Instance Methods vs Class Methods

Now that we have the basics of classes down, let's explore two different methods we may build into class: **instance methods** and **class methods**.

Instance Methods

So far we've been only dealing with instance methods with our classes. Like it's name suggests, an instance method is one that is called on an *instance* of a class. Let's check out an instance method:

```
class Dog
  def initialize(name, bark)
    @name = name
```

```
  end

  def speak
    @name + " says " + @bark
  end
end

my_dog = Dog.new("Fido", "woof")
my_dog.speak          # "Fido says woof"

other_dog = Dog.new("Doge", "much bork")
other_dog.speak       # "Doge says much bork"
```

`speak` is an instance method because we can only call it on a `Dog` instance we initialized using `Dog.new`. Remember that if something is an instance of `Dog`, it is an object with a `@name` and `@bark`. Since `my_dog` and `other_dog` are instances, when we call `speak` on them respectively, we can get different behavior because they can have different `@name` and `@bark` values. An instance method depends on the *attributes* or *instance variables* of an instance.

For notation, we'll use **Dog#speak** to denote that `speak` is an **instance**

Class Methods

A class method is a method that is called directly on the class. Let's see how to define a class method:

```
class Dog
  def initialize(name, bark)
    @name = name
    @bark = bark
  end

  def self.growl
    "Grrrrr"
  end
end

Dog.growl   # Grrrrr
```

Notice that we define class method by adding `self.` to the front of a method name. In this context. `self` refers to the `Dog` class itself. Since `growl` is a

class directly . A class method cannot refer to any instance attributes like `@name` and `@bark`! As programmers, we'll choose to build class methods for added utility.

For notation we'll use **Dog::growl** to denote that `growl` is an **class method** of `Dog`.

For example, here is a class method that is a bit more practical, `Dog::whos_louder` :

```
class Dog
  def initialize(name, bark)
    @name = name
    @bark = bark
  end

  def self.whos_louder(dog_1, dog_2)
    if dog_1.bark.length > dog_2.bark.length
      return dog_1.name
    elsif dog_1.bark.length < dog_2.bark.length
      return dog_2.name
    else
```

```
  end

  def name
    @name
  end

  def bark
    @bark
  end
end

d1 = Dog.new("Fido", "woof")
d2 = Dog.new("Doge", "much bork")
p Dog.whos_louder(d1, d2) # "Doge"
```

You may be wondering why we prefer to make `Dog::whos_louder` a class method. We make this choice because the code inside of the method does not

attributes of `@name`, `@bark`.

Wrapping Up

- `Class#method_name` means `method_name` is an instance method
- `Class::method_name` means `method_name` is a class method

Did you find this lesson helpful?

No Yes

✓ Mark As Complete

Finished with this task? Click **Mark as Complete** to continue to the next page!

Return to this page anytime through the **Course Outline**