# Chen Tianze


| Phone | E-Mail | GitHub | Website |
| --- | --- | --- | --- |
| (+86) 175 5416 0152 | chuigda@doki7.club | chuigda | CoreBlogN²G |


## Education

**Qingdao University**, Computer Science, Bachelor 2016.9 - 2020.6

## Skills

- **Programming Languages:** Multilingual (not limited to any specific language), especially experienced in Rust, C, C++, JavaScript. Comfortable with Java, C#, Go, Python, PHP, Julia, Racket (in random order).
- **Computer Graphics:** Understand how rasterization pipeline works, have experience with OpenGL/OpenGL ES, WebGL and Vulkan. Have some research on game engines.
- **Frontend:** Have the experience of using React and Vue, also capable of using vanilla JavaScript.
- **Backend:** Understand HTTP protocol and semantics, relatively familiar with Express.js, and have experience with SpringBoot. Have basic database knowledge, used MySQL and MongoDB.
- **Compilers:** Familiar with parsers, semantic analysis and type checking, participated in several related company and personal projects.
- **Chess:** Lichess 1505 (June 12, 2024)

## Working Experience

**Suzhou Tongyuan Software&Control Co.,Ltd**, Suzhou, China 2023.3 - now

*Compiler engineer*

- Implemented normalisation of Julia IR (intermediate representation) used by Julia static compiler.
- Designed and implemented parser of MATLAB(R) language for our own MATLAB(R) implementation, supporting many dubious syntaxes of MATLAB(R) language and of its dialects. The new parser outperforms the previous ANTLR-based one by 50 times.
- Implemented a networking library for Julia, providing HTTP sever/client and WebSocket server/client. Replacing HTTP.jl and other previous workarounds.
- Implemented a web-based user-interface for wrapping Julia functions to MATLAB(R) functions.
- Implemented VSCode workspace and debugger for our own MATLAB(R) implementation.

**Qingdao Eastsoft Co.,Ltd**, Qingdao, China 2021.4 - 2022.7

*Fullstack engineer*

- Participated in the development of a cloud platform for power IoT, which uses SpringBoot + MySQL + MongoDB + JPA/MyBatis for backend and Vue2 + iView + eCharts for frontend
- Developed a PDF report generator service based on Express.js + node-canvas + eCharts + LaTeX, where images are generated using node-canvas + eCharts

**Shanghai Tianxin Technology Co.,Ltd**, Shanghai, China 2020.6 - 2021.3

*Fullstack engineer*

- Designed and implemented a protocol for uploading videos and audios from low-bandwidth embedded devices to the cloud.
- Implemented an audio and video encoder based on FFmpeg, which encodes the audio and video files uploaded by IoT devices into formats that can be played directly by mainstream browsers and video players (MP4 container + H264 video stream + YUV420P frame + AAC audio).
- Implemented the connection between the encoding server and the Node.js business server through RPC over HTTP.
- Helped solving several problems with the OV5640 camera on IoT devices.
- Implemented a utility for testing WebRTC servers, which uses the same technology as the selected alternative for WebRTC clients.

**Sourcebrella Inc.**, Shenzhen, China 2018.7 - 2018.9

*Compiler engineer*

- Investigated bugs of clang-3.6, fixed them by writing patches and porting patches from higher versions of clang. Made eosio, Firefox, Android and some other projects compile with clang-3.6.
- Introduced a WebAssembly backend (skeleton only) to llvm-3.6, making it capable of generating LLVM IR targeting WebAssembly.

## Personal projects

**2V64**, Rust asynchronous runtime https://github.com/chuigda/2V64

*Rust, Unix, async*

- Simple single-thread/multi-thread Rust asyncronous runtime.
- Implemented TCP socket based on Unix `poll` API (two ways: hand-written `Future` and using tokio `AsyncRead`/`AsyncWrite`), and implemented a small HTTP server based on this.

**9T56**, Type theory research https://github.com/chuigda/9T56

*ML, Python, Typst, Type theory*

- Translated paper Basic Polymorphic Typechecking and blog Understanding Algorithm W
- Implemented algorithm $\mathcal{W}$ in the paper *Understanding Algorithm W* and algorithm $\mathcal{J}$ in *Basic Polymorphic Typechecking* using Python.
- Designed additional language features (mainly imperative features) and their type checking rules, which are not included in standard ML.

**Project-WGX**, 3D VTuber https://github.com/chuigda/Project-WGX

*Java, Vulkan, OpenGL, CG*

- Designed the extensible `reactor` system based on the characteristics of `MonoBehavior` and ECS, according to actual needs.
- Implemented the renderer based on Vulkan and OpenGL ES2, supporting multi-threading: when using Vulkan as the backend, it can utilize Vulkan's multi-threading and asynchronous features; when using OpenGL ES2 as the backend, it encapsulates operations performed on the rendering thread as synchronous operations through channels.

**Project-602**, Chess software https://github.com/chuigda/Project-602

*JavaScript, HTML5, WebGL, CG*

- Designed and implemented user interface with Vanilla JavaScript.
- Designed and implemented highly stylized 3D chessboard rendering based on WebGL.
- Implemented games against AI with fairy-stockfish engine through WebAssembly.
- Designed and implemented a highly extensible script system based on JavaScript, and developed a visual editor based on Blockly, which can achieve interactive teaching functions similar to game BOT.vinnik or software PyChess.

**vulkan4j**, Java CG API bindings based on FFM https://vulkan4j.doki7.club

*Java, Kotlin, Vulkan, OpenGL, OpenAL, CG*

- Designed and implemented extraction of function and type definitions from Vulkan registry `vk.xml`/`video.xml`, OpenGL registry `gl.xml`, Vulkan memory allocator header file `vma.h`, GLFW header file `glfw3.h`/`glfw3native.h` and OpenAL header files `al.h`/`alc.h`/`alext.h`/`efx.h` using automatic scripts. Java bindings are generated from these extracted metadata.
- Provided moderate Java abstractions to make these APIs easier to use and more type-safe.
- Ported full Vulkan tutorial.

**Project-WG**, 3D VTuber https://github.com/chuigda/Project-WG

*C++, Qt, OpenGL, CG*

- Implemented renderer with C++, Qt and OpenGL (traditional fixed pipeline).
- Implemented face tracking data recever: via UDP from OpenSeeFace or WebSocket from VTubeStudio.
- Implemeted extensible plugin system, supporting plugins written in C++.

**Project-PL5**, Scheme dialect https://github.com/chuigda/Project-PL5

*C, Compiler*

- Imperative first Scheme dialect with special built-in function mechanism.
- Implemented error handling scheme based on C `setjmp` and `longjump`, which can print stack trace and recover from errors.
- Can be extended through C API, and also supports Rust.

## Miscellaneous

- Contributed to opensource projects **flomonster/easy-gltf** (*Rust*), **franciscoBSalgueiro/en-croissant** (*JavaScript + Rust*), **durch/rust-s3** (*Rust*), **webrtc-rs/webrtc** (*Rust*), **KyleMayes/vulkanalia** (*Rust*) and **open-webrtc-toolkit/owt-client-native** (*C++*).
- Attended Rust China Conference 2020 as a lecturer.
- In 2019 Tianjin Lanqiao training, the Hahadoop OJ (online judging system, supporting distributed judging machines) developed by me won the excellent training project award.
- Got 310/500 in CCF-CSP in 2018
- Got the second prize in the provincial competition of Lanqiao Cup in 2017