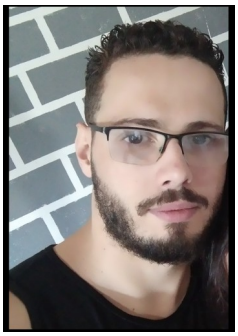
Deivson Ferreira



I am a full stack developer that have developed different projects like trading robots, games, and custom libraries, using a variety of programming languages, like imperative, object oriented, high level and low level languages.

I have a good knowledge in software optimization, memory management, project design and I know how to work with 3d rendering, file parsing, image manipulation, string manipulation, physics detection, Sql Data Bases, Server side development, web pages and algorithms like flood fill, A-star(path-finding) and DDA (Digital Differential Analyzer).

PROGRAMMING LANGUAGES

C#(Intermediate-Advanced) | **JAVA**(Intermediate) | **MQL5 (**Advanced**)**
JAVASCRIPT(Intermediate) | **C(**Beginner-Intermediate**)** | **SQL(**Intermediate**) |**
Vulkan API (Beginner) | **Batch**(Intermediate)

PROGRAMMING CONCEPTS

Programming Logic(Advanced**) | Project design (**Intermediate-Advanced**) |**
OOP (Advanced**) | MVC (**Intermediate-Advanced**) | Libraries development (**Advanced**) |**
Unit test(Advanced**)**

LANGUAGE SKILLS

Portuguese (Brazil): Mother Tongue:
English: Listening C1 | Reading C1 | Spoken production B2 | Spoken interaction B2 | Writing B1
Italian: Listening A2 | Reading A1 | Spoken production A1 | Spoken interaction A1 | Writing A1

WORK EXPERIENCE

25/02/2019 – 28/08/2022
I worked as a freelancer for almost four years, creating the design and implementation of Expert advisors (trading bots) and custom indicators in C# and MQL5 language (Object-oriented language based on C++) for the software MetaTrader5 (MetaQuotes Company), Developing personalized trading strategies, automated investment systems, and custom libraries to increase the customers trading speed of financial assets as cryptocurrencies, commodities, and currencies pairs on the international stock market.

07/02/2009 –5/12/2018
Sales, computer maintenance, and customer service on a retail store.

To know about my projects, read the second page.

CONTACT INFORMATION

Email Contact: devdcj@outlook.com | **Linkedin:** www.linkedin.com/in/deivson-ferreira-2b3723260

I'm currently living in Brazil, but I'm available to relocate immediately.

Some Projects I have Worked, or still in development

Trading Robots:

When developing trading bots my tasks are, listen the trading strategy from the customer, and later analyzing how to automate it (and if is possible to automate it or not) , after that , creating the bot design and logic that will execute the trades, implementation of new indicators or conversion of indicators, implementing all validations for the the market data the server have send, validating the robot data to send to the server and making sure the trades are accepted, validating if the user only inserted correct data in the bot configuration and showing to hin any inconsistency, runtime tests in different scenarios to find bugs or inconsistency in the the implemented trading logic and the customer trading strategy, giving users support for some time after the delivery of the project, and analyze the consistency and portability of the bot in different broker servers (the data between brokers can be different).

My trading bots are usually develop in C# (or C# based languages) and MQL5/MQL4, but sometimes are developed in scripting language specific for the platform the user is trading.

To make my work of porting code easy, I developed a custom library in C# avoiding the use of the platform standard library, using very basics types and OOP concept, this allowed me to port the code to different programming language without problems.

Trading bot example:
VROC and Moving Averages Channel

One of the many trading bots I have created using C#, later converted to MQL5, this one in particular uses two moving averages, a moving average channel, vroc indicator and some internal processing, like risk, price distance, account balance, and more, to make the tradings decisions.

This trading robot works checking if the vroc from the last candle (bar) has crossed its own moving average, if true, it checks if the last price bar have closed beyond its moving average channel, if true, it executes a market trade in the direction of the channel. After a trade, the price have to return to the channel to allow new trades. It have a custom configurations so the user can invert the direction of the trades, decide the value he wants to trade, the amount he are willing to risk, days allowed to trade, acceptable loss and more. In one variant of this bot I was asked to develop and add some custom indicators.

I will not give more than one example of trading bot because the descriptions would be almost the same.

Video: https://youtu.be/mu60fXwa-hU

Procedural Voxel Terrain Generation
(24/11/2022 – Being Developed)

A procedural voxel terrain generation in development using the C (not c++) programming language and the vulkan API. To create the terrain I'm using a noise image as height map, and applying to it some custom math calculations to define the output in the voxels, the math parameters can be used to change the height, propagation, and the shapes of the terrain. to load images to be used as textures, I'm using the STB image library, to create the camera movement and matrix Uniforms I'm using the CGLM library, all memory allocations, entities persistence and RAM to GPU streams are developed without using libraries for learning purpose. It can render simple tile based strings (Letters and Numbers) that can be edited in runtime. In the current state it can load OBJ files and spawns different mesh shapes like cubes, diamonds, and hexagons.

Video1: https://youtu.be/_LL6uUi6kEo | Video2: https://youtu.be/BEZBgtXf_hM

Custom Game Engine
(25/03/2020 – Being Developed)

A game engine I'm creating in java-script. I'm avoiding to use language specific features to make easier to port the code to another programming language, (some code was ported from a canceled project I was developing in the Unity game engine in C#), for now I have finished some features like path finding (A-star), flood fill, procedural terrain generation, basic ai system, inventory , factions, simple physics collision, tile drawing, animation drawing and camera occlusion system. Almost all the features are designed in modules, that can be placed together to create different games using the same systems with no modification to the core of the module and making easy the version control.

Video: https://youtu.be/pEs0yGonj5U