MUNI
FI

Přednáška 4
Algoritmické myšlení

CORE013 Vývoj softwarových systémů: od myšlenky k funkčnímu řešení

4. Algoritmické myšlení

- Algoritmické myšlení
- Programování jako koncept
- Jak vypadá práce programátora
- Programování prakticky

Domácí práce a příprava na tuto přednášku

- Proklikejte si portál iMyšlení
- Proklikejte si portál CS Unplugged

ALGORITMICKÉ MYŠLENÍ

Algoritmické myšlení

- Algoritmické (informatické) myšlení nám umožňuje
  - vzít složitý problém,
  - pochopit, v čem je jeho podstata, a
  - vyvinout možná řešení.
- Tato řešení pak můžeme představit způsobem,
  - kterému porozumí počítač, člověk nebo oba.

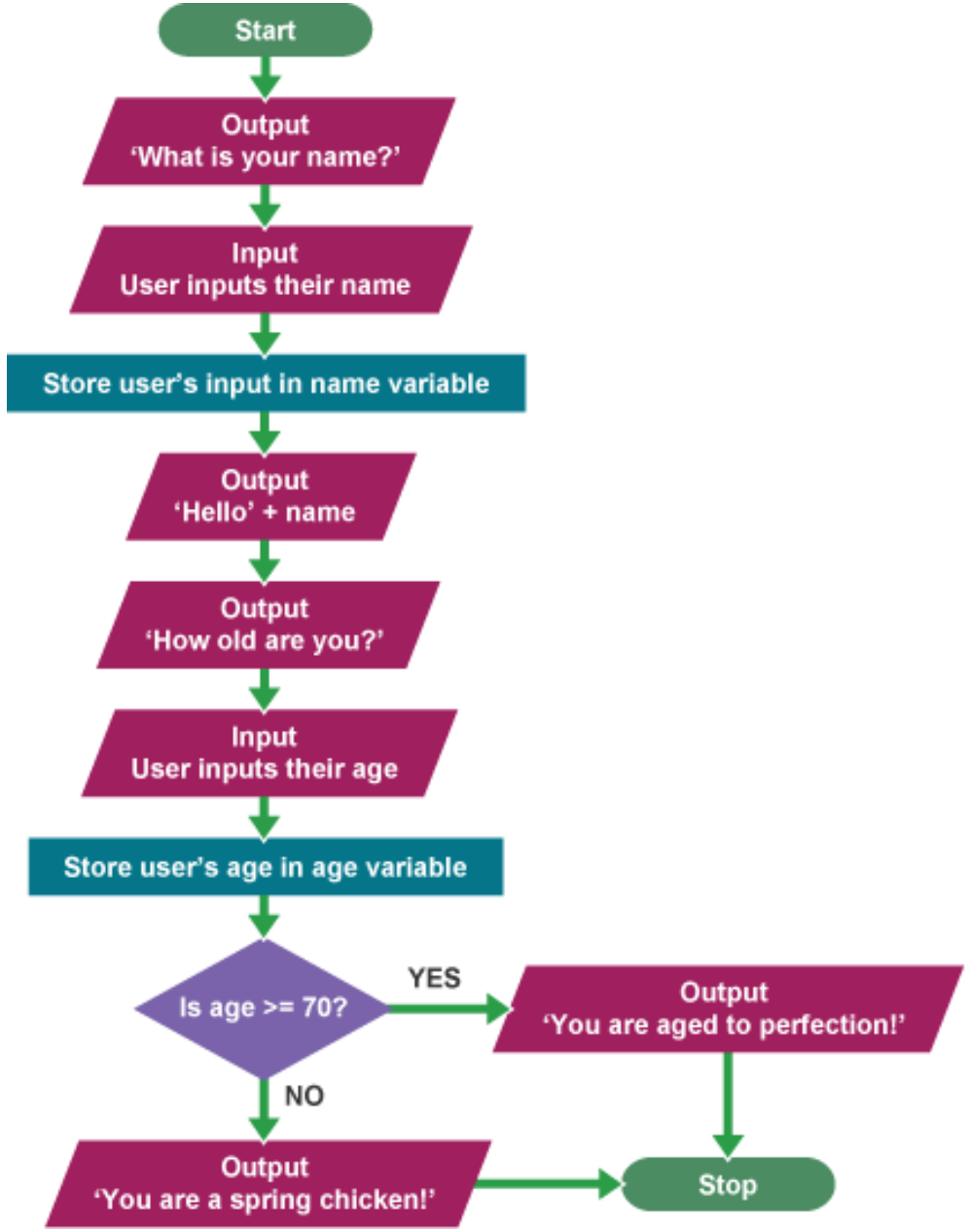
- **Programování** říká počítači, co má dělat a jak to má dělat.
- **Algoritmické myšlení** umožní rozmyslet, co přesně chceme počítači zadat.

Pilíře algoritmického myšlení

- Dekompozice
  - rozdělení složitého problému nebo systému na menší, lépe zvládnutelné části
- Rozpoznávání vzorů
  - hledání podobností mezi problémy a v rámci každého z nich
- Abstrakce
  - zaměřit se pouze na důležité informace, ignorovat nepodstatné detaily
- Algoritmy
  - vývoj podrobného řešení problému nebo pravidla, která je třeba při řešení problému dodržovat

Algoritmy

- Algoritmus je plán, sada podrobných pokynů k vyřešení problému.
- Algoritmus musí být jasný.
  Musí mít počáteční bod, konečný bod a sadu jasných pokynů mezi nimi.
- Počítače jsou jen tak dobré, jak dobré jsou algoritmy na pozadí.



https://www.bbc.co.uk/bitesize/guides/zpp49j6/revision/3

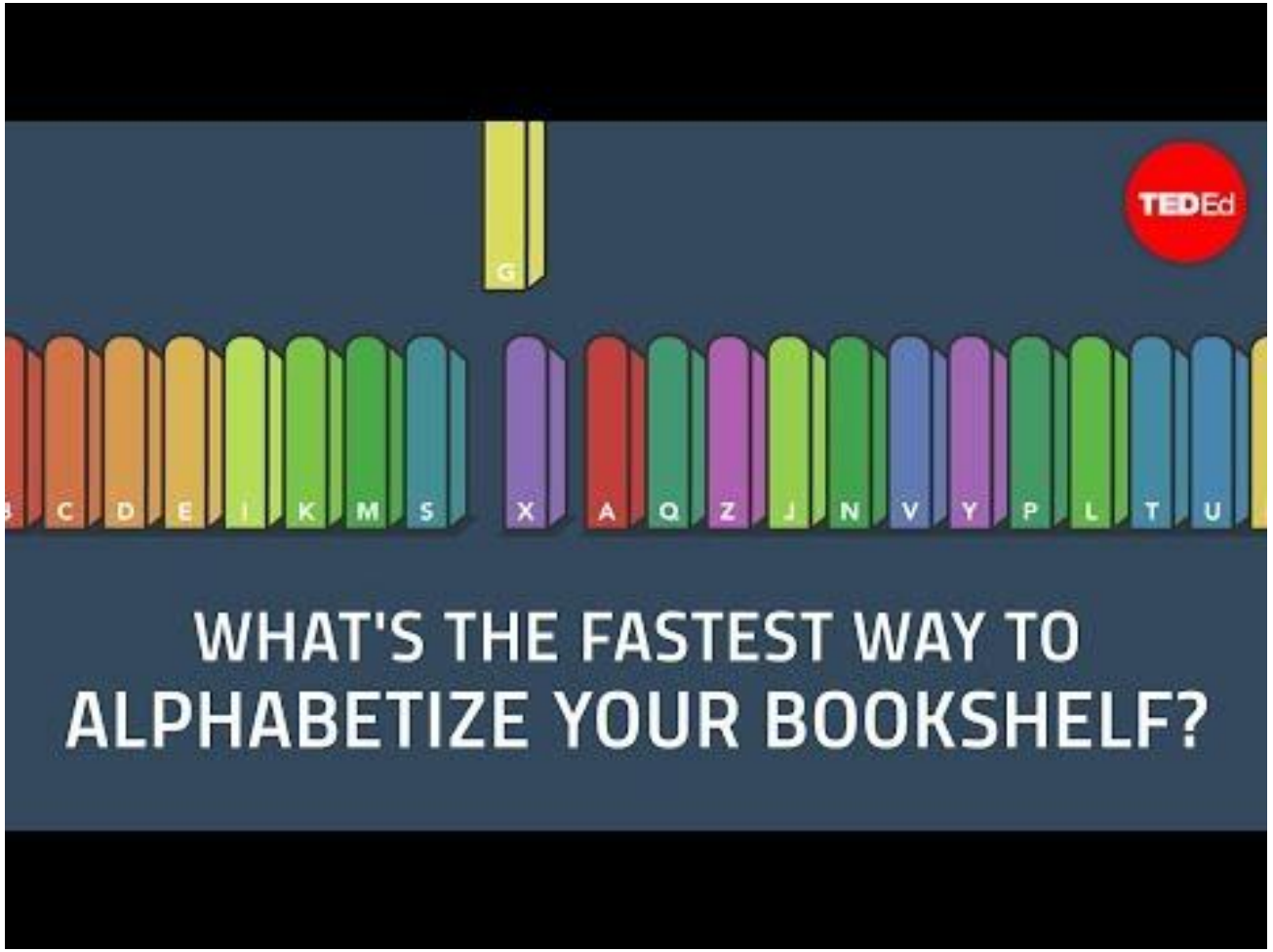
Algoritmy v každodenním životě

- Vaření podle receptu
- Manuál na sestavení nábytku
- Vyhledávání pojmu ve slovníku
- Sčítání pod sebou
- Skládání puzzle
- Zavazování tkaniček

https://www.bbc.co.uk/bitesize/guides/zpp49j6/revision/1

MUNI FI

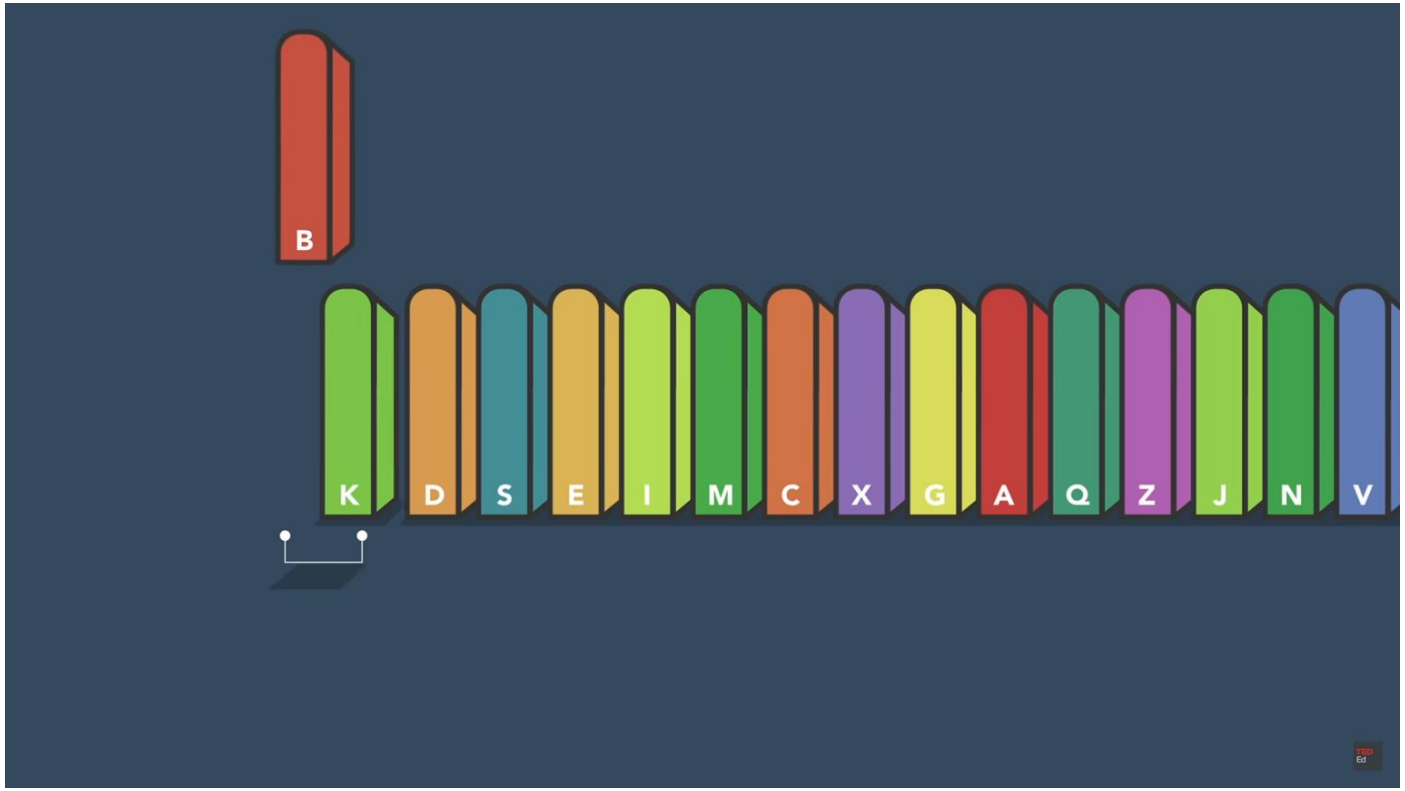
Řadící algoritmy



MUNI FI

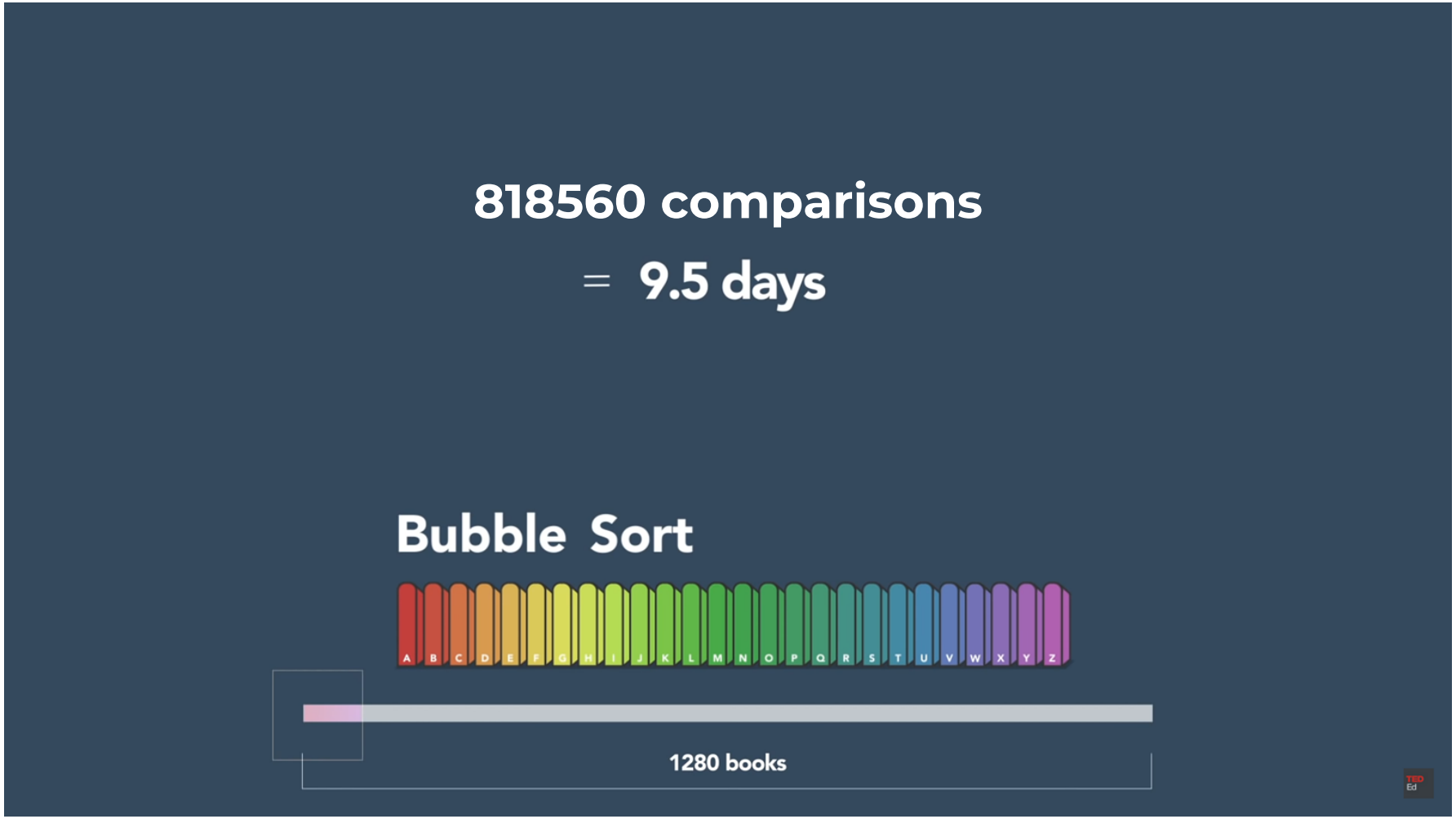
Bubble Sort

- Vezmeme první prvek a vyměňujeme jej za každý následující prvek, dokud jsou ve špatném pořadí. Až “probublá” na svou pozici, opakujeme postup s dalším prvkem.



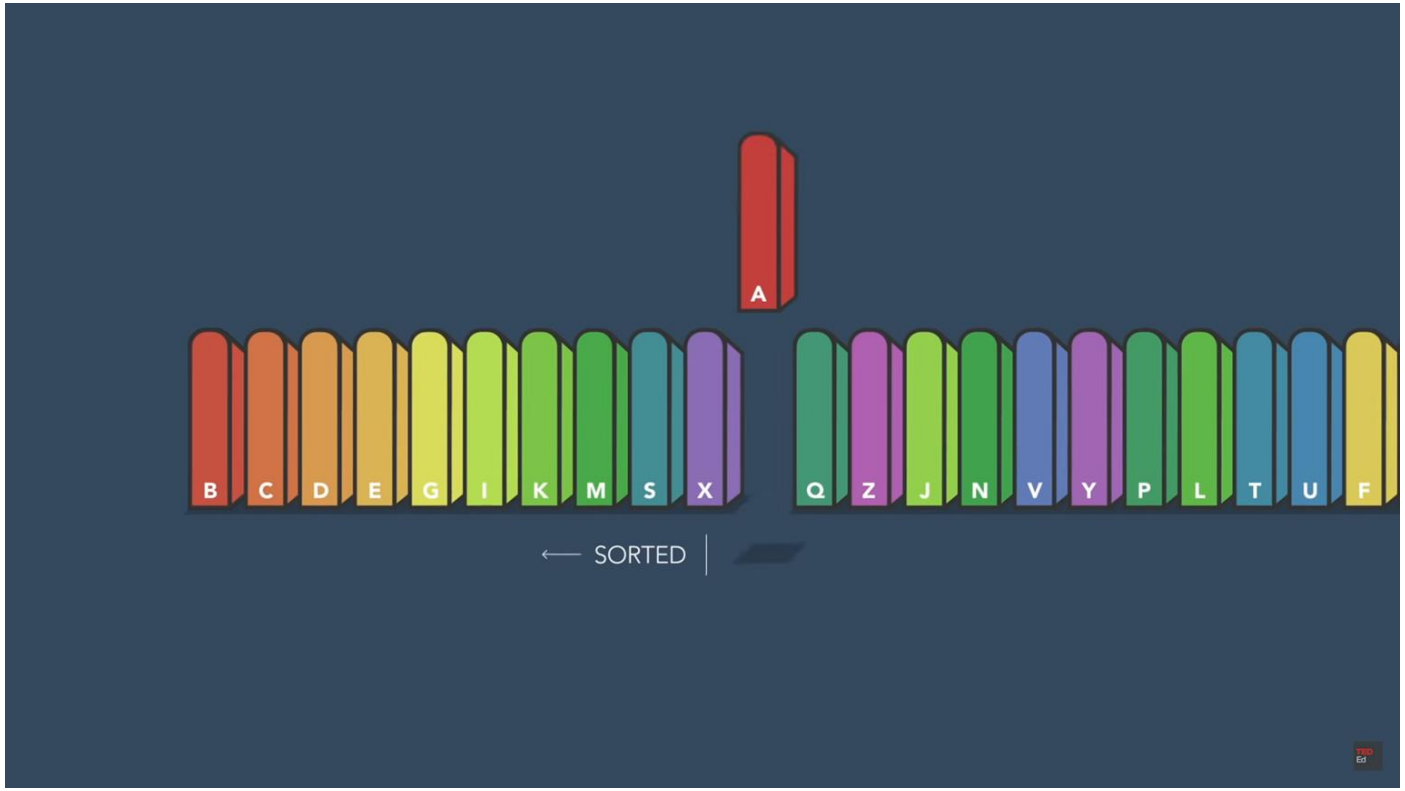
MUNI FI

Bubble Sort

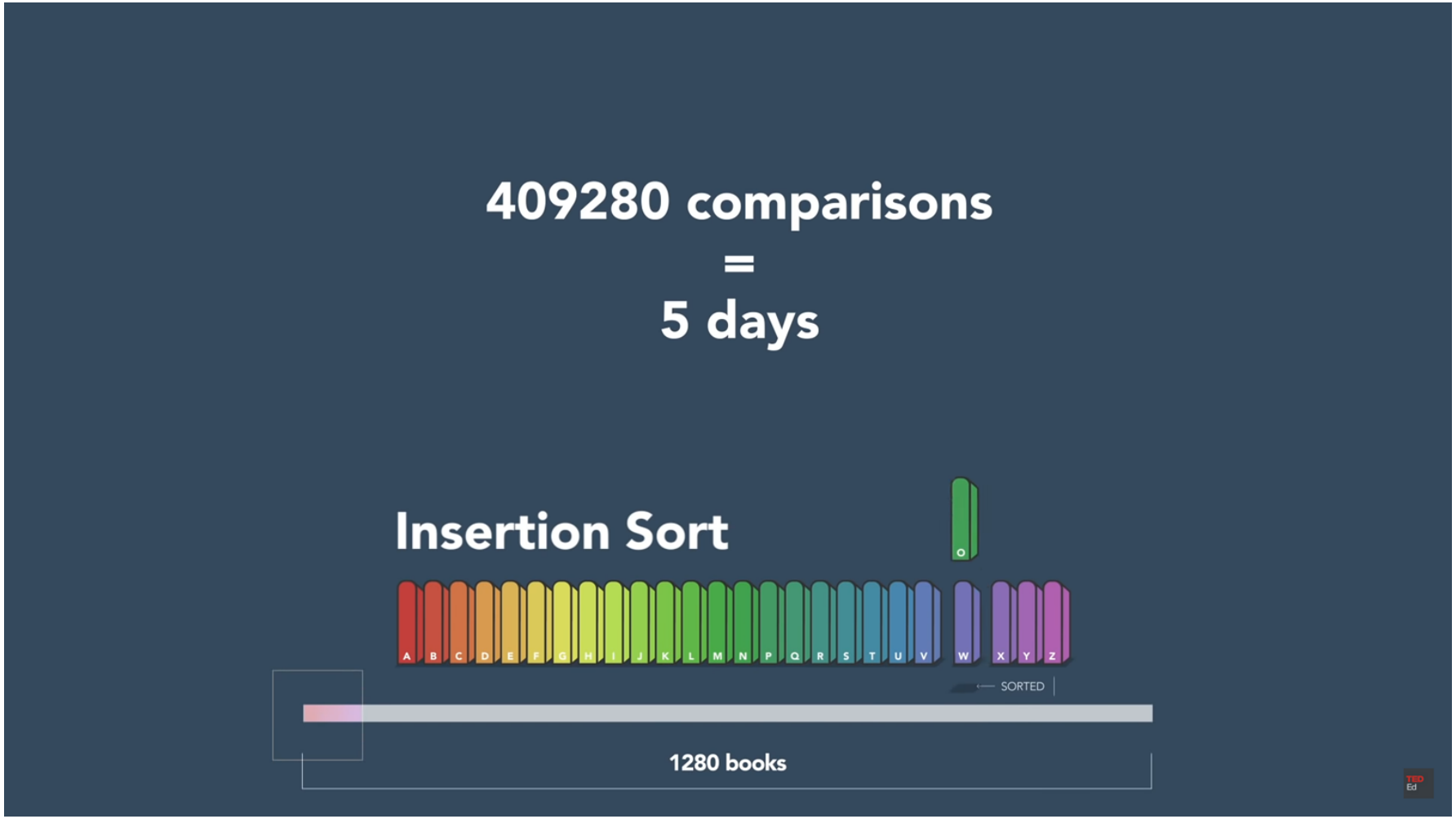

818560 comparisons
= 9.5 days
Bubble Sort
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
1280 books

Insertion Sort

- Seřadíme první dva prvky, poté bereme každý další prvek a vkládáme jej na správnou pozici v seřazené části seznamu.



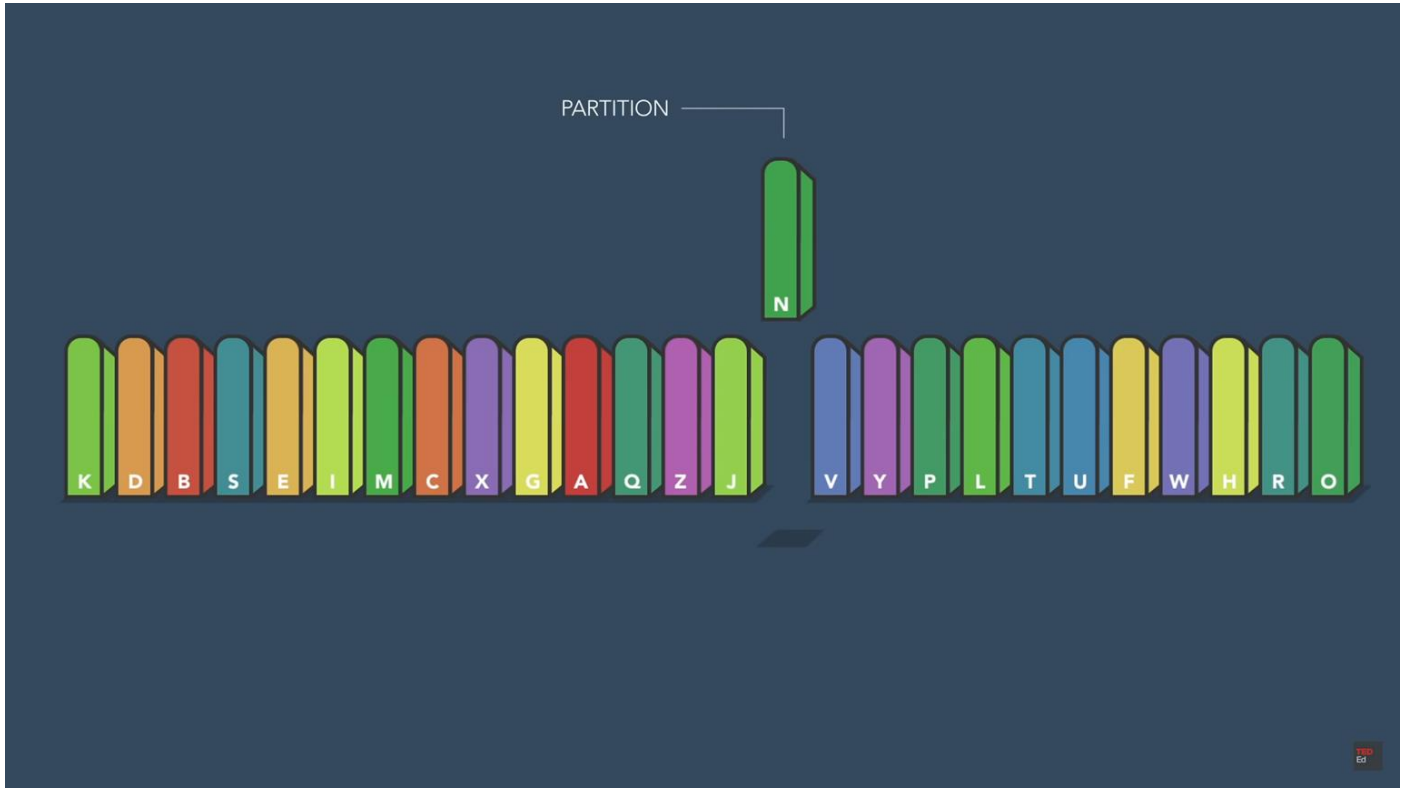
Insertion Sort



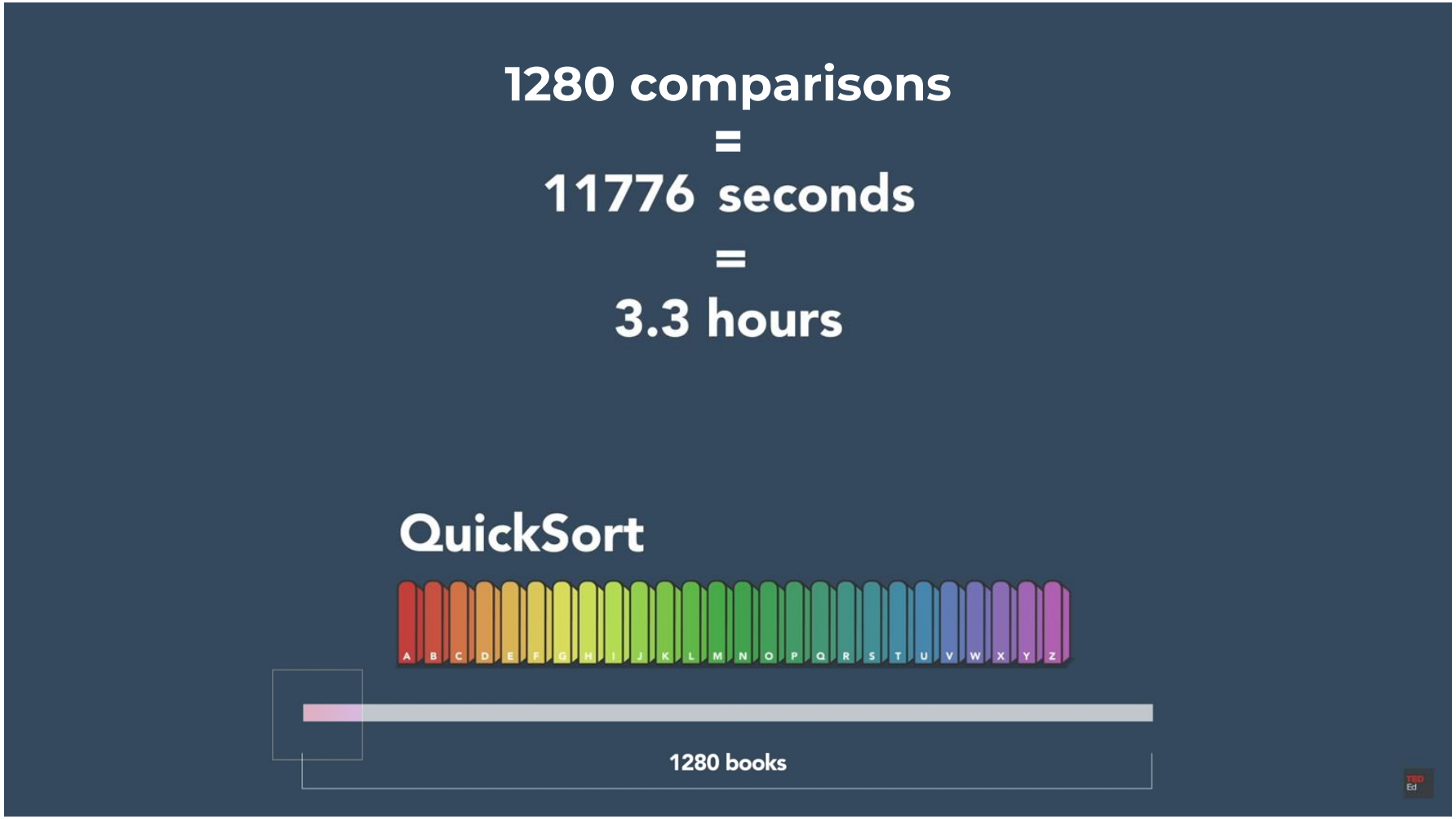
MUNI FI

Quick Sort

- Vybereme náhodný prvek, zbylé rozdělíme, jestli patří nalevo nebo napravo od něj. Toto opakujeme nkrát, dokud nerozdělíme seznam na malé podčásti, které doseřadíme např. insertion sortem.



Quick Sort



MUNI FI

PROGRAMOVÁNÍ

MUNI FI

Základní pojmy

- Algoritmus
  - série instrukcí vedoucí ke splnění určitého úkolu
- Program
  - spustitelný software, běží přímo v OS počítače
- Programovací jazyk
  - sada příkazů, znaků a klíčových slov, vlastní syntax
- Syntax
  - pravidla jak psát a strukturovat kód v konkrétním programovacím jazyce

```
// display a menu of all available models

db.getView('room_loops', 'models',
  function (err, data) {
    if (err) {
      return alert(err);
    }
    var n = data.rows.length;
    for (var i = 0; i < n; ++i) {
      var doc = data.rows[i].key;
      var s = url + '?modelid=' + doc._id;
      $('<li/>').append($('<a>')
        .attr('href',s)
        .text(doc.name))
      .appendTo('#navigatorlist');
    }
    var p = $('<p/>').appendTo('#content');
    p.append( $('<a/>').text('Home').attr('href',url) );
    p.append( document.createTextNode( three_spaces ) );
    p.append( $('<a/>').text('Back').attr('href',url) );

    $('<p/>')
      .text( 'Please select a model in the list below.' )
      .appendTo('#content');

    var prompt = n.toString() + ' model'
      + pluralSuffix( n ) + dotOrColon( n );

    $('<p/>').text( prompt ).appendTo('#content');
  }
);
```


MUNI FI

Základní pojmy

- Zdrojový kód
  - kód napsaný v nějakém programovacím jazyce
- Byte code
  - mezistupeň mezi zdrojovým a strojovým kódem, kód čitelný virtuálním strojem
- Strojový kód
  - série instrukcí určená přímo pro procesor
- Překladač (kompilátor)
  - software, který překládá zdrojový kód do strojového

Zdrojový kód, byte code, strojový kód

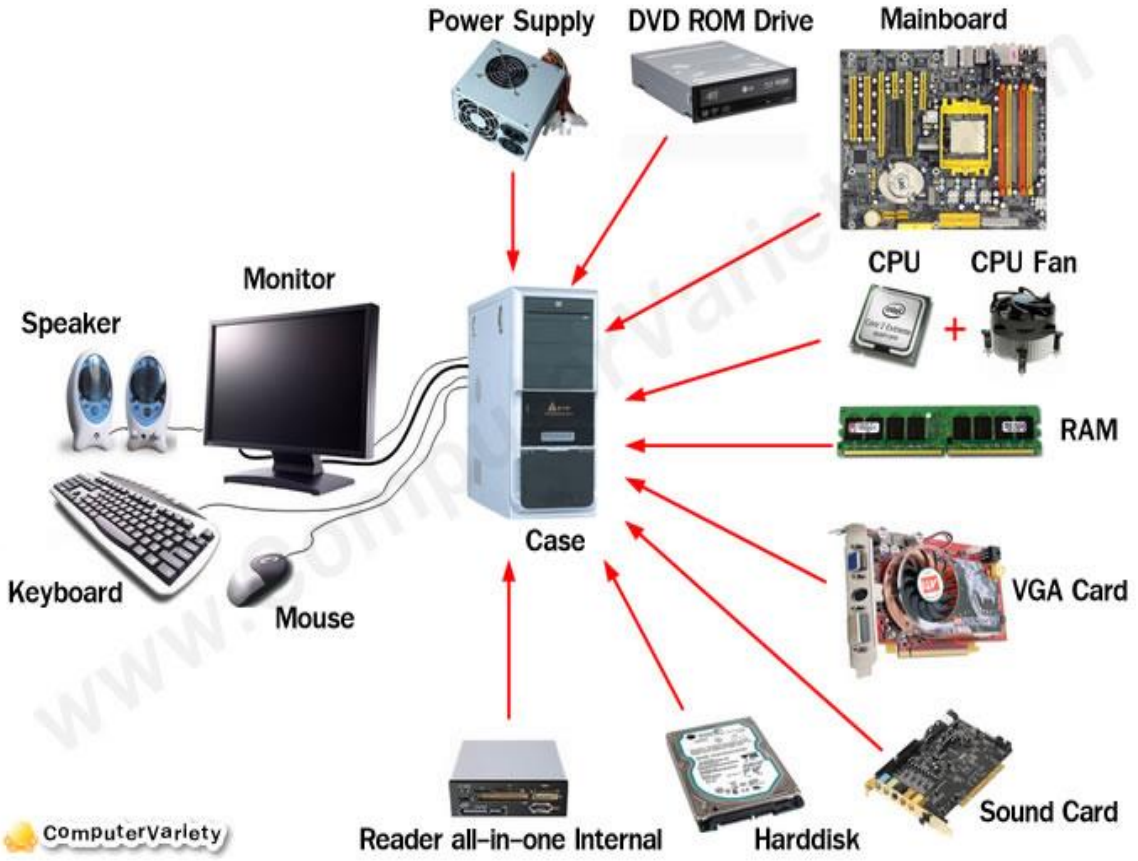
```
1  package store;
2
3
4  /**
5   * Write a description of class Store here.
6   *
7   * @author (your name)
8   * @version (a version number or a date)
9   */
10 public class Store
11 {
12     private Product[] products; // products stored in the store
13
14     /**
15      * Constructor for objects of class Store
16      */
17     public Store()
18     {
19         products = new Product[10];
20
21         products[0] = new Product("pen");
22         products[1] = new Product("book");
23     }
24
25     /**
26      * An example of a method - replace this comment with your own
27      *
28      * @param  y   a sample parameter for a method
29      * @return     the sum of x and y
30      */
31     public Product findProduct(String name)
32     {
33         //for (int i = 0; i < products.length; i++) {
34         //    if ((products[i] != null) && (products[i].getName() == name)) {
35         //        return products[i];
36         //    }
37         //}
38
39         int i = 0;
40         while (i < products.length) {
41             if ((products[i] != null) && (products[i].getName() == name)) {
42                 return products[i];
43             }
44             i++;
45         }
46
47         return new Product("Not found!");
48     }
49 }
50
```

```
1  Êþº¾   2 *
2
3   ¸        »
4     !¸ "
5     #¸ $¸ %¸ &¸ ¸products¸ ¸[Lstore/Product;¸ ¸<init>¸ ¸()V¸
   ¸Code¸ ¸LineNumberTable¸ ¸LocalVariableTable¸ ¸this¸ Lstor
   e/Store;¸ ¸findProduct¸ #(Ljava/lang/String;)Lstore/Product
   ;¸ ¸name¸ ¸Ljava/lang/String;¸ ¸i¸ ¸I¸ StackMapTable¸
6  SourceFile¸
7  Store.java♠  ¸ store/Product♠ ¸ ♠¸ ¸pen♠  '¸ ¸book♠ ( )¸
8  Not
   found!¸ ¸store/Store¸ ¸java/lang/Object¸ ¸(Ljava/lang/Strin
   g;)V¸ ¸getName¸ ¸()Ljava/lang/String; !
9     ¸  ¸ ♠     ¸ ¸ ¸ ¸  f ¸ ¸  ,*. ¸*¸
10 ¾ µ ¸*´ ¸»  Y¸. ¸S*´ ¸»  Y¸. ¸S±     ¸  ¸ ¸  ¸ ¸ ¸
   ¸ ¸ + ¸ ¸  ♠ ¸  , ¸ ¸  ¸ ¸ ¸ ¸ ¸ ¸  ~ ¸ ¸  8¸=*´ ¸¾¢
   &*´ ¸2Æ ¸*´ ¸2¶ ¸+¦
11 *´ ¸2°„ ¸§ÿ×»  Y¸. ¸°  ¸ ¸  ¸ ¸  '  ( ) ! * ( , . /
    ¸
    ¸  8 ¸ ¸  8 ¸ ¸ ¸  6 ¸ ¸  ¸  ¸ ¸ü ¸¾¸ ¸ ¸
```

```
01101010011010100010110110010010101100101010101001010101
01111000101011110001101110111000010101001010100110101010
01010100010010010110101000101001011100011001010100100110
00110101010111101011011110100100100010110101010100000101
00110101001101010001011011001001010110010101010100101010
10111100010101111000110111011100001010100101010011010101
00101010001001001011010100010100101110001100101010010011
00011010101011110101101111010010010001011010101010000010
```

Co musí umět počítač / programovací jazyk?

- Počítat
  - procesor, grafická karta
  - sekvence, podmínky, cykly
- Pamatovat si
  - hard disk, RAM
  - proměnné
- Komunikovat
  - klávesnice, myš, síťová karta
  - input a output
- Znovuvyužívat
  - funkce, objekty, knihovny, moduly



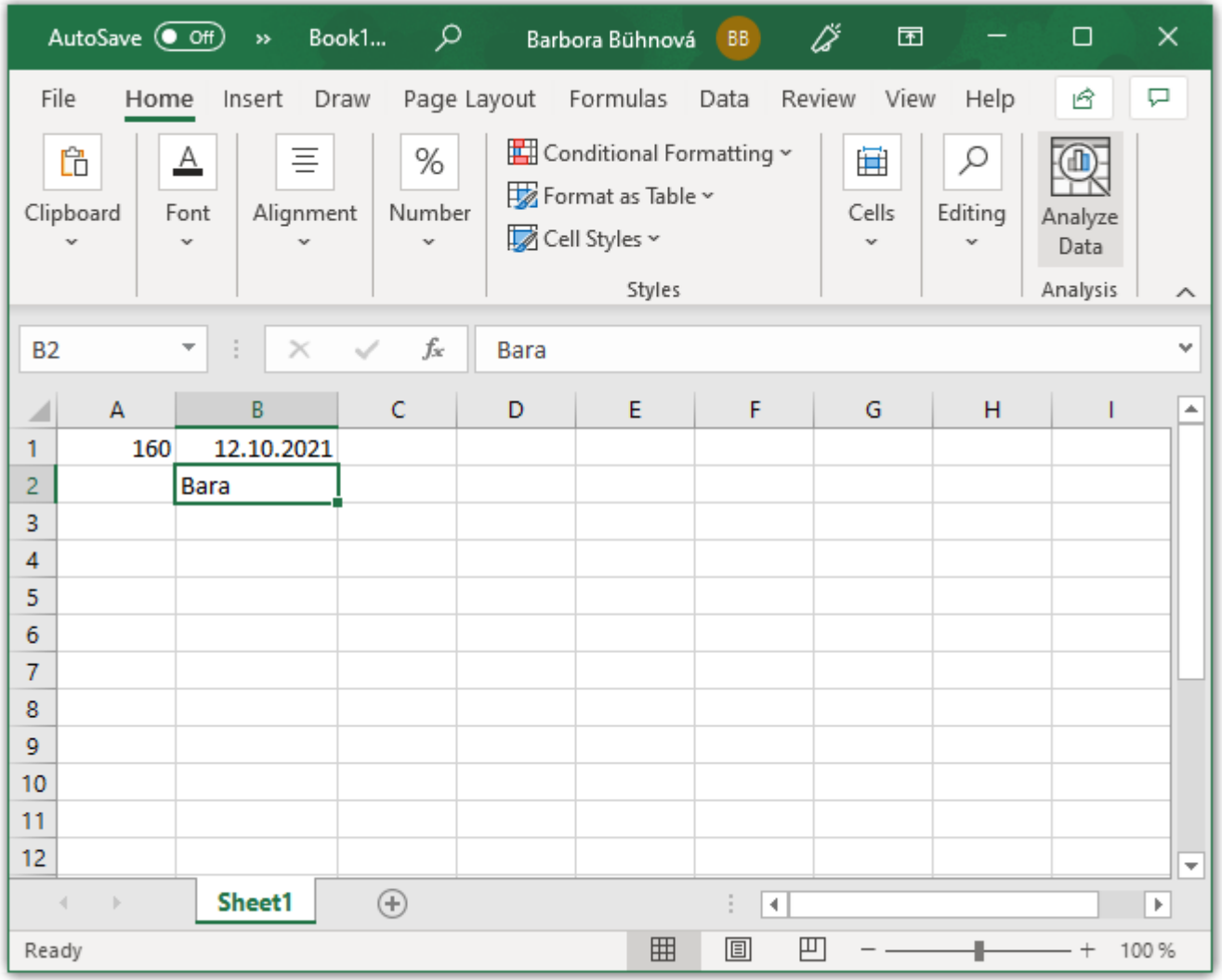
MUNI FI

Základní prvky programovacích jazyků

- Proměnné
  - Místa, kam můžeme ukládat data a výsledky v průběhu běhu programu
- Funkce
  - Pojmenované skupiny příkazů, které můžeme používat opakovaně i v jiných částech kódu
- Datové typy
  - Druhy hodnot používané v daném jazyce, definují jejich význam a operátory a funkce, které na ně můžeme použít. Každý typ má svůj obor hodnot.
- Knihovny, moduly
  - Sada již definovaných funkcí, které můžeme použít ve svém kódu.

Proměnné

- Název proměnné
- Uložená hodnota
- Datový typ

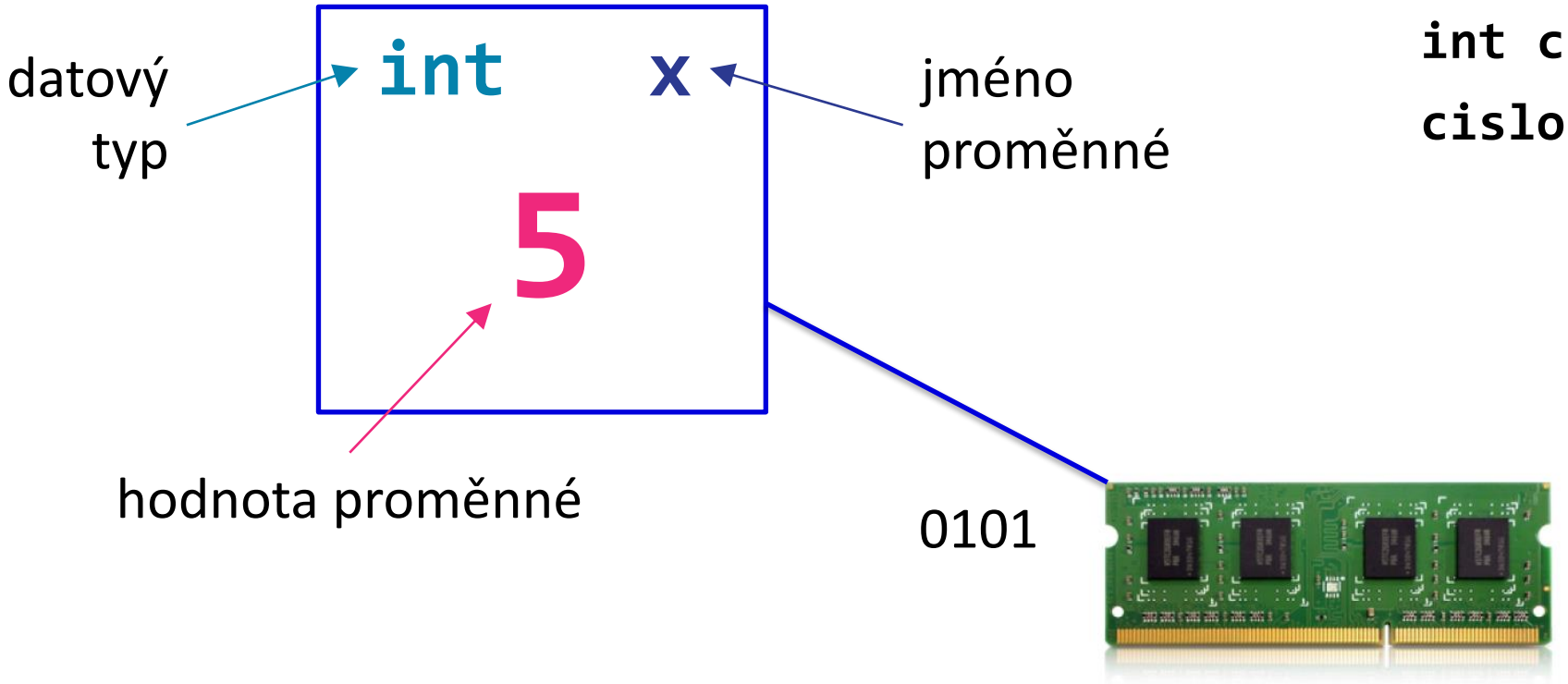


MUNI FI

Proměnné

```
int cislo = 5;
```

```
int cislo;
cislo = 5;
```

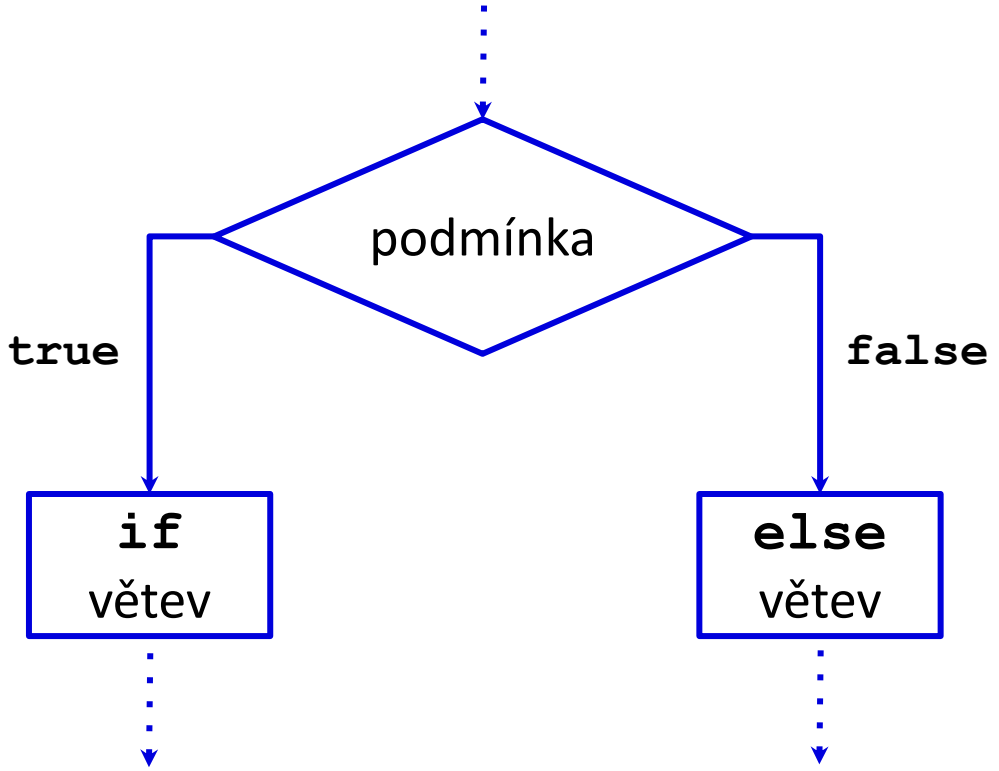


Základní datové typy

- Integer
  - Celé číslo, např. 19
- Float (floating-point number)
  - Číslo s plovoucí řádovou čárkou - znázorňuje reálná čísla, např. 3.14
- String
  - Řetězec - znázorňuje posloupnost znaků (slovo, text, …), např. `"Hello, World!"`
- Boolean
  - Pravdivostní hodnota, může nabývat pouze hodnot `true` a `false`
- Array
  - Pole - struktura, která znázorňuje posloupnost hodnot, např. `[1, 4, 19, 3, 800]`

Podmínky

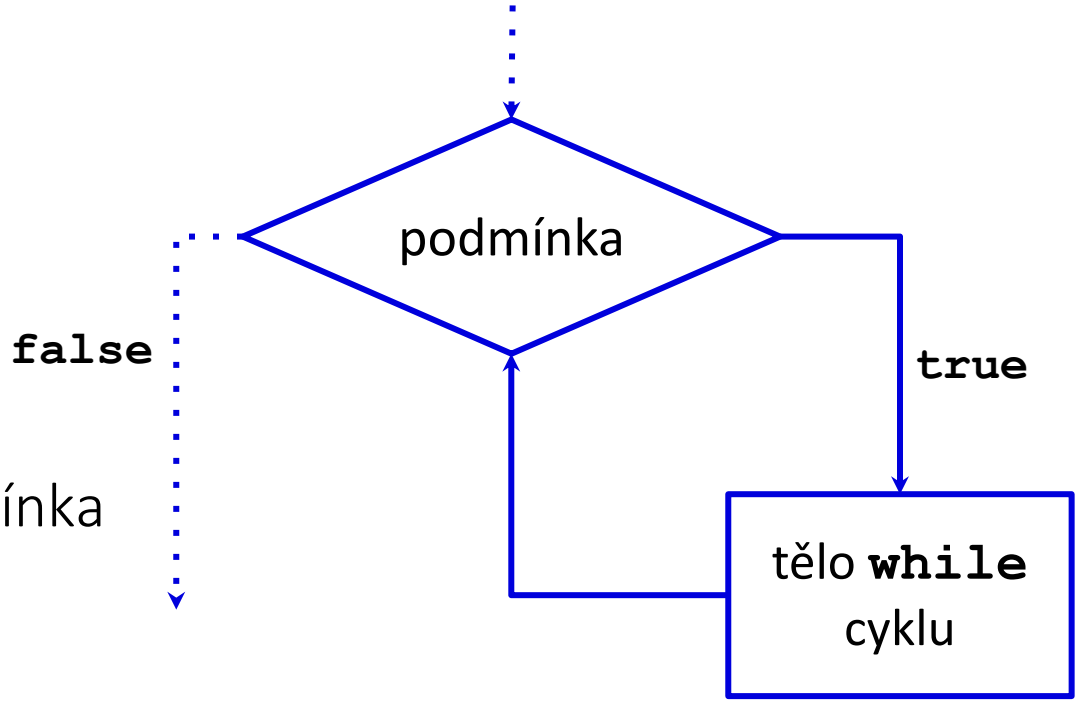
- Příkaz if umožňuje větvení programu.
- V případě, že je splněna podmínka, spustí se blok kódu v if struktuře. Pokud splněna není, tento blok se přeskočí.
- ```
  if (is_raining):
      print("I'll stay home.")
  else:
      print("Let's go out!")
  ```



MUNI FI
  ```

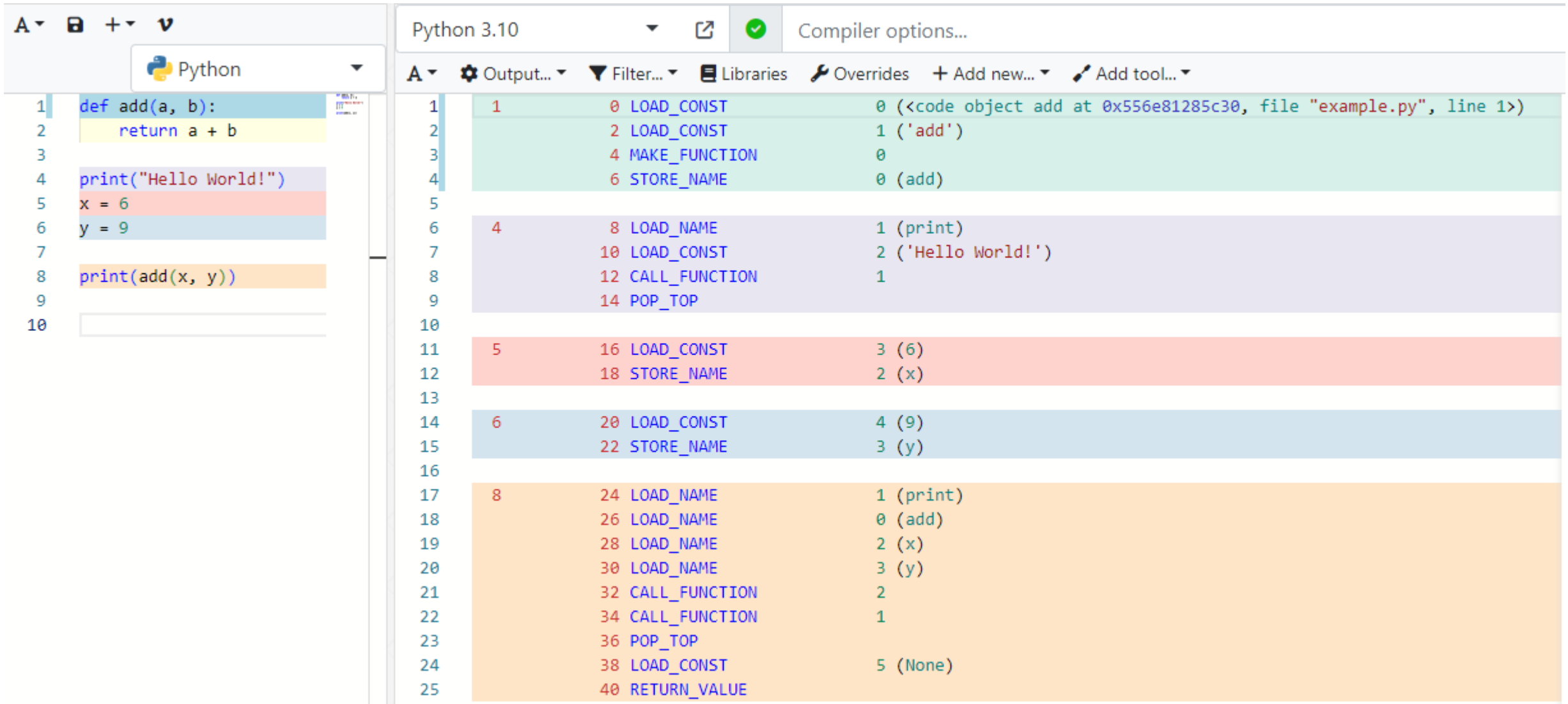
Cykly

- umožňují nám opakovat blok kódu
  - **`while`**
    - opakuje se dokud je splněna vstupní podmínka
  - **`for`**
    - opakuje se pro každou položku v sekvenci



MUNI FI

Compiler Explorer - odkaz na ukázku

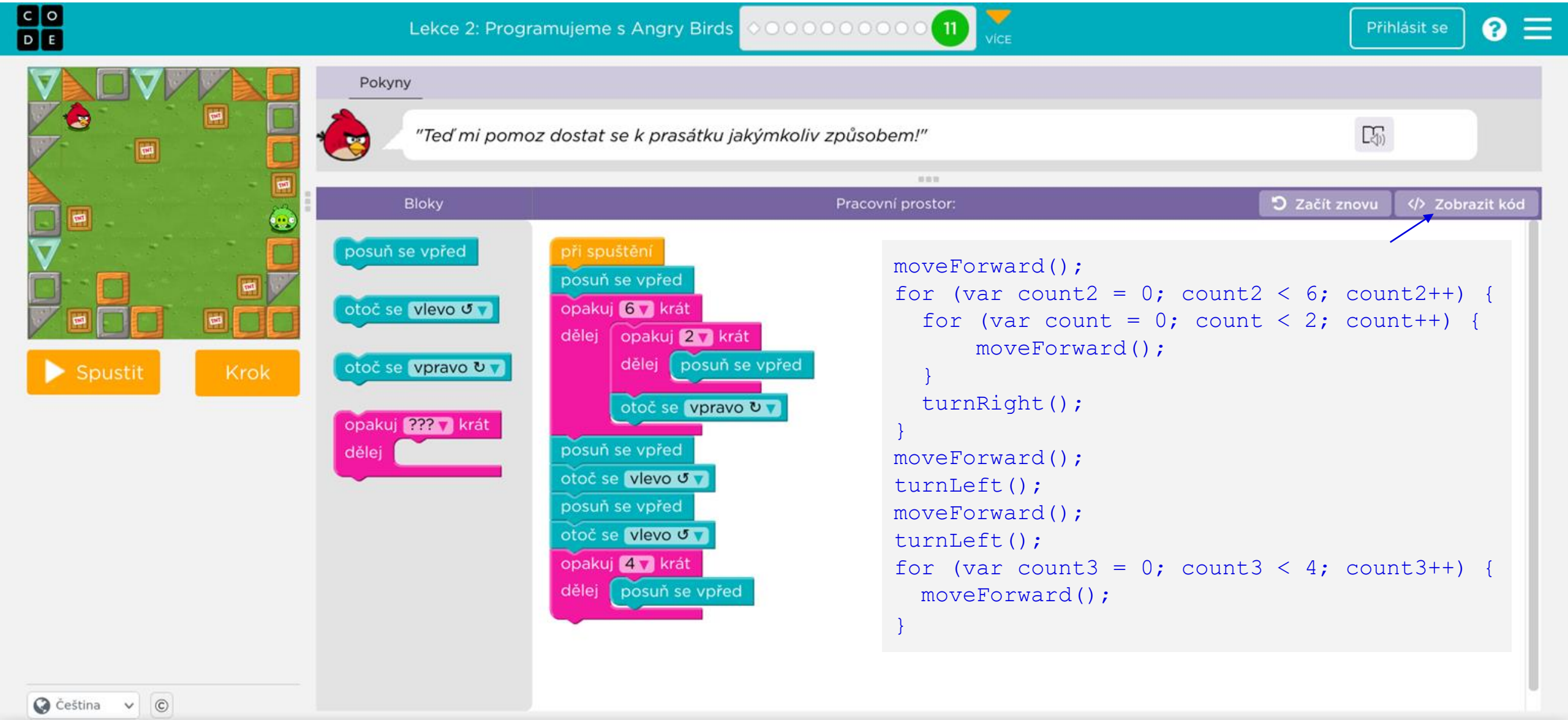


PŘÍSTUPY K PROGRAMOVÁNÍ

Procedurální

- **Lineární** přístup k programování, postavený na **funkcích** a **podprocesech**
- Má globálně oddělenou datovou část od funkční, která s těmito daty pracuje
- Např. C, Rust, Python, MATLAB, Go, ...

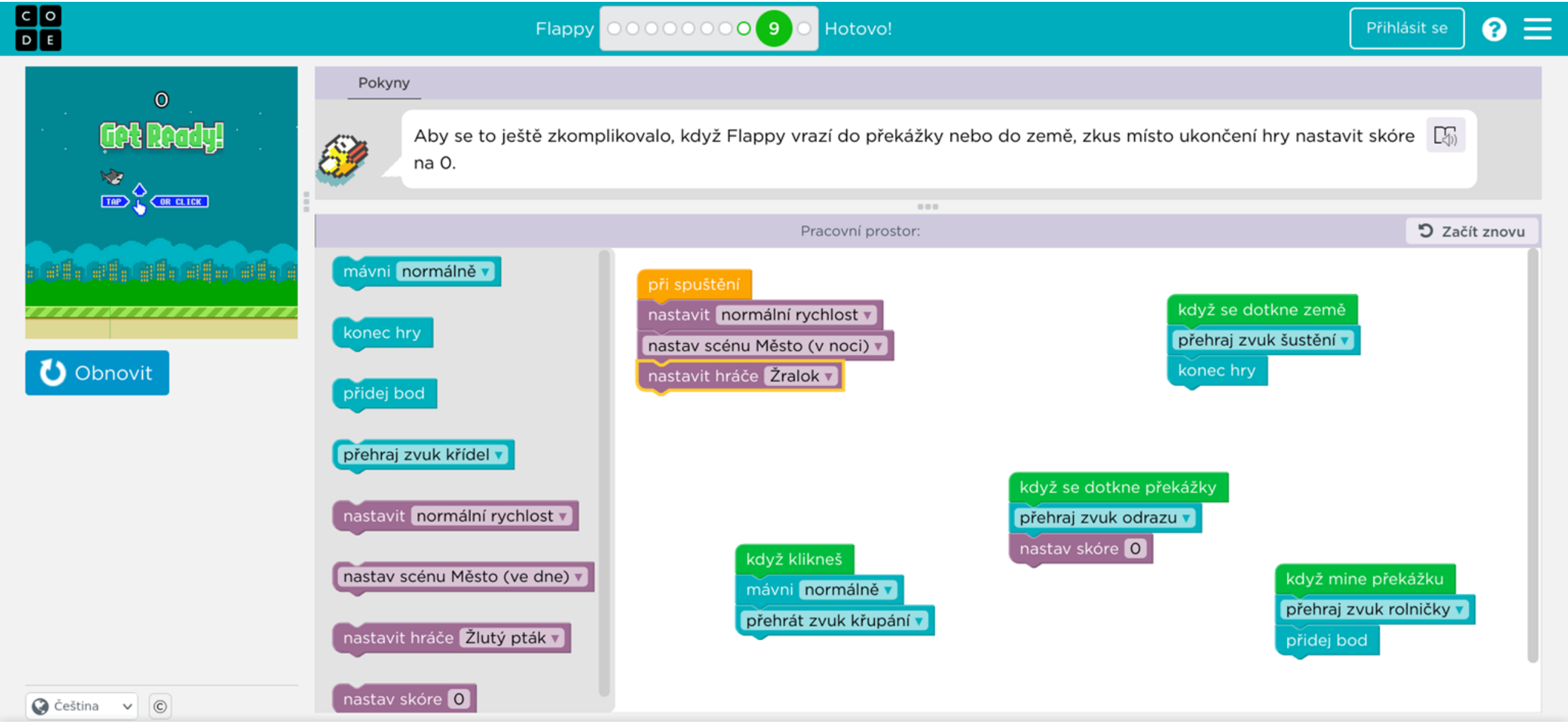
Procedurální - Angry Birds



Event-based

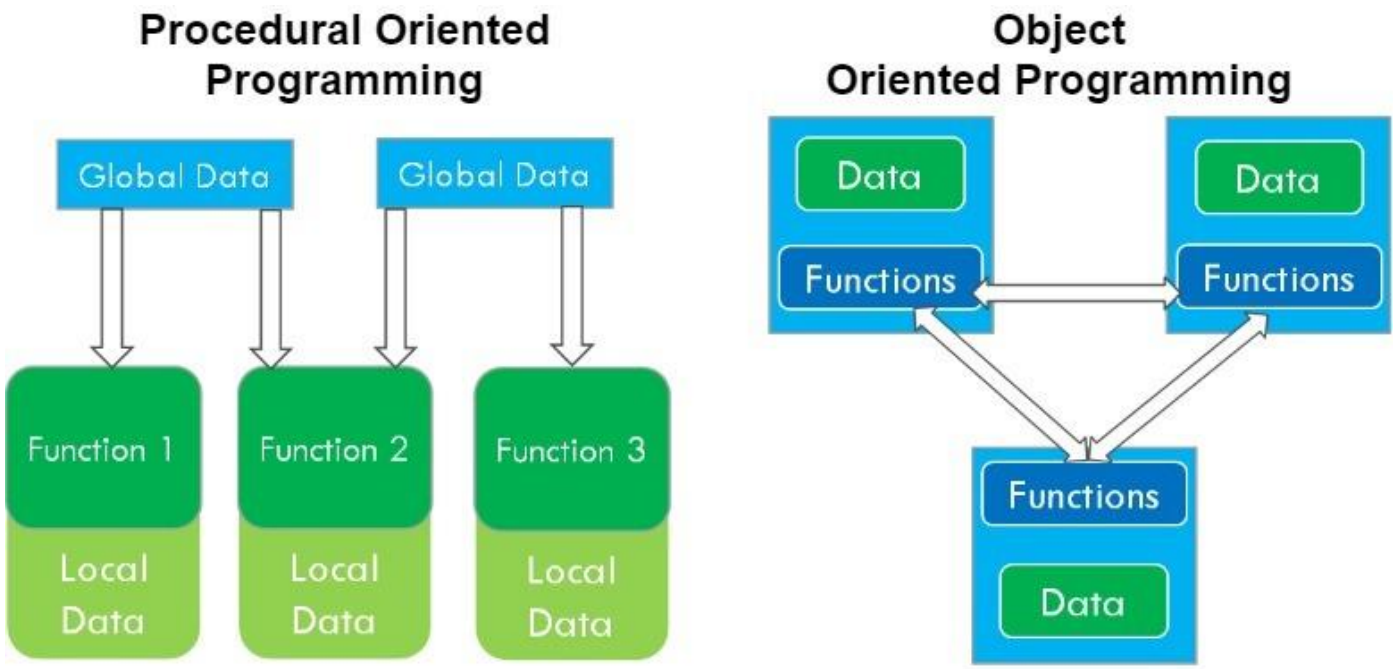
- Na rozdíl od procedurálních neprobíhá lineárně, ale jako **reakce na různé události**
- Těmito událostmi mohou například být:
  - akce uživatele (klikání myší, stisknutí klávesy)
  - signály ze senzorů
  - zprávy z jiných programů
- Např. Visual Basic, Java, C#, ...

Event-based - Flappy



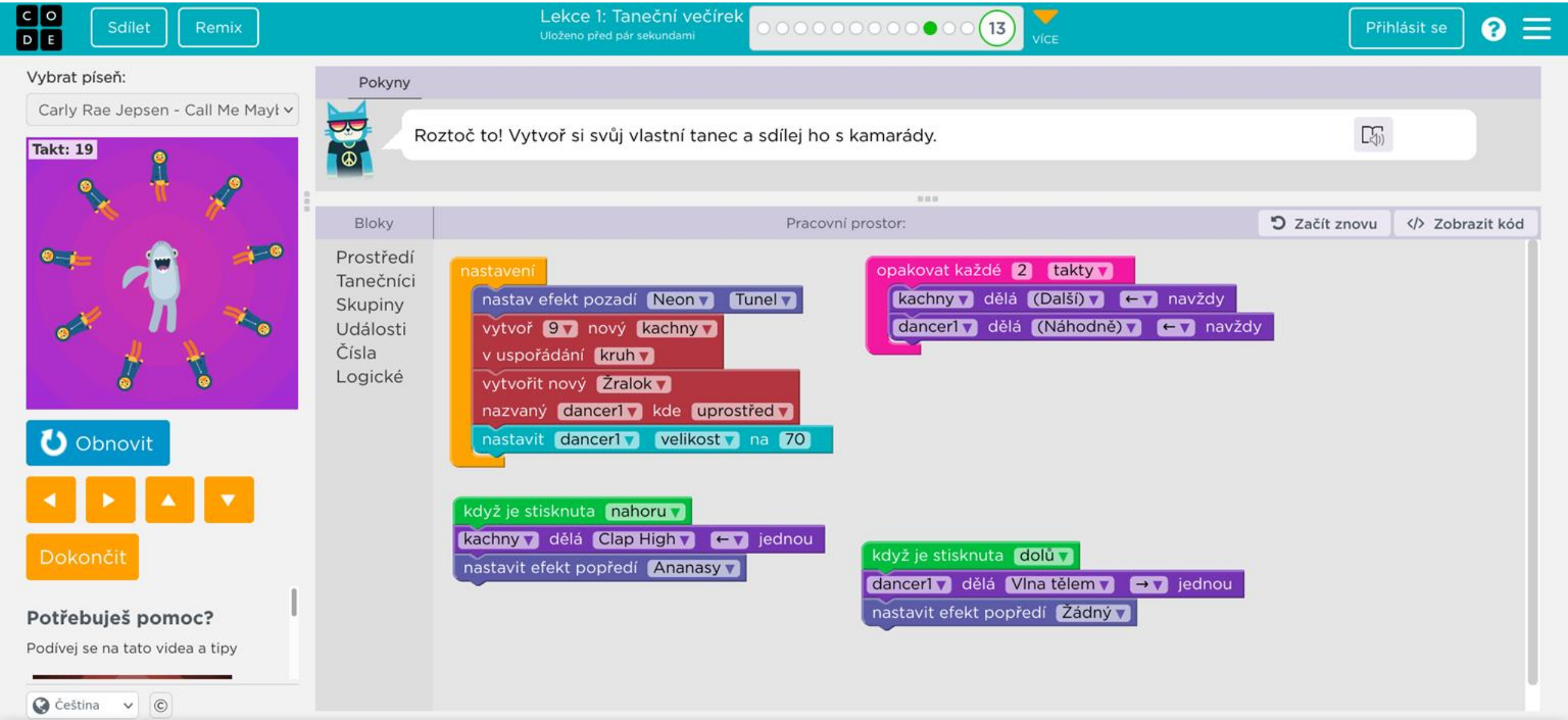
Object-oriented

- Program je dělen do **objektů**, které obsahují **datovou část a metody**, které s těmito daty mohou pracovat.
- Např. C#, Java, C++, Ruby, Smalltalk, ...



https://devopedia.org/object-oriented-programming-concepts

Object-oriented - Dance Party



BEST PRACTICES V PROGRAMOVÁNÍ

Best practices v programování

- **Čistý kód,** pojmenování, konvence zarovnání, SOLID principy
- **Code ownership,** zodpovědnost za kód, který napíši, i do budoucna
- **Průběžné testování**
- **Týmová spolupráce** při psaní kódu, oprava kódu po ostatních
- **Inkrementální přístup** k programování, ať už píšu novou věc, rozvíjím nebo dokonce předělávám starší věc, vždy je třeba postupovat po malých inkrementálních krocích.
- **Průběžná integrace kódu** do společné code base
- **Dokumentace** a technical writing
- **Refactoring** a **návrhové vzory**

Refactoring Guru

Refactoring

Refactoring is a systematic process of improving code without creating new functionality that can transform a mess into clean code and simple design.

Start from the very beginning

Dirty Code

Dirty code is result of inexperience multiplied by tight deadlines, mismanagement, and nasty shortcuts taken during the development process.

Learn more

Clean Code

Clean code is code that is easy to read, understand and maintain. Clean code makes software development predictable and increases the quality of a resulting product.

Learn more

Refactoring Process

Performing refactoring step-by-step and running tests after each change are key elements of refactoring that make it predictable and safe.

Learn more

Code Smells

Code smells are indicators of problems that can be addressed during refactoring. Code smells are easy to spot and fix, but they may be just symptoms of a deeper problem with code.

Learn more

Premium COURSE

21 code smells, 66 refactorings

Interactive examples in Java/C#/PHP

No time limits. Study at your own pace

Learn more about the Course

Refactoring Techniques

Refactoring techniques describe actual refactoring steps. Most refactoring techniques have their pros and cons. Therefore, each refactoring should be properly motivated and applied with caution.

Learn more

DESIGN PATTERNS

Design patterns are typical solutions to common problems in software design. Each pattern is like a blueprint that you can customize to solve a particular design problem in your code.

What's a design pattern?

Benefits of patterns

Patterns are a toolkit of solutions to common problems in software design. They define a common language that helps your team communicate more efficiently.

More about the benefits »

Classification

Design patterns differ by their complexity, level of detail and scale of applicability. In addition, they can be categorized by their intent and divided into three groups.

More about the categories »

Catalog of patterns

List of 22 classic design patterns, grouped by their intent.

Look inside the catalog »

History of patterns

Who invented patterns and when? Can you use patterns outside software development? How do you do that?

More about the history »

Criticism of patterns

Are patterns as good as advertised? Is it always possible to use them? Can patterns sometimes be harmful?

More about the criticism »

Dive Into DESIGN PATTERNS

Check out our ebook on design patterns and principles. It's available in PDF/ePUB/MOBI formats and includes the archive with code examples in Java, C#, C++, PHP, Python, Ruby, Go, Swift, & TypeScript.

Learn more about the book

PROGRAMOVACÍ JAZYKY

Diskuze

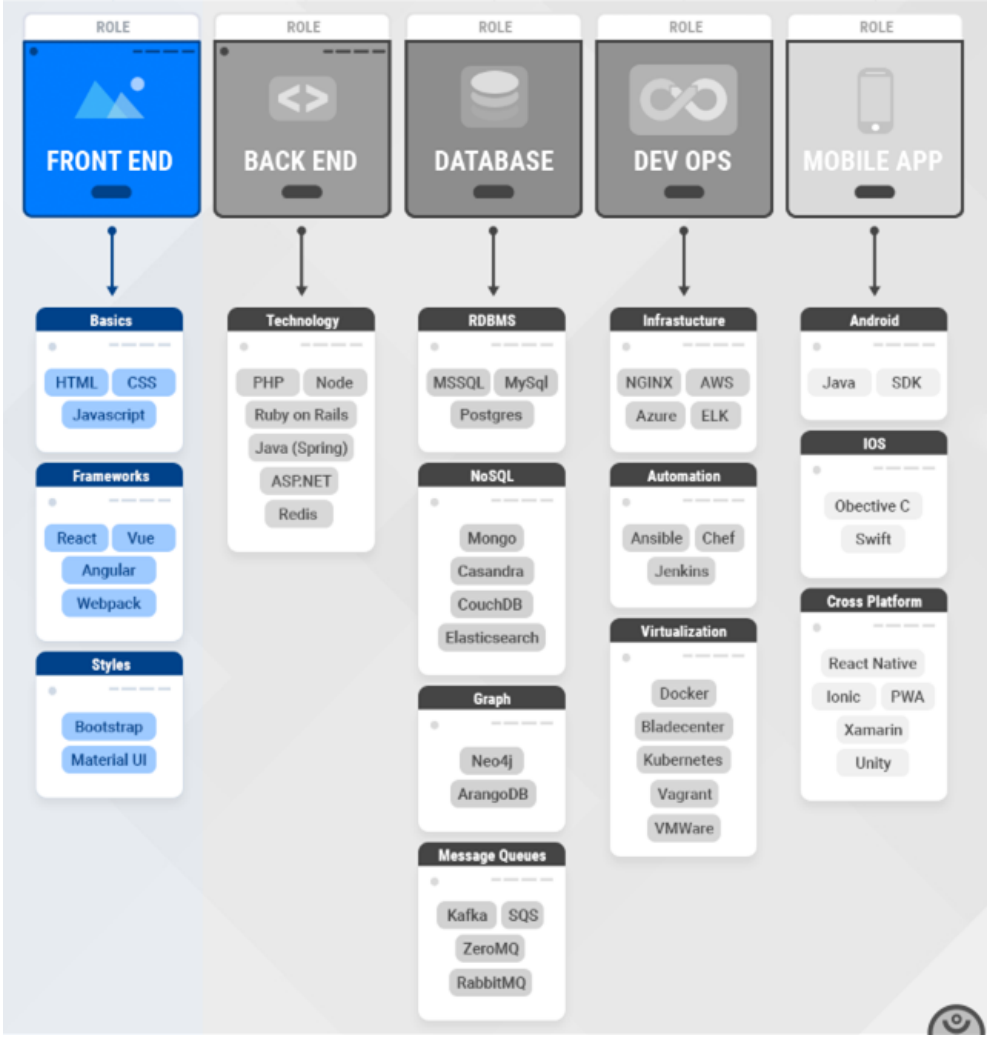
- Jaké programovací jazyky znáte? Na co se používají?
- Kolik programovacích jazyků je potřeba znát?
- Kolik programátorů jsou samouci?
- Jak dlouho je potřeba se programování učit než mohu začít programovat reálnou aplikaci?

Jazyky a frameworky

https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_eas
t,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

MUNI FI

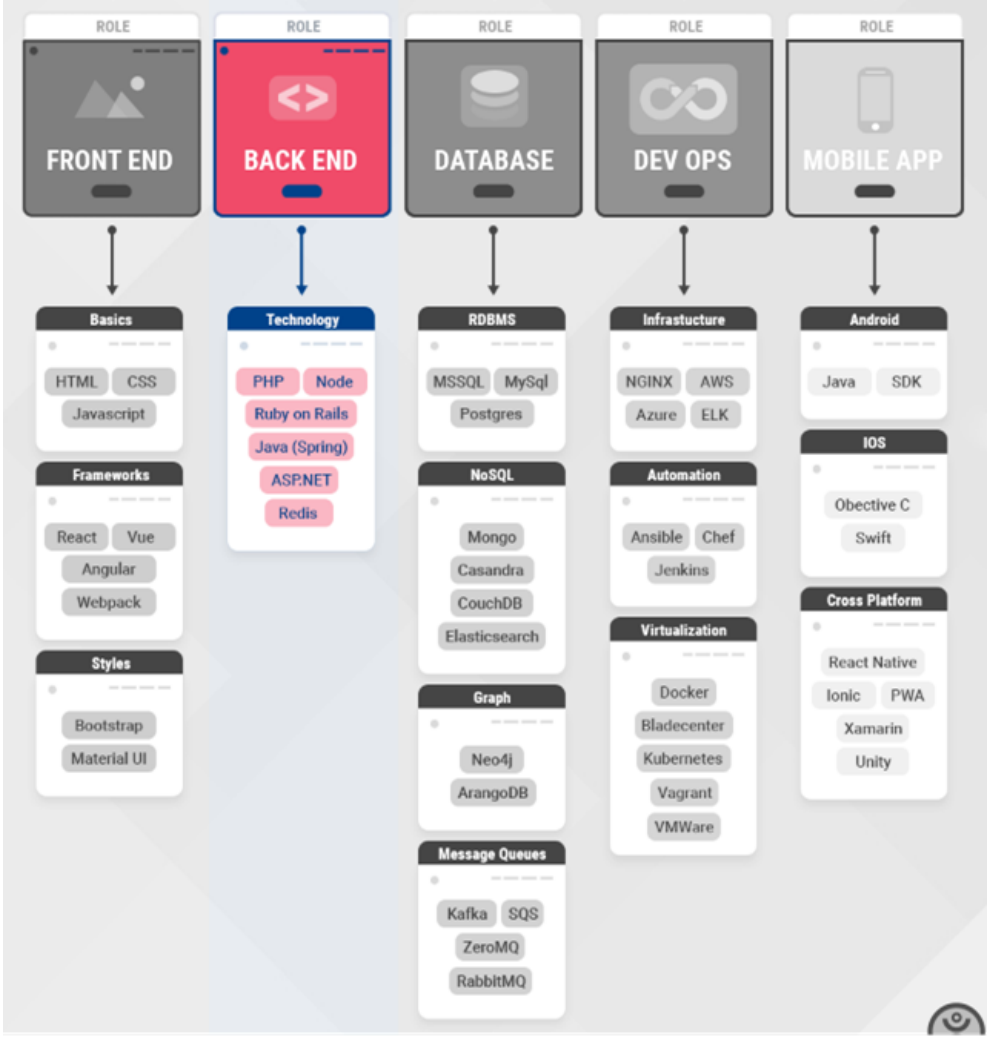
Frontend



https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_east,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

MUNI FI

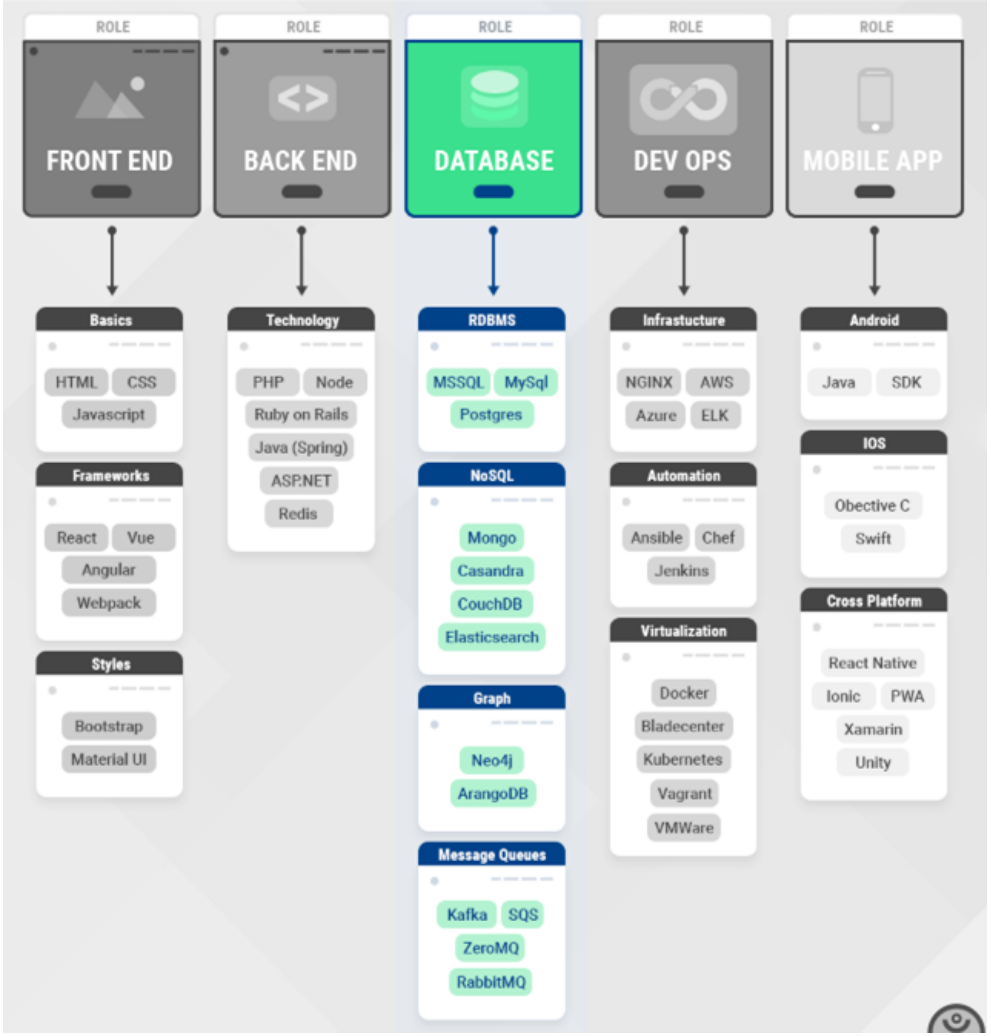
Backend



https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_east,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

MUNI FI

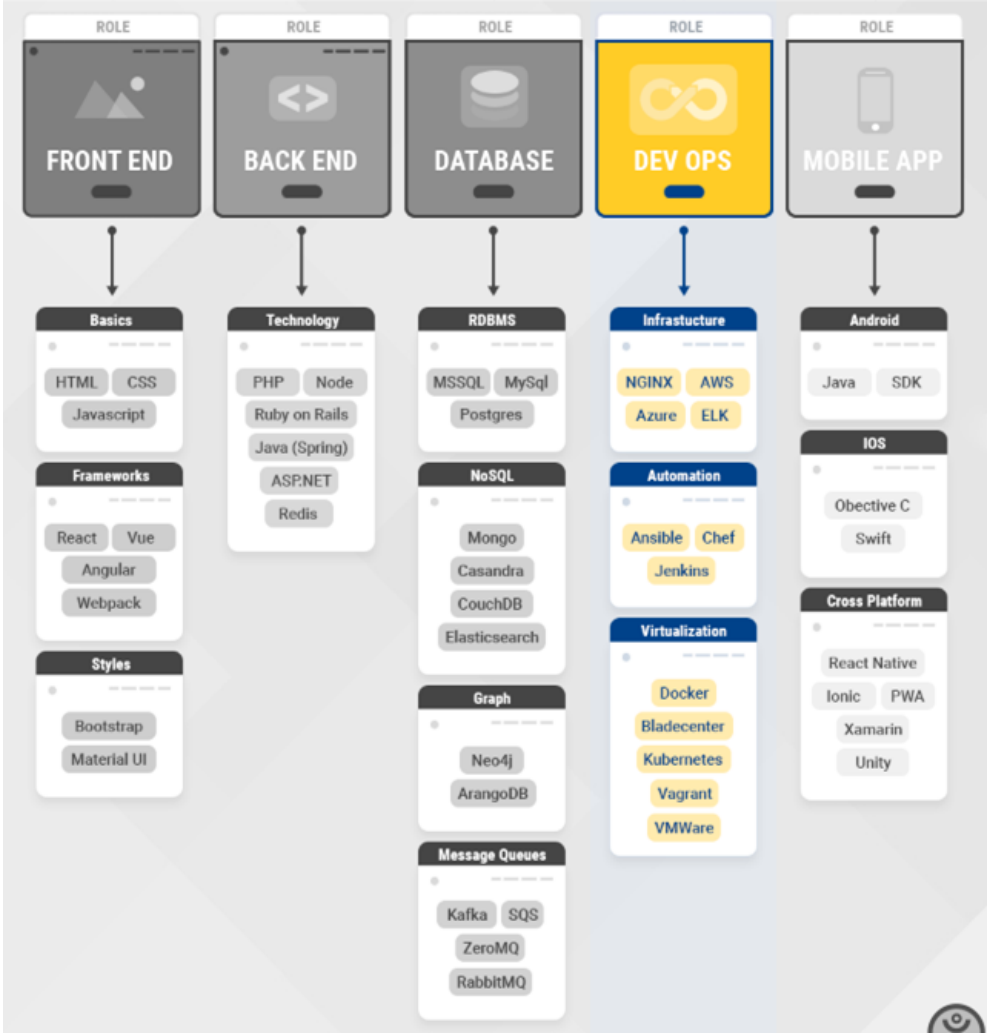
Databáze



https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_eas t,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

MUNI FI

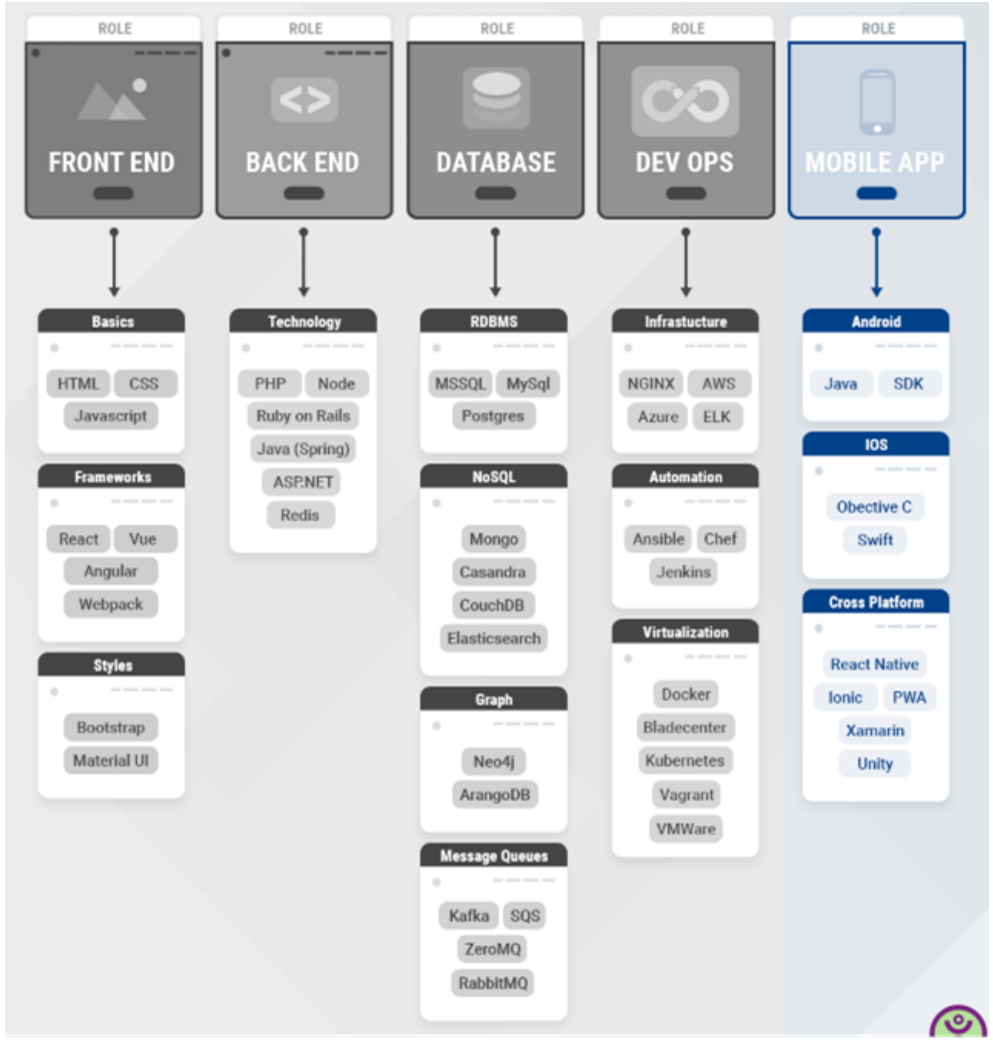
DevOps



https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_east,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

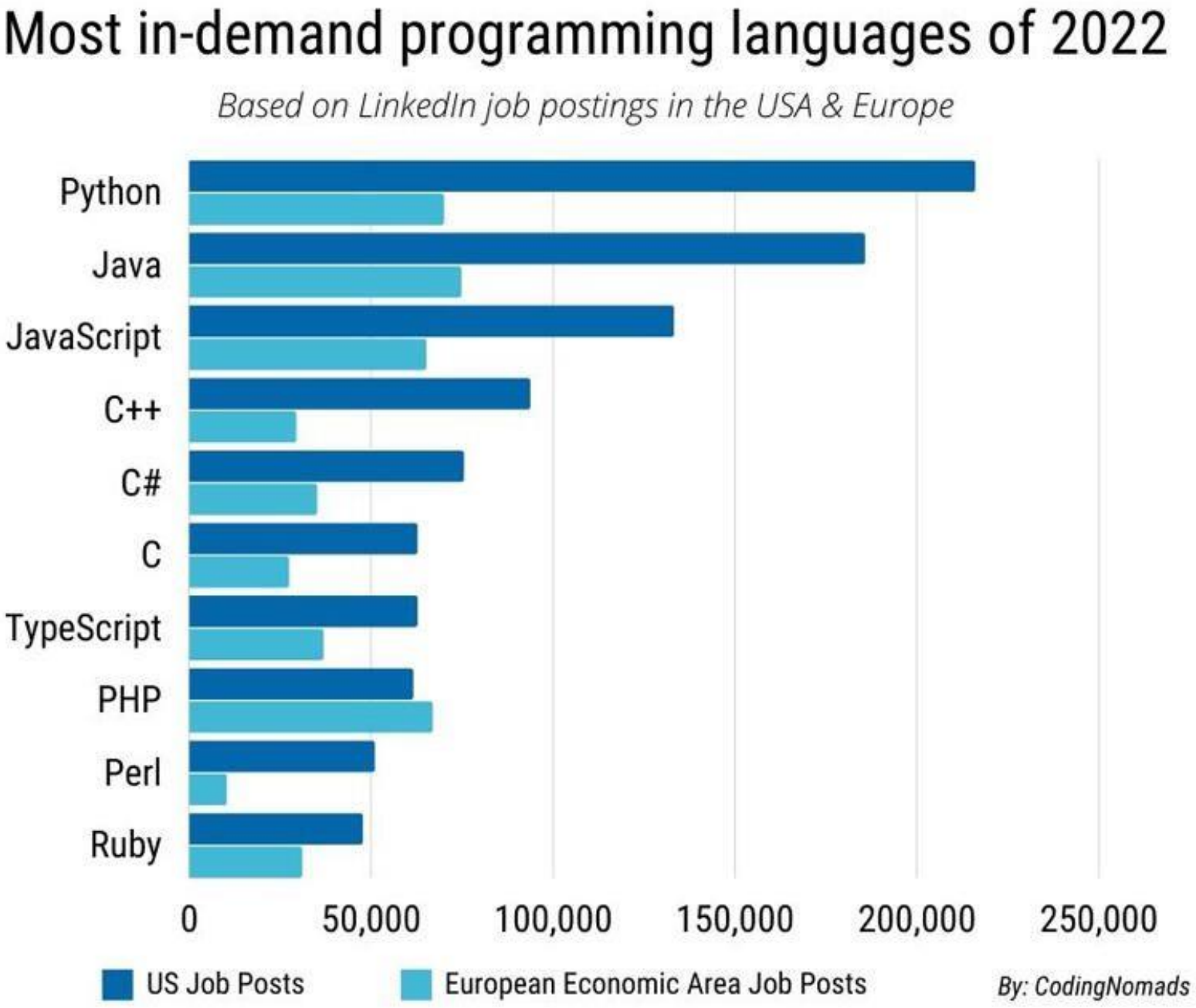
MUNI FI

Mobilní aplikace



https://res.cloudinary.com/cybercoders/image/upload/c_scale,g_south_east,l_cc_logo_bug_wenazs.png,w_40/v1557870077/Full_Stack_ebvf4s.png

MUNI FI


Most in-demand programming languages of 2022
Based on LinkedIn job postings in the USA & Europe
Python
Java
JavaScript
C++
C#
C
TypeScript
PHP
Perl
Ruby
0
50,000
100,000
150,000
200,000
250,000
US Job Posts
European Economic Area Job Posts
By: CodingNomads

WHICH PROGRAMMING LANGUAGE

SHOULD I LEARN FIRST?

WHAT IS PROGRAMMING?

Writing very specific instructions to a very dumb, yet obedient machine.

LANGUAGES

PYTHON · JAVA · C · PHP · C++

JAVASCRIPT · C# · RUBY · OBJECTIVE-C

START

WHY DO YOU WANT TO LEARN PROGRAMMING?

- For my kids → Start with Scratch, then move on to...
- I don't know, just pick one for me
- Just for fun
- I'm interested → Have a brilliant idea/platform in mind?
  - Nope. Just want to get started → I prefer to learn things...
    - The easy way
    - The best way
    - The slightly harder way → Auto or Manual car?
      - Auto
      - Manual
    - The really hard way (but easier to pick up other languages in the future)
  - YES → Which platform/field?
- Improve myself → Have a brilliant idea/platform in mind?
- Make money → Get a job → Which platform/field?
- I have a startup idea! → Which platform/field?

Which platform/field?

- I want to work for big tech companies
- Doesn't matter, I just want $$$
- Front-end (web interface) → Web → Back-end ("brain" behind a website) → I want to work for...
  - Corporate → What do you think about Microsoft?
    - I'm a fan!
    - Not Bad
    - Suck
  - Startup
- 3D/Gaming
- Mobile → Which OS?
  - iOS
  - Android
- Enterprise

Do you want to try something new with huge potential, but less mature?

- NOT SURE
- NO → Which one is your favourite toy?
  - Lego
  - Play-Doh
  - I've an old & ugly toy, but i love it so much!
- YES → Web → Does your web app provides info in real-time, like twitter?
  - YES → JS
  - NO

THE LORD OF THE RINGS ANALOGY TO PROGRAMMING LANGUAGES

Python
The Ent

DIFFICULTY ★☆☆☆☆

Help little Hobbits (beginners) to understand programming concepts

Help Wizards (computer scientists) to conduct researches

Widely regarded as the best programming language for beginners

Easiest to learn

Widely used in scientific, technical & academic field, i.e. Artificial Intelligence

You can build website using Django, a popular Python web framework

POPULARITY ★★★★½

USED TO BUILD YouTube, Instagram, Spotify

AVG. SALARY $107,000

Java
Gandalf

DIFFICULTY ★★★☆☆

Wants peace & works with everyone (portable)

Very popular on all platforms, OS, and devices due to its portability

One of the most in demand & highest paying programming languages

Slogan: write once, work everywhere

POPULARITY ★★★★★

USED TO BUILD Gmail, Minecraft, Most Android Apps, Enterprise applications

AVG. SALARY $102,000

C
One Ring

DIFFICULTY ★★★☆☆

The power of C is known to them all

Everyone wants to get its Power

Lingua franca of programming language

One of the oldest and most widely used language in the world

Popular language for system and hardware programming

A subset of C++ except the little details

POPULARITY ★★★★★

USED TO BUILD Operating systems and hardware

AVG. SALARY $102,000

C++
Saruman

DIFFICULTY ★★★★☆

Everyone thinks that he is the good guy

But once you get to know him, you will realize he wants the power, not good deeds

Complex version of C with a lot more features

Widely used for developing games, industrial and performance-critical applications

Learning C++ is like learning how to manufacture, assemble, and drive a car

Recommended only if you have a mentor to guide you

POPULARITY ★★★★½

USED TO BUILD Operating systems, hardware, and browsers

AVG. SALARY $104,000

JavaScript
Hobbit

DIFFICULTY ★★☆☆☆

Frequently underestimated (powerful)

Well-known for the slow, gentle life of the Shire (web browsers)

"Java and Javascript are similar like Car and Carpet are similar" - Greg Hewgill

Most popular clients-side web scripting language

A must learn for front-end web developer (HTML and CSS as well)

One of the hottest programming language now, due to its increasing popularity as server-side language (node.js)

POPULARITY ★★★★☆

USED TO BUILD Paypal, front-end of majority websites

AVG. SALARY $99,000

C#
Elf

DIFFICULTY ★★★☆☆

Beautiful creature (language), used to stay in their land, Rivendell (Microsoft Platform), but recently started to open up to their neighbours (open source)

A popular choice for enterprise to create websites and Windows application using .NET framework

Can be used to build website with ASP.NET, a web framework from Microsoft

Similar to Java in basic syntax and some features

POPULARITY ★★★★½

USED TO BUILD Enterprise and Windows applications

AVG. SALARY $94,000

Ruby
Man (Middle Earth)

DIFFICULTY ★★☆☆☆

Very emotional creature

They (some Ruby developers) feel they are superior & need to rule the Middle Earth

Mostly known for its popular web framework, Ruby on Rails

Focuses on getting things done

Designed for fun and productive coding

Best for fun and personal projects, startups, and rapid development

POPULARITY ★★★½☆

USED TO BUILD Hulu, Groupon, Slideshare

AVG. SALARY $107,000

PHP
Orc

DIFFICULTY ★★☆☆☆

Ugly guy (language) and doesn't respect the rules (inconsistent and unpredictable)

Big headache to those (developers) to manage them (codes)

Yet still dominates the Middle-earth (most popular web scripting language)

Suitable for building small and simple sites within a short time frame

Supported by almost every web hosting services with lower price

POPULARITY ★★★★☆

USED TO BUILD Wordpress, Wikipedia, Flickr

AVG. SALARY $89,000

Objective-C
Smaug

DIFFICULTY ★★★☆☆

Lonely and loves gold

Primary language used by Apple for Mac OS X & iOS

Choose this if you want to focus on developing iOS or OS X apps only

Consider to learn Swift (newly introduced by Apple in 2014) as your next language

POPULARITY ★★★☆☆

USED TO BUILD Most iOS Apps and part of Mac OS X

AVG. SALARY $107,000

ACTUALLY...
IT DOESN'T REALLY MATTER HOW YOU START.

You need to know at least few languages to understand the underlying concepts. Just get your feet wet!

TO GET STARTED, CHECK OUT THE FULL LIST OF BEST TUTORIALS AND TOOLS FOR EACH PROGRAMMING LANGUAGE AT:

CARLCHEO.COM/STARTCODING

SPECIAL THANKS TO
Prithviraj Udaya for his awesome The Lord of the Rings analogy on Quora: http://www.quora.com/If-there-was-a-war-of-programming-languages-which-would-you-support-and-why

SOURCES
Salary data from Indeed.com | http://stackoverflow.com/questions/245062/whats-the-difference-between-javascript-and-java | http://spectrum.ieee.org/static/interactive-the-top-programming-languages
http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext | http://www.itworld.com/article/2693638/big-data/the-most-in-demand--and-valuable--programming-languages.html

IMAGES
http://nightfurylive.com/media/2013/12/Smaug-fan-art-under-the-lonely-mountain.jpg | http://blog.elope.com/five-things-you-should-know-about-the-one-ring/

PRESENTED BY

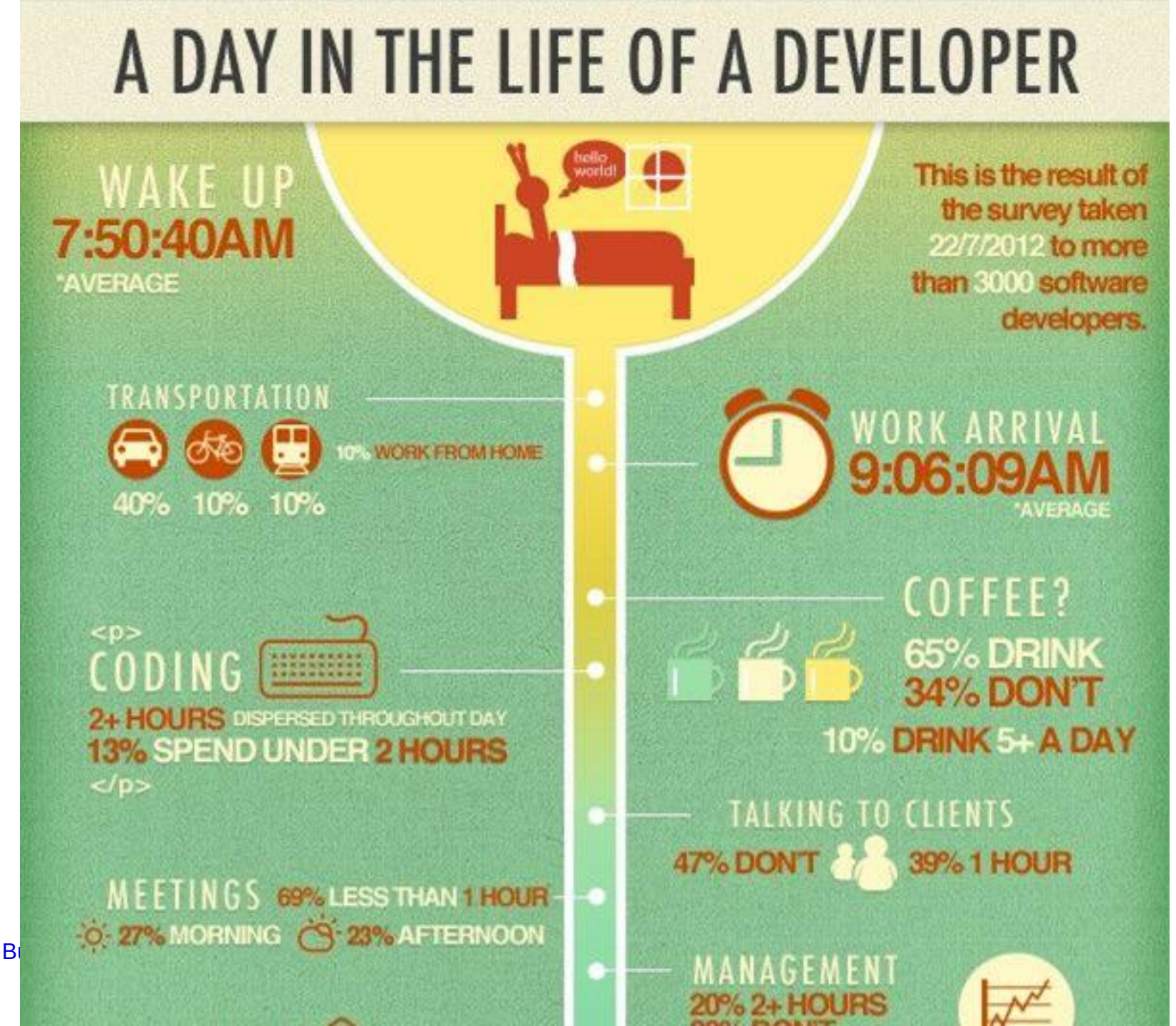
CarlCheo.com

PRÁCE PROGRAMÁTORA

Práce programátora

- Algoritmické myšlení
- Používání nástrojů, automatizace práce
- Skládání systémů z existujících celků (knihovny, kombinace jazyků)
- Schopnost práce s neurčitostí, proměnlivostí, předvídání dopadů
- Učení se – schopnost rychle si osvojit novou technologii, vyhodnotit rizika, přemýšlet předem nad problémy, když technologie zastará
- Komunikace se zákazníkem a kolegy, manažerská práce, emaily

Den v životě programátora



https://www.pinterest.com/pin/426153183465817785/

MUNI FI

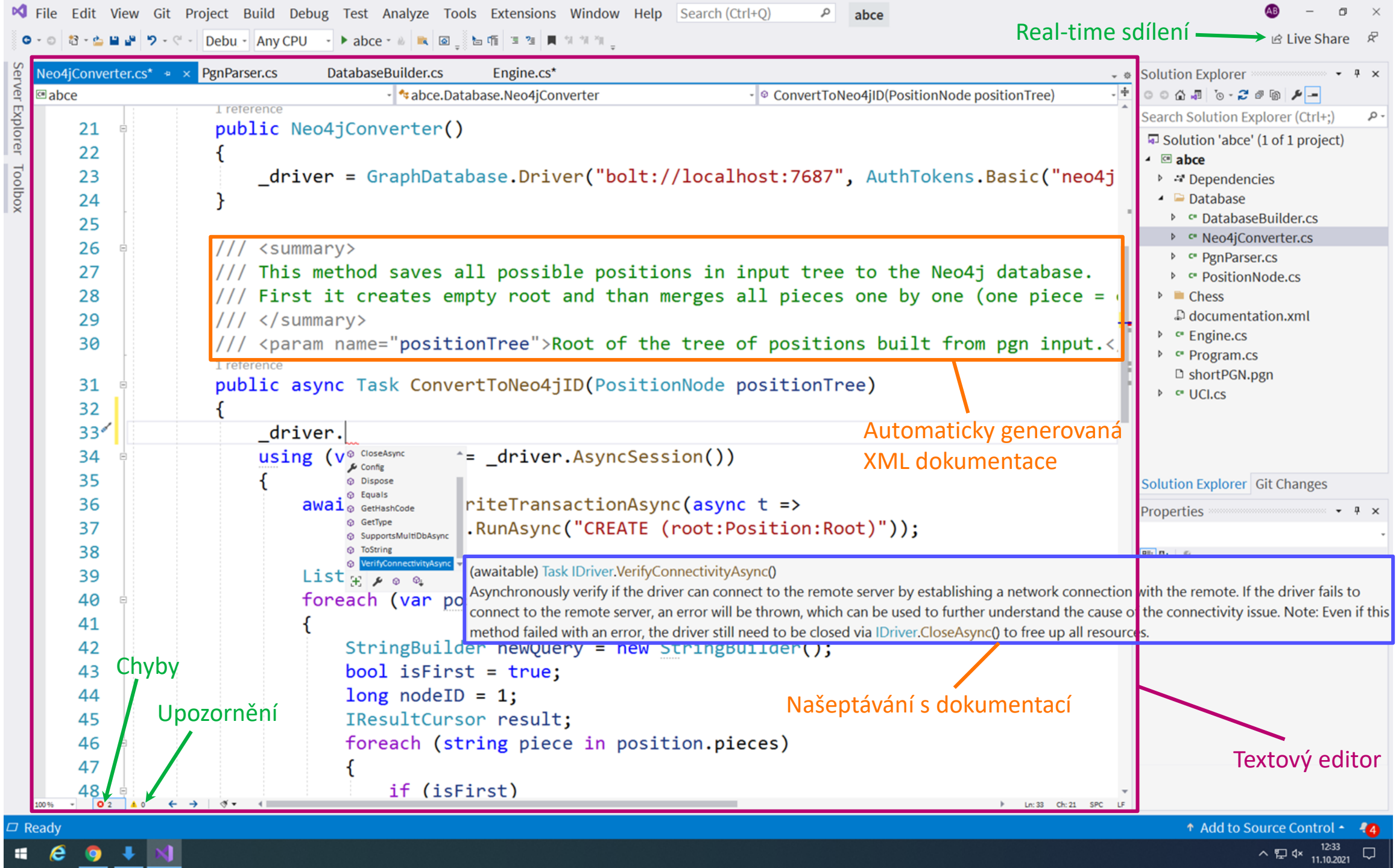
Den v životě programátora

https://www.pinterest.com/pin/426153183465817785/

MUNI FI

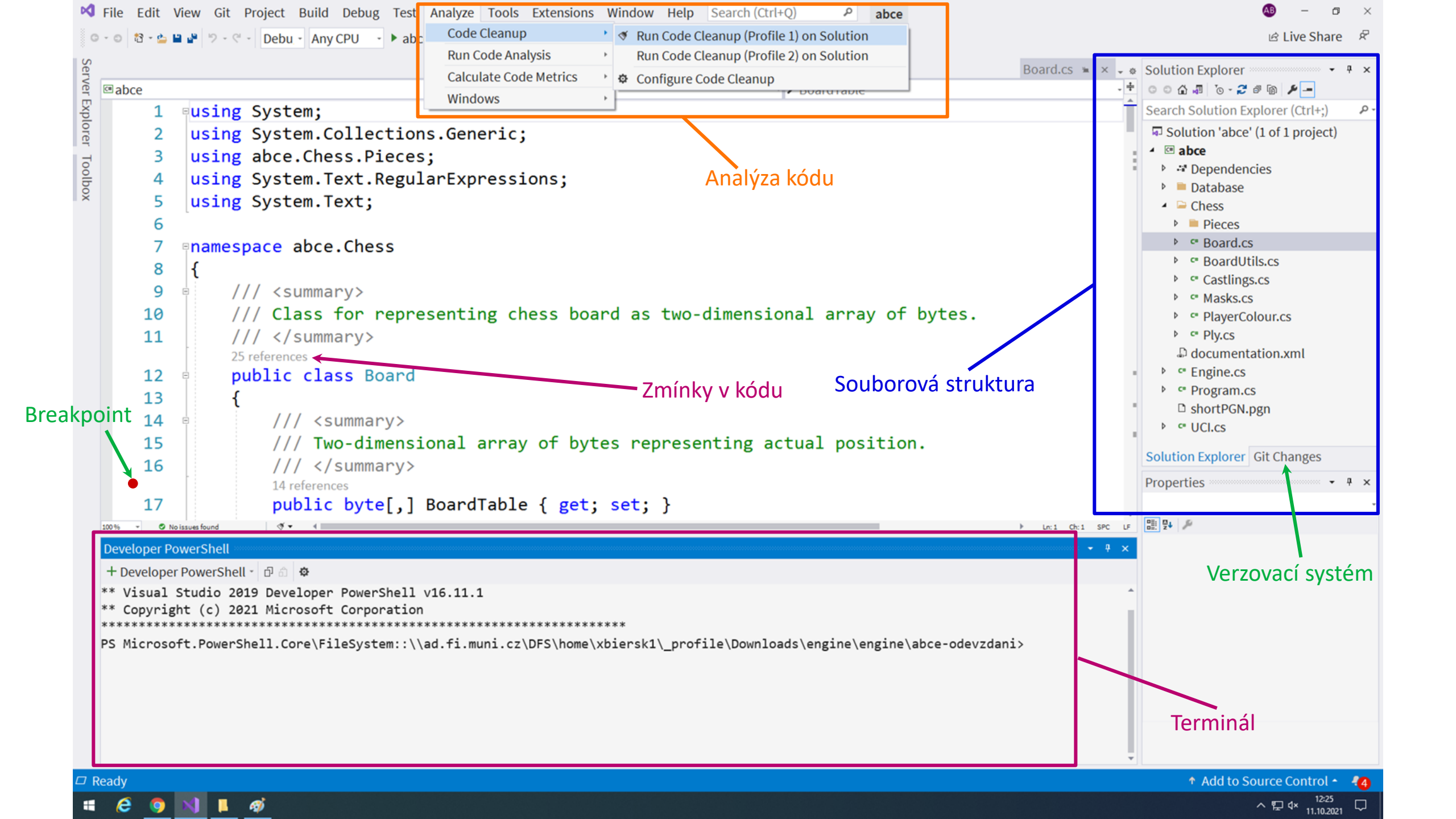
Jaké nástroje využívám pri programování?

- IDE (Integrated Development Environment)
  - Komplexní nástroj pro vývoj
  - Obsahuje nejruznejší nástroje (vcetne analytických), ale je velmi nárocný na hardware
  - Např. prostredí „Visual Studio“.
- Editor
  - Nástroj, který stejne jako IDE dokáže podbarvovat kód a tím velmi zjednodušit programování.
  - Je určen pro vývoj méne nárocných projektu, neobsahuje žádné pokrocilé nástroje
  - Oproti IDE má velmi nízké nároky na hardware
  - Např. editor „Visual Studio Code“.


Real-time sdílení
Live Share
Automaticky generovaná XML dokumentace
Našeptávání s dokumentací
Textový editor
Chyby
Upozornění
Solution Explorer
Git Changes
Properties
(awaitable) Task IDriver.VerifyConnectivityAsync()
Asynchronously verify if the driver can connect to the remote server by establishing a network connection with the remote. If the driver fails to connect to the remote server, an error will be thrown, which can be used to further understand the cause of the connectivity issue. Note: Even if this method failed with an error, the driver still need to be closed via IDriver.CloseAsync() to free up all resources.

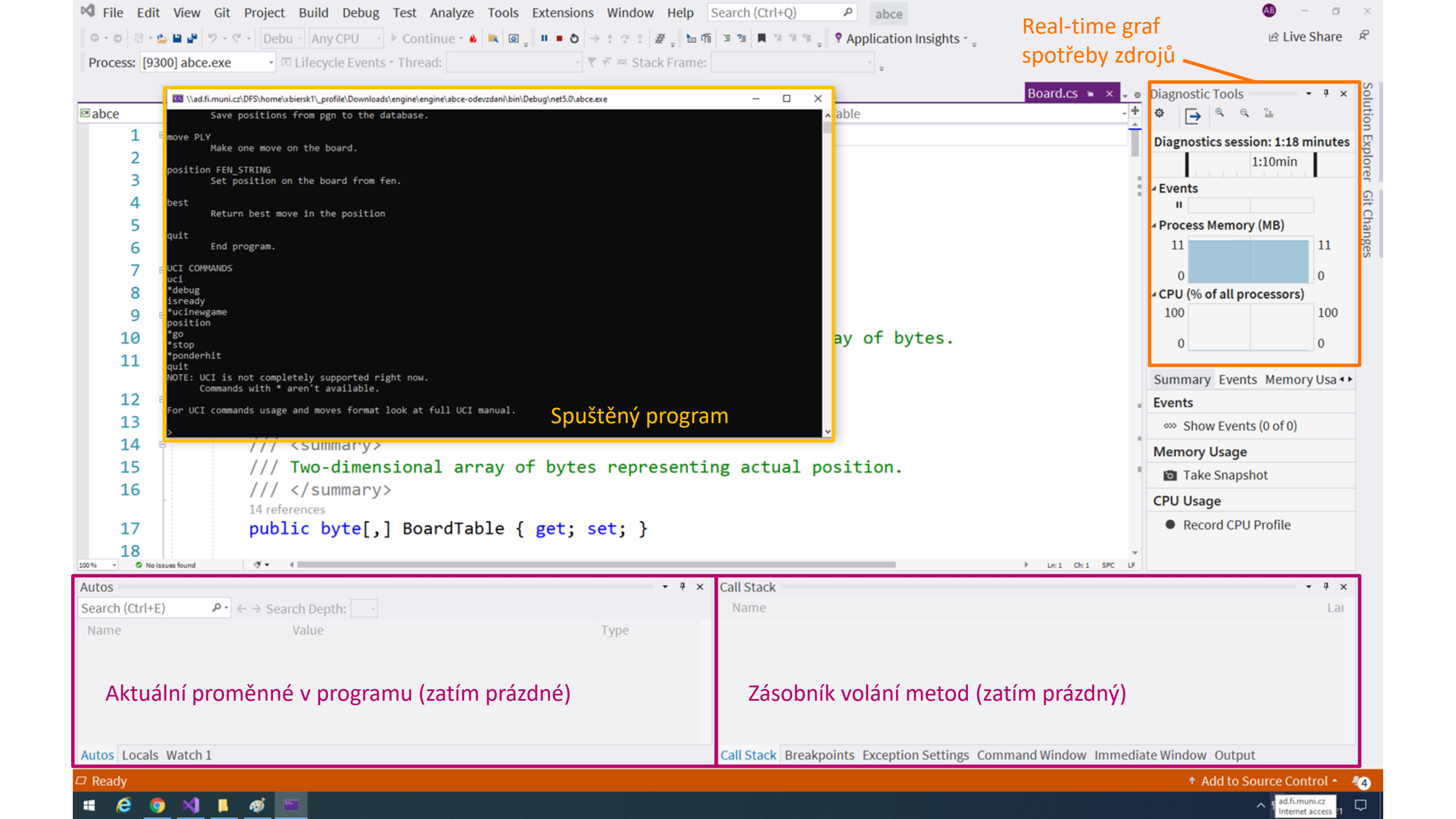
IDE – Základní pohled na Visual Studio 2019

- Fialově ohraničené je okno, kde upravujeme kód.
- Uprostřed oranžově je krátká XML dokumentace, kterou dnešní IDE dokáží samy generovat (vytáhnout z funkce název a parametry, vytvořit šablonu a programátor jen vyplní obsah).
- Vpravo nahoře je Live Share - nástroj pro real-time spolupráci.
- Vlevo dole vidíme kolik chyb a upozornění je aktuálně v kódu.
- Uprostřed vidíme rozepsanou funkci, kterou se nám našeptávač snaží doplnit.


Analyze
Code Cleanup
Run Code Cleanup (Profile 1) on Solution
Run Code Cleanup (Profile 2) on Solution
Configure Code Cleanup
Run Code Analysis
Calculate Code Metrics
Windows
Analýza kódu
using System;
using System.Collections.Generic;
using abce.Chess.Pieces;
using System.Text.RegularExpressions;
using System.Text;
namespace abce.Chess
{
/// <summary>
/// Class for representing chess board as two-dimensional array of bytes.
/// </summary>
25 references
public class Board
{
/// <summary>
/// Two-dimensional array of bytes representing actual position.
/// </summary>
14 references
public byte[,] BoardTable { get; set; }
Zmínky v kódu
Breakpoint
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'abce' (1 of 1 project)
abce
Dependencies
Database
Chess
Pieces
Board.cs
BoardUtils.cs
Castlings.cs
Masks.cs
PlayerColour.cs
Ply.cs
documentation.xml
Engine.cs
Program.cs
shortPGN.pgn
UCI.cs
Souborová struktura
Solution Explorer
Git Changes
Properties
Verzovací systém
Developer PowerShell
** Visual Studio 2019 Developer PowerShell v16.11.1
** Copyright (c) 2021 Microsoft Corporation
PS Microsoft.PowerShell.Core\FileSystem::\\ad.fi.muni.cz\DFS\home\xbiersk1_profile\Downloads\engine\engine\abce-odevzdani>
Terminál
Ready
Add to Source Control

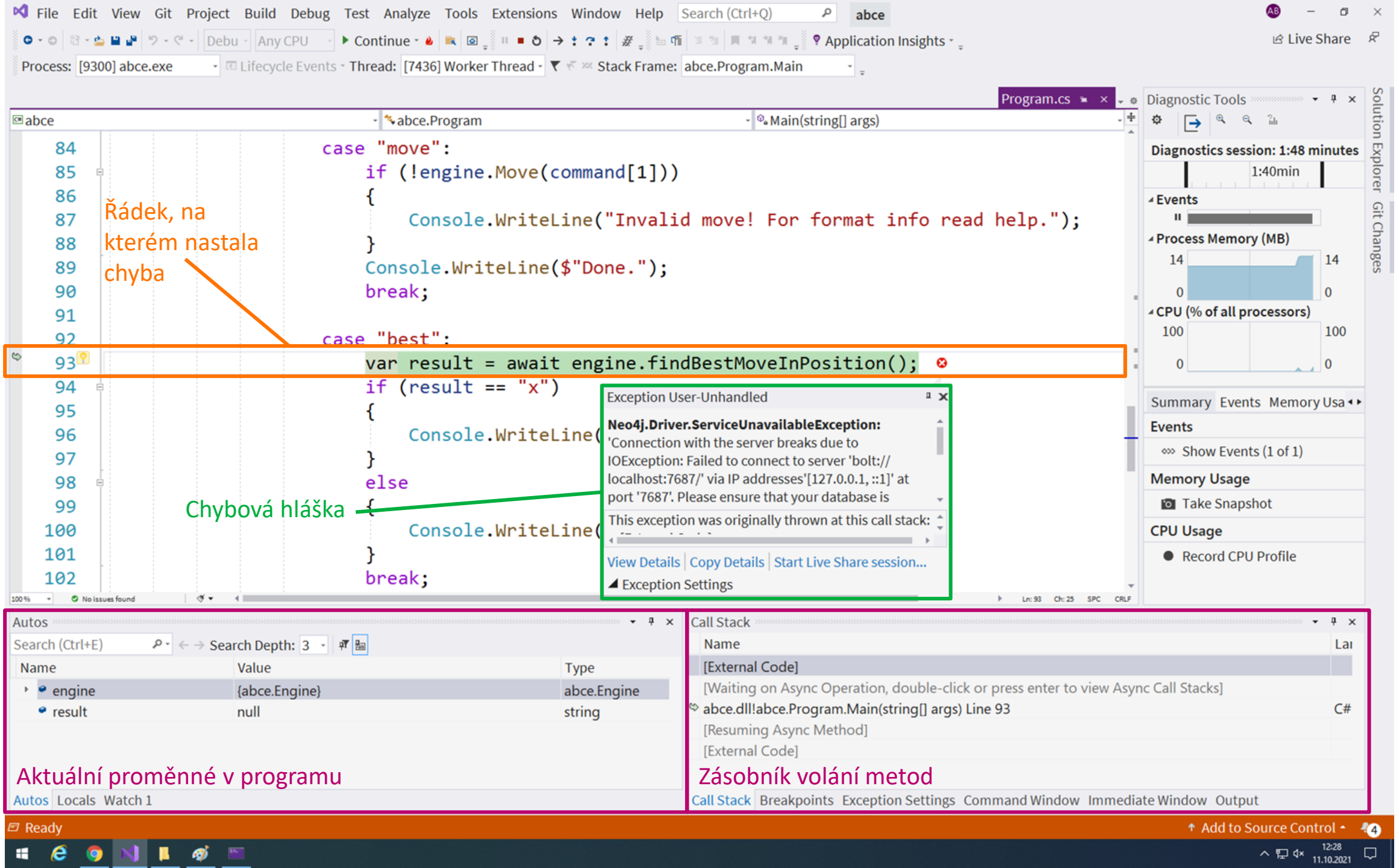
IDE – Další funkcionalita ve Visual Studiu 2019

- Nahoře vidíme nabídku s čištěním a analýzou kódu.
- Napravo je okno se souborovou strukturou celého projektu s možným překlikem do gitu.
- Uprostřed fialově reference (kolikrát se funkce vyskytuje ve zbytku kódu).
- Nalevo debuggovací breakpoint - místo, kde debugging zastaví, abychom mohli zjistit stav programu a krokovat.
- Dole terminál - okno pro přímé příkazy operačnímu systému.


Real-time graf spotřeby zdrojů
Spuštěný program
Aktuální proměnné v programu (zatím prázdné)
Zásobník volání metod (zatím prázdný)
Diagnostic Tools
Diagnostics session: 1:18 minutes
1:10min
Events
Process Memory (MB)
CPU (% of all processors)
Summary
Events
Memory Usage
CPU Usage
Show Events (0 of 0)
Take Snapshot
Record CPU Profile
Autos
Locals
Watch 1
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Ready
Add to Source Control

IDE – Úvodní stav debuggingu ve Visual Studiu 2019

- Uprostřed je okno se spuštěným programem - jedná se o konzolovou aplikaci (šachový engine), která aktuálně zobrazuje úvodní nápovědu k použití a čeká na svůj první příkaz.
- Proto je stack prázdný a nejsou v seznamu žádné proměnné.
- Napravo vidíme aktuální spotřebu procesoru a paměti, zároveň i délku debuggingu.

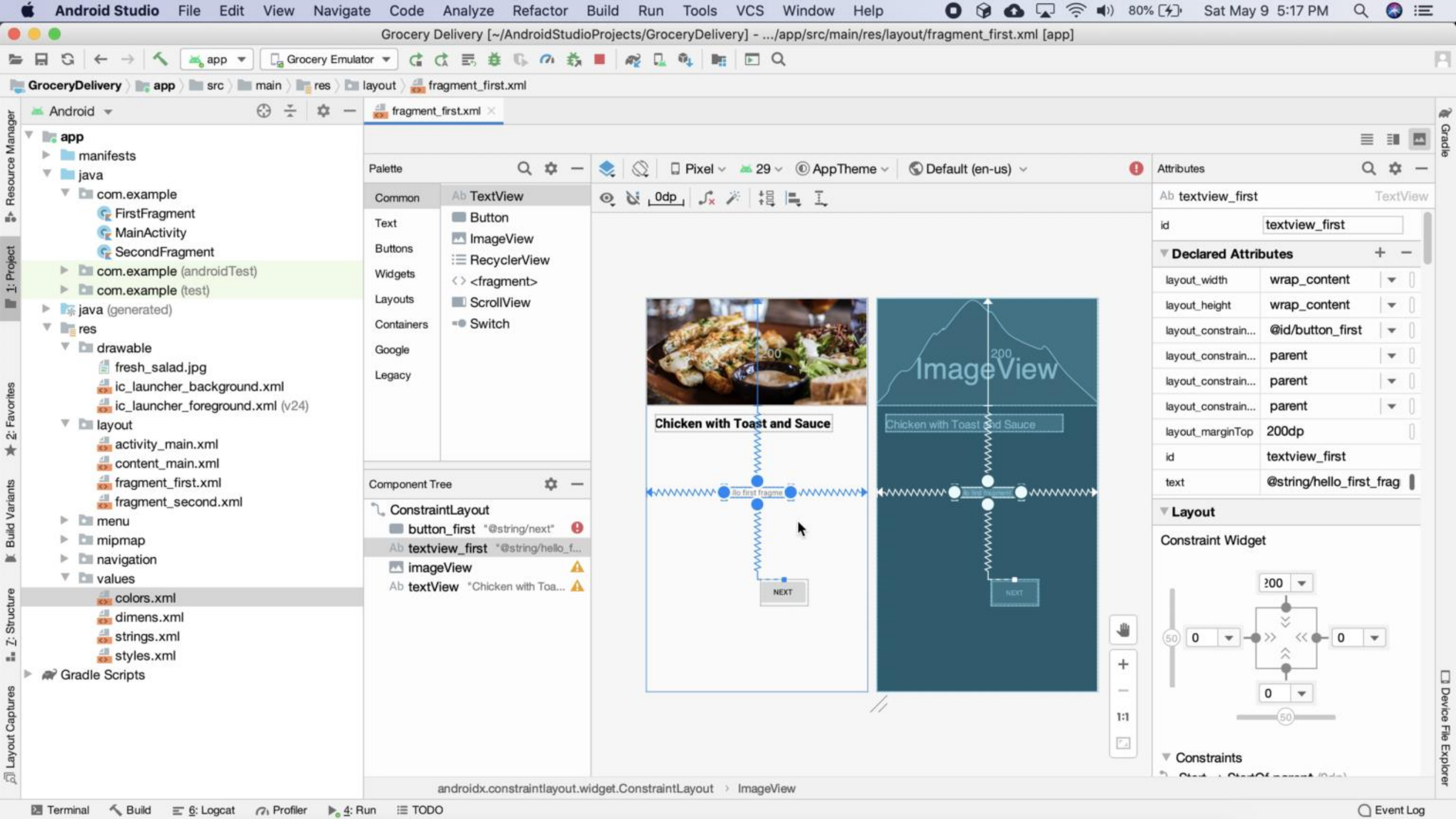

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
abce
Live Share
Application Insights
Process: [9300] abce.exe
Lifecycle Events
Thread: [7436] Worker Thread
Stack Frame: abce.Program.Main
Program.cs
abce
abce.Program
Main(string[] args)
84 case "move":
85 if (!engine.Move(command[1]))
86 {
87 Console.WriteLine("Invalid move! For format info read help.");
88 }
89 Console.WriteLine($"Done.");
90 break;
91
92 case "best":
93 var result = await engine.findBestMoveInPosition();
94 if (result == "x")
95 {
96 Console.WriteLine(
97 }
98 else
99 {
100 Console.WriteLine(
101 }
102 break;
Řádek, na kterém nastala chyba
Chybová hláška
Exception User-Unhandled
Neo4j.Driver.ServiceUnavailableException:
'Connection with the server breaks due to IOException: Failed to connect to server 'bolt://localhost:7687/' via IP addresses'[127.0.0.1, ::1]' at port '7687'. Please ensure that your database is
This exception was originally thrown at this call stack:
View Details
Copy Details
Start Live Share session...
Exception Settings
Diagnostic Tools
Diagnostics session: 1:48 minutes
1:40min
Events
Process Memory (MB)
CPU (% of all processors)
Summary
Events
Memory Usa
Show Events (1 of 1)
Memory Usage
Take Snapshot
CPU Usage
Record CPU Profile
Solution Explorer
Git Changes
Autos
Search (Ctrl+E)
Search Depth: 3
Name
Value
Type
engine
{abce.Engine}
abce.Engine
result
null
string
Aktuální proměnné v programu
Autos
Locals
Watch 1
Call Stack
Name
[External Code]
[Waiting on Async Operation, double-click or press enter to view Async Call Stacks]
abce.dll!abce.Program.Main(string[] args) Line 93
C#
[Resuming Async Method]
[External Code]
Zásobník volání metod
Call Stack
Breakpoints
Exception Settings
Command Window
Immediate Window
Output
Ready
Add to Source Control
12:28
11.10.2021

IDE – Debugging (chyba) ve Visual Studiu 2019

- Program narazil na chybu -> zobrazí řádek, přes který nemůže pokračovat a chybovou hlášku.
- Z té můžeme vyčíst, že se nebylo možné připojit k serveru s databází (příkaz “best” má vracet nejlepší tah v pozici, kterou najde v databázi, ta ale během pořizování screenshotu nebyla spuštěná)
- Z hlášky můžeme vyčíst o jakou databázi se jedná (Neo4j zmíněná pár snímků zpět), přes jakou IP adresu a port se snažíme připojit.
- Na konci hlášky se nám snaží napovědět možné řešení “Please ensure ....”

IDE – Debugging (chyba) ve Visual Studiu 2019

- V okně s proměnnými vidíme dvě, jeden je objekt Engine, na kterém voláme metody pro hledání tahů. Druhý je result, kam se měl uložit string s nejlepším tahem, ale jeho hodnota je null, protože se pouze inicializoval, ale pak nastala chyba.
- Na zásobníku už taky vidíme volání metod.
- V diagnostic tools vpravo vidíme, že chyba nastala po minutě a 48 vteřinách běhu debuggingu.


Android Studio File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Grocery Delivery [~/AndroidStudioProjects/GroceryDelivery] - .../app/src/main/res/layout/fragment_first.xml [app]
GroceryDelivery app src main res layout fragment_first.xml
Android
app
manifests
java
com.example
FirstFragment
MainActivity
SecondFragment
com.example (androidTest)
com.example (test)
java (generated)
res
drawable
fresh_salad.jpg
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
layout
activity_main.xml
content_main.xml
fragment_first.xml
fragment_second.xml
menu
mipmap
navigation
values
colors.xml
dimens.xml
strings.xml
styles.xml
Gradle Scripts
fragment_first.xml
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy
TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch
Component Tree
ConstraintLayout
button_first "@string/next"
textview_first "@string/hello_f...
imageView
textView "Chicken with Toa...
Pixel
29
AppTheme
Default (en-us)
0dp
Chicken with Toast and Sauce
NEXT
ImageView
Attributes
textview_first
TextView
id
textview_first
Declared Attributes
layout_width
wrap_content
layout_height
wrap_content
layout_constrain...
@id/button_first
parent
parent
parent
layout_marginTop
200dp
id
textview_first
text
@string/hello_first_frag
Layout
Constraint Widget
Constraints
androidx.constraintlayout.widget.ConstraintLayout
ImageView
Terminal
Build
6: Logcat
Profiler
4: Run
TODO
Event Log
Resource Manager
1: Project
2: Favorites
Build Variants
7: Structure
Layout Captures
Gradle
Device File Explorer

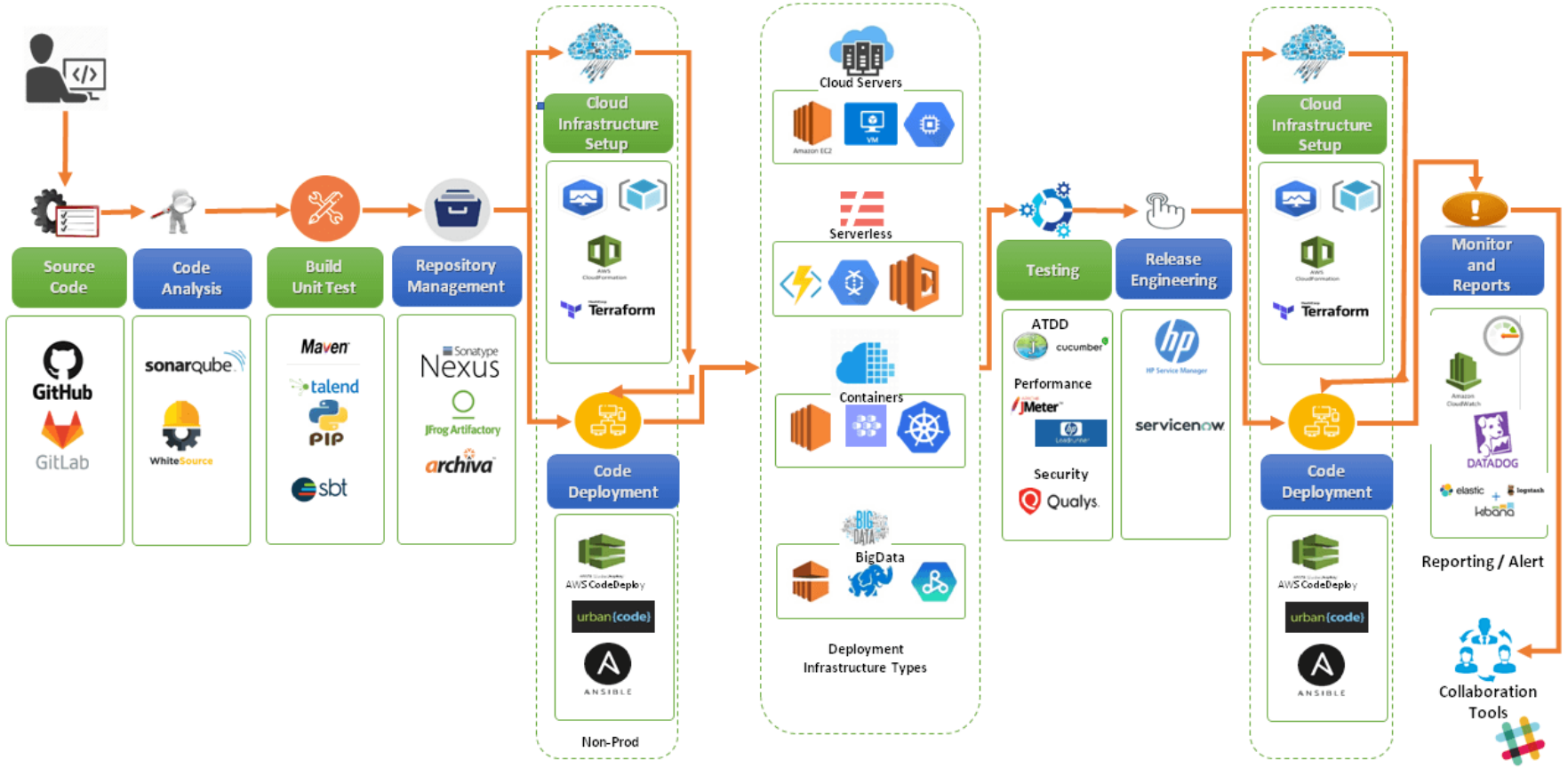
IDE – Základní pohled na Android Studio

- Vpravo simulátor mobilního telefonu, simulovat lze různá zařízení a různé verze systému Android
- Uprostřed konfigurace grafického rozhranní, nastavení tlačítek, apod.
- Vlevo adresářová struktura

Další oblíbené nástroje a prostředí vývojáře

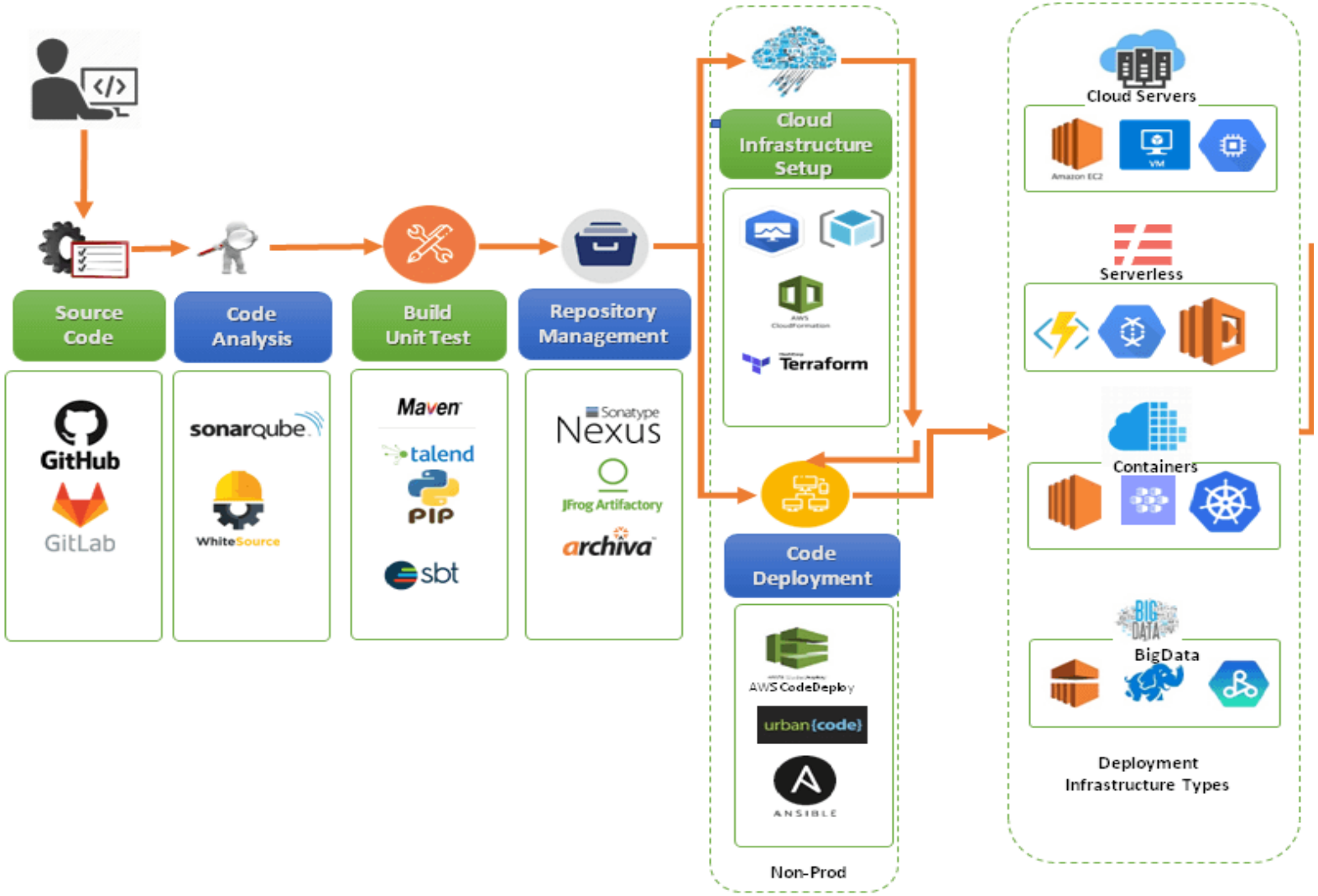
- **Requirements analysis** and design modeling tools
- **Programming environments** that automate parts of program construction processes (e.g., automated builds)
- **Software configuration management** and version control
- **Testing tools** including static and dynamic analysis tools
- **Continuous integration** and release management
- **Issue tracking**
- **Project management** tools

Nástroje



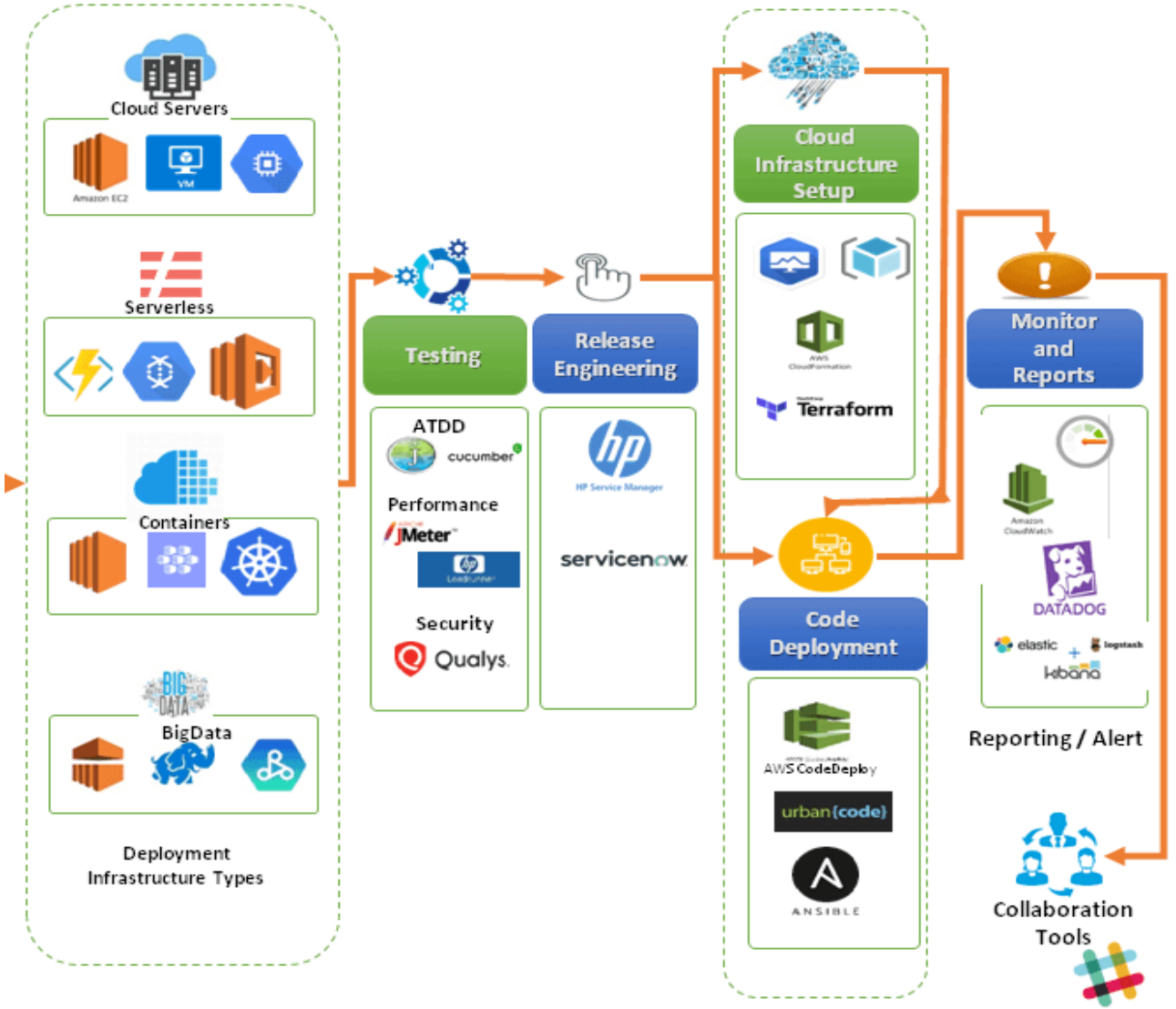
MUNI FI

Kód, infrastruktura a nasazení



MUNI FI

Testování, vydání a spolupráce



MUNI FI

CO NÁS ČEKÁ PŘÍŠTĚ

MUNI FI

5. Uživatelská přívětivost

- Uživatelské rozhranní vs. UX
- Uživatelský průzkum, modeloví uživatelé
- Modelování a mapy aplikace, wireframes
- Uživatelské testování

Domácí práce a příprava na příští přednášku

- Začátečníci – na https://code.org/ všechny základní úrovně Angry Birds, Flappy, Dance Party
- Pokročilí – projít si Refactoring Guru nebo některý z tutoriálů Czechitas nebo Compiler Explorer