Feature #4151 : Enumerable#categorize and Hash creation

```
# {a: 1, b: 2} => {a: do_sth(1), b: do_sth(2)}
h.each_with_object({}) { |(k,v),h| h[k] = do_sth(v) }
h.map_value { |v| do_sth(v) } # or #update_values, ...
```

Which one do you prefer ?

- We need some natural way to build a Hash
  - Primitives for simple cases : `map_value`, `map_key` (`rekey`)
  - and something more general like `categorize` or `associate`
- Examples

```
# Easy hash creation (not involving useless Array)
(3..4).associate { |i| i**i } # {3 => 27, 4 => 256}
# Counting: %w[G C C] => {"C": 2, "G": 1}
enum.categorize(:op => :+) { |e| [e, 1] }
# Grouping
[['a', 4], ['b', 7], ['a', 2]].categorize { |e| e }
# => {'a' => [4, 2], 'b' => [7]}
```

- I want to trigger the discussion so this can advance and get more feedback from core members