Symbols

Link to Ed Lesson

A link to the lesson can be found here.

What is a symbol?

- In ruby, a symbol is a data type (just like strings, integers, arrays etc)
- Symbols look similar to strings but are represented with a : (colon) at the start and WITHOUT quotation marks. For example:

  ```
  "hello" #this is a string.

  :hello #this is a symbol.
  ```

Symbols don't take integers

- You CAN NOT use an **integer** in symbols. For example:
  - :1 will NOT work as a symbol.
  - :one WILL work as a symbol.

Symbols vs strings

- Symbols point to the SAME MEMORY location.
  - This means if you create a symbol called :hello it will have a **dedicated memory location** for all symbols that are declared :hello .
  - Therefore, **repeated symbols** with the same name will use **less memory** than repeated strings with the same name/ set of characters.
- Strings point to DIFFERENT MEMORY locations each time a string is created.
  - This means if you create a string "hello" it will have one memory location and if you create a second string also called "hello" it will have a different memory location than the first string (even though it has the same name/set of characters).
- Symbols will NOT allow **string operations** to be performed. For example:
  - String **concatination** will NOT work with symbols.
  - Therefore, assigning a symbol to a **variable** does NOT make much sense.

Object ID

- In ruby every data type is an object and each of them have a **unique memory location** that it points to.
- To find the memory location, use the method .object_id
  For example:
  - "hello".object_id will point to a memory location

Why use symbols?

- Since symbols point to the same memory location, they are a lot **faster to access than strings**.
- Most commonly, a symbol is used like **label** which makes them particuarly useful when used in **conjuction** with the **hash** data type.