Henry **Zhangxiao**

BACHELOR OF COMPUTER SCIENCE · HONOURS · CARLETON UNIVERSITY

(+1) 613-608-7348 | ✉ henry_zhangxiao@hotmail.com | 🏠 www.henryzhangxiao.com | HenryZhangxiao | henryzhangxiao

Skills

Languages	**Python**, **C**, **C++**, **Java**, **Makefile**, **Shell**, JavaScript, TypeScript, HTML/CSS
Technologies	**Linux/RHEL/CentOS**, **Git**, **OpenShift**, **Kubernetes**, **Private Cloud**, QNX, Jenkins, Docker/Podman, React, GDB, GCC, RPi

Experience

IBM   *Markham, ON*

SYSTEM EVALUATION AND RELIABILITY TEST (SERT)   *May 2022 - Aug 2023*

- Deployed and **regression tested CD**, LTSR, and **Future** release candidate builds using automated **Jenkins CI/CD pipelines** on **AWS**, **Azure**, and **private cloud Fyre clusters**
- Created multiple **Jenkins pipelines** to test **release candidate builds** by enhancing, upgrading, and testing our deployment **Makefiles** and **Bash scripts**
- Maintained a managerial dashboard built in **Django** with over **10,000** GitHub issues and synchronized using **crontab**
- Maintained and upgraded an automation results dashboard built in **JavaScript** and **React**
- Integrated an **automated AWS** resource cleanup script leveraging a **containerized Docker environment**

IBM   *New York, NY*

IBM SAAS PLATFORM • MULTI CLOUD SAAS PLATFORM - SITE RELIABILITY ENGINEERING   *Jul 2023 - Aug 2023*

- **Automated** the **deployment** and **provisioning** of **Prometheus** to provide monitoring and alerting functionality on OpenShift 4.12
- Defined and upgraded existing clusters using **Helm** and **Helmfile**
- Defined and rendered **.gotmpl** template files for Prometheus, AlertManager, and Thanos

IBM   *Austin, TX*

CLOUD PAK CONFIGURATOR • IBM USER EXPERIENCE DESIGN TEAM - CROSS PAK CONSISTENCY • RED HAT   *Jul 2022 - Sep 2022*

- **Led development** of a **Dynamic Plugins** pilot project using OpenShift Container Platform 4.11
- Created a mockup of the UI using design prototypes from **Figma** and **InVision**
- Developed using **TypeScript**, **Patternfly**, and **Podman**
- Deployed locally using **OKD** and online using **Fyre clusters**
- Built and pushed to **Quay.io** image registry using **Docker**

Projects

QNX Car Simulator   *Carleton University*

HTTPS://GITHUB.COM/HENRYZHANGXIAO/QNX-CAR-SIMULATOR   *Dec 2022*

- A **real-time** car simulator written in **C** using **QNX Neutrino** utilizing **QNX SDP 7.1**
- Hosted locally using a server created with **name_attach** that receives and delivers **messages** and **pulses**
- Implemented modularly using the concept of **parent-child processes** and **threads** to satisfy **microkernel** properties

Nintendo Switch Autoclicker   *Personal Project*

HTTPS://GITHUB.COM/HENRYZHANGXIAO/NINTENDO-SWITCH-AUTOCLICKER   *Jun 2023*

- An **automated** button clicker for the Nintendo Switch using a **9g Micro Servo** and a **Raspberry Pi**
- Executed with **Python3** using the **PiGPIO library** and ran on **Raspberry Pi OS** (Debian)
- Servo controlled using **Pulse Width Modulation (PWM)**

Carleton Library Autobooker   *Personal Project*

HTTPS://GITHUB.COM/HENRYZHANGXIAO/CARLETONLIBRARYAUTOBOOKING   *Aug 2023*

- A **headless automated script** to book Carleton University library study rooms
- Developed in **Python** using **Selenium WebDriver** and **ChromeDriver**
- Support for Windows, Linux, and Darwin

Quests of the Round Table   *Carleton University*

HTTPS://GITHUB.COM/HENRYZHANGXIAO/QUESTS-OF-THE-ROUND-TABLE   *Apr 2022*

- A multiplayer card game coded in **Java** with the use of **Java Networking** and **JavaFX**
- Developed using the **Gang of Four** design patterns for regulation of **code maintenance**, **code abstraction**, and **object interaction**
- Used **Apache Maven** and **Git** to ensure version control while working in an **agile** and **model-driven environment**

Yume   *Carleton University*

HTTPS://GITHUB.COM/HENRYZHANGXIAO/YUME   *Apr 2022*

- A **2D real-time** game written in **C++** using **OpenGL**, **OpenAL**, **SOIL**, **ALUT**, **GLEW**, **GLFW**, and the **GLM** libraries
- **Physical**, **parametric**, and **hierarchical movement** all handled through **matrix transformations**
- Graphics are drawn using **vertex shaders** and **fragment shaders** using **GLSL**