







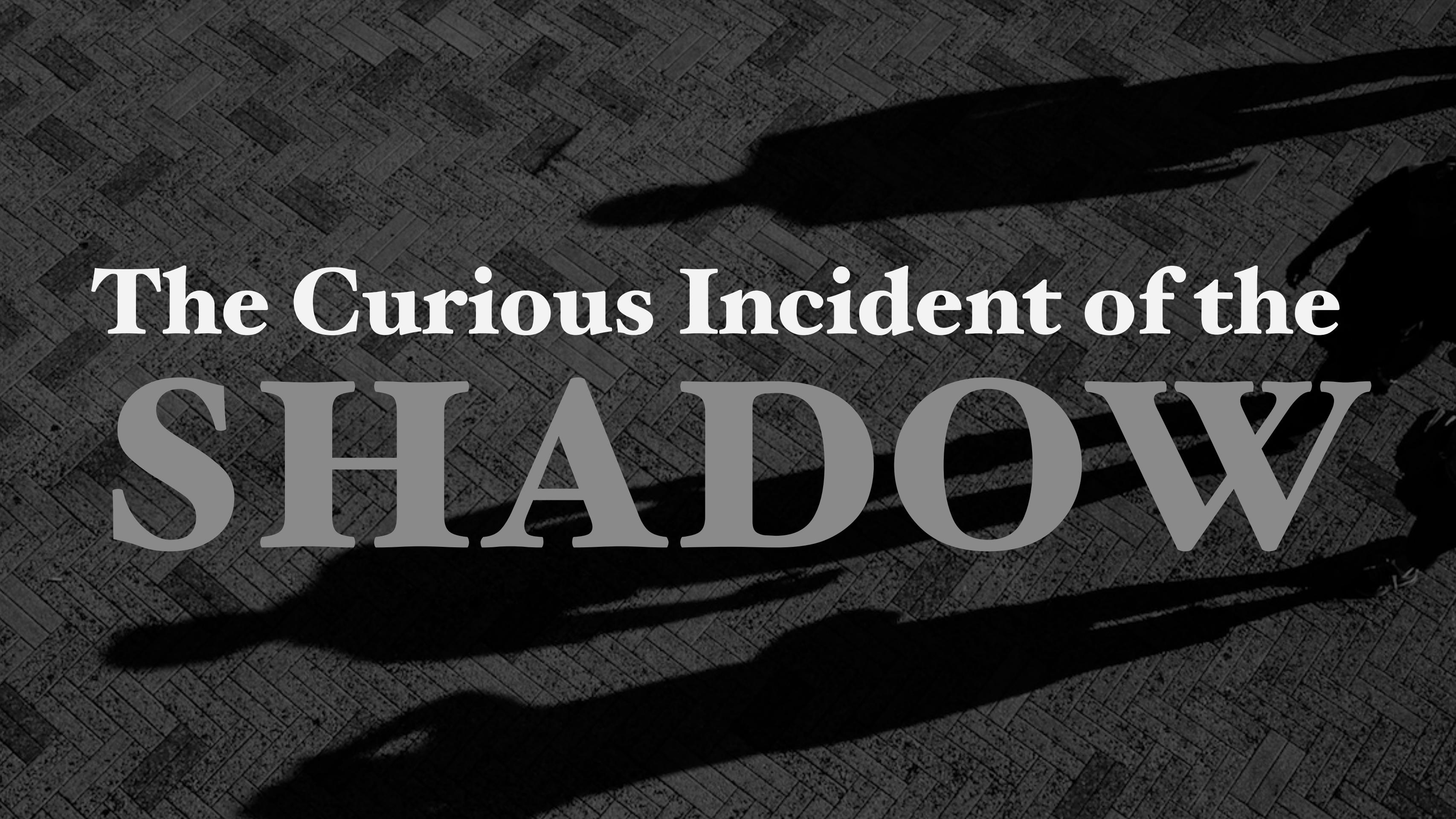
The

# The Curious



# The Curious Incident

# The Curious Incident of the

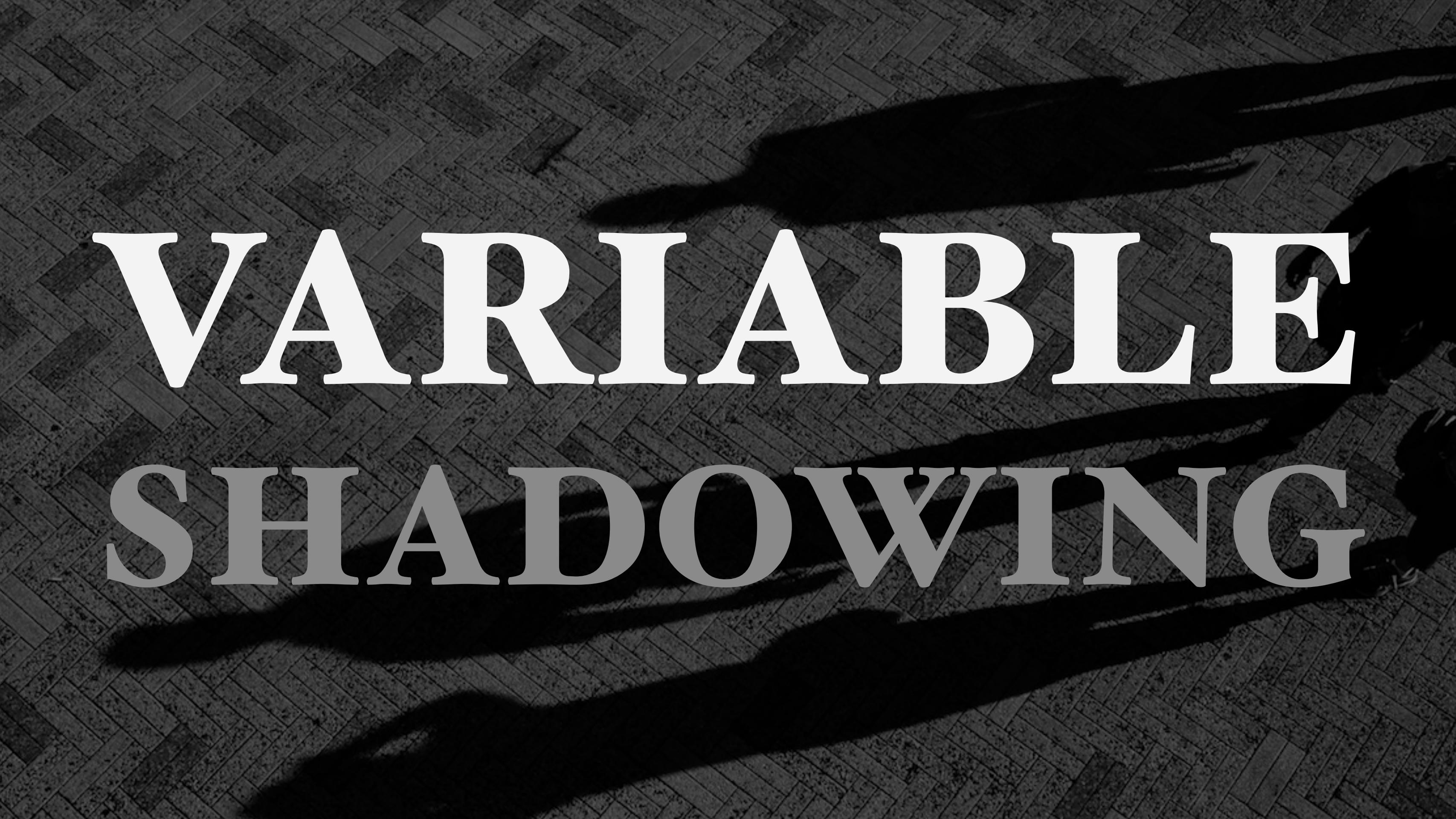


# The Curious Incident of the STeADoW

The Curious Incident of the  
**SHADOW**  
in the Run-Time



SHADOWY



**VARIABLE  
SHADOWING**

**shadow.rb**

**shadow.rb**

**x = 42**

# shadow.rb

```
x = 42
```

```
3.times { |x| puts "x is #{x}" }
```

# **shadow.rb**

```
x = 42
```

```
3.times { |x| puts "x is #{x}" }
```

```
$ ruby shadow.rb
```

```
x is 0
```

```
x is 1
```

```
x is 2
```

# shadow.rb

```
x = 42  
3.times { |x| puts "x is #{x}" }
```

```
$ ruby shadow.rb
```

```
x is 42  
x is 42  
x is 42
```

# **shadow.rb**

```
x = 42
```

```
3.times { |x| puts "x is #{x}" }
```

```
$ ruby shadow.rb
```

```
x is 0
```

```
x is 1
```

```
x is 2
```

DOES RUBY  
CARE ABOUT  
SHADOWING?

# shadow.rb

```
x = 42
3.times { |x| puts "x is #{x}" }
```

# **shadow.rb**

```
x = 42
3.times { |x| puts "x is #{x}" }
```

```
$ ruby -w shadow.rb
```

# shadow.rb

```
x = 42
3.times { |x| puts "x is #{x}" }
```

```
$ ruby -w shadow.rb
shadow.rb:2: warning: shadowing outer local variable - x
shadow.rb:1: warning: assigned but unused variable - x
x is 0
x is 1
x is 2
```

BUT WHY  
THOUGH?

# shadow.rb

```
x = 42
```

```
3.times { |x| puts "x is #{x}" }
```

# shadow.rb

```
x = 42
3.times do |x|
  puts(
    "Local x is #{x} and \"\
     'outer' x is #{'What goes here??'}"
  )
end
```

# shadow.rb

```
y = 42
3.times { |x| puts "x is #{x} and y is #{y}" }
```

# shadow.rb

```
y = 42
3.times { |x| puts "x is #{x} and y is #{y}" }
```

```
$ ruby -w shadow.rb
x is 0 and y is 42
x is 1 and y is 42
x is 2 and y is 42
```

"[Matz] thinks that shadowing by itself is a bad habit and should be discouraged."

– *Shouhei Urabe, quoting Matz*



# RuboCop

## ShadowingOuterLocalVariable



# INSTANCE METHOD SHADOWING

"In Ruby, *local variable names* and  
*method names* are nearly identical."

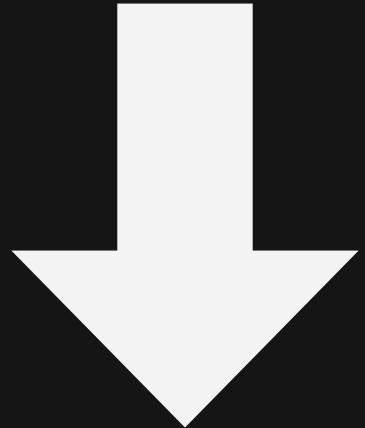
"If you have not assigned to one of  
these ambiguous **names**, Ruby will  
*assume you wish to call a method.*"

"Once you have assigned to the **name**, Ruby will *assume* you wish to reference a *local variable*."

"The local variable is created when  
the parser encounters the assignment,  
not when the assignment occurs."

– *Local Variables and Methods Assignment section of Ruby's syntax documentation*

TOP



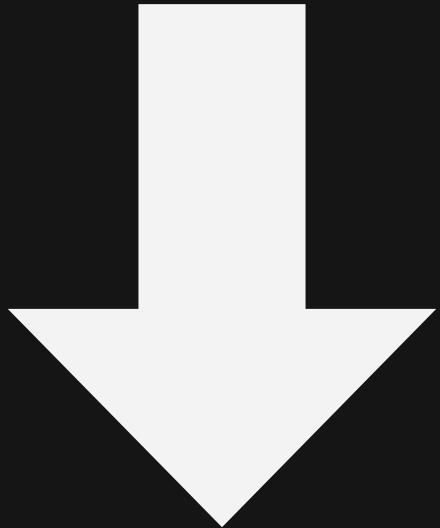
BOTTOM

NAME S

CAN

CHANG E

# METHOD



# VARIABLE



person.rb

**class Person**

**end**

```
class Person
```

```
  def initialize(name = nil)
    @name = name
  end
```

```
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

end
```

```
class Person  
  def name  
    @name  
  end
```

```
  def name=(new_name)  
    @name = new_name  
  end
```

```
  def initialize(name = nil)  
    @name = name  
  end
```

```
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
$ irb
irb(main):001:0> require './person.rb'
true
irb(main):002:0>
```

```
$ irb
irb(main):001:0> require "./person.rb"
true
irb(main):002:0> Person.new("Paul").say_name
```

```
$ irb
irb(main):001:0> require "./person.rb"
true
irb(main):002:0> Person.new("Paul").say_name
My name is nil
nil
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
$ irb
irb(main):001:0> require "./person.rb"
true
irb(main):002:0> Person.new("Paul").say_name
My name is nil
nil
irb(main):003:0>
```

```
$ irb
irb(main):001:0> require "./person.rb"
true
irb(main):002:0> Person.new("Paul").say_name
My name is nil
nil
irb(main):003:0> Person.new.say_name
```

```
$ irb
irb(main):001:0> require "./person.rb"
true
irb(main):002:0> Person.new("Paul").say_name
My name is nil
nil
irb(main):003:0> Person.new.say_name
My name is "Unknown"
nil
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```

```
class Person
  attr_accessor :name

  def initialize(name = nil)
    @name = name
  end

  def say_name
    if name.nil?
      name = "Unknown"
    end
    puts "My name is #{name.inspect}"
  end
end
```





GET TO THE CODE

```
irb(main):002:0> Person.new("Paul").say_name
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   if name.nil?
14:     name = "Unknown"
15:   end
16:
17:   puts "My name is #{name.inspect}"
18: end
```

```
[1] pry(#<Person>)>
```

```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   if name.nil?
14:     name = "Unknown"
15:   end
16:
17:   puts "My name is #{name.inspect}"
18: end
```

```
[1] pry(#<Person>)> name
nil
[2] pry(#<Person>)>
```

ERR...WHAT?

```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   if name.nil?
14:     name = "Unknown"
15:   end
16:
17:   puts "My name is #{name.inspect}"
18: end
```

```
[1] pry(#<Person>)> name
nil
[2] pry(#<Person>)>
```

```
[1] pry(#<Person>)> name  
nil  
[2] pry(#<Person>)> next
```

```
[1] pry(#<Person>) > name  
nil  
[2] pry(#<Person>) > next
```

From: /person.rb @ line 17 Person#say\_name:

```
10: def say_name  
11:   binding.pry  
12:  
13:   if name.nil?  
14:     name = "Unknown"  
15:   end  
16:  
=> 17:   puts "My name is #{name.inspect}"  
18: end
```

```
[2] pry(#<Person>) >
```

```
[2] pry(#<Person>) > next
```

From: /person.rb @ line 17 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
13:   if name.nil?
14:     name = "Unknown"
15:   end
16:
=> 17:   puts "My name is #{name.inspect}"
18: end
```

```
[2] pry(#<Person>) > name
```

nil

```
[3] pry(#<Person>) > exit
```

My name is nil

**SPOOKY**

SPOOKY

QUANTUM

SPOOKY

QUANTUM

RUBY

**SPOOKY  
QUANTUM  
RUBY?**



```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts name.inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
[1] pry(#<Person>)>
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts name.inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
[1] pry(#<Person>)> name
nil
```

```
[1] pry(#<Person>) > name  
nil  
[2] pry(#<Person>) > next  
"Paul"
```

From: /person.rb @ line 14 Person#say\_name:

```
10: def say_name  
11:   binding.pry  
12:  
13:   puts name.inspect  
=> 14:   if name.nil?  
15:     name = "Unknown"  
16:   end  
17:  
18:   puts "My name is #{name.inspect}"  
19: end
```



defined?

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
[1] pry(#<Person>)>
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
[1] pry(#<Person>)> next
"method"
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
[1] pry(#<Person>) > defined?(name)
"local-variable"
```



```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
irb(main):002:0> Person.new("Paul").say_name
```

```
From: /person.rb @ line 13 Person#say_name:
```

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```

```
irb(main):002:0> Person.new("Paul").say_name
```

From: /person.rb @ line 13 Person#say\_name:

```
10: def say_name
11:   binding.pry
12:
=> 13:   puts defined?(name).inspect
14:   if name.nil?
15:     name = "Unknown"
16:   end
17:
18:   puts "My name is #{name.inspect}"
19: end
```



**DO NOT**

DOON'T

SHADOW

```
def say_name
  if name.nil?
    self.name = "Unknown"
  end
  puts "My name is #{name.inspect}"
end
```

```
def say_name
  name = self.name || "Unknown"
  puts "My name is #{name.inspect}"
end
```

```
def say_name
  name = self.name || "Unknown"
  puts "My name is #{name.inspect}"
end
```

```
def say_name
  name = name() || "Unknown"
  puts "My name is #{name.inspect}"
end
```



**THANKS!**  
**@paulfioravanti**

