30分で
読める!

RCC自作会誌

ritsumeikan computer club

RCC 著

まえがき

　冊子を手にとってくださり、誠にありがとうございます。
会誌編集担当の3^5です。さごじょーと読みます。
立命館コンピュータクラブ(以下RCC)は、プログラミング、DTM、デジタルイラスト製作や電子工作など、コンピュータに関わる分野を対象として学習・研究活動を行っているサークルです。
　この会誌はRCCの主な活動の紹介や近況、また活動の中でもメインであるプロジェクト活動の成果報告、そして会員の個人活動を紹介するコラムを掲載しています。プロジェクト活動のページでは、今年度の前期に活動した内容を紹介しています。コラムのページではWeb系や機械学習など最近流行りのプログラミング技術だけではなく、レトロPCや数学系、デザイン系の話まで掲載しており、RCCの多様性を感じていただけると思います。
　今年度のRCCは新入会員こそ昨年度より少しだけ少なかったものの、1回生の殆どが部室に頻繁にきて活動してくれていて、定例会議前などになると部室が狭く感じるほど賑やかになりました。個人コラムも例年より会員が積極的に提出してくれており、RCCの活動がますます盛り上がっているのを感じます。これだけ読み応えのある文章を書いてくれた会員の皆様には編集担当者として本当に感謝しています。例会やTwitterで厳しく提出を煽ってごめんなさい。
　さて、会員の活動の結晶とも言えるこの会誌、とてもタイトルのように30分で読み終えられる量ではないかもしれませんが、笑いながら最後まで読んでいただけると編集者としてこれほど嬉しいことはありません。どうぞ、ごゆっくりお楽しみください。

2019年10月

会誌編集担当：西見元希

目次

1章

イントロダクション

1.1　立命館コンピュータクラブとは

　立命館コンピュータクラブ(Ritsumeikan Computer Club、 以下RCC)は、立命館大学びわこ・くさつキャンパスを拠点とした学術系サークルです。

1.2　活動内容

　RCCは主に2点の活動を軸に行っています。

- 研究活動
  情報に関する分野について、自由にテーマを決めて調査・議論を行い、情報技術と社会についての見識を深めていきます.
  研究成果は報告書という形でまとめ、Webサイト上での公開や、本会誌にまとめて配布をしています.
- 制作活動
  ソフトウェア、ハードウェア、映像、音楽などの制作を行います。情報技術を使うものであれば自由に制作できます。会内では、Welcomeゼミ、ハッカソン、夏季制作など制作活動を体験できるイベントを設けています。

1.3　イベント

　毎年9月頃に、関西情報系学生団体交流会(以下、KC3) を開催しています。これは、関西圏にある大学のコンピュータ関連団体を対象として、お互いの活動を紹介しあったり、勉強会を開催して新たな技術を学んだりしながら、他の団体の方々と交流するイベントです。今年度の開催で開催10周年を迎え、参加団体は13団体、参加人数は131名という大規模なものになっています。(Website:kc3.me)

　また、昨年度は2月に冬KC3(仮)という名前でハッカソンイベントも開催し、非常に盛り上がりました。今年度も冬に何らかのKC3関連イベントを開催する予定になっています。

1.4　普段の活動

　普段は、週一回の定例会議があり、活動予定などの連絡が行われます。この時にライトニングトーク(LT)という、情報系の内容をテーマとして自由にプレゼンテーションを行う時間もあります。また、全体向けに勉強会が開催されることもあります。それ以外で特に予定のない時間は、自由にサークルルームに来て自分の作業をしたり開発をしたりしている会員もいます。

1.5　活動拠点(サークルルーム)

　サークルルームには、様々な備品が保管されています。本棚には研究活動で使用する資料や、制作活動で参考になる技術書が数多く保管されています。

1.6　入会について

　新規入会は、通例では年度初めの5月から受付を開始しています。学年を問わず、本学の学生ならどの回生からでも入会することができます。手続きとしては、サークルルームに来ていただき、会費の徴収と入会届の記入をするだけで完了します。

　会費は年間6000円です。会の重要な資金源となっていて、日々の活動や情報のシステム維持、会員専用PCの購入費などに利用されています。

1.7　アクセス

　サークルルームは、BKCバイオリンク一階のサークルルーム8です。食堂や書店などがあるリンクスクエアの裏側の建物の1階に位置しています。サークル宛に質問や連絡等がございましたら、以下の連絡先までお願いします。

---- RCC Information ----

Address:〒525-0058
滋賀県草津市路地東1丁目1-1
バイオリンク1Fサークルルーム8

Email:rcc.liaison@gmail.com
Twitter:@rits_cc
Website:www.rcc.ritsumei.ac.jp

バイオリンクの入り口

1.8　2019年度 年間行事予定

月	行事	内容
4月	Welcomeゼミ	新入生と上回生が共同で勉強・製作
5月	前期活動開始	前期プロジェクト活動のスタート
8月	追い込み合宿	前期活動の制作物・報告書の仕上げ
8月	プロジェクト 成果発表会	前期活動の成果(制作物・研究)を発表
8月	ハッカソン	2泊3日の開発イベント
9月	KC3	関西情報系学生団体交流会
9月	RCC総会	RCC会内の前期総会
10月	後期活動開始	後期プロジェクト活動のスタート
12月	学園祭	制作物の展示・会誌頒布
12月	クリスマス会	プレゼント交換など楽しいイベント
2月	追い込み合宿	後期活動の制作物・報告書の仕上げ
2月	プロジェクト 成果発表会	後期活動の成果(制作物・研究)を発表
2月	RCC総会	RCC会内の後期総会
2月	ハッカソン	2泊3日の開発イベント
3月	追い出し会	卒業する4回生と過ごせる最後のイベント

1.9　ロゴ&Webサイトリニューアル話

RCCのロゴが新しくなりました。

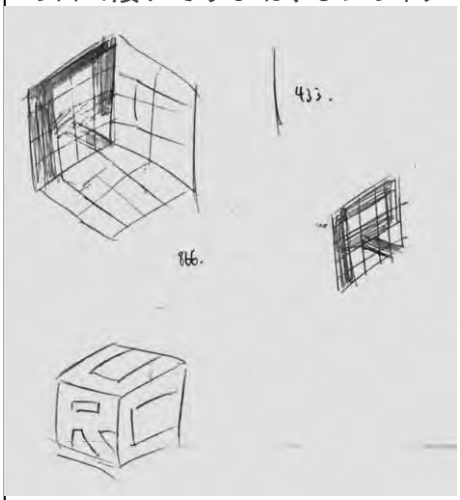
地味でしょうか？退屈だと思いましたか？
いいえ、ロゴに冒険は必要ありません。すべてがここに詰まっています。
シンプルなデザインは、見た人すべてにRCCという文字を認識させます。
三文字で組み立てられた立方体は、RCCのものづくりの独創性と創造性を表現。
時代に流されない無難なカラーリングは、どんな場面でも違和感を生み出しません。
背景が何色であれ、ただ自然に溶け込みます。
まるで最初からこのロゴだったかのように。

新しいRCCロゴ、これは堅実なアップグレードです。

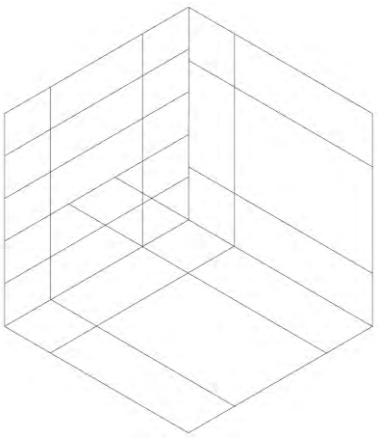
ロゴ制作について

　このロゴは僕がコンペ締め切り3日前の夜中3時にNetflixでクレイジー・リッチ！を観ながら作りました。上のApple風の文章はその映画を観終わった4時頃に深夜テンションで書きました。良い映画でした。金持ちになりたい。
　RCCの3文字って本当にロゴにしづらくて、苦悩した結果がこの立方体の形です。ラフを考えている時は下にあるように別バージョンの立方体も考えていましたが、ゲームキューブのロゴに似すぎているので不採用にしました。立方体がものづくりの独創性や創造性を表してるっていうのは後付けです。配色は悩んだ結果、ロゴが今後長期に使用されるであろうことを考慮して立命館大学に合うカラーリングにしました。

　最後に、全然関係ないですけどこの前公開されたパリオリンピック/パラリンピック2024のロゴ凄いですよね、3つのイメージを1つで表現するあの発想に憧れます。



↑授業中にノートに描いたラフ
(数字は三角関数の計算の跡)



↑ちゃんとIllustratorでちゃんと三角関数
を使って製作したちゃんとした枠組み

↑旧RCCロゴ
(誰が作ったんだろう…)

担当者：taken

Webサイトリニューアル

　こんにちは、てんちょです。この度、新しいロゴに合わせ、RCCのWebサイトを完全リニューアルしました。

“19.09.26 ALL RE-NEWAL!!”として宣伝しました

注目ポイント

・スマートフォンに対応！

・アイキャッチや画像を使用し見やすく！

・SNSで「RCC」と分かりやすく！

Twitterで共有した時の画像

　その他にも、パンくずリストや、関連記事・SNSの共有機能についても見直しを行い、非常に見やすく・わかりやすくしました。

パンくずリストやSNS共有ボタンの例

　面白い記事や興味があるページがあれば、みなさん是非TwitterやLINEなどで多くの人に共有していただければ、嬉しいです。

　また、メタ的な話にはなりますが、SEO対策や記事のカテゴリー構成なども見直しており、カテゴリごとにカラーリングを変えたりと、現状のRCCを反映できたのかなと担当者は思っています。

　このリニューアルを期に、コンテンツの充実や更新の頻度が上がっていくように、渉外局さんには頑張っていただきたいと考えています。

Welcome to RCC！

面白いRCCを是非見つけてくださいね。

担当者：3回生のてんちょ

2章

プロジェクト活動

プロジェクト活動とは、前期と後期の2回に分けて行われるRCCのメイン活動です。情報科学の研究をし、その成果の発表を活動の基本として会員間で相互の親睦を図るとともに学術分科の想像と発展に寄与することを目的とします。

今年度前期では7つのプロジェクトが立案され、ここではその活動の成果を報告します。

2.1　イラスト班

ペイントソフト「CLIP STUDIO PAINT」の機能や、どのようにイラストを描いていけばよいかを調査し、活動内で共有しました。

2.2　競技プログラミング班

与えられた問題を速く正確に解く「競技プログラミング」における重要なアルゴリズムを学習し、実際に過去問を解いて演習しました。

2.3　LT班

LT(ライトニングトーク)において伝えたい内容を効果的に伝えるデザインを実際にLTを作成しながら学習しました。

2.4　DTM班

学園祭に向けて自作した楽曲を発表することを目標とし、作曲の知識を深め、実際に楽曲を作成することで各個人の技術力向上に努めました。

2.5　自作キーボード班

キーボードの歴史や仕組みを学び、最終的に一人一個自作することを目標として活動しました。

2.6　メタプログラミング班

メタプログラミングとその活用法について研究しました。

2.7　プログラム意味論班

プログラムの持つ「意味」を数学的に追求し、それを活かして抽象構文木を解釈するインタプリタを実装しました。

イラスト班

原 佑馬
情報理工学部　2回生

小柳 雅文
情報理工学部　1回生

岡本 陽太
情報理工学部　2回生

北村 優奈
情報理工学部　2回生

坪倉 奏太
情報立命学部　2回生

中川 拓海
情報理工学部　2回生

中山 凌一
情報理工学部　2回生

服部 瑠斗
情報理工学部　2回生

松本 幸大
情報理工学部　2回生

伊藤 聡子
情報理工学部　3回生

1.　活動概要

ペイントソフトである CLIP STUDIO PAINT の機能を基礎から学習し, 快適に高クオリティのイラストを描くことができるようになることを目指す. そのために, 毎回の活動では担当者を決め, それぞれが与えられたテーマについて調査し, それを活動内で報告することとする.

2.　CLIP STUDIO PAINTについて

https://www.clip-studio.com/clip_site/clipstudiopaint/scenes/design

本活動において利用したソフトである「CLIP STUDIO PAINT」は,　多機能付きのペイントソフトである. 基本的な機能は無料で利用することができるが，有料版のライセンスを購入することで,「CLIP STUDIO PAINT PRO」として全ての機能を利用することができるようになる. 本活動内では,班員全員が有料版を購入し，その機能について学習した.

3.　活動内容について

本班の活動は, 概要で述べたように毎回の活動で担当者を決め, それぞれが与えられたテーマについて調査し, 活動内で報告することとした. 全7週の活動におけるテーマは, 実際に絵を描く為に必要となる順番で定められており, その内容と担当者, 概要は次のとおりである.

- 第1週目

担当者: 原 佑馬

テーマ: CLIP STUDIO PAINTの基本的な使い方を学ぶ

概要:

基本的な設定や, プロジェクトの作成・保存方法などの基本的な使い方を学習した. また, この回で担当の割り振りも行った.

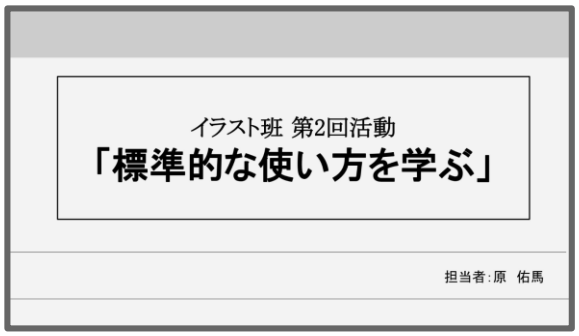
- 第2週目

担当者: 原 佑馬

テーマ: CLIP STUDIO PAINTの標準的な使い方を学ぶ

概要:

CLIP STUDIO PAINTを使用し, 本格的なイラストを描くために必要不可欠であるツールの使い方や, 機能の詳細設定の方法などを学習した.



- 第3週目

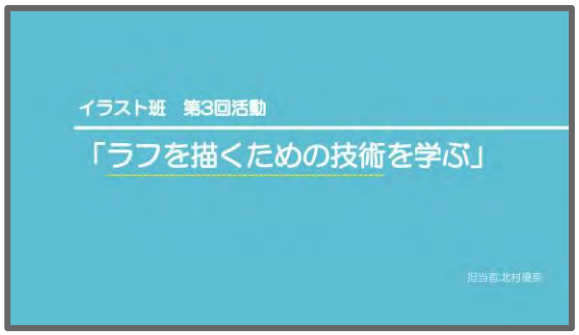
担当者: 北村 優奈

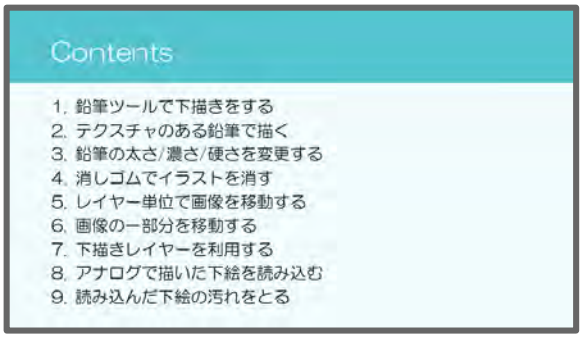
テーマ: ラフ(下書き)を描くための技術を学ぶ

概要:

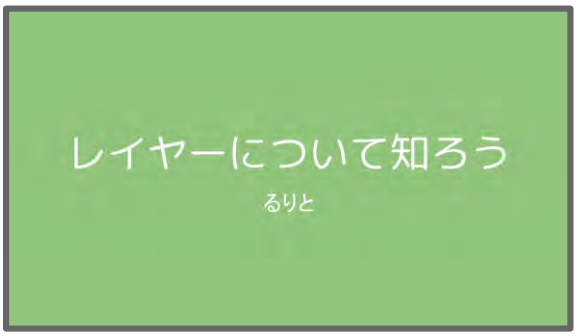
ラフ(下書き)を描くために必要な知識や, CLIP STUDIO PAINTでの下書きの書き方(主にペンの種類や, 色, レイヤの設定など)を学んだ.

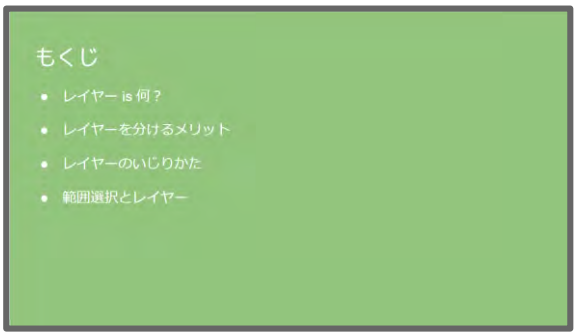
この回から具体的な絵の描き方の学習となった.



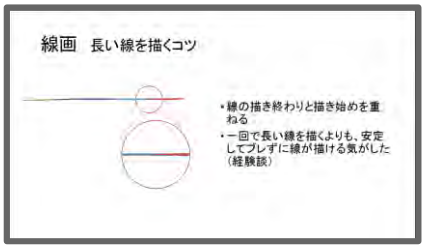


- 第4週目
  担当者: 服部 瑠斗
  テーマ: レイヤーに関することを学ぶ
  概要:
  　　デジタル絵を描く際に必須となるレイヤに関することについて学んだ. また, CLIP STUDIO PAINT上でのレイヤの操作についての学習も行った.

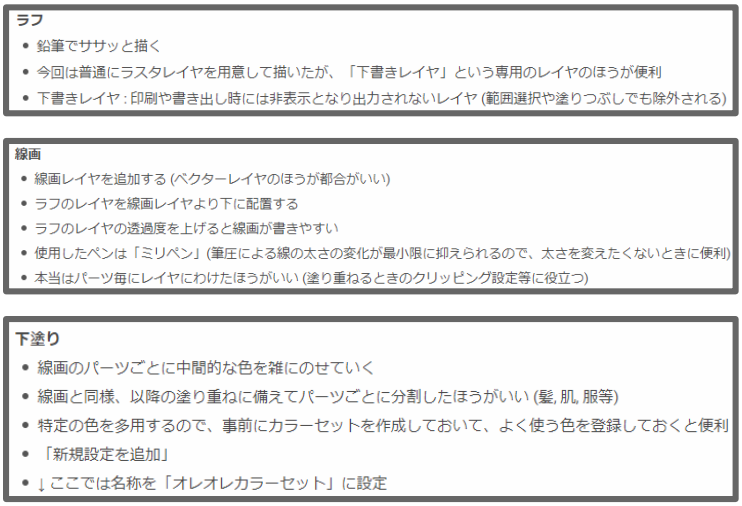




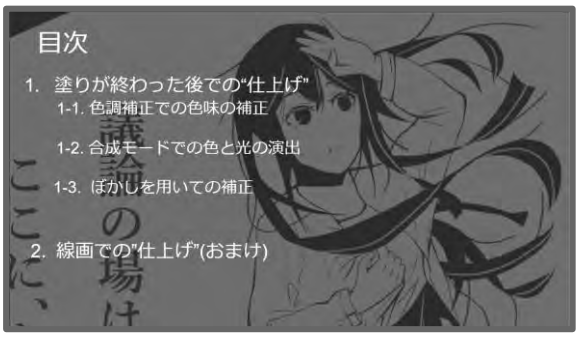
- 第5週目
  担当者: 伊藤 聡子
  テーマ: ラフを完成させるための技術を学ぶ
  概要:
  　　第3週目において描き方を学んだラフ(下書き)を, 線画(色塗りをする前の状態のイラスト)に仕上げるための技術を学んだ.



- 第6週目
  担当者: 松本 幸大
  テーマ: 色塗りに関する技術を学ぶ
  概要:
  　　第5週目において描き方を学習した線画に色を塗るための技術を学んだ.



- 第7週目
  担当者: 坪倉 奏太
  テーマ: 仕上げのための技術を学ぶ
  概要:
  　　これまでの回で学習したことを踏まえて完成させた絵に調整を加え, 完成させるための技術を学んだ. また, ここでは, 実際にCLIP STDIO PAINTを用いて絵を描くにあたって知っておくと良い事柄についても学習した.



4.　まとめ

　　本活動では, 主に CLIP STUDIO PAINT でイラストを描くことに焦点を当てて活動を行った. ここで学んだことを踏まえ, 班員各々がイラストを描く練習を更に重ねることで, 本プロジェクトの最終目標として掲げられた「CLIP STUDIO PAINT の機能を駆使し, 快適に高いクオリティのイラストを描くことができるようになる」を完全に達成することができると考えられる.

競技プログラミング班

服部　瑠斗
情報理工学部2回生

小村　漱一朗
情報理工学部 3回生

中野　海人
情報理工学部 3回生

稲垣　和真
情報理工学部 3回生

西見　元希
情報理工学部 2回生

浜田　直弥
情報理工学部　2回生

石川　流聖
情報理工学部　1回生

1. 活動の概要

本プロジェクトは、競技プログラミングを通してプログラミングにおけるアルゴリズムの知見を深めるために発足した。競技プログラミングに用いたサイトは AtCoder である。AtCoder が開催している AtCoder Beginner Contest や班内の活動内で開催したバーチャルコンテストを通して実際に問題を解き、その解法を活動で共有し、お互いの知見を高めたり、いろいろなアルゴリズムの手法について会員が解説し、理解を深めるというものである。以上の 2 点を主として本プロジェクトは活動した。

1. 競技プログラミングについて

競技プログラミングとは、出された問題に対し、その問題を解くプログラムを速く正確に作成し、解くまでにかかった時間を競う競技である。具体的には以下の4つの要素からなる。

- 問題文
  解くべき問題の内容が記されている。
- 制約
  問題文で出てきた数値や文字列に、どのような制約が与えられているかが記されている。
- 出力
  標準出力にどのような形式で与えるかの条件が記されている。
- 入出力例
  実際にプログラムに与えられる入力と、プログラムが出力すべき文字列が記されている。

3. 学習内容

3.1 アルゴリズム

本プロジェクトでは主に以下の 2 つのアルゴリズムを学習した。

- 深さ優先探索 (dfs)
- 累積和

3.1.1 深さ優先探索

深さ優先探索とは、全探索を行うアルゴリズムの種類の一つである。競技プログラミングでは主に、状態の遷移が分岐するような処理の実装に用いられている。深さ優先探索の特徴として、与えられた状態の深さに注目して探索を行うという点が挙げられる。また深さ優先探索を用いるメリットとして、再帰を用いて実装した場合コードがシンプルに記述することが出来るという点が挙げられる。

3.1.2 累積和

累積和とは、計算量を圧縮するために用いられるアルゴリズムの種類の一つである。競技プログラミングでは主に、配列上の区間の総和を求める処理の実装に用いられている。累積和の特徴として、前処理を行うことによって元々の配列が保持している情報に加えて配列のある区間の総和も求めることが出来るという点が挙げられる。

3.2 競技プログラミングにおけるテクニック

本項では、この活動において学ぶことのできた競技プログラミングにおけるテクニックを述べる。

3.2.1 マクロの活用

define マクロの活用は競技プログラミングにおける提出コードの冗長性を大きく削減し、より簡潔なコードを実現する。中でも単純な繰り返しにおける rep マクロは最も頻繁に利用される。次のコードは rep マクロの定義と利用例、マクロを用いなかった場合との比較である。

```cpp
#include<iostream>
using namespace std;

#define rep(i,n) for(int (i)=0;(i)<(n);++(i))

//マクロを利用しない場合のコード
int main(void){
  int n=10;
  vector<int> a(n);
  for(int i=0;i<n;++i)
    a[i]=i;
  for(int i=0;i<n;++i)
    cout << a[i] << endl;
  }

//マクロを利用した場合のコード
int main(void){
  int n=10;
  vector<int> a(n);
  rep(i,n) a[i]=10;
  rep(i,n) cout << a[i] << endl;
}
```

このコードでは要素数 10 の可変長配列に要素の座標と同じ数値を格納しそれを出力しているが、rep マクロを用いることでコードが短くなり可読性が向上したのが確認できるだろう。

3.2.2 ラムダ式の活用

ラムダ式は関数型プログラミングにおいてよく用いられる言語機能であるが、競技プログラミングでの主流言語である C++ にもその機能があり、ソートなどの引数として極めて便利に利用することができる。次のコードはラムダ式を利用した整数列のソートの例である。

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(void){
  int n=10;
  vector<int> a(n);

  //a={0,1,2,3,4,5,6,7,8,9}
  rep(i,n) a[i]=i;
```

```
//aを降順にソート
sort(a.begin(),a.end(),
      [](int x, int y) -> auto{return x>y;});

//0246813579のように偶数と奇数に分けてソート
   sort(a.begin(),a.end(),
      [](int x, int y) ->
         auto {if(x%2==y%2)return x<y;
               else if(x%2)return x<y;
               else return x<y;});
}
```

このようにラムダ式を用いたソートは非常に汎用的なソートが行える。

4. 活動で得られたもの

本プロジェクトで得られたものは大きく 2 つに分けられる。

1つ目は、競技プログラミングの問題に対してどのアルゴリズムやテクニックを用いるかを判断する力である。具体的には、制約を見ることによって判断することが出来る。理由としては制約から入力される値の上限が分かる。上限の値が判明すればその値が入力された場合の最悪計算量（オーダー）を大まかに求めることが出来る。最悪計算量が判明すれば、そこから全探索が出来るのか出来ないかといったアルゴリズムやテクニックの選択が可能になる。

2つ目は、アルゴリズムやテクニックを実際のコードに落としこみ問題を解く力である。一般的にアルゴリズムやテクニックをただ学ぶだけでは実践で苦労してしまう。そこで本プロジェクトでは、問題の内容を細分化することによって実践に落としこむ方法をいくつかプロジェクトメンバー間で共有することによって問題を解く力を習得した。

5. 展望

今回の活動では、基礎的なアルゴリズムやテクニックの学習やバーチャルコンテストを用いた実践の体験を行った。今後の展望としては、活動で扱ったアルゴリズムなどを組み合わせた新しいアルゴリズムを用いて問題を解く力を養う為の活動を行いたいと考えている。

参考文献

競技プログラミングを知ろう https://book.mynavi.jp/manatee/detail/id=56242

LT班

程 瑞希
理工学部 3回生

山口 流星
理工学部 3回生

渥美 柾彦
理工学部 3回生

稲垣 和真
情報理工学部 2回生

立川 泰暉
情報理工学部 2回生

林 紘也
情報理工学部 1回生

宇佐 基史
理工学部 1回生

深田 紘希
情報理工学部 1回生

堀田 隆成
情報理工学部 1回生

1.はじめに

文責：程瑞希

我々情報系の大学生にとって，各種イベントでライトニングトーク(以下，LT)という5分間のプレゼンテーションを行ったり傍聴する機会は多い．特に本サークルでは毎週行われる定例会議でLTを発表する機会があり，部員は1年に原則1度のLT発表を義務付けられている．本プロジェクトでは情報分野において重要な要素であるLTに対して，「どんなに素晴らしいものも伝わらなければ意味がない」を念頭に，プロジェクトリーダーを筆頭として，班員のLTの全体的なクオリティの向上を目指して活動を行った．

2.活動内容

活動をするにあたり気をつけた点や学んだ点を以下に活動内容として記述する．

2.1.1.自己紹介スライド

文責：林紘也

プレゼンテーションにおいて冒頭の30秒は，聴衆がそのプレゼンを聴くに値するか判断する時間であり，冒頭の自己紹介で聴衆を引き込めるかどうかがプレゼンの重要な要素の1つである．以下私の作ったスライドを元に私の考える自己紹介スライドの要点について述べる．

2.1.2.情報量の取捨選択

名前，学部，回生などの覚えてほしい重要な情報は簡潔に書く．掴み部分の邪魔，またその逆にならないよう気をつける．趣味や単位の話は発表者のことを知っていなければよほどのことでないと印象に残らない．自己紹介の目的と矛盾するので除く．

2.1.3.掴み要素

幼稚園での似顔絵コンクールという明らかにどうでもいい，関係ないことを真面目に書くことで，突っ込みどころを作る．また真面目な文章のなかに「すくすく」といった雰囲気にそぐわない言葉で軽い笑いをいれる．これらで聴衆の興味関心を引きつけることを狙った．このように自己紹介スライドでの掴みは疑問点やインパクトを与え，興味関心を引かせるがその後の本題の邪魔をしない程度のものにする．

2.2.1.デザインの4原則

文責：深田紘希

デザインには，見やすく内容が伝わりやすいといわれている原則がある．LT班の活動ではそれをスライド作成の基本とした．私が作成したスライドを例に4原則をどう生かしたかについても書き留める．

近接

「近接」は関連する項目をまとめてグループ化することだ．関連する情報同士を近づけ ないことでページの構造と内容の直感的な手がかりを提供する．近接を活用するために余白を利用するとよい．

整列

「整列」は各要素を意図的に配置することだ．すべての項目は他のものと視覚的に関連していなければならない．

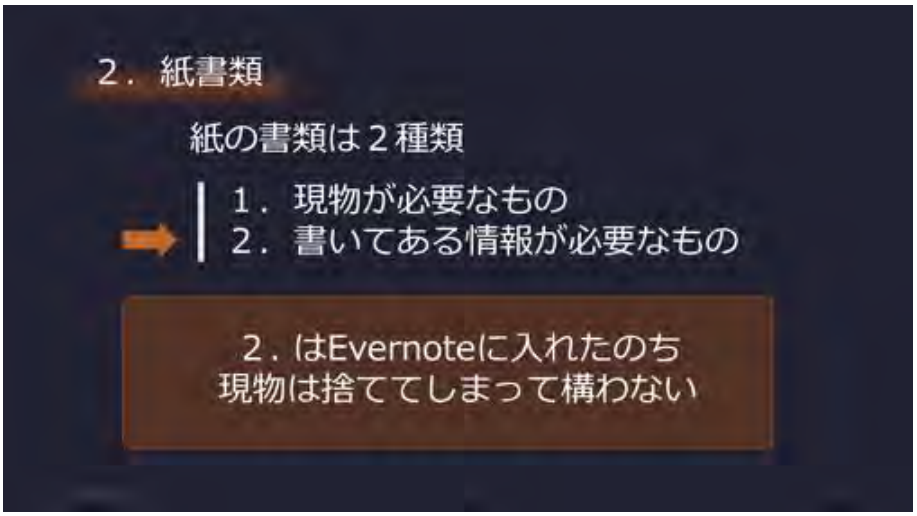


図2 近接と整列の例

対比

「対比」は異なる要素をはっきりさせることだ．色やフォントにおいて特に重要で，詳しくはのちの「色」，「フォント」で意識するところだ．

図3 対比の例1

図4 対比の例2

反復

文責：海原義樹

「反復」はデザイン上何かの特徴について，全体を通して繰り返すことであり，要素を一定のルールで繰り返すことで一貫性を持たせることができる．反復の例としてスライドの背景にテクスチャと呼ばれるものを用いることができる．テクスチャは，背景の画像の一形態で，何らかの意味を成すものではないが，背景 に模様を加えることで聴衆の印象を良くする働きがある．これを用いた例を図1に示す．また，この他にも，スライドの四隅などに，発表者や内容についてのアイコンや，ページ数などを配置することによって発表のわかりやすさや一貫性を高めることができる．

図5 反復の例1

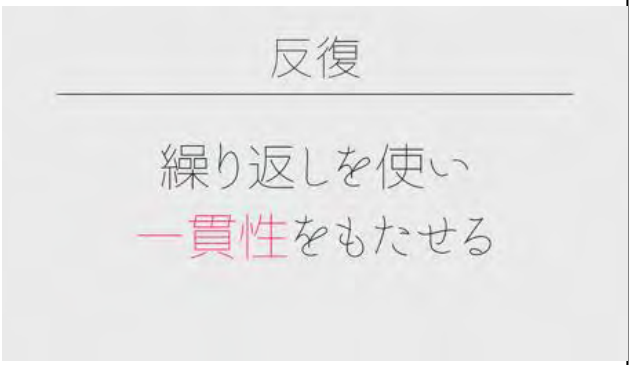


図6 反復の例2

色

文責：佐藤祐樹

配色の重要性

プレゼンテーション，ポスター，その他の創作物は，適切な配色を行うことで目的とする効果を引き出すことが可能である．例えば，視認性，識別性が重要な交通標識では彩度を高くする等のデザインがされている．しかし，プレゼンテーションにおいては前述のような高彩度は不必要であり，むしろ目 の疲労等の弊害を引き起こすため，低彩度でのデザインが望ましい．また，プレゼンテーションの場合は，その性質上，プロジェクターが用いられることが多いため，スライドの背景を白にすると眩しくなり，文字の視認性が下がる等の支障が生じる．この問題は背景を暗色等の明度が低い色にすることによって解決できる．

配色の質の向上

上記の通り，目的に応じた色をその都度選択することの重要性は明らかであるが，毎回状況に合わせて色を作成するのはあまりにも面倒である．しかし，一般にプレゼンテーションに適した色というのは存在する．そこでPowerPointのカラーパレットという機能 を使う．これは指定した色を保存しておくことができるというものである．「PowerPoint color palette」などと検索するとプレゼンテーションに適した色を集めたカラーパレットを見つけることができる．これを登録することにより，容易にプレゼ ンテーションの配色の質を向上させることができる．

フォント

文責：堀田流星

スライド上で使用するフォントを変えることで与える印象を変えるのは当然のことである.（例えば明朝体がフォーマルな場で使用されることが多い一方，ゴシック体がカジュアルな印象を与えるのは言うまでもない.）そのため実際の使用を踏まえた感想をここでは述べる.私がLT班で作成したスライドで主に使用したフォントは「游ゴシック」，「なつめもじ」[1]，「あんずもじ」[2]である.
游ゴシックは最近のOSでは標準で使用することのできるフォントで，無料で使えるフォントの中では大変美しく，使いやすいフォントである.「なつめもじ」「あんずもじ」は無料で配布されているフォントの一つで手書きに近いやわらかな印象を与えるフォントだ.私は最終的にスライドのほとんどにこのフォントを使用した.他にも班員の使っていたフォントを紹介しようと思う.「JKゴシック」や「軽フォント」「ようじょふぉんと」等である.以上のフォントをこのプロジェクト活動を通して知り，活用することで私のスライド並びに班員のスライドは洗練された印象を持ち，よりエレガントな表現が可能となった.

参考資料

[1] http://www8.plala.or.jp/p_dolce/index.html
[2] 同上
筒井美希,エムディエヌコーポレーション(MdN), ” なるほどデザイン”
https://unsplash.com/
https://www.transparenttextures.com/
https://sozaikoujou.com/30695

DTM 班

青木 雅典
理工学部 3回生

岡本 陽太
情報理工学部 2回生

狩野 優人
生命科学部 4回生

黒柳 裕大
生命科学部 1回生

齋藤 竜也
情報理工学部 1回生

高山 紗世梨
情報理工学部 2回生

田中 達也
情報理工学部 1回生

程 瑞希
情報理工学部 3回生

平井 柊太
情報理工学部 2回生

堀越 俊行
情報立命学部 2回生

正岡 玲於奈
情報理工学部 3回生

松本 幸大
情報理工学部 2回生

吉田 亨平
情報理工学部 3回生

1. 活動概要

DTM班は，12月に開催される学園祭に向けて自作した楽曲を発表することを目標に活動した.
そのために，作曲に関する知識を深め，実際に楽曲を作成することで各個人の技術力向上に努めた.

2. 活動内容

- DTM講習会

DTM初心者向けの入門講座をおこなった．この講座では，初心者を対象にDTM環境の構築や音楽の基礎知識についての説明を全6回に分けて行った.

2.1 第１回

作曲の基礎である、スケールや耳コピの方法を中心に大まかな曲作りのアウトラインの習得を行った。実際にDAWを起動し、スケールの音を鳴らしてみることも行った

2.2 第２回

楽器についての理解を深めた。具体的には講習資料のスライドに合わせてDAWも利用し、実際に楽器名と対応する形で音を鳴らして耳を慣らすことを試みた。

2.3 第３回

前年度のDTM班にて問題に挙がっていたDTMにおける金銭面の問題を考慮し、フリーVST・AUの紹介を行った。

2.4 第４回

第１回に触れたスケールについての知識補充と合わせて、コード進行の基礎についての講習を行った。主に、長音階と自然的短音階・和声的短音階・旋律的短音階について理解を深めた。

2.5 第５回

前回に引き続きコード進行の基礎についての講習を行った。カデンツの話を中心に、長音階や短音階を使用した進行についての理解を深めた。

2.6 第６回

前回に引き続きコード進行の基礎、その中でもカデンツについての詳細な知識を得ることや、実際の楽曲を聴くことでどのようにカデンツが曲の構成や雰囲気に作用しているのかについて学んだ。

3. 活動で得られたもの

今回の活動で得られた事は２つある．１つ目はコード進行やスケールなどの音楽の基礎知識の獲得である．これは活動で行われた勉強会を通じて得られたものである．２つ目は曲の構成の理解と制作方法である．これは耳コピや，オリジナル曲の制作をすることによって得られたものである．

4. 問題点

4.1 班員の講習会への参加率

他のプロジェクト活動と活動時間が被っていることが多々あったことや，例会後という比較的夜遅くでの活動となったため，十分に活動に参加出来ていない者がいた．

4.2 フォローアップ

全く音楽経験の無い班員に対する資料の解説が不十分であったことや耳コピによる楽曲制作の練習に時間を使いすぎたために，音楽の基礎知識の中でも十分に説明できていないものがあった．

自作キーボード班

青木雅典
理工学部 3回生

廣田公大朗
情報理工学部 2回生

堀越俊行
情報理工学部 2回生

阿部竜也
情報理工学部 1回生

玄元奏
情報理工学部 3回生

伊藤聡子
情報理工学部 3回生

西見元希
情報理工学部 2回生

芹澤拓也
情報理工学部 2回生

山岡聖弥
情報理工学部 3回生

吉田享平
情報理工学部 3回生

野﨑弘晃
情報理工学部 3回生

1．活動概要

本プロジェクトでは，実際にキーボードに触れて，そのインタフェースを学び，得られたものを元に作成するという理念の下，活動を行った．

第 1，2 週はコンピュータと人間を繋ぐインタフェースとして，キーボードの歴史について学んだ．実際に代表的なキーボードである，Apple Keyboard や PC-9801V などのキーボードに触れ，それぞれの構造やコネクタ，押下圧のかかり具合についての理解を深めた．

第 3，4 週はキースイッチの種類や主要な製造元について学んだ．最終的には基板を設計して自作のキーボードを作成するため，打鍵感などを元に個人個人の好みのメカニカルキースイッチを選好した．

第 5 週以降は，プロジェクトリーダー主導の下，オープンソースのプリント基板設計 CAD である KiCadの操作方法を学び，各々が自分の設計したいキーボードに応じた回路図と基板を設計した．

また，本プロジェクトは通年プロジェクトのため後期は設計した基板の発注，キーボードの実装を行う予定である．

1．キーボードの歴史

a．キーボード

コンピューターをはじめとする電子機器と人間を繋ぐインターフェースのことである．構造には一体型と組み合わせ型が存在，一体型は量産に適しており，組み合わせ型は個人レベルでの制作に向く．一体型にはメンブレン方式，静電容量方式，座屈バネ方式があり，組み合わせ型にはメカニカル方式がある．

b．インターフェース

コンピューターと周辺機器の接続を行うための規格や仕様，またはユーザーがコンピューターなどを利用するための操作方法や概念のことである．

c. コネクタ

時代とともにコネクタの形態は変化を繰り返している．この項目では，各時代で代表的なコネクタについて紹介する．

- RJ

  Registered Jack の略．米連邦通信委員会に登録されたもののことを言う．通信用ケーブルに使われる．

- D-SUB

  D-subminiature の略．9ピンの DE-9 コネクタは，日本国内向けの独自規格機で採用されていた．

- DIN9P

  ドイツ工業品標準規格 (DIN) によって規格化されたコネクタの一つ．PC-98 のマウスのコネクタに採用されていた．

- PS/2

  ミニDIN6P コネクタの形状を採用した PC の入出力ポートのコネクタの一種．この名前は，IBMPS/2 で採用されたことに由来する．

- USB

  Universal Serial Bus の略．ホスト機器に周辺機器を接続するためのバス規格の1つである．コンパック， DEC， IBM， Intel， MS， NEC，ノーテルネットワークスの 7 社が合同で 1994 年に開発，USB1．0 は 1996 年に登場した．現在は Apple， HP， Intel， MS， Renesas，STmicro の 6 社が主導の USB-IF が策定・管理を行っている．

d. キーボード本体

各時代で人気を得たキーボード本体について紹介する．

- IBM

  1981 年に発売された IBM PC は OS に MS-DOS，CPU にインテルを採用しており現代的な意味における PC の原点ともいえるものとなった．このときに付属した鍵盤に採用されたのが当時 IBM が特許を取得していた座屈バネ機構の鍵盤であり，長年に渡り大きな支持を得ることとなった．

- Apple

  1984 年に登場したMacintosh は今の Mac シリーズの原型となる製品で，従来式の CUI ではなく GUI とマウスを採用した革命的な PC だった．GUI操作を意識したキーボードの幕開けとして、今日のMacintosh配列に非常に近い形のキーボードである。

- PC98

  NEC が発売した PC シリーズの略称．初期はメカニカルスイッチだったが晩期はメンブレン方式のものである．

- 大量生産型

  それまでのこだわりを持って作られてきたキーボードは PC の大衆化・低価格化の流れによって市場から消えていき，製造コストの小さいゴム椀機構のメンブレン方式が主流となった．

3. キースイッチについて

キースイッチとは，ユーザの入力により電気信号を発する機械部分のことであり，キーの重さ(押下圧)や打鍵感は様々な種類が存在する．ここでは，自作キーボードで使用するスイッチについての特徴を述べる．

a. メカニカルキースイッチ

メカニカルキースイッチには，主に以下のような種類が存在する．

- Cherry MX
  Cherry 社 (ドイツ) が開発したスイッチ．Cherry MX の特許が失効したため，様々な互換品が製造されている．
- MX 互換
- ALPS
  アルプス電気株式会社 (日本) が開発したスイッチ．80~90 年代はほとんどのキーボードに採用されていた
  ほどであるが，現在は生産が終了し入手困難である．現在は互換品の Matias が代表的である．
- ALPS 互換
- ロープロファイル

b. 打鍵感

- Clicky
  クリック感があり，カチッと音がする．
- Linear
  キーを押した分だけ押下に必要な力が大きくなる．
- Tactile
  作動点で押下圧が重くなることでクリック感を作り出している．

c. 押下圧

キーの重さ，打鍵感を軸の色で識別している．代表的な 3 種類を以下に示す．

- 青軸
  Clickyでカチャカチャと音がする．押下圧は 60cN (100cN=1N)
- 茶軸
  Linearで静かなタイピングが可能．押下圧は55cN
- 赤軸
  小さな力でタイプできるため手が疲れにくく，タイプ音も静か．押下圧は 45cN

メーカーによって打鍵感が微妙に異なり，スイッチ選定の際は様々なスイッチを試し打ちすることが推奨される．

4. KiCad使い方

本プロジェクトでは，基板の設計に KiCad を用いた.

a. 導入

KiCad 公式サイトからダウンロードできる．KiCad の主な対応環境は，以下の通りである.

- Windows
- macOS
- Linux

b. 日本語化

デフォルト言語は英語となっているため，日本語化を行うと良い.
kicad.jpから日本語パッチを指定の場所に入れると，日本語化を行える.

c. プロジェクト作成

インストールした KiCad を起動し，図 1 のように新規プロジェクトの作成を行う．HOME ディレクトリに KiCad ディレクトリを作成し，KiCad ディレクトリ直下でプロジェクトを管理すると良い.

d. 回路設計

KiCadのEeshemaで回路設計を行う.
シンボル (部品) をつないで回路を構成する.

e. フットプリントの設定

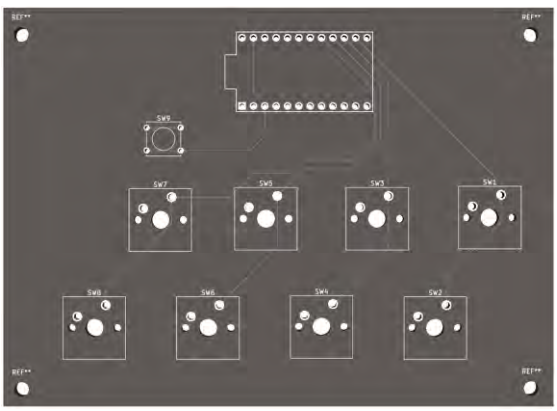
シンボルに使用するパーツ情報を紐づける．実際に使用する部品のフットプリントを割り当てる.

f. PCB設計

回路図を元にPCB (プリント基板) を設計していく．部品の配置，配線をパズルを解くように構成する.

5. 設計した基板

班員が設計した基板がこちら



メタプログラミング 班

西見 元希
情報理工学部 2回生

小泉 孝弥
理工学部 3回生

八木田 裕伍
情報理工学部 1回生

松本 幸大
情報理工学部 2回生

服部 瑠斗
情報理工学部 2回生

中尾 龍矢
情報理工学部 1回生

平井 柊太
情報理工学部 2回生

吉田 享平
情報理工学部 3回生

1. 活動概要

このプロジェクトは，メタプログラミングやその活用方法を学習・研究するというものである．また，発足背景として，RCCで久しく行われていない，「研究」を主目的としたプロジェクトを発足させたいというものと，好きな言語で特定の製作対象物を持たないコーディングをすることのできる場をRCCに設けたいというものがあった．活動は週2回行い，1日目は個人研究のための時間とし，2日目は班員の内一名が成果を報告した．班員は全員自身の研究や学習の成果を文書形式で提出することとしていた．

2. メタプログラミングとは

メタプログラミングとは，

> *メタプログラミングとは，ロジックを直接コーディングするのではなく，あるパターンをもったロジックを生成する高位ロジックを定義する方法のこと．*
> *主に対象言語に埋め込まれたマクロ言語によって行われる．*
> *(Wikipedia:メタプログラミングより引用)*

というものであり，本プロジェクトでは

- ソースコードを生成するコードを書くプログラミング手法
- 言語仕様を変更することで機能拡張を行う手法

と定義する．

プログラミング言語の種類によってどちらを行うことができるかは変わってくる．前者を行うものにはC++， Haskell，Rustなどがあり，後者を行うものにはRuby， Pythonなどがある．LispやJavaScriptは両方を行うことのできる言語機能を持つ．メタプログラムの実用例として有名なものにはRuby on Railsやコンパイラなどがある．

3. 活動成果

1. Common Lisp

a. 背景

競技プログラミングにおいてCommon Lispはややインターフェースにおいて使いにくい部分が存在するので，ユーティリティ群としてのDSLを実装し，コンテストに活かそうとしたものである．

a. 実装項目

実装したユーティリティマクロの代表として次のものがある．

- arrayの参照と破壊的変更を同時に行うvref
- オブジェクトを標準出力し改行するprintln
- 一次arrayのラップとしてのvec
- integer型のvecのラップvint

a. 実装前と実装後のコードの比較

```
;;DSLを使用しない場合
(defvar v
  (make-array 10 :element-type
'integer :initial-element
0 :adjustable nil :fill-pointer 10))
(dotimes (i 10)
  (setf (aref v i) i))
(dotimes (i 10)
  (format t "~a~%" (aref v i)))

;;DSLを使用した場合
(defvar v (vint 10 0))
(dotimes (i 10) (vref v i i))
(dotimes (i 10) (println (vref v i)))
```

a. 成果

このDSLを実装して，数回の競技プログラミングのコンテストにCommon Lispで出場したが，C++で解いた方が解きやすいことが分かった．筆者は結局C++でAtCoder緑に到達した．

2. Rust

a 背景

前からRustに興味があったので，メタプログラミング班で
Rustを勉強しつつ，メタプログラミングを勉強しようと思った.

b. 内容

前述に書いた通り，Rustに関しての知識が乏しかったので
競技プログラミングサイトであるAtCoderを用いて，Rustの
基礎文法や仕様について学んだ.

c 成果

Rust の基礎文法をある程度は定着させることはできた.
また，Rust言語の特徴である変数の可変・不可変や様々な型についても学ぶことができた．しかしながら，Rustにはまだまだ難解な概念が多く存在するため，今後も言語仕様の理解を深めていきたい．そして．この班の目標であるRustでのメタプログラミングをする方法についても学んでいきたい.

2. C++

a. 背景

メタプログラミングという単語を初めて聞いて，興味を持ったので勉強することにした.

b. 内容

勉強した内容は以下の箇条書きにある通りである。

i. C＋＋言語のメタプログラミングの基礎
    1. テンプレート関数
       単に型の違いだけで二つも関数を作らなければいけない場合，templateを用いるだけで二つの関数を一つにまとめることができる
    1. クラステンプレート
       関数と同様にクラスにのtemplateを使用できる
    1. テンプレート関数の特殊化
       テンプレートを使用において，<>の中に使用する型を記述する際，特定の型を入力したときのみ内容を変えること（特殊化）が出来る

i. 任意の型の可変数の引数を持つ関数の作成

ii. 任意の型の変数の型の判定をするプログラム

a. 成果

今までに知らなかった知識が入ったので，かなり新鮮な気持ちで勉強が出来た。
そして，今私が制作中のプログラムの一部に採用し，さらなる理解の獲得，効率化していこうと思っている。

4. JavaScript

a. 背景

JavaScriptでWebアプリを作るときのデータ管理ライブラリであるReduxを使用する場合のデメリットとして，実装しようとしている機能に対して記述量が多くなり，単純にタイプ数が増える上，保守性も下がることもあるというものがある．s2sは開発者が書いたソースコードを元に動的にソースコードを生成することで，開発者の負担を削減するソフトウェアである．今回はその仕組みの研究を行った．

a. 内容

Babel Handbook (https://github.com/jamiebuilds/babel-handbook) を読んでBabelの基本的な概念を学習し，Babelのコード (https://github.com/babel/babel) や型定義 (https://github.com/DefinitelyTyped/DefinitelyTyped)，s2sのコード (https://github.com/akameco/s2s) や読みながらs2sの仕組みを学習すると共に簡単なプラグインを作成した．

a. 成果

当初の目的であったTypeScriptでRe-ducksを書くためのs2sプラグインは完成できなかったが，Babel及びs2sの仕組みは理解でき，既存のプラグインのソースコードを読んで意味を理解できるようになった．途中まで作成しているので後期ではその完成とBabelを使用したさらに発展的なプラグインを作成したい．

4. 今後の展望

このプロジェクトは通年プロジェクトであるため，前期活動で得られた言語仕様に関する知見を活かして更に発展的な技術を学習したりDSLを実装したりしていく．また，一定の成果が既に得られた者に関しては研究する言語を別のものに変えていくことも検討している．

プログラム意味論班

松本 幸大
情報理工学部 2回生

小泉 孝弥
理工学部 3回生

深田 紘希
情報理工学部 1回生

西見 元希
情報理工学部 2回生

山口 流星
理工学部 3回生

堀田 隆成
情報理工学部 1回生

八木田 裕伍
情報理工学部 1回生

田中 聖也
情報理工学部 1回生

本班はプログラムの持つ意味を追求することに関心を持った会員が『プログラミング言語の基礎概念 五十嵐 淳(著) サイエンス社』をもとに操作的意味論の知見を深めたプロジェクトである．「ML言語の評価に基づく意味を導出システムで与えること」を目標として，実際にプログラミング言語の一種である『OCaml』を用いて実装も行った．

本文は理論と実装の二部構成となっている．ただし，理論パートは直感的に理解しやすいよう，厳密さを一部取り除いてまとめている点について留意されたい．詳しくは本会のホームページに掲載されている報告書ないし参考図書を参照されよ．

- 理論パート　評価に基づくML言語の導出システム -

● 操作的意味論

操作的意味論とは，プログラムの動作を記述することによってもたらされる意味について論理と数学的知識を持って研究する学問である．

● 判断と導出システム

型や判別式など，何かしらの**判断**を，**推論規則**を用いて導出する記述体系のことを**導出システム**という．導出システムは，判断と推論規則を含むものと考えるとよい．ある判断が与えられて，その判断がある導出システム上の持つ推論規則に従って導出できれば，判断は正しいことになる．この時，判断の**内容が誤っていても構わない**ことに注意する．例えば，内容的に誤っている判断を誤りだと導出する推論規則を持つ導出システムを作ることも出来るためである．

急に小難しい単語が列挙されたが，案ずることは無い．導出システムの例を挙げよう(あくまでイメージの範囲で)．

自然数の足し算に関する導出システムを用意する．

判断は，「N + M = L が導出できる(N,M,Lは 0を含む自然数)」とする．自然数の足し算は次の２つの推論規則があれば可能である．

- N + 0 = N　(片方の自然数を0とすると，恒等的に成立する．)
- N + (M+1) = (N+M) + 1 (この操作は繰り返すことが出来る．)

判断として，「1 ＋ 2 ＝ 3 が導出できる」を与える．この時，初めに1つ目の推論規則を用い，次に2つ目の推論規則を M = 0 として用い，最後に M = 1 として再び繰り返すことで，判断が導出でき，この判断は正しいと言える．といった具合である．

この導出システムを大きく拡張し，「ML言語の評価に基づく意味を導出システムで与える」ために，必要となる概念を一部まとめておく．

- **式(Exp)**

式はML言語上ではプログラムそのものである．具体的には整数値，真偽値，条件式，算術式，let式，関数などがある．

真偽値(bool)は，TrueとFalseのみを値に持つ．**条件式**は，もしAならBそうでないならCという式(A,B,C: Exp) ．**算術式**は，算術演算子(＋，＊など)を含む式.

- **評価と簡約**

評価(Eval)は，式を計算し，**値(Value)**を与えることである．
一方で**簡約**は，値(結果)までの過程を結果として与える．つまり式を簡約することは，その式がより単純になった別の式で表されることを示す．簡約を繰り返しそれ以上簡約できなくなる時，式は値となる．

● ML言語と導出システム EvalML

ML言語は，式がプログラムとなり，処理が評価であるプログラミング言語である．この言語の意味を追求する導出システムとして，**EvalML**を定義する．先に述べたように導出システムは，判断と推論規則を含むが，同時に扱えるデータも厳密に定義する必要がある．推論規則についてはここでは触れないが，どのような計算を行うことが出来るかイメージが掴めさえすれば十分である．

● EvalML1Err (基本的な計算)

扱うデータは5種類である．データの**集合**は大文字で表現し，集合の要素を小文字で以下のように略記する．

- 整数値(**Int**)，真偽値(**Bool**)，値(**Value**)，式(**Exp**)，演算子(**Prim**)
  - i ∈ Int：整数値　　b ∈ Bool：真偽値(true，false)
  - v ∈ Value：整数値か真偽値のどちらかの値を持つ．
  - e ∈ Exp：整数値，真偽値，条件式，算術式
  - op ∈ Prim：＋，ー，＊，＜

判断は次の4種類である．
ただし，評価は推論規則（特にop）によって簡約を進めて値を得る．

- 式 e を評価すると値 v である．
- 2つの整数値を演算(＋，－，＊)すると整数値 i である．
- 2つの整数値を比較(＜)すると真偽値 b である．
- 式 e を評価すると error である．

いくつか例を挙げておく．

- (4＋(2＊3)) という式を評価すると10という値である．
- (3<4) という式を評価するとtrueという値である．
- (2+false) という式を評価するとerrorである．
  ＊ error は値ではなく，実行時エラーとしてプログラムの実行を終了する．(実際の処理では評価前にエラーを検出する)

● EvalML3 (定義・環境・関数・再帰の追加)

導出システム EvalML1 での判断は非常に単純であり，MEMORYのない電卓の感覚に近いと言える．より複雑な計算を行うために，次の4つの概念を導入し，EvalML1を拡張することを考える．なお，ページの都合により，導出システム EvalML3 の定義は割愛する．また，OCamlはML方言の一種であり，以降これに準拠した定義が中心になる．

- **定義(let)**

ここでの**定義**とは，「式（の値）に名前をつけて，別の式の中でその名前を通じて式の値を参照するための機能」であり，**変数**は記号に付けられた名前である．

定義するためには，次のような**let式**で表す．

let <変数> = <式１> in <式２>

let式の読み方は，「変数を式１(の値)として式２の値を求める」である．例として，次のように変数宣言すると，「変数 x = 1 + 2 として，x + 2 の値を求めよ → つまり 5 」ということである．

let x = （1 + 2） in （x + 2）

これだけを見ると，C言語などの，int x = 1 + 2; という変数の初期化と，変数を用いた算術式の計算に過ぎないように見えるかもしれない．しかし，ML言語の処理とは異なることに注意しよう．C言語の場合は，上のように変数宣言をしたとしても，変数 x は自由に値を変更できる．

しかし，ML言語の変数は，変数宣言の有効範囲の関係により変数の値は式１に束縛されて変更することが出来ない．なお，変数宣言の有効範囲についての説明はやや複雑なため，割愛する．

- **環境**

値が不定な変数 a があるとすると，式 a + 2 の評価も不定である．「何を当たり前なことを．」と思うかもしれないが，この概念は重要である．”変数と値の組” これを環境と定義する．つまり，

環境 (a = 2 , b = 3) のもとで，式 a + b は 5 である．

ということである．このように環境は変数名と対応する値がセットになって，評価が行われる．よって，EvalML1で使っていた判断「式 e を評価すると値 v である．」は，次のように拡張される．ただし，環境 ε の中身は空の場合もあり得る．

判断：環境 ε のもとで，式 e は 値 v となる．

- **関数**

関数は，頻繁に表れる計算パターンを抽象化して繰り返し使用できる式であり，非常に重要な概念である．関数は次のように定義する．

fun <変数> → <式>

関数も式であるため，定義のテイギに基づいて let式 を用いると次のように名前を付けることが出来る．

let <関数名> = fun <変数> → <式> in <処理>

処理には [<関数名> 値] と記述すると，関数に値を渡すことが出来る．これを**関数適用**という．

例えば，５を２乗する関数であれば，次のように定義すればよい．

let square = fun x → x * x in square 5

また，関数の関数を定義することも出来る（関数は”式”より）．これは**高階関数**と呼ばれるが，ここでは割愛する．

- **再帰**

再帰とは，いままさに定義しようとされているものが現れることであり，数列の漸化式などを考えると分かりやすいと思われる．

再帰の定義は，let式と似たように記述できる．

let rec <変数> = <式１> in <式２>

- 実装パート　意味論を意識したMLインタプリタ作成 -

このプロジェクトの後半では，前半に学習した意味論を復習しながらプログラミング言語OCamlを使用して抽象構文木を簡約することのできるインタプリタを作成した．ここではその簡単な紹介を行う．

● 基礎的な算術式を評価する抽象機械の定義

ここでは整数・加算式・乗算式のみを算術式とし，それらをOCamlインタプリタ上で扱える概念として型定義を行った．

```
type expression =
    Num of int
  | Add of expression * expression
  | Multiply of expression *
expression
```

これが算術式をexpression型の定義であり，値の例は次のようになる．

- Num 3:算術式3という意味
- Add (Num 2, Num 3):算術式2+3という意味
- Multiply(Add(Num 1, Num 2), Num 4):算術式(1+2)*4という意味

また，抽象構文木を直感的に表示する関数も作っておく．

```
let show exp =
  let rec showexp = function
    Num a -> string_of_int a
    | Add (x,y) -> "(" ^ showexp x ^ " + " ^ showexp y ^ ")"
    | Multiply (x,y) -> "(" ^ showexp x ^ " * " ^ showexp y ^ ")"
  in "<< " ^ showexp exp ^ " >>"
```

これは関数showの中でローカル関数showexpを作り，その結果を<<　>>で囲んだ文字列を返す関数である．

showexpは算術式を文字列で表記するが，

- その算術式が整数ならば単にその整数のstring
- Add(x + y)という加算式ならば(x + y)という文字列
- Multiply(x + y)という乗算式ならば(x * y)という文字列

をそれぞれ返す．

これを使うことで，先程のMultiply(Add(Num 1, Num 2), Num 4)という算術式は<< ((1 + 2) * 4) >>というように表示される．

次に，算術式を1段階だけ簡約する関数reduceを実装する．この関数の動作は次のようになる．

- 整数を受け取ったらそれ以上簡約できないのでそのまま返す
- 加算式を受け取ったら，可能であれば左辺を簡約する．左辺が簡約不可能であれば右辺を簡約する．両方とも簡約不可能であればその式はAdd (Num x, Num y) の形になっているのでNum (x+y) を返す．
- 乗算式も加算とほぼ同様のことを行う．

この関数はOCamlのパターンマッチという機能のおかげで直感的に記述することができる．

```
let is_reducible = function
    Num a -> false
  | _ -> true

let rec reduce = function
    Num a -> Num a
  | Add (Num x, Num y) -> Num (x + y)
  | Add (x,y) ->
      if is_reducible x then Add (reduce x, y)
      else Add (x, reduce y)
  | Multiply (Num x, Num y) -> Num (x * y)
  | Multiply (x,y) ->
      if is_reducible x then Multiply (reduce x, y)
      else Multiply (x, reduce y)
```

これで実際に式の簡約ができるようになった．(以下，実行例)

```
# let e = Add (Add (Add (Num 6, Num 2), Num 3), Add (Num 1, Num 4));;
val e : expression =
  Add (Add (Add (Num 6, Num 2), Num 3), Add (Num 1, Num 4))
# reduce e;;
- : expression = Add (Add (Num 8, Num 3), Add (Num 1, Num 4))
# let e2 = Add (Add (Add (Num 6, Num 2), Num 3), Multiply (Num 1, Num 4));;
val e2 : expression =
  Add (Add (Add (Num 6, Num 2), Num 3), Multiply (Num 1, Num 4))
# reduce e2;;
- : expression = Add (Add (Num 8, Num 3), Multiply (Num 1, Num 4))
# reduce (reduce e2);;
- : expression = Add (Num 11, Multiply (Num 1, Num 4))
# reduce (reduce (reduce e2));;
- : expression = Add (Num 11, Num 4)
# reduce (reduce (reduce (reduce e2)));;
- : expression = Num 15
```

これ以上解説するにはページが足りないので，残りは本会のWebページにアップロードされている報告書を参照してほしい．

3章

会員コラム

　この章では、学年にかかわらず、会員一人ひとりの情報系の技術に関する知見や興味のある分野に対する熱意や想い、その魅力などを綴ってもらいました。情報系の技術に初めて触れたことについて語ってくれた会員もいれば、既に精通していて専門的な内容を語ってくれた会員もいます。

　コラムの内容は、アプリケーション、Webサービス、プログラミング言語、競技プログラミング、DTM、機械学習、アルゴリズム、デザインと多岐にわたり、本会の活動の幅広さと、各会員がどのような分野に興味を持っているのかを感じていただけるかと思います。

　初めて情報系に触れる方でもわかりやすいように書かれておりますので、様々なコラムに目を通して情報系の魅力を感じていただけたらと思います。ぜひお気に入りのページを探してみてくださいね。

AtCoder10ヶ月で水色になった話

学部/回生	情理 / セキュリティネットワーク / 1回生
名前	石川琉聖 (xryuseix)

初めまして，石川琉聖です．さて，AtCoderってご存知ですか？簡単に言うとプログラミング能力判定サイトで，特に競技プログラミングという早く正確にアルゴリズムの構築ができるか判定してくれるサイトです．　AtCoderにはユーザのクラスタリングを行なっており，レートと呼ばれるプレイヤーの能力値に応じてユーザー名に色がつくという仕組みになっています．

AtCoderのクラスタリングに関して，社長の高橋直大氏はこのようにおっしゃっています．

> 400ごとに色がついていて、**赤・橙・黄・青・水・緑・茶・灰・黒**、という順番になってます。
>
> 誤解を恐れずに超ざっくりとイメージでの評価をするなら、
>
> - 灰色は参加すれば誰でもなれるので意欲以外の保証はなし。
> - 学生で茶色なら優秀だがエンジニアとしてはちょっと物足りない。派遣とかで来たエンジニアが茶色あれば一安心。
> - 緑あれば大抵の企業でアルゴリズム力は十分。AtCoder的には決して上位ではないが、他社評価サイトなら最高評価。
> - 水色だと基礎的なアルゴリズム処理能力については疑いのないレベル。
> - 青以上は一部上場のIT企業でも、一人もいないことが結構あるレベルになる。
> - 黄色からは化け物。競プロの問題を解く機械だと思っておけば良い。
> - 橙はあたまおかしい。
> - 赤はもうなんか世界大会とかに招待されたりする。
>
> って感じに思っておけばいいです。誤解をしたくない人は、以下に詳しい説明があります。
>
> 引用：(http://chokudai.hatenablog.com/entry/2019/02/11/155904)

本記事ではAtCoderで水色になるまでどんなことをしたのか，という事に関して書きます．

まず，AtCoder水色とは，AtCoder内で上位約15%，全エンジニアで約１～２％と言われています．ただ，この１～２％というのはアルゴリズム能力の事であり，実際の技術力を指している訳ではありません．

1. プログラムをいっぱい書きました.

　10ヶ月前の私はプログラムをほとんど書けなかったので，とにかく数をこなしました．わからない処理があればその度に検索しました．特にStandard Template Libraryとイテレータとオブジェクト指向の勉強は苦戦しました.

2．オンラインコンテストに多数出場しました.

　水色になるまでAtCoder36回出場し，他サイトでもCodeforces27回，LeetCode９回，ICPCなどに参加しました．コンテストのような，どう頑張っても模範解答が見られない状態で，かつ限られた時間内で問題を解くというのは集中力や思考力の向上に繋がりました.

３．先人が作った賢いアルゴリズムをいっぱい勉強しました.

　正確な数はわかりませんが，20くらいのアルゴリズムを勉強しました．AtCoder水色はここら辺のアルゴリズムを理解し，応用できる必要があります.

・動的計画法...ナップサック問題が解けるだけではなく，LCSや桁DPができた方がいいです.

・ダイクストラ法...二頂点の最短距離やその経路復元ができる必要があります.

・ワーシャルフロイド法...これを使うだけの問題はあまりなく，問題を上手に分解し，このアルゴリズムを使って距離を求めるのと，問題の答えが同じことを確認し，実行する，といったことが多い気がします.

４．結局どれくらいの問題が解ければいいのか.

　AtCoder400点が解けるようになれば十分です．ただし，400点は300点と違い，動的計画法とグラフ系問題の難易度が急に上がります.
　問題文をよく理解し，具体的に図を書いてみたり，数字を書いて実験してみたりすると解ける問題が増えるかもしれません．また，上記二種類の問題は「慣れが必要」とよく言われます．勿論私もそう思っていて，とにかく回数を積んで問題のイメージをうまく掴めるようになる必要があると思います.

是非，良きAtCoderライフを楽しんでみてください！

パスワード管理のススメ

学部/回生	理工学部　電気電子工学科　1回生
名前	佐藤　祐樹

突然ですが、みなさんはアカウントごとにパスワードを変えていますか？私は変えていませんでした。どうせ大丈夫だろうとたかをくくっていたからです。しかし、この油断により私の個人情報が危殆に瀕することになりました。

ことの発端は1通のメールでした。「ご利用のApple IDがiMessageのサインインに利用されました」というメールが、普段使っていないメールアドレスの受信箱にありました。今までにこういったメールが届いたことはなく、ついにフィッシングメールが自分にも来たか、と思いました。一般にフィッシングメールは不特定多数に送るため宛名の記載がありません。しかし、このメールには私の名前がありました。さらに、よく見てみると同じようなメールがもう1件ありました。これはおかしいぞ、ということでその場でパスワードを変更しました。ところで、Apple IDでは、そのアカウントにログインしたデバイスを記録する機能があります。しかし、それには秘密の質問に答えるか、Apple端末に送られたPINコードを入力するすることが必要です。質問など当然記憶になく、端末もなかったため、本当にアクセスされたのかどうかも分からず、釈然としないまま、時が経つのを待つしかありませんでした。

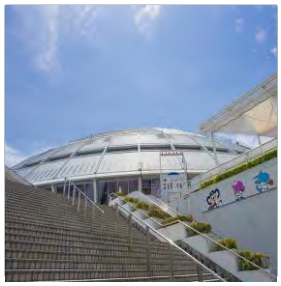
昼になって、ほかのアカウントもパスワードを変更しなければいけないということに気付きました。まず、Googleのパスワードを変更し、その際、Googleアカウントにパスワードマネージャが付属していることに気付きました。再利用されているパスワードや脆弱なパスワードをリストとして表示してくれる便利なサービスです。これを参考にほとんどすべてのパスワードを変更しました。幸いにもアクセスされた形跡はありませんでした。

帰宅し、確認すると送られてきていたメールはフィッシングメールではありませんでした。しかたなく、いくつかのアカウントを削除しました。

パスワードを使いまわしていたために、こんな事態になってしまったというのは明らかでした。Googleパスワードマネージャの「再利用されているパスワード」に、ずらっとアカウントが並んでいたからです。どこかのアカウントのパスワードが漏れ、Apple IDにアクセスされたと考えられます。

こんな事態にならないように、みなさんはパスワードを使いまわさずに、それぞれ違うパスワードを設定してください。先に述べたようなパスワードマネージャで一括管理すると便利なので、ぜひ使ってみてください。

Wi-Fi環境をマシにしました



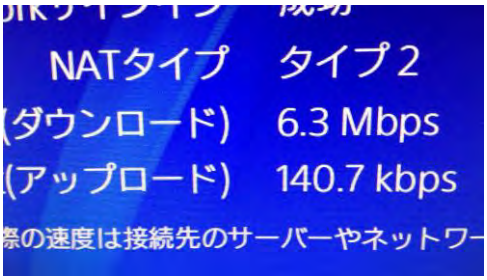
学部/回生	情報理工学部 SA 1回生
名前	小柳雅文

１：経緯

FPSをやってるときに急に落ちたり、Pingが50を超えたり、上りの回線速度が1000kbpsになったりするのでちょっとでもマシにしようと思い、いろいろ試しました。

２：ルータの買い替え

うちに前からあったルータはIEEE 802.11n の一世代前のものだったのでIEEE 802.acの8000円ぐらいのに買い換えました。ダメでした。1000kbpsが２Mbpsになったぐらいでした。



gm回線↑

３：Wi-Fi中継器の設置

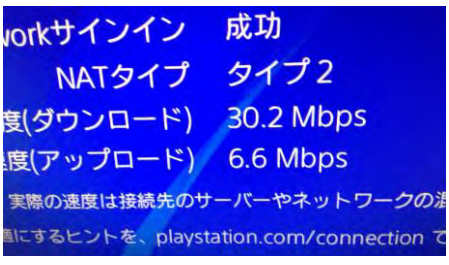
WiFi中継器とはその名の通り、WiFi電波を中継してくれるやつです。中継器のおかげで下り15Mbps上がり5Mbpsとか出るようになりましたが、普通に平均以下の数字です。

４：親に土下座

部屋の壁ぶち抜いて有線にしてくれと頼みましたが、銃で撃ちあうゲームのためにそんなことしてくれるわけもなく普通に断られました。

５：ルータの位置の移動

ルータの位置を5メートルほど回線が悪い場所の方向に移動させました。これが一番効果ありました。



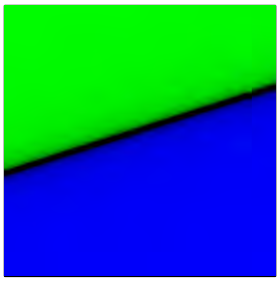
神回線（当社比）→

６：結論

回線悪すぎてマウス投げたり、キーボード叩きつけたり、机蹴っ飛ばして足を腫らしたりしちゃう人は骨折する前にルータの位置を変えてみてください。

グラフィックプログラム

学部/回生	情報理工学部・SAコース 1回生
名前	中尾　龍矢



おれは一回生に三人いる「たつや」のなかでも最弱っっ！！
疲れてくると変になります。
C/C++一筋でやってきました。最近はPythonとか練習しています。
グラフィック系とか科学計算系に興味があり、DirectX・OpenGL・レイトレーシングとか夏休みにやってました。
DirectXとかいうクソムズライブラリでうんコードを量産し、夏休みを大変無駄にしてしまいました。
今はOpenGLとPythonを真面目に勉強していて、何とか上手くいっているので少し安堵しています。
ちなみに右上の緑と青のアイコンはレイトレーシングをしていたときに副産物としてできたものです。
講義で習うことを、ゆくゆくはPythonで実際に使ってみたりと、やりたいことはたくさんあります。
ちなみにカクテルは「夢一夜」が好きです。
まるで一夜限りに咲く花のよう・・・といった具合に賞味期限15分の甘酸っぱく怪しげな彩りで儚い夜を演出します。
僕は飲んだことはありません。そんなもの飲まなくても疲れればどこでも夢の中ですがね。
知識の乏しさ故、これといって書くことなんてほとんどないのですが、強いていうならWindowsでグラフィック系のプログラミングを始める際のガイドライン程度です。
C/C++言語は動作がほかの言語よりも速くリアルタイム性を要するアニメーションなどをプログラムしたい際にはおすすめです。
使用ソフトはVisualStudio2017です。

・DXライブラリ

圧倒的初心者にお勧め。
グラフィックの基礎から応用まで幅広く学べます。
導入に関してはOpenGLよりも初心者には少しだけ難易度がありますが、そこさえ乗り切ってしまえば、後は酒池肉林を自由自在、翼を授かります。Visual Studio以外でも対応しているソフトが複数あり、Mac版も存在します。画面上に点や線などの図形、画像の読み込み貼り付け、加工が比較的簡単で、さらには、3D用の描画関数が存在するなど、初心者から中級者までの道しるべとなります。

・OpenGL

初・中級者向け
導入に関してはVisual Studioを持っていた場合、簡単すぎて馬鹿になります。
それ以外の場合はそこそこハードルがあります。
C/C++意外に、Pythonでも使用可能だったりと、門戸はそこそこ広くあいています。
大学で習う行列の知識を少しだけ要します。（ぶっちゃけなくてもいけます）
リアルタイムで3Dアニメーションがしたいときはとりあえずこれです。
Unity？知らない子ですね。
ただしプログラミング難易度は初心者にはそこそこハードルがあり、専門用語の知識のほかに、Windowsプログラミングの知識も要します。
勉強をしながら少しずつ学んでいく人にはお勧めです。

・DirectX

ラスボス。
初心者がまず挑んで
勝てる相手ではない。
バージョンは大体9～12まであり、
9は初心者向けだそうです。
（私がやったときは全くそんなこと
なかったです。）
3Dで本格的（組織で行うレベル）なゲームでも作成可能です。
DirectX12＆Windows10の相性は抜群で本来の威力を際限なく発揮できるそうです。（制作者談）
専門知識がかなり必要です、もはやその道にかなり詳しい感じの。
インターネットでも参考になる情報があまりなく（個人的意見）、C++言語以外にHLSLなどのシェーダプログラミングの知識もあるとベターです。

・画像などは「いらすとや」の素材をしています。
・内容は実体験、様々なサイトの情報を集めた、ざっくりとしたものです。

iOS13で強化されたARの話

学部/回生	学部 情理　学部・コース　SN　1回生
名前	unknow…

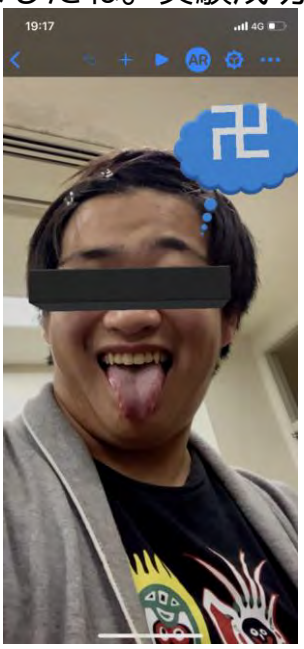
ついった〜
やってないもん

　初めまして、前期で4単位落としたunknowです。いきなりですが、皆さんiPhone使ってますか？みんなiPhone大好きですよね！ちゃんとiOS13にアプデしました？iOS13ではダークモードの追加やカメラやマップ、Face ID機能の強化が話題になりましたね！個人的にはバニラのキーボードで半角カナが打てるようになったのが嬉しいです：）
　ところで…iPhoneでARでなんか作って遊べるようになったことって知ってます？そりゃもちろんAR人気だしみんな触ってますよね！今日は何回目だ？って怒られるかもしれないけどこの新機能の話をします。

以下本編

1. Hello AR World🌅

　はい。早速AR開発のためのアプリを入れてもらいます。Reality Composer(https://apps.apple.com/jp/app/reality-composer/id1462358802)
んで、早速アプリを起動させてみてください。右上の＋のとこから新しいプロジェクトを立ち上げることができます。次の画面ではアンカーを設定する、ってのが出ますね。これはどういうことかというと、どこにARで設定したモノを表示させるか、ってことです。好きなもの選んでください。顔のやつとか選ぶと結構わかりやすいし面白いです。はい、選んだらUnityみたいな画面になりましたね。顔のやつを選んでるとこの時点でもう遊ぶことができます。左上のARって書いてあるボタンがあるので押して見てください。内カメラが起動して顔を認識したらなんか吹き出しが出ましたね。実験成功です。もう一度押すと編集画面に戻ります。



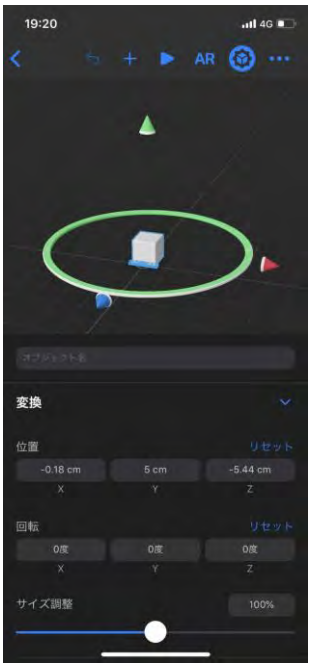
←こういう感じで遊ぶことができる(楽しい)
(目隠しのとこと吹き出しはAR)

2．編集画面 ⚙️

左上の＋ボタンを押してみてください。新たに表示させるオブジェクトを選び追加することができます。（unity等で.usdzというファイルに変換すれば自分の好きなオブジェクトを使うことができるはずです）

オブジェクトの位置、形、色、サイズ、素材感を変えたい場合オブジェクトを押してから左上の歯車と箱が合体したみたいなボタンを押してください。いじれます。

左上の団子みたいなボタンを押しビヘイビアをいじるとオブジェクトの動きをいじれるようになります。ここでは画面上のオブジェクトをタップしたときの反応なども編集することができる。



←Unityのような感覚で
編集することができる

3．作ったもの㊙️

上の作業をやったらとりあえず作りたいものは作れると思います。こっから書く内容が思いつかなかったので僕が作ったものでも紹介して個人コラムを閉めようと思います。

i. 名刺を読み取って追加の情報を表示

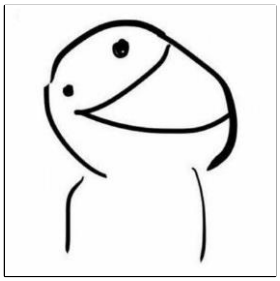
ミリマス６thSSAで交換した自分の名刺を読み取るとキャラのどこが好きとか出るっていうのを作りました。（版権的に画像は載せられない気がする😢）

ii. 関〇夫の画像を読み取ると某友愛結社のシンボルを表示

都市伝説の番組でよく出てくるあのおじさんの画像を読み込むとコンパスと定規で作られた有名なモチーフのアレ（RealityComposer上でオブジェクトを組み合わせて作った）が出てくる。（こっちも画像乗せると”組織”に目をつけられるので割愛）

過去の経験について

学部/回生	情報理工学部・実世界情報コース　1回生
名前	新藤



私は高校生の時にロボットを作ったり、実験を行ったりする部活に所属していた。その時にあった失敗談やトラウマの話をしていこうと思う。

①何故か動かないロボット

私は前述のように高校生の時ロボットを作っていた。Bluetoothを活かして、無線通信で動くロボットを作っていた。しかし、そこにはいくつもの失敗があった。

・初戦

当時、私はArduinoというハードウェア(基板)とソフトウェア(開発環境)からなるシステムを用いて、無線化を行っていた。プログラミングが完了し基板への書き込みも終わった。あとは、回路を組むだけという段階だった。

俺「さて、回路も組み終わったし、あとは、動くか試すだけだ。楽勝ンゴwwwwwwww」

と草を生やしていた。全く初めてやるやつが何を言っているんだか...このフラグは早々に回収された。

俺「あれ...動かねぇ...」

モーターが動かなかったのだ。当然パニックになる俺。プログラム、電子回路などを見直す。しかし、問題点が見つからない。そこで、部活動の先輩のアドバイスももらい、電流を計ってみたところ、回路の途中のモータードライバー(モーター制御装置)に通電していないことが判明した。実はこのとき、モーターを3つ用いていたのだが、動くものと動かないものがあった。同様にモータードライバーも3つ用いていた。つまり、回路ではなくモータードライバーがイカれていた可能性があった。

私の学校では、発表する日が決まっていた。そして、このとき、もう時すでに遅しという状況だった...こうして徹夜が決定した..(徹夜して有線で動くようにしましたorz....)

・最終戦

私はそれでもあきらめず何度も挑戦した。そして、最終的には何とか成功した？のだが、最後はかなりあっけなかった。

俺「何で動かないんだ....」

落胆する俺、このとき既に何度も失敗を重ね、そのたびに徹夜をした....(もはや、トラウマものだった)しかし、今回は違った。モータードライバーは変えたし、プログラムも何度も確認した。回路もモーター1個、モータードライバー1個で組んだ物の時は成功していた。

俺(なんで動かないんだ....)

俺の頭に徹夜の二文字が浮かぶ。その時、俺はとある事に気づいた。

俺(あれ？+-逆じゃね？)

動いた。組みなおしたときに+-逆になっていたのだ。喜びよりも呆れが勝っていた....

俺「えぇ...」

②奴らは、進歩のために犠牲になったのだ....
　ここからは、私が壊した物とその理由、症状を列挙していく
・モーターのギア：無数
　負荷をかけすぎた
　ギアが砕けた
・Arduino：3台以上
　過電流(多分9V電池を電源にしていたのが理由)
　プログラムを書き込めなくなった
・コンデンサー：1個
　＿人人人人人人＿
　＞爆　発　し　た＜
　￣Y^Y^Y^Y^Y^Y^Y￣
　多分、ショートした
・iPhone：1台
　OSが吹き飛んだ
　悪徳ソフトウェアをつかまされた(冷めるが、その直前にバックアップをとって置いたのに加えて、バッテリーが既にイカれていたのでほぼ無傷だった)
③終わりに....
　その他にも、様々な失敗をしてきたが、これ以上話すと、当時のトラウマが甦って吐きそうになるのでやめておこうと思う。
最後に、一言。何かやるときは事前に調べておくか、詳しい大人と一緒にやろうね！おじちゃんとの約束だぞ!

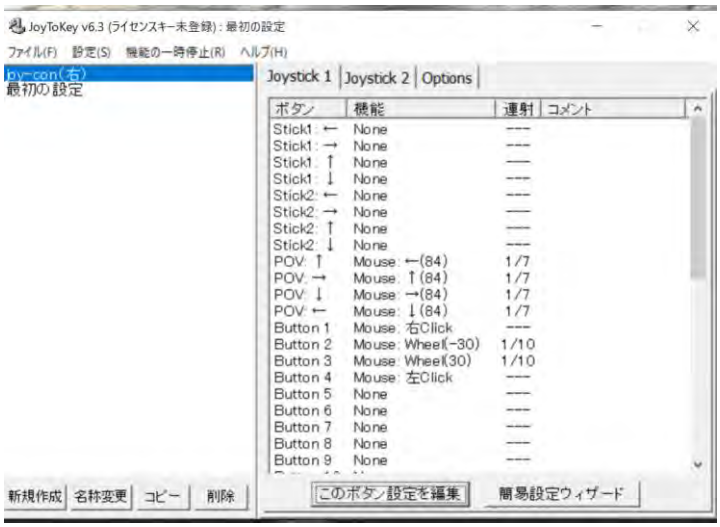
joy-conをPCにつなぐ話

学部/回生	情報理工学部・知能情報コース1回生
名前	林紘也

どうも初めまして、ほとんど部室に行かないので半分幽霊の林です。幽霊なので会誌を書くという話を締め切りが過ぎてから知りました、林です。

書くことがないので前からやろうと思っていたNintendo Switchのjoy-con接続を今からやります。僕は家でPCをテレビにつなげて寝ながら動画を見ているのですが、操作するためにいちいち動くのが面倒くさいのでjoy-conをPCにつなげて
リモコンにしていきます。

接続の準備は簡単でPCがBluetoothに対応さえしていればOKです。なお僕はWindowsをつかっているので、Windowsでのやり方を説明します（macでも同様にできるとは思います）。Bluetoothの設定画面を開き、joy-conをデバイスに追加
します。このときjoy-con側面のシンクロボタンをおしてください。成功すればjoy-conが一覧に表示されます。ただこのままでは使えません、joy-conにキー割り当てがされていないからです。その設定にはいろいろやり方があると思いますが僕はJoyToKeyというソフトを使いました。検索すればすぐ出てきて今のところ無料でつかえます（いずれ有料になるかも）。後はソフトをひらいてボタンを押したらそのボタンの欄が光るのでそのボタンに好きに設定を割り当てれば終了です。ただしこの設定はJoyToKeyのアプリの起動中のみ適用されるので注意してください。



↑
JoyToKeyの設定画面。適当にマウスの機能を割り当てました。
画像でページを埋めるなという意見は知りません

よくわからなかった人は「joy-con　PC」とかで検索すればもっと詳しいやり方が出てくると思います、というか出てきました。やるときはそっちを見てください。

プレゼン用のリモコンなんかにも使えそうなのでSwitch持ってる人は試してみたらいいんじゃないかなと思います。

ゲーム制作でプログラミング学習

学部/回生	情報理工学部・実世界コース 1回生
名前	ふかだ

- はじめに-

大学から情報系の勉強を始めた深田です。
プログラミング言語の基本文法を学ぶことを目的に、ゲーム制作をしました。

-作ってみたゲームと感想-

・ブロック崩し

弾と2次元配列のブロックと当たり判定を作れば完成します。基本的な入出力や、条件分岐、変数、配列、関数などの知識が必要になるので初心者にうってつけな　感じがします。サンプルコードもネット上に転がっていることが多いので、完成　させやすいです。

・シューティングゲーム

自機と敵機とが、同じ直線的な弾を打ち合うだけのプログラムです。作り終わった後に、異なる軌道の弾の追加や個性的な敵の追加など、拡張性が高いところがよいです。オブジェクト指向の恩恵をじかに感じることができました。

・テトリス

升目で仕切られた古典的２Dゲームというのは、ルールに則って１つの2次元配列を書き換えすることによって実現されています。テトリスはその中でも代表的な　ゲームといえるでしょう。
テトリスは作り終わった後、長い時間遊ぶことができます。愛着がわきます。
ホールドやTスピンの判定、難易度変更、レスポンスの改善、得点表示など、ちゃんと作ろうとすると大変になります。
ぷよぷよに改変することもできて、再帰呼び出しの勉強になります。

・ブラックジャック

ヒット、スタンド、サレンダーのみのシンプルなルールにします。
カードにはマークと数字の2要素があるので、ちゃんとデータを関連付けないと後で大変なことになります。
ブラックジャックは相手を用意することができます！ ブラックジャックのディーラーはルール上、人の意思が介在しないので、簡単に作ることができます。
1に1と10両方の意味を持たせたり、数の総和が同じになったときの勝利判定とかが大変でした。

// ...あれ？プログラミングっていうゲーム、たっぷり遊べるのでは？

言語について思ったこととして、Processingのように、はじめからWindowsFormが立ち上がる言語がいい気がします。print ""とかのターミナルの出力だけじゃゲーム制作的に味気ないけれど、CやJavaなどで自作するのはハードルが高めですからね。

僕もまだまだ始めたばかり。これからも勉強していきたいなーと思います！

オタク、DTM始めたってよ

学部/回生	生命科学部　応用化学科
名前	黒栁　裕大

あっはーーーー！！！！！！！
こんにちはーーーーーー！！！！

【自己紹介】（読み飛ばし可）
　やたらと声のでかいオタクです。よろしくお願いいたします。
　僕は生命科学部なんです。僕がなんで情報理工系団体のRCCに入ったかって言うと、この題名にもあるDTMってのをやりたかったからなんです。そんでもってその理由が初音ミク大好きだからっていうアレなんですけどポ！！！（顔の赤くなる音）ちなみに、どのくらい好きかっていうと初音ミクのタオル抱きしめながら授業受けるくらいには好きですね（高校の頃）ちなみに音楽経験とかはまっったくありません！！！！よっ！無能！！
　さてさて、自分語りも程々にしてそろそろ本題行きましょうかー！！

【DTMってなんぞや！】

　はい、こんにちは。事故紹介、読んでくれた人も、あろうことか読み飛ばした（任意の悪口）も思ったことと思います。DTMってなんや！
　DTMっていうのは、簡単に言うと、パソコンで音楽を作ろう！！ってことです。

【DTM、何が必要なの？？】

　{**絶対必要**}
・DAW（音楽を作る用のソフト）　・パソコン
　{**あればうれすいい**}
・ボーカロイド　・MIDIキーボード　・イヤホン
　　とりあえずDTMを始めるにあたってこれだけあれば十分！
DAWについてはワイくんは初音ミクv4xについてきた
Studio one 4を使っています。DAWについては慣れやし人によって使いやすさとかも違ってくると思うので一概にこれがいい！！っていうのは僕の口からは言えないのですけども、とりあえず無難に行くなら
cubase とかその辺が良さそう(急に適当になるオタク)

【音楽やったことないんだけど大丈夫？】

うす！！！！全然大丈夫っす！！！！！

DTMは楽器を弾いたりする技術は一切要りません！そりゃやってないよりはやってた方が音楽的には有利ですけども！知識なんかは後から身につけてけばいいんですよはいはい。実際私も大学入るまでは楽器なんて小学校でリコーダーやっただとかカスタネットの紐グルグルして遊んでたとかその程度ですし。

ただ、ある程度の音楽的センスはあった方がいいかもしれませんし、(ここで言うセンスっていうのは才能でなく経験が育む音楽感みたいなもの)そのセンスを伸ばすために1つ練習法を紹介しておきます。

【耳コピのお話】

耳コピって言います。よろしくお願いします。

耳コピっていうのは、ある曲を耳で聞いてそのままコピーしよう！ ってことです。これをすることで、どこにどんな音をおくと良くなるだとか、どの楽器をどんなふうに使うだとか、そういうことをお勉強するわけです。たのし〜〜〜〜〜

これ、1曲だけでもやってみるとほんとに全然違うんですよ。ある程度音をどこにおこうみたいな目処が建てられるようになります(それの巧拙はおいといて)

あと、普段聞いている曲をまた別の視点から見ることができるようになるんですよ。ここにこんな音入ってたんだ〜！っていうのが鬼増えます。鬼鬼鬼。普段なんとなーく聞いてなんとなーくいいな〜って思ってる(もちろんそういう楽しみ方もグレイトなんですけども)曲で
おっほぉ^〜ここに変な音入ってる〜キモティ〜みたいな楽しみ方もできるようになります。これを「耳が育つ」って呼んでます。

【まとめ】

欄が小さすぎてポッポ。みんなもDTMやろう！

単一チャネルでの
Speech-Speechブラインド音源分離

学部/回生	情報理工学部・実世界情報コース1回生
名前	阿部 健太朗

初めまして，阿部 健太朗と申します。健太朗と言っても，健康でもなければ太ってもなく，朗らかでもない人間です。そう，名前は意味を持たないのです。

さて，そんな私が今年（2019年）の夏に行ったプロジェクトとして，「単一チャネルでのSpeech-Speechブラインド音源分離」というものがあります。「複数人の声やノイズが混ざり合ったうるさい環境の中から，特定の音源のみを取り出す」というアーキテクチャです。

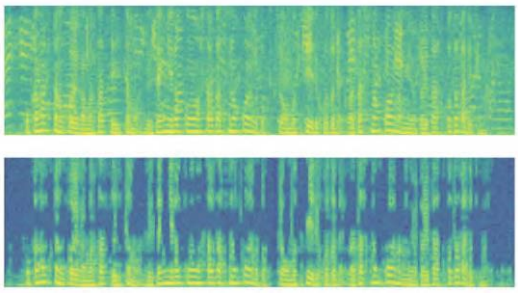
人は通常「選択的聴取（カクテルパーティ効果）」と呼ばれる”聴き分ける”能力を有してるのですが，これを工学的に実現したものを「ブラインド音源分離」と呼びます。このような”聴き分ける”能力を，単一チャネル（一台のマイク）で実現するブラインド音源分離の実装を試みました。

① 処理の流れ

② 短時間フーリエ変換（STFT）

目的音源（特定話者の声）を取り出す際の前処理として，STFTを利用する。これにより，音声データを時間周波数成分に分解することが可能である。

右図（上：分離前，下：分離後）がSTFTによって生成されたスペクトログラムであり，各時間周波数成分に対して“目的音源であるか”の推定を行う。



目的音源の成分のみを残し，その他元音源に含まれる非目的音源のマスクキングを行うことで，音源分離を実現した。

③ LSTM（Long short-term memory）

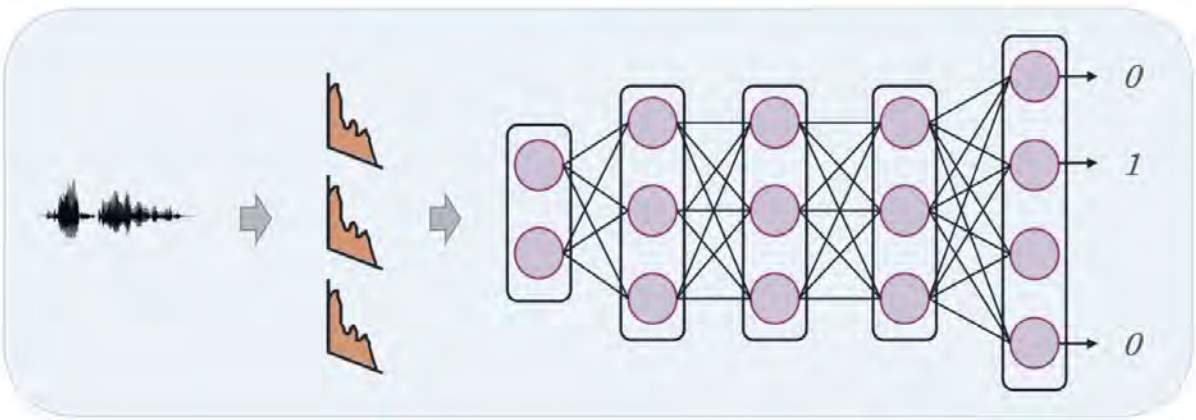
LSTMとは，時系列データに有用なニューラルネットワークアーキテクチャであり，これを用いることで時間周波数マスクの推定を行う。

LSTMはRNNと基本構造がよく似ており，RNNを訓練する際に生じる勾配爆発および損失問題に対処するために開発された。

今回使用したニューラルネットワークは，LSTM層を入力層とし，中間層にはfeed-forward層を2層用いた計4層で構成した。

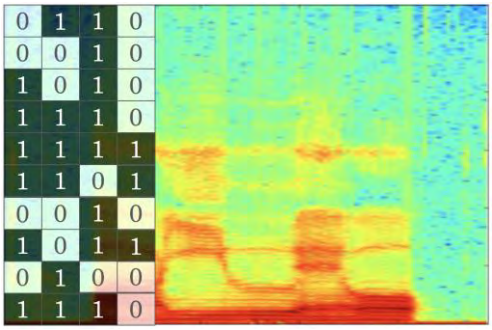
④ ブラインド音源分離

本システムの一連の流れは，以下の通りである。逐次入力されるサンプルに対してその都度マスキングを行うことにより，リアルタイム性を確保している。



【音源分離の手法】

前述した通り，STFTとLSTMを用いた音源分離の実現を図る。元の音源（時間領域）をSTFTで周波数領域に変換した後，時間毎のパワースペクトルをLSTMへの入力とし，時間周波数マスクを推定，非目的音源をマスキング（右図）する。



通常，Speech-Speech音源分離を行う際には，「複数台マイクを設置する」「カメラで話者の口元を撮影する」など，複数デバイスで行うことが多く，単一チャネルでの実現は困難であるとされていた（Noise-Speech音源分離は単一チャネルで実現可能）。そこでこのような手法を提案し，単一チャネルでのSpeech-Speechブラインド音源分離を実現した。

付　録

聴覚障害とは，単に音が小さく聴こえるだけでなく，「周りの音すべてが混ざってしまい，音の聴き分けが困難」というものである。選択的聴取は人間が先天的に備えている能力であるのだが，生まれつきの問題や加齢に伴う能力の低下，その他外的要因などにより，日本国内でおよそ32万人（2018年度，厚生労働省）が聴覚障害を患っているのが現状である。

聴覚障がい者は聴き分けることが難しいため，補聴器をつけたとしても，カフェやテレビがついた環境などでは会話を行うことが困難である。音源分離の先行研究はいくつか存在するのだが，それらは必ずしも彼らのニーズに即した物ではない。そこで，聴覚補助を目的とし，「聴き分ける」機能を搭載した選択的聴取システムの開発を行なった。

本システムは「私の声のみを取り出す」ことに最適化しているため，「対象話者」の切り替えが困難であるという課題がある。展望として，新規音響コンテキストへの対応の高速化が可能であるCADNNを応用したいと考えている。

Linuxデスクトップをカスタマイズしよう！

学部/回生	情報理工学部・SAコース・一回生
名前	やぎちゃん（八木田 裕伍）ygkn.github.io

皆さん楽しいLinuxライフをお過ごしですか？
Linuxを開発用OSとして使用している人はシェルやターミナルなどCUIをカスタマイズしていられる方も多いです。一方、デスクトップやアイコンなどのGUIはあまりカスタマイズしていない人が見受けられます。
LinuxはCUIだけではなく、GUIも様々なカスタマイズができます。ここではUbuntu (GNOME) を例に、Linuxのデスクトップをクールにカスタマイズする方法を紹介します。

GNOME Tweaks

GNOMEの見た目に関する高度な設定をするには GNOME Tweaks をインストールする必要があります。パッケージ `gnome-tweaks` でインストールできます。

```
$ sudo apt install gnome-tweaks
```

デスクトップ

アプリケーション（GTKのウィンドウの枠線や色など）、カーソル、アイコン、GNOME Shell（トップバー、アクティビティ、アプリの検索など）について設定を変更することでカスタマイズできます。

ログイン画面

UbuntuはGDM(*GNOME Display Manager*)というデスクトップマネージャーの設定を変更することでカスタマイズできます。

起動画面

実は、Ubuntuのログイン画面が出てくるまでのスプラッシュ画面もカスタマイズできます。
このスプラッシュ画面はPlymouthというソフトウェアで表示されており、設定を変更することでカスタマイズできます。

gnome-looks.org

https://www.gnome-look.org

　上記の設定を変更するテーマを検索できるサイトです。「dark」「blue」などのキーワードで検索するとイメージ通りのテーマが作られます。

カスタマイズの例 - macOS風

　GNOMEがmacOSとUIの構成が似ているからか、UbuntuをMac風にカスタマイズするのは定番で（筆者も昔やっていました。当時はまだOS Xでした）、gnome-looks.orgにもmacOSやOS X風のテーマが数多くあります。今回はそこからいくつかピックアップします。

McMojave
https://www.gnome-look.org/p/1275087/
　macOS Mojave風GTK+ テーマ
Mojave CT icons
https://www.gnome-look.org/p/1210856/
　macOS Mojave風アイコンテーマ
MacOS MOD
https://www.gnome-look.org/p/1241071/
　macOS風カーソルテーマ

おわりに

　僕は最近新しいパソコンを~~*My New Gear…*~~手に入れたのでデスクトップも衣替えしようかなと太陽の塔をイメージしたデスクトップにしてみました。皆さんもぜひかっこいい/かわいい/綺麗なデスクトップを作ってみてください！

DTMを書く...はず...

学部/回生	情報理工学部・実世界情報コース・1回生
名前	momonga

初めましてmomongaです。最近はDTMにハマっており、ハマっております。目標は中毒性高い音楽を作ることです。ということで今回はDTMについて語っていく....訳ではありません。はいごめんなさい。語りたいのは山々ですが、語れるほど強くないのでそれについてはまたいずれ....

はいということで今回はROS2についてつらつら書いていこうと思います。急にROS2でなんでやって感じですが、ROS2について書いていこうと思います。

・ROS2とはなんぞ？

ROSというのはRobot Operating Systemの略称で、ロボットアプリケーション開発のためのミドルウェアです。つまりなんやねんというところで、ロボットを動かすアプリケーションを作成するためのあれやこれやが詰まってるものです。ROS2はその第2世代として生まれたものです。具体的に説明すると「**publish**」「**subscribe**」「**service**」「**client**」の機能があります。今回は「**publish**」「**subscribe**」について話していきます。

・publishとsubscribe

publishとsubscribeについて説明していきます。まず前提として、ROSではトピックと呼ばれる通信方式を用いています。トピックというのは何かメッセージを送る上で送信者と受信者がお互いを特定しない非同期通信をする方式です。その送信のことを「**publish**」受信のことを「**subscribe**」と呼んでいます。因みに送信者を「publisher」、受信者を「subscriber」と呼んでいます。

・ROS2の通信

トピックを図で表してみました。上記でも書いたようにROS2ではトピックという通信方式を用いています。トピック名を指定することでお互い（publisherとsubscriber)を特定しない通信を実現しています。

実際にメッセージを送ってみます(図のpublisher）。

```
ros2 topic pub /chatter std_msgs/String "{data: Hello world!}"
```

ros2 topic pubというコマンドを入力します。
これはトピック形式の「**pub**」（トピックを送信する側という意味）です。
/chatter これが先に言っているトピック名の指定です。今回は「chatter」っていう名前のトピックだよーっていう感じです。
std_msgs/ String これでメッセージの型を指定しています。これを実行すると、

```
data: Hello world!
---
data: Hello world!
---
data: Hello world!
---
data: Hello world!
```

こんな感じにメッセージを送信してくれます。

```
ros2 topic echo /chatter
```

次にメッセージを受信していきます(図のsubscriber）。
ros2 topic echo送信した側と同じようにトピック形式で「**echo**」送信したメッセージを表示するというコマンドです。
/chatterでトピック名の指定です。これで「chatter」を指定することでこの名前のトピック持ってきてねーって感じです。実行すると、

```
publisher: beginning loop
publishing #1: std_msgs.msg.String(data='Hello world!')

publishing #2: std_msgs.msg.String(data='Hello world!')

publishing #3: std_msgs.msg.String(data='Hello world!')

publishing #4: std_msgs.msg.String(data='Hello world!')
```

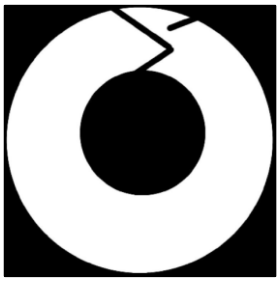
このように**pub**で送っているメッセージを表示してくれます。（これ受信か？というツッコミは禁止、受信や）
まあ、こんな感じでRobotに命令を送り「動いて！お願い！！」ってします。

...僕はあくまでDTMしたいので僕はDTMをする！！！！

来年はDTMの何かしら書きたいです。

手作りQRコード

学部/回生	お理工な学部、ロボティクス学科　1回生
名前	USA（宇佐、良いニックネームが欲しい）



　初めに断っておくが、タイトル欄にあるのはあくまで本稿のトピックスであり題名ではない。本稿の正式なタイトルは「デジタルというコンテンツ形式が”締め切りと”という概念に及ぼす心理的影響の検証および考察」である。

　検証の結果としては、読者の方々がこのページに抱いた第一印象がそれにあたる。このページを手抜きと考える読者は多かろうが、果たして本当にそうなのか、是非QRコードの先に訪れて確かめて見てほしい。考察の部分を記してある。

　情報系の内容が本題なので、断っておく。上記のQRコードはQRコードの仕組みを理解したうえでWindowsペイントでチクチクと書き上げた物だ。近年は無料でQRコードを作れる時代なので生産性の観点から言うと全くの労力と時間の無駄である。

　簡単にQRコードの制作過程を。①原稿を書く②どこかのサイトに原稿を挙げる③原稿を挙げたサイトのURLをJISを参考に8ビットに変換④1＝白，０＝黒として2×４＝８マスからなる長方形に書き直していく⑤位置補正のマーク（どのQRコードにもある正方形のこと）をまず下地に書き込む⑥右下部分に「データモード」と「URLの文字数」の宣言を意味する白黒を書き込む⑦33×33（＝1089）マスの残りの部分に”間違えないように”URLを意味する白黒の長方形を埋めていく。

　もしQRコードが手書きで書こうと思ってかける物、また、肉眼で読もうと思って読める物とご存じでなかった読者の方々がいらっしゃったら是非ググってみてほしい。わかりやすい解説の載ったサイトが多く出てくる。構造は思いのほか単純であることがおわかり頂けるだろう。　読者の方々の見識が広まるとともに、私の途方もない苦労の一端を垣間見てくだされば、これ、幸いである。

楽にVTuberになる

学部/回生	情報理工・システムアーキテクト・2回生
名前	Rennka

ねえ、ばーちゃるゆーちゅーばーなりたくない？

こんにちは、Rennkaです。最近何かと話を聞くVirtual YouTuber。そんな存在になって、「顔出しせずに配信してみたい」、なんなら「楽にできないかな」、ということでバーチャルユーチューバになってみました。

使用したもの

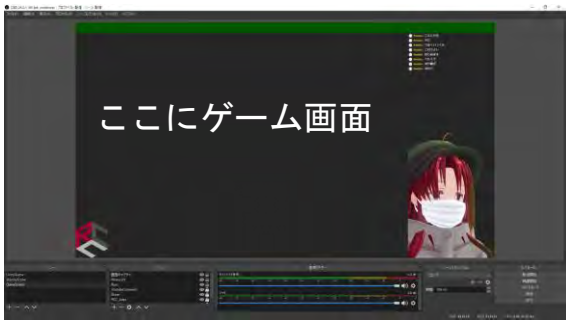
- iPhone 11
- Windows PC
- OBS
- カスタムキャスト
- Reflector 3
- YouTube

関係

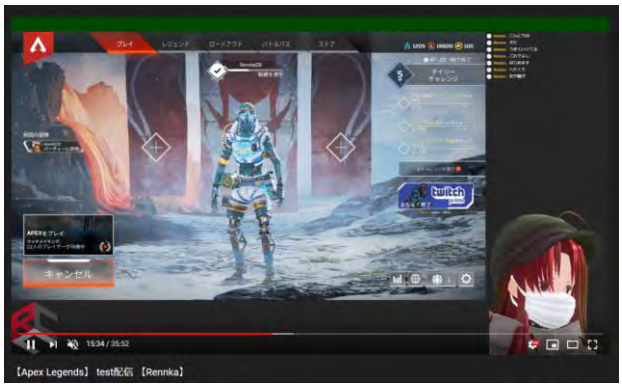
Reflector 3で
ミラーリング

引用元:https://obsproject.com/

OBSでゲーム画面・iPhone画面等
配信画面を整える



配信する



初配信 https://www.youtube.com/watch?v=4hBGMcJ8BsY

宝くじ + 機械学習で5000兆円稼ぐ

学部/回生	情報理工学部知能情報コース2回生
名前	るりと (@ruriro0125)

あらすじ

るりとです。某**Ri-one**の方で機械学習(主に強化学習)についての論文を書く必要が生じて毎日泣きながら機械学習のおべんつよをしています。さて、最近RNN(再帰型ニューラルネットワーク)を実装を通しておべんつよしていく中で私が金欠だったことを思い出しました。(学習兼ゲーム用のPCを購入したため)せっかく実装をするならお金を稼げるようなものにしたいという邪な😏に囚われた私は不幸にも宝くじに目をつけてしまいました...

今回目をつけた宝くじについて

今回はナンバーズ３という3桁の番号を予想するという宝くじの当選番号(ボックス狙い)を予測しようと思います。(簡単に実装出来そうだったので...)

過去の宝くじの情報収集について

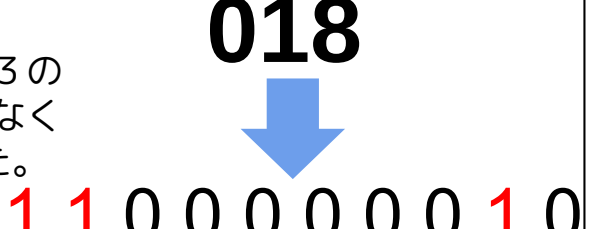
過去のデータは当然スクレイピングを用いて収集しました。み◯ほ銀行のサイトはクソだったので、楽天のサイトでやりました。今回はPythonのライブラリ"*BeautifulSoup*"を用いました。実装部分の内容は下にあるgithubのリポジトリを見ていただけると把握できると思いますのでここでは省略します。

今回用いた機械学習の理論について

あらすじにも書いていた通り今回はNNの種類の一つであるRNNを使います。NNでは入力データを独立しているものとして扱う部分を、RNNでは入力データを連続したもの(時系列)として扱います(**多分**)。今回はRNNの種類の一つであるLSTMというものを使います。LSTMはどうやら過去のデータがどれだけ今現在のデータに影響を与えるかどうかの部分が基本的なRNNとの差らしいです(勉強中なので間違っていたらTwitterの方で教えていただけると助かります)。

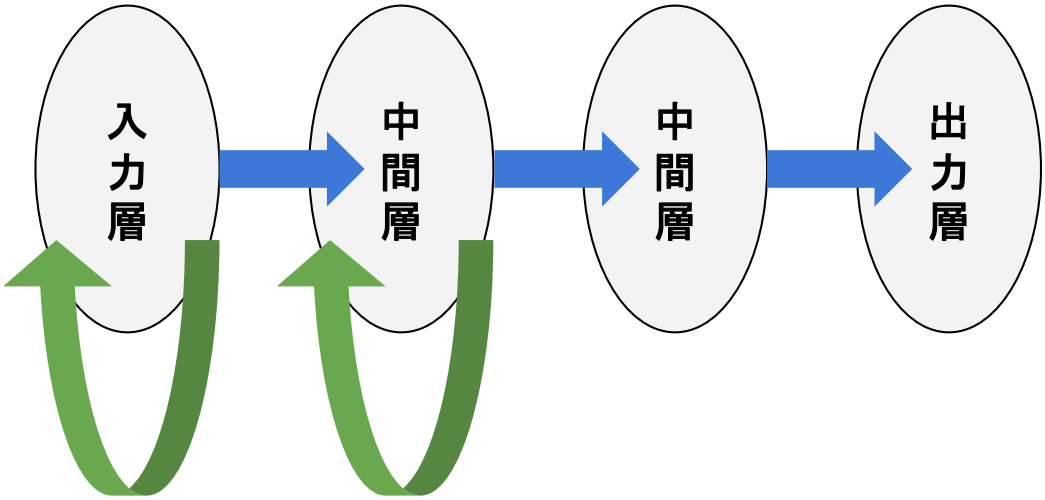
データの前処理について

今回は入力データ(過去のナンバーズ３の当選番号)を数字のまま入力するのではなく右図のように整形して入力を行いました。
また同様に正解ラベルも整形しました。
今回は数字の大きさを特徴量として捉えてほしくなかったので整形しました。



ニューラルネットの部分について

今回は、入力層(再帰)-中間層(再帰)-中間層-出力層という構成にしました。~~理由は特に無いです。~~



実装部分について

今回は*Chainer*という素晴らしいライブラリを用いて実装しました。
*Chainer*は~~猿でも書けるように~~Trainerというものが提供されています。
簡単に言いますと、Trainerに学習させたいモデルとかデータを渡してあげると自動で学習を進めてくれるというとても便利なものです。(**多分**)
ただ今回はデフォルトで提供されているTrainerだけでは物足りない(LSTMに対応が出来ない)ので一部拡張しました。

学習させた結果

誤差が発散してしまいました。ランダムよりは当選確率は高いはず...

epoch	main/loss	validation/main/loss	main/accuracy	validation/main/accuracy
10	10.3718	29.8825	0.0304054	0.0166667
20	403.427	784.571	0.0307808	0.0166667
30	3802.39	6820.96	0.0307808	0.0166667
40	18066.2	30853.7	0.0307808	0.0166667

改善したい所

1:ハイパーパラメーターのチューニング

今回のコードではいくつものハイパーパラメーターが存在します。
(活性化関数、誤差関数、過去のデータの一度辺りの使用量など)
現状はフィーリングで設定しているのでここを修正できればもう少しは役に立つ学習コードになるのかなと思います。

2:誤差関数

今回の誤差を求める部分では、単純にMSEを用いているだけでした。
しかしこれでは足りない部分があります。どの部分がどう足りないかは今後勉強して知識を身に付けていく予定です。
こんな感じでうまくいきませんでした。悲しいのでとりあえず予想番号で一度買ってみたいと思います。では、さらば。

コード置き場

https://github.com/ruritoBlogger/get-5000000000000yen

ゴムを上においたら反応しないタッチパネルがありました	
学部/回生	情報理工学部SA2回生
名前	powt

みなさんはじめまして！初めましてでない人にもはじめまして！
はい。これで自己紹介パート……終わり!! !!

それはさておき、この記事のテーマはタッチパネルです。有名？というか、よく使われている３つの種類のタッチパネルを紹介したいと思います。

１　抵抗膜式

スマホとかの画面は昔これでした。0.1mmぐらい凹んで内側の回路に流れる電流が変化することで、触れていることを検知しています。圧力で検知しているので、指とか、タッチペンでなくても操作できますが、マルチタッチは苦手です。また、傷つきやすいです、つらい。

２　静電容量式

いい名前の響きですね！特にキーボードだともっとよく聞こえます。今のスマホはこれです。パネルと、導電性のあるもの（指やタッチペン）の間で、回路を作ることで、操作を検知しています。抵抗膜式と比べればマルチタッチも得意です、傷にもまま強いです。また、厳密には画面に触れていなくても操作できることもあります。

３ 光学式

画面の上に網目状にLEDなどで光線を格子状に配し（リアル見えない網戸!）、光が反対側に届かなくなる＝そこに障害物がある＝タッチされている。と云う風に感知しています。LEDの寿命までずっと使えたり、何でも（導電性がなくても）操作できたり、画面に機械を埋め込んでいるわけではないので、画面がくすむこともなく使えたりする利点があります。しかしながら、致命的なことに画面上で感知しているわけではないので、操作者が触れている位置と機器が感知する場所が視差によりずれます。

まとめ？
タイトルの疑問？を解決するために調べていたわけなんですが、こんなふうになってるとは思っていませんでした。（小並感）タッチパネルを使うときはこの辺のことも考慮に入れたいですね。

ｐｓ．タイトルのやつは３でした

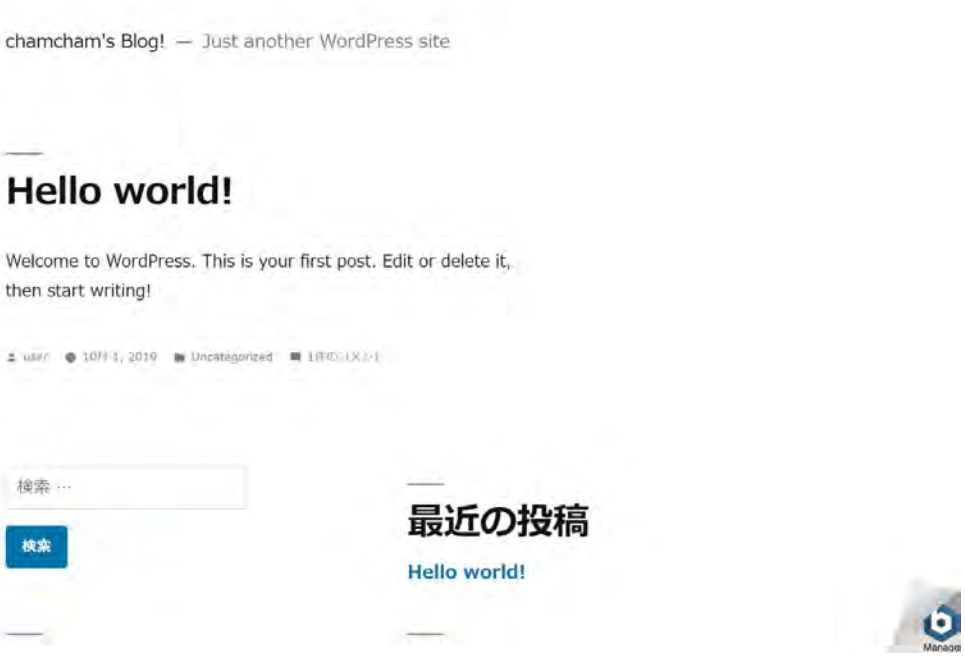
お手軽WordPressの環境構築

学部/回生	情報理工学部SA2回生
名前	ﾁｬﾑﾁｬﾑ

こんにちは、ﾁｬﾑﾁｬﾑです。
RCCの中で広報とかをしている渉外局というところに属しています。
会公式Webサイトで記事を投稿する際にWordPressを使うのですが、つい最近、記事を投稿するだけで自分はWordPressについて何も知らないな…ということに気がつき絶望しました。
絶望してもどうしようもないので、一から、まず環境設定からやってみようと思って、大昔に、某緑のプログラマのための技術情報共有サービスで見つけた記事を参考にしてWordPressの環境構築をしてみました。

工程
クラウドサーバーにGCPのものを使用し、WordPressのコンテナをデプロイして、あと色々しました(雑)。他には途中でお名前.comでドメインをとって、他はほぼ参考サイト通りにできました！

やったー!生まれたてのWordPressだー!☆✨

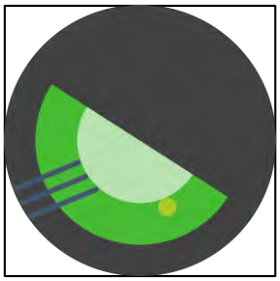


せっかく環境構築ができたので、これからがんばってWordPressのお勉強をしていこうと思います！

参考サイト
「【※2020/1/1より約300円/月が有料になります】1時間で出来る！最強のWordPress環境構築(永久無料※2019/12/31まで)」
https://qiita.com/ryuta69/items/dbb0db5cf7099b7a7cc4

お前のフロント開発議論は間違っている

学部/回生	情報理工 画像・音メディアコース / 2回生
名前	shuta



※突然の 煽りに注意 最上川

shutaです。近況はデザイナーしつつフロントエンド(SPA開発・Webアプリケーション)を開発してて、主に開発にはフレームワークにVue/Nuxt、React、言語にはJavaScript、TypeScript(JSX、TSX)を用いています。マークアップも書きます。デザインはFigma。よくお尻が燃えています🔥、気づいたら消してやって下さい(気づかないので)
デザイナーやってるくせにこんなギッチギチに字詰めのもの書いてます。これが人生ってやつです、１回生はよく見ておきなさい。ところで、

「お前のフロント開発議論は間違っている」

です。急に否定で？？？だと思います。(僕も急に否定されたら、？？？なので)ある話題を話したいがためだけに断腸の思いで煽っています...。
話に入るんですが、最近だと TSで頑張る頑張らない とか CSSの設計どうするか? とか WebpackやParcelで、ウオウオするパクよ～(死) みたいな議論を観測している範囲でよく見かけます。楽しそう。でもね

「ユーザーが使ってて気持ちいいUIを作るには」

みたいなのを詰めて議論してる場面を全然見かけることがないです。
どうしてもUIの作り込みとかパフォーマンス改善の部分は、実装段階の中でも最低限完成すれば後回しにされがちだったり、便利なうぇっぶふれーむわーくに任せたら良い、みたいな雰囲気なのでしかたのないアレはありますが、ちゃんと話したいよね～~~そういう話をしないお前らは間違っとる、何をやってもダメ~~という感じのモチベで記事を書いてます。
ここから具体的な内容に入るのですが、フロントエンドでアニメーションと言うと思い浮かぶのがcss、JavaScript(のライブラリ)、HTML5のCanvas、あとはWebGLなどという感じです。今回はCSSとJSでのアニメーション、特に「どっちでアニメーションしていくのが良いのか」に照準をあわせていきます。
結論からいくと両方うまく使おうね♡です。(なんか議論の腰を折ったみたいでごめんなさい)、では早速いきます

- cssのみでアニメーション

処理にもよるんですが(filter: blur など処理めちゃ重い)、だいたいJSよりも圧倒的に早く書けます。ファイルの大きさとか見ればわかります。じゃあ待ちの時間が少なくなるので快適なUIが提供できてうれしくなるから、全部cssでかいちゃお、になりそうなんですが、通常のcssの欠点として

- エラー検出が厳しすぎる
- スコープがグローバルで何かと困る

みたいなのがあって、linterとかcssのお友達(stylus、sassなど)で殴ればマシになるんですがそれでもやっぱりアレです。つらくかなしい。

- JavaScriptのみ(ライブラリ使用)

まず、よほどのことが無い限りクラスのつけ外しとかでのアニメーションを生のDOM操作でやるのはパフォーマンス的にもコード書く側のお気持ち的にもやめとけという感じです。本当に何も生まれない(クソコードなら生まれますが)のでやめるんだ、悪いことはいわないので今すぐそのjQueryをおろしなさい(手遅れ～～～～～オアアアアアアアアアアアアアア～～～～♡♡♡♡♡♡♡♡)みたいなアレをちょくちょく見たのでやっぱりDOM生で扱ってる感が少ないライブラリ、フレームワークを使用するのが賢明です。個人的にはGSAPのTweenMaxがおすすめです。書き方もすごいシンプル(cssのkeyframesに似ている)で分かりやすいかつ、requestAnimationFrameでアニメーションするので体験が良い、VueとかReactにもバシバシ入れてTS化してもちゃんと型がついてて超うれしくなっちゃうな～になります。JSだとスコープの管理がまだcssよりしやすくて実行時エラー検出出来るので、まあ、コードの間違いには気づけます。cssでやるよりも精神衛生的には幾分いいです。パフォーマンスとかさえどうにかなるなら僕もJSでやっていきというモチベでやっています。

- css、JavaScriptの併用

そんな感じで結局人間は何でも中間を好むという感じになっちゃいました。日頃考えているのは

cssだけで解決できそうだな...と思ったらcssでやってそれ以外JS

です。どういうことかというと、ホバーで発火するアニメーションだけならcssのhoverでよいしょ、よいしょしたら読み込み軽いし実装も楽に終わるけど、mouseenter・mouseleaveとかでアニメーション変えるならJS(Vue、Reactならちゃんと用意されてる)でやらざるを得ない(cssでやれるかもしれないですが自発的に知りたくもないです、複雑なので)みたいな感じとか、１つのイベントで発火してそこから繋がって複数アニメーションする的な実装であれば、パフォーマンス気にするならkeyframesで頑張ってみるが、基本TweenMaxとかのJSでやる みたいなやり方です。場面によって使い分けるが当たり前ですが良いです。なぜならSexyに解決出来るので。

あとは使えるならVueとかReactユーザーなら、そのライブラリに生えているアニメーション向けのオブジェクトとかをじゃんじゃん使用してくのがいいです。例えばページ遷移向けアニメーションならDOMレンダリングのライフサイクルに無理にねじ込むとかじゃなくて(別に細かい発火タイミング調整したい、とか手の運動したいとかなら良いですが)、Vueのtransition使うとか。アレはvueのscoped cssと相性◎でかなり便利です。Reactはreact-transition-groupとかその他ライブラリでウオーしてるのが日常になります。フロントに**銀の弾丸はない**(これが言いたかっただけ)ので色々探して :yatteiki: をしましょう。ありがとうございました。

「奥」の世界と仮想現実

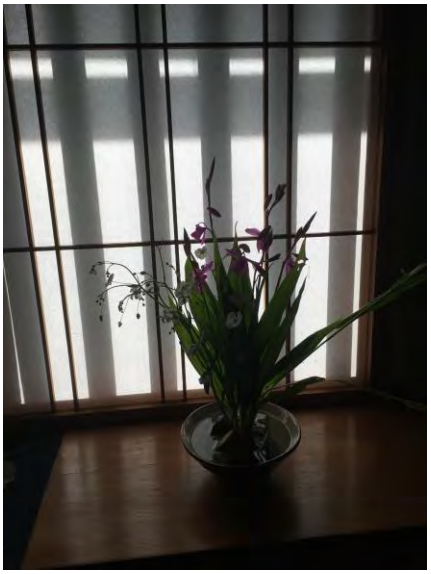
学部/回生	情報理工学部実世界情報コース2回生
名前	坪倉奏太(@ukstbkrs)

技術的なことについてはみなさんが書かれていると推測されるので，私は他の観点から一筆しようと思います．最近，xRなどと現実世界を再現，拡張する技術が持て囃されていることは読者諸兄が一番把握されていることかと存じますが，なるほど紹介動画等を見てみればその高度さには驚かされるものがあります．

しかしながら，製品や作品だけではなく，xRという技術全体を俯瞰してみると「ちょっと物足りないな」と感じることが少なくありません．つまり，「見た目等の幾何学的要素にのみ拘っていて，その表面の深層－“奥”の世界－を扱う視点」がどうも欠落しているように思えてならないのです．では，「奥」の世界とは，「奥」の世界を扱うとはどういうことかと言いますと，我々が普段見ているところの「表」の世界と対になるもので，その眼には見えない背景，「表」の裏に佇むなにものかを考える，ということです．仏教の用語で言えば「縁起」といったものになるでしょうか．

この「奥」の世界を扱った事例は，私達が住む日本に豊富に存在しています．例えば，有名な神社は大抵森の中にありますが，この森は「奥」の世界を私達により想起させ，「現実との境界である鳥居を超えれば，あちら側は神々の住まう世界である」と印象付ける効果を狙っているものだと思います．灯りのない日本家屋の床の間に生けられた花を見るとき，何か吸い寄せられるような心持ちがすることも「奥」の世界を感じているものだと言えます．床の間の濃い陰に花が存在することにより，「奥」の世界をより深くさせているのです．

これらの例のように，言語化し尽くせない，実際に体感しないとわからない妙味がある，それが「奥」の世界を扱う表現の特徴でしょう．



確かに，今のxR技術のように物体の表面的要素のみを再現することのみでも，Virtualな空間は現実世界に限りなく近づくでしょう．ただし，それは何時までも「奥」の世界にまでは到達はしないと思います．計算機の構造から考えると不可能だとも言えます．

ただ，私達の先人は言語化できない，言語で表し尽くせないものを言語で，絵画で，建築で表そうと努力を重ねてきました．そうして創られた作品には今日まで残る傑作が多いのは言うまでもないでしょう．「不可能だからやらない」のではなく，敢えてそれを計算機で表そうとする，これが後世に向けて私達のやるべきことだと思います．

感想等もしあればTwiiterにて言ってくだされば．

参考文献:
水津一郎「景観の深層」 地人書房
谷崎潤一郎「陰翳礼讃」 新潮文庫
加藤周一「日本文化における時間と空間」 岩波書店

Pythonかけない

学部/回生	情報理工学部・実世界情報コース・2回生
名前	ひなたくん

8月アメリカ行ってきました。留学です。いいですね。ボストンとかいってそこにあったMonster全部飲みました。Mango locoが一番うまいです。

ところで留学先で色々適当に過ごしてたんですが、なんかtwitterBOT作れって言われたのでそれのことを適当に。
とりあえずBOT作れぇぇぇあああああああああ！！！！！って言われたときに「弊」って言いながらやったことなかったんで適当ググってたらコード書かなくてもいいやつ何個かあったのでそれツカウカーになってたら軒並み使えなくなってたんですよねぇｶﾅｼｱ。まぁ色々理由はあるんで仕方ないとは思いますが。結局python3.7.3とtweepy3.8.0でやりました。
一応やってって言われたこととしては任意のTL見て、特定の単語拾って、urlなり言葉なりを送って〜ってことだったので適当に書いてました。でちょっとうろ覚えなんですけど、tweepyの事情かなんかで単語で引っ張ってきた人の返信するためのID(tweepyの引っ張ってくるIDは同名のがあったりで割とたるい)が関数でひっぱれなくなってたんですよ。それで、写真見たらわかるんですけど、

```
huga = [x.strip() for x
```

まぁ、こんな感じで続けていって迫真正規表現してやるしかなかったんですよ...これなんですけどtweepyの記事みてたら全然3.8のがないんで、多分こうするしかないんかなぁ...ってなりました(教えて偉い人)。
tweepy事情で少し面倒だったのはこれくらいです。あと僕の知見なんですが、unixはcronじゃなくてlaunchctl使うらしいです。これ全く知らなくて結構沼ったんでメモ感覚で書いておきます。まぁこんな感じでpythonかけなくても好きなtwitterBOT作れるのでみなさんも作りましょう！あとtwitterAPI通すのがちょっと面倒です。500字かなんか忘れたんですけど、英文を適当にかかないといけないです。

ラズパイを求めて

学部/回生	情報理工学部　SNコース　2回生
名前	鉛筆騎士（@ArthurPencilgon）

「議事録、取るの面倒だなー」
そう思ったことはありませんか？ない？あらそう……僕はありました。
という訳で**「議事録自動生成機」**を作ってみました。

〈用意するもの〉
・お好みのラズパイ
・AS-289R2プリンタシールド
・プリンタシールド用電源ACアダプター（5V4A）
・感熱紙
・USBマイク

〈手順〉
1．ラズパイのTXとプリンタのRX、お互いのGNDを接続する。
2．USBマイクをラズパイにセットする。
3．感熱紙をプリンタシールドにセットする。
4．ラズパイのソフトウェアをいい感じに設定する。
5．Google Cloud Speech APIを使って音声認識を行う用意をする。（環境変数をチョメチョメしてPythonで動かす）
6．完成

と、こんな感じで結構適当ですが、議事録自動生成機ができました。やったぁ！と思って写真を撮ろうとしたら、**会誌執筆の2日前に携帯と2万円したプリンタシールドが壊れました。**
悲しいね。

実物はこんな感じです（イメージ）
参考：https://fabcross.jp/category/make/sorepi/20180731_receipt_01.html

Kotlinの安全

学部/回生	情報理工学部情報理工学部SNコース2回生
名前	たちかわ(takion)

こんにちは、今期も1限が沢山あって健康的な生活を過ごせるたちかわです。今回の会誌ではプログラミング言語Kotlinの特徴の１つでもあるnull安全について書きたいと思います。

←このロゴマークはやかんらしいです。

Kotlinとは？

KotlinとはJetbrain社が開発したJVM言語のことです。JVM言語なのでJava Virtual Machineで動きます。
Kotlinは2017年のGoogle I/OでAndroidの公式開発言語になり2019年のGoogle I/Oで推奨言語になりました。
また、Kotlinの設計上のコンセプトとして実用主義、簡潔、安全、相互運用性などがあります。

null非許容型

Kotlinは型名だけの変数にnullを代入することは許容されていません。

null許容型

そこでKotlinでは型名の後に?を付けることで変数がnull許容型として扱われるようになりnullを代入することができるようになります。

```
var name : String? = "RCC"
name = null
println(name) //実行結界はnullとなります。
```

しかし、null許容型の変数はnullが代入されている可能性があるのでKotlinではNullPointerExceptionを防ぐためにメソッドやプロパティなどアクセスするとコンパイルエラーになります。

スマートキャスト

Kotlinでは変数の型のチェックを行うことでその変数の型を自動で変換してくれるスマートキャストという機能があります。
スマートキャストによってnull非許容型として変換されたので、.lengthを呼び出すことができるようになりました。

```
val name : String? = "RCC"
if(name != null){
    println(name.length)
    //実行結果が3になります。
}
```

安全呼び出し

ただコードを書くときに何度も型のチェックを行うのは手間が掛るので
Kotlinには安全呼び出し演算子と呼ばれるものが用意されています。
安全呼び出し演算子はプロパティやメソッドにアクセスする際に"."記号
のの前に"?"を付けることでnullのチェックとメソッドなどへのアクセス
を同時に行ってくれます。もしnullだった場合はnullが返されます。

```
val name : String? = "RCC"
val animal : String? = null
println(name?.length)//出力結果は3になります。
println(animal?.length)//出力結果はnullになります。
```

今まで紹介してきたこれらのKotlinの仕様は変数や引数がnullの可能性が
あるならば危険と判断してメソッドなどへのアクセスを制限しています。
そしてnullでないことが明示的な変数だけを操作できるようにすることで
null安全を実現しています。他にもKotlinには魅力的な言語仕様がたくさ
ん存在するので皆さんもぜひKotlinに触れてみてください。

ロゴの引用元
https://en.wikipedia.org/w/index.php?title=File:Kotlin-logo.png

ヘッドホンのすゝめ

学部/回生	情報理工学部 SAコース 2回生
名前	髙山

こんにちは、髙山です。最近かっこいいEDM作りに奮闘していますが、全然できません。辛いです。今回購入したヘッドホンがとても良かったので、私がヘッドホンをどのように選んだのかと実際に購入したV-MODAのCrossfade II Wireless Value Editionというヘッドホンを紹介します。

1 ヘッドホン選び

1.1 使用目的を明確にした

ヘッドホンは製品によって個性があるのでどのような用途で使うのか、何を重視したいかを明確にすることでより自分にあったものを選べました。今回の用途はDJをする時と通学時に聴くことでした。最も重視したことは低音がパワフルであること、DJをする時に落ちないようなつくりになっていることでした。

1.2 音以外で選ぶ基準にした項目

次にヘッドホンを選ぶ際に音以外で私が見た項目を4つ紹介します。音の味付け具合は本当にヘッドホンによって違いが出るので店頭で聴くことを大事にしています。

1.2.1 開放型・密閉型

開放型は高音の伸びが良く広い音場感が出せるのが特徴です。音漏れが非常に大きいので外で使うのには向いていません。逆に密閉型は音漏れがしにくい分高音が出しにくいという欠点があります。私はDJする時と通学時に使いたかったので密閉型にしました。

1.2.2 インピーダンス・音圧感度

ポータブルオーディオプレイヤーに繋いで聴く予定のある方は音圧感度とインピーダンスに注意してください。音圧感度が小さかったりインピーダンスが高すぎると、再生音量が小さくなってしまい十分な音量を得ることができないときがあります。ただインピーダンスが高いとノイズの少ない良い音になるので, 良い音で聴きたい方はヘッドホンアンプを使用するのも一つの手としてあります。スマートフォンで再生する予定でしたので、インピーダンスはそんなに大きなものを選ばないようにしました。

1.2.3 Bluetooth機能

Bluetooth機能を備えているかも重要です。Bluetooth機能があれば外で使いやすく利便性が上がりますが、その分有線での音質が下がってしまったり値段が高くなったりします。通学時に使用するので有線だと邪魔になるかなと思いBluetooth機能が備わったものを選びました。

1.2.4 側圧

ヘッドホンの側圧が強いと頭から落ちにくくなりますがその分長時間身につけていると頭や耳が痛くなってしまいます。DJ用ヘッドホンは側圧が強いものが多いです。ティッシュ箱に挟むことで側圧を弱める事もできるみたいです。DJをする時動いたりして落ちる危険性があるので側圧が強いものにしました。

2 Crossfade II Wireless Value Editionは最高

V-MODAのヘッドホンは低音域がパワフルで装着感も強くちょっと動いたくらいでは落ちないような設計になっているので, DJから支持を得ています.
このヘッドホンは簡単に言うと多くのDJに愛用されているモデルCrossfade M-100にBluetooth機能を加えたモデルになっています。
低音域がパワフルで、中音域も高音域もクリアに聴こえます。高音域の音が耳に刺さるなんてことは無いです。臨場感がすごいです。ダンスミュージックだけでなくポップスやロックもこれで楽しむことができます。ハイレゾにも対応してます。
ワイヤレスに関してですが、有線状態で一番良い音が鳴るように設計されているため、普段使いはワイヤレスで手軽に音楽を楽しみ、家ではアンプなどに繋いで最高の音質と臨場感を堪能することもできるんです！
ああ、最高...！14時間駆動可能なので出先で充電が切れる心配がない上、折り畳むことができるので持ち運びにも便利でデザインも良いです。
最高以外の何物でもないです！

皆さんも良きヘッドホンライフを！

Ren’pyって知ってる！？

学部/回生	情報理工学部 SAコース 2回生
名前	uMa

0. 自己紹介

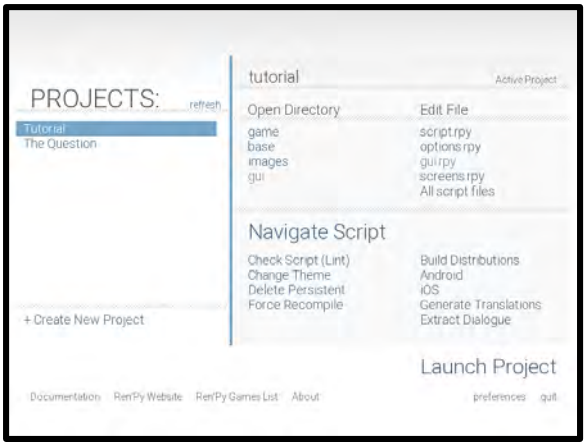
RCC所属 情報理工学部 システムアーキテクト(SA)コース所属のuMaです. 情報系のことを勉強し始めたのは大学からで, やりたいことも明確には決まっていないのでまだまだ知識は浅いですが, 色々なことに挑戦して日々邁進しております...

1. Ren’pyって知ってる!?

ところで, Ren’pyって知ってる!?(唐突) 自己紹介の中で述べたように, 私は色々なことに挑戦していますが, その中の1つにゲーム制作というものがあります. ゲーム制作する上で, 色々なことを学べますし, 成果物も残るので, 割と楽しんで取り組んでいます. ゲームにも色々とジャンルはありますが, 今回お話しするRen’pyは, ビジュアルノベルゲームを簡単に制作することができるソフトウェアです. 比較的簡単にハイクオリティのゲームを制作できるので, かなりおすすめです.

2. じゃあ, 使ってみましょう?

公式サイト(https://www.renpy.org) のDownload Latest Versionからダウンロードします. 後はダウンロードしたフォルダの中にあるファイルを実行するだけです. Ren’pyは, Windows, MacOS, Linuxに対応していますので, それぞれにあったファイルを実行してください.(Windowsはrenpy.exe MacOSはrenpy Linuxはrenpy.sh) すると, 下のような画面が開きますので, そこからプロジェクトを作成したり, 編集したり, プレイしたりすることができます.



3. どうやって使うの?

それでは実際にRen'pyを使ってみましょう. といっても限られたページ内で全てを説明することは出来ませんので, どうやってRen'pyの使い方を学習すれば良いのか, その方法について記述したいと思います.

- **公式のドキュメントを見る**
  実はここまでで説明した導入の方法なども全てここに記述されています.
  Ren'pyの素晴らしいところは, 日本語の公式Documentも存在する所です.
  ドキュメントを読むことで, 基本的な使い方から発展的な使い方までを学習することができるので, 興味があれば, ダウンロードする前に読んでみるのも良いでしょう.

- **チュートリアルをプレイする**
  Ren'pyには, チュートリアルが用意されています. チュートリアルはノベルゲーム形式で用意されており, もちろんRen'pyを用いて制作されています. これも非常に分かりやすく使用方法を学ぶにはおすすめの方法ですが, 英語での紹介になっており, 日本語版は(おそらく)用意されていません.

4. まとめ

ここまでで紹介したように, Ren'pyは導入するのも非常に簡単であり, pythonのスクリプトを埋め込めたり, プロジェクトフォルダ内の画像を変更したりすることでUI部分をカスタマイズできたりするので, 初心者でも上級者でも使える優れたソフトウェアであると言えるでしょう. 公式のドキュメントや, チュートリアルも非常に充実しているため, もしノベルゲームの作成に興味があれば, 利用してみるのも良いかもしれません.

5. おまけ

- **Ren'pyで制作されたゲーム**
  Ren'pyは英語圏で多く用いられており, 有名なノベルゲームがRen'pyによって作成され, 公開されています. その中でも特に日本でも有名なものに「**Doki Doki Literature Club!**」というものもあります. あれ？この名前, どこかで…？
- **その他のノベルゲーム制作ツール**
  もちろんRen'py以外にも様々なノベルゲーム制作ツールが存在します. 例えば, 「NScripter」や「TyranoScript」という日本人の方が制作したものがあり, これによって制作された有名な作品もいくつかあります. 気になった方は是非調べてみてください!

unityでTCP通信

学部/回生	情報理工学部 SAコース 2回生
名前	gyuho shilouchi (城内牛歩) @rccigyuhone

unityでTCP通信をしてみました.
unity to unity であればUDPで十分ですが，今回はサーバー経由で通信を行うかつ，ストリーミングのような実装を求めたため，無理やり繋げました.
まあ，今回が特殊なだけで僕は普段photonをガンガン利用しています.

まず前提知識ですが，unityではTCPListenerとTCPClientというクラスを扱うことができます．（C#のSystem.Netからusing）
なので，通常のC#でのTCP通信と同様に行うことで，unity上でTCP通信を実装することができます.
しかし，unityは通常のC#とは少し異なった，ゲームを作成することを前提とした俗称unityC#という言語を用います．（UnityEngineをusingしているだけですが）
なので，配信の受け手側の実装を気を付けて行わなければありません．頑張りましょう．頑張りました.

では実装に移ります.
クライアント側です.
まず，TcpClientという便利な関数を使用してサーバーと接続します.
接続先は，以下のように確認することができます.

```
//TcpClientを作成し、サーバーと接続する
tcp = new TcpClient(ipaddr, port);
Debug.Log("サーバー( " + ((IPEndPoint)tcp.Client.RemoteEndPoint).Address + " : " +
    ((IPEndPoint)tcp.Client.RemoteEndPoint).Port + " )と接続しました( " +
    ((IPEndPoint)tcp.Client.LocalEndPoint).Address + " : " +
    ((IPEndPoint)tcp.Client.LocalEndPoint).Port + " )。");
```

次に、サーバー側の実装です.
クライアント側と同じような要領でできます.
まず，TcpListenerという便利な関数を使用してクライアントの接続を待ちます.
接続先は、以下のように確認できます

```
// TCPListnerを作成し、クライアントを待ち受ける
tcpListener = new TcpListener(IPAddress.Any, 8052);
tcpListener.Start();
Debug.Log("Server is listening");
Debug.Log("Connect: " + connectedTcpClient.Client.RemoteEndPoint);
```

今回の実装ではサーバー側，クライアント側の実装にあまり違いを持たせず，送信部と受信部に似たような実装をしています.
これより，それぞれに関して記述したいと思います．まずは受信部です.
接続が完了したら、stream objectを取得し、byte配列を待ち受けます.
このとき，取得を待ち受けるのはbytesなので、宣言しておいてください.
ちなみにここでは1024ぐらいの長さで宣言してあります．ただ，1024を超えた場合，while文の判定がfalseになるので，出力は最後まで行われます.

```
// stream objectを取得する
using (NetworkStream stream = socketConnection.GetStream())
{
    int length;
    // 飛んでくるbyte配列をちょっとずつ取得する
    while ((length = stream.Read(bytes, 0, bytes.Length)) != 0)
    {
        var incommingData = new byte[length];
        Array.Copy(bytes, 0, incommingData, 0, length);
        // とりあえずstringに変換して表示する
        string serverMessage = Encoding.ASCII.GetString(incommingData);
        Debug.Log("server message received as: " + serverMessage);
    }
}
```

次に送信部です．
送信の場合も，受診時と同じようにstream objectを取得します．
その後，送信したい文字列(文字列でなくても大丈夫。例えばintとか。あとはjson形式の文字列をstring形式にして送信したりします。)をbyte配列に変換し，書き込みます．

```
// 書き込みのためにstreamを確保する
NetworkStream stream = connectedTcpClient.GetStream();
if (stream.CanWrite)
{
  string serverMessage = "This is a message from your server.";
  // メッセージをbyte配列に変換する
  byte[] serverMessageAsByteArray = Encoding.ASCII.GetBytes(serverMessage);
  // streamに書き込む
  stream.Write(serverMessageAsByteArray, 0, serverMessageAsByteArray.Length);
  Debug.Log("Server sent his message - should be received by client");
}
```

これで終わろうと思ったのですが，書き忘れてたことがありました。これだけだったら普通に.NETのリファレンスを見れば一発でかけたんですが，Unityに持ってこうとするとこれだけでは足りません．なぜならメインスレッドで動くことになってるからです．これでは受け取りだけしてUnityに反映できませんね．そこで，スレッドを分けてあげる必要があります．(コルーチンでもいいですが，スレッド分けることをおすすめしたいです．)

```
// バックグラウンドスレッドで実行する
tcpListenerThread = new Thread(new ThreadStart(ListenForIncommingRequests));
tcpListenerThread.IsBackground = true;
tcpListenerThread.Start();
```

これで，本当に終わりです！
受信，送信をバックグラウンドスレッドで行わないといけないのは一番気をつけないといけないことですね．ちなみにコルーチンはメインスレッドでなければ動かせません．そしてUnityのメインスレッドの邪魔をしてはいけません．
これを，送信部・受信部や接続部ごとに関数にして動かせばしっかりと動きます．詳しい実装は https://github.com/flaireclair/UnityTCPSocket をご覧ください．
ありがとうございました．

Arch Linux install guide

学部/回生	情報理工学部・SNコース2回生
名前	3^5

こんにちは、3^5です。前書きでもお会いしたのでまたお前かという感じですね。この一年で初心者には難しいと言われている**Arch Linux**というOSをインストールしたり飛ばしたりしてきたのでその経験から得た知見を記します。(追記：2019年10月6日にbaseグループがbaseパッケージに置き換えられた件を書き足しています。)

僕がいつもやっているインストールの手順はこんな感じです。

1. インストール用のライブUSBを作成する
2. USBから起動してパーティションなどの設定をする
3. ベースシステムをインストール
4. 諸々の設定をする
5. GUI環境を構築する

順に作業の概要と躓いたところなどを説明していきます。

1のUSBの作成ですがこれには罠があって、Ubuntuのライブ USB作成によく使われるUnetbootinを使って作成するとブートローダーの設定が上書きされて失敗するので注意が必要です。Linuxを使っているなら**ddコマンド**(の使い方を勉強する気持ち)でやりましょう。

～ライブUSBを作るのに困らない程度のddコマンドの使い方～

引数はこの3つ。

- bs= (block size) 一度に読み書きするデータ量(byte)
- if (input file) 標準入力の代わりに読み込むファイル
- of (output file) 標準出力の代わりに書き出すファイル

実行例はこんな感じ。

```
dd bs=4M if=./archlinux.iso of=/dev/sdx
```

ddが直接やるような作業、とても危険なので普通は管理者権限がいります。ddはdata destroyerの略ともいう。変なところに書き込むと取り返しがつかないのでofは最後に書こうね。

2ではHDDやSSDのパーティション分けを行いフォーマットする作業を行うのですが、僕はここでWindows Boot Managerの入ったパーティションを、フォーマットする予定のパーティションと間違えてしまったため、Windowsのデータがすべて消失するという事故を起こしました。どのパーティションに何を入れているかはちゃんと覚えておきましょう。また、Windows Boot Managerは基本的にsda1に配置されるということにも注意。(Windowsはこれだから困る)

3のインストールは特に躓いたことはないので、皆さん大丈夫だと思います。

と書こうと思っていたのですが、ちょうど最近、事情が変わったのでいろいろ説明をしておきたいと思います。
3はpacstrapというスクリプトを使って、Arch Linuxを使うに当たって必要なパッケージをインストールするという作業です。
今まではマウント先にbaseパッケージグループ(最小限の動作に必要であるLinuxカーネルと、いくつかのパッケージ)を指定すればよかったのですが、ちょうどこのコラムを書き終わったくらいにbaseグループが新しくbaseパッケージとして1つのパッケージにまとめられてしまったために指定すべきパッケージが大きく増えました。
インストール時のコマンドは大体こんな感じに変わります。
　今まで：`pacstrap /mnt base`
　これから：`pacstrap /mnt base` **linux** ...(他もあるけど最低限この2つ)
古い記事ばっか見てlinuxパッケージ入れ忘れんなよ、ということです。
他にも、linux-firmwareパッケージやnanoエディタなどもbaseからは除外されているので必要に応じて指定してあげる必要があります。

参考：baseパッケージに置き換えられて除外されたパッケージ
cryptsetup device-mapper dhcpcd e2fsprogs findutils gawk gettext inetutils jfsutils linux-firmware logrotate lvm2 man-db man-pages mdadm nano netctl reiserfsprogs s-nail sysfsutils systemd-sysvcompat texinfo usbutils xfsprogs

この件についての参照記事:
Arch Linux の base グループがパッケージに置き換えられた話
https://qiita.com/suzukeno/items/a978b2de8a34b9a91c95

　4では文字通りいろんな設定をしますが、とくに**ブートローダーのインストール**あたりが選択肢がいろいろあって困りやすいところかと思われます。Arch LinuxではsystemdとGRUBの2つが人気のあるブートローダーですが、個人的にはsystemdの方が設定が簡単で追加のパッケージインストールも必要ないのでおすすめです。
　5のGUI環境は僕は一通り構築できるようになるまで数ヶ月かかったのですが、選択肢が非常に多い上に**X Window System**についての知識も必要になってきたりするので少し難しいです。僕はManjaro Linuxというインストールが非常に簡単なArch Linux派生のOSをインストールし、GUI環境の完成された設定を一度見たおかげで設定ができるようになりました。この勉強法結構おすすめです。
とりあえず使えたらそれでいいという人はgnomeなんか入れたらすぐ使えますよ、Ubuntuと変わらんけど。

この一年で躓いた箇所はこんな感じです。解決法とか参考になればと思います。散々沼った僕から言えるのは、**Arch Wikiをとにかく読め**というお話ですね。以上、3^5でした。

かしこい鯖の飼い方

学部/回生	情報理工学部SNコース 2回生
名前	ねこぱんだ @hirotta3a

今や一家に二台と言われているサーバ機
でも、新品は値段が高くて飼えない…

そんな貴方に中古サーバ機の飼い方を伝授します

ヤフオク！

Xeon 64gb	検索

サーバ機御用達CPUのIntel Xeon搭載、RAM64GBのラックマウントサーバや、タワー型ワークステーションが出てきます
RAMの値は適宜変えてください
ラックサーバを飼う場合は大人しくラックに載せるか、壁に立て掛けましょう
ワークステーションはサーバ用マシンと比べて電源周りなどの冗長性に欠ける面がありますが、スペック的には十分利用可能です

10万～20万円の商品がほとんどですが、ときどき3～5万円程度のものが出ています
中古業者の営業が始まる休日明け、企業の決算前などの時期を狙って探すと良品が多いです

CPUやマシン自体の型番を検索しながら絞り込みましょう

ただ、オークションなのでうっかりしていると3万で買うつもりが6万円で落札してしまったりするので気をつけなければなりません

僕は→のHP z620を6万円で落札しました
(ワークステーションですが…)

- Xeon R E5-2667 ×2
- RAM 64GB
- HDD 2TB×2

楽しい自宅サーバ生活を！！

GASで会内ツール作った話

学部/回生	情報理工学部 画像・音メディアコース　3回生
名前	あいてにあむ

幣会内で、
「回答すべきフォームが多すぎてわかりにくい」
との声を受け、
Google Apps Scriptで
フォーム一覧と自分の回答状況を確認できるWebアプリ的なやつを作りました。
スマホでもきれいに表示されるようにしています。
幣会ではGoogleフォームを利用しており、GASを用いることでフォームの回答内容にアクセスし、未回答か回答済かを調べるようにしています。
Web上での公開は、GASのHtmlServiceを利用しています。

いかたこうなぎを描く♪

学部/回生	理工学部 数理科学科 3回生
名前	nagin → unagi

なんかいぷろぐらみんぐげんご 『Piet』

遡ること約２年，まだC言語がさっぱり理解できなかったあの頃…
某勉強会にて出会った衝撃的なプログラミング言語『Piet』をほんの少し触れさっぱり分からず．そして今日，今度こそお絵描きをしてみようと試みました．

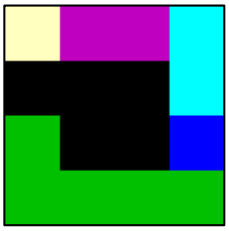
＊ Pietの特徴

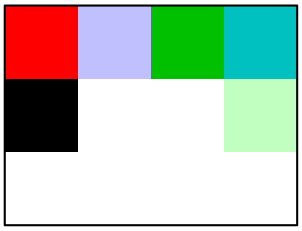
- ドット絵でプログラミングが可能　（最大の特徴）
- ドットの色の差で命令を記述　(プログラミングするということ)
- **「プログラムを終了させること」**がとにかく難しい
- 同じプログラムでも，色んな作り方が可能

＊ 今回挑戦した作品

- 標準入力から整数を２つ受け取り，足し算するプログラム

１．本格的なお絵描きの前に，まずはプログラミングをする．簡単．



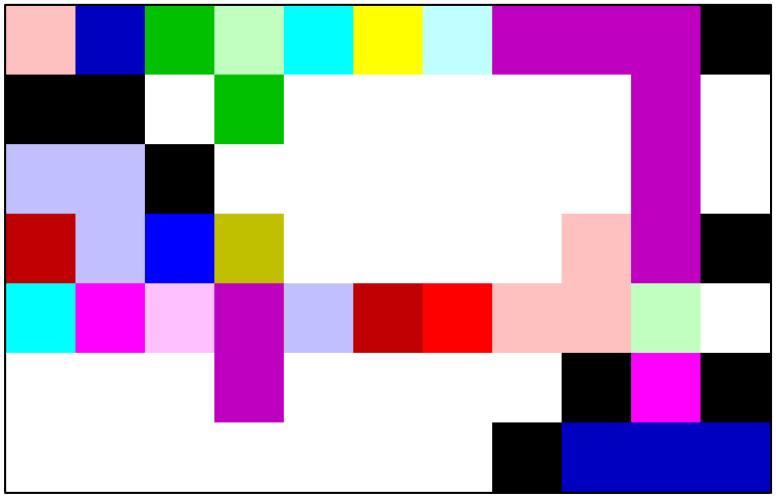


２．１.を用いてお絵描きに挑戦！ モデル → ∈(・◎・)∋ ！(unagi)

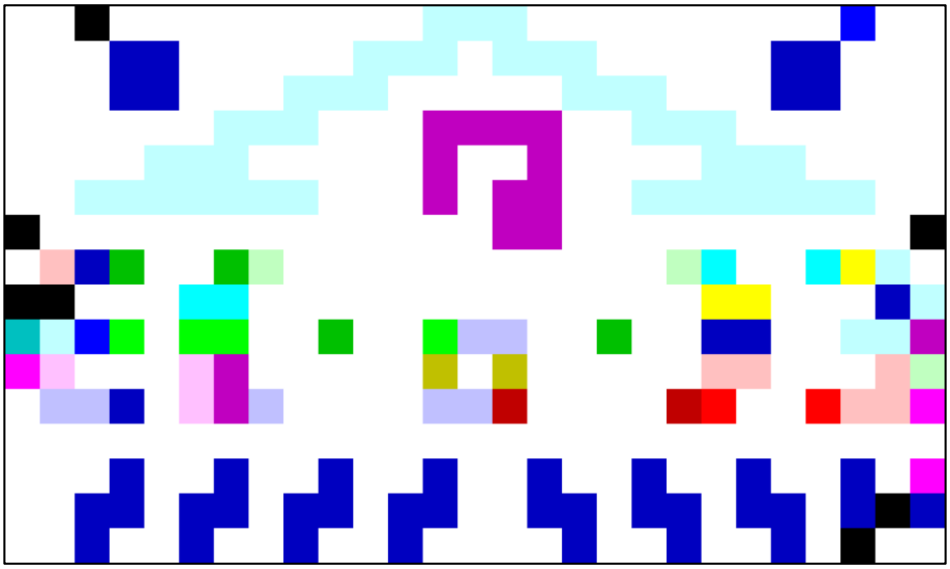
この絵のほとんどにはプログラム要素がありません．The お絵描き．
そう聞くと味気ないように思えるかもしれません．ですが，意味のあるプログラムを含ませながらお絵描きをするのは想像以上に大変です…．(とはいえまだ簡単)

・ 階乗を計算するプログラム

１．先ほどと同様，まずはプログラミングを....
しかし，かなり苦労.



２．１．を用いてお絵描き！ タイトルのいかたこうなぎを描く！



【ひとこと】
まず，このプログラムは当時の勉強会で作ることが出来なかったものです．イラストへのプログラムの組み込みがとても上手く行き，感動も大きかったです．

本題のいかたこうなぎなのですが，大学の講義中になんとなく思いついたキャラクターです．可愛いでしょう?ね？ねっ？足は８本です．
いかの部分が美味しく，お魚が寄ってきている感じをイメージしました．

＊おわりに　　いかたこうなぎは，体がちぎれて増えます．

参考資料リンク：https://www.slideshare.net/KMC_JP/piet-80098546

Mac を買って デザインを始めることになった話
～タスク王に俺はなる～

学部/回生	情報理工学部 セキュリティ・ネットワークコース 3回生
名前	副執行委員長の てんちょ（@kanon_k4）

約1年前の話、Macbook Proを買いました。
（理由は、「RAM 4GBのPCから逃げたい」＋「悪ノリ」です。）

新しいPCで何かやってみたい！

大体の人は、そんなこと思いますよね。
たまたま、学友会という立命館大学の自治組織の中にデザインの大先生がいたこともあり、少しお勉強をすると、完全にハマってしまいました。

（なお、本業はインフラ系で普段はオンプレサーバやネットワーク管理などしてます）

まぁ、気づいていたら、本誌の冒頭にも書いてあるように、RCCのWebサイトを完全リニューアルしてました。
ここで旧サイトを供養しておきたいと思います。

旧サイトさん

今までありがとう

お疲れ様でした！

ってことで、こんな感じに新サイトを作りました。

タスク王の
↘魔剤タワー↙

完成版がこちら

ちなみにWebサイト以外にも、Twitter向けのアイキャッチとかも積極的に作成したりしてます。(他団体なので画像はありませんが...)

せっかくデザインを学び始めたので、
名刺も新デザインにになりました！
リメイクした“Kanon”のアイコンを基調としたカラーデザインをベースとしてます。
（某企業の人にも「前の名刺よりもよくなった！」とお褒めをいただきました。）

ちなみにここ半年くらいでポートフォリオとか、ブログとか始めるようになったんですが、ここまでデザインこだわる前だったので、今後直していこうと思ってます。是非よかったら見てみてください。

ポートフォリオ：https://k0ui1.jp

ブログ：https://www.kanon-k4.com

個人的な話になりますが、研究室配属が無事(?)終わり、ネットワーク関連の研究室に配属されることになりました。RCCで学んだ技術を活かしたいなーって思ってます。

研究室とRCCと学友会とRiST(情報理工学部プロジェクト団体)の4つのタスクをいかにして処理していくかは、永遠の課題です。1つに集中すると3つが回らないので、4人に分身しないといけない気がしてきました。

次は、某C3のサイトをリニューアルするかもしれません。

タスク王への道はまだまだ続く。

OSSに貢献しよう

学部/回生	情報理工学部知能情報コース3回
名前	IK (@get_me_power)

IK

OSSに貢献しよう

はじめに

学生の間に技術系バイトをしたり、インターンに参加して自分のスキルを磨く話はよく聞く。 しかし、OSSに貢献する話は個人的にあまり聞かない印象を受ける。我々にとってOSSは思っている以上に身近であり、だれでも開発に参加できるものであるということを、このセクションで知ってもらえれば幸いである。

OSSとは

オープンソースソフトウェアの略である。利用者の目的を問わずにソースコードを使用、調査、再利用、拡張などが可能なソフトウェアの総称である。

有名なOSSを例として以下に挙げる

- Vim （テキストエディタ）
- Visual Studio Code（テキストエディタ）
- Ruby（プログラミング言語）
- Ruby on Rails（フレームワーク）
- Arch Linux（OS）
- マストドン（Webアプリケーション）

OSSに貢献する手順

至ってシンプルである。以下にGitHubで管理されているソフトウェアへの貢献の手順について記す

1. リポジトリをforkする
2. patchを書く
3. Pull Requestを送る

後はメンテナからの指示を待つだけ

> ⚠ 場合によってはテストコードを書けと言われることもある。patchを書く前に貢献したいリポジトリの雰囲気を掴むことは重要だ。過去のcommit履歴を見たり、CloseされたPull Requestなどを見る事を勧める。

OSSで得られること

英語のスキル

もちろん、基本的なやり取りは英語になるので英語のスキルは必須である。
OSSで英語でのコミュニケーションを体験し、慣れることができるだろう。

実用的なソフトウェアのコードを触れる事ができる

OSSとして公開されているソフトウェアは多岐に渡る。Vim pluginやWebアプリ（マストドンなど）、ライブラリやフレームワーク、OSなど様々である。
自分の興味があったり、詳しい分野に関する有名なソフトウェアについて知ることができるだろう。

広い視野を身につけることができる

OSSに貢献することは様々なことを考える機会になる。ソフトウェアを保守しつつ、革新していく流れが開発に参加することでわかっていくだろう。以下にいくつか例を挙げる。

後方互換性

自分の書いたpatchは、果たして古いバージョンに対応しているのか、これは重要な要素である。基本的に古いバージョンでも新しいバージョンでも動くコードを書かなければならない。ネックな部分だが、ここを疎かにしてしまうと、UserもDeveloperも減ってしまうだろう。

マルチプラットフォーム対応

基本的にどのOSでも動くことが望ましいことであり、当然それは難しいことである。

どのOSにも対応しなければユーザは減ってしまうことも事実だ。OSの違いや特性を理解した上でpatchを書くことを要求される。

ドキュメント修正、typo修正

ユーザや開発者にとってドキュメントは重要だ。時々、これらの内容が古かったりより良い方法があるのにそれを書いていないことがある。それを更新、修正することも開発者の役目だ。

新機能の追加

ソフトウェアを保守することは大変重要なことだが、保守だけではユーザは減ってしまう。

旧APIの更新だったり、version UPを行ったりと時代に追いつく必要がある。

最後に

世の中には数多くのOSSが存在し、多くのユーザがそれを使っている。バイトやインターンシップで経験を積むこともいいことだが、OSSにPull Requestを出してもいい経験ができると思う。僕は様々なVimのプラグインに貢献して、いろいろなことを得ることができた。 Mergeされれば嬉しいし、そのソフトウェアを使っているユーザが自分の書いたコードを使っていることもとても嬉しいことだろう。この記事を読んで、OSSに貢献する人が増えることを強く望む。

認証 VS 認可

学部/回生	情報理工学部 実世界情報コース 3回生
名前	ヨシエダ ユキホ

どうも、ユキホです。認証・認可技術大好きJKです(大嘘)。さて、今回は認証と認可について述べようと思います。タイトルでは「認証 VS 認可」となっていますが、対になっている概念ではありません。これはタイトル詐欺。

目次

1. 認証・認可とは？

再度述べますが、認証と認可は対になる概念ではありません。また混同されがちですが、メロンとメロンパン、JavaとJavaScriptくらい違います。みなさんに馴染みの深いのは認証だと思います。ログインと言えばピンとくる方もいるかもしれません。認証(Authentication、略してAuthN)とは、ユーザがシステムに対してユーザ本人であることを証明することです。IDとパスワードをペアにして送信する方法やIDと指紋を証明する方法など多岐に渡ります。ここで重要なのは、ユーザを識別する要素とユーザしか知り得ないもしくは持ち得ない要素をペアにして証明していることです。

対して認可(Authorization、略してAuthZ)は、他のサービスに対してユーザのリソース(情報)にアクセスする権限を許可することを指します。認可を提供するサービスにもよりますが、読み取り権限、読み込み権限など最低限の権限を与えることが多いです。Twitterを例にすると、「ユーザのタイムラインを読み取る」権限が挙げられます。ここで重要なのは、ユーザが第三者(あるサービス)に対して自分のリソースにアクセスする権限を与えていることです。

まとめると以下のようになります。認証と認可が全く別の概念だということがわかると思います。

認証

ユーザが
本当にそのユーザかどうかを
システムに証明すること

認可

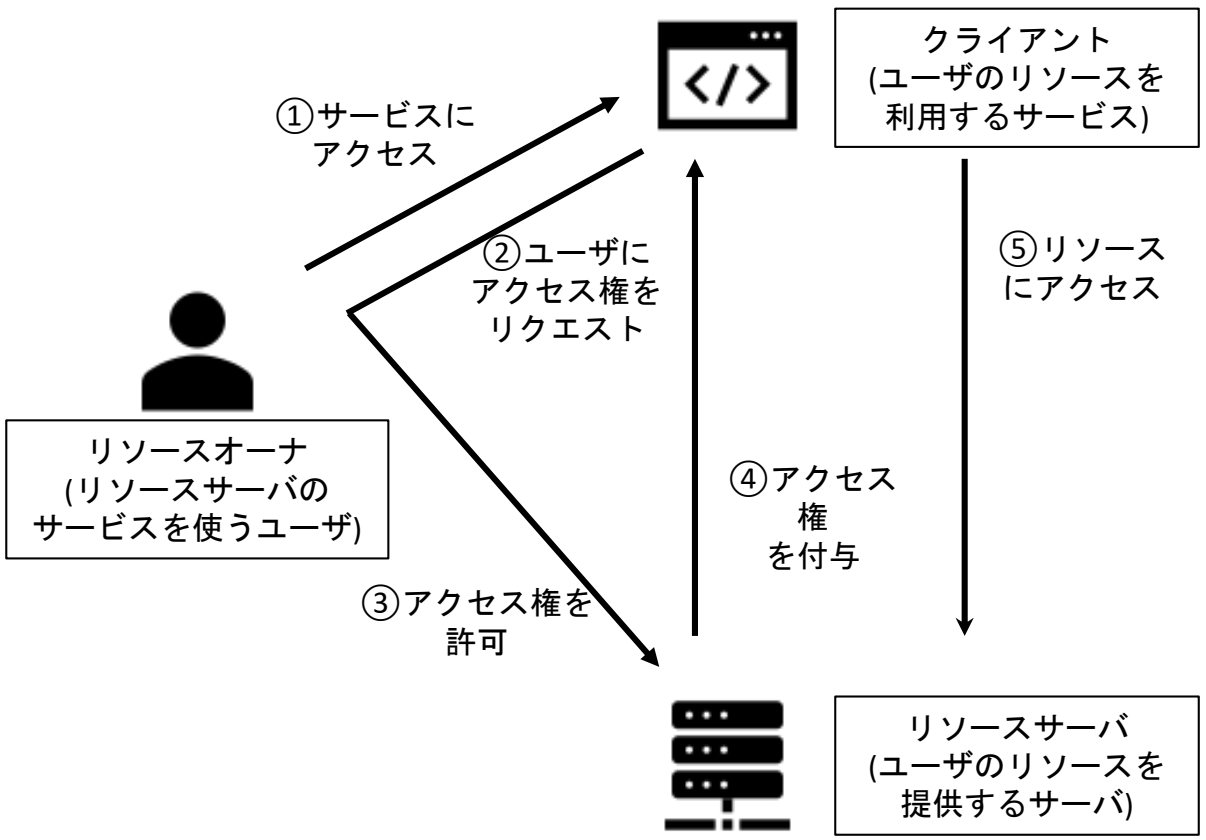
ユーザが他の
サービスに対して
ユーザのリソースにアクセス
する権限を与えること

2. 認証・認可を支える技術

前節では認証・認可の考え方について述べました。ここではそれを支える技術について述べたいと思います。これらの技術の大半はIETF (Internet Engineering Task Force)が発行するRFC (Request For Comments)という文章群で標準化されています。

まず認可に関する技術です。現在はOAuth2.0(RFC6749、読み方：オー オース)というフレームがオープンスタンダードとなっています。OAuth2.0は『サードパーティーアプリケーションによるHTTPサービスへの限定的なアクセスを可能にする認可フレームワーク』(*1)です。LINEやFacebook SlackもOAuth2.0を実装したサービスを提供しています。ちなみにTwitterはOAuth1.0aとOAuth2.0を併用しています(*2)。ツイートを解析するWebサービスはOAuth2.0を利用したサービスの1つです。ここでOAuth2.0の仕組みを下図で簡単に説明します。クライアントを「ツイートを解析するWebサービス」、リソースサーバを「Twitter」と考えていただければ想像しやすいと思います。

OAuth2.0の仕組み



*1 https://openid-foundation-japan.github.io/rfc6749.ja.htmlより引用
*2 https://developer.twitter.com/en/docs/basics/authentication/overview/oauth

続いて認証に関する技術です。近年注目を浴びているOpenID Connect (略してOIDC)について述べます。OIDCはOAuth2.0に認証機能を追加したもの(原文翻訳：『OAuth 2.0 プロトコルの上にシンプルなアイデンティティレイヤーを付与したもの』(*3))です。次節で詳しく述べますが、OAuth2.0はしばしば認証代わりとして利用されています。場合によっては、非常に重大なセキュリティホールとなるため、認証をしたいときはOIDCを利用しましょう。

3. 「OAuth認証」という間違い

Googleなどで「OAuth認証」と検索すると結構な数の記事がヒットします。しかしながらこれらはすべて間違いです。ここまで読んだくださった方は察しがつくと思いますが、OAuthは認可のフレームワークです。認証で使用されるという想定はされておらず、もちろんセキュアではありません。家の鍵を持っている人が家主であるとは限らないのです。詳しく知りたい方は「The problem with OAuth for Authentication.」(*4 *5)を読んでみてください。

なぜ間違った使い方をされるのでしょうか。これは筆者の見解ですが、認可をする過程でリソースサーバ側でログイン処理が入るケースがあり、これを「認証した」とするサービスが出てきたと考えられます。

4. まとめ

これまでの話をまとめると、認証は「システムにユーザが本人か証明すること」、認可は「ユーザ自身のリソースのアクセスを他のサービスに許可すること」でした。そして認証はOpenID Connectが認可はOAuth2.0が主な技術として支えています。

この記事を読んでいただいた方の中で興味を持たれた方はRFC(*6)やOpenID Japan(*7)を覗いてみると楽しいかもしれません。また新たな認証技術であるFIDO(*8)も面白いかもしれません。

ここまで読んでいただき、ありがとうございました！ユキホでした！

---

*3 https://openid-foundation-japan.github.io/openid-connect-core-1_0.ja.html より引用
*4 http://www.thread-safe.com/2012/01/problem-with-oauth-for-authentication.html
*5 *4の日本語解説 https://www.sakimura.org/2012/02/1487/
*6 https://www.ietf.org/standards/rfcs/
*7 https://www.openid.or.jp/
*8 https://fidoalliance.org/?lang=ja

論文の再現実装をしてみた

学部/回生	情報理工学部　先端画像音メディアアーキテクト 実世界知能情報グローバルコース 3回生
名前	gengen(Twitter: @gengen_ml)

はじめに

　こんにちは，スクレイピングから自然言語処理，画像や音声処理，機械学習の勉強をしているので，人に言われるままにコース名を補完していったら，ほとんどのコースを網羅してしまったgengenです．(本来の所属は実世界情報コースです．そこ，コースが迷子とか言わない！)

今回は音楽情報処理の国際学会であるISMIRで2017年に発表された，Singing Voice Separation with Deep U-Net Convolutional Neural Networksという論文で発表された機械学習のモデルの再現実装をした内容を書きたいと思います．

　本論文の概要を簡単に説明させていただくと，音声信号からボーカルの成分や各楽器の成分を分離するという音源分離と呼ばれるタスクを，音声の処理ではなく画像の処理としてU-Netで行おうという試みです．

U-Netとは

　先ほどから何度か出てきているU-Netという単語は，2015年に発表されたセグメンテーション(領域識別)のためのencoder-decoderモデルの名称です．画像を入力として画像を出力します．構造としては図1のようなUの形になっていることからU-Netと呼ばれています．左半分の畳み込み層で元画像が持つ特徴を獲得(encode)して，右半分の逆畳み込み層で復元する(decode)という仕組みから，encoder-decoderモデルと呼ばれています．医療用として細胞のセグメンテーションなどで使われていたみたいです．ちなみに図1はオリジナルのU-Netのアーキテクチャで，本論文で提案されたU-Netとは異なるので注意してください．

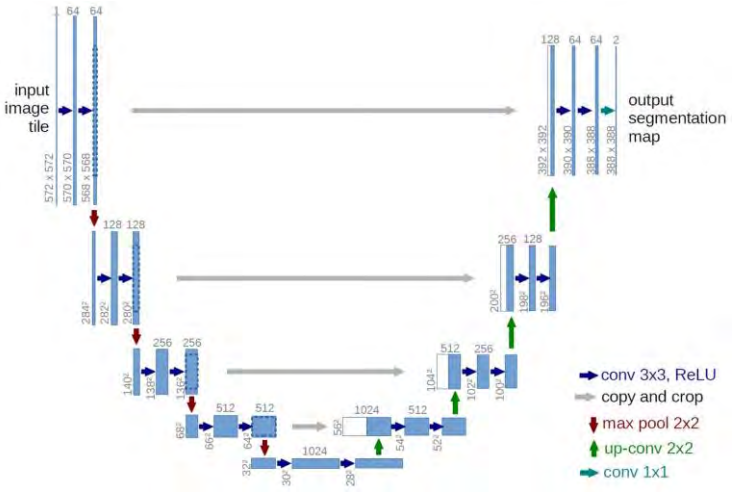


図1 U-Netの構造
(https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/)

U-Netは画像を入力として，画像を出力するモデルでしたが，音声をどのようにして画像に変換するのでしょうか？今回の論文では音声のデータをスペクトログラムという画像データに変換して入力としていました．この「音声」を「画像」として処理するというところにこの論文の面白さがあると僕は思います．

スペクトログラムとは

スペクトログラムとは音声信号を窓関数に通して，フーリエ変換を用いて周波数スペクトルを計算した結果を，横軸に時間，縦軸に周波数，その音の強さを色で示した三次元のグラフです．

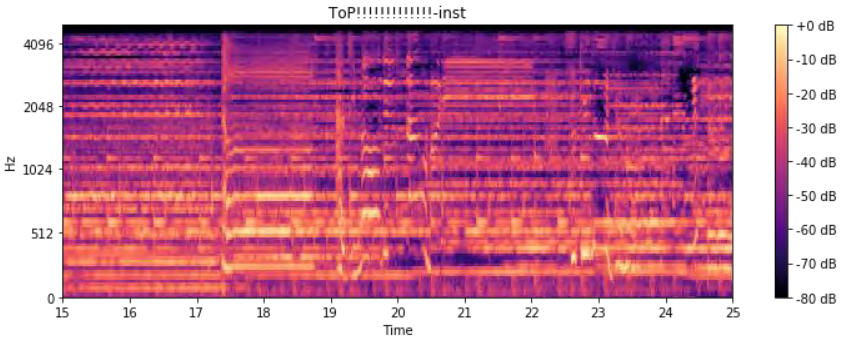


図2 ToP!!!!!!!!!!!!!という楽曲のOff vocal(インスト)のスペクトログラム

マスク処理

マスク処理とは，特定の部分のみを抽出し，それ以外の部分を消すための画像処理のことです．今回のタスクでは，U-Netにmix音源(歌声+伴奏)のスペクトログラムを与え，ニューラルネットに入力と同じサイズの「マスク」画像を出力させます．このマスクを入力である元音源に掛けることによって，歌声の部分だけを抽出することができます．

実装

実際のネットワークの構築には深層学習用のフレームワークであるChainer，音声処理の部分にはlibrosaというライブラリを用いて，各種の音声における前処理と論文で発表されたモデルを構築しました．

その後，DSD100という音源分離用のタスクでよく用いられている楽曲群を用いて学習させました．この楽曲群は各楽器ごとのマルチトラック音源(vocal, ピアノ, ドラム等)と全てが合わさっているmix音源が収録された合計100曲分からなるデータセットです．

mix音源はそのまま入力として用い，教師データとしてvocalを除いた全てのマルチトラック音源を合わせた音源を生成して用いました．学習時の損失関数は論文にあるように入力であるmix音源と出力であるマスク画像の積(ココでの積の結果の理想的な値は伴奏部分であることを思い出して下さい)とターゲットである伴奏音源との差のL1ノルムです．

結果

今回はTop!!!!!!!!!!!!!という楽曲を対象にテストしてみました．その結果が図3です．この曲は出だし約20秒は伴奏のみの曲であり，vocalのスペクトログラムを可視化したもの(図3右)を見ていただければ，その部分はごっそりと音声の強さがないことがわかると思います．この結果をみるとある程度はうまくモデルを学習できたのかなと思います．

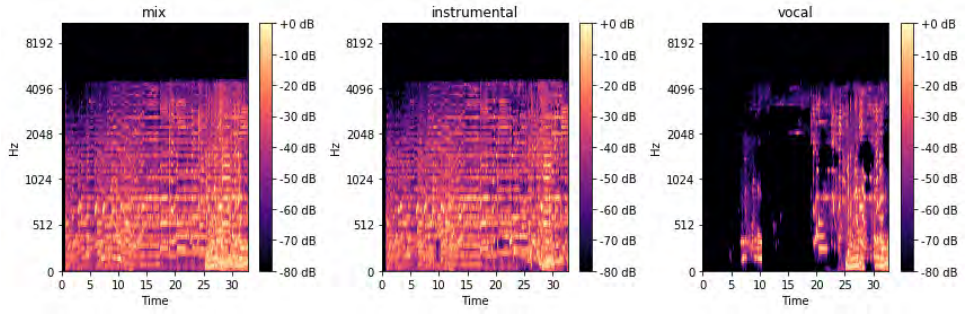


図3 Top!!!!!!!!!!!!!の各音源のスペクトログラム

考察(今後の予定)

ToP!!!!!!!!!!!!!の6から10秒のあたりや，他の楽曲でテストしてみた結果など，綺麗に分離できないものやなんとなく籠った音になるものなど聴覚的にうまくいかないものも多かったです．

この要因として，学習させたデータセットが小さいものである可能性が挙げられます．また，DSD100のデータセットは洋楽で男性ボーカルの楽曲が多く，実際に僕が分離したい楽曲としては邦楽で女性ボーカルのものが多いのですが，これらの差異からうまく結果が得られていないのかも知れないです．

よって，今後の展望としては，画像系の学習では回転などでdata augmentationをすると精度が上がることもあることから，この分野で有効かは解りませんが，音源のdata augmentation(データセットをランダムに繋ぎ合わせたり)を試したり，日本の楽曲データセットを集めて試してみたいです．僕の観測範囲内では比較的楽に入手できるデータセットがDSD100くらいと集めるだけ集めたものの，学習させる環境的に容量が足りず活用できていないMedleyDBとMUSDBしかなく，どちらも日本語の楽曲データセットではないので，どうしたものか頭を悩ませています...

籠った音になることに関しては改善方法が思い浮かばないので誰か教えてほしいです...何かしらの情報をお持ちの方は**Twitter(@gengen_ml)**で是非僕に情報をください！！

参考文献

Andreas Jansson, Eric Humphrey, Nicola Montecchio, Rachel Bittner, Aparna Kumar, Tillman Weyde. “Singing Voice Separation with Deep U-Net Convolutional Networks”, 18th International Society for Music Information Retrieval Conference, Suzhou, China, 2017.

そうだ。何もわからないけどLINE Botを作ろう

学部/回生	情報理工学部知能情報コース　3回生
名前	伊藤聡子(いとさと)　Twitter : @it0sat0

そうだ。自宅管理Botを作ろう。

そんなことから今まで全く触ったことがない技術ばかりを使ってLINE Botを作ることにしました。いとさとです。

0. 前書き

「ラ・イトサトール」

これが私の家の通称です。ネットユーザーたちの注目を常に集めている兄貴の前住居の名前が捩られており、「あれかな？」と心当たりのある方もいるかもしれません。私のあだ名と響きが近いが故についた通称なのですが、私の家は立地がいいことで仲の良い友人たちと食事や宴会をし盛り上がることも多い我が家。「飲み物ある？」「お菓子ある？」「なん号室だっけ？」などのやりとりも多々あります。そんなやりとりを無くし、ものが溢れる我が家の片付け時間を引き伸ばしてくれると期待して開発をすることにしたのが、LINE Botでした。

その過程と努力、そして、苦悩を記そうと思います。

1. 使用したもの

開発に使用したものは、以下の５つです。

①LINE Messaging API　…　LINE Botなどを楽に開発するためのツール
②LINE Bot Designer　…　コードを書かずにLINE Botのデザインができるソフト
③Heroku　…　簡単にWebアプリの運用ができるサービス
④Node.js　…　サーバーサイドの処理を書くための JavaScript
⑤フリー素材のアイコン　…　素材として使う

はじめに⑤で素材集めをし、②でデザインをしつつ①③④を使っての開発となりました。

2. 開発過程

開発に割ける時間はわずか10時間ほどと、どこからどう見ても絶体絶命という状態からの開発が始まりました。

2.1 構成を考える

何を開発するにも、構成を考えないことには先には進めません。

まず、いちいち文字を打つ手間をなくすため、リッチメニュー利用することにし、載せる情報を以下に決めました。

①住所(部屋番号)情報　②Wi-Fi情報　③食べ物情報　④飲み物情報

④に関しては、ソフトドリンクとアルコール飲料に分け、追加で詳細情報も入れることにし、コンテンツの数を６つにしました。これは登録できるリッチメニューのコンテンツの最大数でもありました。つまり、都合が良いようにコンテンツ数の調節を行いました。

2.2 LINE@の登録

LINE Developersからプロバイダーとチャネルを作成し、基本情報の入力等を行います。アイコンは友人がプレゼントしてくれた右のもので、アプリ名(アカウント名)はもちろん「ラ・イトサトール」にしました。

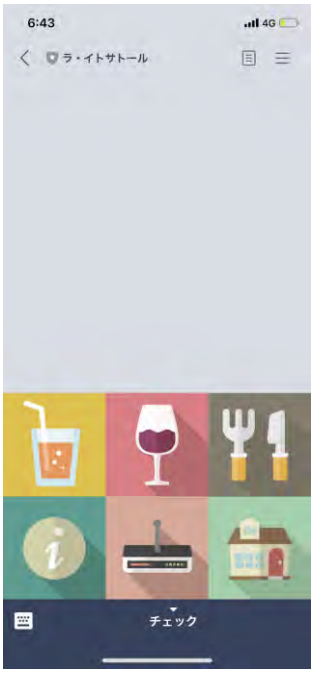
このまま続きで、友達追加した際の自動送信メッセージも適当に登録します。

2.3 コードを書いて実装する

LINE Botは全てWeb上で登録や設定できるものではありません。コードを書く必要が出てきます。今回は、使ったことがなかったものの資料が豊富だったHerokuとNode.jsをコンビで使うことにしました。

「資料いっぱいあるし、なんとかなるやろ～」と思いながら始まった実装でしたが、なんと、それ通り実装してもうまく動かない箇所が出てきてしまう！　初心者には厳しい展開！とはいえ、そこで焦っていても何も進まず、進むのは時間のみ。手を動かすしかないのです。深夜0時から始まった開発は、複数の資料を組み合わせることでようやく基本的な動きが実装でき、その時点で空がしらんでくる時間となっていました。

2.5 コードを書かない部分の実装



リッチメニューの実装を、**LINE@ Manager**で行います。リッチメニューというのは、UNIQLOやニトリの公式LINEのトーク画面で、**キーボードの位置からぬるっと出てくる画像のゾーンのこと**を指します。右の画像の、画像が６つ並んでいる部分です。

商用利用もOKのフラットデザインのアイコンを、サイズ調整及び結合を行い、アップロードします。その後、６つのコンテンツそれぞれの設定を行いました。一切苦しむこともなく、スムーズに進む作業。素晴らしかったです。バグがなければ画像をタップすることで自動で設定された単語が送信され、それぞれのコンテンツが実行されるようになります。

2.6 あとはバグとの戦い

「コンテンツとして登録した画像が表示されない！」

これは大事件です。リッチメニューは表示されるものの、そこから登場するコンテンツの画像が表示されないという事件が起こったのです。原因は**画像の縦横比とサイズ(データ量)**。LINE Botは、画像に関する制約が厳しかったのです。なんども確認したつもりが、画像を編集し書き出しを行った際に縦横比が変わっていたりサイズが大きくなったり……。このバグが解消する頃には、太陽が昇りきっていました。

3. 完成と今後

個人情報の塊なので、「ぜひ登録してください！」とはできないのですが、全体的にポップな雰囲気に仕上がり、食べ物情報では在庫のある食材やお菓子だけでなく、私が作ることのできるメニューも載せることとしました。自炊をしろという自分への戒めにもなればいいかと。

今後は、ケーキ教室に通い始めたこともあり、何か部室に持っていく際には、事前に一斉連絡を流すといった一般のBotとして運用するだけでなく、冷蔵庫の中身の管理なども行えるようにしていく予定です。最後に一言。

皆さんは画像の扱いちゃんとしましょうね！
時間の浪費だけはやめましょうね！！！

4. 参考文献(めちゃめちゃありがたかったサイト一覧)

・FLAT ICON DESOGN(http://flat-icon-design.com/)
・今更ながらLINEBotを作ってみた(https://orfool.com/programing/1557/)
・LINEのBot開発 超入門（前編） ゼロから応答ができるまで
(https://qiita.com/nkjm/items/38808bbc97d6927837cd)
・LINE BOTをHeroku + Node.jsでやるまで
(https://qiita.com/TakuTaku04/items/cb71f10669a9e9cbf71b)
・Messaging APIリファレンス(https://developers.line.biz/ja/reference/messaging-api/)

セグ木と代数構造

学部/回生	理工学部 数理科学科 3回生
名前	小泉 孝弥

皆さんは「セグメントツリー」というデータ構造を知っていますか?
セグメントツリーは数列に対して、ある操作を高速に行うことができます．セグメントツリーを使うことで以下の問題を解くことができます．

問題1
与えられるもの
N: 数列の要素の個数．(最大で10^5くらい)
a_{i} : Nこの整数からなる数列．
q: 与えられるクエリの数．(最大で10^5くらい)
ここで、クエリとは
1. 数列のi番目の要素を別の整数に書き換える．
2. 数列のlからrまでの区間和を求めてそれを出力する．
のどちらかが与えられます．

このクエリを高速で処理できるアルゴリズムを作成してください．

要するに、数列の中身を書き換える操作や、数列の区間和を求めるみたいな指示がいっぱいくるので、それを高速に処理してね．っていうことです

この問題は上にも書いた通り、セグメントツリーを使うことで解くことができます．(セグメントツリーの仕組みについては今回は説明しません)

では、どのような問題な問題をセグメントツリーで解くことができるのでしょうか?
それを考えるために、もう一問セグメントツリーを使う問題を考えます

問題2
与えられるもの
N: 数列の要素の個数．(最大で10^5くらい)
a_{i} : Nこの整数からなる数列．
q: 与えられるクエリの数．(最大で10^5くらい)
ここで、クエリとは
1. 数列のi番目の要素を別の整数に書き換える．
2. 数列のlからrまでの最大値を求めてそれを出力する．
のどちらかが与えられます．

このクエリを高速で処理できるアルゴリズムを作成してください．

問題1と問題2は、区間の和を求めるのか最大値を求めるのかという点で異なっています．しかし、異なる点があるにも関わらず、同じアルゴリズムを用いて解くことができます．これは、「和」と「最大値」には共通する性質があるからです．その性質とは以下のようなものです．

1. 結合律が成立する. すなわち、任意の整数a, b, cに対して
(a + b) + c = a + (b + c), max(max(a, b), c) = max(a, max(b, c))
が成立する.
ex. (2 + 3) + 5 = 2 + (3 + 5), min(min(2, 3), 5) = min(2, min(2, 5))
2. 単位元が存在する. すなわち、任意の整数aに対して
a + 0 = 0 + a = a, max(a, 0) = max(0, a) = a
が成立する.
ex. 3 + 0 = 0 + 3 = 3, max(3, 0) = max(0, 3) = 3

これらの性質を満たすものは「モノイド」と呼ばれます.

定義(モノイド)
集合GとG上の演算*が与えられます． 以下の2つの条件を満たす時に
(G, *)をモノイドと言います.
1. 結合律が成立する.
2.Gに*の単位元が存在する.

モノイドへの一般化
最後に以下のような問題を考えます.

与えられるもの
N: 数列の要素の個数．(最大で10^5くらい)
a_{i} : Nこの整数からなる数列.
q: 与えられるクエリの数．(最大で10^5くらい)
ここで、クエリとは
1. 数列のi番目の要素を別の整数に書き換える.
2. 数列のlからrまでのOpeを求めてそれを出力する.
のどちらかが与えられます.
(Opeには何かしらの演算(+や-など)が入ります)

このクエリを高速で処理できるアルゴリズムを作成してください.

実は、これらの問題は演算Opeがモノイドの条件を満たす時に、セグメントツリーで解くことができます.

【Ubuntu】GPU周りの環境構築やってみた

学部/回生	情報理工学部知能情報コース3回生
名前	HiroRittsu

事のはじめは今から１年ほど前。RCCとはまた別の団体で購入されたAlienwareにUbuntuをデュアルブートするところから始まる。実はそれまでにも数十台というWindowsPCにUbuntuをデュアルブートすることはあったため、AlienwareへのUbuntu導入はそこまで難しくはないだろうと思っていた。しかし実際は先輩や同期も含めて導入を試みるも、GPUドライバ関連の導入で苦戦、結局半年ほどかかってしまった（このときの導入の流れはRCC2018アドベントカレンダーで紹介している）。

それから半年。今度は研究室配属で使えるようになった個人用デスクトップPC。WindowsよりもUbuntuを使いたかったため、今までと同じようにデュアルブートをしたところ、今までの経験をフル活用したが、これまた３日ほど時間を費やしてしまった。今回はその時に得た、導入に際して気をつけるべき事などを紹介する。

導入PCのスペックは以下の通り。

LEVEL∞	
CPU	i7-9700k
GPU	RTX2080ti
RAM	64GB
主記憶媒体	HDD（3TB＋4TB）

（さすがハイエンドゲーミングPC...果たして使いこなせるか...）という話は置いといて。今回はHDDが２つあったため、片方はWindowsOS用、もう一つをUbuntu用とした。また、インストールするのはUbuntu18.04.3、そしてその上でPython3.6のTensorflow-GPUが正常に動作するところまで環境構築を行った。

1,インストールUSBの作成
　デュアルブートをする上で必ず必要になるインストールUSB。ただ、ここで早速落とし穴が存在する。それはインストールUSBを作るツールと、展開するisoファイルのカーネルバージョンである。特にisoファイルに関してだが、日本語版Ubuntuをインストールしてしまうと、カーネルがかなり古いものが入ってしまう。カーネルが古いとネットワーク系のドライバを始めかなり不具合が多くなってしまう。そのため、日本語版ではなく、**Ubuntu公式のオリジナルisoファイルを展開すべきである。**
また、インストールUSBを作成するツールもいくつか存在するが、PCの相性によってはUSBが認識、もしくは起動しないことがある。そのため、もし**BIOSの設定を設定したにもかかわらず問題が発生する場合は、作成するツールを変更して対応**する。

2,BIOSの設定
　Ubuntuをインストールする上で必ずしなければならないBIOSの設定を行う。ここでキーとなる単語は「セキュアブート」と「ブート順序」。基本的には他の記事にかかれている手順を追っていけばできるが、「めんどい～」というそこの貴方。**何があってもセキュアブートだけは解除すべし**。そしてもう一つ、**ブート順序はインストール後もとに戻すべし。**
実はPC、特にマザーボードの相性にもよるとは思うが、今回私が導入したPCでは、USBのブート順序を最優先にしたままセットアップをしていたところ、突然キーボードやマウスなどのUSB機器が反応しなくなるという現象が発生した。もし同じような現象が発生した場合は、ブート順序を見直してほしい。

3,grubの設定
　おそらくGPUがあるPCで８割（当社比）の確率でインストールUSBが起動してもデスクトップに行く前にフリーズしてしまうだろう。その現象を回避するために、インストールUSBが起動してすぐ出てくる黒、もしくは紫の画面で”e”キーを押下し、**”quiet splash”の部分を”nomodeset”に書き換えれば起動する。**

4,CUDA＆cuDNNのバージョン
　そしてUbuntuのデュアルブートが成功し、GPUドライバの導入（割愛）が成功したら、最後の難関であるCUDAとcuDNNの導入である。特に何も考えずに最新バージョンを入れてしまうと痛い目に遭ってしまう。**必ずTensorFlow公式が発表しているバージョン表を見つつ、NVIDIA公式から導入する必要がある。**　　（PS：紙が足りなかった。

カスタムプロンプト

学部/回生	情報理工学部 知能情報コース3回生
名前	カイト

BashのプロンプトをかっこよくカスタマイズできるStarshipの紹介です。

StarshipはRustで作られたカスタムプロンプトのためのプラグインです。

まず、Rustをインストール。
端末を立ち上げて以下を入力。

```
$ curl https://sh.rustup.rs -sSf | sh
```

次にStarship のインストール。

```
$ cargo install starship
```

これで完了。bashrcに以下を追記しましょう。

```
$ eval "$(starship init bash)"
```

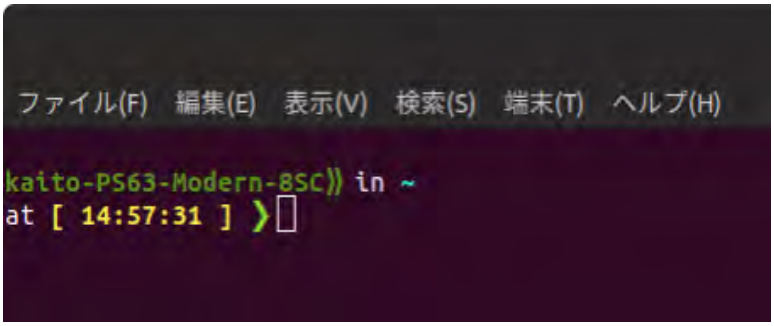
次にプロンプトをカスタマイズします。
カスタムプロンプトの設定はstarship.tomlに記述します。

```
$ touch ~/.config/starship.toml
```

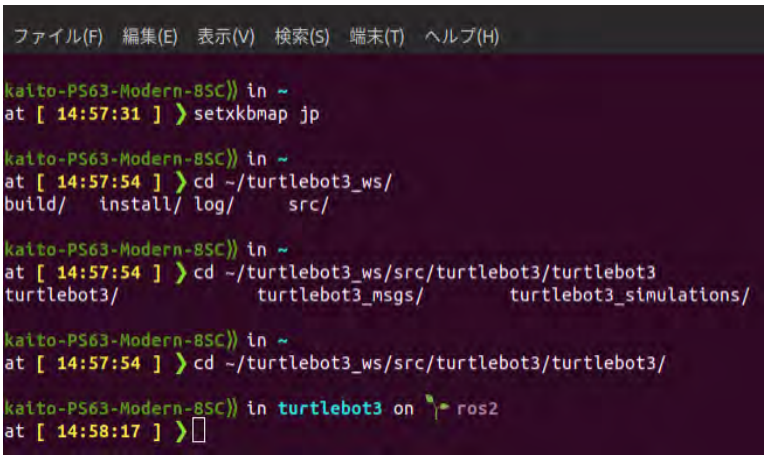
以下のような設定を書けば、とりあえずは使えます。

```
[battery]
full_symbol = " "
charging_symbol = " "
discharging_symbol = " "

[[battery.display]]  # "bold red" style when capacity is between 0% and 10%
threshold = 10
style = "bold red"

[[battery.display]]  # "bold yellow" style when capacity is between 10% and 30%
threshold = 30
style = "bold yellow"

[git_branch]
symbol = " "
truncation_length = "4"
truncation_symbol = ""

```

さっきの設定で読み込んでやると、以下のようなかっこいいプロンプトができます。



他にも、Gitで管理するディレクトリだとこんな感じ



バッテリーの残量が少なくなると以下のように表示することもできます。

MacやLinux環境なら自分好みにカスタマイズできるので、ぜひ、試してみてください（Windowsユーザーはお帰りください）

参考

- Bashのプロンプトをおしゃれにするstarship　Qiita
  https://qiita.com/unhappychoice/items/3b774310d95e2124eb77
- Ubuntuのターミナルをかっこよく　Qrunch
  https://qrunch.net/@SUXSlFvZo6bPOyFW/entries/A2fu2XXdEkKtAVGo

エンジニアに囲まれるデザイナー

学部/回生	理工学部機械工学科3回生
名前	taken

プログラミングができないのにRCCに入り、プログラミングができないのにRCCに居続け、プログラミングができない機械工学科の僕は、居場所を確保するためにRCCでかれこれ2年ほどデザインを触っています。部員同士の技術的な会話にはついていけないですが、「デザイン要員」としてたまに利用されたりして結構楽しいです。RCCのおかげで専門学校や芸大の純粋なデザイナーには無い幅広い経験ができてると思ってます。
そんな僕がこの1年間で体験したことを紹介します。

Twitter

春休みにTwitterで色を4枚貼り付けたら結構バズりました。

僕は特に黒がデザイン的に重要だと思っていて、#000000の真っ黒よりもこういったちょっと薄い黒が今のトレンドになってる気がします。最近だとiOS13のSafariで見出しやタイトル部分の文字が一部自動的に太くかつこういう色になりましたよね、あれ好きです。

LT班

去年の会誌に掲載したように僕はLT(ライトニングトーク、エンジニア向けのプレゼン)のふざけたスライドを作ってるんですが、半年間RCC部員と一緒にLTスライドについて勉強しました。詳しくはこの会誌冒頭の方にあるプロジェクト活動欄にLT班の活動報告として掲載されています。

Experiences

春頃に関西の学生デザイナーが活動しているデザインの事務所に入りました。夏休みにはインターンに2週間行き、デザイン+αの業務をしました。詳しい内容は書くことができませんが、エンジニアが主体の企業さんだったので、RCCと似たような感じでエンジニアに囲まれながらデザインに携えることに親近感が湧いてとても有意義な体験でした。

Works

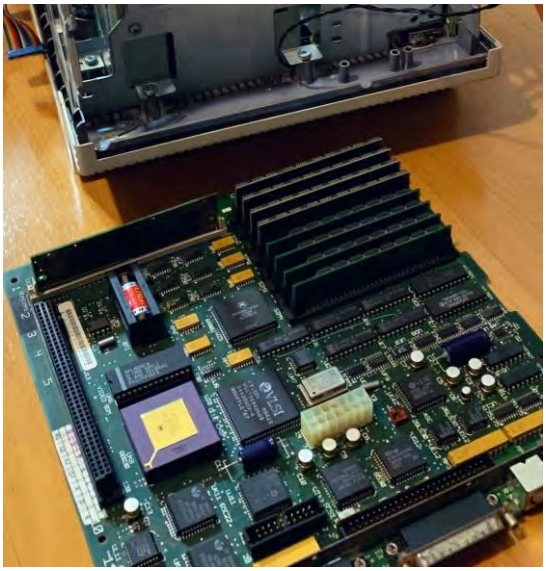
RCCの新しいロゴとこの会誌の表紙を作りました。ロゴの製作について詳しくは会誌冒頭のロゴについての記事をお読みください。

Macintosh SE/30 修理した。

学部/回生	RCC部 (部ではない) 執行委員長（ｷﾞﾘｷﾞﾘ3回生）
名前	@aonrjp 詳細はこちら!!→ aonori.dev

今時の青少年なら誰しも行ったことがある、アキバのレトロPC専門店『BEEP』で買った起動不可の Macintosh SE/30 を修理した話。

とりあえず起動させたい



購入時は電源投入不可だったが清掃してネットワークカードを取り除くと画面は点灯。手持ちの3.5”フロッピーにMac OS System 6 のインストールイメージを焼いても、正しく読まれず吐き出されてしまう。エミュレータのvMacでは正しく動くので、フロッピードライブを別途入手してなんとか起動。今時SCSIのHDDなんて遅いし怖いので、microSDをSCSIデバイス化できるSCSI2SDを利用した。

同人ハード最高!!

比較的新しいOSにしたい

一度Mac OSの立ち上げに成功すれば、以降はOSインストールのためにフロッピー十数枚を入れ替える必要はない。問題はUSBなどのインターフェースがないSE/30に、どうやってMacBinaryやOSイメージをコピーするか。今回はSCSI2SDを使っているため、SDカードを別のLinuxマシンでHFSとしてマウントすると、直接ファイルの読み書きができたので、これを通して必要なソフトとイメージをコピーした。当時の高価なSCSI-HDDよりも爆速・大容量で読み書きできてしまうSDカード、すごい。

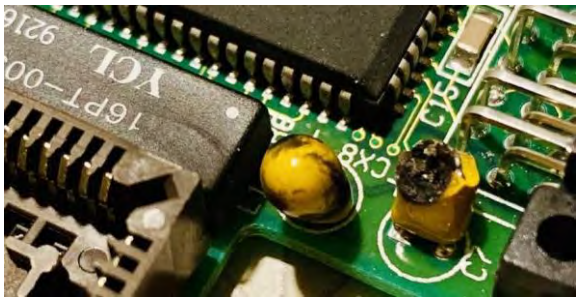
必要なソフトを入れてしまえば、今時のMacとそう変わらない。MacBinaryで圧縮されたイメージを展開して、kt_7.5.3.smiから漢字トーク7.5.3をインストールするだけだ。当時から日本語フォント Osaka やFinder の操作感など今の OSX にも通じるコンテンツや理念が数多く見て取れる。

素晴らしい。

KT 7.5.3 Installer

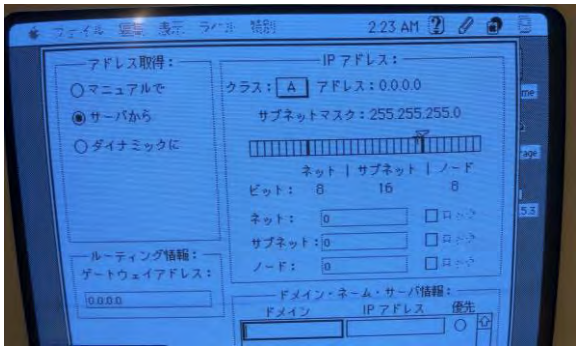
World Wide Web

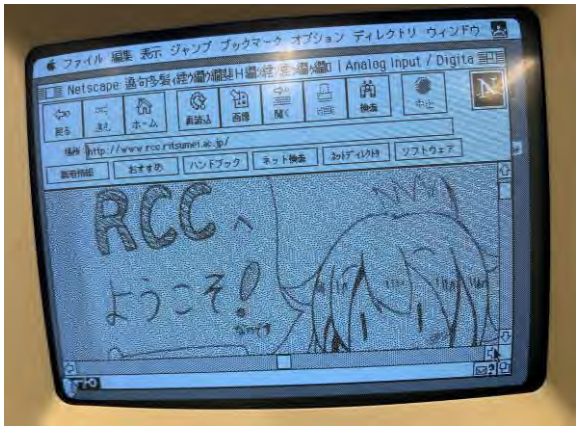
CERNによってWWWが発明された1989年こそ、Macintosh SE/30の発売年である。冒頭取り除いたネットワークカードを修復すれば、初期のブラウザを使ってインターネットに接続できるはず。



動作不良の原因は電源ラインに入ってたタンタルコンデンサが破裂していたから。新しいのを買ってきて取り替えてからSE/30に挿し直すと起動して、ちゃんと認識された。

ネットワークカードのDriverを入れて、MacTCPからネットワークの設定をする。設定項目は最近のIPv4 LANの構成と同様であるので、特別意識することは無い。サブネットマスクの設定インターフェースが面白い。





ブラウザは懐かしのNetscape Javascript周りがまともに動かず、SSLもまともに使えないので、簡素なWebサイト、つまりexample.comや阿部寛のHP、あるいは本会Webサイトがちょうどよかった。画像は重たいけどちゃんと表示されて感動。Unicodeという概念が無いので日本語表示はShift-JISだ!!

最後に一言

まだ数十年という比較的浅い歴史のコンピュータ、今ならまだギリギリ実機に触れながら時代の流れをトレースできます。「あの時触れられなかったもの」が「今なら触れる」となるのが「ジャンク品」という宝の山。そのまま使うもよし、修理して使うもよし、魔改造して新たな価値を見出すもよし。時代を支えた機材たちが、コンピュータサイエンスを学ぶ僕たちを待ち構えているのです...

参考文献

祝 Macintosh 30周年!! 速度の壁を破ったSE/30 | Mac
https://weekly.ascii.jp/elem/000/000/197/197096/

中国って便利

学部/回生	理工学部 電子情報工学科3回生
名前	渥美柾彦

夏休み特にやることがないので中国にいってきました。　率直な感想として、超楽しかった。そしてなにより便利。そして一度も現金を使用するどころか、見かけもしませんでした。なぜでしょう？
「Alipay」って皆さんご存じですかね？コンビニとかドラッグストアで見かけたことがある人もいるかもしれません。旅行中、このサービスのおかげで現金を一度も使用しなかったのですが、現地に着くまでは、中国で使われている「決済アプリ」程度の理解でした。しかし中国ではAlipayなしでは生きていけません。

とくに、地下鉄。利用する際は、Alipayのアプリで、その地域の地下鉄のヴァーチャルICOCAのようなものをインストールしておきます。そしてインストールしたQRカードの画面を改札にかざすだけ、そして下車する際は、再度改札に画面をかざすだけ。引き落としもすぐ確認できます。
「Suicaと変わらなくない」と感じた人もいるでしょう。皆さんスマートウォッチってご存知でしょうか。はいそうです、AppleWatchとかそういった類のものです。実際に僕が中国に滞在している間は、Alipayがつかえる、NFCに対応したスマートウォッチを使っていたため、腕時計を改札にかざし地下鉄の乗り降りをしていました。どうでしょうみなさん？便利でしょう。
Alipayはアリババグループの傘下の会社です。そして、アリババグループには中国のAmazonとよばれる、「Taobao」という会社があります。この会社が提供するサービスもとにかく便利です。もちろんAlipayでのオンラインでの決済払い戻しの速さも便利ですが、それ以上にオンラインショッピングの欠点であった、「イメージしていたものと違う」が起こりにくいのです。Taobaoには、出品者と会話する機能があり、その返信も私が利用した際は1分以内でした。日本でいうと、個人の間で取引される、「メルカリ」に近いサービスなのかもしれません。

まだまだ紹介したい中国の便利なサービスはたくさんあります。最近では、顔認証で決済などあるそうです。しかし、最後に残念なお知らせです。Alipayを使用するには中国の銀行の口座が必要なのです。ですが、Alipayを使わなくとも中国はとても観光のしがいのある国であるので、興味のある方は一度行ってみてください。

＜画像引用元＞
Alipay:https://seeklogo.net/alipay-logo-vector-eps-ai-free-download-92454.html

奥付

サークル名:立命館コンピュータクラブ
責任者:青木 雅典
発行日:2019/12/1
印刷所:株式会社 RED TRAIN様
HP:http://www.rcc.ritsumei.ac.jp
Twitter:@rits_rcc

30分で
読める!

RCC自作会誌

ritsumeikan computer club