## Integrated Support for HOM

*(Higher-Order Messaging) Feature #6594*

Implementing HOM should not require intermediate object, which creates unnecessary overhead.

<table>
<thead>
<tr>
<th>CURRENTLY</th>
<th>PROPOSAL</th>
</tr>
</thead>
</table>
| class HOM < BasicObject  
  def initialize(&function)  
    @function = function  
  end  
  def method_missing(op, *a, &b)  
    @function.call(op, *a, &b)  
  end  
 end | Integrate HOM into language as actual messaging mechanism.  
Use special syntax to define HOM methods. |
| class Example  
  def delegated_object  
    @delegated_object ||= Something.new  
    HOM.new do |op, *a, &b|  
      case op  
        when :method1, :method2  
          @delegated_object.public_send(op, *a, &b)  
      end  
    end  
  end | class Example  
  def delegated_object => op, *a, &b  
    @delegated_object ||= Something.new  
    case op  
      when :method1, :method2  
        @delegated_object.public_send(op, *a, &b)  
    end  
  end |

There are many more uses for HOM. See feature request for references.