Block Sweetie: Learning Web Application Development by Block Arrangement

Motoki Miura

Department of Basic Sciences, Faculty of Engineering, Kyushu Institute of Technology
1-1 Sensui, Tobata, Kitakyushu, Fukuoka, 804-8550 JAPAN
Email: miuramo@mns.kyutech.ac.jp
Web: https://ist.mns.kyutech.ac.jp/

***Abstract*—In order to build a Web application, understanding of the programming language, databases, SQL, and HTTP is indispensable. We had proposed a PHP-based framework that can facilitate understanding of these concepts and reduce burdens of web applications development for novice learners. However, the novice learner had troubles with syntax errors caused by mistypes. To relief the anxiety of text editing, we have developed Block Sweetie that introduces block-programming editor for the proposed framework. The visual representation and the intuitive operations on block arrangements prevent troubles of input errors and mistypes. We have designed Block Sweetie with "switch and observe" approach, which reduces extra burdens such as saving files and reloading pages. Therefore, novice learners can efficiently perform trial and errors. We have also prepared sample block programs that help novice learners to understand related topics and technologies in a short time. From our preliminary experimental lecture, we found that Block Sweetie was effective for understanding differences of HTTP methods.**

Keywords—Novice Web Developer, Web Framework, Block Editor, Web Application Design

I. Introduction

The World Wide Web is one of the most influential media platforms for sharing knowledge, exchanging ideas, and working collaboratively. People can easily publicize their thoughts, findings and ideas through social networking services. There are several tools and services for collecting and managing user-generated data. For example, SurveyMonkey[1] provides functions of online survey. Google docs and forms also enable the users to create a personalized survey form that is connected to a spreadsheet. These tools and services are helpful for collecting information. However, we consider that everyone should know and have the skill of making customized web application for effective business and sophisticated, creative working process design.

However, recent web technologies for developing web applications have been varied and complicated. Generally, in order to build a simple Web application, understanding of the programming language for dynamic page generation is indispensable. In addition, knowledge of database, SQL, and HTTP is also required. Therefore, for novice learners, the threshold of the Web application is higher than that of the Web page.

[1]https://www.surveymonkey.com/

We have been developed a system named Sweetie Editor [1] and Sweetie Framework [2] that makes it easy to construct a web application on a Web browser. By using the editor and the framework, it is possible for novice learners to understand the overview of simple web applications with built-in samples. Also, thanks to the simple and concise notation of Sweetie Framework, novice learners can develop their web applications with short description. However, the novice learners tend to feel anxiety of text editing from inexperience of text editor (see Fig. 1 left). Also, syntax errors of the source code disrupt the efficiency of learning by trial and errors.

In order to further lower the threshold of building a web application based on the Sweetie Framework for the novice learners, we have developed a web application development system using a block editor.

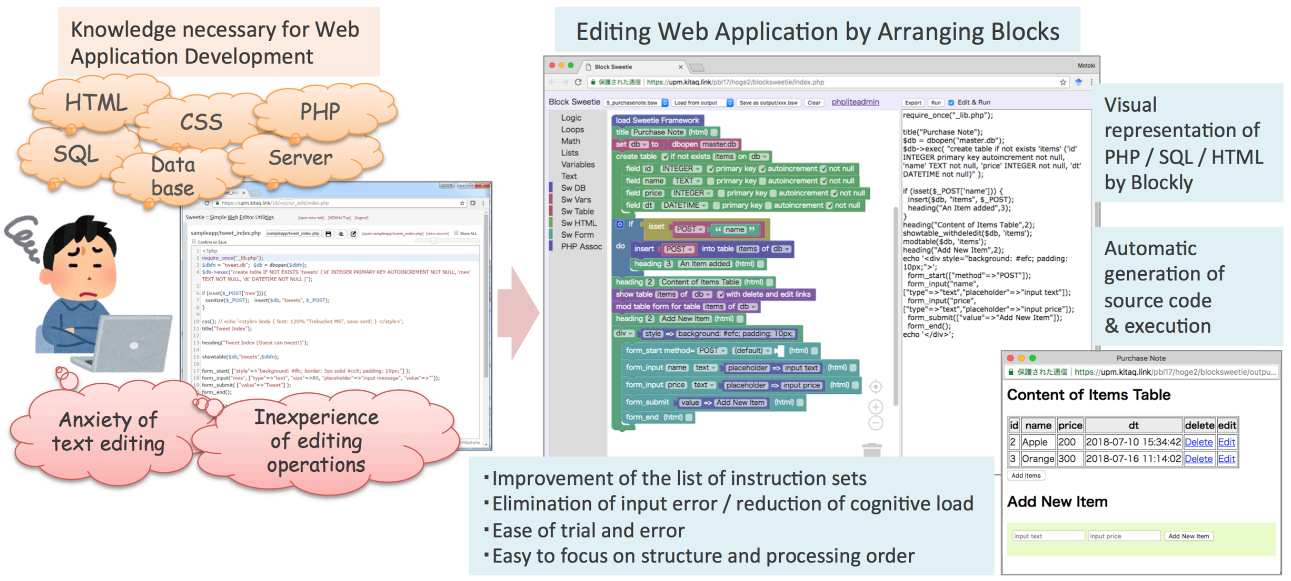
II. Block Sweetie

Block Sweetie is a block-based visual programming editor specialized for simple web application based on Sweetie Framework[2]. It is well known that the block-based programming helps beginners learn more traditional text-based languages[3]. We apply the block-based programming to the web application development.

Block Sweetie basically provides function of editing PHP script. Since the PHP script can contain functions of Sweetie Framework, it is possible to connect SQLite database and creating/managing tables. Fig. 1 right shows the overview of Block Sweetie. Novice learners are expected to open two browser windows for development and learning: one is an editor window for editing PHP script by block arrangement, and the other is an application window for confirming behavior of the web application. When the novice learner rearranges the block or changes a parameter, Block Sweetie system immediately exports corresponding PHP script on the right column of the editor window, and automatically reloads the application window to apply changes of the script. Therefore, the novice learner can easily check and confirm the correspondence between the blocks and the script source code.

A. System Design and Merits

In this section, we describe our system design and its merits to novice learners as well as instructors. The main purpose of introducing block editor is to ease the novice students to trial and errors and have experience of various options of


Knowledge necessary for Web Application Development
HTML
CSS
PHP
SQL
Data base
Server
Anxiety of text editing
Inexperience of editing operations
Editing Web Application by Arranging Blocks
Visual representation of PHP / SQL / HTML by Blockly
Automatic generation of source code & execution
・Improvement of the list of instruction sets
・Elimination of input error / reduction of cognitive load
・Ease of trial and error
・Easy to focus on structure and processing order
Block Sweetie
Export
Run
Edit & Run
Logic
Loops
Math
Lists
Variables
Text
Sw DB
Sw Vars
Sw Table
Sw HTML
Sw Form
PHP Assoc
Purchase Note

```
require_once("_lib.php");

title("Purchase Note");
$db = dbopen("master.db");
$db->exec( "create table if not exists 'items' ('id'
INTEGER primary key autoincrement not null,
'name' TEXT not null, 'price' INTEGER not null, 'dt'
DATETIME not null)" );

if (isset($_POST['name'])) {
  insert($db, "items", $_POST);
  heading("An Item added",3);
}
heading("Content of Items Table",2);
showtable_withdeledit($db, 'items');
modtable($db, 'items');
heading("Add New Item",2);
echo '<div style="background: #efc; padding:
10px;">';
  form_start(["method"=>"POST"]);
  form_input("name",
["type"=>"text","placeholder"=>"input text"]);
  form_input("price",
["type"=>"text","placeholder"=>"input price"]);
  form_submit(["value"=>"Add New Item"]);
  form_end();
echo '</div>';
```

Content of Items Table

id	name	price	dt	delete	edit
2	Apple	200	2018-07-10 15:34:42	Delete	Edit
3	Orange	300	2018-07-16 11:14:02	Delete	Edit

Add items

Add New Item

input text | input price | Add New Item

Fig. 1. (left) conventional text-based editor (right) Block Sweetie: block editor window and application window

web applications with minimum burdens. For learning web application development, novice learners should understand the relationships among several technologies such as HTML, CSS, PHP, and database. In order to understand the relationships, we consider that the novice learner should actively build their own hypotheses, and check them by their subjective operations.

By introducing block notation and block editor to the web development, following merits are provided to the learners.

- Eliminating of syntax errors caused by mistypes.
- Reducing anxiety of editing program.
- Reducing of cognitive load of remembering functions by block list shown on the left column.
- Not necessary to remind orders and meanings of arguments in function call.
- Learners can focus on the structure of the program as well as HTML.

Compared to the text-based programming editor, it is possible to reduce the trouble of text typing. Also, the categorized block list helps the novice learners to find functions. The block list gives hint what is possible on the development environment.

We have designed our system so that when the changes on block editor immediately converts the blocks to the source code, and reflects and updates the application window. The "immediate updating" design brings the following merits.

- Promoting trial and error and quick reviews by the immediate updating/reloading.
- Learners can learn various concepts and knowledge of web application development with minimum burdens.

By adopting the "immediate updating" design, we introduce "switch and observe" approach for improving learning effects. For example, we prepared `<div>` and `<span>` drop-down switch for block/inline elements, and `<ul>` and `<ol>` drop-down switch for unordered/ordered lists. These drop-down switches are expected to find difference of the options. We also introduced "in-html" check-box for understanding of both Sweetie Framework functions and plain HTML/PHP. This design approach is widely appeared on the block definition of the Block Sweetie. From Fig. 11 to Fig. 12 in Appendix show part of our blocks designed for Block Sweetie.

Similar to the design principals of Sweetie Editor, we have developed the Block Sweetie as a web application. We have incorporated the Block Sweetie into the Sweetie Editor. Therefore, it is easy to deploy individual / group working spaces for each learner/group. For that purpose, we have developed Block Sweetie by using Google Blockly [4] which is a versatile block programming editor runs on a web browser. Google Blockly originally provides functions of exporting PHP scripts. We have extended the export functions to generate source code with Sweetie Framework.

III. Block Code Samples

In this part, we describe block code samples of Block Sweetie, and intentions of them.

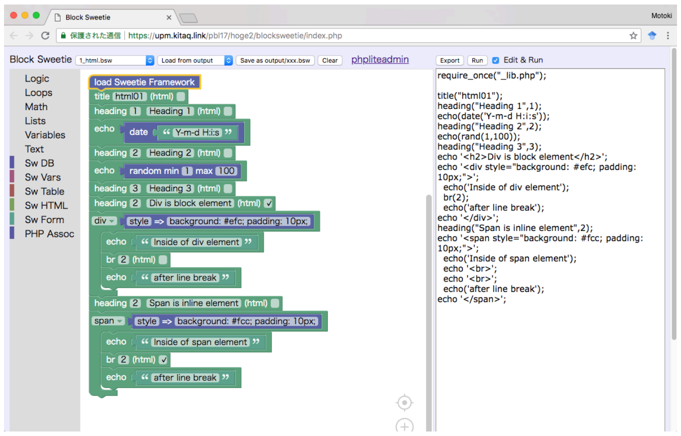
For outline, we have prepared the following block code samples.

- HTML and PHP basics (Fig. 2)
- HTML Table and List (Fig. 3)
- PHP Associate Array (Fig. 4)
- HTML Form (Get and Post methods) (Fig. 5)
- SQLite introduction (Fig. 6)
- Sample Application (Purchase Note) (Fig. 1 right)

These block code samples appears from a drop-down list of Block Sweetie editor. The novice learners can freely edit, modify, and save them for their personalized space.

A. HTML and PHP basics

Fig. 2 shows HTML and PHP basics. In this sample, we introduce title(), heading(), and br() functions provided by Sweetie Framework as well as echo, date, and random functions of PHP. The Sweetie Framework (SwF) functions provide "in html (html)" check-box. When the check-box is clicked, the corresponding SwF function is expanded. For example, the last br(2) block outputs two `<br>` by echo. The `<div>` and `<span>` blocks can optionally specify a PHP array that is converted to their attributes. In this sample, the novice learner can change the background color by CSS notation.

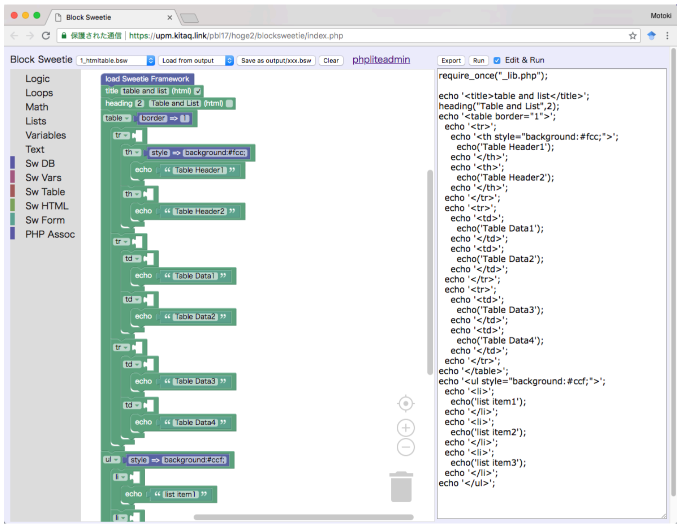


```
require_once("_lib.php");

title("html01");
heading("Heading 1",1);
echo(date('Y-m-d H:i:s'));
heading("Heading 2",2);
echo(rand(1,100));
heading("Heading 3",3);
echo '<h2>Div is block element</h2>';
echo '<div style="background: #efc; padding:
10px;">';
  echo('Inside of div element');
  br(2);
  echo('after line break');
echo '</div>';
heading("Span is inline element",2);
echo '<span style="background: #fcc; padding:
10px;">';
  echo('Inside of span element');
  echo '<br>';
  echo '<br>';
  echo('after line break');
echo '</span>';
```

Fig. 2. HTML and PHP basics sample

B. HTML Table and List

Fig. 3 shows a complicated HTML sample with a table and a list. In this sample, we expect the novice learners to understand the structure of the table and the list with the visualized nesting. Note that the `<ul>` drop-down provides alternative `<ol>`, and `<th>` provides `<td>`, respectively.

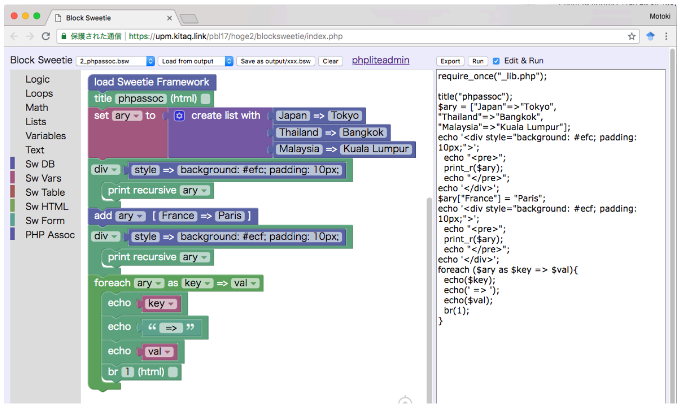


```
require_once("_lib.php");

echo '<title>table and list</title>';
heading("Table and List",2);
echo '<table border="1">';
  echo '<tr>';
    echo '<th style="background:#fcc;">';
      echo('Table Header1');
    echo '</th>';
    echo '<th>';
      echo('Table Header2');
    echo '</th>';
  echo '</tr>';
  echo '<tr>';
    echo '<td>';
      echo('Table Data1');
    echo '</td>';
    echo '<td>';
      echo('Table Data2');
    echo '</td>';
  echo '</tr>';
  echo '<tr>';
    echo '<td>';
      echo('Table Data3');
    echo '</td>';
    echo '<td>';
      echo('Table Data4');
    echo '</td>';
  echo '</tr>';
echo '</table>';
echo '<ul style="background:#ccf;">';
  echo '<li>';
    echo('list item1');
  echo '</li>';
  echo '<li>';
    echo('list item2');
  echo '</li>';
  echo '<li>';
    echo('list item3');
  echo '</li>';
echo '</ul>';
```

Fig. 3. HTML Table and List sample

C. PHP Associate Array

Fig. 4 shows how to use and handle associate array of PHP. The initial pair of array can be defined by [create list with] block. We provide "foreach" statement to handle array data. The output of the script is shown in Fig. 9.

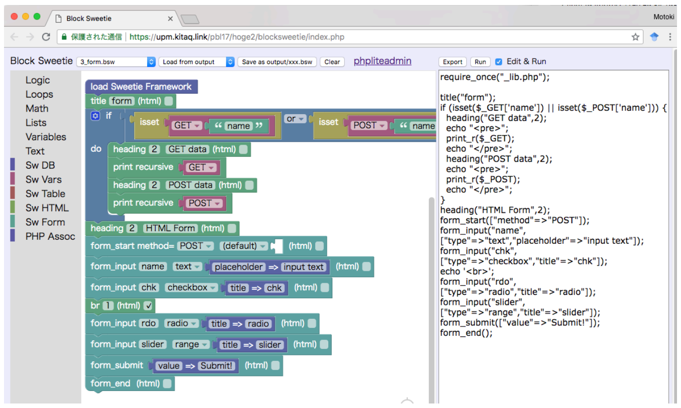


```
require_once("_lib.php");

title("phpassoc");
$ary = ["Japan"=>"Tokyo",
"Thailand"=>"Bangkok",
"Malaysia"=>"Kuala Lumpur"];
echo '<div style="background: #efc; padding:
10px;">';
  echo "<pre>";
  print_r($ary);
  echo "</pre>";
echo '</div>';
$ary["France"] = "Paris";
echo '<div style="background: #ecf; padding:
10px;">';
  echo "<pre>";
  print_r($ary);
  echo "</pre>";
echo '</div>';
foreach ($ary as $key => $val){
  echo($key);
  echo(' => ');
  echo($val);
  br(1);
}
```

Fig. 4. PHP associate array sample

D. HTML Form (Get and Post methods)

Fig. 5 shows a sample of simple HTML Form. The script accepts both GET and POST methods. The novice learner can understand the difference of the methods by selecting method of [form_start] block, and how the form data is passed to the PHP script. The [form_input] block allows the novice learner to select various input types such as text, password, file, check-box, radio, and range.



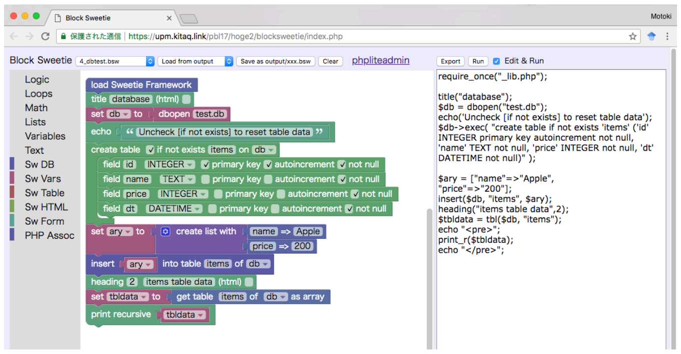
```
require_once("_lib.php");

title("form");
if (isset($_GET['name']) || isset($_POST['name'])) {
  heading("GET data",2);
  echo "<pre>";
  print_r($_GET);
  echo "</pre>";
  heading("POST data",2);
  echo "<pre>";
  print_r($_POST);
  echo "</pre>";
}
heading("HTML Form",2);
form_start(["method"=>"POST"]);
form_input("name",
["type"=>"text","placeholder"=>"input text"]);
form_input("chk",
["type"=>"checkbox","title"=>"chk"]);
echo '<br>';
form_input("rdo",
["type"=>"radio","title"=>"radio"]);
form_input("slider",
["type"=>"range","title"=>"slider"]);
form_submit(["value"=>"Submit!"]);
form_end();
```

Fig. 5. HTML Form (Get and Post methods) sample

E. SQLite introduction

Fig. 6 provides a simple example of creating/inserting DB table. We have designed the table definition as the [create table] and [field] blocks. When the novice learner uncheck the "if not exists" on the [create table] block, the table data is discarded. The [field] block provides data types of integer, text, datetime, real, blob, and image as drop-down list. The novice learner can select the data types by simple operation.

In the former Sweetie Editor, learners had to write SQL statements directly for preparing SQLite tables. Block Sweetie


Block Sweetie
Motoki
保護された通信 | https://upm.kitaq.link/pbl17/hoge2/blocksweetie/index.php
Block Sweetie
4_dbtest.bsw
Load from output
Save as output/xxx.bsw
Clear
phpliteadmin
Export
Run
Edit & Run
Logic
Loops
Math
Lists
Variables
Text
Sw DB
Sw Vars
Sw Table
Sw HTML
Sw Form
PHP Assoc
load Sweetie Framework
title database (html)
set db to dbopen test.db
echo " Uncheck [if not exists] to reset table data "
create table ✓ if not exists items on db
field id INTEGER ✓ primary key ✓ autoincrement ✓ not null
field name TEXT primary key autoincrement ✓ not null
field price INTEGER primary key autoincrement ✓ not null
field dt DATETIME primary key autoincrement ✓ not null
set ary to create list with name => Apple price => 200
insert ary into table items of db
heading 2 items table data (html)
set tbldata to get table items of db as array
print recursive tbldata

```
require_once("_lib.php");

title("database");
$db = dbopen("test.db");
echo('Uncheck [if not exists] to reset table data');
$db->exec( "create table if not exists 'items' ('id'
INTEGER primary key autoincrement not null,
'name' TEXT not null, 'price' INTEGER not null, 'dt'
DATETIME not null)" );

$ary = ["name"=>"Apple",
"price"=>"200"];
insert($db, "items", $ary);
heading("items table data",2);
$tbldata = tbl($db, "items");
echo "<pre>";
print_r($tbldata);
echo "</pre>";
```

Fig. 6. SQLite introductory sample

reduces the errors of editing SQL statements by interlocking blocks. Also, attributes of the primary key, auto-increment, not null can be easily set with check-boxes. Therefore, errors in SQL statements can be reduced, and novice learners can focus on the meaning and differences of the options.

Incidentally, the novice learner can check the database table by opening phpLiteAdmin tool[5] from the top link. This function allows the learner to confirm the result of the execution, as well as the management of data in detail.

F. *Sample Application (Purchase Note)*

Fig. 1 right shows a complete sample application of purchase note. The application is composed with the blocks and its usage appeared in the previous samples. Therefore, we expect the novice learners to modify and arrange the block sample by their own skills and knowledge.

Sweetie Framework provides `showtable()` function which displays database table data in HTML table format. Also, `modtable()` function inserts special Javascript snippet for modifying table row data. These functions are useful and convenience, however, the internal mechanism is currently hidden from the learners. We consider that the definition of such functions should be represented as blocks for further improvement.

IV. Preliminary Experimental Lecture

We conducted a preliminary experimental lecture. Five 3rd-year undergraduate students attended the lecture. The students were interested in web application development, and three of them already have experiences of HTML and Ruby on Rails.

We explained the basics of web application development with PHP by showing the samples explained in section III and modifying them. We instructed several features derived from original Google Blockly such as zooming and enable/disable blocks. The explanation time was 40 minutes. We also provided PCs for each student, and the students could operate Block Sweetie. After the explanation, we asked the following open-ended questions. The students answered them through a web form.

1) Describe what you learned about the PHP associate array.
2) Describe what you learned about HTML form and its data processing by PHP.
3) Describe what you learned about the database (SQLite) and how to use it.
4) Describe what you learned about the development of the web application using PHP. (You can compare with other methods such as Rails.)
5) List as much as possible of good points and bad points (improvement points) about this tool.

A. *Results and Comments*

For the first questions, the students answered as follows.

- I knew it could store keys and values.
- I understood the data structure of PHP. It is similar to hash in Ruby and dictionary in Python. I felt high readability compared to index call (note: foreach statement).
- I felt that PHP is more complicated writing style than Rails.
- Although it was harder to see than ruby, I felt it was created immediately by changing the contents of the block and submitting it.
- It is possible to display data collectively on one line (note: show_table() function).

For the second questions, the students answered as follows.

- Regarding GET and POST, I have not known the difference. I thought POST was the only method for sending data. I could know the actual differences and preferences of GET and POST.
- Differences between GET and POST.
- There is a difference between sending form by GET or sending by POST.
- There are GET and POST methods for receiving data, and the size that can be stored in GET is small.
- I did not understand the difference between POST and GET deeply, so I think the lecture was useful.

For the third questions, the students answered as follows.

- Many descriptions such as null and primary key that are written in db/migate/* files in Rails were seen. Also interaction with the database was similar. I did not have any knowledge about SQL, so I decided to study about that.
- SQLite manages data as a table.
- A database can be created by executing some codes.
- You can save one file as a database.
- I learned the way of PHP something like Rails.

For the fourth questions, the students answered as follows.

- Both of them are frameworks for generating HTML and Javascript code, whereas Rails describes actions by dividing it into three categories, Model, View, and Controller, the framework (Sweetie) is described in one file. I was able to understand it intuitively. I could also compare with HTML, so I felt effective for novice learners.

- I learned that I could create web applications in languages other than ruby (PHP).
- PHP notation is directly linked with the processed output. I think that the point is different by compared with Ruby.
- It was difficult to distinguish processing than Rails. I felt that it was similar but differences were variables and function names.
- I think it will take time for both Rails and PHP to build from the basic elements of actions, views, and databases. But I felt that the basic way to do was the same.

For the fifth questions, we summarized good points and bad points respectively.

Good points:

- It is possible to describe behavior very intuitively. It can also be used for HTML learning.
- Easy to understand structure.
- Looks easy.
- Because it is divided into blocks for each sentence, I felt it was easy to debug.

Bad points:

- Block tends to become large, readability and maintainability are likely to deteriorate accordingly (It will be solved by folding function like vim).
- When you get used to it, text may be easier than block.
- The block program was hard to see.
- The user might be able to write program without understanding deeply.

B. *Discussion*

We could confirm that most of the student recognized the meaning, structure, and functions of HTML, CSS, PHP, and SQLite, even though the lecture time was less than one hour. Of course, we should consider that three students already had experiences of HTML and Ruby on Rails. They could relate and compare with their knowledge and skills. However, we found that all students did not know the difference and usage of GET/POST method, and they could learn them properly by the lecture.

In future work, we will perform the experimental lecture for students who have no knowledge and skills about web development.

V. Related works and systems

The impact of block-based interface on introductory programming is well known[3]. There are many software tools to adopt block editor interface for novice programming. For example, Scratch[6], Google Blockly[4], and Pencil code[7] are famous educational tools for novice programmers. Arakliotis et al. applied the block interface for Arduino programming[8]. We have adopted the block interface for introductory web application development.

There are several tools and services for creating/developing web application with simple operations on graphical UIs. Bubble[9] incorporates interface builder for placing components such as buttons, lists, input forms, and map views. The user develops the web application by relating them. Web Performer [10] is a tool to automatically generate Web applications from GUI. Web performer is built as an Eclipse IDE plugin. The user can develop full-fledged web applications by saving burdens. These tools reduce the cost and efforts for developing practical web applications. However, these tools hide the source codes and details for simplicity and abstraction. Block Sweetie was designed to show both the blocks and the corresponding source codes. By comparing the blocks and the codes, we expect the learners to understand not only the fundamental concepts of web application but also the detail of programming and technology.

Nagataki et al. developed a web-based learning tool for database education called sAccess [11]. sAccess focuses on introductory computer science education for high school and college students. sAccess provides block-based query building interface for manipulating database tables, and allows the students to learn query building without mastering complex syntax. sAccess automatically shows the query results for each step. This feature is superior to Block Sweetie because the learner should insert blocks explicitly for confirmation of results. On the other hand, Block Sweetie covers wider topics for web application development.

VI. Conclusion

In order to promote web application development for all students, we have introduced block interface for our Sweetie Framework, previously proposed for novice learners with text-based programming. Block Sweetie lowers the threshold of building a web application by eliminating anxiety of text editing and unexpected syntax errors by mistypes. The modification by the block is immediately reflected to the output. Therefore, the novice learner can perform trial and errors without any burdens of saving source codes, exporting files, and reloading a web page.

Since we have introduced a “switch and observe” approach for designing blocks, the novice learner can check the differences and options with minimum operations. For example, check boxes and drop-down lists reduce the troublesome as well as modification time.

Through the preliminary experimental lecture, we could confirm that Block Sweetie is effective for understanding basic concepts of HTTP methods and PHP syntax. We also found that the initial quick tour of the web application development could be presented within one hour. We will continue to verify the effect of the Block Sweetie through experimental lectures for novice students.

Block Sweetie can be downloaded from the following URL. https://github.com/miuramo/blocksweetie

References

[1] Motoki Miura. Lightweight Web Authoring Environment. In *20th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES2016)*, Vol. 96, pp. 887–895, September 2016.

[2] Motoki Miura. Sweetie Framework: Simple but Practical Web Application Development Environment. In *12th International Conference on Knowledge, Information and Creativity Support System*, pp. 149–154, November 2017.

[3] David Bau, Jeff Gray, Caitlin Kelleher, Josh Sheldon, and Franklyn Turbak. Learnable programming: blocks and beyond. *Communications of the ACM*, Vol. 60, No. 6, pp. 72–80, 2017.

[4] Neil Fraser. Ten Things We've Learned from Blockly. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)*, pp. 49–50, October 2015.

[5] phpLiteAdmin: The Web-based Database Management Tool for SQLite. http://www.phpliteadmin.org/. (Visited: July 20, 2018.).

[6] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, et al. Scratch: programming for all. *Communications of the ACM*, Vol. 52, No. 11, pp. 60–67, 2009.

[7] David Bau, D Anthony Bau, Mathew Dawson, and C Pickens. Pencil code: block code for a text world. In *Proceedings of the 14th International Conference on Interaction Design and Children*, pp. 445–448. ACM, 2015.

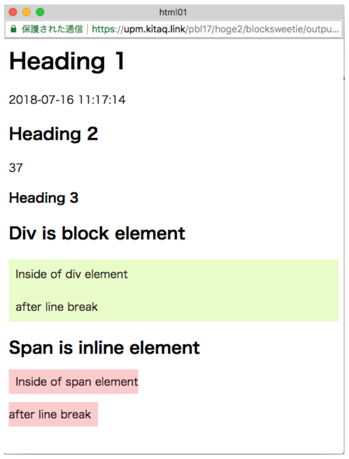
[8] S Arakliotis, DG Nikolos, and E Kalligeros. Lawris: A rule-based arduino programming system for young students. In *2016 5th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1–4. IEEE, 2016.

[9] Bubble Group Inc. Bubble: Visual Programming. https://bubble.is/. (Visited: Dec 24, 2017.).

[10] Canon IT Solutions. Web Performer. https://www.canon-its.co.jp/products/web_performer/. (Visited: Dec 24, 2017.).

[11] Hiroyuki Nagataki, Yoshiaki Nakano, Midori Nobe, Tatsuya Tohyama, and Susumu Kanemune. A Visual Learning Tool for Database Operation. In *Proceedings of 8th Workshop in Primary and Secondary Computing Education (WiPSCE 2013)*, pp. 40–41, November 2013. http://saccess.eplang.jp/.

Appendix


html01

保護された通信 | https://upm.kitaq.link/pbl17/hoge2/blocksweetie/outpu...

Heading 1

2018-07-16 11:17:14

Heading 2

37

Heading 3

Div is block element

Inside of div element

after line break

Span is inline element

Inside of span element

after line break

Fig. 7. Output of Fig. 2: HTML and PHP basics

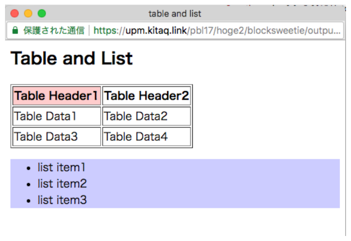


table and list

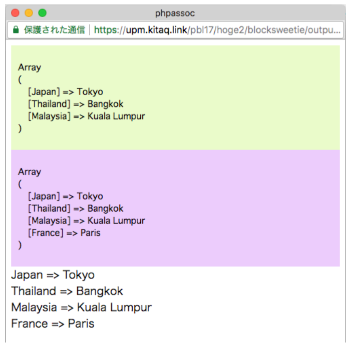
保護された通信 | https://upm.kitaq.link/pbl17/hoge2/blocksweetie/outpu...

Table and List

Table Header1	Table Header2
Table Data1	Table Data2
Table Data3	Table Data4

- list item1
- list item2
- list item3

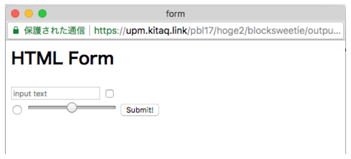
Fig. 8. Output of Fig. 3: HTML Table and List


phpassoc

保護された通信 | https://upm.kitaq.link/pbl17/hoge2/blocksweetie/outpu...

```
Array
(
    [Japan] => Tokyo
    [Thailand] => Bangkok
    [Malaysia] => Kuala Lumpur
)

Array
(
    [Japan] => Tokyo
    [Thailand] => Bangkok
    [Malaysia] => Kuala Lumpur
    [France] => Paris
)
Japan => Tokyo
Thailand => Bangkok
Malaysia => Kuala Lumpur
France => Paris
```

Fig. 9. Output of Fig. 4: PHP Associate Array


form

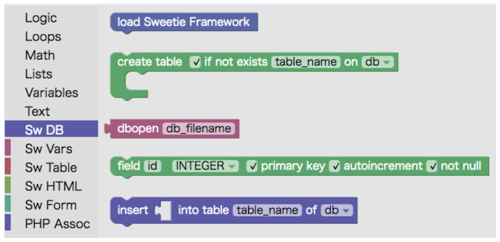
保護された通信 | https://upm.kitaq.link/pbl17/hoge2/blocksweetie/outpu...

HTML Form

input text

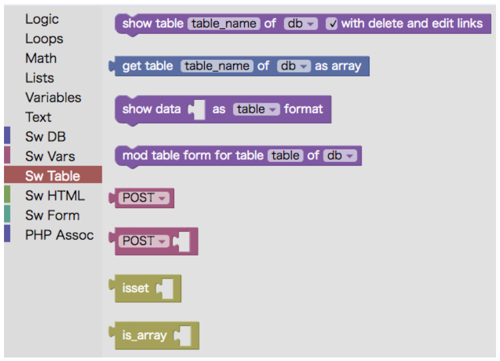
Submit!

Fig. 10. Output of Fig. 5: HTML Form



Logic
Loops
Math
Lists
Variables
Text
Sw DB
Sw Vars
Sw Table
Sw HTML
Sw Form
PHP Assoc
load Sweetie Framework
create table ✓ if not exists table_name on db
dbopen db_filename
field id INTEGER ✓ primary key ✓ autoincrement ✓ not null
insert into table table_name of db

Fig. 11. DB Operation Blocks



Logic
Loops
Math
Lists
Variables
Text
Sw DB
Sw Vars
Sw Table
Sw HTML
Sw Form
PHP Assoc
show table table_name of db ✓ with delete and edit links
get table table_name of db as array
show data as table format
mod table form for table table of db
POST
POST
isset
is_array

Fig. 12. Table Output Blocks