## **Strings**

After the character type, we'll study its extension called string.

#### WE'LL COVER THE FOLLOWING

- Definition
- Concatenation
- String Length
- Accessing Elements
- The Existence of a Character

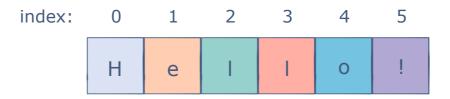
## Definition #

A string is a collection of characters joined together.

Double quotes are used to enclose the value of a string, e.g., "Hello".

Since a string is a collection, we can access each individual character in the string as well. The characters are indexed from 0 to n-1 where n is the length of the string.

Unlike characters, strings can be of any length, including 1.



Length of this string is 6

"Hello World".

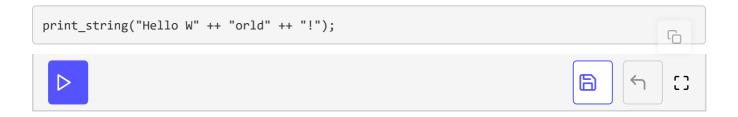
We can print strings using <code>Js.log()</code> (prints in a new line each time) or <code>print\_string()</code> (prints in the same line):

```
Js.log("Hello World");

/* Alternate method */
print_string("Hello World");
```

### Concatenation #

Reason allows us to append strings together using the ++ operator.



# String Length #

The length of a string can be obtained using the <a href="String.length">String.length</a>() method. Keep in mind that an empty space in a string also counts as an element.

```
Js.log(String.length("Hello World")); /* 11 */
Js.log(String.length("a")); /* 1 */
```

# **Accessing Elements**

A character in a string can be accessed using its index. The simplest approach is to use the .[] notation:

```
("Hello World").[4]
```

The line above will return o as it is the fifth character in the string. The index is enclosed inside the square brackets. This process is known as **indexing**.

Here it is in action:



An alternative approach is to use the <a href="String.get()">String.get()</a> method, but we'll leave that as a self-exercise.

### The Existence of a Character #

We can check if a particular character exists in a given string using the String.contains() method. It requires the following template:

```
String.contains(string, char)
```

Here's the method in action:



This is helpful when we need to do a character search in a large piece of string.

The last data type we need to discuss is the **unit**, but we'll leave that for a later section in order to give it context. For now, let's move on to the concept of polymorphic operators.