

4 Function

A function is simply a bunch of code bundled in a section. This bunch of code ONLY runs when the function is called. Functions allow for organizing code into sections and code reusability.

Using a function has ONLY two parts. (1) Declaring/defining a function, and (2) using/running a function.

Name of function

That's it, it's just a name you give to your function. Tip: Make your function names descriptive to what the function does.

Return (optional)

A function can optionally spit-out or "return" a value once it's invoked. Once a function returns, no further lines of code within the function run.

Invoke a function

Invoking, calling or running a function all mean the same thing. When we write the function name, in this case `someName`, followed by the brackets symbol `()` like this `someName()`, the code inside the function gets executed.

```
// Function declaration / Function statement
```

```
function someName(param1, param2) {
```

```
// bunch of code as needed...
```

```
var a = param1 + "love" + param2;
```

```
return a;
```

```
}
```

```
// Invoke (run / call) a function
```

```
someName("Me", "You")
```

Parameters / Arguments (optional)

A function can optionally take parameters (a.k.a arguments). The function can then use this information within the code it has.

Code block

Any code within the curly braces `{ ... }` is called a "block of code", "code block" or simply "block". This concept is not just limited to functions. "if statements", "for loops" and other statements use code blocks as well.

Passing parameter(s) to a function (optional)

At the time of invoking a function, parameter(s) may be passed to the function code.