Rails Girls Study group 2023

Introductory Lecture 0

24 May 2023

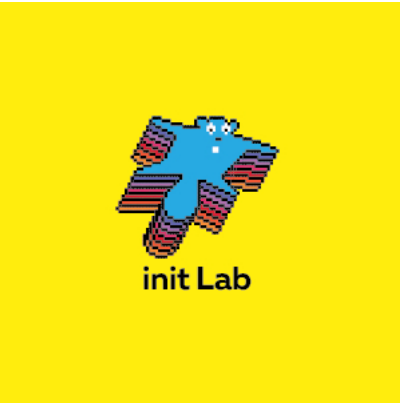
Защо сме тук?

- Преди всичко да споделяме знания
- Сблъскване с повече и по-различни проблеми
- Да научим нови умения и подходи
- Да работим по нещо което ни харесва!
- Да използваме всичко това за да направим цялостен проект!

Защо не сме тук

- Да следваме стриктна програма или очакван краен резултат

- Преди всяко събиране ще се разбираме за ден, в зависимост от това кога има налични инструктори.
- След работно време – от 19:30 до 22:00 ч.
- На всяка сбирка ще има поне един инструктор.

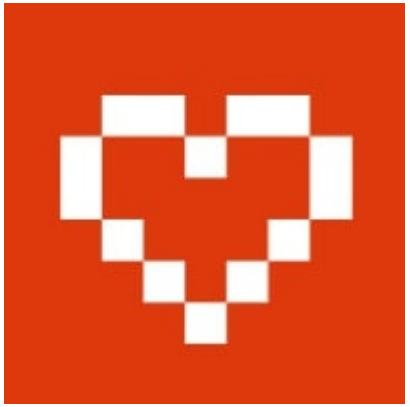


- Няма да има изрична работа за “домашно”, но ще има насоки и ще може да правите неща самостоятелно, ако имате време и желание.
- Безплатни и свободни за достъп, насочени предимно към дами.
- Няма никакви предварителни изисквания към участничките.

Уеб проект по ваша идея

- ще имате време за идеи до другата седмица
- решавате дали да работите самостоятелно или в екип
- избирате (донякъде) технологии за ползване
- ще ползваме github за съхранение на задачи и проекти
- при желание и интерес може да даваме и допълнителни материали/задачи за между сбирки
- също при желание може да подготвяме лекции от наша страна на всякакви интересни теми

- Нека си припомним какво научихме на Rails Girls:



1. HTML
2. CSS
3. Ruby
4. основна структура на всеки един уеб проект (например машината за гласуване, която всеки тук е правил)

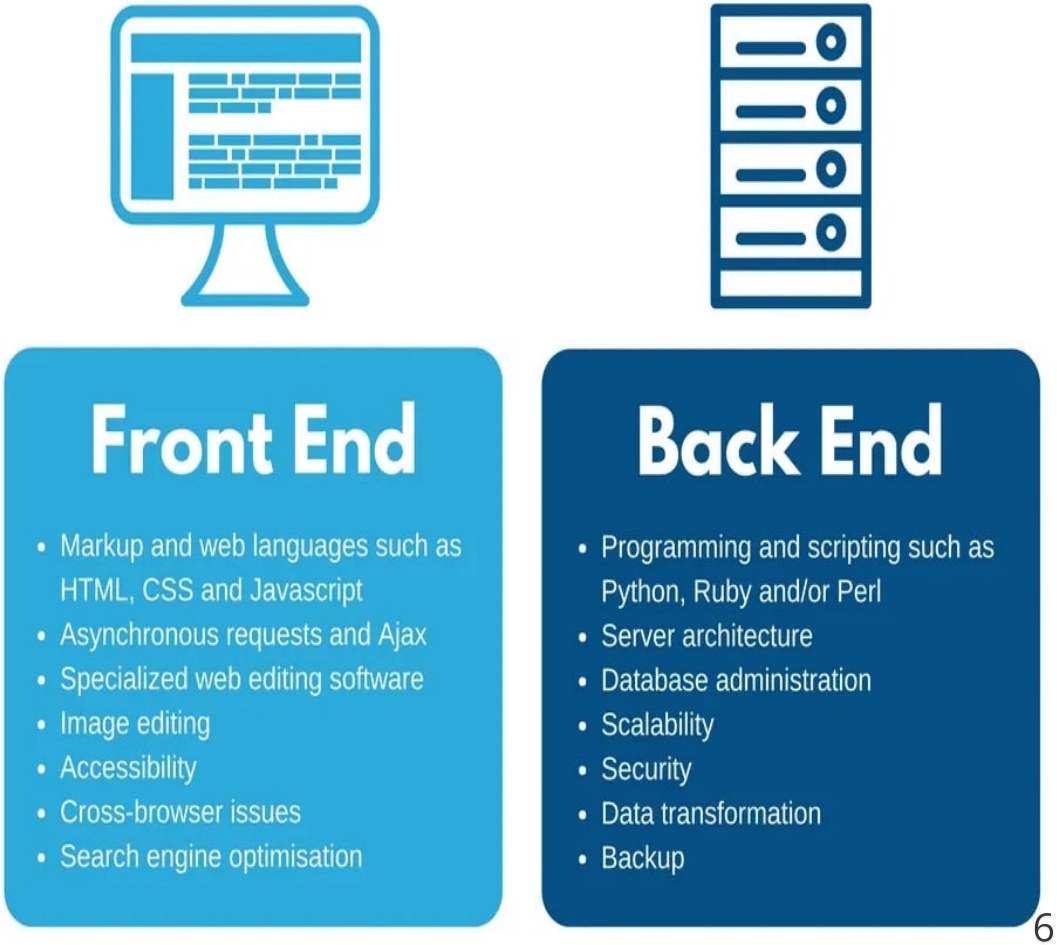
- Процес на работа
- Работа в екип
- Сблъскване с непознати проблеми

- Да изберем технологии за проекти

2. Език по избор за бекенд

3. Vue.js за фронтенд

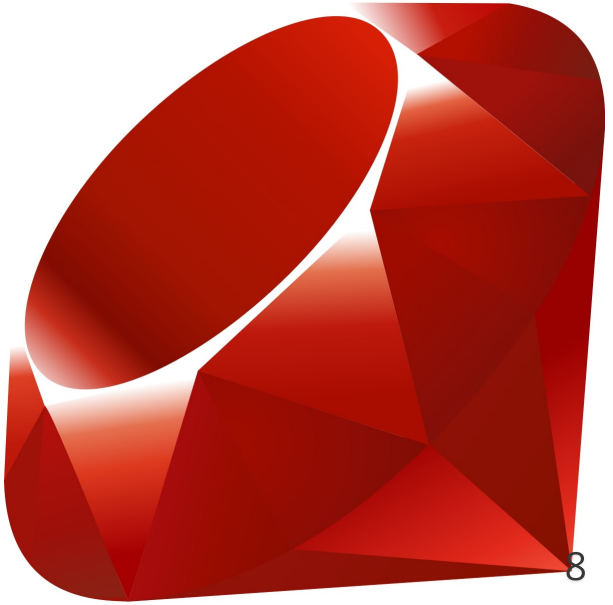
- Да си изготвим планове за действие
- (не задължително) да сформираме екипи



- Ruby
- Go
- Python
- C#
- Javascript

- pros & cons
- used by: Github, Soundcloud
- Hello world

```
puts "Hello World!"
```



Key Features of Ruby

Developer happiness

Ruby puts developer-happiness first. It has an elegant syntax that is natural to read and write.

Metaprogramming magic

Ruby code can write and invoke Ruby code.

Large standard library

Ruby's standard library provides a wealth of classes utilities for common tasks.

Garbage Collection

Garbage collection done via mark and sweep, stays out of your way.

Flexible package manager

Packages (gems) can be centrally managed, but can also include custom or private gem stores.

Strong, dynamic typing

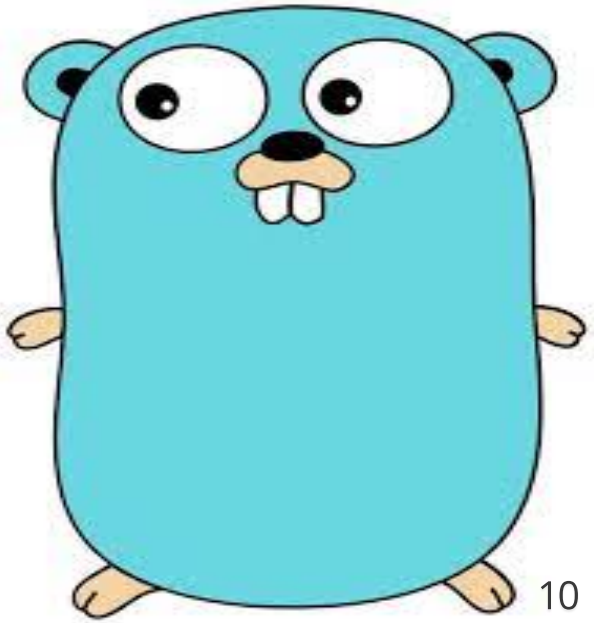
Ruby is strong and dynamically typed and supports 'Duck Typing'. Everything in Ruby is an object

- pros & cons
- used by: Google
- Hello world

```
package main

import "fmt"

func main() {
    fmt.Println("Hello world!")
}
```



Key Features of Go

Speed

Go is fast for computers and humans. Go provides fast compilation, testing, package mgmt, and more

Concurrency

Go's concurrency mechanisms make it easy to write programs for multicore and networked machines

Statically Typed

Go's type system is expressive yet lightweight, enabling flexible and modular program construction

Portable

Go compiles to a single static executable, and cross-compilation is easy

Minimal

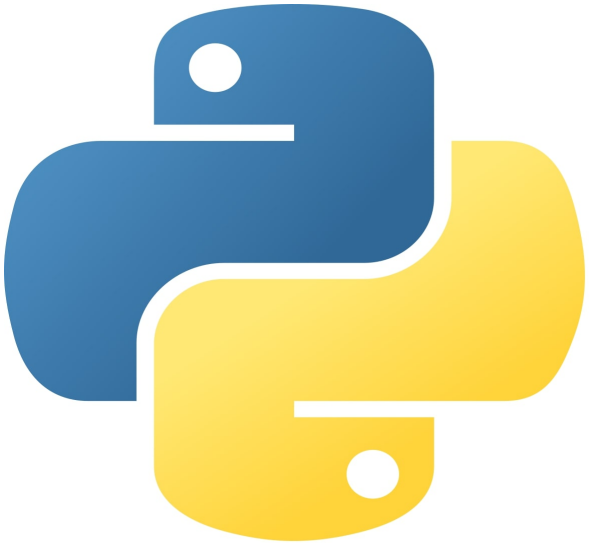
Go's design reduces clutter and complexity. Your favorite feature may be missing, by design!

Modern

Go has > 1 million developers and every major cloud provider uses core infrastructure written in Go

- pros & cons
- used by: every AI company
- Hello world

```
print("Hello world!")
```



Key Features of Python

Batteries Included

Epic & well-documented standard library. Need more? Use a PyPi package - there's one for everything.

Easy

Human-friendly Syntax and a vibrant, supportive community. Quick to learn & intuitive to use.

Extensible

Need to call Fortran from a web API? Done. Need to process images using C? Python can do that.

Flexible

Duck, dynamic, & strong typing. Easy to debug. Fun for experiments, robust for large applications.

Multi-paradigm

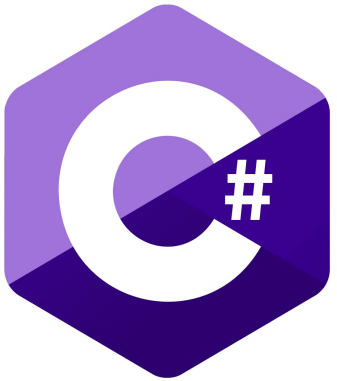
OOP, structured, functional, & aspect-oriented. Adaptable to how you structure your programs.

Ubiquitous

Accepted for almost any use. Easily interface with other languages & execute almost everywhere.

- pros & cons
- used by: Microsoft
- Hello world

```
namespace HelloWorld
{
    class Hello {
        static void Main(string[] args)
        {
            System.Console.WriteLine("Hello World!");
        }
    }
}
```



Key Features of C#

Cross-platform

C# runs on almost any platform and chipset.

Modern

C# is a modern, fast-evolving language.

General purpose

C# can be used for a wide variety of workloads, like websites, console applications, and even games.

Multi-paradigm

C# is primarily an object-oriented language, but also has lots of functional features.

Documentation

Documentation is excellent and exhaustive, making it easy to get started with C#.

Tooling

C# has excellent tooling, with linting and advanced refactoring options built-in.

- pros & cons
- used by: Everyone
- Hello world

```
console.log('Hello World');
```

Key Features of JavaScript

Runs almost everywhere

Build web-pages, write backend, create database scripts, make mobile apps, design CLI-s, and more.

Use any programming style

Use prototype-based, object-oriented, functional, or declarative programming styles, and more.

No types required

Dynamically and weakly typed by default, gain typed confidence using Flow, JSDoc, or TypeScript.

Concurrency is safe

Async/await, dedicated workers, or state sync in shared workers. No deadlocks or race conditions.

Largest package registry

No need to reinvent the wheel. Build on top of > 1.3 million packages (April, 2020).

Designed by a committee

Frequent updates, following ECMAScript, a general purpose, cross platform, vendor-neutral standard.

- pros & cons
- Vue.js
- Защо избрахме това?
- Hello World

HTML

```
<div id="app">
  {{ message }}
</div>
```

Vue

```
new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue.js!'
  }
})
```

- Основи на езика, който изберете за бекенд
- Основи на javascript и по-конкретно Vue.js
- Основни структури и функции
- Работа с бази данни
- Работа с файлове
- Тестване
- Често ползвани практики и дизайн
- Структура и архитектура на проекта (ще бъде направена от инструктурите, но ще трябва да знаете защо е така)
- и мнооого други неща

- така е, затова нашата предложение за подход е:

Да направим още 1-2 сбирки с по-учебно съдържание, но основният начин за учене на нови неща ще е проба и грешка по време на работа по избрания проект.

- Защо? Така най-лесно ще имаме постоянен фийдбек и виждаме какво работи или не работи както искаме.Също няма да има нужда да минат няколко такива срещи преди да започнем реална работа.
- Идеи? Това е само начална идея - ако измислим по-добър вариант винаги можем да се адаптираме

- най-първоначални идеи 💡
- ще започнем с малко основни задачки 📝
- при желание може да покажем още допълнителни за вкъщи 🙌

Rails Girls

Page (https://rails-girls-sofia-study-group.github.io) Git (https://github.com/rails-girls-sofia-study-group)

Discord (https://discord.gg/VqKUzMgbWt)

Git

Git (https://github.com/)

Frontend

vuejs (https://v1.vuejs.org/)

Backend

Ruby (https://www.ruby-lang.org/en/) Go (https://go.dev/) Python (https://www.python.org/) JS (https://www.javascript.com/) C# (https://learn.microsoft.com/en-us/dotnet/csharp/)

Introductory Lecture 0

24 May 2023