Hashes

In this video we're going to take a look at **hashes** in IRB.

Hashes are a lists of **key** and **value** pairs. We can create a hash literal by placing inside curly braces:

```
marge = {:name => 'Marge Simpson'}
```

`:name` is a symbol because it starts with the colon character. Then we use the => character which is known as a **hash rocket**. This is used to assign the value to the key `name`.

It's a common practice to use symbols for the keys like we've done here, because they use memory more efficiently than strings in Ruby.

Creating Hashes Using a Shorthand

There's also a shorthand notation that can be used in Ruby to create hashes.

```
homer = {name: 'Homer Simpson', job: 'Nuclear Safety Inspector', children: ['Bart', 'Lisa', 'Magg
```

Hash Rockets

Notice in the hash that's created that all of the keys are symbols, even though they weren't created as symbols I put the colon on afterwards and the hash rockets have been entered automatically.W e can also use different objects for the values as well, we don't just have to use strings.

If we want to check that key exists in a hash, we can use the `include?` method:

```
homer.include?(:job)
```

Length method

If we want to know how many values there are in a hash, we can use the `length` method:

```
homer.length
```

We get the answer of 3, because we entered three key-value pairs.

You can add new values to a hash by writing the key that you want to add, and placing it equal to its value:

```
homer[:wife] = marge
```

Nested Hash

If we want to access the values of a nested hash, we do this by referencing each key in order:

```
homer[:wife][:name]
```