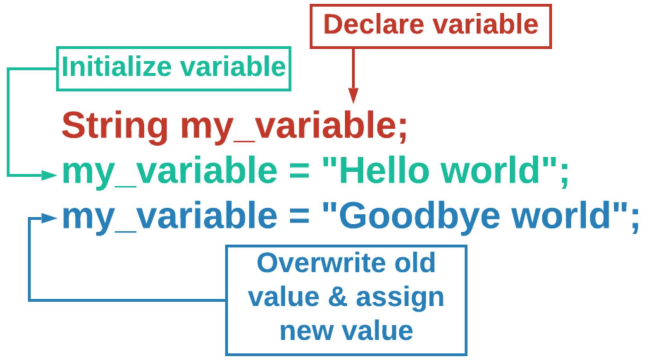
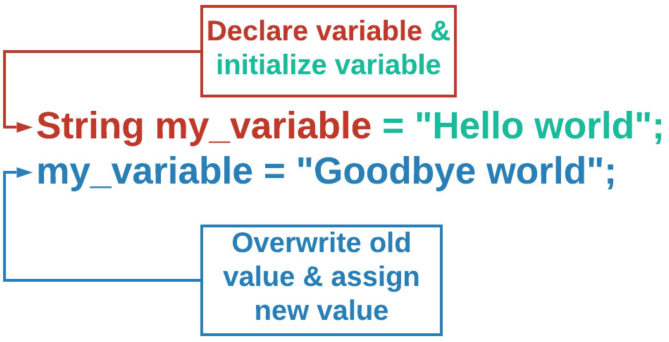
Initializing, Assigning, and Accessing

Initializing Assigning Values

We call the process of setting the **initial** value of a variable **initialization**. You can do this separately after the declaration or combine it into the same statement as the declaration.





Since the value stored in a variable can change, we call changing the value **assigning** or **re-assigning**. Use the assignment operator to give a variable a new value.

Accessing Variables

Enter the code below and see the results of the `println` commands. Use the code visualizer to see how the value of `my_variable` changes.

```
String my_variable = "Hello world";
System.out.println(my_variable);
my_variable = "Goodbye world";
System.out.println(my_variable);
```

When we use a variable's name to get the value like in the `println` statements above, we say we are **accessing** the variable.

Code Visualizer

TRY IT

```
Hello world
Goodbye world
```

Declaring, Initializing, and Assigning Variab…

Construct a program that initializes variable `my_variable` to 5 and prints it out.

Then, re-assign `my_variable` to 10 and print it out.

The output of the code you are constructing looks like:

```
5
10
```

You will not need to use all of the blocks.

Drag from here

```
my_variable = 5;
```

```
double my_variable = 5;
```

```
my_variable = "10";
```

```
int my_variable = 10;
```

Construct your solution here

```
int my_variable = 5;
```

```
System.out.println(my_variable);
```

```
my_variable = 10;
```

```
System.out.println(my_variable);
```

You will use an `int` instead of a double since 5 and 10 are whole numbers. Additionally, if you use `double` it would print out `5.0` and `10.0` by default.

You cannot assign a String (`"10"`) to an `int`.

You do not need to re-declare when you reassign (`int my_variable = 10`).

```
int my_variable = 5;
System.out.println(my_variable);
my_variable = 10;
System.out.println(my_variable);
```

Check It!