Chaining if-else Statements

This lesson explains how if-else statements can be chained together to achieve more complex coding goals.

WE'LL COVER THE FOLLOWING

The "`if`, `else if`, `else`" chain

One of the powers of statements and expressions is the ability to use them in more complex ways. In addition to expressions, scopes can contain other statements. For example, an `else` scope can contain an `if` statement. Connecting statements and expressions in different ways allow us to make programs behave intelligently according to their purposes.

Example

The following is a bit of complex code written under the agreement that riding to a good coffee shop is preferred over walking to a bad one:

```
import std.stdio;

void main() {
    bool existsCoffee = false;
    bool existsBicycle = false;

    if (existsCoffee) {
        writeln("Drink coffee at home");

    } else {

      if (existsBicycle) {
          writeln("Ride to the good place");

      } else {
          writeln("Walk to the bad place");
      }
    }
}
```

The code above represents the phrase “if there is coffee, drink at home; else if there is a bicycle, ride to the good place; otherwise, walk to the bad place.” Let’s complicate this decision process further: instead of having to walk to the bad place, let’s first try the neighbor:

```
import std.stdio;

void main() {
    bool existsCoffee = false;
    bool existsBicycle = false;
    bool neighborIsHome = true;

    if (existsCoffee) {
    writeln("Drink coffee at home");

    } else {
        if (existsBicycle) {
            writeln("Ride to the good place");

        } else {
            if (neighborIsHome) {
                writeln("Have coffee at neighbor's");

            } else {
                writeln("Walk to the bad place");
            }
        }
    }
}
```

How to chain statements in a compact way?

Such decisions like “if this case, else if that other case, else if that even other case, etc.” are common in programs. Unfortunately, when the guideline of always using curly brackets is followed obstinately, the code ends up having too much horizontal and vertical space. Ignoring the empty lines, the 3 `if` statements and the 4 `writeln` expressions above occupy a total of 13 lines.

In order to write such constructs in a more compact way, when an `else` scope contains only one `if` statement, then the curly brackets of that `else` scope are omitted as an exception of this guideline.

Let us leave the following code untidy as an intermediate step before showing the better form of it. No code should be written in such an untidy way.

The following is what the code looks like after removing the curly brackets of the two `else` scopes that contain just a single `if` statement:

```
import std.stdio;

void main() {
    bool existsCoffee = false;
    bool existsBicycle = false;
    bool neighborIsHome = true;

    if (existsCoffee) {
    writeln("Drink coffee at home");

    } else

    if (existsBicycle) {
        writeln("Ride to the good place");

    } else

       if (neighborIsHome) {
           writeln("Have coffee at neighbor's");

       } else {
           writeln("Walk to the bad place");
       }
}
```

If we now move those `if` statements up to the same lines as their enclosing `else` clauses and tidy up the code, we end up with the following more readable format:

```
import std.stdio;

void main() {
    bool existsCoffee = false;
    bool existsBicycle = false;
    bool neighborIsHome = true;

   if (existsCoffee) {
   writeln("Drink coffee at home");

    } else if (existsBicycle) {
    writeln("Ride to the good place");
    } else if (neighborIsHome) {
    writeln("Have coffee at neighbor's");


    } else {
    writeln("Walk to the bad place");
    }
}
```

Removing the curly brackets allows the code to be more compact and lines up all of the expressions for easier readability. The logical expressions, the order that they are evaluated and the operations that are executed, when they are true, are now easier to see at a glance. This common programming construct is called the **"if, else if, else"** chain.

In the next lesson, you will find a coding challenge.