スマートスピーカーのアプリケーション開発を支援するプログラミング学習環境の開発

本多 佑希[1]　島袋 舞子[1]　浅子 秀樹[2,3]　兼宗 進[1]

概要：スマートスピーカー（AI スピーカー）は AI による音声認識を搭載したスピーカーであり、現在普及が進んでいる。本発表では、スマートスピーカーのプログラムを作成して機能を追加できるプログラミング学習環境を提案する。開発した学習環境では、Web ブラウザ上で JavaScript やドリトルなどで記述することができる。開発したプログラムはサーバーに格納され、LINE 社の Clova から呼び出せるようにした。Clova に向かって話しかけた言葉から、適切なプログラムを起動するように設定することで、処理結果の文字データを音声として発話することが可能になる。本発表ではシステムの概要を紹介し、高等学校の新教科「情報 I」「情報 II」での活用について提案する。

キーワード：スマートスピーカー、 プログラミング教育、 プログラミング環境

Development of a programming learning environment to support application development of smart speaker

1. はじめに

近年、スマートスピーカーの普及が進んでいる。スマートスピーカーは AI による音声認識技術を用いて利用者が発した音声を認識し、適切なアプリケーションを起動してくれるデバイスである。スマートスピーカー自身がネットワークに接続されているため、ネットワークを介したり、Bluetooth を用いたりして様々な機器を操作することもできる。スマートスピーカーによっては、予め用意されているアプリケーションの他に自作アプリケーションを開発することも可能である。

高等学校の新教科「情報 I」では問題解決のために音声認識や AI などのライブラリや API を活用したプログラミングの学習が扱われる [1]。スマートスピーカーは AI を用いた音声認識を行っていることから、スマートスピーカーのアプリ開発はこの「情報 I」の内容の学習に適している。しかし、スマートスピーカーのアプリケーション開発を行うためには、アプリの処理を担当するサーバを用意するか、クラウドサービスを使用する必要があるため、環境構築の敷居が高い。

本研究では、スマートスピーカーの一つである LINE 社の Clova のアプリ開発を支援するプログラミング学習環境を開発した。Clova は標準でバッテリーを内蔵していることから、持ち運びが容易である。授業での利用を考えた際には、本体を移動させるたびに電源を落としたり起動したりするのは手間になるため、利点であると考えられる。この環境は、プログラムの編集やサーバの用意を支援する。プログラムの編集や公開は Web ブラウザ上で動作する専用のプログラム編集環境上で行う。学習者が開発したプログラムは専用のサーバ上に格納される。全ての動作が Web ブラウザ上で完結していることから、利用者は専用のソフトウェアのインストールは不要である。

2. スマートスピーカーのアプリ開発

2.1 スマートスピーカー

スマートスピーカーは音声により操作することができる機器である。利用者が発した音声をもとに音声認識エンジン (AI) がテキストに変換して適切な処理を行う。この音

[1] 大阪電気通信大学
Osaka Electoro-Communication University
[2] 新潟大学
Niigata University
[3] 愛知教育大学
Aichi University of Education

声認識エンジンはサーバ上に存在し、スマートスピーカーはネットワークを介して音声認識エンジンとやり取りを行う。

一般的なスマートスピーカーのシステムのモデルを図 1 に示す。利用者が発した音声データは、スマートスピーカーによってサーバに送信される。サーバは、受け取った音声データを音声認識エンジンによって文字列データに変換した後、文字列データ (例えば「明日の天気は？」や「音楽をかけて」など) によって、アプリケーションの処理を担当するプログラムを起動する。プログラムは Web 上に存在するプログラムを呼び出す形で実行される。プログラムが実行された後、結果をスマートスピーカーが受け取って発話するという流れになる。

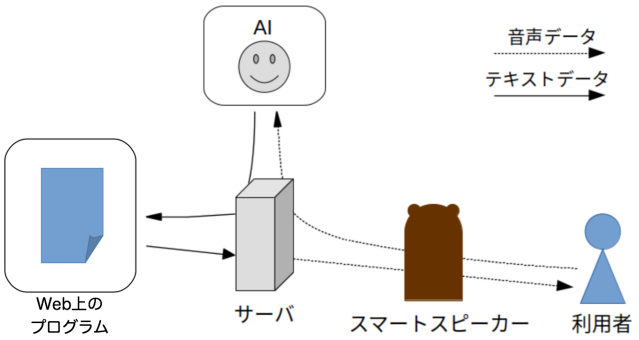

図 1 一般的なスマートスピーカーのシステムモデル

2.2 音声認識の辞書

音声認識を行うエンジンがユーザの発話した音声を認識する際には、開発者が登録した辞書に基づいて認識を行う。この辞書には、一般的にはユーザの発話した内容の意図を認識するための情報と、ユーザが指定する名詞を認識するための情報が含まれている。意図を認識するための情報は一般的にインテント、名詞を認識するための情報は一般的にはスロットやエンティティなどと呼ばれる (本稿ではスロットと呼ぶ)。

例えば表 1 に示すように「明日の天気を教えて」や「2019 年 5 月 1 日の気温を教えて」といった「日付」と「知りたい情報」を元に情報を知りたいといった意図のインテントを想定した場合、「2019 年 5 月 1 日」や「明日」といった日付を認識するスロットと「天気」や「曜日」といった特定のテキストを認識するスロットが必要になる。

表 1 ユーザが発話するテキストの例

日付	-	知りたい情報	-
明日	の	天気	を教えて
2019 年 5 月 1 日	の	気温	を教えて
7 月 15 日	の	曜日	を教えて

2.3 プログラムの開発環境

スマートスピーカーのアプリのプログラムを開発する場合、一般的には各社が提供しているソフトウェア開発キットを用いる。本研究で対象とした Clova の場合、Clova Extension Kit ソフトウェア開発キット (以下、CEK SDK) が提供されており、これを用いて開発を行う。CEK SDK は LINE 社のサーバから送られるリクエストをプログラムから参照しやすい形に整え、アプリ毎に固有な ExtensionID が適切なものかを照合し、サーバに対するレスポンスを適切な形式に変換する。

図 2 に Node.js 向け CEK SDK を用いたアプリ開発のプログラムの例を示す。この例はコイントスを題材にしたアプリの例であり、アプリ起動時に Clova が「コイントスをします。表と裏、どちらが出るでしょうか？」と Clova が発話して、ユーザに回答を求める。ユーザの回答を受けると、乱数によって「当たりです」「外れです」のどちらかを返答するアプリである。11 行目で Clova が認識したインテントの情報、12 行目で Clova が認識したスロットの情報を取得している。9-33 行目ではサーバから送られたリクエストに対するイベント処理を定義しており、ユーザの回答を受けた際のイベント処理はここで定義している。他のリクエストに対するイベント処理を追加することで、より細かなアプリの挙動を設定することができる。

2.4 既存のアプリケーション開発支援サービス

スマートスピーカーのアプリケーション開発を支援するサービスはいくつか存在する。voiceflow [2] や NOID [3] は、ブロックを組み合わせる形で動作を指定しながらアプリケーション開発を行う環境である。本環境はテキストベースのプログラミングでアプリケーションを開発するため、ユーザが開発を行う手法が異なっている。

これらの環境は、ブロックの繋がりによってデータの流れが視覚化されるため、処理がわかりやすい利点がある。本環境では、開発したプログラムの動作を確認する動作確認機能を開発しており、データの流れはこの機能でテストする形で確認する方式を取っている。

voiceflow は GoogleHome や AmazonEcho に、NOID は AmazonEcho に対応している。

3. 本プログラミング学習環境について

3.1 システムの概要

本環境は、LINE 株式会社から販売されている Clova [4,5] のアプリケーション開発の中で、プログラムの開発と Web への公開の部分を支援するものである。

図 3 に、本環境のシステムの概要を示す。図 1 の「ユーザ」「AI」にあたる部分は同様であるため、省略している。本環境は、予め Clova の標準的な方法 [6] でインテントやスロットの辞書データを含めたアプリの情報登録は行なわれていることを想定している。

```
var clovaSkillHandler = clova.Client.
    configureSkill()
  .onLaunchRequest(function(responseHelper){
    responseHelper.setSimpleSpeech({
       lang: 'ja',
      type: 'PlainText',
      value: 'コイントスをします。表と裏、どちらが出るでしょうか？'
    })
  })
  .onIntentRequest(function(responseHelper){
    var sessionId = responseHelper.getSessionId();
    var intentName = responseHelper.getIntentName();
    var slots = responseHelper.getSlots();

    var text;
    var num=Math.floor(Math.random()*2);
    if(num==0)text='表が出ました。';
    else text='裏が出ました。';

    switch(slots['answer']){
      case '表':
        if(num==0)text+='当たりです';
        else text+='外れです';
        break;
      case '裏':
        if(num==1)text+='当たりです';
        else text+='外れです';
        break;
    }
    responseHelper.setSimpleSpeech(
       clova.SpeechBuilder.createSpeechText(text)
    );
    responseHelper.endSession();
  }).handle();
var clovaMiddleware = clova.Middleware({
  applicationId: 'xxx.yyy.zzz'
});
```

図 2 CEK SDK(Node.js) を用いたプログラムの一例

本環境は Node.js で実装し、サーバの立ち上げには Express を使用している。学習者がプログラムを開発する際には Web ブラウザから専用のプログラム編集環境にアクセスする。プログラムの編集、保存、Web への公開など全ての動作を Web ブラウザから行うことができる。

3.2 プログラム編集環境

学習者がプログラムを編集する際には、Web ブラウザ上で動作する専用のプログラム編集環境を用いる。この環境は特定の URI にアクセスするだけで利用することができる。編集環境の画面を図 4 に示す。

図 4 中の中央部がテキストエディタになっており、ここにプログラムを記述する。記述後には、保存ボタンを押してプログラムを保存する。

編集したプログラムは、LINE 社のサーバから呼び出せるように Web 上に公開する必要がある。本環境では、起動ボタンを押すことで開発したプログラムを公開することができる。公開する際には、アプリが実行された時に Clova が話すテキスト (図 5 の「どんな情報を知りたいですか？」や、図 8 の「コイントスをします。表と裏、どちらが出るか当ててください。」に相当する部分) を指定する。

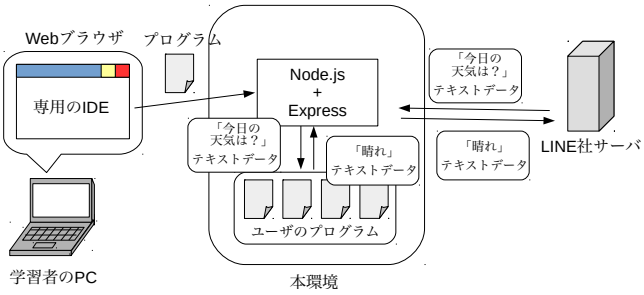

図 3 本環境のシステムの概要図

3.3 プログラムの動作確認

動作確認ボタンから動作のテストを行える。動作確認はチャットのように対話的なやり取りで行うことができる。このやり取りは、Clova で実際にアプリを実行した際の対話を意識している。図 5 に、動作確認機能の実行画面を示す。実行した結果ややり取りの内容は、画面に動的に表示される。

3.4 プログラムの実行

図 3 で示したように、プログラムは LINE 社のサーバからのリクエストを受けて実行される。このリクエストは http の POST リクエストであり、パラメータとしてインテントやスロットの情報が渡される。本環境は、リクエストが送られた URI を解析し、適切なプログラムを子プロセスとして立ち上げて実行する。そして、実行結果をレスポンスとして返すという流れになっている。

3.5 JavaScript でのプログラミング

図 6 に、JavaScript でプログラミングする場合の最も簡単なプログラムを示す。この例のように、main 関数を定義し、その中にプログラムを記述する。後述する「clova.listen」で指定されたインテントを認識した際には、そちらで指定されている処理が実行される。Clova に発話させるためには、「clova.speech」関数にテキストを渡して実行する。また、スロットに関しては、main 関数や clova.listen で設定した関数に、引数としてオブジェクトの形で渡される。例えば、図 6 の環境で「testSlot」という名前のスロットを参照する際には、「`param['testSlot']`」のような形で参照することになる。表 3 に、Clova 用ライブラリの命令一覧を示す。

コイントスのプログラム例を図 7 に示す。この例は図 2

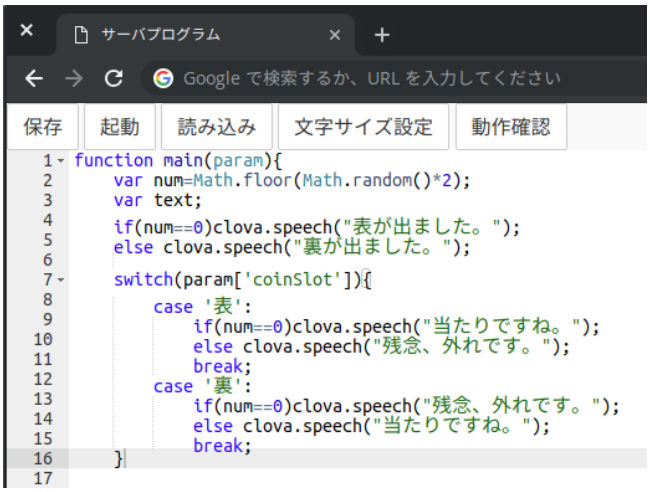

図 4 プログラム編集環境の画面

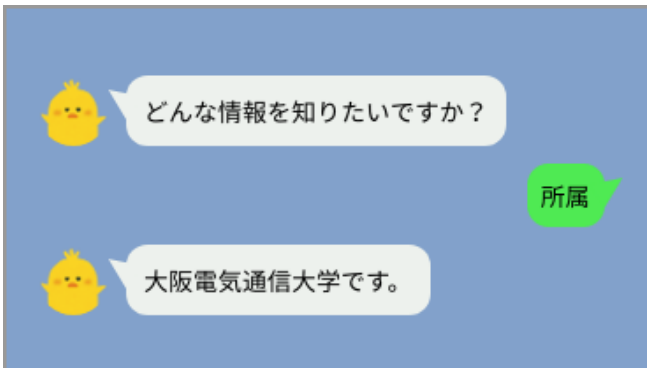

図 5 動作確認機能でテストを行った例

```
function main(param){
  var text="こんにちは";
  clova.speech(text);
}
```

図 6 最も簡単なプログラム (JavaScript)

のプログラムを、本環境向けに修正したものである。「answer」として「表」もしくは「裏」を受け取ることを想定している。また、動作確認機能でこのプログラムをテストした際の例を図 8 に示す。図 2 のプログラムの中でイベント定義や設定を行っている部分が隠蔽され、ユーザの発話した内容を受け取り、発話した内容によって Clova に発話させる内容を決めて発話させる部分に集中することができるようになっている。

図 9 に、図 7 を改良し、複数回コイントスが行えるようにしたプログラム例を示す。Clova がコイントスの答え合わせをした後、「もう一度やりますか？」と問いかけをし、ユーザに「はい」「いいえ」の回答を求める。ユーザが「はい」を発話した場合には、また表と裏のどちらが出るか当てるよう促し、アプリの実行を継続する。「いいえ」を選択した場合には、「終了します。」と発話してアプリを終了する。動作確認機能でこのプログラムをテストした例を図 10 に示す。

表 2 Clova 用ライブラリの命令一覧（JavaScript）

オブジェクト	メソッド	動作
clova	speech	引数を Clova が発話する
clova	listen	第一引数に渡された名前のインテントを認識したら、第二引数に渡された関数を実行するよう設定する
clova	continue	アプリの実行を終了せず、追加でユーザに入力を求める

```
function main(param){
  var num=Math.floor(Math.random()*2);
  if(num==0)clova.speech("表が出ました。");
  else clova.speech("裏が出ました。");

  switch(param['answer']){
    case '表':
      if(num==0)clova.speech("当たりです");
      else clova.speech("外れです");
      break;
    case '裏':
      if(num==0)clova.speech("外れです");
      else clova.speech("当たりです");
      break;
  }
}
```

図 7 コイントスのプログラム (JavaScript)

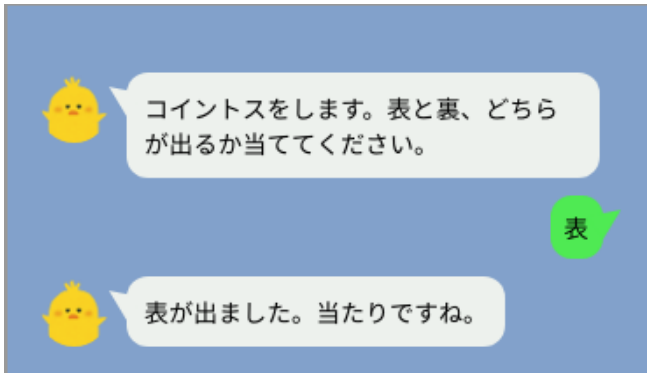

図 8 動作確認機能で図 7 のプログラムをテストした例

図 9 の 1-18 行目、20-25 行目に示すように、`clova.listen('intentName',func)` の形で記述すると、特定のインテントを認識した際の処理を指定することができる。`clova.listen` で指定していないインテントを認識した場合には、通常通り main 関数を呼び出す。また、通常は関数の実行が終わると自動的に Clova のアプリは終了するが、23 行目に示すように `clova.continue()` を実行すると、アプリの実行を継続することができる。この機能を使うことにより、Clova が返答した後に更にユーザに情報を求めることができる。

```
clova.listen('answerIntent',function(param){
  var num=Math.floor(Math.random()*2);
  if(num==0)clova.speech("表が出ました。");
  else clova.speech("裏が出ました。");

  switch(param['answer']){
    case '表':
      if(num==0)clova.speech("当たりです。");
      else clova.speech("外れです。");
      break;
    case '裏':
      if(num==0)clova.speech("外れです。");
      else clova.speech("当たりです。");
      break;
  }
  clova.speech('もう一度やりますか？');
  clova.continue();
});

clova.listen('YesOrNoIntent',function(param){
  if(param['YesOrNoSlot']=='はい'){
    clova.speech('表と裏、どちらが出るか当ててください。');
    clova.continue();
       }else{
    clova.speech('終了します。');
  }
});
```

図 9 複数回プレイできるコイントスのプログラム (JavaScript)

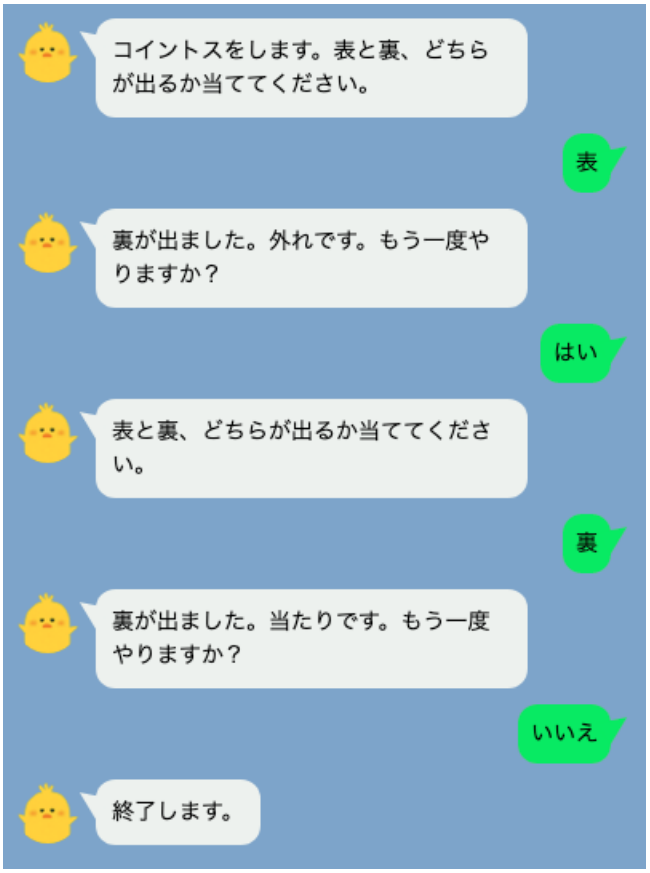

図 10 動作確認機能で図 9 のプログラムをテストした例

3.6 ドリトルでのプログラミング

本環境では、ドリトル [7] でプログラムを開発することもできる。ドリトルのプログラムを JavaScript に変換する変換器 [8] を用いて実現している。

図 11 にドリトルでのプログラミングの最も簡単なプログラムを示す。「クローバー！話す」によって、Clova に発話させることができる。また、スロットに関しては、「クローバー：動作」や「クローバー！聞く」で設定したブロックに、引数としてオブジェクトの形で渡される。例えば、図 11 の環境で「testSlot」という名前のスロットを参照する際には、「`param:testSlot`」のような形で参照することになる。表 3 に、Clova 用ライブラリの命令一覧を示す。

図 12 にプロフィールを教えてくれるプログラムの例を示す。このプログラムを動作確認機能を使ってテストした例は、3.2 章で示した図 5 と同じである。info というスロットに渡されたテキストに対応した言葉を Clova に発話させるプログラムである。

図 13 に四則演算を行ってくれるプログラムの例を示す。Clova は左辺と右辺の値を求めた後、どのような演算を行うかをユーザに問いかける。その後、「足す」「引く」などのテキストによって適切な演算を行い、結果を返す。動作確認機能を使ってテストした例を図 14 に示す。

4. 本環境を用いた学習

4.1 情報 I

学習指導要領の解説の中で、問題解決のためのプログラミングとして「画像認識や音声認識及び人工知能などの既存のライブラリを組み込んだり, API を用いたり [9]」が挙げられているように、高等学校の新教科「情報 I」には音声認識や AI などを題材としたプログラミングが含まれている [1]。本環境ではスマートスピーカーを題材としていることから、既存のライブラリや API を組み合わせて、スマートスピーカー用のアプリを開発することができる。また、音声認識や人工知能を意識させることが可能だと考えている。Clova のアプリがどのように動いているのかを説明する過程で、Clova Developer Center で登録したスロットがどういう役割を果たしているのかや、その裏側で動いている AI のことを考えさせることにより、これらの存在や役割について意識させることが可能であると考えている。

4.2 情報 II

3.2 章で述べたように、本環境では開発したプログラムのテストをサポートする機能も提供している。Clova Developer Center でアプリの概要を設計し、本環境での開発、テスト、実機で運用するというプロセスを通して高等学校の「情報 II」の「情報システムとプログラミング [1]」の学習に活用することができる。Clova Developer Center と、本環境を組み合わせたアプリの開発プロセスは次のように

```
クローバー：動作＝「|param|
  クローバー！"こんにちは" 話す。
」。
```

図 11 最も簡単なプログラム (ドリトル)

表 3 Clova 用ライブラリの命令一覧（ドリトル）

オブジェクト	メソッド	動作
クローバー	話す	引数を Clova が発話する
クローバー	聞く	第一引数に渡された名前の インテントを認識したら、 第二引数に渡されたブロックを 実行するよう設定する
クローバー	継続	アプリの実行を終了せず、 追加でユーザに入力を求める
クローバー	動作	「クローバー！聞く」で 設定されてないインテントを 認識した際に実行される

```
クローバー：動作＝「|param|
  情報＝param:info。
  「情報＝＝"所属"」！なら「
    クローバー！"大阪電気通信大学です。" 話す。
  」そうでなければ「情報＝＝"専攻"」なら「
    クローバー！"制御機械工学専攻です。" 話す。
  」そうでなければ「情報＝＝"誕生日"」なら「
    クローバー！"７月１５日です。" 話す。
  」そうでなければ「
    クローバー！"知らない情報です。" 話す。
  」実行。
」。
```

図 12 プロフィールを教えてくれるプログラム (ドリトル)

```
クローバー！"numberIntent"「|param|
  左辺＝param:left。
  右辺＝param:right。
  クローバー！(左辺＋'と'＋右辺＋'をどうしますか？
      ')話す。
  クローバー！継続。
」聞く。

クローバー！"exprIntent"「|param|
  左辺＝param:left。
  右辺＝param:right。
  演算＝param:expr。
  「演算＝＝"足す"」！なら「
    クローバー！(左辺＋右辺)話す。
  」そうでなければ「演算＝＝"引く"」なら「
    クローバー！(左辺－右辺)話す。
  」そうでなければ「演算＝＝"掛ける"」なら「
    クローバー！(左辺×右辺)話す。
  」そうでなければ「演算＝＝"割る"」なら「
    クローバー！(左辺÷右辺)話す。
  」実行。
」聞く。
```

図 13 四則演算をしてくれるプログラム (ドリトル)

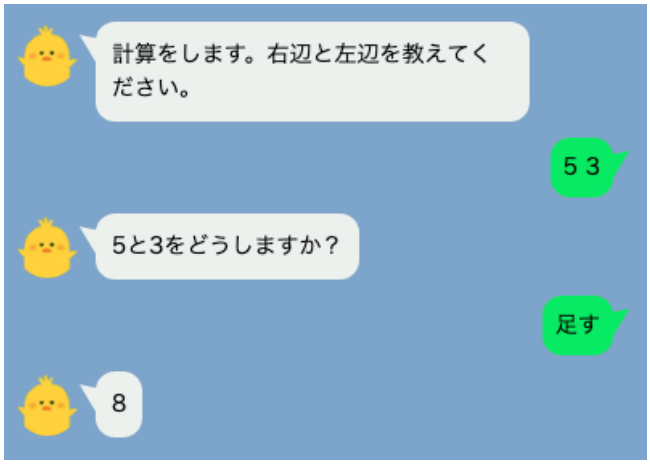

図 14 動作確認機能で図 13 のプログラムをテストした例

なる。

(1) アプリの設計 (Clova Developer Center)
(2) 開発 (本環境のプログラム編集画面)
(3) テスト (本環境の動作確認機能)
(4) 運用 (LINE Clova)

(1) で「どのようなスロットを受け取るか？ユーザにとって使いやすくするにはどうするか？」とったアプリの概要を設計し、(2) で「受け取ったスロットに対して適切な処理を記述」、(3) で「プログラムが想定通り動作しているか？」を確認し、(4) で運用を行うことになる。これらのプロセスを踏んでアプリを開発することによって、実務でのプログラミングに近い学習を行うことができ、より効果的に「情報システムとプログラミング」を学習できることを期待している。

5. まとめ

本研究では、スマートスピーカーである LINE 社の Clova のアプリ開発を支援するプログラミング学習環境を開発した。この環境では、プログラムの開発やサーバの立ち上げ、動作確認等を Web ブラウザ上で容易に行うことができる。サーバの用意が簡略化されたことにより、学習者はより容易に Clova のアプリ開発が行えるようになった。

この環境は高等学校の情報 I の学習に活用できると考えている。また、Clova Developer Center でアプリを設計し、本環境での実装と動作確認機能でのテスト、実機での運用というプロセスを踏むことにより、情報 II の「情報システムとプログラミング」にも活用できると考えている。今後は、実装を進めつつ授業での実践などを踏んで教育効果の評価を行いたい。

参考文献

[1] 文部科学省, 高等学校学習指導要領, 2018.
[2] Voiceflow, Inc.: voiceflow, 入手先〈https://www.voiceflow.com/〉(参照 2019-05-31).
[3] 株式会社アイリッジ: NOID, 入手先

〈https://www.noid.ai/〉(参照 2019-05-31).
[4] LINE 株式会社: LINE Clova, 入手先〈https://clova.line.me/〉(参照 2019-05-31).
[5] Kang, I.: Clova: Services and Devices Powered by AI, *Proc. The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*, pp.1359-1359 (2018).
[6] LINE 株式会社: Clova Developer Center β, 入手先〈https://clova-developers.line.biz〉(参照 2019-05-31).
[7] 大阪電気通信大学 兼宗研究室: プログラミング言語「ドリトル」, 入手先〈http://dolittle.eplang.jp/〉(2018/11/08 参照).
[8] 本多佑希, 長慎也, 大村基将, 久野靖, 並木美太郎, 兼宗進. Dolittle のオンラインプログラミング環境の開発. 情報処理学会 コンピュータと教育研究会, CE(134), pp.1-5, 2016.
[9] 文部科学省, 高等学校学習指導要領 (平成 30 年告示) 解説 情報編, 2018.

