Hashes

Link to Ed Lesson

A link to the lesson can be found here

What is a hash?

A hash is a **ruby data type** used to define a collection of *key value pairs*.

Syntax for Hash data type

The below code is how ruby interprets hashes.

```
    hash = {:key => "value"}
```

- **Keys** can be a *symbol* or a *string* but it is **highly recommended** to use a symbol as the key in a hash and NOT a string as the key.
- This is because **strings require new memory locations** each time they are created while symbols do not.
- **Values** can be any data type (string, integer, symbol, another array or another hash).
- Example 1 below has a string as the value.
- Example 2 has an integer as a value.
- Example 3 has multiple data types as values (including an array and hash as values within the hash).

1.
   ```
       hash = {:key => "value"}
   ```

2.
   ```
       hash = {:key => 20}
   ```

3.
   ```
       hash = {:name => "bob", :age => 10, :title => :painter, :hobbies => ["painting", "music"], :schedule => {:monday => "painting class", :tuesday => "music class" }}
   ```

- Multiple **key value pairs** are seperated by a comma. For example:

  -
    ```
        hash = {:key_one => "value_one", :key_two => "value_two",
    :key_three => "value_three" }
    ```

Shorthand/alternate syntax

- Alternate syntax replaces the => (fat arrow) with a : (colon) at the front of the key.

1.
   ```
       #with fat arrow
       hash = {:key => "value"}

       #replacing fat arrow
       hash ={key: "value}
   ```

- When using this shorthand, ruby STILL interprets the code with the fat arrow. In other words, there is **no difference** between the shorthand hash or the fat arrow hash.

- The shorthand just makes it **simpler for developers to code and read.**

- The shorthand also reduces the number of characters used.

Accessing Hash elements

- To access an element in a hash, you need to pass the key between square brackets. Then, the keys **relative value** will be displayed (unlike arrays where you access elements with square brackes by using an **index** number). For example, if the hash was:

  ```
  phone_numbers = {:bob => "0414141414", :peter => "0415151515",:charlie
  => "0416161616"}
  ```

  - To access the **value** for the first key value pair (bob in the above example) you would need to pass the key in square brackets after calling the variable name (as below).

    ```
    puts phone_numbers[:bob]
    ```

  - This will return the value of the key symbol named bob (below).

    ```
      => 0414141414
    ```

  - The same goes if we puts the key symbol for peter (as below).

    ```
    puts phone_numbers[:peter]
    ```

  - The value of the key symbol named peter (below).

    ```
    0415151515
    ```

- If a key is defined as a symbol, it needs to be accessed as a symbol and therefore CANNOT be accessed as a string. For example, if the hash was:

  ```
  profile = {:name => "isaac", "age" => 3}
  ```

  - Accessing the values will be **different** for **key one** and **key two** as key one one is a **symbol** and key two is **string**.

    ```
    puts profile[:name]

    puts profile["age"]
    ```