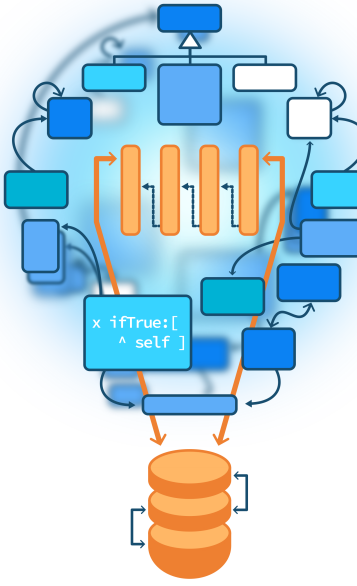
Shared variables

A Pharo code idiom

S.Ducasse, L. Fabresse, G. Polito, and P. Tesone



Goals

- Revisit *shared* variables (e.g., ClassVariables in Smalltalk jargon)
- Think about scope of sharing

Instance variables are local to one object

Nothing new!

- An instance variable value is **only accessible and local** to the object it belongs to
- If you modify an instance variable, you only modify that variable

Shared variables are shared by all the instances of a hierarchy

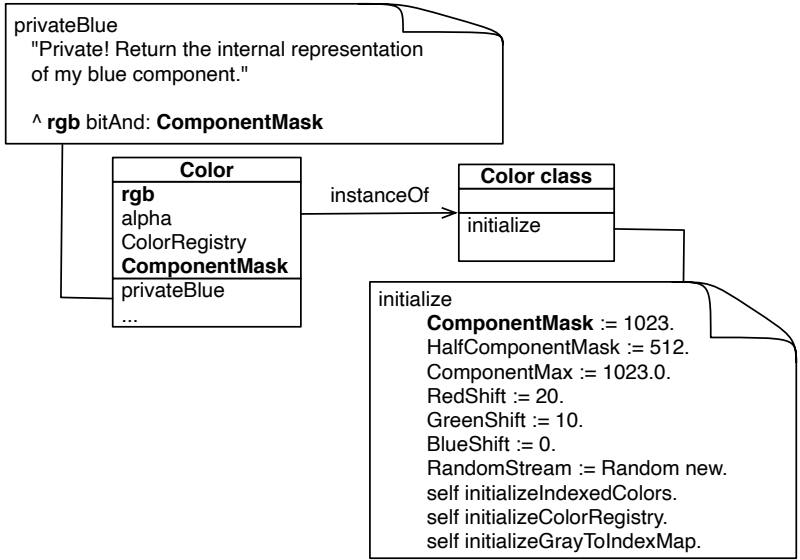
- **All** the instances of a class and its subclasses **share the SAME** shared variable
- If you modify a shared variable, it impacts all the instances
- A shared variable is usually initialized at class load time (class initialize method)
- Shared variables
  - called ClassVariables in Smalltalk
  - called SharedVariables in Pharo

Color example

All the instances of Color and its subclasses share ComponentMask

```
Object << #Color
  slots: { #rgb . #cachedDepth . #cachedBitPattern . #alpha };
  sharedVariables: { #RedShift . #CachedColormaps . #IndexedColors .
  #ComponentMax . #ComponentMask . #ColorRegistry . #GreenShift . #BlueShift };
  package: 'Colors'
```

Color's ComponentMask is a shared variable



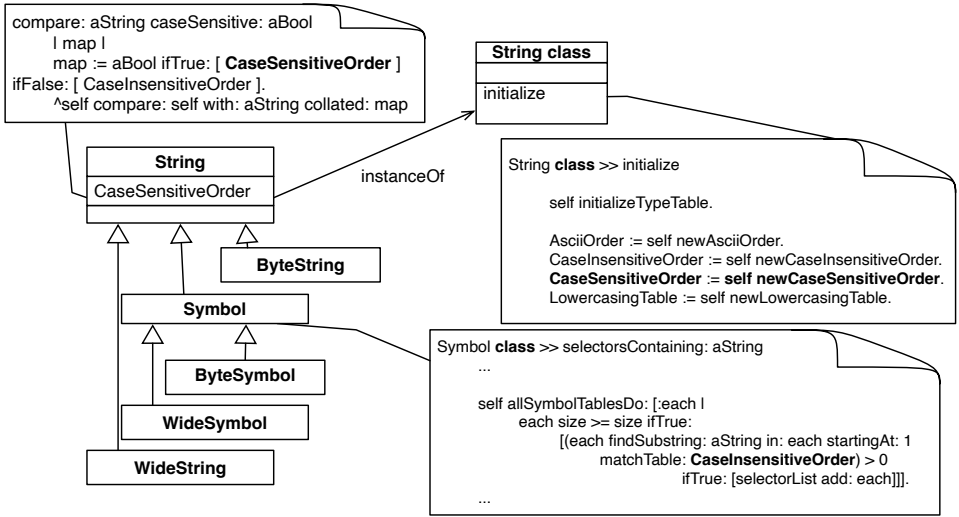
Shared variables: accessible from instance methods

```
Color >> setRed: r green: g blue: b
  "Initialize this color's r, g, and b components to the given
  values in the range [0.0..1.0].
  Encoded in a single variable as 3 integers in [0..1023]."

  rgb == nil ifFalse: [ self attemptToMutateError ].
  rgb := (((r * ComponentMax) rounded bitAnd: ComponentMask) bitShift: RedShift)
    + (((g * ComponentMax) rounded bitAnd: ComponentMask) bitShift: GreenShift)
    + ((b * ComponentMax) rounded bitAnd: ComponentMask).
  cachedDepth := nil.
  cachedBitPattern := nil
```

Shared variables: accessible from class methods

```
Color class >> initialize

  ComponentMask := 1023.
  HalfComponentMask := 512.  "used to round up in integer calculations"
  ComponentMax := 1023.0. "a Float used to normalize components"
  RedShift := 20.
  GreenShift := 10.
  BlueShift := 0.
  self initializeIndexedColors.
  self initializeColorRegistry.
  self initializeGrayToIndexMap.
```

Shared variable example: String



Shared Variables of String

```
ArrayedCollection << #String
  sharedVariables: { #CaseSensitiveOrder . #CSSeparators .
  #CSNonSeparators . #UppercasingTable .
  #CSLineEnders . #LowercasingTable . #CaseInsensitiveOrder .
  #TypeTable . #Tokenish . #AsciiOrder };
  package: 'Collections−Strings'
```

Shared variable CaseSensitiveOrder accessed in subclass method

```
ByteSymbol >> beginsWith: prefix
  "Answer whether the receiver begins with the given prefix string.
  The comparison is case−sensitive."

  prefix class isBytes ifFalse: [^super beginsWith: prefix].

  self size < prefix size ifTrue: [^ false].
  ^ (self findSubstring: prefix in: self startingAt: 1
      matchTable: CaseSensitiveOrder) = 1
```

Shared variable accessed in subclass class method

```
Symbol class >> selectorsContaining: aString
  "Answer a list of selectors that contain aString within them. Case-insensitive.
  Does return symbols that begin with a capital letter."

  ...

  self allSymbolTablesDo: [:each |
    each size >= size ifTrue:
      [(each findSubstring: aString in: each startingAt: 1
        matchTable: CaseInsensitiveOrder) > 0
          ifTrue: [selectorList add: each]]].
  ...
```

Implications

- There is a difference between Shared variables and instance variable of the metaclass
- There is a difference between:

```
Object << #BorderStyle
  sharedVariables: { #Default };
  package: 'Morphic−Core'
```

and

```
BorderStyle class
  slots: {#default};
  package: 'Morphic−Core'
```

Implications: One for all

```
Object << #BorderStyle
  sharedVariables: { #Default };
  package: 'Morphic−Core'
```

There is only one instance of BorderStyle for all the subclasses: SimpleBorderStyle BottomBorderStyle ComplexBorderStyle ...

Implications: One for each

```
BorderStyle class
  slots: {#default};
  package: 'Morphic−Core'
```

There is one instance for *EACH* of all the subclasses (potentially the same depending on the creation logic)

Conclusion

- Pay attention modifying shared variables potentially impacts many objects
- Can be used to support different sharing optimization (see other Lectures)

Produced as part of the course on http://www.fun-mooc.fr

Advanced Object-Oriented Design and Development with Pharo

A course by
S.Ducasse, L. Fabresse, G. Polito, and P. Tesone

Inria
LearningLab

2023