


Vývoj Aplikácií s Viacvrstvovou Architektúrou

Logovanie





Čo je dobré používať na Logovanie?



JDK

JRE

Java SE API

Compact Profiles

Prečo logovat?

“Logs are like car insurance. Nobody wants to pay for it, but when something goes wrong everyone wants the best available”

-- Pablo Kraan



Prečo logovať?

- Čo sa práve deje v aplikácií?
 - Čo všetko sa dialo predtým, než nastala výnimka?
-
- Zachytávanie nezvyčajných udalostí alebo chýb v programe
 - Monitoring aplikácie
 - Konzolové aplikácie
 - GUI aplikácie
 - Mobilné aplikácie (Logcat)

Vlastný Log

```
System.out.println("Štart " + new Date());
```

```
System.console().printf("Štart %s", new Date());
```

Vlastná Logovacia metóda

```
public void test() {  
    println("Štart " + new Date());  
}
```

```
private void println(String message) {  
    System.out.println(message);  
}
```

Vlastná Logovacia trieda

```
public static boolean DEBUG = true;
```

```
public void test() {  
    println("Štart " + new Date());  
}
```

```
private void println(String message) {  
    if (DEBUG) {  
        System.out.println(message);  
    }  
}
```


Vlastná Logovacia trieda

```
public static boolean DEBUG = Boolean.getBoolean("DEBUG");

public void test() {
    println("Štart " + new Date());
}

private void println(String message) {
    if (DEBUG) {
        System.out.println(message);
    }
}
```

Cesta nikam

- Ciele
 - Console out, err.
 - Súbory, aký formát(y)?
- Úrovne udalostí
- Umiestnenie zdroja
- Formáty časovej pečiatky (Timestamp)
- Informácie o vlákne
- Konfigurácia

Kľúčové koncepty a architektúra

- Oddelenie Logging API a implementácie
- Pluggable implementácia
- Moderné logovacie frameworky
 - Apache Log4j 2
 - SLF4J + Logback
 - Apache Commons Logging (API only)
 - JUL
- Log4j 1 (nepoužívať vôbec)

Čo môžem použiť na logovanie?

1. `System.out/System.err`
2. `java.util.logging (JUL)`
3. Jakarta (Apache) commons logging (JCL)
4. `log4j`
5. `slf4j`
6. `logback`

System.out/System.err

- Len pri skúšaní izolovaných vecí a bez nepotrebných závislostí
- Funguje na všetkých verziách Javy
- Nemožno konfiguračne vypnúť na rôznych prostrediach, čo má za následok výkonnostný a bezpečnostné problémy a
- Nemožno konfiguračne nastaviť úroveň správ

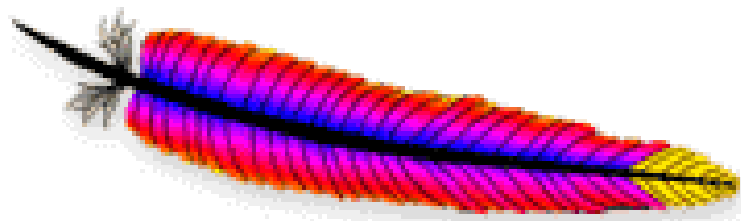
Java Util Logging (JUL)

- Od verzie 1.4

```
1.  package sk.stu.fiit;
2.  import java.util.logging.*;
3.  public class Nose {
4.      // Obtain a suitable logger
5.      private static Logger logger = Logger.getLogger("sk.stu.fiit.app");
6.      public static void main(String argv[]) {
7.          // Log a FINE tracing message
8.          logger.fine("doing stuff");
9.          try {
10.             Wombat.sneeze();
11.         } catch (Exception ex) {
12.             // Log the exception
13.             logger.log(Level.WARNING, "trouble sneezing", ex);
14.         }
15.         logger.fine("done");
16.     }
17. }
```

Jakarta Commons Logging (JCL)

- Toto riešenie malo za cieľ unifikovať rozhrania medzi rôznymi logovacími frameworkami
- Samo o sebe logovanie nerieši, len posiela správy do konkrétnych implementácií - log4j, JUL
- Rozhranie je jednoduché



Apache CommonsTM
<http://commons.apache.org/>

log4j

- Poskytuje významné vylepšenia, ktoré boli dostupné v Logback ale opravenými chybami v jeho architektúre
- Verzia 2 posledná verzia
- Lepší výkon pri použití knižnice Disruptor
- Podporuje použite viacnásobného API
- Android programy môžu používať iba Log4j a LogBack alebo balíčky thirdparty



slf4j


- Knižnica rieši rovnaký problém ako JCL - teda unifikácia rozhranie nad rôznymi logovacím API.
- Nové komerčné projekty i rad open-source projektov dnes používa práve toto API alebo logback
- Slúži ako abstrakčná vrstva pre rôzne logovacie frameworky čím dovoľuje používateľovi si použiť požadovaný framework
- Stačí pridať do dependencies
- Fasády na pozadí používajú 1 fyzicky logger Konfigurujeme log4J ale logujem do Fasády





slf4j

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
public class STU {
    public static void main(String[] args) {
        Logger logger =
            LoggerFactory.getLogger(STU.class);
        logger.info("Hello World");
    }
}
```



Fasáda - Facade

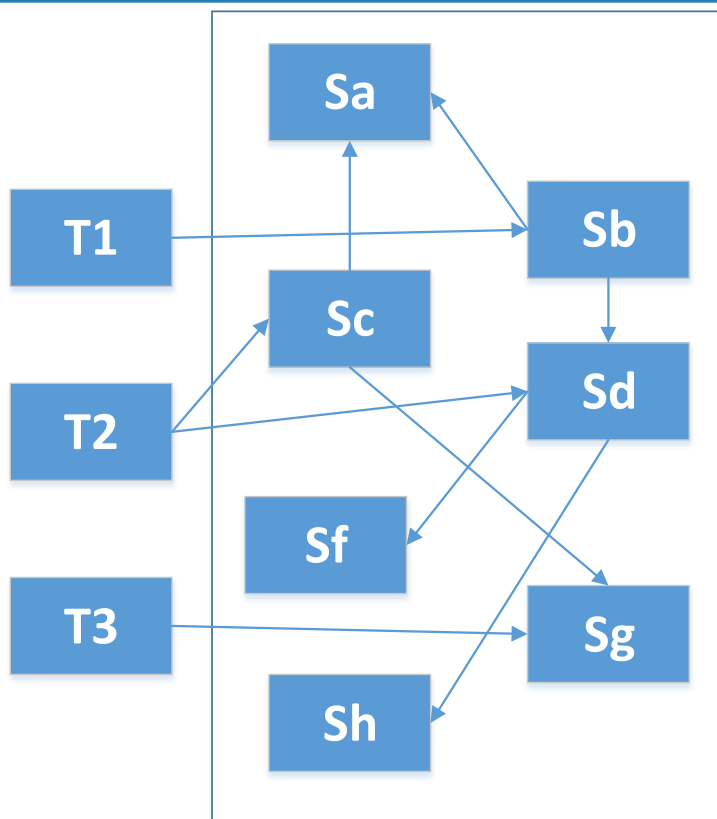
- Používame ak, nejaký **system začína byť pre** svojich **používateľov príliš zložitý** vzhľadom **k oblastiam úloh, ktoré chcú** s jeho pomocou **riešiť**.
- Ukazuje ako **nahradiť** sadu rozhraní jednotlivých subsystémov **zjednotením rozhraním zastupujúcim celý systém**.
- **Definuje** tak **rozhranie vyššej úrovne**, ktoré **uľahčí využívanie podsystémov**.
- Jej cieľom je **zjednodušiť rozhranie celého systému** a **znižiť počet tried**, s ktorými **musí používateľ priamo, či nepriamo komunikovať**.

Fasáda - Facade

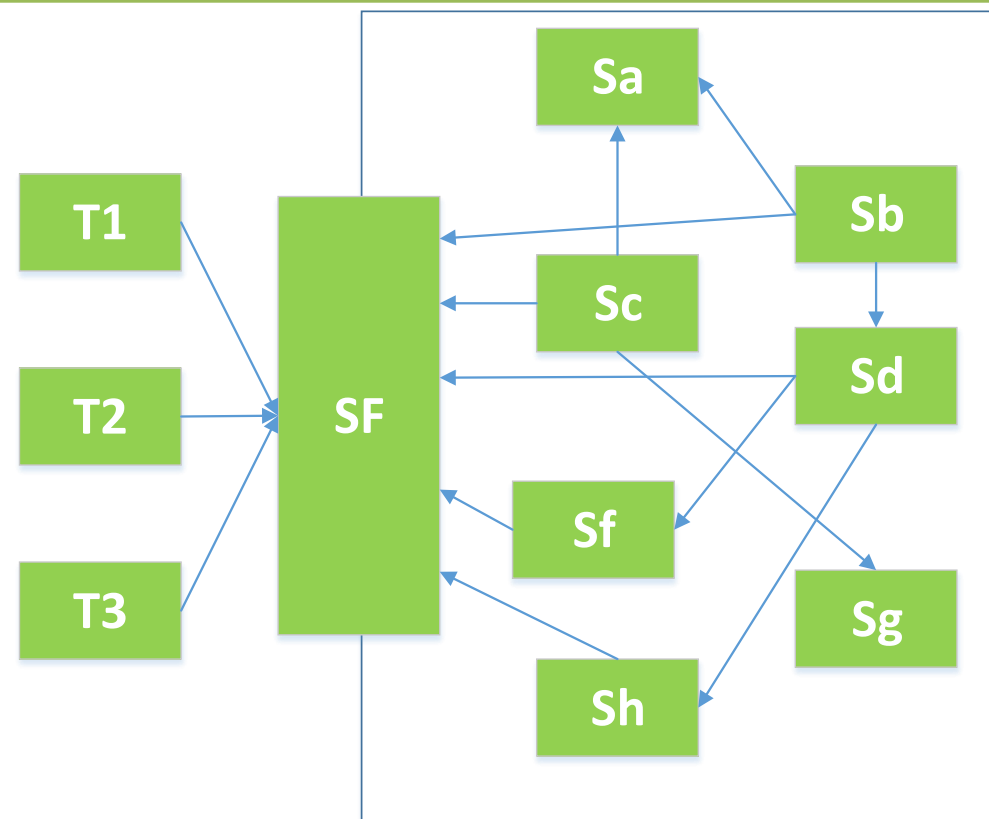
- Poskytuje **jednotné rozhranie k množine rozhraní** v subsystemu. Fasáda definuje rozhranie vyššej úrovne, ktoré používanie subsystemu uľahčí.

Ako nám fasáda zjednodušuje život

Spolupráca tried v zložitom systéme



Zjednodušenie pomocou fasády



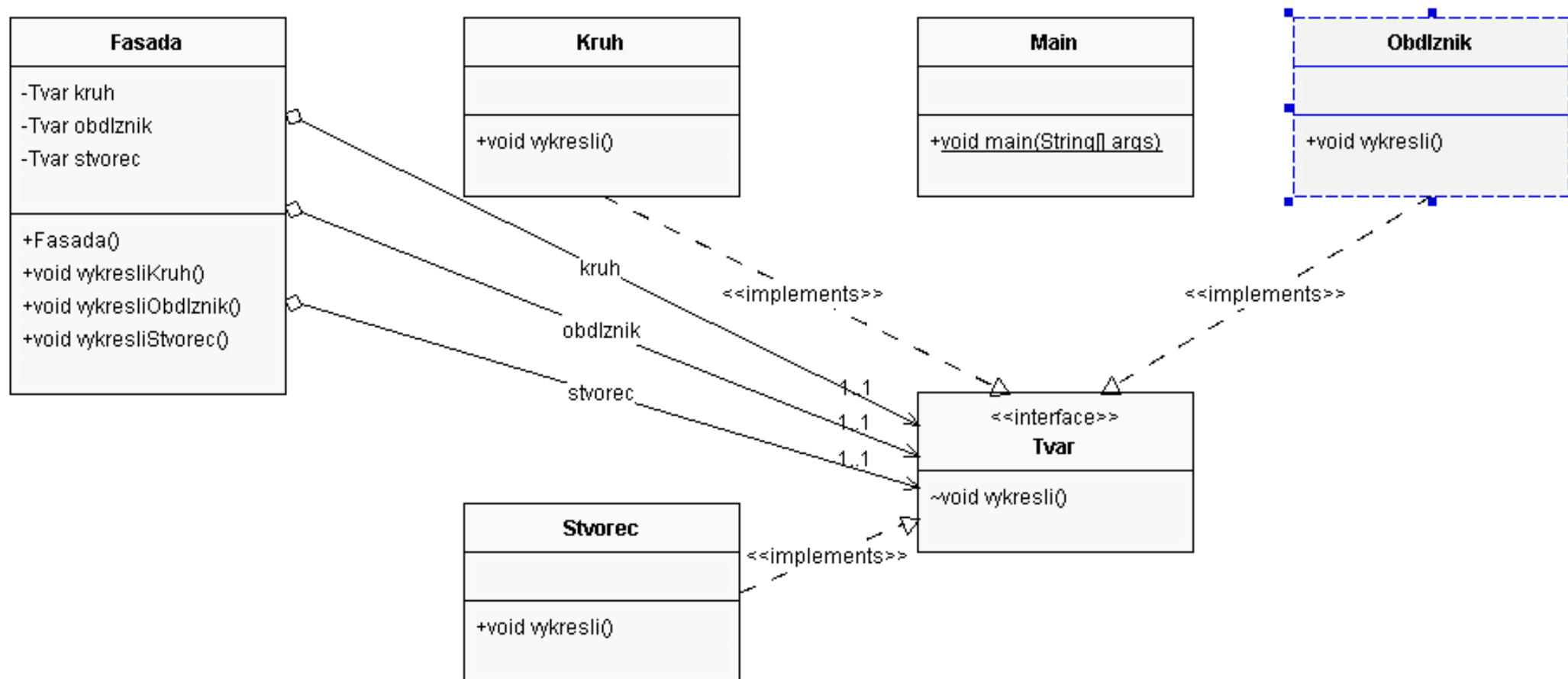
Podobne ako pod fasádou domu je schovaný komplexný systém...

Príklady Fasáda - Facade

- `Javax.swing.JOptionPane`
- `Java.lang.System`
- `Java.lang.Runtime`
- `Java.lang.Process`

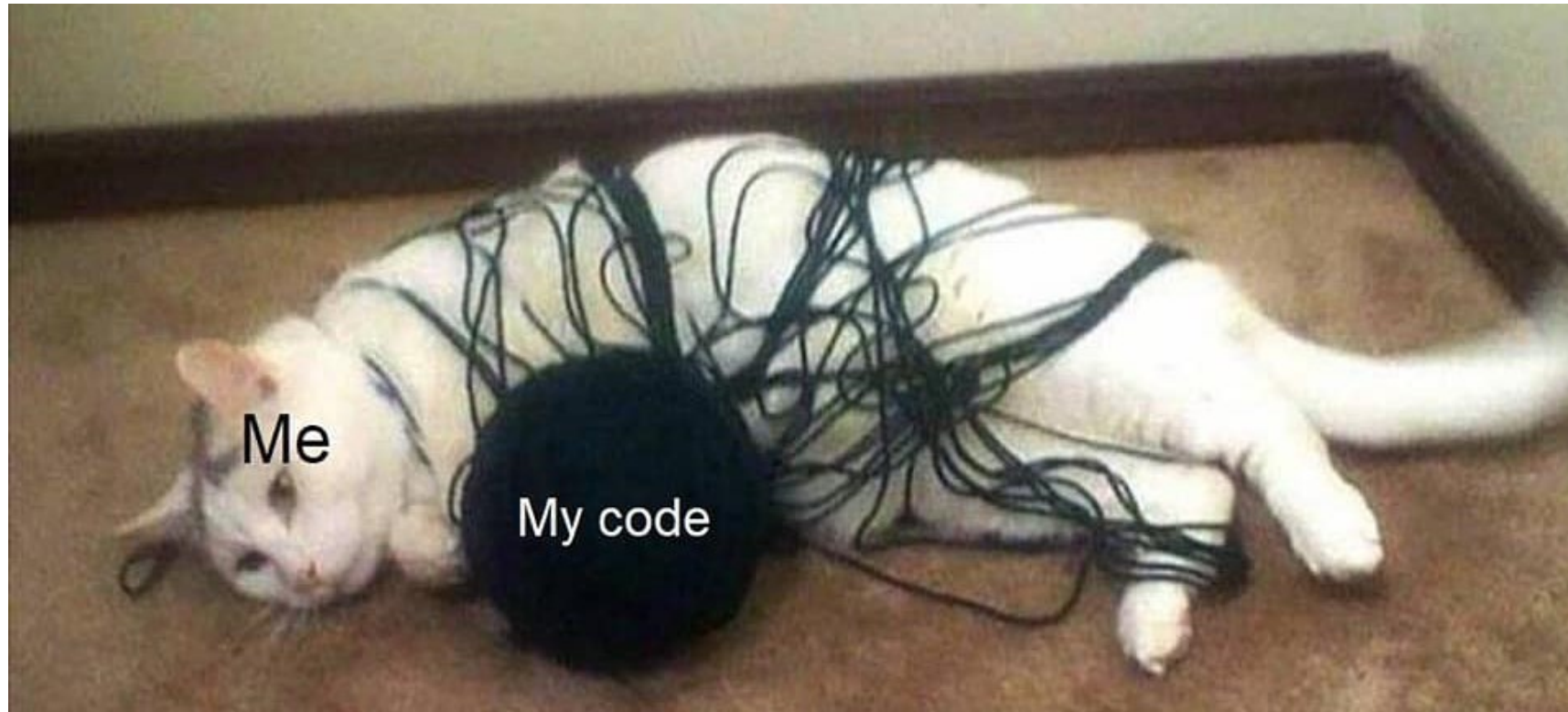
Má zmysel, keď cena vytvorenia fasády (trieda, či skupina trieda) je evidentne nižšia než cena štúdia kompletného systému používateľmi, ktorí ho nebudú využívať celý resp. cena správy celého systému.

UML Fasáda - facade



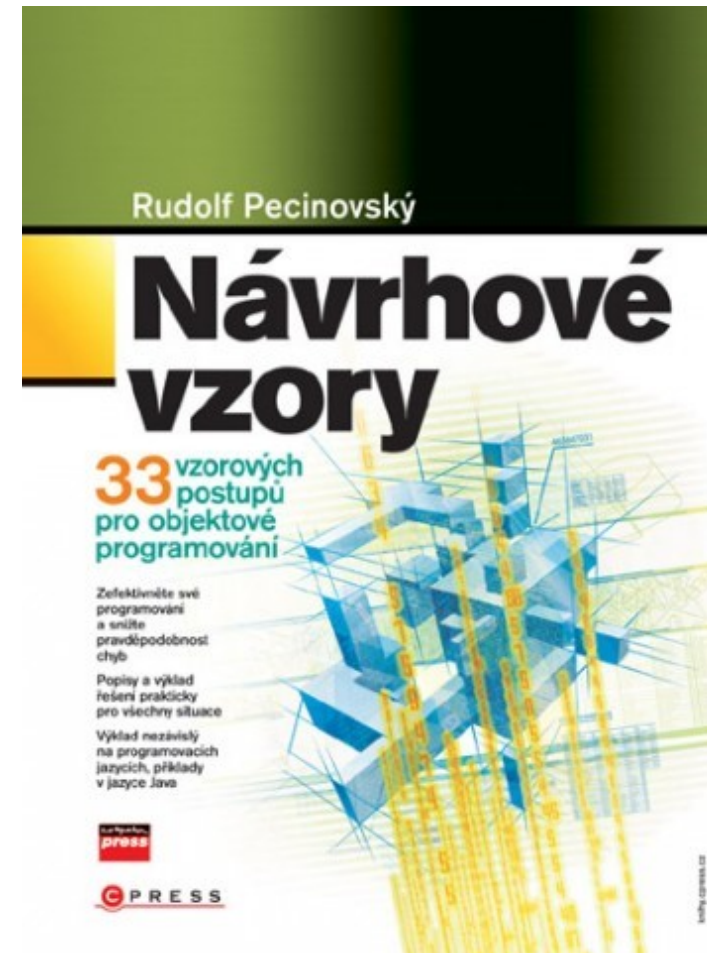
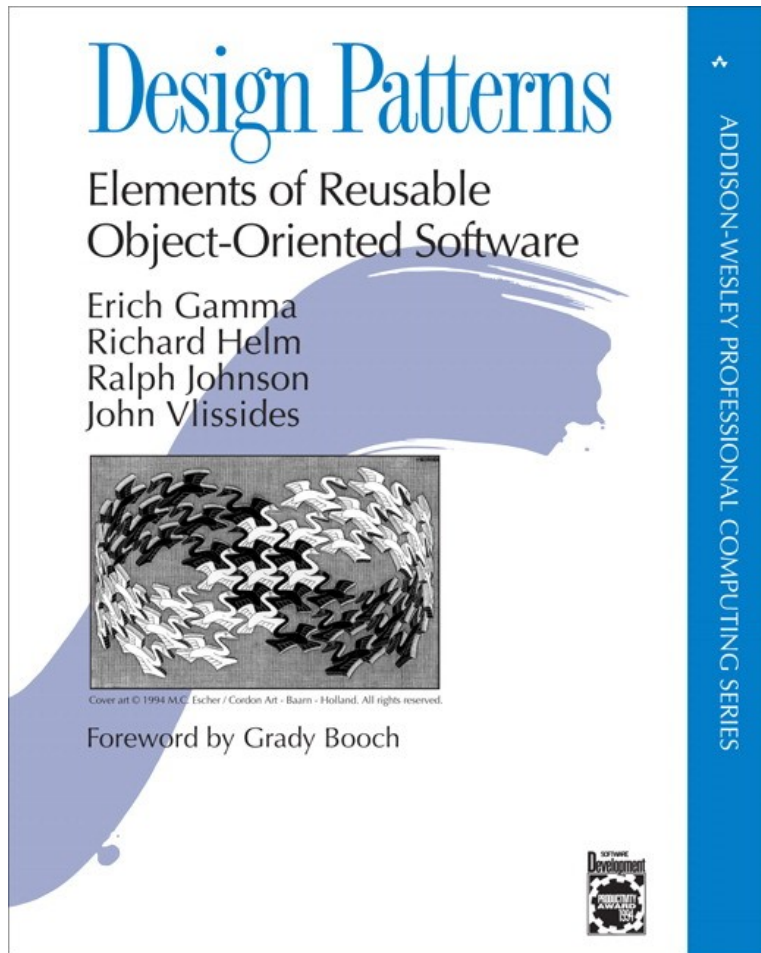
Ako by ste vysvetlili fasádu 5-ročnému dieťaťu alebo 90-ročnej babičke?

24



Google pohovory

Gang of four - GOF



[Click here to next software adventure](#)

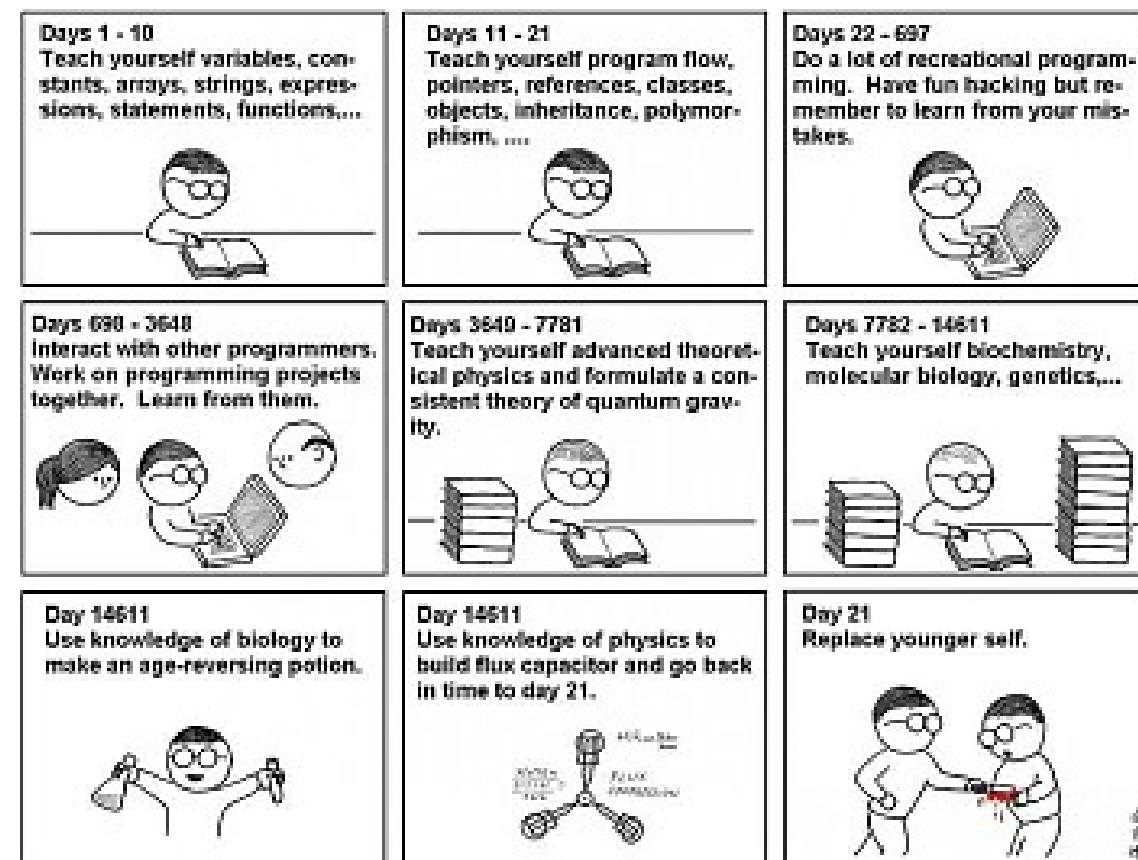
Koľko a aké sú návrhové vzory?

THE 23 GANG OF FOUR DESIGN PATTERNS

C	Abstract Factory	S	Facade	S	Proxy
S	Adapter	C	Factory Method	B	Observer
S	Bridge	S	Flyweight	C	Singleton
C	Builder	B	Interpreter	B	State
B	Chain of Responsibility	B	Iterator	B	Strategy
B	Command	B	Mediator	B	Template Method
S	Composite	B	Memento	B	Visitor
S	Decorator	C	Prototype		

2 kruté fakty design paternov

- Väčšina ľudí nepochopí návrhové vzory na prvý krát
- Dokonca ani samotní tvorcovia
- Ľudia často používajú návrhové vzory, alebo ich časti a ani o tom nevedia...



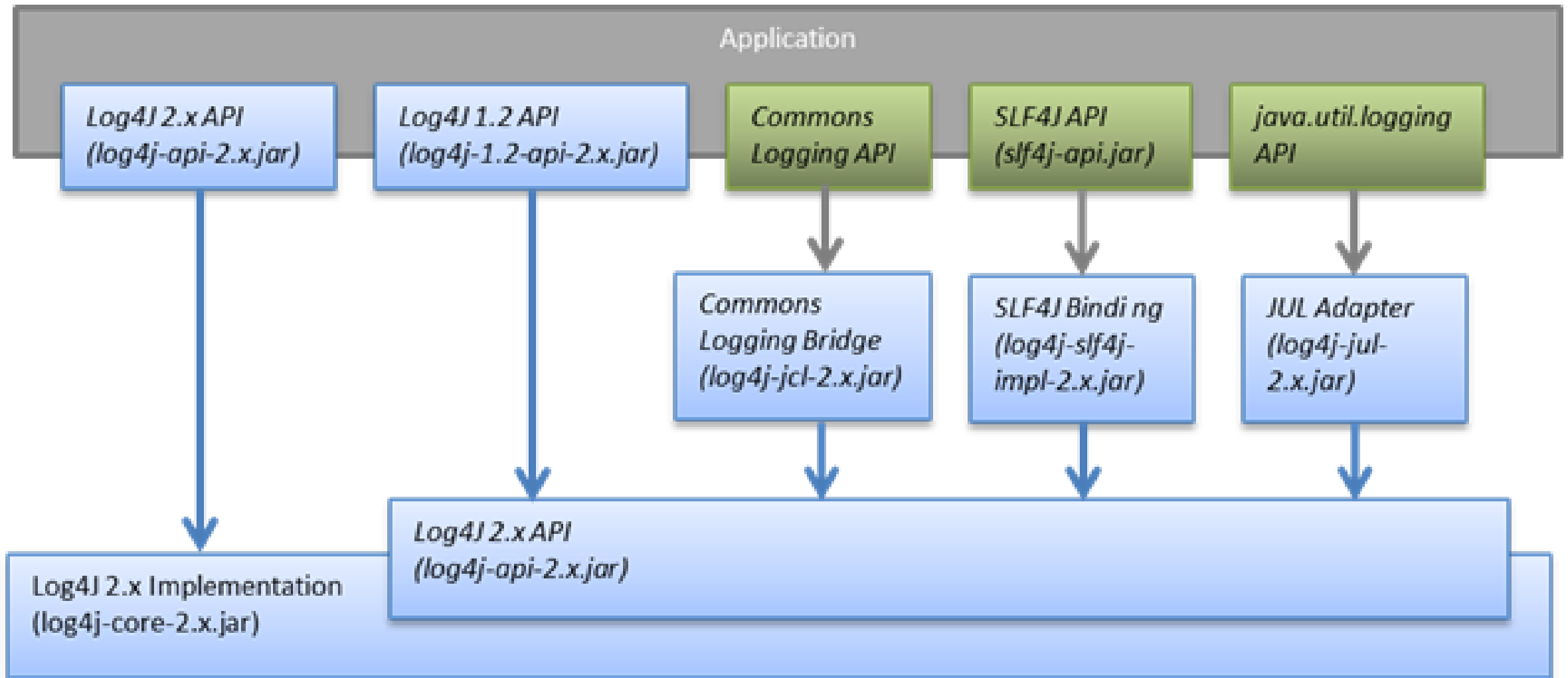
Trpezlivosť ruže prináša

Logback

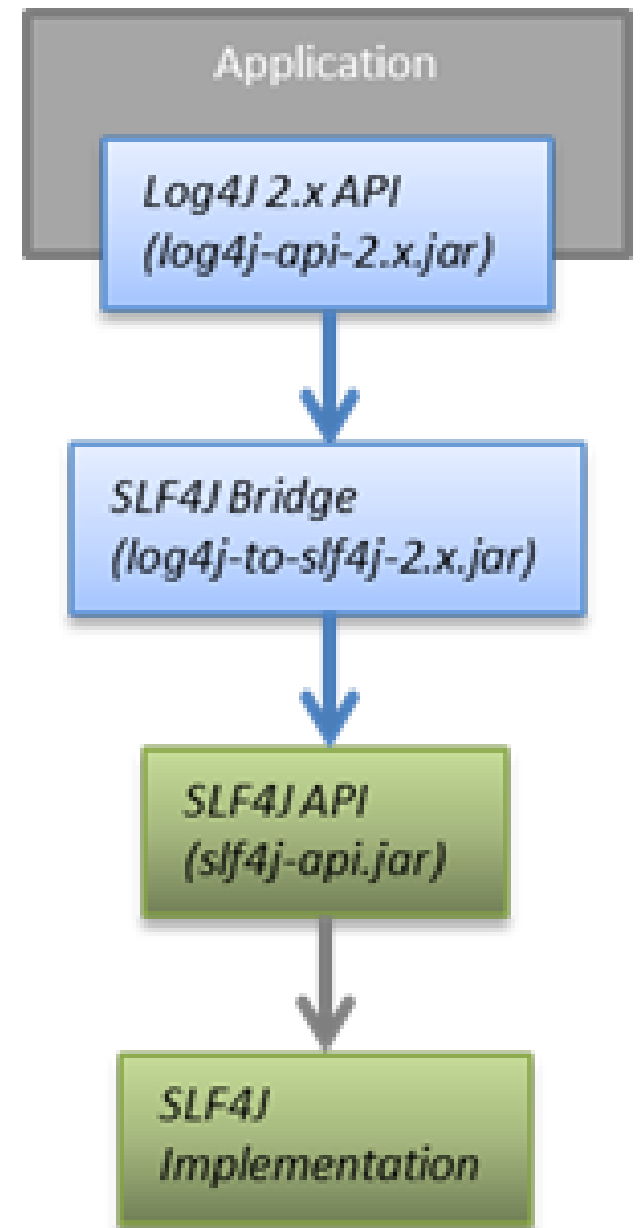
- Autorom je tvorca Log4j aj slf4j - ČEKI Gülcü
 - Podmienená konfigurácie, pre nastavenie úrovne logovania v závislosti na prostredí
 - SiftingAppender, ktorý umožňuje logovať podľa dynamického parametra, napr zalogovaného používateľa.
- <http://logback.qos.ch/reasonsToSwitch.html>



Log4j API architektúra



Pluggable implementácia SLF4J



Kľúčové koncepty

- API
- Konfigurácia
- Prispôsobenie/zapojiteľnosť

Kľúčové koncepty APIs

- LogManager
 - Level
 - Logger
 - Marker
-
- Sledovanie toku so značkováním
 - Kontextový zásobník a kontextová mapa

LogManager

- Tvorba Loggerov
- `LogManager.getLogger("sk.stu.fiit.NasaTrieda");`
- `LogManager.getLogger(NasaTrieda.class);`
- `LogManager.getLogger();`

LogManager a Továrne na správy

- Message factories
- Login OK for user {} from terminal {}
- Login OK for user {0} from terminal {1}
- Login OK for user %s from terminal %s
- ETL processed %,d records

Továrne na správy (Message factories)

- `org.apache.logging.log4j.LogManager.getLogger(Class|String|Object, MessageFactory)`
 - `FormattedMessageFactory`
 - `LocalizedMessageFactory`
 - `MessageFormatMessageFactory`
 - `ParameterizedMessageFactory`
 - `ParameterizedNoReferenceMessageFactory`
 - `SimpleMessageFactory`
 - `StringFormatterMessageFactory`



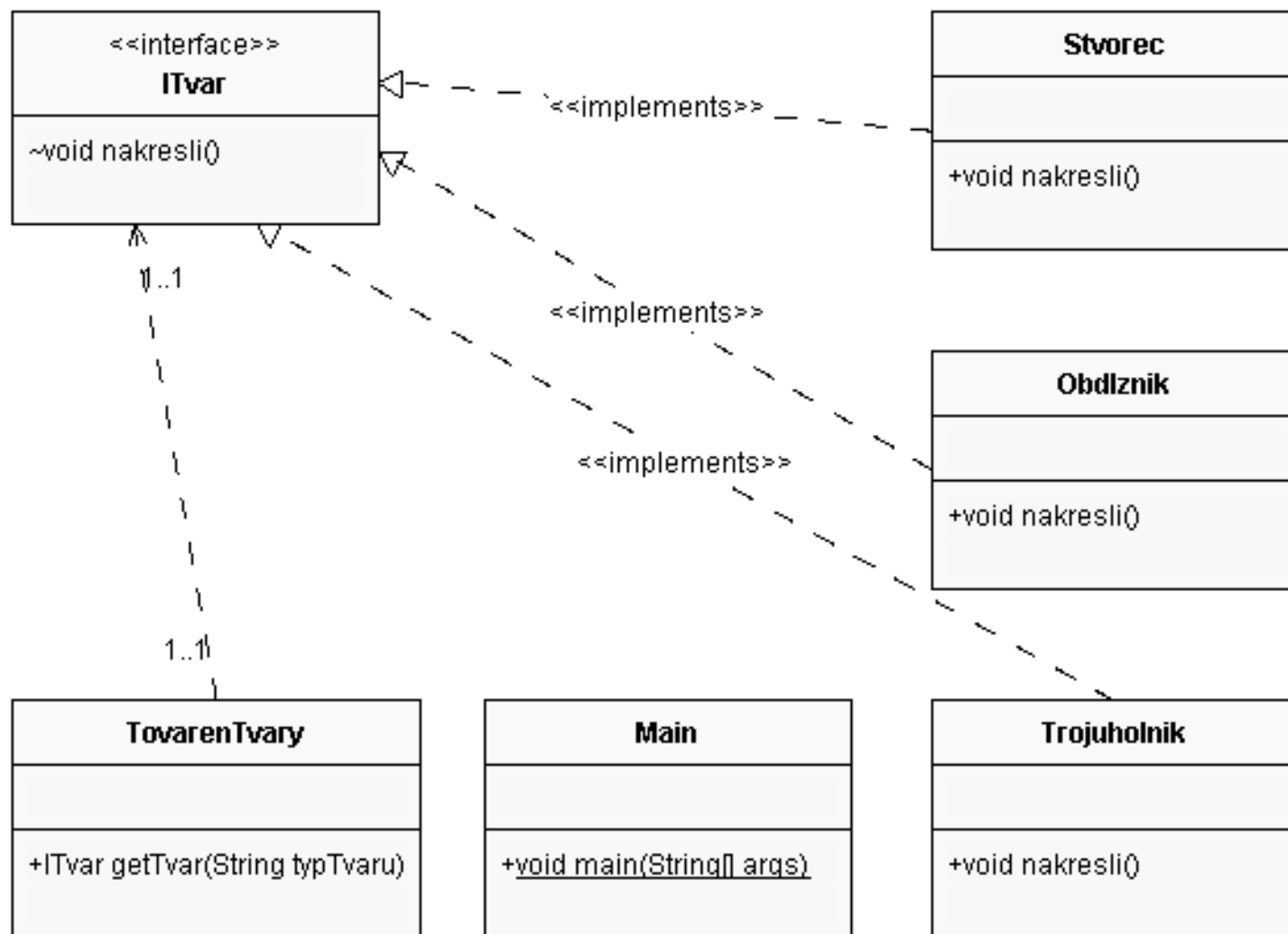
Továrenská metóda - Factory method

- Jeden z **najpoužívanějších vzorov**
- Spadá to kategórie tzv. tvorivých vzorov
- **Vytvárame objekt bez toho, aby sme odhalili logiku klientovi**
- Na novovytvorený objekt sa odkazujeme pomocou spoločného rozhrania

Továrenská metóda - Factory method

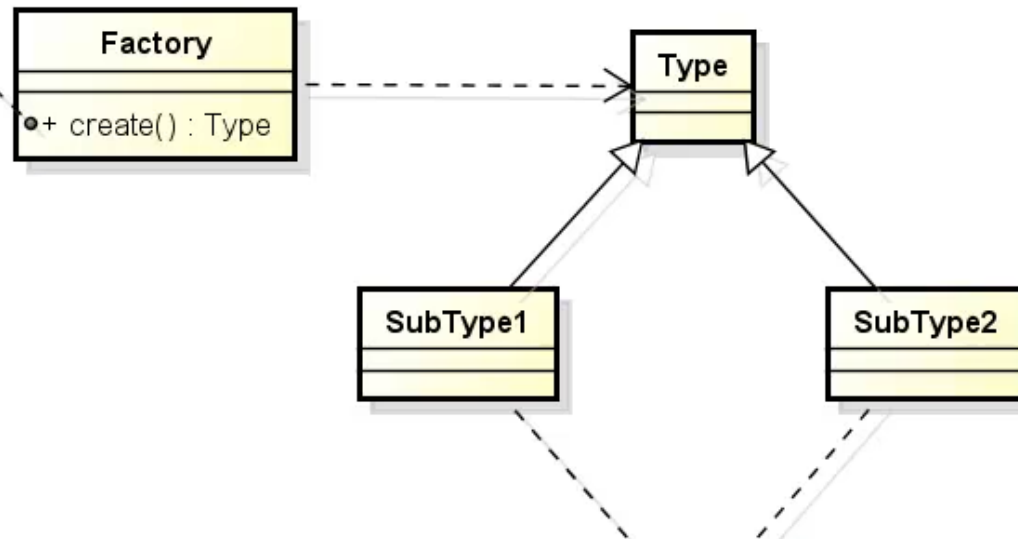
- Deklaruje **rozhranie s metódou pre získanie objektu**
- Rozhodnutie o **konkrétnom type vráteného objektu** ponecháva **na svojich potomkoch** t.j. **prekrývajúcich verziách metódy**
- **Nie je statická, ale inštančná!**
- Neriešim implementáciu a **sústredujem sa na rozhranie**

UML Továrenská metoda



UML Továrenská metóda

klient na vytvorenie
inštancie použije metódu
create()



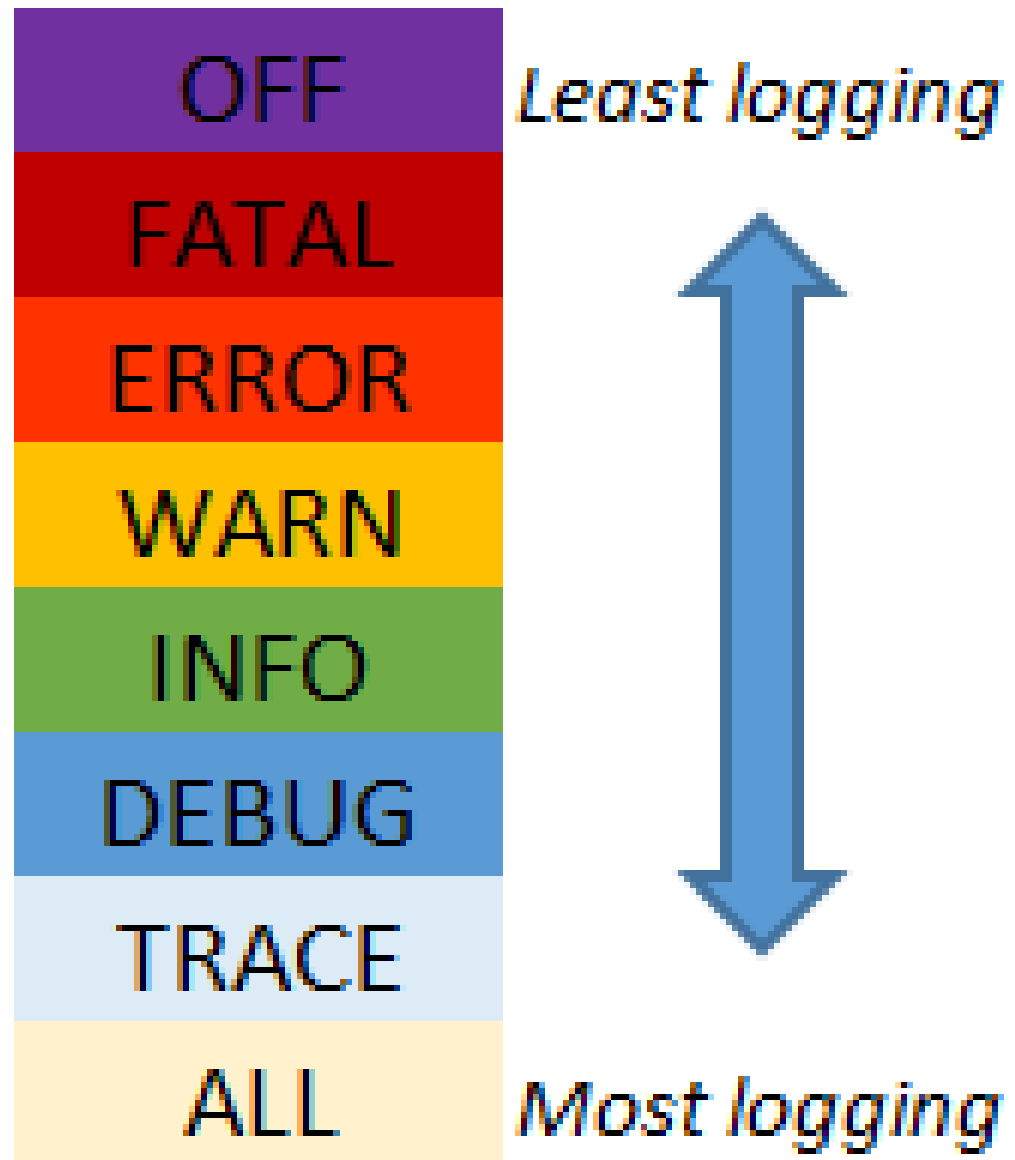
zákazník nemusí poznať
inštančný typ

Úrovne (Level)s

- Aká dôležitá je táto udalosť?
- Kto potrebuje vedieť o tejto udalosti?
- Vlastné úrovne

6 vstavaných úrovní

Log4j levels



Používanie úrovní v aplikáciách

INFO : User Alice logged in.

WARN : User Bob entered an invalid password.

ERROR: User Bob entered an invalid password 3 times, user locked out.

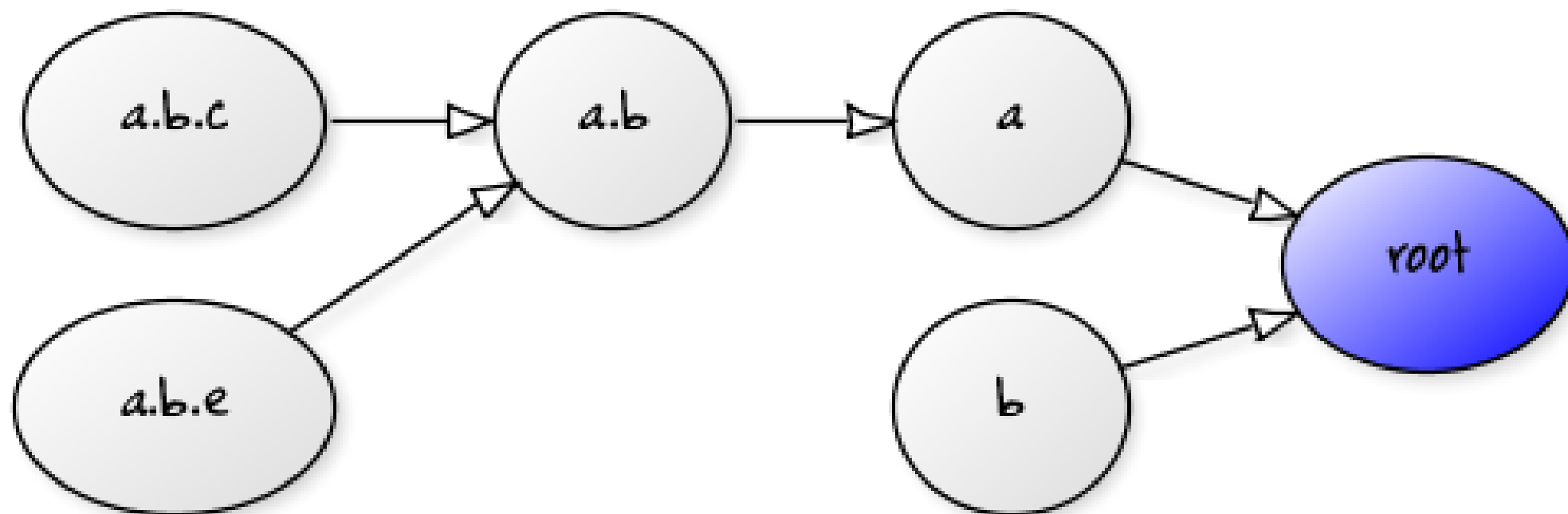
DEBUG: Loading user Bob from database JDBC

INFO : Emailed user Bob: 3 login attempts rejected, user locked out.

ERROR: Email to Alice bounced back; subject: Your weekly summary.

Pochopenie úrovni vs. Loggerov

- **Správa:**
 - Čo to hovorím svojmu publiku?
- **Úrovne**
 - Aké je to dôležité?
- **Loggeri**
 - Kto teraz hovorí?



Hierarchické loggery

- Loggery sú pomenované
- Názvy sú hierarchické: a.b.c, a, b, a, ...

Hierarchické loggery příklady

1. Kitchen
2. Kitchen.Oven
3. Kitchen.Oven.BottomOven
4. Kitchen.Oven.BottomOven.Door
5. Kitchen.Oven.BottomOven.Timer
6. Kitchen.Oven.BottomOven.Light
7. Kitchen.Oven.TopOven.Door
8. Kitchen.Oven.TopOven.Timer
9. Kitchen.Oven.TopOven.Light
10. Kitchen.Dishwasher
11. Kitchen.Refrigerator
12. Kitchen.Refrigerator.Door
13. Kitchen.Refrigerator.Filter
14. Kitchen.Refrigerator.Light
15. Kitchen.Refrigerator.Freezer.Door
16. Kitchen.Refrigerator.Freezer.Light
17. Atd'.

Logger konfigurácia

- Hierarchické názvy:
sk.stu.fiit.app.foo
- Nakonfigurujte Logger alebo celú hierarchiu

```
<Root level="warn">  
  <Logger name="javax.management"  
    level="INFO" />  
  <Logger name="sk.stu.fiit"  
    level="INFO" />  
  <Logger name="sk.stu.fiit.app.foo"  
    level="DEBUG" />  
  <Logger name="sk.stu.fiit.app.bar"  
    level="TRACE" />
```

Používanie Loggera

- `logger.debug("User {} logged in", user);`
- `logger.info("User {} logged in", user);`

Logger granulita

- **1 globálny logger**
 - Koreňový (root) logger
 - Nah.
- **Jeden na triedu**
 - Používaj FQCN
- **Jeden na inštanciu?**
 - V špecifických prípadoch
 - Používaj FQCN + info o inštancii
 - Dávaj pozor na veľa objektov s krátkou životnosťou

Šikovné názvy Loggerov

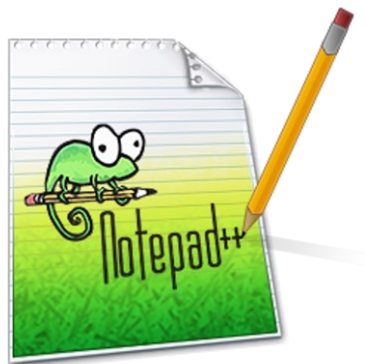
- Jeden na inštanciu
- Je nutné použiť názov Loggera na jednoznačnú identifikáciu
 - `com.example.server.socket.8080`
 - `com.example.rest.get.GetCustomer`

Aké IDE mám použiť?

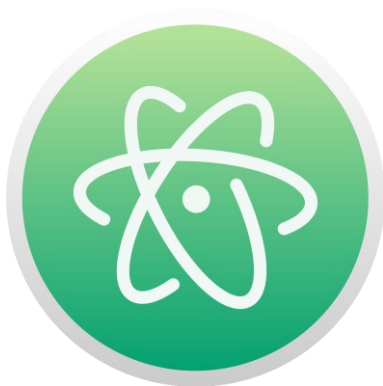


Integrated development environment

Aký editor mám použiť?



```
:::  
iLE88Dj. :jD88888Dj:  
.LGitE888D.f8GjjjL8888E;  
iE :8888Et. .G8888.  
;i E888, ,8888,  
D888, :8888:  
D888, :8888:  
D888, :8888:  
D888, :8888:  
888W, :8888:  
W88W, :8888:  
W88W: :8888:  
DGGD: :8888:  
:8888:  
:W888:  
:8888:  
E888i  
tW88D
```



Updaty a aktualizácie



jdk1.6.0_21



jdk1.6.0_29



jre6



jdk1.6.0_26



jdk1.7.0_21



jre7



**JAVA DEVELOPERS NEVER RIP,
THEY JUST GET GARBAGE COLLECTED.**

- ILIKE.NITTY-WITTY.COM

Čo sa oplatí prečítať?

Slovensko a Česko

- Albatrosmedia
- Kopp
- Grada
- Wolters Kluwer
- BEN
- Veda

Zahraničie

- O'Reilly
- Manning
- Packt
- Apress
- Wiley
- No Starch Press

YouTube tutorials

O'Reilly

Čo sa oplatí/neoplatí prečítať SK/CZ



Mistrovství a Kuchárka

Čo sa oplatí/neoplatí prečítať EN



Head First



Vývojári



Miroslav

Domov

Vytvoriť



Vývojári

Verejná skupina

Informácie

Diskusia

Oznámenia

Členovia

Podujatia

Videá

Fotky

Súbory

Hľadať v tejto skupine



Skratky

Podnikanie na Slove... 2

UK Manazment Externe...

Testovacia firma

VITA - Virtual It Academy

Startupisti 2

Rubyslava 2

Vývojári

Zobraziť viac



Ste člen

Upozornenia

Zdieľať

Viac

Napísať príspe...

Pridať fotku/vi...

Živé video

Viac



Napište něco...

Fotka/video

Divácka párty

Označiť priat...



NOVÁ AKTIVITA



Roland Mondek

10 h

POZVAŤ ČLENOV

+ Zadať meno alebo e-mailovú adresu...



ČLENOVIA

5 505 členov



POPIS

Skupina softvérových vývojárov. Táto skupina by mala byť miestom... Zobraziť viac

TYP SKUPINY

Všeobecné

VAŠE STRÁNKY



IT Academy



VITA - Virtual It Academy

KONTAKTY



Evka Rybárska



Jarmila Palenčárová



Stefan Orosi



Ivana Ivka Jasaňov

Hrá Word Blitz



Ivana Pavlíková



Martin Vanko



Lucia Kovačičová

4 h



Lošák Filip



Andrej Nejedlik



Gabika Zubrikova

SKUPINOVÉ KONVERZÁCIE



Vytvoriť novú skupinu

ĎALŠIE KONTAKTY (93)

Hľadať



Home

PUBLIC

 Questions

Tags

Users

COLLECTIVES

 Explore Collectives

FIND A JOB

Jobs

Companies

TEAMS

 Create free Team

Tags

A tag is a keyword or label that categorizes your question with other, similar questions. Using the right tags makes it easier for others to find and answer your question.

[Show all tag synonyms](#)

java

Java is a high-level object oriented programming language. Use this tag when you're having problems using or understanding the language itself. Thi...

1827413 questions

419 asked today, 2408 this week

javascript

For questions regarding programming in ECMAScript (JavaScript/JS) and its various dialects/implementations (excluding ActionScript). Note...

2335556 questions

779 asked today, 4877 this week

javaafx

The JavaFX platform enables developers to create and deploy Graphical User Interface (GUI) applications that behave consistently...

36355 questions

6 asked today, 50 this week

java-8

for questions specific to Java 8 which is version 8 (internal number 1.8) of the Java platform, released on 18 March 2014. In most cases, you should also...

22076 questions

9 asked today, 40 this week

java-stream

for questions related to the use of the Stream API. It was introduced in Java 8 and supports functional-style operations on streams of values, such...

10293 questions

5 asked today, 26 this week

java-native-interface

The Java Native Interface (JNI) gives both the ability for JVM implementations to run system native code and the ability for native code t...

9404 questions

12 asked this week, 38 this month

rx-java

RxJava – Reactive Extensions for the JVM – a library for composing asynchronous and event-based programs using observable sequence...

6796 questions

6 asked this week, 27 this month

javascript-objects

for questions related to JavaScript objects.

6151 questions

20 asked this week, 118 this month

java.util.scanner

A simple text scanner in the JDK which can parse primitive types and strings using regular expressions.

javaafx-8

JavaFX 8 (previously named JavaFX 3) introduces a new API for JavaFX technology. JavaFX 8 supports 3D and brings up a Retina-Display Support. It ...

java-me

Java Platform, Micro Edition, or Java ME, is a Java platform designed for embedded systems.

facebook-javascript-sdk

Facebook's JavaScript SDK provides a rich set of client-side functionality for accessing Facebook's server-side API calls. It can collaborate with any SDK...

Popular

Name

New

Mrkni na náš YouTube kanál a daj odber

→ [WWW.YOUTUBE.COM/C/IT-ACADEMYSK](https://www.youtube.com/c/IT-ACADEMYSK) ←