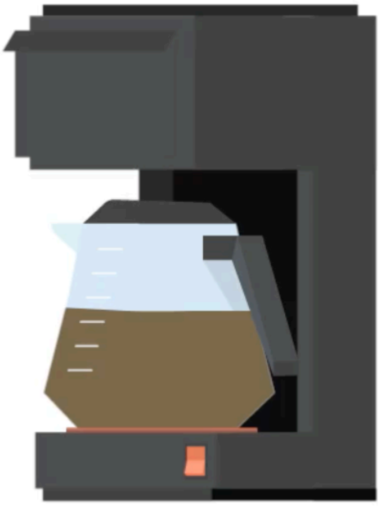
team treehouse : swift functions and optionals

- functions
  - * self-contained chunks of code that perform a specific task



- for example coffee maker is a machine that does a certain function
  - * and has two inputs
    - ** water
    - ** coffee beans
  - * and produces an output
    - ** coffee
  - * and inputs of the coffee maker need to be of specific type
    - ** i.e. ground coffee

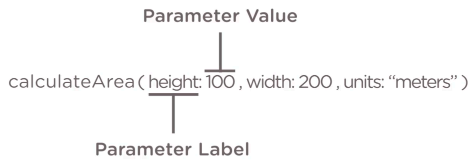
```
print("swift functions")
```

- *swift functions* is a parameter that is passed into our *print function*

naming functions

- must *start* with a *letter* and not a number or special character
- preferable *use camel casing* i.e. calculateArea
    * where *subsequent words* are *capitalised*
- the name must describe its *purpose*
    * don't name a function *a* or *calc*
        ** it does not explain the purpose of the function
    * give it a **meaningful name**

functions



```
calculateArea( height: 100 , width: 200 , units: "meters" )
```

- the parameters *height* and *width* are local parameters
    * can't be used outside of that function

tuples

- where does the name come from?
    * we have words like
        ** single
        ** double
        ** triple
        ** quadruple
        ** quintuple
        ** N-tuple
    * where the first word is the latin number suffixed with *uple*
    * since our list is not restricted to any number of values

        - we can call it a **tuple**

optionals

- when writing software, you are always manipulating data
    - data can be stored into a variable
    - or passed into a function
- then you have to perform a set of tasks
    - to change it around
    - or return it as a result
- our searchName function searches for a given string within an array
- but what happens when we don’t pass it anything
- it is *expecting data* but did not receive anything
    - that happens very often when writing software
- the program doesn’t know what to do in this circumstance
    - and the only choice it has is to *shut down* the program
        - this *shut down* is called a *crush*
- swift has *safeguards*
    - one of these *safeguards* is called an *optional*
- optionals can **either** have a **value** or a **nil**
    - adding a **?** to a type makes it an optional
        - String?
        - Int?
        - Double?
        - Bool?
        - Array?
- **bang !** operator **unwraps** an optional value