Pl 🇵🇱 Eng 🇬🇧

Patryk Jaseniuk

Nysa, Polska

patryk.jaseniuk@gmail.com

LinkedIn

GitHub

About Me 👨‍💻

Hello, I'm Patryk, and I enjoy working with computers using TYPESCRIPT and REACT. Currently, I'm focusing on developing web applications, especially user interfaces. I would like to join a team of developers and create something beautiful and useful.

Projects 📁

Sailing Ships Simulator

TYPESCRIPT NEXT JEST

A 2D simulator/game written in TypeScript. Users can control a ship by adjusting sail and rudder parameters using their mouse. The simulator showcases the essence of sailing ship operation, allowing users to trim, tack, jibe, sail with the wind, and beat against it. The entire ship is constructed from molecules connected by "springs" (soft body dynamics).

Parys Gym Website

MANTINE TYPESCRIPT NEXT REACT STORYBOOK

Promotional website for Parys Gym in Nysa.

Skills 💪

Programming Languages 💻

TYPESCRIPT C++ JAVA C# PYTHON PHP

- **TypeScript** - Currently, my favorite programming language. I value the ease of running applications on various devices (just a web browser or Node). I avoid JavaScript due to the lack of typing. I aim to write code in a functional style, using constants, as it's easier to understand (no side effects in functions).
- **C/C++** - The first language I learned. I liked it until I discovered how convenient it is to work with functions in TS/JS (first-class citizens). C++ provides more control (no garbage collector).

- **Java** - The second language I learned, and I developed my object-oriented programming concepts with it.
- **C#** - Similar to Java.
- **Python** - I used it for Raspberry Pi programming. It's similar to JS (lacks typing, first-class functions), but I didn't like the use of tabulation instead of curly braces.
- **PHP** - Not my favorite.

Frameworks/Libraries 📚

REACT NEXT JEST

- **React** - After learning React, I fell in love with creating user interfaces.
- **Next.js** - A convenient framework/tool for building TS+React applications.
- **Jest** - I try to cover as much code as possible with tests.

Tools 🛠️

GIT VISUAL STUDIO CODE LINUX MINT

- **Git** - I use it for version control of projects I work on independently. For each new feature, I create a new branch and merge it into the main branch upon completion. I try to name commits according to `conventional commits`.
- **Visual Studio Code** - It's my favorite IDE. I appreciate its simplicity and versatility (a vast number of extensions).
- **Linux** - An open-source operating system that provides greater control over hardware.

Natural Languages 🌐

- **English** - B2.
- **Polish** - Native.

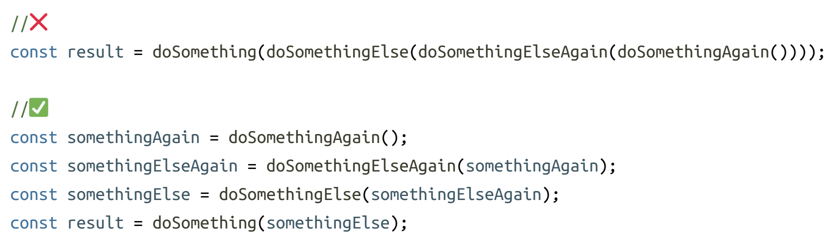
Coding Style 📝

Naming

I write long names, sometimes consisting of several words, and avoid abbreviations. For example: `CollidingTriangle`, `FluidInteraction`.

Code Readability

While writing code, I use many intermediate constants instead of invoking functions within functions. If I can name a piece of code (what it does), I extract it into a function. I aim not to comment the code, as the names defined in the code (constants, functions, interfaces, etc.) should be self-explanatory.



```
//❌
const result = doSomething(doSomethingElse(doSomethingElseAgain(doSomethingAgain())));

//✅
const somethingAgain = doSomethingAgain();
const somethingElseAgain = doSomethingElseAgain(somethingAgain);
const somethingElse = doSomethingElse(somethingElseAgain);
const result = doSomething(somethingElse);
```

Functional Programming

I try to write code in a functional manner, meaning I don't use mutable data (let var) but prefer constants (const). This assumption makes using language constructs with a separate namespace block (e.g., for, if, switch case) pointless because any name defined there is inaccessible outside that block. Moreover, functions always return a value, or they don't make sense.