Using a Figure Drawing Language

P.S!

Tutorial

28th Nov 2016

Contents

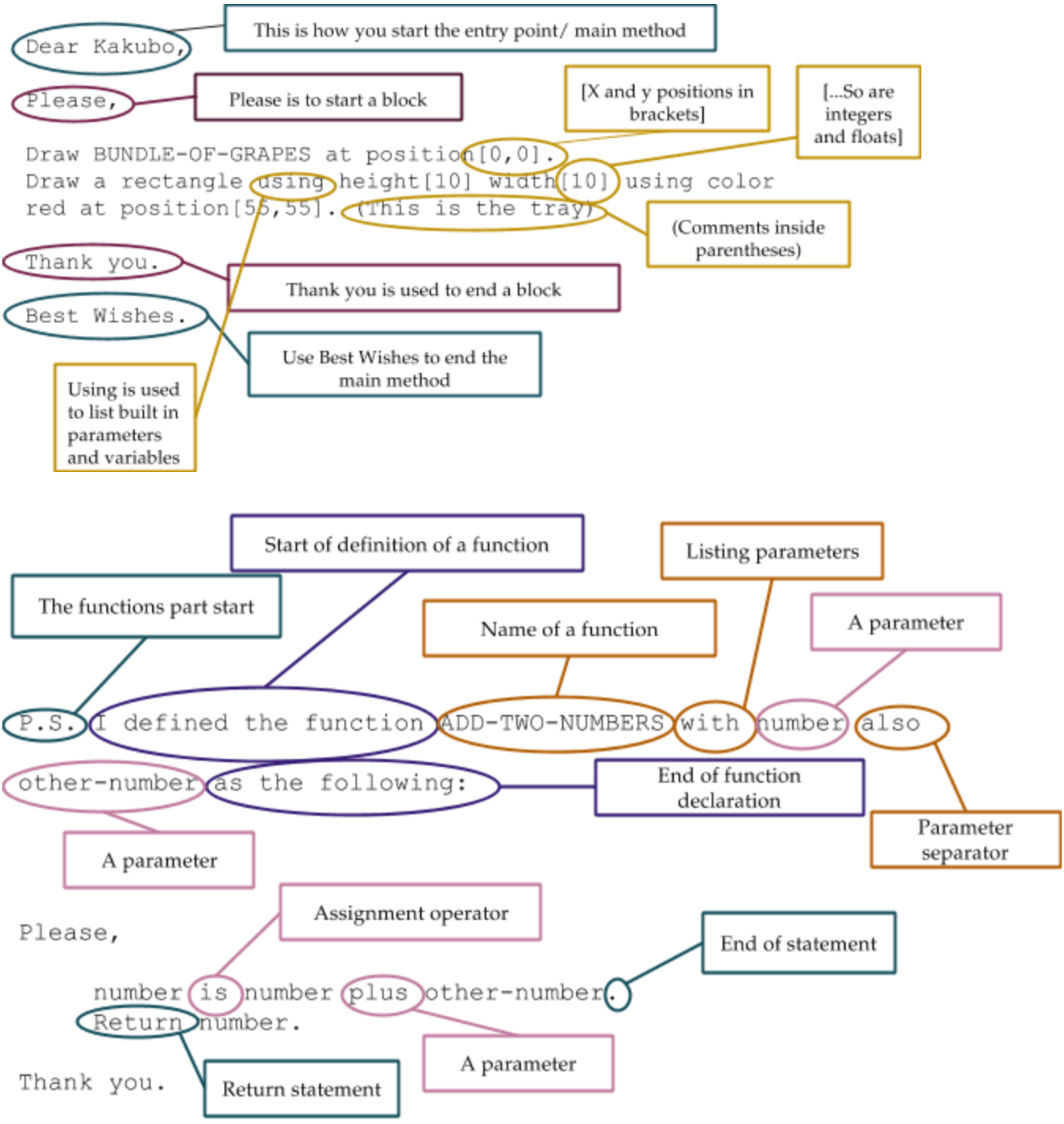
1 | Overview

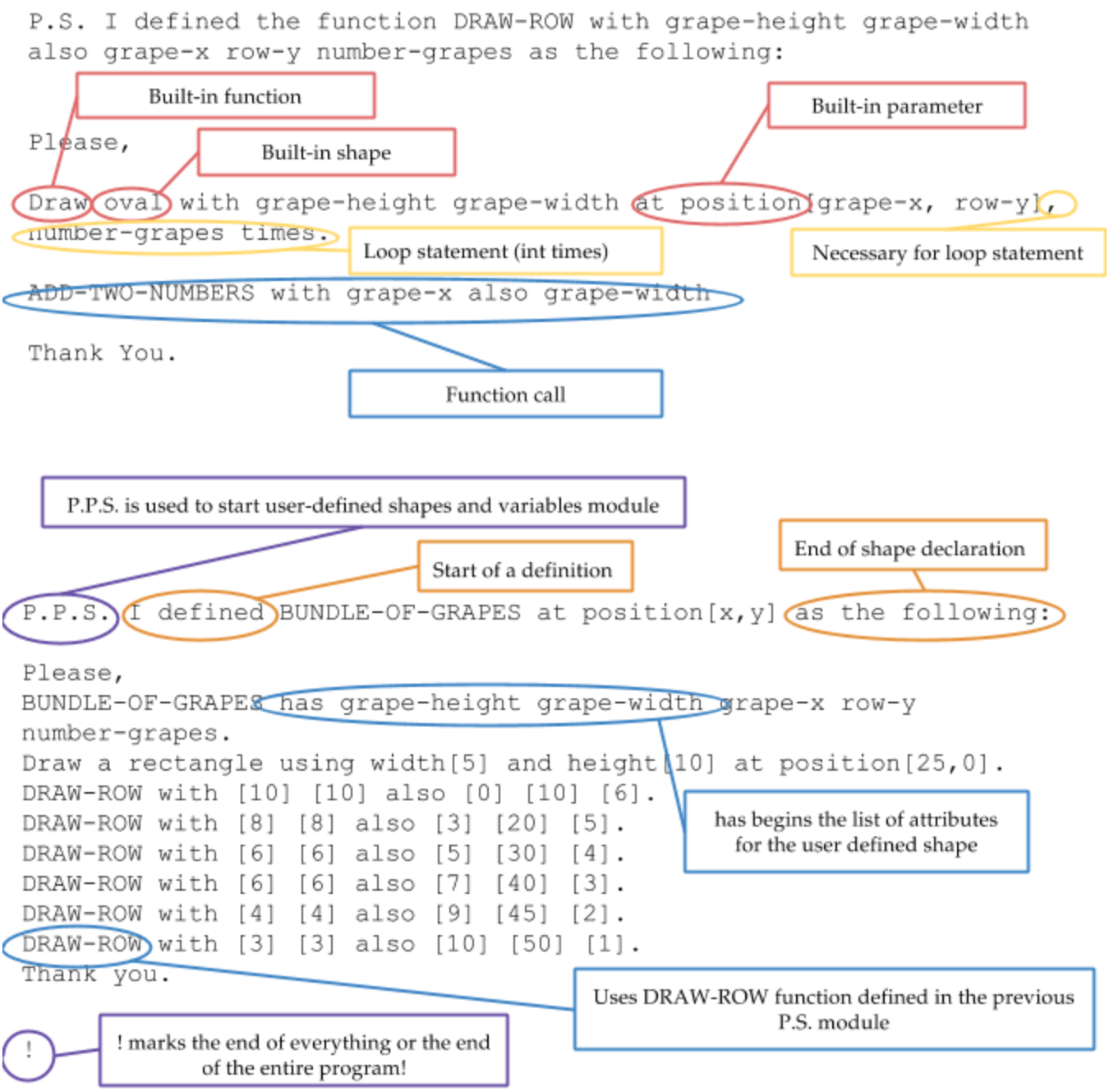
In this tutorial we will be going over a sample program and discuss it in detail using diagrams.

2 | Getting Started

Here is the sample program we are going to use for this tutorial. This sample program draws a simple bundle of grapes.

```
Dear Kakubo,

Please,

Draw BUNDLE-OF-GRAPES at position[0,0].
Draw a rectangle using height[10] width[10] using color red at position[55,55].
(This is the tray)

Thank you.
Best Wishes.

P.S. I defined the function ADD-TWO-NUMBERS with number also
other-number as the following:

Please,
number is number plus other-number.
Return number.

Thank you.

P.S. I defined the function DRAW-ROW with grape-height grape-width also grape-x
row-y number-grapes as the following:

Please,

Draw oval with grape-height grape-width at position[grape-x, row-y], number-grapes
times.
ADD-TWO-NUMBERS with grape-x also grape-width

Thank You.

P.P.S. I defined BUNDLE-OF-GRAPES at position[x,y] as the following:

Please,
BUNDLE-OF-GRAPES has grape-height grape-width grape-x row-y number-grapes.
Draw a rectangle using width[5] and height[10] at position[25,0].
DRAW-ROW with [10] [10] also [0] [10] [6].
DRAW-ROW with [8] [8] also [3] [20] [5].
DRAW-ROW with [6] [6] also [5] [30] [4].
DRAW-ROW with [6] [6] also [7] [40] [3].
DRAW-ROW with [4] [4] also [9] [45] [2].
DRAW-ROW with [3] [3] also [10] [50] [1].

Thank you.
!
```


Dear Kakubo,
This is how you start the entry point/ main method
Please,
Please is to start a block
[X and y positions in brackets]
[...So are integers and floats]
Draw BUNDLE-OF-GRAPES at position[0,0].
Draw a rectangle using height[10] width[10] using color red at position[55,55]. (This is the tray)
(Comments inside parentheses)
Thank you.
Thank you is used to end a block
Best Wishes.
Use Best Wishes to end the main method
Using is used to list built in parameters and variables
Start of definition of a function
Listing parameters
The functions part start
Name of a function
A parameter
P.S. I defined the function ADD-TWO-NUMBERS with number also
other-number as the following:
End of function declaration
Parameter separator
A parameter
Please,
Assignment operator
End of statement
number is number plus other-number.
Return number.
A parameter
Thank you.
Return statement


P.S. I defined the function DRAW-ROW with grape-height grape-width also grape-x row-y number-grapes as the following:
Built-in function
Built-in parameter
Please,
Built-in shape
Draw oval with grape-height grape-width at position[grape-x, row-y],
number-grapes times.
Loop statement (int times)
Necessary for loop statement
ADD-TWO-NUMBERS with grape-x also grape-width
Thank You.
Function call
P.P.S. is used to start user-defined shapes and variables module
End of shape declaration
Start of a definition
P.P.S. I defined BUNDLE-OF-GRAPES at position[x,y] as the following:
Please,
BUNDLE-OF-GRAPES has grape-height grape-width grape-x row-y
number-grapes.
Draw a rectangle using width[5] and height[10] at position[25,0].
DRAW-ROW with [10] [10] also [0] [10] [6].
DRAW-ROW with [8] [8] also [3] [20] [5].
DRAW-ROW with [6] [6] also [5] [30] [4].
DRAW-ROW with [6] [6] also [7] [40] [3].
DRAW-ROW with [4] [4] also [9] [45] [2].
DRAW-ROW with [3] [3] also [10] [50] [1].
Thank you.
has begins the list of attributes for the user defined shape
Uses DRAW-ROW function defined in the previous P.S. module
!
! marks the end of everything or the end of the entire program!

The Core Part

In order to write the source code, the user need to know a few essentials. The main code, like main function in Java, will be written between `Dear Kakubo` and `Best Wishes` (with or without a dot). `Please` and `Thank you` specify blocks, like {...} in Java. The user defined functions section go after `Best Wishes` and start with `P.S.` Each function will have the following format:

```
I defined the function FUNCTION-NAME as the following:

Please,
________________________________________________
________________________________________________
Thank you.
```

If the function has some parameters passed to and have some value to return, it will look like following:

```
I defined the function FUNCTION-NAME with ID as the following:

Please,
________________________________________________
________________________________________________
Give back ID.
Thank you.
```

To return a value the reserved words `return` or `give back` can be used followed by a variable or value. The parameter grammar will be discussed later.

The user-defined shapes are defined after P.P.S. and will have the following format:

```
I defined SHAPE-NAME as the following:

Please,
________________________________________________
________________________________________________
Thank you.
```

It may have a name to reference later, otherwise, it might be quickly drawn without a name and not be able to reference it later. Details of the user-defined shape section will be discussed later.

The source code need to be ended by the `!`. If the course code doesn't have a `P.S.` and `P.P.S.` part, after `Best Wishes` write `!`. If the program have `P.S.` section, but `P.P.S.` is not provided, then put `!` after `P.S.` If the program doesn't have a `P.S.` but `P.P.S.` after `Best Wishes`, write `P.P.S.` part and end it with `!`.