Turtle Graphics

Before continuing with loops, we are going to learn how to create graphical output with the **Turtle Graphics** library. Like a pencil on paper, the Turtle object leaves a line as it moves around the screen.

Turtle Syntax

The first step is to create a Turtle object to move around the screen.

```
Turtle tina = new Turtle(); //creates a Turtle object called tina
```

Here are some basic commands to use with `tina` the Turtle object.

Command	Parameter	Description
`tina.forward(n)`	Where `n` represents the number of pixels	Move the turtle forward
`tina.backward(n)`	Where `n` represents the number of pixels	Move the turtle backward
`tina.right(d)`	Where `d` represents the number of degrees	Turn the turtle to the right
`tina.left(d)`	Where `d` represents the number of degrees	Turn the turtle to the left

Turtle Commands

Let's try this very simple command below. Copy it into the text editor on your left and then click the `TRY IT` button to see the graphical output.

```
Turtle tina = new Turtle(0, 100); //change parameters to make tina visible
tina.forward(100);
```

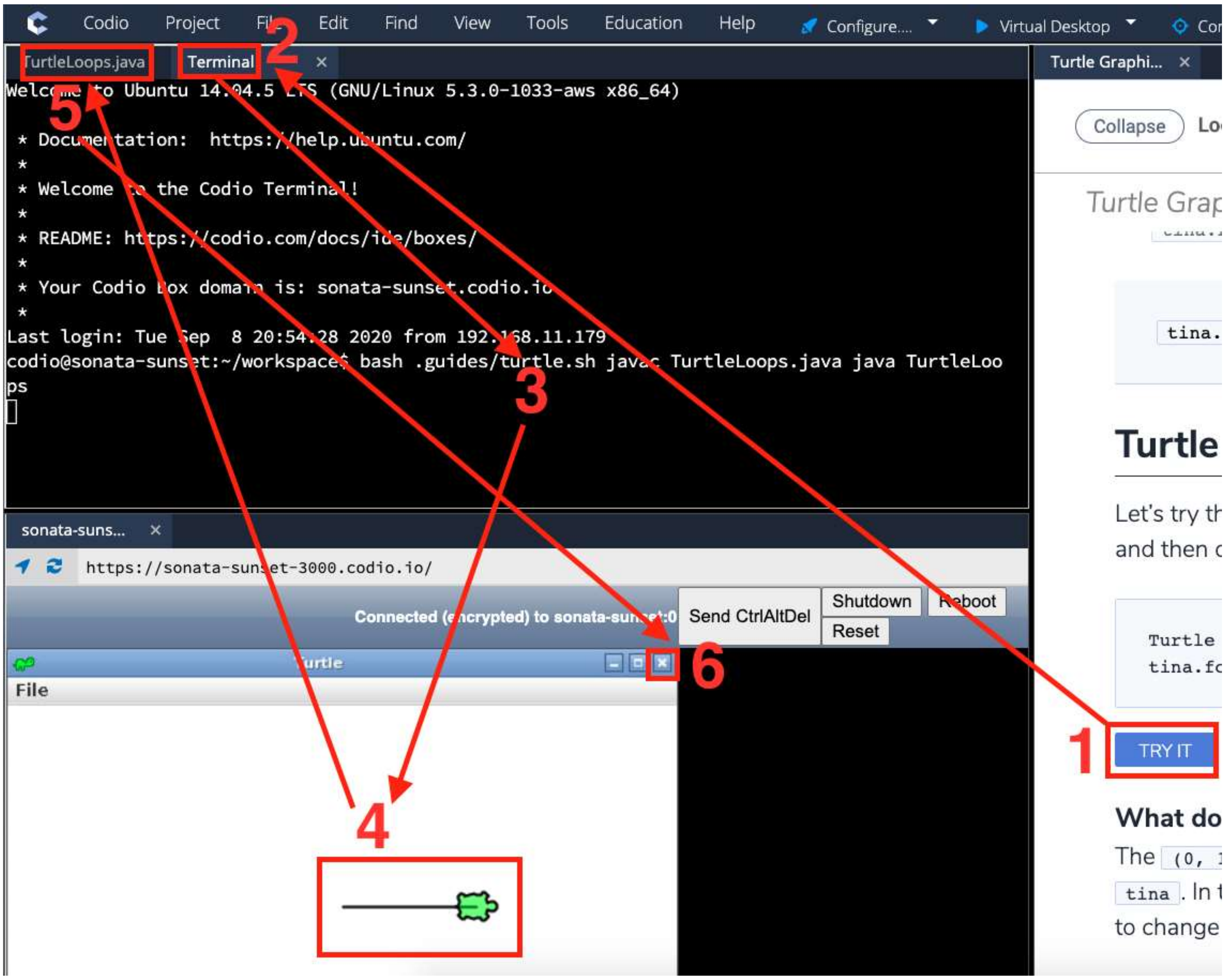
TRY IT

What does the `(0, 100)` inside `Turtle()` do?

The `(0, 100)` inside `Turtle()` enables you to set the latitude and longitude of `tina`. In the example, `tina` starts at `0` pixel latitude and `100` pixels longitude. Feel free to change these parameters so that `tina` the Turtle can be seen on your screen.

Turtle Output

Below is an image highlighting what happens after the `TRY IT` button is clicked.



```
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 5.3.0-1033-aws x86_64)

 * Documentation:  https://help.ubuntu.com/
 *
 * Welcome to the Codio Terminal!
 *
 * README: https://codio.com/docs/ide/boxes/
 *
 * Your Codio Box domain is: sonata-sunset.codio.io
 *
Last login: Tue Sep  8 20:54:28 2020 from 192.168.11.179
codio@sonata-sunset:~/workspace$ bash .guides/turtle.sh javac TurtleLoops.java java TurtleLoo
ps
```

1. `TRY IT` button is clicked by the user.
2. The `Terminal` tab is opened.
3. The terminal runs the command to compile the program and to display the graphical output.
4. The output is displayed as a canvas on the bottom left panel.
5. Click on the `TurtleLoops.java` tab to go back to the text editor if you want to make changes to the program.
6. Click on the `x` icon to close the canvas and exit the program. Alternatively, you can also press the `Ctrl` and `z` keys (Windows) or the `control` and `z` keys (Mac).

Recognizing For Loop Pattern

Given the following code snippet:

```
tina.forward(100);
tina.right(90);
tina.forward(100);
tina.right(90);
tina.forward(100);
tina.right(90);
tina.forward(100);
tina.right(90);
```

Select **all** of the following that will produce the same output as the code above using a `for` loop?

```
for (int i = 0; i < 4; i++) {
  tina.forward(100);
  tina.right(90);
}
```

```
for (int i = 1; i < 4; i++) {
  tina.forward(100);
  tina.right(90);
}
```

```
for (int i = 1; i < 5; i++) {
  tina.forward(100);
  tina.right(90);
}
```

```
for (int i = 20; i < 24; i++) {
  tina.forward(100);
  tina.right(90);
}
```

All of the choices above are correct **except** choice #2. It's important to recognize that the `tina.forward(100);` and `tina.right(90);` commands occur exactly **four** times. Thus, a loop that iterates those commands four times is needed. While all of the choices have a loop header that runs four times, choice #2 has a header that only runs **three** times and is therefore incorrect.