Metaprogramming with Macros

Eugene Burmako

École Polytechnique Fédérale de Lausanne
http://scalamacros.org/

10 September 2012

Macros

Macros

Macros realize the notion of textual abstraction.

Macros

Macros realize the notion of textual abstraction.

Textual abstraction:

- Recognize pieces of text that match a specification
- Replace them according to a procedure

Example

```
(let (x 42) (print x))
```

```
((lambda (x) (print x)) 42)
```

Example

```
(let (x 42) (print x))

(defmacro let args
  (cons
   (cons 'lambda
         (cons (list (caar args))
               (cdr args)))
   (cdar args)))

((lambda (x) (print x)) 42)
```

Why macros?

- Deeply embedded DSLs (database access, testing)
- Optimization (programmable inlining, fusion)
- Analysis (integrated proof-checker)
- Effects (effect containment and propagation)
- ...

Today's talk

Macrology is vast:

- Notation
- Variable capture
- Typechecking meta-programs
- Syntax extensibility
- ...

Today's talk

Macrology is vast:

- Notation
- Variable capture
- Typechecking meta-programs
- Syntax extensibility
- ...

Surveyed papers are versatile as well.

Today's talk

Going into all the details is a genuine pleasure

But instead let me tell you a story

Alexandre Dumas

Outline

The story of bindings

Anaphoric `if`

```
(aif (calculate)
  (print it)
  (error "does not compute"))
```

The aif macro

```
(aif (calculate)
  (print it)
  (error "does not compute"))

(defmacro aif args
```

```
(let (it (calculate))
  (if it
    (print it)
    (error "does not compute")))
```

Start with a template

```
(aif (calculate)
  (print it)
  (error "does not compute"))

(defmacro aif args
        (let              (it  (car args))
          (if  it
             (cadr args)
             (caddr args))))

(let (it (calculate))
  (if it
    (print it)
    (error "does not compute")))
```

Surround with parentheses

```
(aif (calculate)
  (print it)
  (error "does not compute"))

(defmacro aif args
  (list 'let (list (list 'it  (car args)))
    (list 'if 'it
              (cadr args)
              (caddr args)))))

(let (it (calculate))
  (if it
    (print it)
    (error "does not compute")))
```

Quasiquote

```
(aif (calculate)
  (print it)
  (error "does not compute"))

(defmacro aif args
       `(let              (it  ..........)
          (if  it
             ............
             ............)))

(let (it (calculate))
  (if it
    (print it)
    (error "does not compute")))
```

Unquote

```
(aif (calculate)
  (print it)
  (error "does not compute"))

(defmacro aif args
       `(let              (it ,(car args))
          (if  it
            ,(cadr args)
            ,(caddr args))))

(let (it (calculate))
  (if it
    (print it)
    (error "does not compute")))
```