A Simple Interactive Program

This lesson will discuss using the command line to pass input to an application in Dart.

We'll cover the following

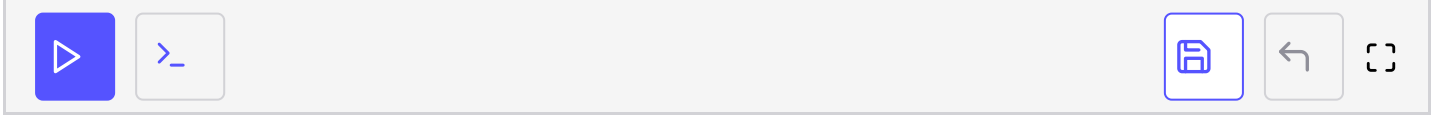
The *Hello World* program we wrote in a previous lesson was simply displaying some output. But in many cases, we come upon scenarios where we need to take input from the user and print that input onto the console.

Let's write a program for a personalized greeting application. The program will ask the user for their name and then display a personalized greeting for that user.

> The coding program in this lesson requires input from the user in order for it to execute successfully.
>
> Before you press **RUN**, you must select the **>_STDIN** button located next to the **RUN** button which will provide an input field where you can type your input.
>
> After you have typed the input, press **RUN** to execute the program.

```
import 'dart:io';

main() {
    print("Hello " + stdin.readLineSync());
}
```



Understanding the Code

The code above might look a bit intimidating. But don't worry, let's go over it one

line at a time.

Libraries

On **line 1** of the code above, we are importing the `dart:io` library.

In computer programming, a library is a collection of similar code that you can reference in your own code. When you **import** a library, you can access all the code in that library.

Every library has a particular purpose. The `dart:io` library provides I/O (Input/Output) support for non-web applications. We imported `dart:io` because the program we wrote requires input from the user.

Taking Input in Dart

Dart provides multiple methods that can be used to take external input from a user depending on the type of input required by the program. For this course, we will be using the `readLineSync()` method (function) which reads a line from the input.

The complete expression we will use is `stdin.readLineSync()`. **stdin** stands for standard input, letting the compiler know that the program specifically requires some sort of input from the user. `readLineSync()` is a built-in method which we have already discussed above.

Line 4 is displaying *Hello* and then concatenating (joining) it with the input provided by the user using the addition (`+`) operator (a topic we will discuss later in the course).

That's pretty much all there is to the code written above.

Try using the print statement yourself in a challenge in the next lesson.