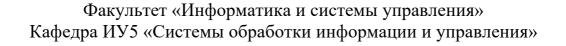
Московский государственный технический университет им. Н.Э. Баумана



Курс «Парадигмы и конструкции языков программирования»

Отчет по домашней работе «Приложение на SolidJS и Tauri»

Выполнил: студент группы ИУ5-32Б Мажитов В. Проверил: преподаватель каф. ИУ5 Гапанюк Ю.Е.

Описание задания

- 1. Выберите язык программирования (который Вы ранее не изучали) и (1) напишите по нему реферат с примерами кода или (2) реализуйте на нем небольшой проект (с детальным текстовым описанием).
- 2. Реферат (проект) может быть посвящен отдельному аспекту (аспектам) языка или содержать решение какой-либо задачи на этом языке.
- 3. Необходимо установить на свой компьютер компилятор (интерпретатор, транспилятор) этого языка и произвольную среду разработки.
- 4. В случае написания реферата необходимо разработать и откомпилировать примеры кода (или модифицировать стандартные примеры).
 - 5. В случае создания проекта необходимо детально комментировать код.
- 6. При написании реферата (создании проекта) необходимо изучить и корректно использовать особенности парадигмы языка и основных конструкций данного языка.
 - 7. Приветствуется написание черновика статьи по результатам выполнения ДЗ.
- 8. Черновик статьи может быть подготовлен группой студентов, которые исследовали один и тот же аспект в нескольких языках или решили одинаковую задачу на нескольких языках.

Текст программы

Файл app.tsx

```
import { createSignal } from "solid-js"
import { invoke } from "@tauri-apps/api/tauri"
function App() {
  const [best, setBest] = createSignal<number | null>(null)
  const [firstThing, setFirstThing] = createSignal("")
  const [secondThing, setSecondThing] = createSignal("")
  const [loading, setLoading] = createSignal(false)
  async function getBest() {
    setLoading(true)
    invoke("get_best", { firstThing: firstThing(), secondThing: secondThing()
}).then(value => {
      setBest((_) => value as number)
      setLoading(false)
    })
  return (
        class="flex flex-col space-y-4 pt-4 px-4 h-screen dark:text-white dark:bg-
black"
        onSubmit={(e) => {
          e.preventDefault()
          getBest()
        }}
```

```
<h1 class="text-3xl">Что круче?</h1>
       <div class="flex flex-row items-center space-x-4">
         <input
           id="1"
           class="input"
           onInput={(e) => {setFirstThing(e.currentTarget.value); setBest(null)}}
           placeholder="Введи что угодно"
         />
         или
         <input
           class="input"
           onInput={(e) => {setSecondThing(e.currentTarget.value); setBest(null)}}
           placeholder="Введи что угодно"
         />
       </div>
       <button
         disabled={firstThing().length === 0 || secondThing().length === 0 ||
loading()}
         type="submit"
         class="btn-primary transition-all"
         Узнать
       </button>
       {best() && {[firstThing(), secondThing()][best()!-1]}
круче!}
     </form>
 );
export default App;
```

Файл index.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;

@layer components {
    .btn-primary {
        @apply transition-all py-2 px-4 bg-blue-500 text-white font-semibold
rounded-lg shadow-md hover:bg-blue-700 focus:outline-none focus:ring-2 focus:ring-
blue-400 focus:ring-opacity-75 disabled:bg-gray-700;
    }

    .input {
        @apply bg-gray-50 border border-gray-300 text-gray-900 text-sm rounded-lg
focus:ring-blue-500 focus:border-blue-500 block w-full p-2.5 dark:bg-gray-700
```

```
dark:border-gray-600 dark:placeholder-gray-400 dark:text-white dark:focus:ring-
blue-500 dark:focus:border-blue-500
    }
}
```

Файл main.rs

```
// Prevents additional console window on Windows in release, DO NOT REMOVE!!
#![cfg_attr(not(debug_assertions), windows_subsystem = "windows")]
use rtrend::{Client, Country, Keywords, SearchInterest};
use serde_json::Number;
fn get_metric_for_keyword(keyword: &str) -> u64 {
    let keywords: &'static str = String::from(keyword.split("
").collect::<Vec<&str>>()[0]).leak();
    let client = Client::new(Keywords::from(keywords), Country::ALL).build();
    let search_interest = SearchInterest::new(client).get();
    let metric = search_interest["default"]["timelineData"]
        .as_array()
        .unwrap()
        .last()
        .unwrap()
        .get("value")
        .unwrap()[0]
        .as_u64()
        .unwrap();
    metric
#[tauri::command]
fn get_best(first_thing: &str, second_thing: &str) -> Number {
    let first_rank = get_metric_for_keyword(first_thing);
    let second_rank = get_metric_for_keyword(second_thing);
    if first_rank > second_rank {
        Number::from(1)
    } else {
        Number::from(2)
fn main() {
    tauri::Builder::default()
        .invoke_handler(tauri::generate_handler![get_best])
        .run(tauri::qenerate_context!())
        .expect("error while running tauri application");
```

Пример выполнения программы

) python3 main.py

Прямоугольник: ширина: 15, высота: 10, цвет: синий Круг: радиус: 15, цвет: зеленый

Круг: радиус: 15, цвет: зеленыи Квадрат: сторона: 15, цвет: красный