Easysurvey

--- an online survey generate language

PLT Project Language Reference Manual

Team Members:

Taotao Li  tl2453
Luyao Shi  ls2899
Zaisheng Dai  zd2140
Yifan Zhang  yz 2365

Professor:

Stephen A. Edwards
# TABLE OF CONTENTS

1. Language description .................................................................3
2. Input and output ........................................................................3
3. Lexicon .......................................................................................3
   3.1 Token separator .....................................................................3
   3.2 Comments ...........................................................................3
   3.3 Constants ............................................................................3
   3.4 Identifier ............................................................................3
   3.5 Keywords ............................................................................4
   3.6 Separators ...........................................................................5
   3.7 Operators ............................................................................6
      3.7.1 ASSIGN ......................................................................6
      3.7.2 Dot .............................................................................6
      3.7.3 Comparison Operator ......................................................6
   3.8 Data Types ...........................................................................6
   3.9 Attributes ...........................................................................7
   3.10 Embedded Action .................................................................8
4. Declaration ..................................................................................9
   4.1 QuestionSet Declaration .........................................................9
   4.2 Other datatype declaration ......................................................9
5. Function definition .......................................................................9
6. Statements ..................................................................................10
   6.1 Common statements ............................................................10
   6.2 Block statements ..................................................................10
   6.3 If statements ......................................................................10
7. Expressions ................................................................................10
   7.1 Simple expressions ...............................................................10
   7.2 Assignment expressions .......................................................10
   7.3 Special expressions ..............................................................11
   7.4 Call expressions ..................................................................11
   7.5 Condition expressions ..........................................................12
8. Scope .........................................................................................12
9. Conditionals ...............................................................................12
10. Code snippets ............................................................................13
11. Syntax summary .........................................................................15
1. Language Description

Online survey can find its importance in areas such as Market Research, Event Planning, Customer Feedback, Product Planning and Education & Training. EasySurvey is a lightweight language which is designed to make the online survey generation process easy and fun. EasySurvey defines a framework that can allow developers to define basic components of the online survey. Users can use “EasySurvey language” to define “Page” to represent every survey page. Each page can have several “QuestionSet”, each “QuestionSet” can have some numbers of “Question”. Then each question contains its own properties, such as “Title” and “Type” etc.

2. Input and output

EasySurvey will accept the files with .as extension as the input file; the EasySurvey compiler will compile the input file, which will be analyzed in lexically and grammatically. If everything is correct, the compiler will generate the flex page source code (.mxml). Then the output source code can be run and tested under Adobe Flash Builder. Developers can export the application as a package which can be run in any flash-support environment.

3. Lexicon

3.1 Token separator

White space “ “, New line ‘\n’, Carriage return ‘\r’ and Horizontal tab ‘\t’ are the token separators.

3.2 Comments

Comments begins with “/*” and end with “*/”, including everything between them.

3.3 Constants

Integer: a sequence of one or more digits.

String: a string starts with a double quote “ followed by zero or more characters, ended by a double quote

3.4 Identifier

An identifier is defined as a combination of alphanumeric characters [a-z][AZ][0-9] and must start with a alphabet character. Length of an identifier cannot exceed 32.
This language is case sensitive.

3.5 Keywords

if
else
int
new
Main
Question
Title
SingleSelection
MultipleSelection
DropdownList
TextField
UserInput
QuestionSet
Display
AddQuestion
setVisible
Image:
JumpButton:
3.6 Separators

3.6.1 ';' 

';' is used to separate statements.

For example:

```java
Question question1;

QuestionSet qs1 = new QuestionSet();
```

3.6.2 '{' and '}'

'{' and '}' are used to separate sets like function declarations, gather the statements in if block or else block, or gathering parameters when assigning the content of the question.

For example:

1. QuestionLogo (QuestionSet qs1, Image logo)
   ```java
   Main()
   { } (separating function declarations)
   ```

2. if(qs1.q1.UserInput>18)
   ```java
   { }
   else { } (gather statements in if block or else block)
   ```

3. Question Address;
   ```java
   Address.Title="Where do you live?";
   Address.SingleSelection={"Bronx","New Jersy","China"}; (assigning the content of the question)
   ```

3.6.2 ','

',' is used to separate the parameters and identifier.

For example: The third case above.

3.6.2 '(' and ')'

'(' and ')') are used to indicate the token before '(', is a function and gather the function parameters. Also, they can be used to contain the condition sentence of if.

3.7 Operators

3.7.1 ASSIGN

'=' is used to represent assignment, usually used when constructing a new QuestionSet, setting the Title of the question or the content of the question.

For example:

```java
question1.Title=“question1”;
question1.SingleSelection= {“1”, “2”};
QuestionSet questionset1=new Questionset();
```

3.7.2 Dot

'.' is usually behind a union type like QuestionSet or Question, following the elements of that union type.

For example:

```
“question1.Title”, “questionset1.quesstion1.UserInput”
```

3.7.3 Comparison Operator

The comparison operator in EasySurvey includes equal, less than, greater than, less equal and greater equal which are presented as <, >,<=,>=,== associating from left to right.

Constraint: the left hand side of the operator must be userinput. 7.5 show the definition of comparison operator.

3.8 Attributes

**Title** :

Title is one of the attribute of the question. It is a string which will show the informant what the questions are.

**SingleSelection:**
the informant can only choose one of the answers.

**MultipleSelection:**

The informant can choose multiple answers.

**DropdownList:**

The answers will represent in the form of dropdown list.

**TextField:**

The informant should type his/her answer in text field.

**UserInput:**

represent the value user input, every question has one UserInput

### 3.9 Data Types

**int:**

An integer is a sequence of digits. All the integer constants are views as decimal.

Sample:

```c
int y;
```

**Question:**

*Question* is used to define the survey questions. It consists of *Title* and *Type* which might be one of *RadioBox, DropdownList, MultipleBox, and TextArea.*

Sample:

```c
Question Name;

Name.Title="What's your name?";

Name.TextArea={10}; /* the TextArea occupy 10 characters*
```

**QuestionSet:**
**QuestionSet** is used to define a set of related question and display them as a whole part.

Sample:

```java
QuestionSet qs1 = new QuestionSet();
```

**Image:**

Claim an image for the use of displaying it.

**JumpButton:**

Jump to next or another page.

**3.10 Embeded Action**

**Display:**

Display a series of **QuestionSet** or **Image**.

Sample:

```java
Display Image1; Display Qs1;
```

**AddQuestion():**

Add a **Question** to a **QuestionSet**

Sample:

```java
qs1.AddQuestion(Name)
```

**setVisible():**

set the visibility of a **QuestionSet**. The parameter should be integer, while 0 represent that the question would be hidden, the other integer would show the questionset.

Sample:

```java
Qs1.setVisible(1)  /*Qs1 would now be visible*/
```
4. Declaration

\[
decl:
\]

\[
\text{QUESTIONSET ID ASSIGN NEW QUESTIONSET LPAREN RPAREN SEMI}
\]

\[
\text{| datatype var\_decl SEMI}
\]

4.1 QuestionSet Declaration

The declaration of \textit{QuestionSet} has the form

\[
\text{QUESTIONSET ID ASSIGN NEW QUESTIONSET LPAREN RPAREN SEMI}
\]

For example:

\[
\text{QuestionSet ID =new QuestionSet( );}
\]

4.2 Other datatype declaration

Other datatype declaration has the form

\[
\text{datatype var\_decl SEMI}
\]

The \textit{var\_decl} contains a list of variable names. The \textit{datatype} has the form

\[
\text{datatype:}
\]

\[
\text{QUESTION \ | IMAGE \ | JUMPBUTTON \ | INT}
\]

For example: \textit{Question q1, q2, q3;}

5. Function definition

The function definition has the form

\[
\text{fun\_def:}
\]

\[
\text{ID LPARAN args\_list RPARAN LBRACE decl\_list stmt\_list RBRACE}
\]

For example:

\[
\text{QuestionLogo(} \textit{QuestionSet q1, Image logo)}
\]
The args_list is a list of arguments.

For example: int a, Question q1, QuestionSet qs1

The decl_list is a list of declarations.

The stmt_list is a list of statements.

6. Statements

6.1 Common statements

The common statements are composed with expression and a semicolon.

6.2 Block statements

The block statements have the form

```plaintext
stmt:
    LBRACE stmt_list RBRACE
```

stmt_list is a list of statements.

6.3 If statements

If statements have the form

```plaintext
stmt:
    IF LPAREN expr RPAREN stmt %prec NOELSE
    | IF LPAREN expr RPAREN stmt ELSE stmt
```

7. Expressions

7.1 Simple expressions

The simple expressions include int constant or string constant and return their value.
7.2 Assignment expressions

The assignment expressions have the form

\[ expr: \]

\[
\text{ID POINT MULTIPLE ASSIGN LBRACE } \text{string_opt} \text{ RBRACE}
\]

(Assign the content of the question ID, noting the type of the question is \textit{MultipleSelection})

\[
\text{ID POINT SINGLE ASSIGN LBRACE } \text{string_opt} \text{ RBRACE}
\]

(Assign the content of the question ID, noting the type of the question is \textit{SingleSelection})

\[
\text{ID POINT DROP ASSIGN LBRACE } \text{string_opt} \text{ RBRACE}
\]

(Assign the content of the question ID, noting the type of the question is \textit{DropdownBox})

\[
\text{ID POINT TEXT ASSIGN LBRACE } \text{INT_LITERAL} \text{ RBRACE}
\]

(Assign the width of the text question)

7.3 Special expressions

Adding a \textit{Question} to a \textit{QuestionSet} is a special expression. It has the form

\[ expr: \]

\[
\text{ID POINT ADDQUESTION LPARENT } \text{var_decl} \text{ RPARENT}
\]

The \textit{var_decl} is a list of variables(ID).

Displaying a series of \textit{QuestionSet} is a special expression. It has the form

\[ expr: \]

\[
\text{DISPLAY } \text{var_decl}
\]

Setting the visibility of a QuestionSet is a special expression. It has the form

\[ expr: \]

\[
\text{ID POINT SETVISABLE LPARENT } \text{INT_LITERAL} \text{ RPARENT}
\]

7.4 Call expressions

They have the form

\[ expr: \]
7.5 Condition expressions

They have the form

\[ \text{expr:} \]

\[
\begin{align*}
&\text{ID POINT ID POINT USERINPUT EQ expr} \\
&\text{ID POINT ID POINT USERINPUT NEQ expr} \\
&\text{ID POINT ID POINT USERINPUT LT expr} \\
&\text{ID POINT ID POINT USERINPUT LEQ expr} \\
&\text{ID POINT ID POINT USERINPUT GT expr} \\
&\text{ID POINT ID POINT USERINPUT GEQ expr}
\end{align*}
\]

Normally, to get a \textit{UserInput}, the developer should provide the \textit{UserInput} of which \textit{Question} in which \textit{QuestionSet}. That’s why the form has two “ID POINT”.

8. Scope

Global variables have a global scope. The scope of other variables is in their function declaration block.

9. Conditionals

Conditionals are used to determine the order of questions shown to readers. They are defined in the following form.

\textbf{If} (expression of \textit{UserInput}) \textbf{statement} \textbf{else} \textbf{statement}

The else part is necessary, because it will decide which \textit{questionset} to be \textit{display}.

For example:

\begin{verbatim}
if ( qs1.IsOnCampus.UserInput == "OnCampus")
{
    qs2.setVisible(1);
}
else {
    qs3.setVisible(1);
}
\end{verbatim}
10. Code snippet:

/******function declaration******/

QuestionLogo(QuestionSet Qs1, Image imag1)
{
    Display Qs1,
    Display image1;
}

/******main function******/

Main()
{
    /*variable declaration*/

    Question Name;
    Question Address;
    Question IsOnCampus;
    Question Fruit;
    Question OnCampus;
    Question OffCampus:

    Image logo;

    /*******Assign value to variation***********/

    logo = "C:\EasySury\image1x.jpg";

    Name.Title = "What's your name?";
    Name.TextField = (10);

    IsOnCampus.Title = "Are you living on Campus?";
    IsOnCampus.SingleSelection = "off campus","on campus";

    Address.Title = "Where do you live?";
    Address.SingleSelection = "Bronx","New Jersy","China";

    Fruit.Title = "Which is/are your favorite fruit?";
    Fruit.MultipleSelection = "Apple", "Pear", "Banana";
OnCampus.Title="Which area?";
OnCampus.SingleSelection{"MorningSide","WestWood"};

OffCampus.Title="Which district?";
Offcampus.SingleSelection{"Manhattan", "Queens", "Bronx", "Brooklyn"}

/******* questionsets declaration and assignment******/

QuestionSet qs1=new Questionset();
qs1.AddQuestion(Name, Address, Fruit);
qs1.AddQuestion(IsOnCampus);

/******* display the questionset **********/
Display qs1;

/******* function call **************/

QuestionLogo(qs1, logo);

QuestionSet qs2=QuestionSet();
qs2.AddQuestion(OnCampus);
QuestionSet qs3=QuestionSet();
qs3.AddQuestion(OffCampus);

/* Condition example, we will base on the users' input to show the next input. 
  If the informant live on campus, we will display the questionset related to live on campus. */

if (qs1.IsOnCampus.UserInput == “OnCampus”){
    qs2.setVisible(1);
}
else {
    qs3.setVisible(1);
}

}
11. Syntax Summary

\[
\text{program:} \quad \begin{array}{l}
| \text{program decl} \\
| \text{program fun_def}
\end{array}
\]

\[
\text{fun_def:} \ ID \ LPARAN \ \text{args_list} \ RPARAN \ LBRACE \ decl_list \ stmt_list \ RBRACE
\]

\[
\text{args_list:} \quad \begin{array}{l}
| \text{def_args} \\
| \text{args_list COMMA def_args}
\end{array}
\]

\[
\text{def_args:} \ \text{datatype ID} | \text{QUESTIONSET ID}
\]

\[
\text{decl_list:} \quad \begin{array}{l}
| \text{decl_list decl} \\
| \text{var_opt decl}
\end{array}
\]

\[
\text{decl:} \quad \begin{array}{l}
| \text{QUESTIONSET ID ASSIGN NEW QUESTIONSET LPARAN RPAREN} \\
| \text{datatype var_decl SEMI}
\end{array}
\]

\[
\text{var_opt:} \quad \begin{array}{l}
| \text{var decl} \\
| \text{var}
\end{array}
\]

\[
\text{var Decl:} \quad \begin{array}{l}
| \text{var} \\
| \text{var Decl COMMA var}
\end{array}
\]

\[
\text{var: ID}
\]

\[
\text{datatype:} \quad \begin{array}{l}
| \text{QUESTION} \\
| \text{IMAGE} \\
| \text{JUMPBUTTON} \\
| \text{INT}
\end{array}
\]

\[
\text{stmt:} \quad \begin{array}{l}
| \text{expr SEMI} \\
| \text{LBRACE stmt_list RBRACE} \\
| \text{IF LPARAN expr RPARAN stmt} \ %\text{prec NOELSE} \\
| \text{IF LPARAN expr RPARAN stmt ELSE stmt}
\end{array}
\]

\[
\text{expr:} \quad \begin{array}{l}
| \text{INT LITERAL} \\
| \text{STRING LITERAL} \\
| \text{ID POINT MULTIPLE ASSIGN LBRACE string_opt RBRACE} \\
| \text{ID POINT SINGLE ASSIGN LBRACE string_opt RBRACE} \\
| \text{ID POINT DROP ASSIGN LBRACE string_opt RBRACE} \\
| \text{ID POINT TEXT ASSIGN LPAREN INT LITERAL RPAREN}
\end{array}
\]
| ID POINT TITLE ASSIGN STRING_LITERAL 
| ID ASSIGN STRING_LITERAL 
| ID POINT ADDQUESTION LPARENT var_decl RPARENT 
| ID POINT SETVISABLE LPARENT INT_LITERAL RPARENT 
| DISPLAY var_decl 
| ID LPAREN var_opt RPAREN 
| ID POINT ID POINT USERINPUT EQ expr 
| ID POINT ID POINT USERINPUT NEQ expr 
| ID POINT ID POINT USERINPUT LT expr 
| ID POINT ID POINT USERINPUT LEQ expr 
| ID POINT ID POINT USERINPUT GT expr 
| ID POINT ID POINT USERINPUT GEQ expr 

string_list:  
STRING_LITERAL 
| string_list COMMA STRING_LITERAL 

string_opt: 
| string_list 

stmt_list: 
| stmt_list stmt