Class UnboundMethod < Object

Ruby supports two forms of objectified methods. Class Method is used to represent methods that are associated with a particular object: these method objects are bound to that object. Bound method objects for an object can be created using Object#method.

Ruby also supports unbound methods, which are method objects that are not associated with a particular object. These can be created either by calling unbind on a bound method object or by calling Module#instance_method.

Unbound methods can be called only after they are bound to an object. That object must be a *kind_of?* the method's original class.

```
class Square
  def area
    @side * @side
  end
  def initialize(side)
    @side = side
  end
end

area_unbound = Square.instance_method(:area)

s = Square.new(12)
area = area_unbound.bind(s)
area.call   # =>   144
```

Unbound methods are a reference to the method at the time it was objectified: subsequent changes to the underlying class will not affect the unbound method.

```
class Test
  def test
    :original
  end
end
um = Test.instance_method(:test)
class Test
  def test
    :modified
  end
end
t = Test.new
t.test            # =>   :modified
um.bind(t).call   # =>   :original
```

Instance methods

arity

umeth.arity → *fixnum*

See Method#arity on page 591.

bind

umeth.bind(*obj*) → *method*

Bind *umeth* to *obj*. If Klass was the class from which *umeth* was originally obtained, obj.kind_of?(Klass) must be true.

```
class A
  def test
    puts "In test, class = #{self.class}"
  end
end
class B < A
end
class C < B
end
um = B.instance_method(:test)
bm = um.bind(C.new)
bm.call
bm = um.bind(B.new)
bm.call
bm = um.bind(A.new)
bm.call
```

produces:

```
In test, class = C
In test, class = B
prog.rb:16:in `bind': bind argument must be an instance of B (TypeError)
from /tmp/prog.rb:16:in `<main>'
```

name

1.9

umeth.name → *string*

Returns the name of the method *umeth*.

```
um = String.instance_method(:upcase)
um.name   # =>   :upcase
```

owner

1.9

umeth.owner → *module*

Returns the class or module in which *umeth* is defined.

```
um = String.instance_method(:upcase)
um.owner   # =>   String
```

source_location

1.9

umeth.source_location → [*filename*, *lineno*] or nil

Returns the source filename and line number where *umeth* was defined or nil if self was not defined in Ruby source. See Method#source_location for an example.