My first document

John Doe

2013-09-01

1 All about Macros

1.1 the BAD-OR macro

Let us define a two argument BAD-OR macro in terms of LET and IF

```
(defmacro bad-or (a b)
  (let ((tmp a))
     (if tmp tmp b)))
```

Now let us put the macro to work in an environment where TMP is bound

```
(let ((tmp 5))
  (bad-or #f tmp))

=> (let ((tmp 5))
    (let ((tmp #f))
     (if tmp tmp tmp)))

=>  (let ((tmp #f))
      (if tmp tmp tmp))

=>  (if #f #f #f)

=>  #f
```

2 The need for hygienic macros

```
(let ((tmp 5))  =>   5
  (or #f tmp))
```

3 defmacro

```
(defmacro my-unless (condition &body body)
  `(if (not ,condition)
     (progn
       ,@body)))
```

3.1 free variable capture

The problem is when inadvertant variable capture occurs there is no telling when or how the bug will manifest itself. An almost impossible task to debug this sort of problem.

If one bad macro is allowed then all other macros that depend on that will become susceptible to

4 kohlbecker

5 macro stepper and debugger

6 continuation passing macros