



# ここ数年の個人開発史 (2024)

文系エンジニアのハードへの挑戦  
ORION]エンジニアLT会&交流会 #2

2024/11/28  
Ken Shimizu



## はじめに

本日のLTは、私のここ数年の個人開発を通して、文系エンジニアでハードについての知識0からIOT(電子工作)を始めて感じたことなどを、成果物を通して伝えたいとおもいます。

決してIOT(電子工作)はWebエンジニアにとってもハードルの高いものではなく、今作っているアプリケーションの延長上にあって、一歩踏み出すだけです。

まずは動かすことから始めることをおすすめします。

# 自己紹介

職業:証券会社でエクセルとVBAで帳簿への自動記録や、便利なツール類を作っています。

趣味:カヤックフィッシングと個人開発



# Web アプリを作っていた頃

こんな感じで転職のために自宅にPCサーバーを立ち上げてインターネットに自由につながる環境で個人開発を始めました。



- Home
- Products
  - Option Quote
  - Option Calculator
  - Option Strategy
  - Macro Date Input
- Business
  - Greeks
  - Delta Hedge
  - Gamma Hedge
- Technologies
  - becho\_server.c
  - opengamma.html
  - svn.new.project
  - Calendar Panel
  - svn
  - Raspberry Pi X ESP32
- Cam
  - Face Detect
  - Object Detect
  - Object Face Detect
- Hobby
  - Fishing
  - Sake
  - Robot1
  - Robot2

This site introduces you a model of application to demonstrate new technologies, C++, Java, Javascript, Perl, Python, etc. The demonstration means not only just introducing new technologies but also how we organize project and development process with new technologies.

## [Client Broker Option Quoting System](#)

**Key Technology:** [React.js](#), [Redux](#), [Redux Thunk middleware](#), [MariaDB](#), [libwebsockets](#), [QuantLib](#), [RapidJSON](#)

A screenshot of a web application displaying a table with multiple columns and rows of data, likely representing financial or market information.

This sample product introduces a model of high speed Internet application for financial industry through developing a prototype of server-client application over the Internet. The client is single page application (SPA) with React.js and the server is developed from scratch based on C++. The server-client communication is WebSocket. Web Socket and C++ application enables to handle high volume transaction over the Internet.

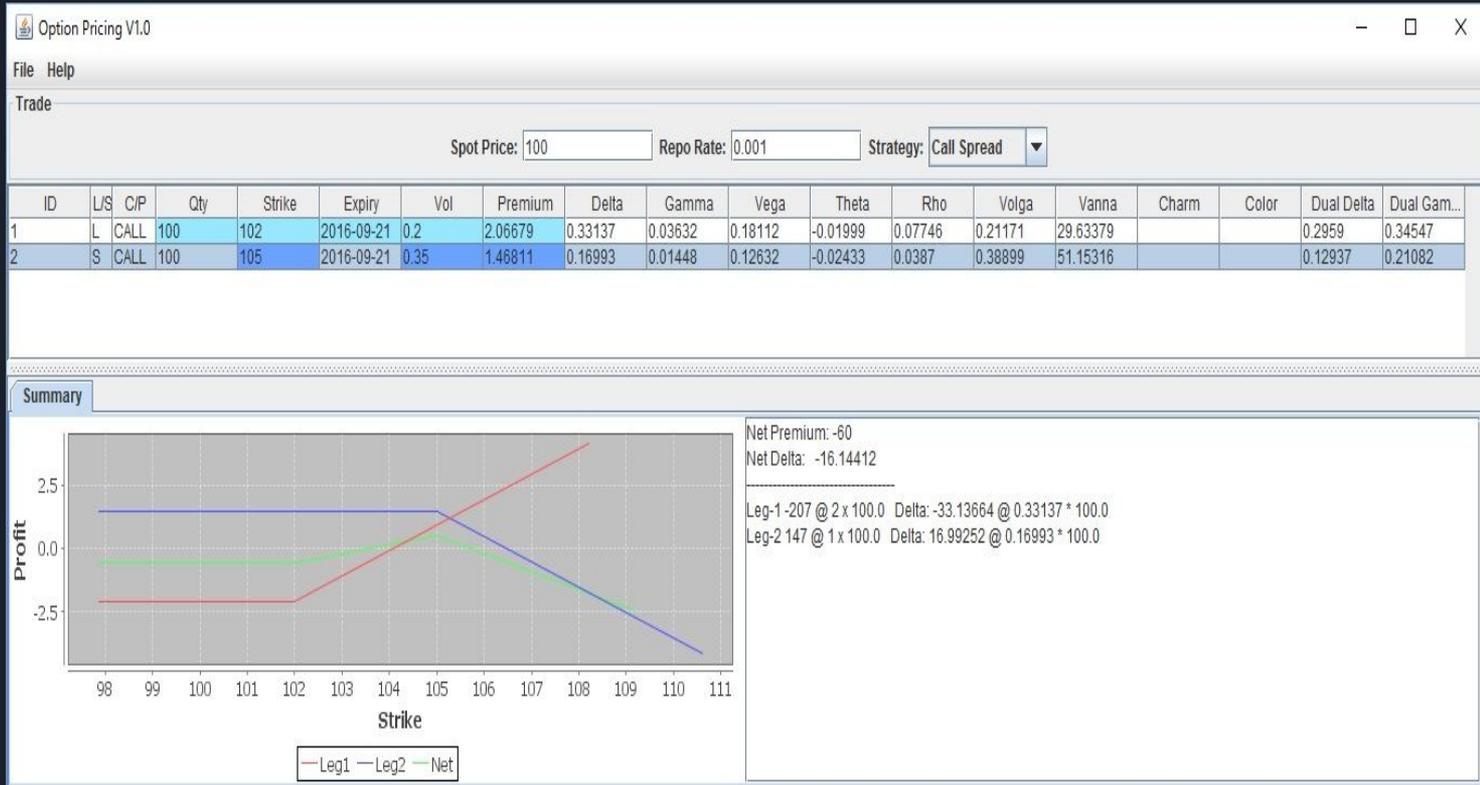
## [Option Calculator](#)

**Key Technology:** [React.js](#), [Redux](#), [libwebsockets](#), [QuantLib](#), [RapidJSON](#)

A screenshot of a web application displaying a table with multiple columns and rows of data, likely representing financial or market information.

This sample product introduces a very simple model of high speed Internet application for financial industry through developing a prototype of server-client application over the Internet. The client is single page application (SPA) with React.js and the server is developed from scratch based on C++ and integrated with QuantLib which is Quant library. The server-client communication is WebSocket. Web Socket and C++ application enables to handle high volume transaction over the Internet.

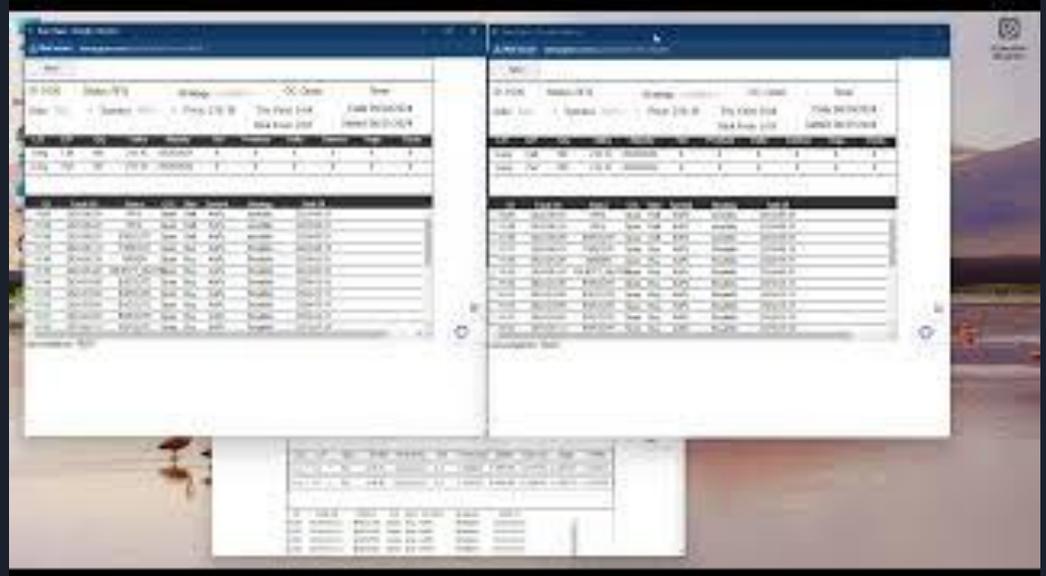
# Java SwingでRMI、RESTを使ったアプリケーション



# クライアントはReactJSで、サーバーサイドはC++でTest Driven開発

クライアントサーバー間は  
WebSocketで、メッセージは  
JSONを使っています。

サーバーサイドは、C++を使って、  
Test Drivenで品質をあげることが  
できるか確認してみたかった。  
C++を使った理由は、  
TCP/IP、Thread、プロセスを直接  
使ってみたかったからです。



# Raspberry Piなるシングルボードコンピュータと出会う

Linuxが動いて、GPIO(何でも簡単にいろいろなものを繋げることができるインターフェース)がついているので、モーターを動かしたり、スイッチを入れたり、電気をつけたりできる。その上Linuxだから、Djangoも動くからWebアプリケーションで何でも動かせるじゃん。



# Raspberry PIと出会い

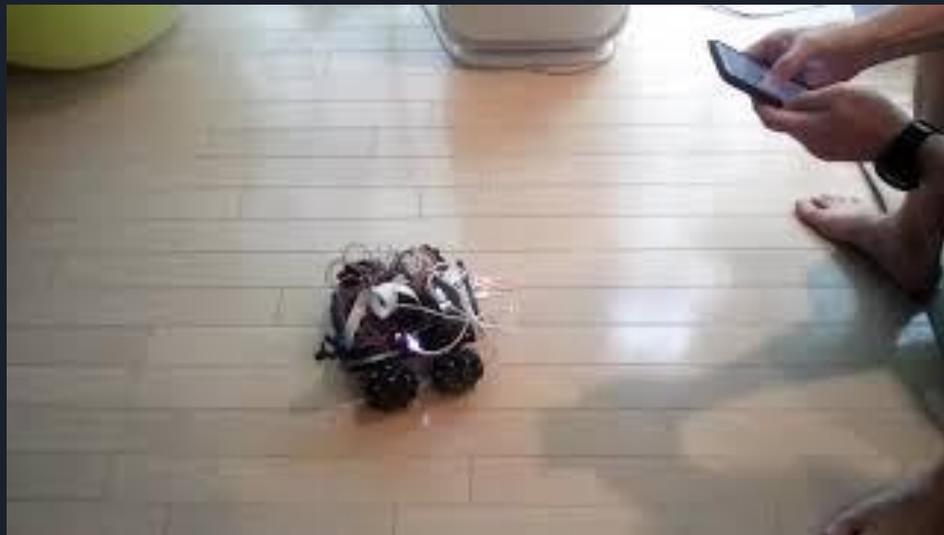
Raspberry PIと出会い、GPIOでサーボモーターやモータードライブを動かすことができることを知って、こんなもの作ってみました。

コントローラーは、Raspberry PI側でApacheを動かしてWiFiを通じて、ReactJSを使ってコントローラーを作ってみた。



## Raspberry PIと出会い(2)

メカナムホイールと  
Raspberry PIで動かして  
みた。

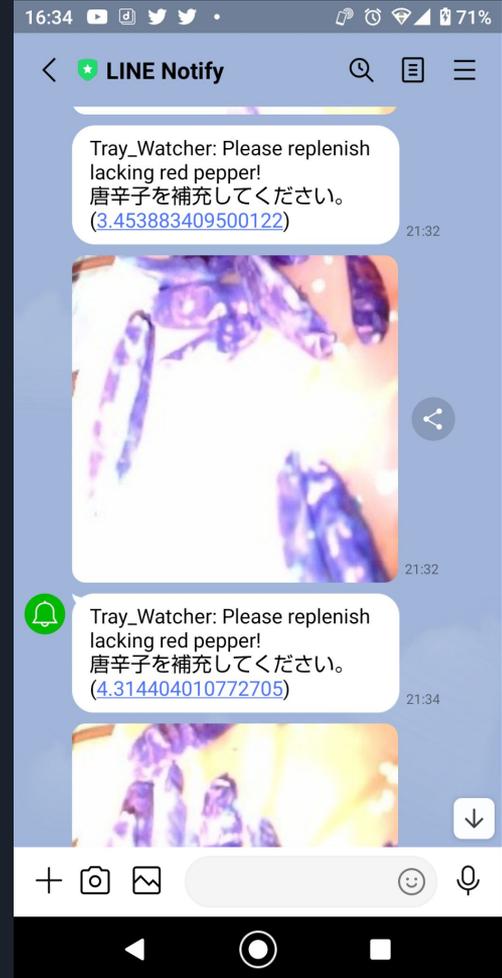


# Raspberry PIでAIを動かしてみた

Open CVのObject Trackingを使って、カメラに写った物体をマウスで指定して、それをカメラに追いかけてみた。



# ESP32-CAM and Raspberry Pi 4B

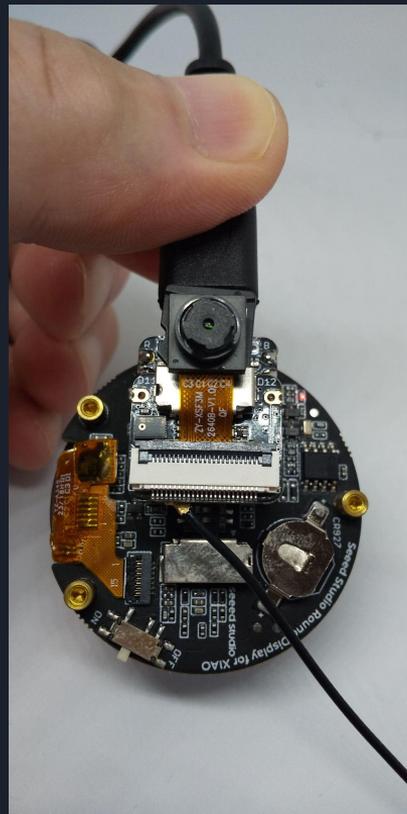


# ESP32でAI動かしてみた

マイコンでもTensorflowが動くことを知って、とりあえずESP32でPerson Detectionを動かしてみたら、遅いけど動いたじゃん！

ちょっと感動して、マイコンとAIの可能性をもっと知りたくなる。

ちなみにこれは、スマートウォッチの開発キット。



# Edge ImpulseとESP32でObject Detection

Edge Impulseを  
使って、ねじとナット  
を認識する  
Object Detection  
モデルを作って試  
してみた。

パフォーマンスは  
9FPSぐらい出ているので、かなり速い  
です。



# Edge ImpulseとSpresenseでObject Detection

Edge ImpulseとSpresenseでObject DetectionをESP32の時と同じモデルを動かしてみると、推論に1.1秒かかっている。

調べると、使っていたESP32-S3ではMCUにTensilicaを使っていて、NPUがSoCでついていたのであった。



# こんなことができるようになるもくもく会があります

月に1度のペースで、ゆるう〜く開催しています。電子工作の経験がなくても、手ぶらでも大丈夫です。

興味がある、作りたいものがある、やりたいことがある、知りたいことがある、声をかけてください。

