**研究引継ぎ資料**

１．本研究の環境構築について

２．システム構成について

３．Json2ruby.rbの動作概要

４．ノードを追加したい場合

５．残っている課題点について

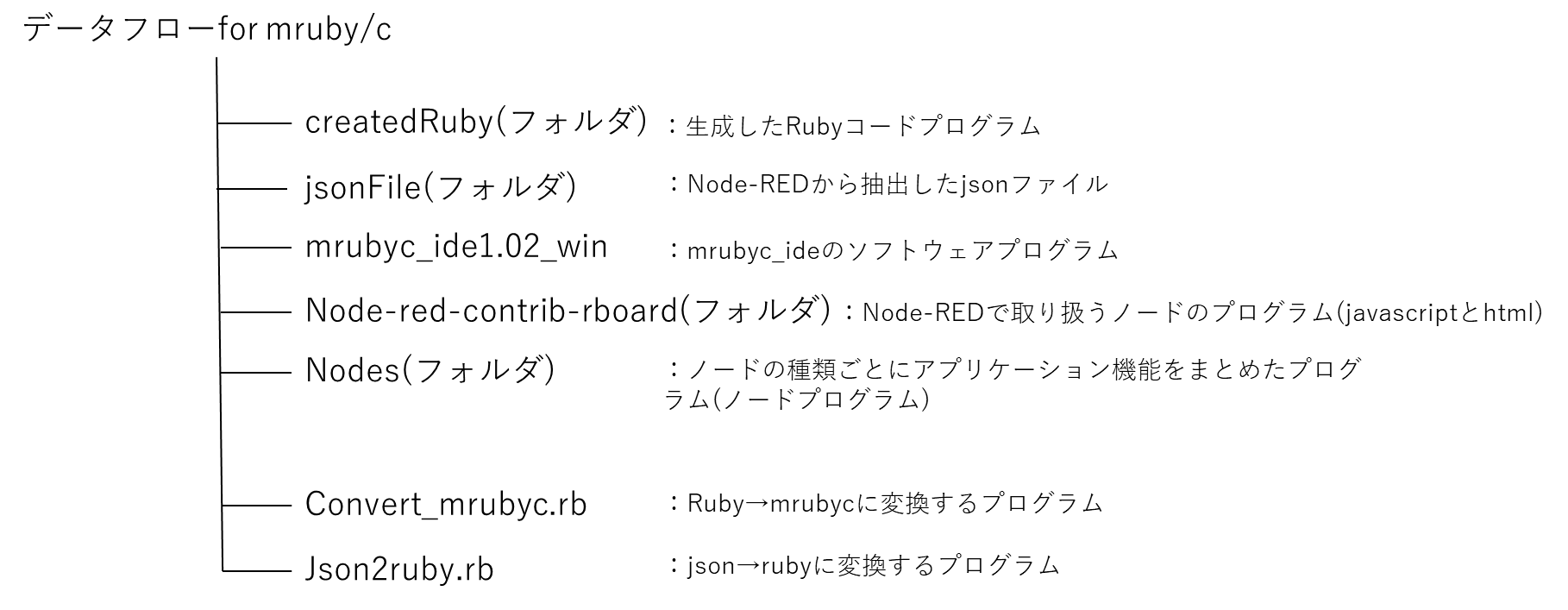
九州工業大学・情報工学部・知的システム工学科・田中研究室　引継ぎ資料

# 本研究の環境構築について

環境構築については、同ファイルにある「データフロー for mruby/c　取扱説明書」を参考にしてほしい。ここで、本システムを実際に体験し、生成されたRubyコードの構成について把握して頂きたい。

# システム構成について

本システムの流れついては、「卒論発表.pptx」にて簡単に解説があるため、確認してほしい。構成については以下の通りである。



ファイルの流れを簡単に説明すると、

1. Node-REDエディタ上でフローを作成し、作成したフロー情報を持つjsonファイルを「jsonFile」フォルダに保存
2. 「Json2ruby.rb」プログラムを実行し、「jsonFile」フォルダにある保存されたjsonファイルを読み込む
3. Jsonファイルを基にRubyコードを自動生成し、「createdRuby」フォルダに格納する
4. ユーザーは生成されたRubyコードを使い、mruby/c変換プログラムやmruby/c　IDEを通してmruby/cコードの入手・マイコンへの実装を行う

このように、Rubyコード生成するプログラムは「json2ruby.rb」であり、Rubyコードの生成ルールを変更したい場合にはこのプログラムを変更させなくてはいけない。また、アプリケーション機能(例えば、LEDを点灯させる等)に修正を加えたい場合には、「Nodes」フォルダ内にある、ノードタイプに対応したノードプログラム(例えばLED.rbやGPIO.rb等)を修正する。しかし、LEDタイプとswitchタイプ、functionタイプのノードに関しては、生成するRubyコード量の削減のため、ユーザーが設定した項目に応じて必要最低限のプログラムのみが生成されるよう仕組みが施されている。これらのノードプログラムを修正する際には、注意してほしい。

# Json2ruby.rbの動作概要

「json2ruby.rb」の大まかな処理の構成として、以下の通り。

1. json2ruby.rbの実行時におけるオプション制御
2. jsonファイルの読み込みと整形、ハッシュ形式へのデータ変換(ノードデータベースの作成)
3. ノードデータベースに基づいたアプリケーション機能のプログラム整理
4. Rubyコードの生成

細かい処理については、ソースプログラム中にコメント文にて記載している。

# ノードを追加したい場合

Node-REDで取り扱うノードの自作については、下記URLにて紹介されている

URL : <https://nodered.jp/docs/creating-nodes/first-node>

[URL : https://qiita.com/zuhito/items/1c65fdbb3743d9f87edd](URL%20:%20https://qiita.com/zuhito/items/1c65fdbb3743d9f87edd)

ここで、Node-REDでのノード自作の流れを把握し、「データフローfor mruby/cファイル＞「Nodes」フォルダ」にある「Format\_NodeType.html」と「Format\_NodeType.js」を活用して自作ノードのプログラムを作成すると効率がよい。ノード自作に関するプログラムを更新した場合は、「npm install」による更新を忘れないように。

本システムで扱う場合の自作ノードのプログラムは、基本的に.htmlファイルを中心に編集している。このhtmlファイルでは、入力ＵＩのレイアウトの追加・変更や、ユーザーによるノードの設定値の管理に関するプログラムが記述されている。これにより、jsonファイル中にあるノードごとのプロパティを自由設定することができる。ここで、簡単な例としてConstantノードを構成するConstant.htmlとConstant.jsを以下に示す。なお、Constantノードはすでに実装されている自作ノードである。

テキスト

自動的に生成された説明

Constant.html

ノードタイプ(ここではConstantタイプ)

複数記述箇所があるので、全て統一すること

ノードタイプ（ここも統一）

入力欄の設定

プロパティ情報を管理できる

Constant.js

テキスト

自動的に生成された説明

（ここも統一）

作成したノードを本システムに適用させ、Rubyコードとして生成できるようにするためには、「データフローfor mruby/c > Nodesフォルダ」内に対応するRubyプログラムを作成する必要がある。Constantノードを作成したのであれば、「Constant.rb」を作成する必要がある。このようなプログラムをノードプログラムと呼んでおり、このプログラムの実装によって、はじめて作成したノードを本システムに適用することができ、Rubyコードとして生成されるようになる。ここで、ノードプログラムであるConstant.rbを以下に示す。

Constant.rb

テキスト

自動的に生成された説明

ユーザーが入力した値を次のノードに送信するため、バッファーに記録

バッファー内から、前ノードからのデータを受け取る

バッファーにある自分宛のデータを削除する

なお、このようなノードプログラムを一般化させたものである「Format\_NodeType.rb」があるので、活用すると良い。

# 残っている課題点について

現在(2021年3月時点)での達成項目は以下の通りである

* データフロープログラミング環境を構築することができた(本システムを介することによって、Node-RED環境上にでのプログラミングでマイコンの制御を行うことができた)
* マイコンボードを制御する上での基本的な機能の実装

課題点としては以下の通りである

* Groveモジュール(センサ)といったモジュール単体をより扱いやすくするためのノードの追加開発
* クラウド上での実装(ブラウザからマイコンボードへのプログラム書き込み機能が一番の壁っぽい…Webserial APIあたりを活用してみたい)
* RBoard以外にも本システムを対応させる

本システムの仕組みを理解していき、これらの課題解決に励んで頂きたい(特にクラウド上での実装)。実装することができれば、プログラミングを勉強したいユーザーに対して、本システムが実装されているクラウドだけを渡してしまえば、めんどくさい環境構築を行わずに済み、データフローによる情報の流れの意識に特化させたプログラミングが行えるようになる。