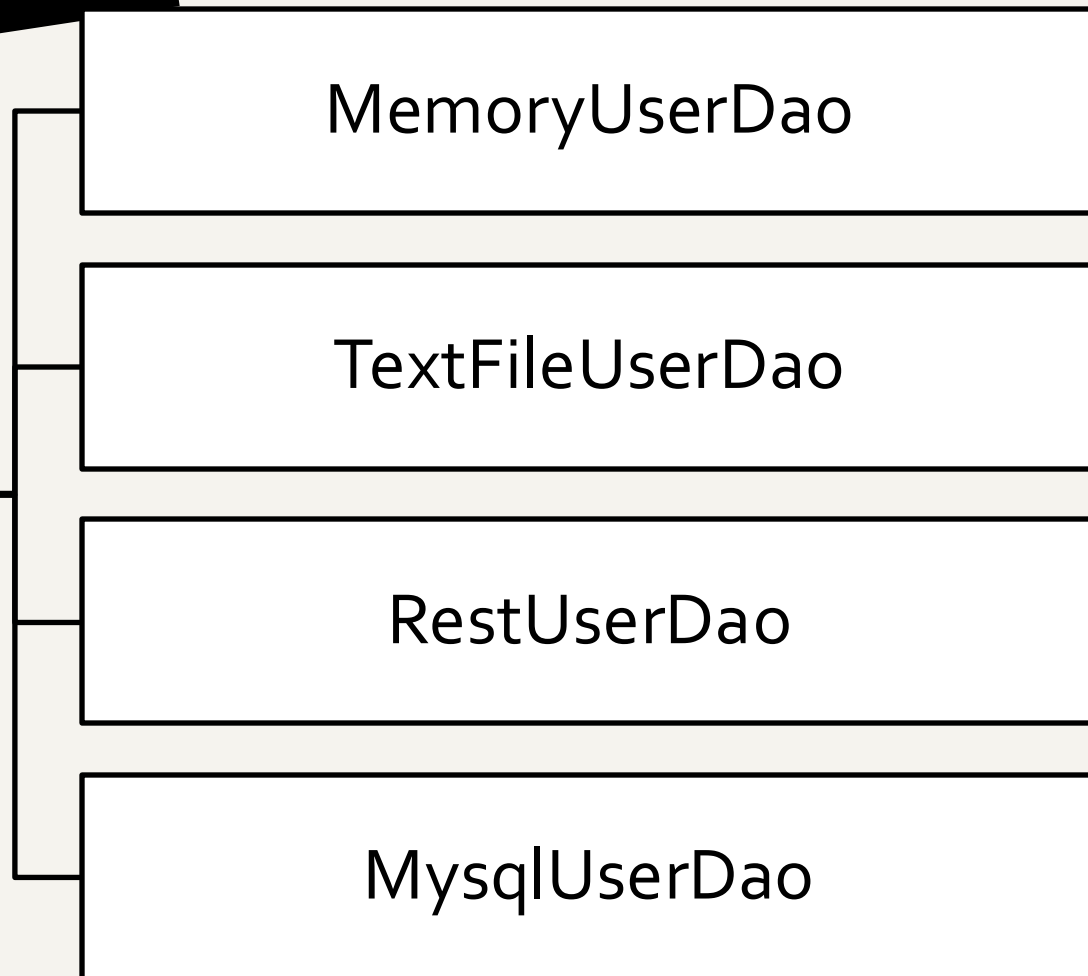
UserDao

MemoryUserDao

TextFileUserDao

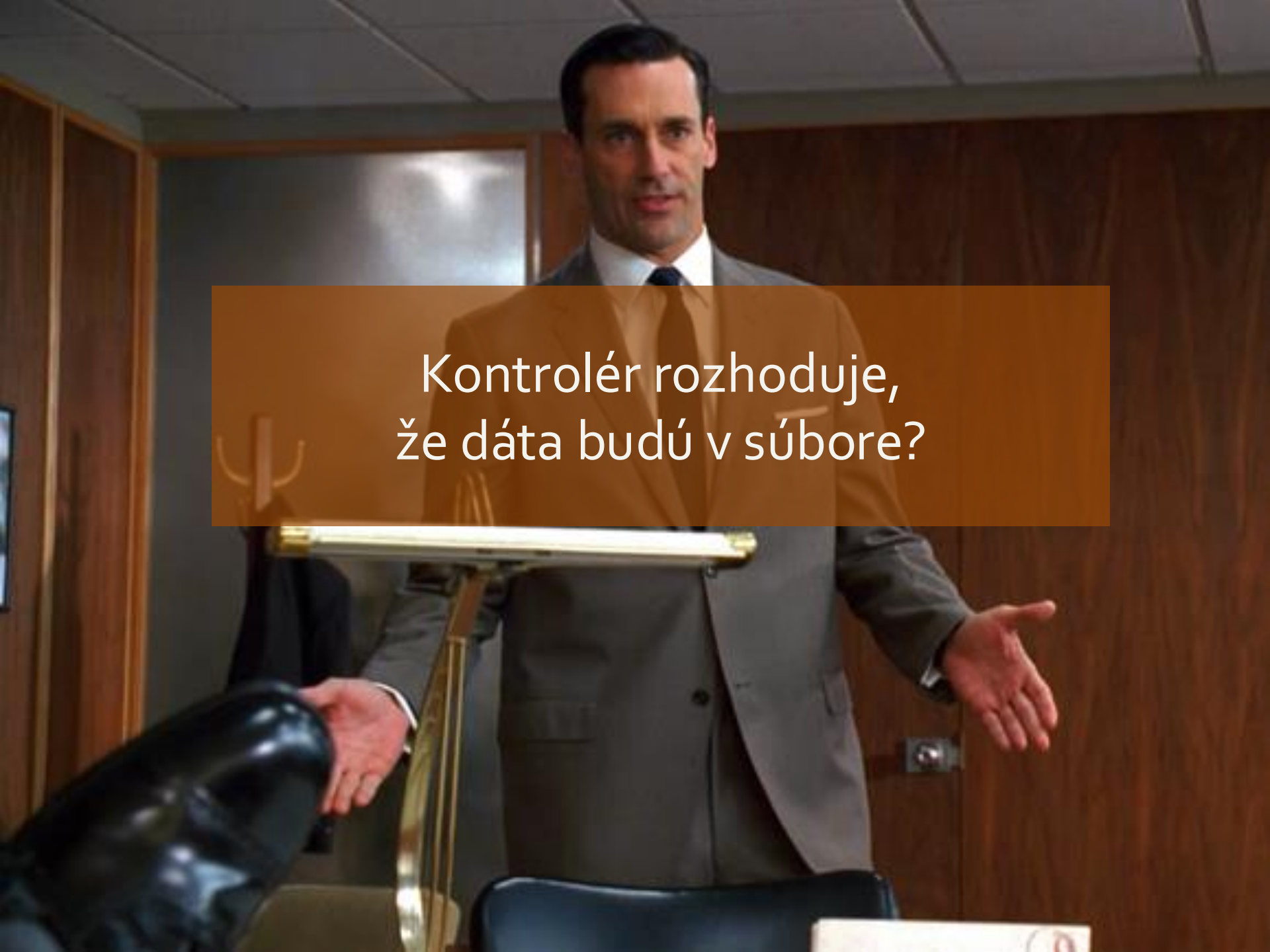
RestUserDao

MysqlUserDao



použitie

```
public class EntranceMainController {  
    private UserDao userDao  
        = new TextFileUserDao();  
    ...  
}
```

A man in a grey suit and dark tie stands behind a gold-colored podium in a room with wood-paneled walls. He is gesturing with both hands, palms up, as if presenting or explaining something. A semi-transparent orange rectangle is overlaid on the center of the image, containing white text.

Kontrolér rozhoduje,
že dáta budú v súbore?

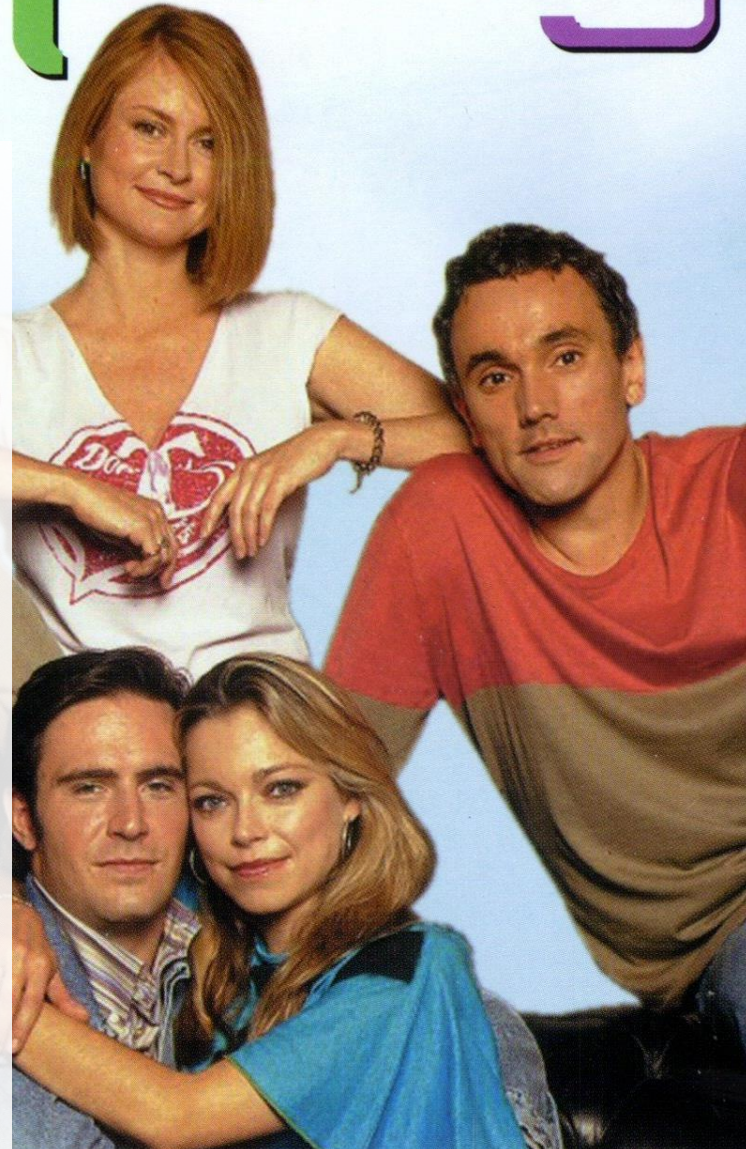
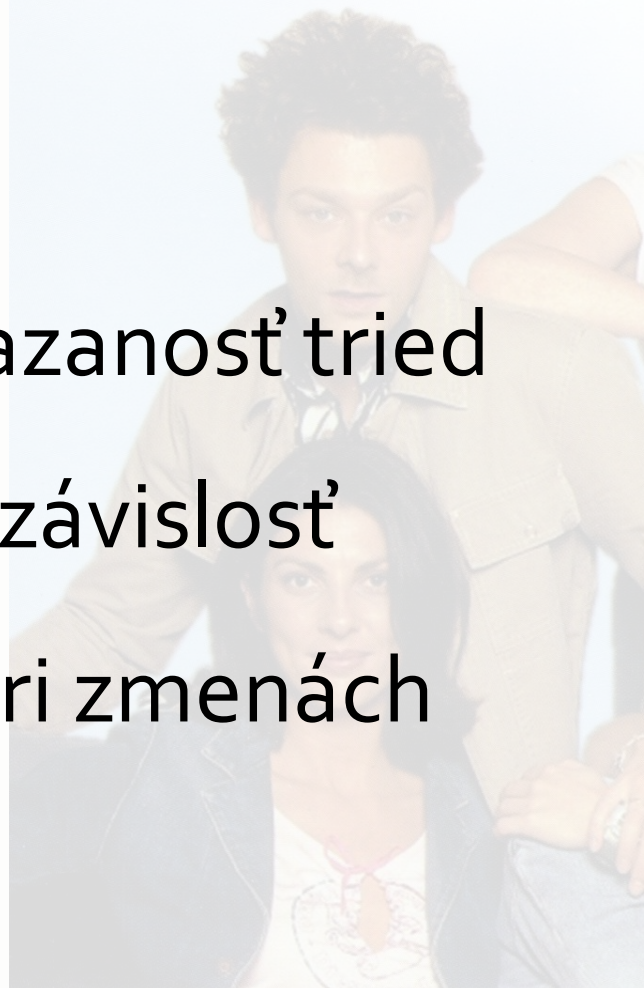
Čo ak máme viac tried v GUI?

- Viac scén a kontrolérov, „vyskakovacie okná“
- Každá trieda z GUI bude mať inštančnú premennú DAO
 - Bude si každá trieda GUI vyrábať vlastnú perzistentnú vrstvu?
 - Budú si musieť objekty hovoriť medzi sebou referenciu na spoločné DAO?
 - A ktorý z nich ju má vyrábať?



Tight Coupling

- úzka zviazanosť tried
- priveľká závislosť
- krehké pri zmenách



A photograph of a grassy hill with a large white sign that reads "HOLLYWOOD PRINCIPLE". The sign is made of large, blocky letters. The background shows a clear blue sky and some trees at the top of the hill. The foreground is a grassy slope with some patches of dirt.

HOLLYWOOD PRINCIPLE

Nevolajte nám, my sa vám ozveme.

The image shows a grassy hillside with the word 'HOLLYWOOD' in large, white, block letters. The letters are slightly weathered and are set against a background of green grass and some trees in the distance. The sky is blue. The overall scene is a classic representation of the Hollywood sign.

HOLLYWOOD

PRINCIPLE

Trieda poprosí niekoho, nech
jej vyrobí a dodá kontraktorov.

továreň

objekt, ktorý vyrába iné objekty

A photograph of a car body on an assembly line. Several yellow robotic arms are positioned around the car, likely for welding or painting. The background shows more of the factory environment with various equipment and structures.

továreň

- Rozhoduje, ktorá implementácia interfejsu sa vyberie
 - pamäť, súbor, databáza,...
- **Našteluje danú implementáciu**
 - login, heslo, umiestnenie, ...

...Factory

```
public class Factory {  
    public UserDao getUserDao() {  
        /* tu sa vráti inštancia vybranej  
           implementácie UserDao*/  
    }  
}
```

`getUserDao()`

nový objekt?

rovnaký
objekt?

z poolu?

akého typu?

použitie továrne

kontrolér potrebuje DAO



továreň vytvorí DAO



kontrolér potrebuje
továreň

trieda potrebuje továreň

```
public class EntranceMainController {  
    private UserDaoFactory factory  
        = new UserDaoFactory();  
}
```

trieda potrebuje továreň

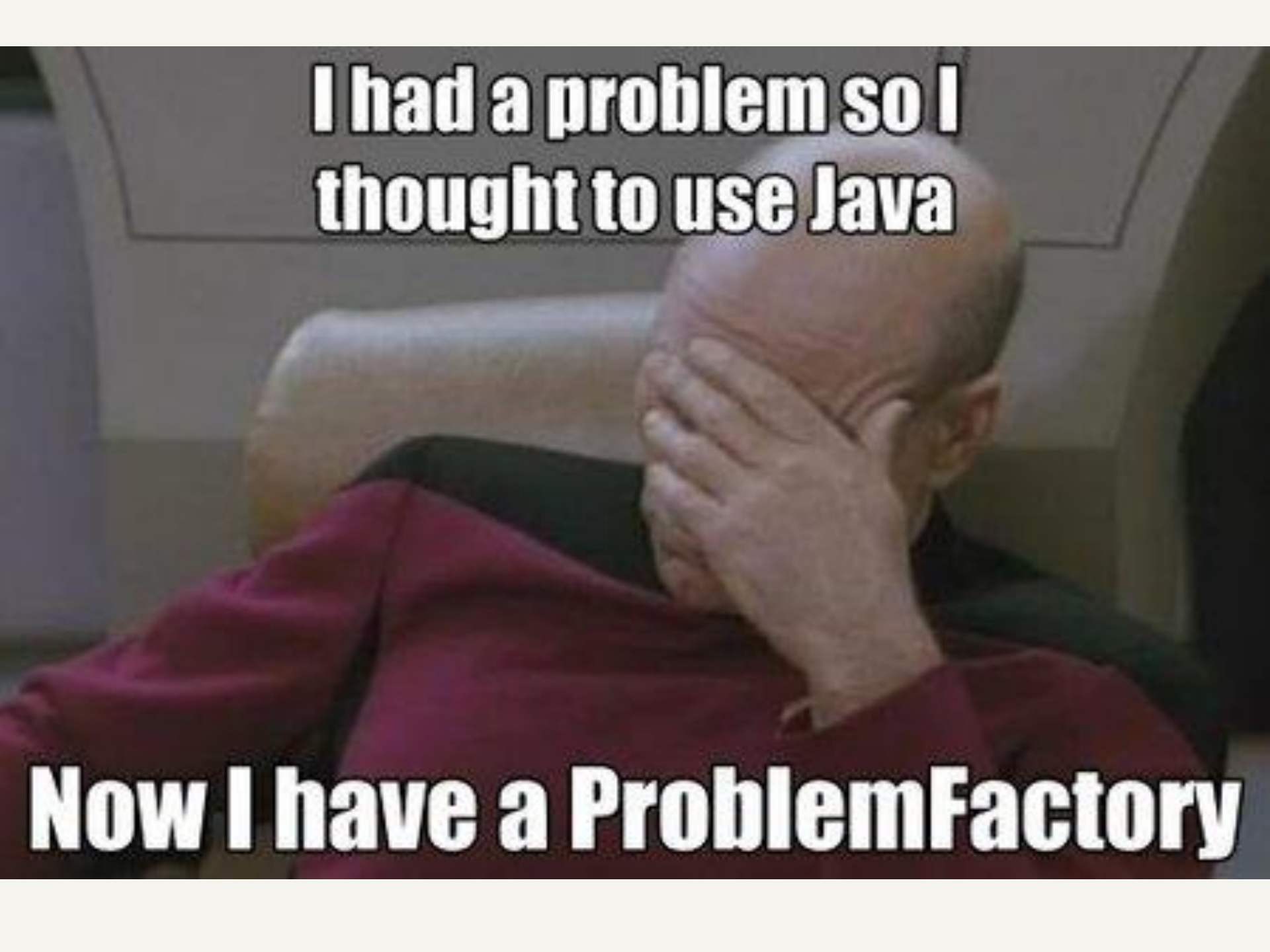
```
public class EntranceMainController {  
    private UserDaoFactory factory  
        = new UserDaoFactory();  
}
```

lolwut?



„Každý problém v
informatike možno
vyriešiť pridaním ďalšej
úrovne nepriameho
prístupu / abstrakcie.“

- Butler Lampson, 1972

A man with a balding head, wearing a red long-sleeved shirt, is sitting in a light-colored chair. He has his right hand pressed against his face, covering his eyes and nose, with a look of frustration or despair. The background is a plain, light-colored wall.

**I had a problem so I
thought to use Java**

Now I have a ProblemFactory



kol'ko tovární
potrebujeme?

FROM ANOTHER TIME COMES A MAN
OF GREAT POWER.
A MAN OF INCREDIBLE STRENGTH.
AN IMMORTAL ABOUT TO FACE HIS
GREATEST CHALLENGE...

SINGLETON

jediná inštancia v
systéme

THE
JAMES
CAMPBELL
CHRISTOPHER YOUNG
MICHAEL KAMEN
E.C. MONTELL
GREGORY WOOD
PETER BELLWOOD & LARRY FERGUSON
RUSSELL MURCAHY
CANNON
SOUNDTRACK AVAILABLE ON TAP RECORDS & TAPE
RELEASED BY COLUMBIA CANNON
COLUMBIA CANNON

FEATURING
ORIGINAL SONGS
BY
QUEEN

Joshua Bloch

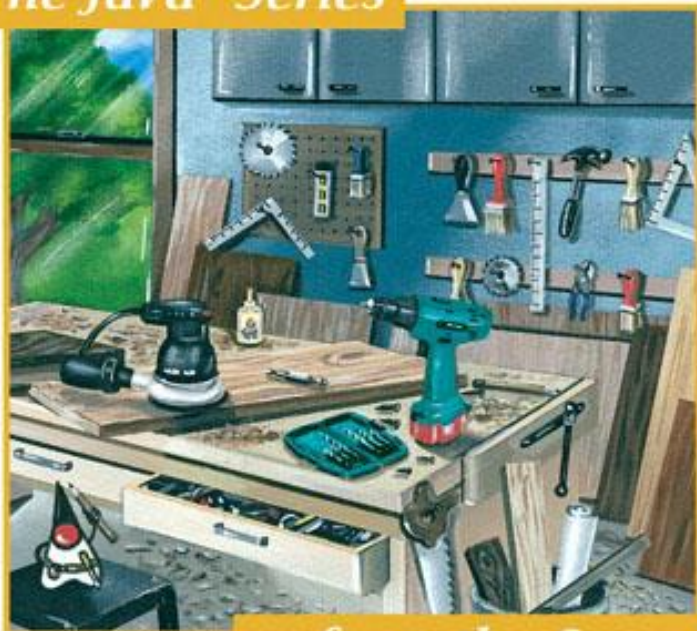
Revised and
Updated for
Java SE 6



Effective Java™

Second Edition

The Java™ Series



...from the Source



Rada 3: singleton
cez privátny
konštruktor alebo
enum

továreň v kóde

```
public enum Factory {  
    INSTANCE;
```

```
    public UserDao getUserDao() {  
        /*...*/  
    }  
}
```

továreň v kóde

```
UserDao userDao =  
    UserDaoFactory.INSTANCE  
        .getUserDao();
```

hack,
ale
korektný



migrácia: 1 riadok!

```
public enum UserDaoFactory {  
    INSTANCE;
```

```
    public UserDao getUserDao() {  
        return new MemoryUserDao();  
        return new MysqlUserDao();  
    }  
}
```

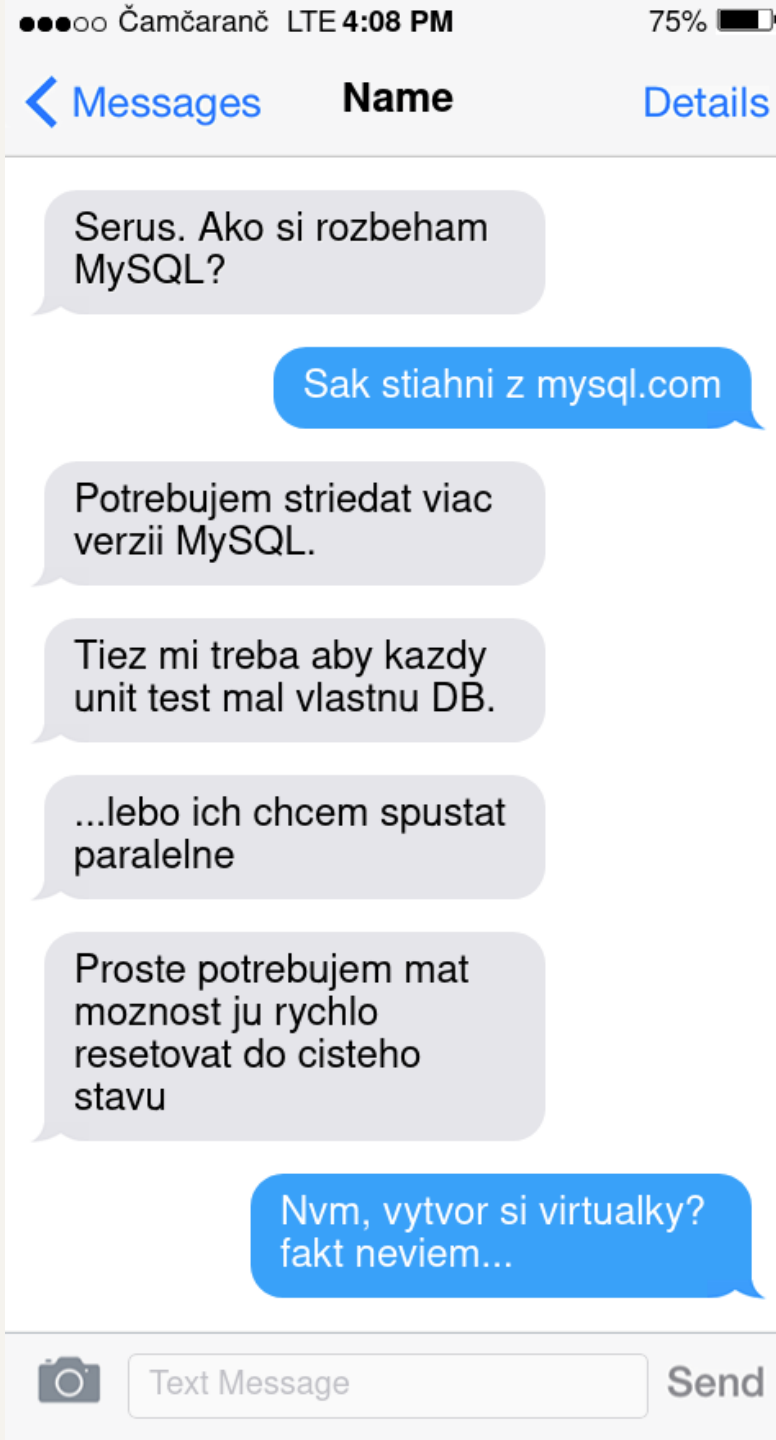


MySQL pre programátorov

Naša entita User v relačnej databáze

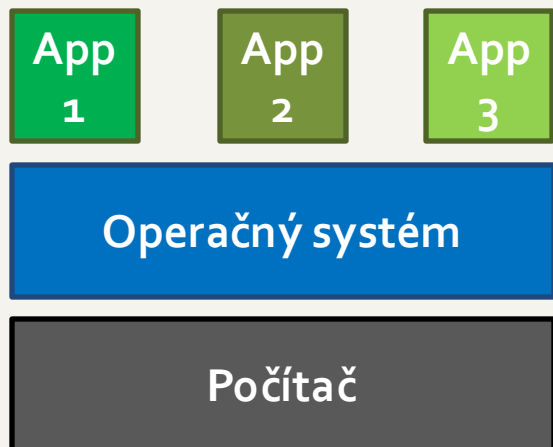
```
CREATE TABLE users (  
  id BIGINT NOT NULL AUTO_INCREMENT,  
  chip_id VARCHAR(20) NOT NULL,  
  name VARCHAR(45) NOT NULL,  
  active BOOLEAN,  
  
  PRIMARY KEY (`id`)  
);
```

Spúšťanie DB

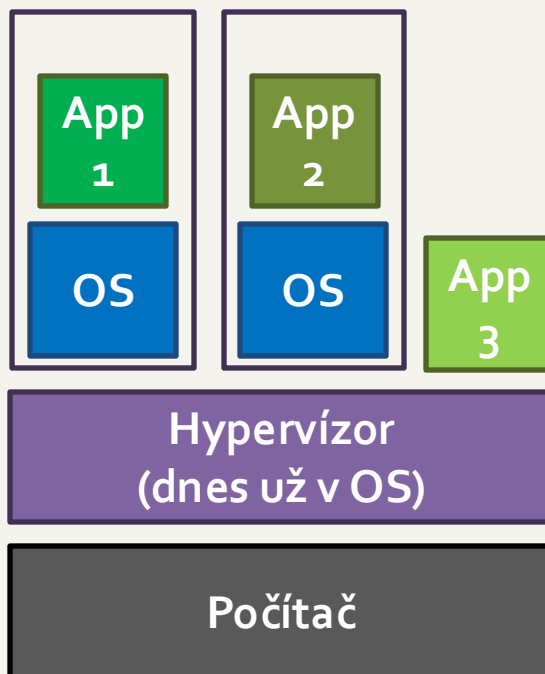


Virtualizácia

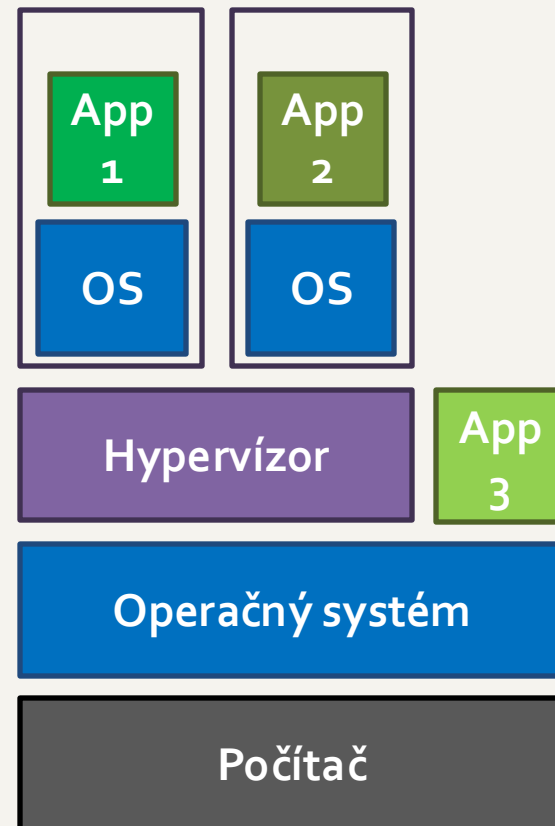
Bežný PC



Virtualizácia
(typu 1)

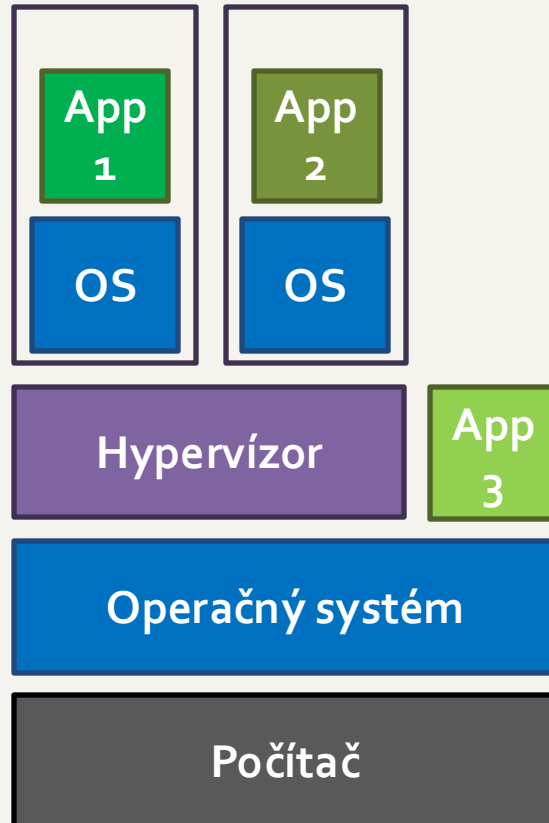


Virtualizácia
(typu 2)

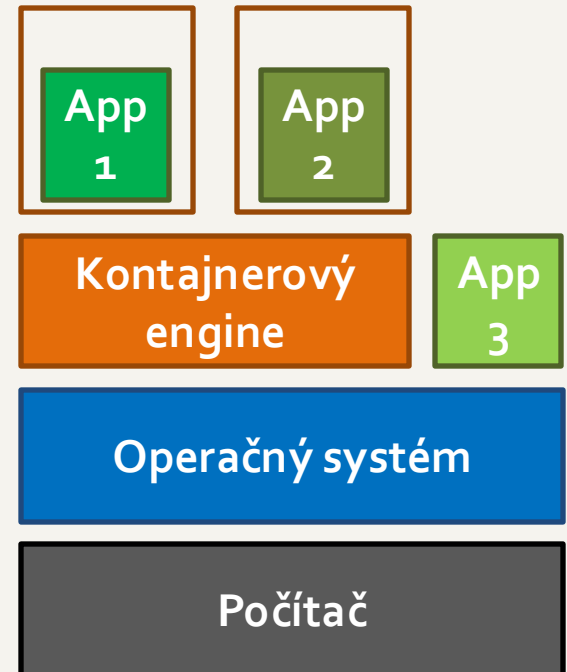


Kontajnerizácia

Virtualizácia
(typu 2)



Kontajnerizácia





- Docker je kontajnerizačný nástroj
 - kontajner - "odľahčená virtuálka"
- Moderná appka potrebuje viacero iných appiek
 - MySQL/PostgreSQL, Redis, Kafka, ElasticSearch, ...
- Docker umožní na jeden klik/príkaz tieto appky
 - sťahovať, spúšťať, vypínať, **resetovať**...
 - skvelé aj na testy (cez TestContainers knižnicu)

MySQL cez Docker Compose

docker-compose.yml

```
services:
  mysql:
    image: mysql:9.7.2
    environment:
      MYSQL_ROOT_PASSWORD: entrance
      MYSQL_DATABASE: entrance
      MYSQL_USER: entrance
      MYSQL_PASSWORD: entrance
    ports:
      - "3307:3306"
    volumes:
      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
```

stiahni nejaký MySQL

konfigurácia

mapuj port host<-kontajner

(voliteľne) sprístupni súbor init.sql do magického priečinka /docker-entrypoint-initdb.d v kontajneri. MySQL ho načíta pri prvom spustení.

Spustíme príkazom **docker compose up**

alebo **docker compose up -f docker-compose.yml**

Otázky?