super

Ruby provides us with a built-in function called super that allows us to call methods up the inheritance hierarchy. When you call super from within a method, it will search the inheritance hierarchy for a method by the same name and then invoke it.

```ruby
class Animal
  def speak
    "Hi I am an Animal"
  end
end

class Dog < Animal
  def speak
   super +  " and a Dog"
  end
end

dog = Dog.new
puts dog.speak
```

In the subclass' speak method we use super to invoke the speak method from the superclass, Animal, and then we extend the functionality by appending some text to the result.

Another more common way of using super is with initialize.

```
class Animal
  attr_accessor :name

  def initialize(name)
    @name = name
  end

  def name
    @name
  end
end

class Dog < Animal
  def initialize(color)
    super
    @color = color
  end

  def color
    @color
  end

  def name
    "Yes..... " + @name
  end
end

dog = Dog.new("brown")
=> #<Dog:0x0000000268f180 @name="brown", @color="brown">

irb(main):040:0> dog.name
=> "brown"
irb(main):041:0> dog.color
=> "brown"
```

In this example, we're using super without arguments. However, the initialize method, where super is being used, takes an argument and adds a new twist to how super is invoked. Here, in addition to the default behavior, super automatically forwards the arguments that were passed to the method from which super is called. At this point, super will pass the color argument in the initialize defined in the subclass to that of the Animal superclass and invoke it. That explains the presence of @name="brown" when the bruno instance is created. Finally, the subclass' initialize continues to set the @color instance variable.

When called with specific arguments, eg. super(a, b), the specified arguments will be sent up the method lookup chain:

```ruby
class Animal
  attr_accessor :name

  def initialize(name)
    @name = name
  end

  def name
    @name
  end
end

class Dog < Animal
  def initialize(name, color)
    super(name)
    @color = color
  end
```

```ruby
  def color
    @color
  end

  def name
    "Yes..... " + @name
  end
end



dog = Dog.new("labdradog","brown")

 => #<Dog:0x0000000090d260 @name="labdradog", @color="brown">

dog.name

 => "Yes..... labdradog"
```