

# Dynamic Features for Groovy 1.6 - 2.0

## Outline

We need to improve the meta programming capabilities that Groovy offers. This document outlines the future improvements proposed for the language.

### Easy overloaded method invocation

Currently it is unnecessarily hard to dynamically obtain and invoke overloaded methods. I propose improvements to the MetaClass so given a class as follows:

```
class OverloadMe {
    def doOne(String s) { s }
    def doOne(Integer i) { i }
}
```

You can easily obtain and invoke the method as follows:

```
def obj = new OverloadMe()
def method = obj.metaClass.doOne

if(method) {
    println method(obj,"one")
    println method(obj,1)
}
```

The way this would work is the MetaClass system would return a MethodProxy or some similar class that would contain references to all MetaMethods and correctly dispatch to the right overloaded method

To avoid ambiguities this should also work with respondsTo

```

def obj = new OverloadMe()
def method =
obj.metaClass.respondsTo(obj, "doOne")

if(method) {
    println method("one")
    println method(1)
}

```

## Easier Per instance Meta programming

Currently it is harder than it needs to be to do per instance meta programming in Groovy. `ExpandoMetaClass` currently applies to all instances and although you can do tricks to make a `ExpandoMetaClass` only apply to an instance it is not clean.

I propose the ability to apply per instance methods by accessing the `metaClass` of the instance. This would be consistent with our usage of the `metaClass` on the `java.lang.AClass`. For example:

```

// a global class level change
String.metaClass.upper = {
    delegate.toUpperCase() }

def str = "hello"

// applies only to this instance
str.metaClass.lower = {
    delegate.toLowerCase() }

```

Achieving the above with Groovy objects will be trivial and just a matter of swapping out the meta class of the instance. For Java objects however we will need a instance to `MetaClass` registry of some sort.

## Make the default `MetaClass` an `ExpandoMetaClass`

I think the whole `ExpandoMetaClass.enableGlobally()` thing needs to go and we need to make EMC (or the new implementation of EMC following the MOP changes) the default

## **Add support for class level and instance level Mixins**

Maybe as per <http://docs.codehaus.org/display/GroovyJSR/Mixins>

Or maybe something that is like Scala traits. Needs more discussion.