

# Vývoj pokročilých aplikácií

## Úvod do predmetu



doc. Ing. **Jozef Kostolný**, PhD.  
Fakulta riadenia a informatiky  
Žilinská univerzita v Žiline  
jozef.kostolny@fri.uniza.sk

Ing. **Martin Mazúch**  
Fakulta riadenia a informatiky  
Žilinská univerzita v Žiline  
martin.mazuch@fri.uniza.sk



**ŽILINSKÁ UNIVERZITA V ŽILINE**  
Fakulta riadenia  
a informatiky

# Obsah

- Základné informácie o predmete
- Java - verzie a novinky
- Zmena prístupu v programovaní architektúr
- Nástroje – IntelliJ IDEA, GitHub





## Informácie o predmete

- 6BI0050 - **Vývoj pokročilých aplikácií (VPA)**
- Prednáška: štvrtok 8:00 – RC009
- Cvičenia: streda 8:00 – RA222 - 20

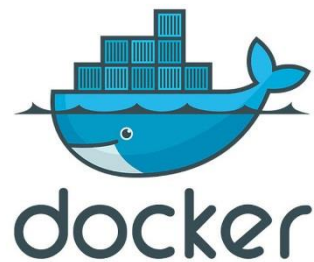
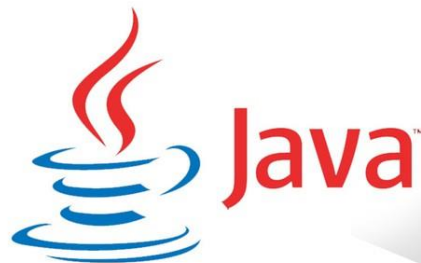


# O čom bude predmet

- Vývoj aplikácií v Jave
- Demonštrácie v iných jazykoch a frameworkoch
  - Jakarta, Spring, Spring boot, C#, Blazor
- Využitie architektúry microservice
  - Docker
- Práca s DB
- Priblíženie vývoja aplikácií do praxe



# Technológie



# Hodnotenie

- Semester (100 b):
  - semestrálna práca – odovzdanie do 13. týždňa
  - obhajoba 13. týždeň a prvý týždeň skúškového
- Skúška (100 b):
  - písomná časť s obhajobou



# Minimálne požiadavky na semestrálnu prácu

## Semestrálna práca

- Využitie Javy, C# ....
- Buildovanie pomocou **Mavenu** ...
- Uložený v **GITe**
- Využitie SQL/NoSQL DB
- Výhoda využitie microservice architektúry

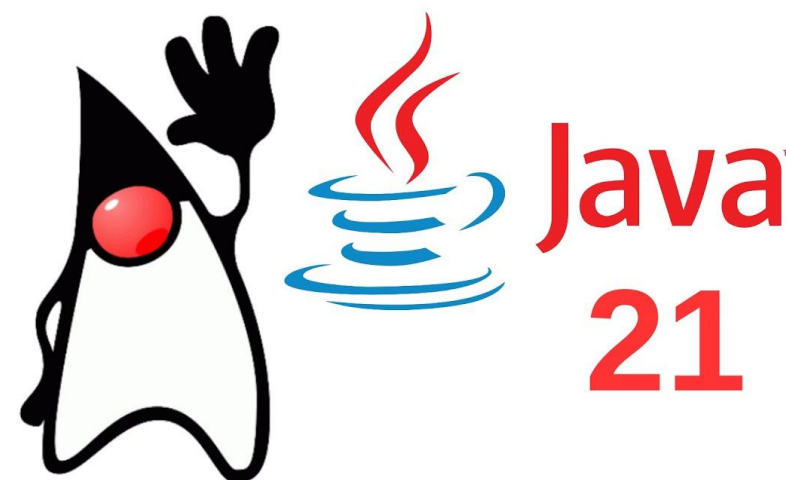
## Odovzdávanie:

- najneskôr do 13. týždňa



# Java história

- JDK 1.0 (January 23, 1996)
- JDK 1.1 (February 19, 1997)
- J2SE 1.2 (December 8, 1998)
- J2SE 1.3 (May 8, 2000)
- J2SE 1.4 (February 6, 2002)
- J2SE 5.0 (September 30, 2004)
- Java SE 6 (December 11, 2006)
- Java SE 7 (July 28, 2011)
- Java SE 8 (March 18, 2014)
- Java SE 9 (September 21, 2017)
- Java SE 10 (March 20, 2018)
- Java SE 11 (September 25, 2018)
- Java SE 12 (March 2019)
- Java SE 13 (September 2019)
- ...
- Java SE 19 (September 2022)
- Java SE 20 (March 2023)
- Java SE 21 LTS (September 2023)
- Java SE 22 – marec 2024





# Java 8 – prelomová verzia



- Vydané: 18.03.2014



# Lambda

## Hlavný koncept“

- zaobchádzať s funkcionalitou ako s argumentom metódy alebo sa pozerať na kód ako na dáta



**Lambda calculus** - výraz vygenerovaný nasledujúcou gramatikou, ktorý môže znamenať definíciu funkcie, aplikáciu funkcie, premennú alebo výraz v zátvorkách:

**$expr \rightarrow \lambda var . expr \mid expr \ expr \mid var \mid (expr)$**

V Java 8 lambda calculus je hlavne metóda bez deklarácie obvyčajne zapísaná v tvare **(parameters) -> { body }**.  
Např.

```
(int x, int y) -> { return x + y; }  
x -> x * x  
( ) -> x
```

- Lambda môže mať 0 alebo viac parametrov, oddelené čiarkami a ich typ môže byť explicitne deklarovaný alebo odvodený z kontextu
- Zátvorky sa nepoužívajú pre 1 parameter.
- Telo môže mať 1 alebo viac príkazov
- Jeden príkaz sa neuzatvára do zátvoriek.



# Nashorn



- Javascriptový engine v Jave
- umožňuje zvýšiť výkon a bezproblémovú interoperabilitu Javy a JavaScriptu.
- Použiteľnosť v kóde alebo v príkazovom riadku
- Možnosť dynamického programovania
- Nahradzuje starší engine Rhino
- <https://github.com/openjdk/nashorn>



# Nashorn



```
Command Prompt - jjs
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Abid>jjs
jjs> var nums = new java.util.Stack();
jjs> nums.push(0);
0
jjs> nums.push(1);
1
jjs> nums.push(2);
2
jjs> nums.push(3);
3
jjs> nums.push(4);
4
jjs> print(nums);
[0, 1, 2, 3, 4]
jjs> nums.pop();
4
jjs> print(nums);
[0, 1, 2, 3]
jjs>
```

```
1 | ScriptEngine engine = manager.getEngineByName("nashorn");
```

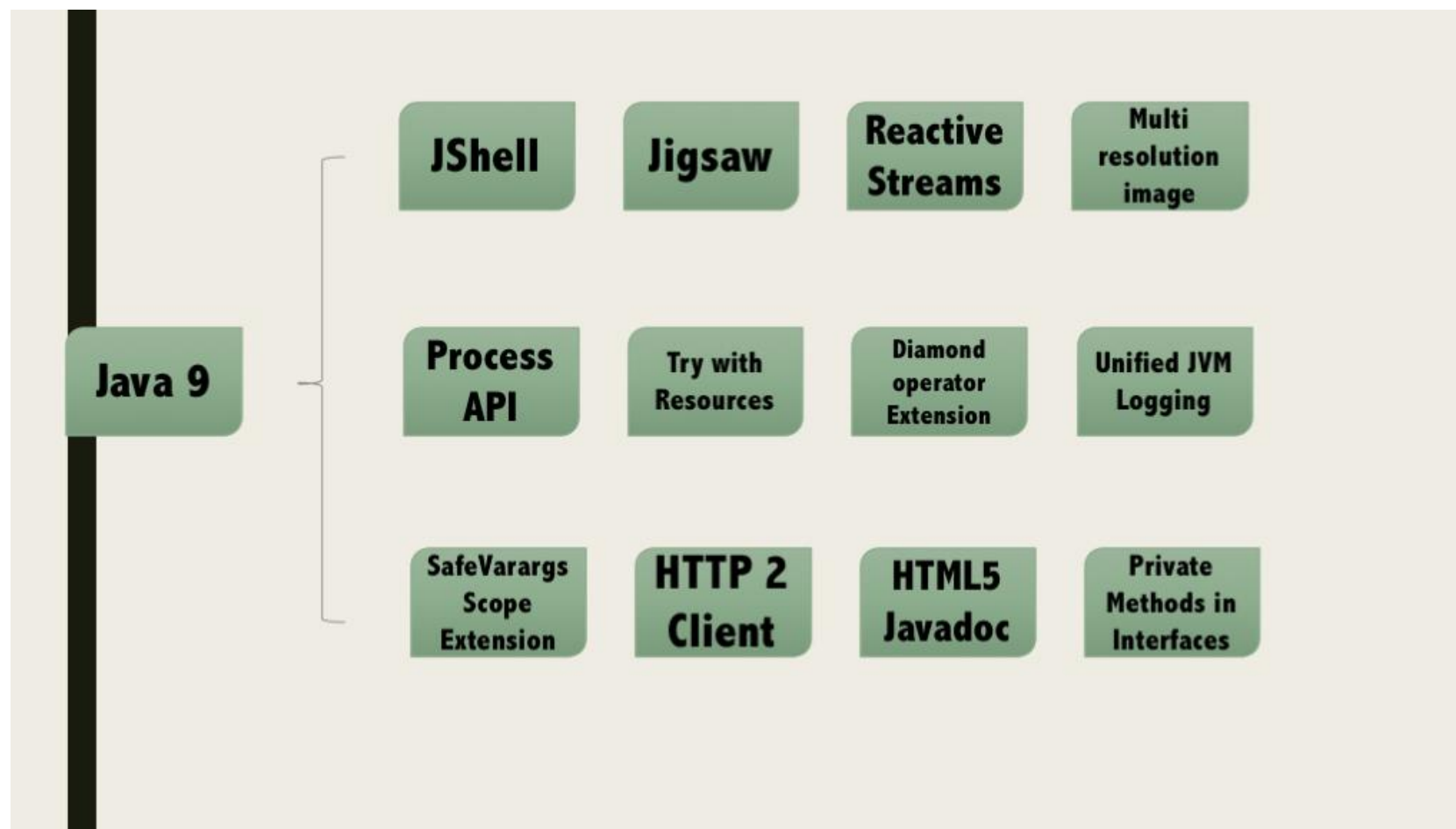
```
1 | String script = "var welcome = 'Hello '"
2 |                 + "welcome += 'World'"
3 |                 + "welcome";
```

```
1 | String result = (String) engine.eval(script);
2 | System.out.println(result);
```

```
Problems @ Javadoc Declaration Console
<terminated> Main [Java Application] C:\Program Files\Java\jre8\bin\javaw.
Hello World
```



# Java 9



- Vydané: september 2017

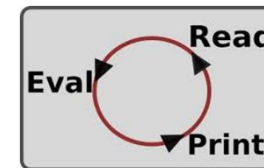


# Linkovanie - jlink

- Nové možnosti pri spájaní závislostí medzi modulmi
- Vytvorenie minimálneho prostredia
- Len to čo naozaj pre beh aplikácie je potrebné
- Nie je potrebné načítavanie celého JDK
- Minimalizácia bežiaceho obrazu a optimalizácia aplikácie



# JShell



- Interaktívny javovský REPL –
  - Read-Eval-Print-Loop
- Spustenie javovského kódu priamo z konzoly
- vyhľadávanie API
- Testovanie funkcií

```
► jshell
| Welcome to JShell -- Version 9-ea
| For an introduction type: /help intro

jshell> "abc".matches("a[bc]{2}")
$1 ==> true

jshell> █
```





# JavaDoc

- Večná dilema: použiť JavaDoc alebo Google??
- Pridanie vyhľadávania v API
- Pridanie kompatibility s HTML5
- Informácie o priradení do JDK verzie

The screenshot shows the Java Platform API Specification website. The top navigation bar includes links for OVERVIEW, MODULE, PACKAGE, CLASS, USE, TREE, DEPRECATED, INDEX, and HELP. The right side of the header indicates the version: Java™ Platform Standard Ed. 9, DRAFT 9-ea+162-jigsaw-nightly-h6252-20170328. Below the navigation bar, there are links for PREV, NEXT, FRAMES, NO FRAMES, and ALL CLASSES. A search bar on the right contains the text 'Map'. The main content area is titled 'Java™ Platform, Standard Edition 9 API Specification'. Below this, there is a section for 'Modules' with a table listing modules and their descriptions. A 'Types' sidebar on the right lists various classes and interfaces related to the search term 'Map'.

**Modules**

Module	Description
java.activation	Defines the Java
java.base	Defines the found
java.compiler	Defines the Lang
java.corba	Defines the Java

**Types**

- java.util.**Map**
- java.nio.channels.FileChannel.**MapMode**
- java.nio.**MappedByteBuffer**
- java.util.**AbstractMap**
- javax.swing.Action**Map**
- javax.swing.plaf.Action**MapUIResource**
- javax.activation.Command**Map**
- javax.swing.ComponentInput**Map**
- javax.swing.plaf.ComponentInput**MapUIResource**
- java.util.concurrent.ConcurrentHash**Map**
- java.util.concurrent.Concurrent**Map**
- java.util.concurrent.ConcurrentNavigable**Map**
- java.util.concurrent.ConcurrentSkipList**Map**
- java.util.**Map.Entry**





# Privátne metódy rozhrania

- Interface môže obsahovať aj správanie okrem popisovačov
- Odpadá nutnosť refaktorovať metódy a vkladať rovnaký kód do všetkých potomkov
- Metódy zostávajú private

```
public interface MyInterface {  
    void normalInterfaceMethod();  
    default void interfaceMethodWithDefault() { init(); }  
    default void anotherDefaultMethod() { init(); }  
    // This method is not part of the public API exposed by MyInterface  
    private void init() { System.out.println("Initializing"); }  
}
```



# Viacverzionové JAR-ko

- Možnosť verzionovania JAR
- Príprava tried pre rôzne verzie Javy
- Výsledná aplikácia je kompatibilná bez nutnosti inštalovania novej verzie

```
multirelease.jar
├── META-INF
│   └── versions
│       └── 9
│           └── multirelease
│               └── Helper.class
├── multirelease
│   ├── Helper.class
│   └── Main.class
```



# Java 10



- Vydané: 20.3.2018



# Kľúčové slovo var

Non-denotable Types

Intersection types

```
var list = List.of(1, 2.0, "3");
```



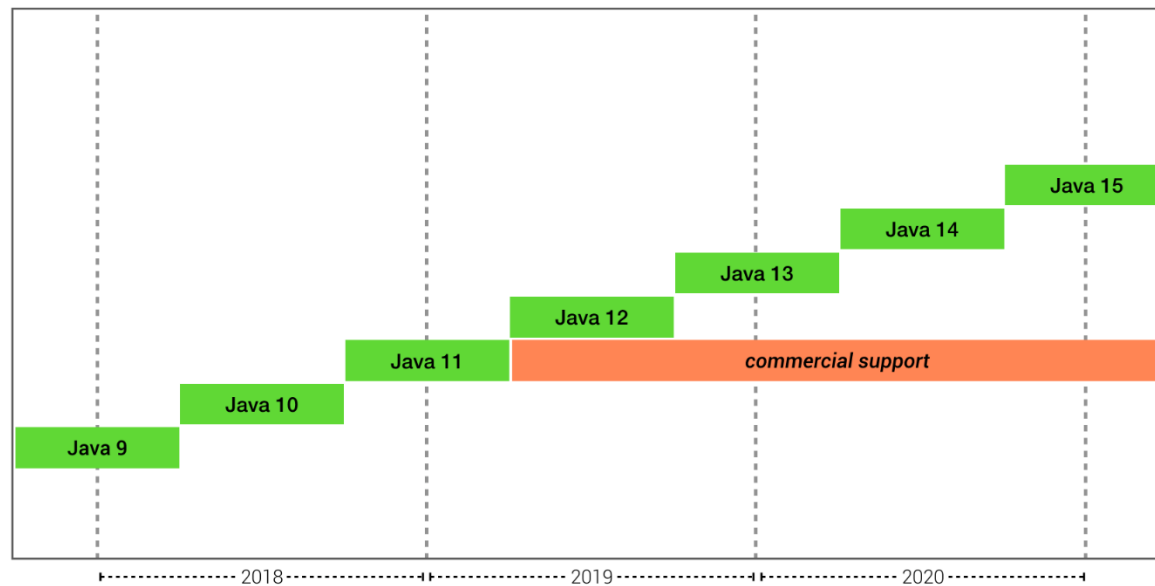
→ List<? extends Serializable & Comparable<..>>

All of these types may show up in errors



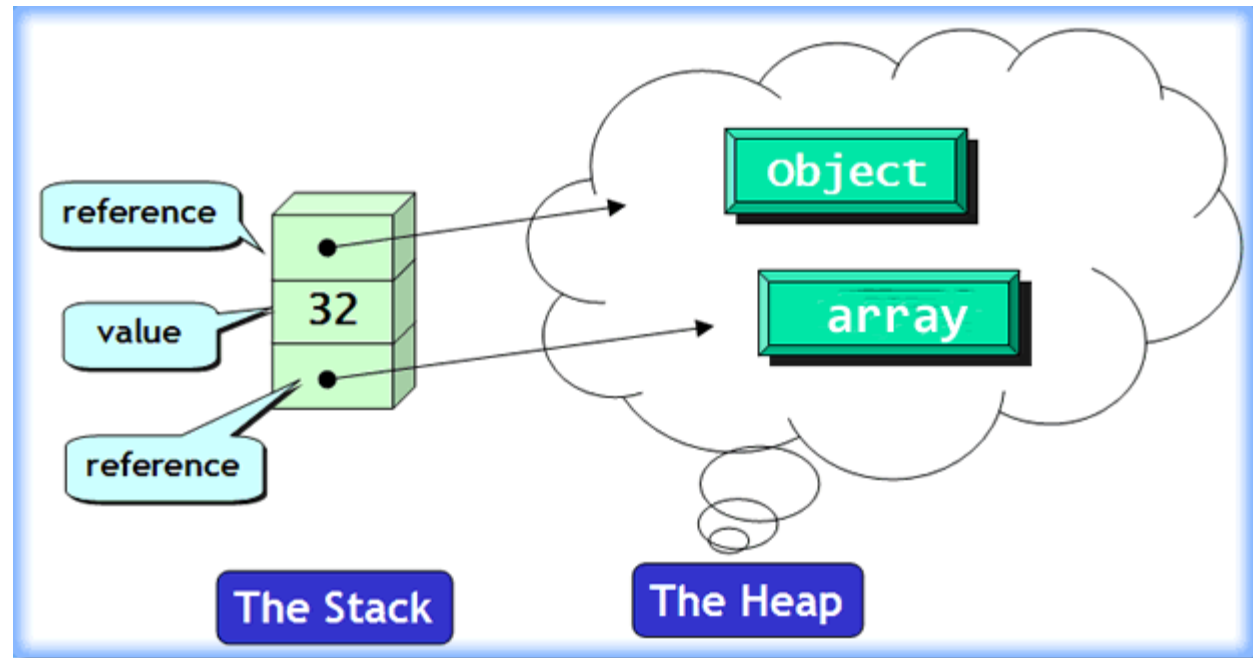
# Nový model verzionovania

- Prechod na polročný interval



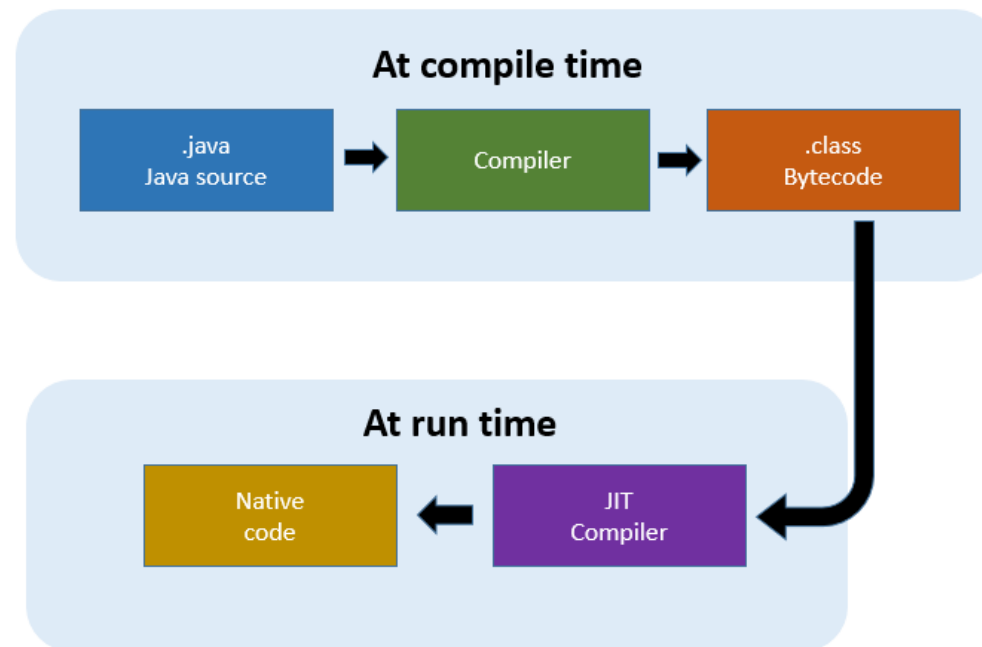
# Rozhranie Garbage-Collectora

- Izolácia kódu



# JIT kompilátor

- JIT – „Just-in-time“
- Experimentálny Graal – JIT kompilátor







# Jednosúborový program

Zdroják:

```
public class HelloJavaScripts {  
  
    public static void main(String[] args) {  
        System.out.println("Hello, Java scripts!");  
    }  
}
```

Spustenie:

```
$ java HelloJavaScripts.java  
> Hello, Java scripts!
```



# Shebang v Java

## Zadefinovanie knižnice:

```
#!/opt/jdk-11/bin/java --source 11
public class HelloJavaScripts {

    public static void main(String[] args) {
        System.out.println("Hello, Java scripts!");
    }
}
```

## Spustenie:

```
chmod +x hello-java-scripts
```

```
./hello-java-scripts
```



## Ďalšie +/-

- Lamba výrazy s var
- Rozšírená funkcionálna String-u
- Kompilácia a spustenie už iba jeden krok

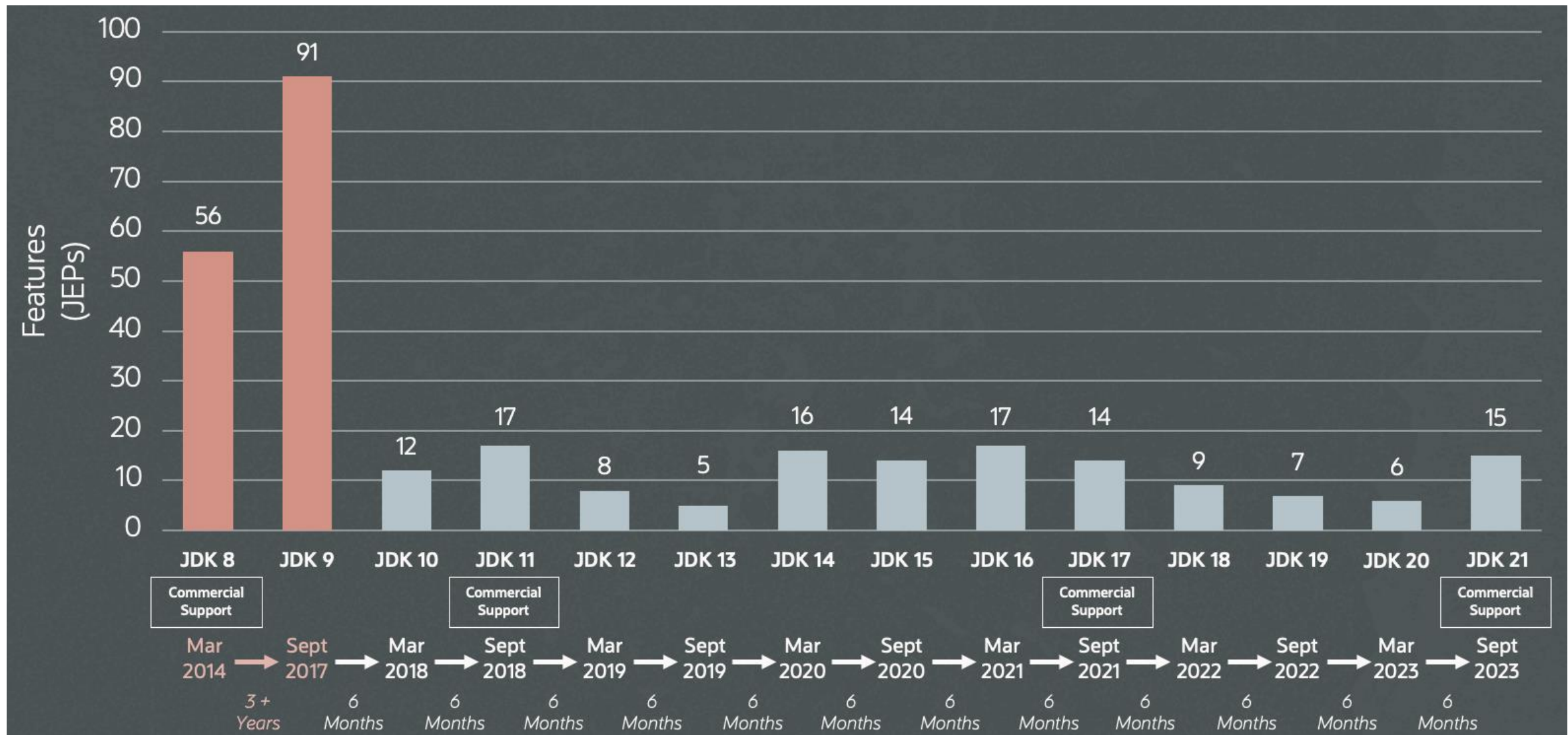
`java HelloJavaScripts.java`

- JVM sa spúšťa pomaly pri „bootovaní“

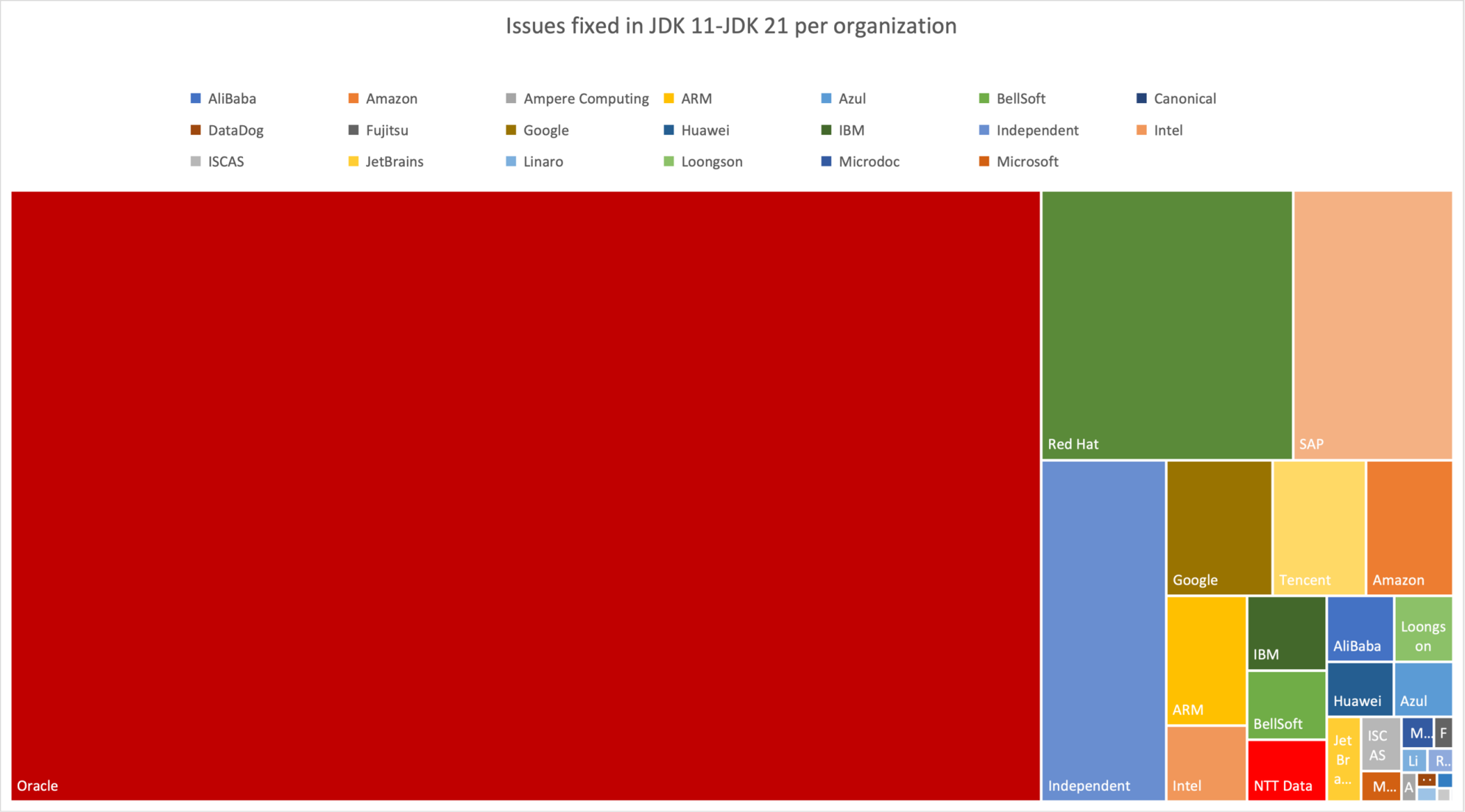
`./hello-java-scripts`



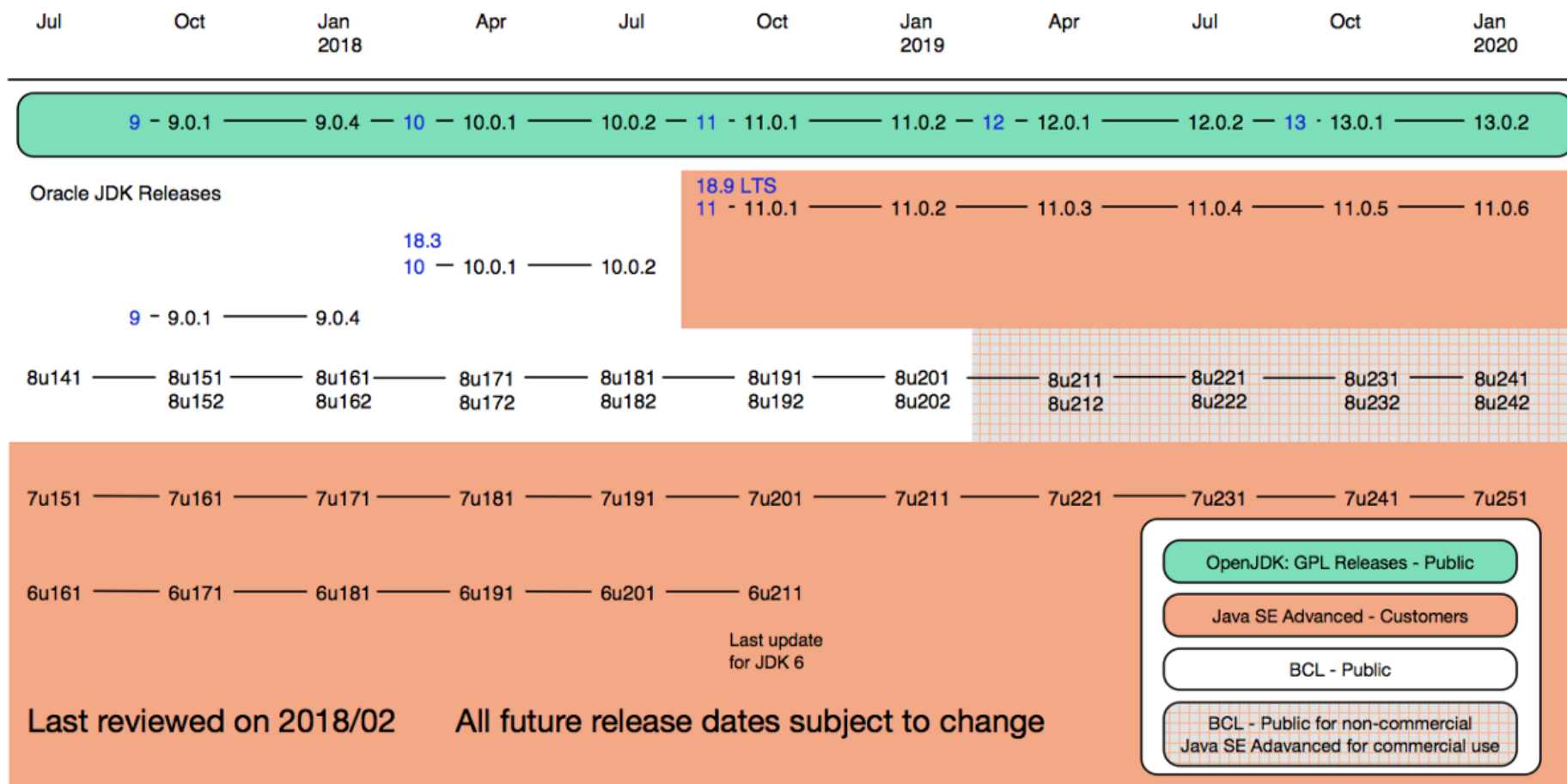
# Dopad zmeny vývojového cyklu



# Dopad zmeny vývojového cyklu



# Je Java stále zadarmo?



# JAVA SE Models

Models Available  
Prior to January 2019

Java SE Advanced

Java SE Advanced Desktop

Java SE Suite

- Upfront License + Annual Support
- No Public Updates for Java SE 8 after Jan 2019

Models Available  
July 2018 and Beyond

Java SE  
Subscription

Java SE  
Desktop  
Subscription

- Monthly subscription
- Includes Public Updates for Java SE 8 or Later
- 1 – 3 Year Term

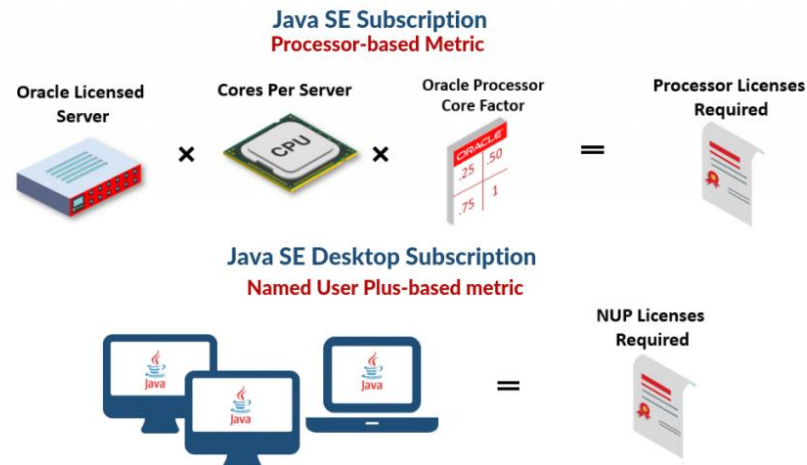


Table 1: Legacy Java Subscription Fees vs. Java SE Universal Subscription Fees

Example	Licensable Quantities	Java SE Desktop and Java SE Subscription Annual Net Fees	Java SE Universal Subscription Annual Net Fee	% Increase/(Decrease)
1	25,000 NUP 10,000 Processors	\$375,000 \$1,500,000	\$2,025,000	8%
2	19,000 NUP 2,500 Processors	\$342,000 \$525,000	\$1,881,000	117%
3	49,500 NUP 5,000 Processors	\$742,500 \$900,000	\$3,118,500	90%
4	1,000 NUP 15,000 Processors	\$24,000 \$2,250,000	\$144,000	(94%)
NUP = Named User Plus				

Source: Gartner (February 2023)



## New **Java-19** Features

- 1 Record Patterns
- 2 Vector API
- 3 Virtual Threads
- 4 Pattern Matching
- 5 Structured Concurrency
- 6 Linux/RISC-V Port
- 7 Foreign Function and Memory





# Java 20



- zdieľanie immutable dáta v rámci vlákien
- záznamové vzory – v JDK 19 iba náhľad
- API pre cudzie funkcie
- fix virtuálnych vlákien
- zjednodušenie viacvláknového programovania
- čo tam asi nebude...
  - univerzálne generiká, asynchrónne rozhranie VM, API na sledovanie zásobníka



# Java 21

- sekvenčné kolekcie
- zlepšenie výkonu Garbage collectoru
- virtuálne vlákna
- deprecated 32bitový port pre x86
- API pre kryptografické funkcie
- pre Javu 22
  - vektor API, matching patern Instanceof, Record classes, string templates....
  - <https://www.makb183.com/2023/10/java-22-what-to-expect.html>

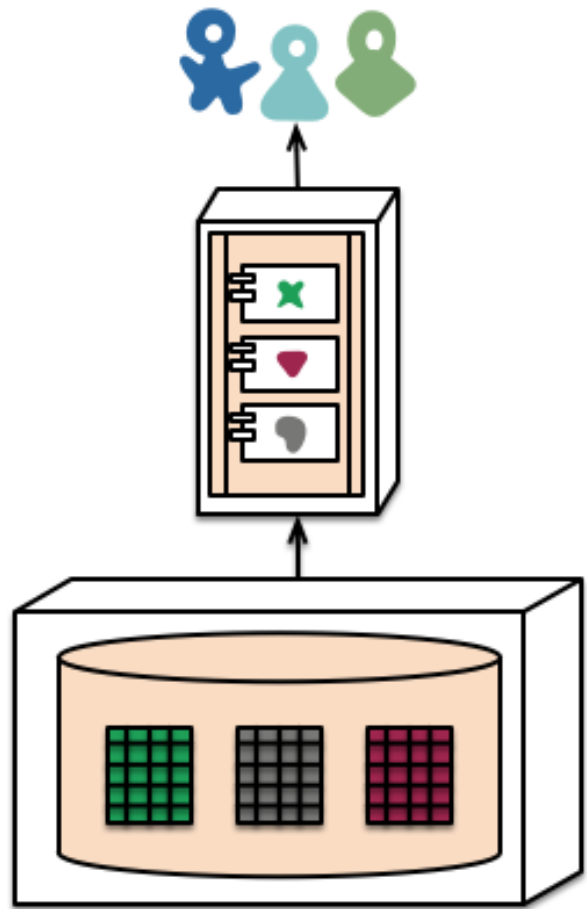


# FIRST TIME WORKING WITH ARCHITECTURE SOFTWARE

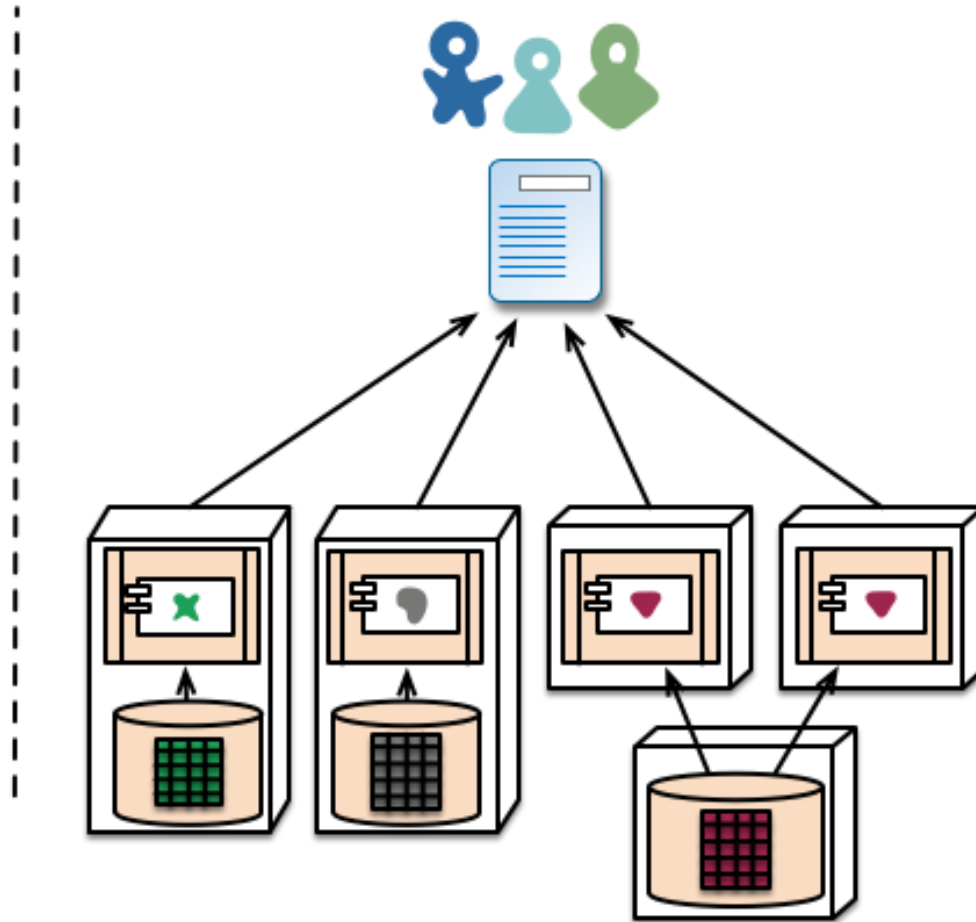




# Zmena prístupu v programovaní architektúr



monolith - single database

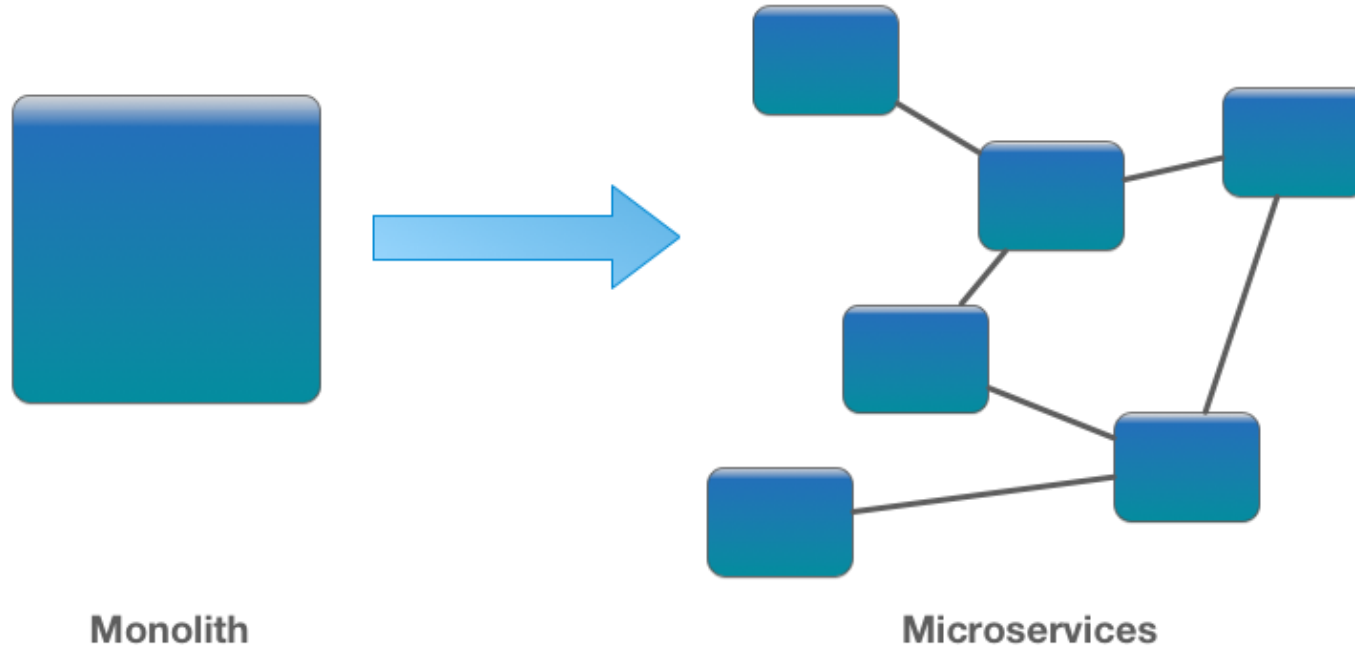


microservices - application databases



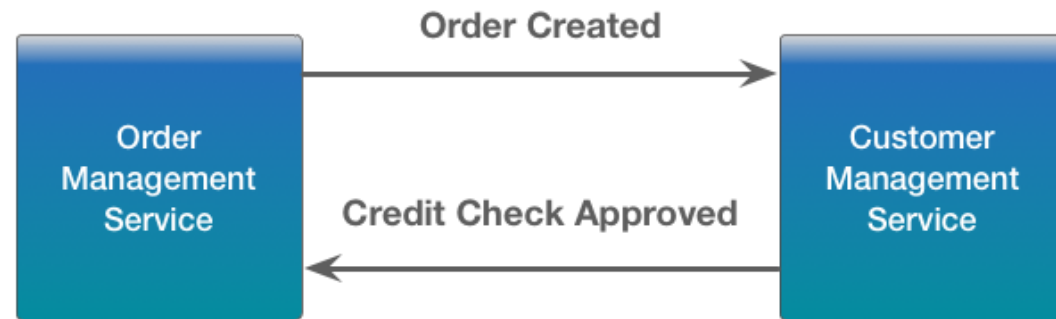
# Prečo microservice?

- Umožňujú rýchlejšie produkovanie inovácií
- Podniky musia inovovať oveľa rýchlejšie, aby zostali pred konkurenciou.



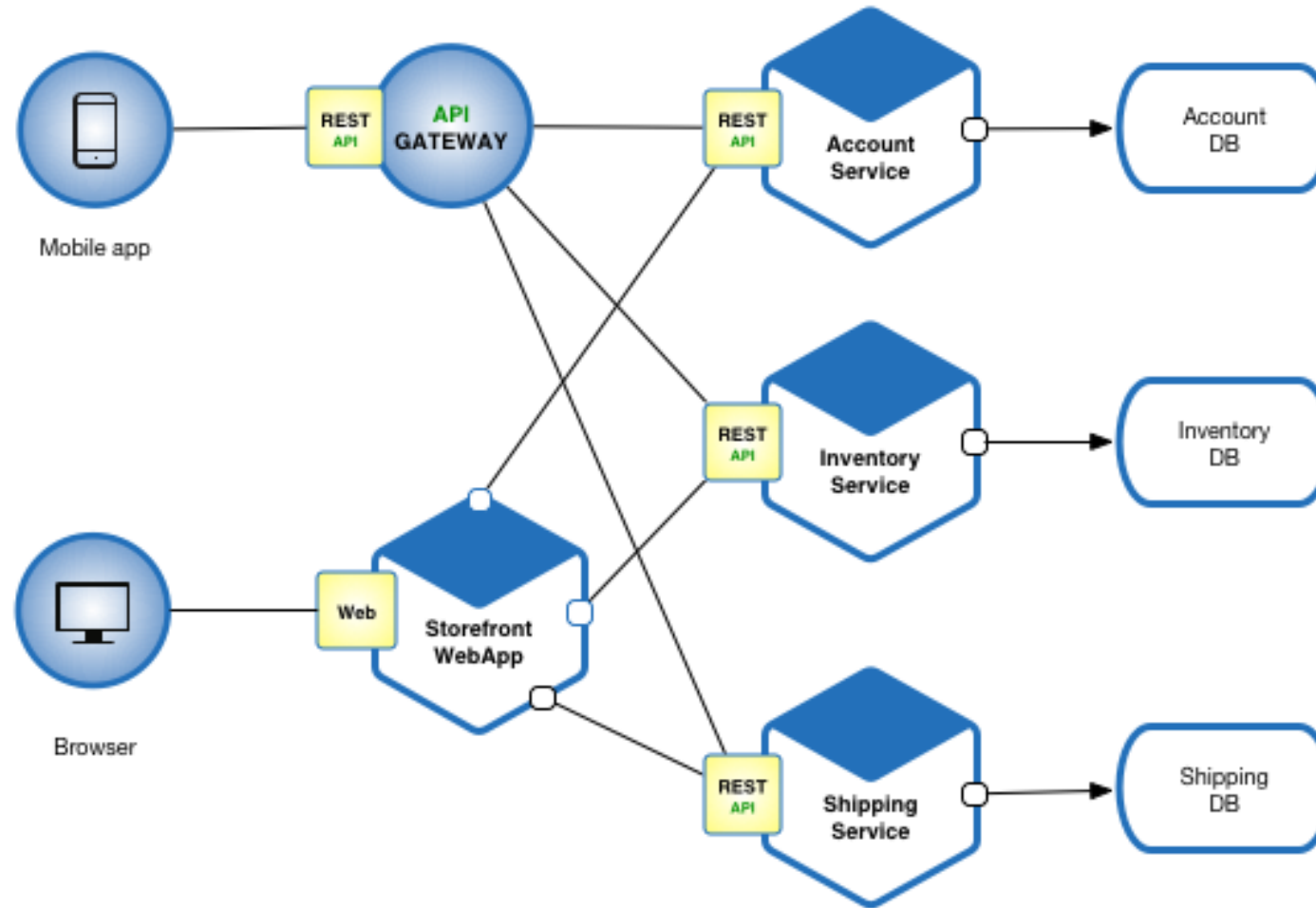
# Problémy s microservice

- Pri vývoji sa musí riešiť problém riadenia distribuovaných dát.
- Každá microservice má svoju vlastnú súkromnú databázu.
- Väčšinou pomocou udalostne riadenej architektúre vývoja
- Jednoduchšia aktualizácia služieb
- Vždy funkčná aplikácia



# Príklad

<http://eventuate.io/exampleapps.html>



# Výhody microservice

- Umožňuje nepretržité poskytovanie a nasadzovanie veľkých a zložitých aplikácií:
  - Testovateľnosť
  - Nasaditeľnosť
  - Organizovanie vývojárskeho tímu
- Každá microservice je relatívne malá:
  - Jednoduchšie na pochopenie
  - IDE pracuje rýchlejšie
  - Aplikácia sa spúšťa rýchlejšie
- Zlepšenie izolovateľnosti chýb
- Nezávislosť od technológie

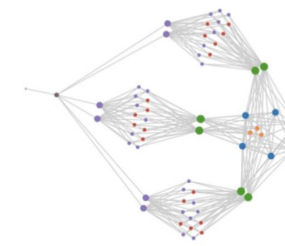




# Používa sa vôbec?

## Aplikovanie:

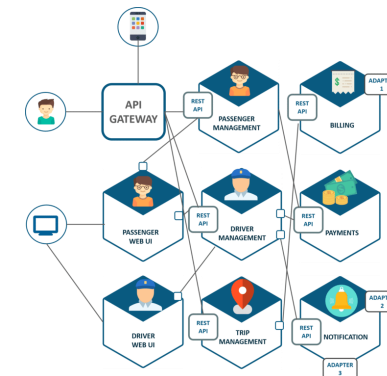
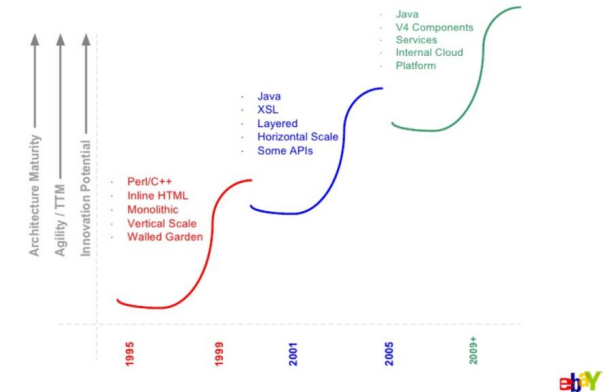
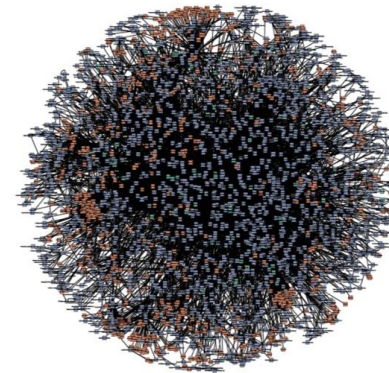
- Netflix
  - streamovacia služba 30% záťaže internetu,
  - viac ako 800 typov zariadení
  - Obsluha milióna volaní denne
- Amazon
  - Pôvodne ako dvojúrovňová architektúra
  - Migrácia a rozširovanie bolo nákladné
  - Dnes obsluha 100-500 služieb
- eBay
  - Zmena na viacero nezávislých služieb
  - Zlepšenie logistiky predaja
  - Efektívnejšia komunikácia na DB
- Uber
  - využívanie API gateway a nezávislých služieb



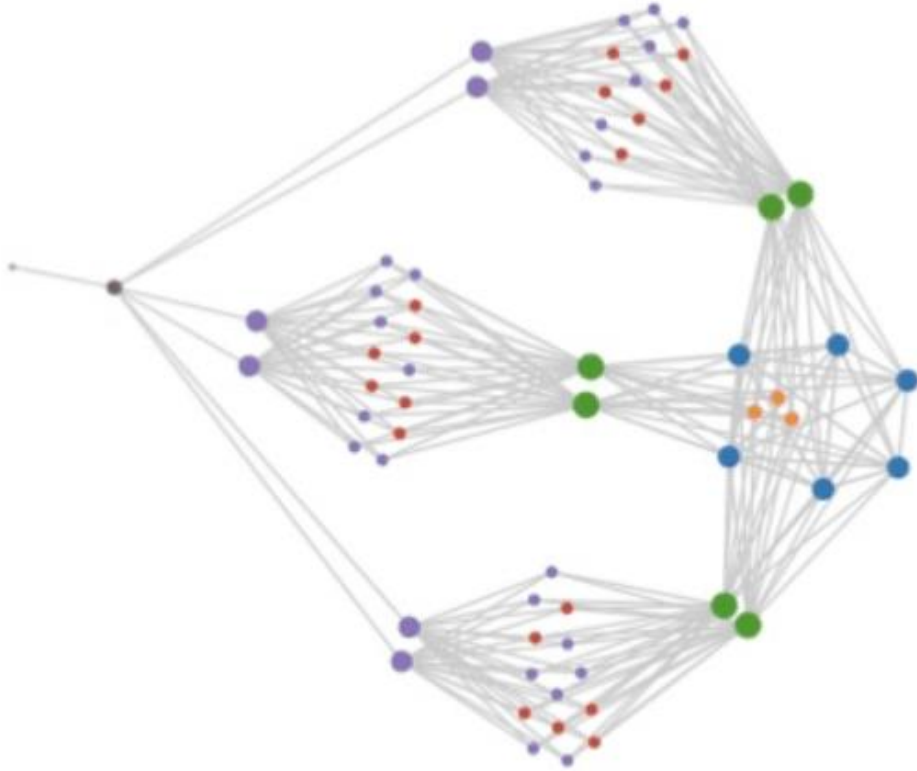
Simplified Architecture



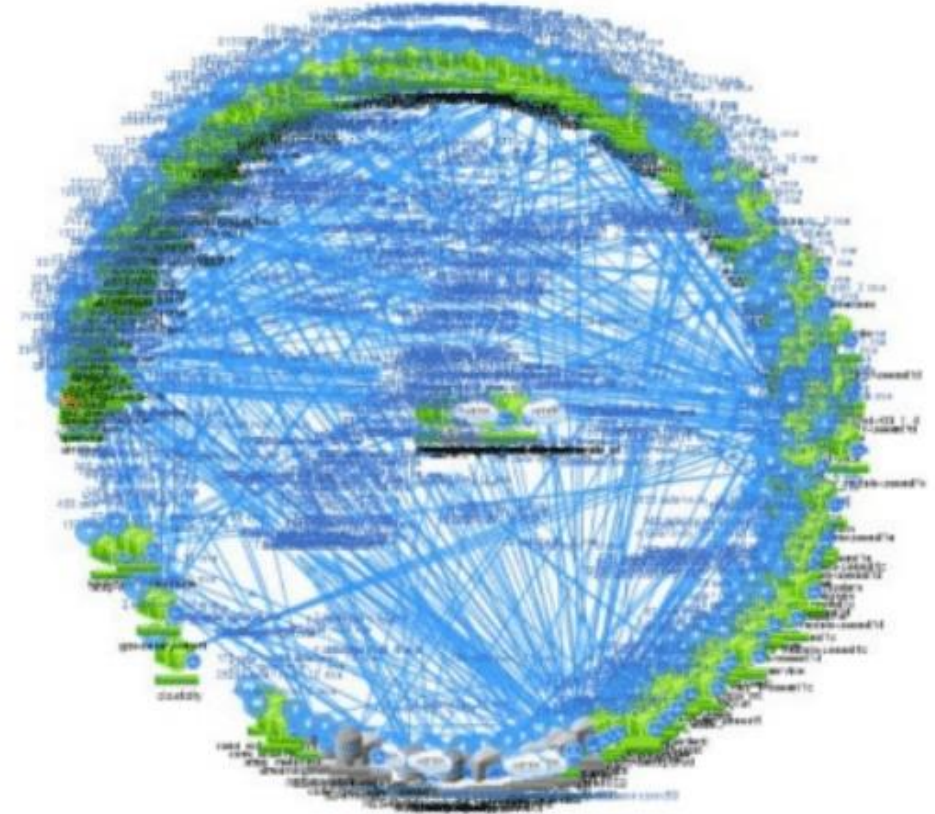
Actual Architecture



# Používa sa vôbec?



Simplified Architecture



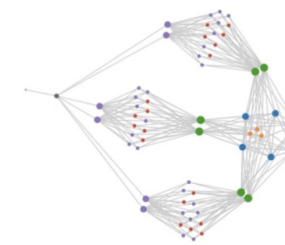
Actual Architecture



# Používa sa vôbec?

## Aplikovanie:

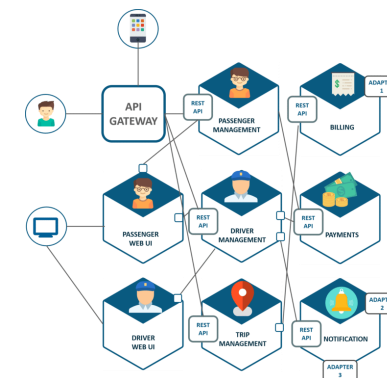
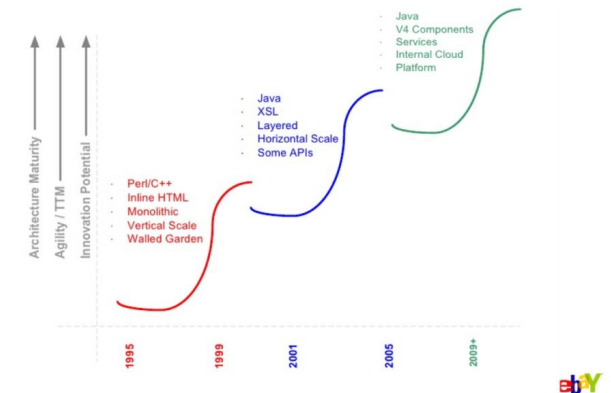
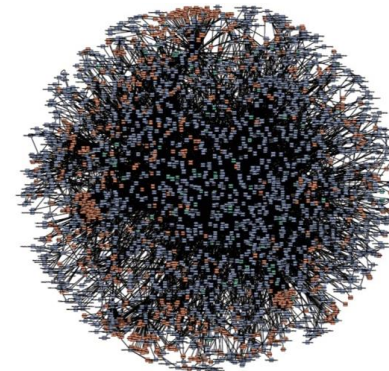
- Netflix
  - streamovacia služba 30% záťaže internetu,
  - viac ako 800 typov zariadení
  - Obsluha milióna volaní denne
- Amazon
  - Pôvodne ako dvojúrovňová architektúra
  - Migrácia a rozširovanie bolo nákladné
  - Dnes obsluha 100-500 služieb
- eBay
  - Zmena na viacero nezávislých služieb
  - Zlepšenie logistiky predaja
  - Efektívnejšia komunikácia na DB
- Uber
  - využívanie API gateway a nezávislých služieb



Simplified Architecture



Actual Architecture

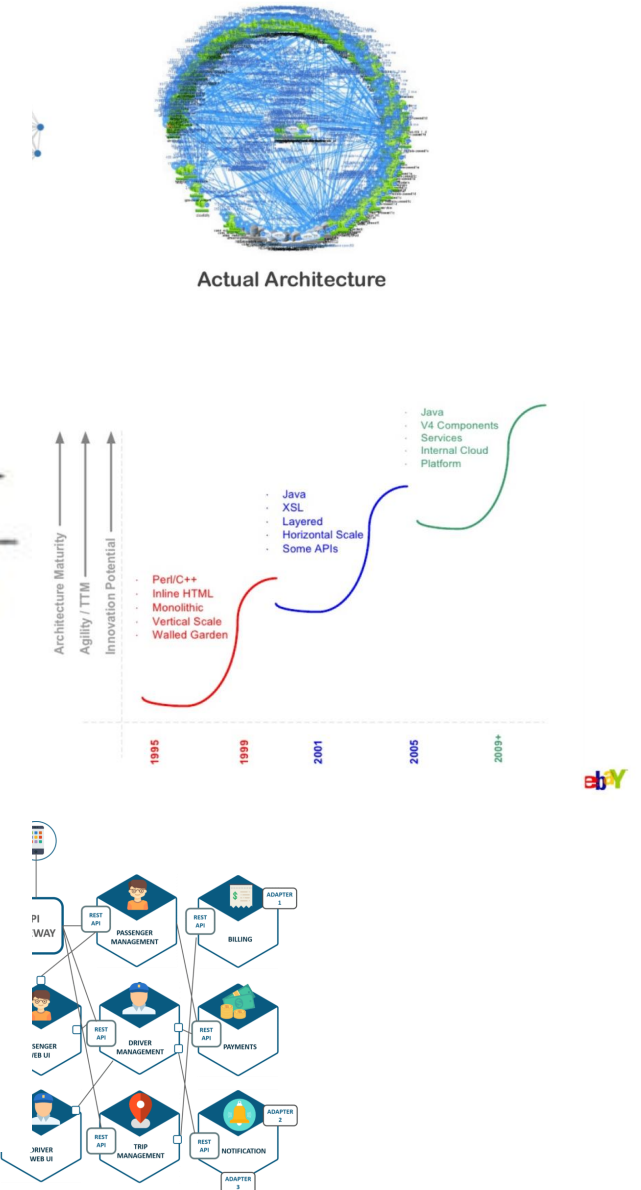
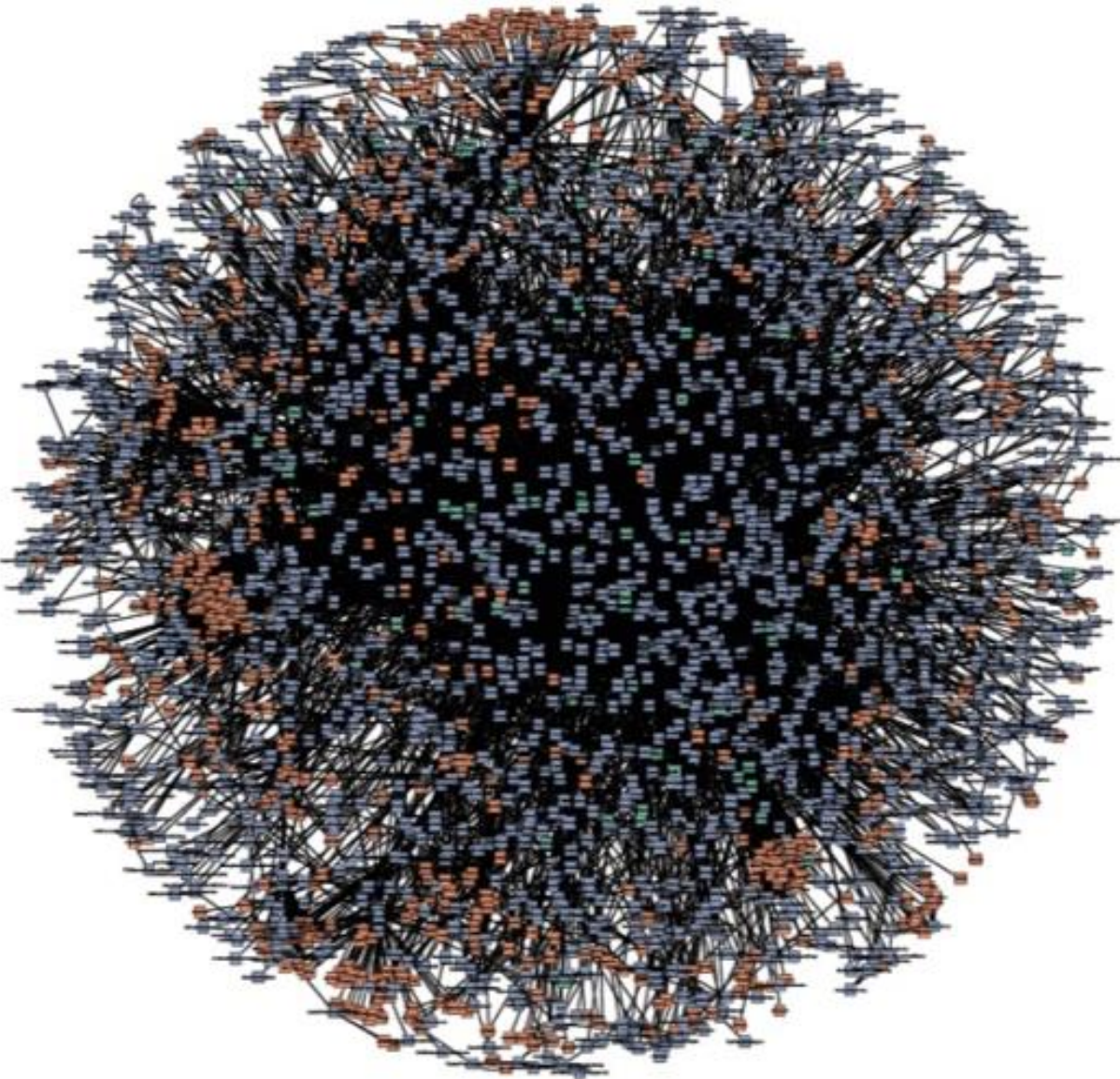




# Používa sa vôk

## Aplikovanie:

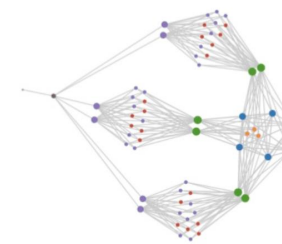
- Netflix
  - streamovacia
  - viac ako 800 t
  - Obsluha biliór
- Amazon
  - Pôvodne ako c
  - Migrácia a roz
  - Dnes obsluha
- eBay
  - Zmena na viac
  - Zlepšenie logi
  - Efektívnejšia k
- Uber
  - využívanie AP



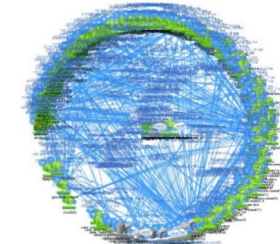
# Používa sa vôbec?

## Aplikovanie:

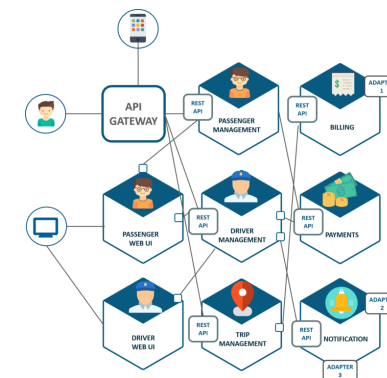
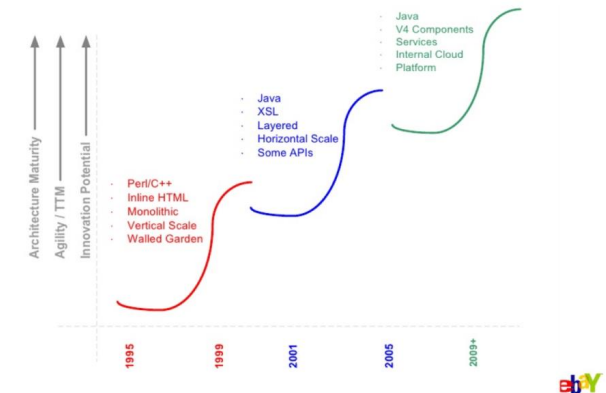
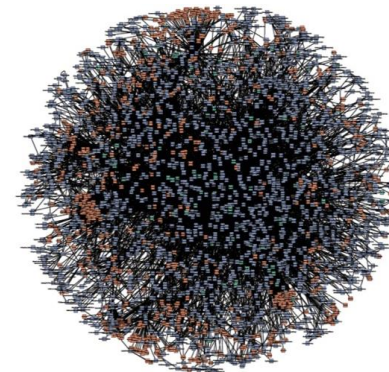
- Netflix
  - streamovacia služba 30% záťaže internetu,
  - viac ako 800 typov zariadení
  - Obsluha milióna volaní denne
- Amazon
  - Pôvodne ako dvojúrovňová architektúra
  - Migrácia a rozširovanie bolo nákladné
  - Dnes obsluha 100-500 služieb
- eBay
  - Zmena na viacero nezávislých služieb
  - Zlepšenie logistiky predaja
  - Efektívnejšia komunikácia na DB
- Uber
  - využívanie API gateway a nezávislých služieb



Simplified Architecture



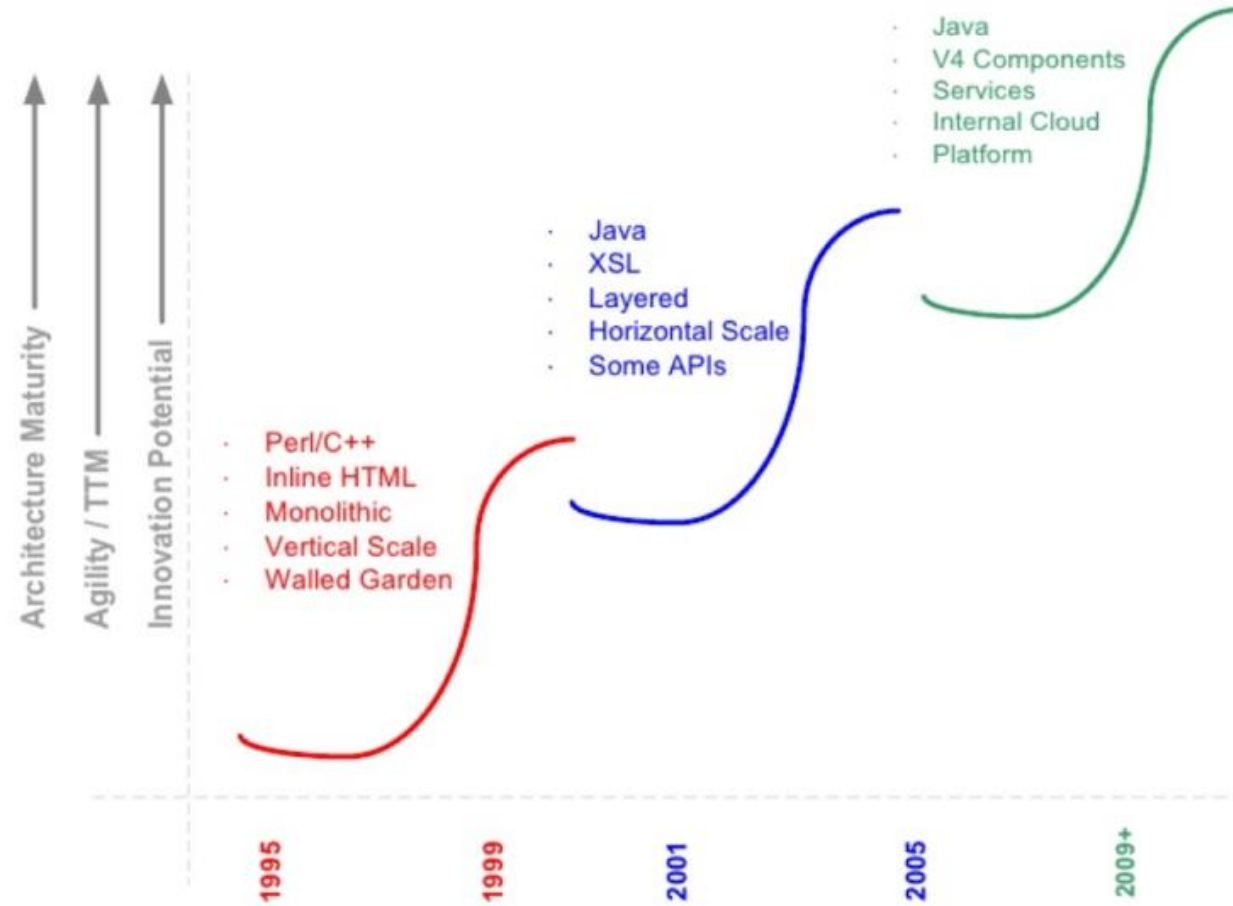
Actual Architecture



# Používa sa vôbec?

Aplikc

- 
- 
- 
- 

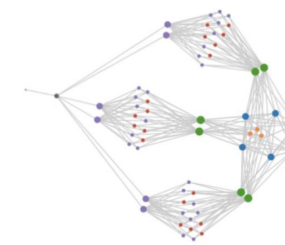




# Používa sa vôbec?

## Aplikovanie:

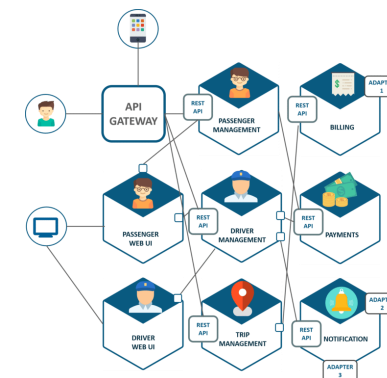
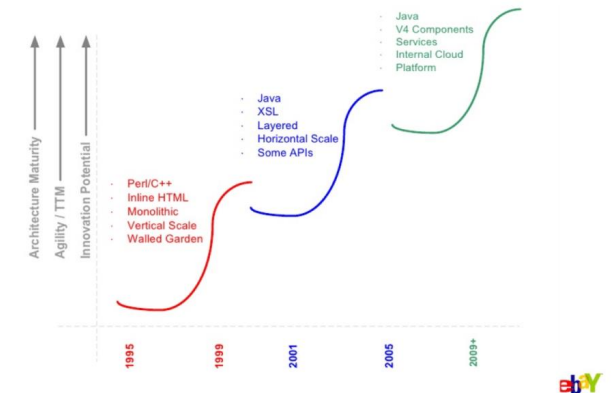
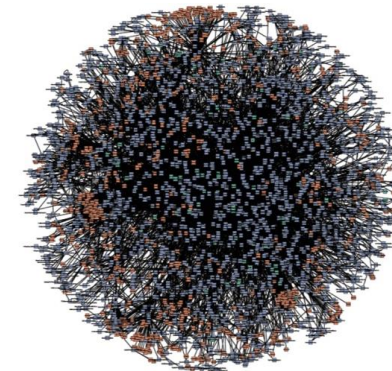
- Netflix
  - streamovacia služba 30% záťaže internetu,
  - viac ako 800 typov zariadení
  - Obsluha milióna volaní denne
- Amazon
  - Pôvodne ako dvojúrovňová architektúra
  - Migrácia a rozširovanie bolo nákladné
  - Dnes obsluha 100-500 služieb
- eBay
  - Zmena na viacero nezávislých služieb
  - Zlepšenie logistiky predaja
  - Efektívnejšia komunikácia na DB
- Uber
  - využívanie API gateway a nezávislých služieb



Simplified Architecture



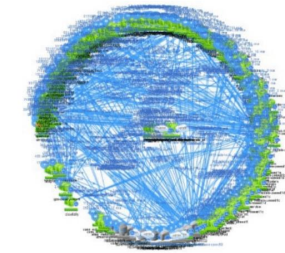
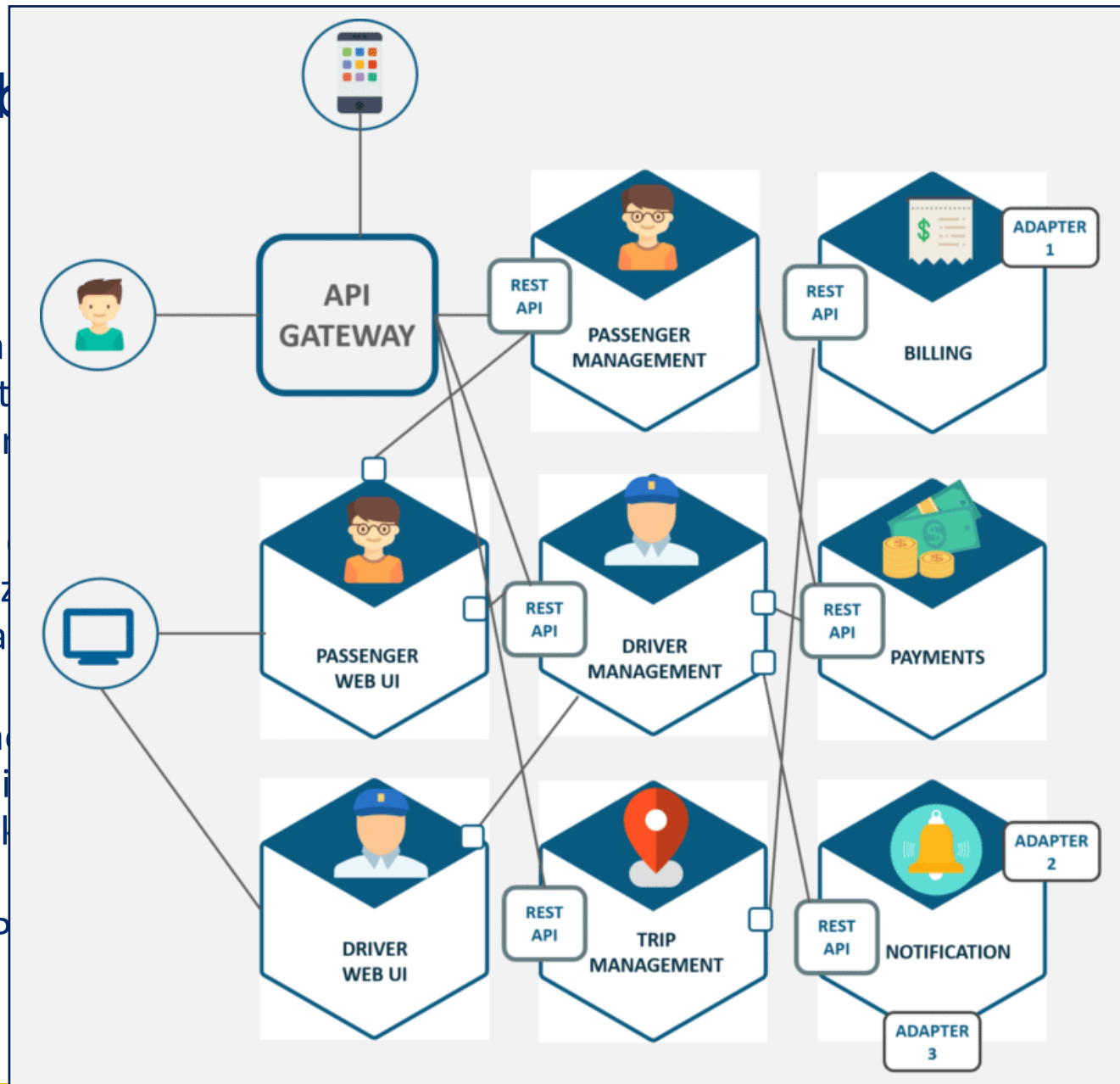
Actual Architecture



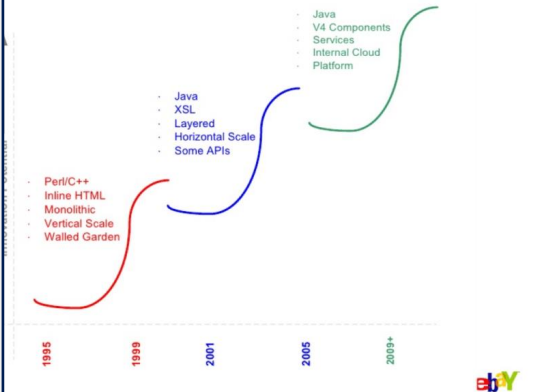
# Používa sa vôk

## Aplikovanie:

- Netflix
  - streamovacia
  - viac ako 800 t
  - Obsluha bilió
- Amazon
  - Pôvodne ako
  - Migrácia a roz
  - Dnes obsluha
- eBay
  - Zmena na viac
  - Zlepšenie logi
  - Efektívnejšia k
- Uber
  - využívanie AP



Actual Architecture

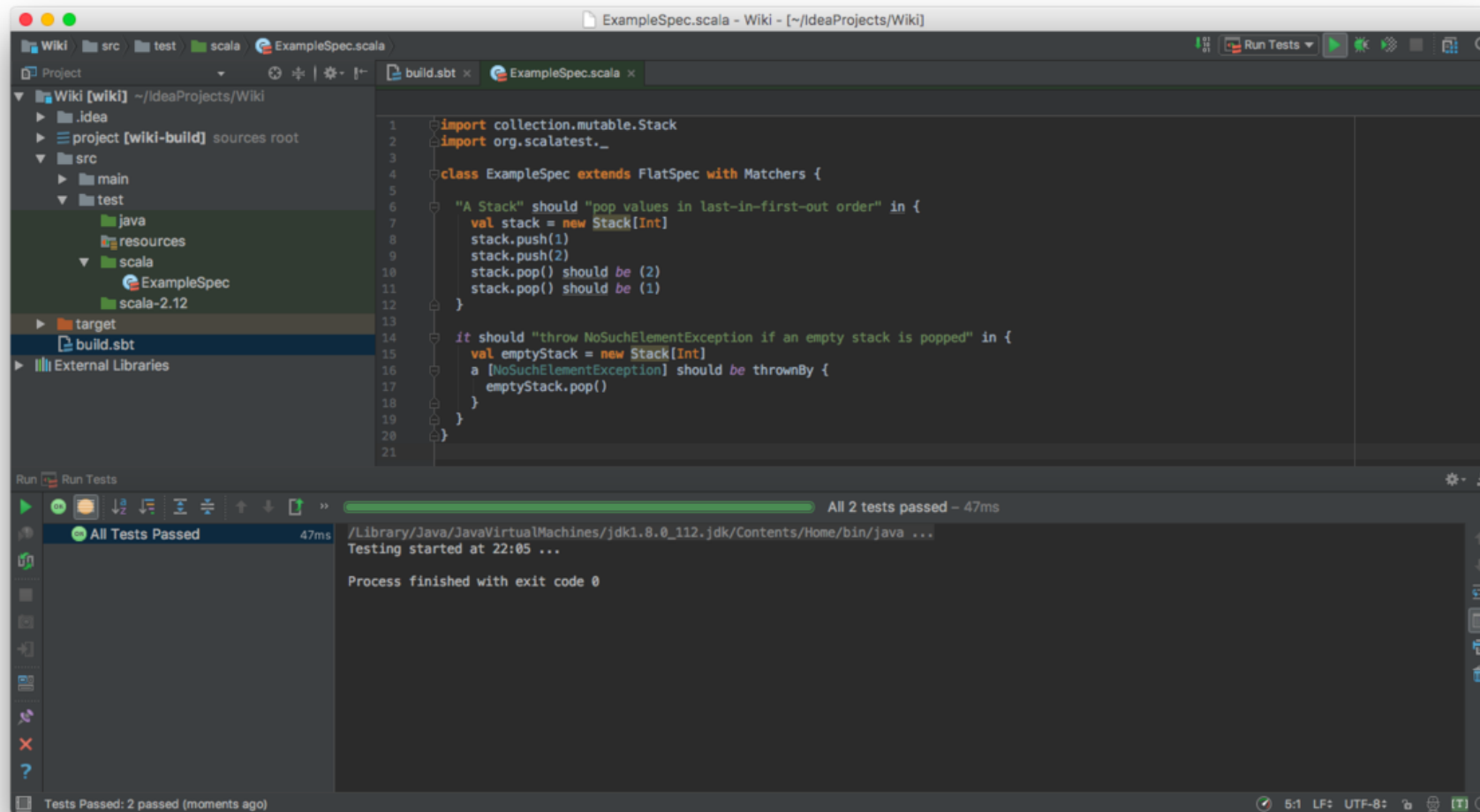




# Nástroje



# IDE – Integrated Development Enviroment

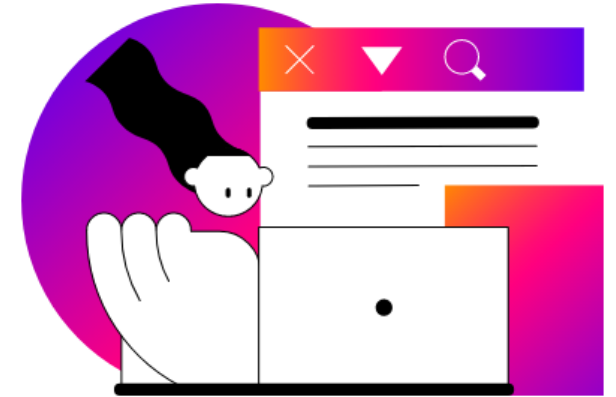


<https://www.jetbrains.com/idea/>



# Verzie

- Voľne dostupná verzia Community
- Pre vývoj Java EE - Ultimate
- Študentské balíčky:
- <https://www.jetbrains.com/student/>



## Options to get a free educational license



Official university email  
address



ISIC or ITIC card



GitHub Student Developer  
Pack [account](#)



Student's/teacher's ID or other  
official document



# JetBrains Student Pack



## ReSharper Ultimate

ReSharper, ReSharper C++, dotCover, dotTrace and dotMemory bundled in one license



## IntelliJ IDEA Ultimate

A complete toolset for JVM-based web, mobile and enterprise development



## AppCode

Smart iOS/macOS IDE



## CLion

Cross-platform C/C++ IDE



## DataGrip

Database and SQL IDE



## GoLand

Capable and Ergonomic Go IDE



## PhpStorm

IDE for Web & PHP



## PyCharm

Powerful Python & Django IDE



## Rider

Cross-platform .NET IDE



## RubyMine

IDE for Ruby and Rails



## WebStorm

Smart JavaScript IDE



# Správa verzií - GIT

## Version control

Zmysel verzionovania softvéru,  
princípy a prínos.

## GIT

Základné fungovanie verzionovacieho  
systému GIT. Výhody a nevýhody.



## GIT príkazy

Init, clone, fetch, commit, branch,  
merge, pull .

## Príklady

Príklady jednotlivých príkazov.



**IS IT BAD TO JOIN DATA IN  
AN APPLICATION, AND NOT IN A  
DATABASE IN MICROSERVICE ARCHITECTURE?**



**Ďakujem za pozornosť**

doc. Ing. **Jozef Kostolný**, PhD.  
Fakulta riadenia a informatiky  
Žilinská univerzita v Žiline  
[jozef.kostolny@fri.uniza.sk](mailto:jozef.kostolny@fri.uniza.sk)

Ing. **Martin Mazúch**  
Fakulta riadenia a informatiky  
Žilinská univerzita v Žiline  
[martin.mazuch@fri.uniza.sk](mailto:martin.mazuch@fri.uniza.sk)