GContracts

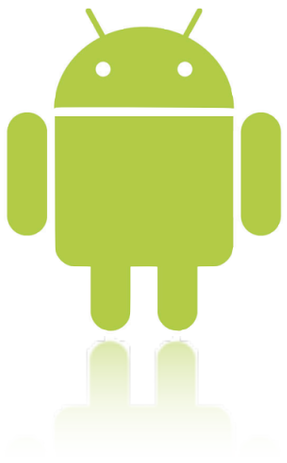
Andre Steingress

@sternegross/@gcontracts

Europe 2011 Copenhagen, Denmark


GRAILS


Groovy




Mit Stellenmarkt
Javamagazin
Internet & Enterprise Technology


GROOVYMAG
for groovy and grails developers
Groovy Under the Hood
Contract-Oriented Programming with Groovy Letting Contracts
Enhance Software Correctness
GrEvoca Condensed A Review of the First U.S. GroovyGrails Conference
Natural Language Processing in Groovy
Slimmed Down Software - A Lean, Groovy Approach
Komplettes
Buch
auf CD!

LINZ, AUSTRIA


Terminator 3: The Redemption developer Paradigm Entertainment publisher Atari
© GameWallpapers.com hosted by JTLnet.com


Am besten eiskalt.
Kohlensäurehaltig.
Red Bull
ENERGY DRINK
Mit Taurin.
Belebt Geist und Körper.
473 ml

www.ejug.at

ABOUT GCONTRACTS.

GCONTRACTS

DESIGN BY CONTRACT (tm) FOR GROOVY
Groovy 1.7+ (was 1.6 loooong time ago)

BSD LICENSE

https://github.com/andresteingress/gcontracts

JUMP START.

```java
import org.gcontracts.annotations.*

// by http://www.javacoffeebreak.com
public class Account
{
    protected double balance;

    public Account( double amount )
    {
        balance = amount;
    }

    public Account()
    {
        balance = 0.0;
    }

    public void deposit( double amount )
    {
        balance += amount;
    }

    public double withdraw( double amount )
    {
        if (balance >= amount)
        {
            balance -= amount;
            return amount;
        }
        else
            // Withdrawal not allowed
        return 0.0;
    }

    public double getBalance()
    {
        return balance;
    }
}
```

LET'S REVIEW THIS ACCOUNT CLASS.

Googled, found at javacoffeebreak.com

PREREQUISITES.

```
@Grab(
   group='org.gcontracts',
   module='gcontracts-core',
   version='[1.2.3,)'
)
```

no additional dependencies...

```
import org.gcontracts.annotations.*
```

3 ANNOTATIONS.

Commonalities

- special **assertion types**
  - specifying boolean expressions
  - **closure** instances as **annotation parameters**

```
@MyAnnotation({ param?.size() > 0 })
```

@Requires

- **precondition**
- **a method's** bouncer

@Ensures

- **postcondition**
- **a method‘s** mother

@Invariant

- **class-invariant**

- **an object‘s** god.

```java
import org.gcontracts.annotations.*

// by http://www.javacoffeebreak.com
public class Account
{
    protected double balance;

    public Account( double amount )
    {
        balance = amount;
    }

    public Account()
    {
        balance = 0.0;
    }

    public void deposit( double amount )
    {
        balance += amount;
    }

    public double withdraw( double amount )
    {
        if (balance >= amount)
        {
            balance -= amount;
            return amount;
        }
        else
            // Withdrawal not allowed
        return 0.0;
    }

    public double getBalance()
    {
        return balance;
    }
}
```

LET'S REVIEW THIS ACCOUNT CLASS.

Googled, found at javacoffeebreak.com

INTERNALS.

compile-time

```
@Ensures({ count() == old.count + 1 })
```

```
{ count() == old.count + 1 }
```

```
{ result, old -> assert count() == old.count + 1 }
```

```
class $_gc_closure1 extends Closure {
    def doCall(...)  {
       // ...
       assert count() == old.count + 1
       // ...
    }
}
```

```
   @Requires($_gc_213123123123.class)
```

run-time

```
JVM Arguments: -ea -da

if ($GCONTRACTS_ENABLED) { ... }

org.gcontracts.ViolationTracker.init()


$_gc_result = Putable$_gc_closure1.newInstance(this,
this).call(element, key)

if (org.gcontracts.ViolationTracker.violationsOccured())

    org.gcontracts.ViolationTracker.rethrowFirst()
```

Closure annotation parameters

In Java, there's a limited set of types you can use as annotation parameters (String, primitives, annotations, classes, and arrays of these). But in Groovy 1.8, we're going further and let you use closures as annotation parameters – which are actually transformed into a class parameter for compatibility reasons.

```
import java.lang.annotation.*

@Retention(RetentionPolicy.RUNTIME)
@interface Invariant {
    Class value() // will hold a closure class
}

@Invariant({ number >= 0 })
class Distance {
    float number
    String unit
}

def d = new Distance(number: 10, unit: "meters")

def anno = Distance.getAnnotation(Invariant)
def check = anno.value().newInstance(d, d)

assert check(d)
```

Closure annotation parameters open up some interesting possibilities for framework authors! As an example, the GContracts project, which brings the "Design by Contract" paradigm to Groovy makes heavy use of annotation parameters to allow preconditions, postconditions and invariants to be declared.

Officially Supported since Groovy 1.8!

SOFTWARE WITHOUT CRISIS

Spock Framework, Groovy, and other IT-related stuff

About Me

Type text to search here...

Home > Groovy > Empowering Annotations with Groovy Closures

Empowering Annotations with Groovy Closures

March 4, 2010 pniederw Go to comments Leave a comment

A few days ago, we received the following feature request for the Spock Framework:

> Run a specification only if JDK 1.6 or higher is present, and skip it otherwise.

Given Spock's extensible nature, it would have been easy to jump straight in and implement this as an extension activated with @JdkVersion(x). But this approach didn't feel right to us. Surely, users would soon come up with other constraints (JDK version less than x, OS version equal to y, and so on), and we wanted a solution that could handle them all. Alas, if Groovy just allowed us to write:

```
1 @RunIf({ jdkVersion >= 1.6 })
2 class MySpec extends Specification { ... }
```

Now, this would be heaven. No more @JdkVersion, @OsVersion, and so on. Instead just *one* annotation that takes an arbitrary Groovy expression (contained in a closure) and is processed by *one* Spock extension. Problem solved – forever. (At this point you might wonder where "jdkVersion" comes from. I assume we would provide some convenience properties like it, but with the power of Groovy at your fingertips, you could always go beyond them.)

Staring at my screen, I became a little saddened because I knew that Groovy wouldn't let me stick a closure into an annotation. But wait – the code in my IDE wasn't underlined in red! Was it a bug, or were I up to something here? Hastily I fired up Groovy's AST Browser and noticed that Groovy's grammar *does* allow closures as annotation values. It is only the compiler's code generator that eventually raises a compile error. At this point I became nervous. Could it be that I was just one AST transformation away from heaven?

In case you haven't heard about AST transformations yet, let me quickly explain what they are: a way to hook into the Groovy compiler and participate in a program's compilation. What's particularly nice about AST transformations is that they are trivial to activate from the outside: Just put a transformation on the class path, and the compiler will pick it up automatically. Some of Groovy's built-in features like @Lazy and @Immutable are in fact based on AST transformations, and so are many of Spock's features.

With a rough idea in my head, I started coding an AST transformation that would allow me to pass closures as annotation values. A few hours later, I had a pretty compelling prototype done. Let me show you an example of what you can do with it *today*. Let's say we wanted to build a simple field-based POGO (Plain Old Groovy Object) validator. To start out, let's define an annotation type:

```
3 @Retention(RetentionPolicy.Runtime)
4 @interface Require {
5   Class value()
6 }
```

The annotation type has one attribute of type Class. That's where we will stick in our closure.

RSS feed

Categories

- Groovy
- Spock Framework

Twitter Updates

- RT @glaforge: Final version of #groovy 1.8 released! Please read the release notes http://bit.ly/groovy-18 and enjoy! 1 week ago
- RT @vaclav_pech: I'm happy to see my article on GPars Java and Groovy actors up on #drdobbs http://bit.ly/i7XuTu 1 week ago
- RT @HamletDRC: Getting started with Spock and Groovy screencast http://goo.gl/KsI0b (vote at http://goo.gl/61fz4) #groovy #spock 3 weeks ago

I speak at GR8Conf Europe 2011 May 17th-19th Copenhagen, DK

I'm speaking at ÜBER CONF

Meta

- Register
- Log in
- Entries RSS
- Comments RSS
- WordPress.com

Works since Groovy 1.6

http://pniederw.wordpress.com/2010/03/04/3/

MORE FEATURES.

LABELING

```
@Requires({
    valid_capacity: count() <= capacity()
    key_not_empty:  !key?.isEmpty()
})
@Ensures({
    updated_count: count() == old.count + 1
    contains_element: has(key)
})
def put (def element, def key) {
    store[key] = element
}
```

INTERFACE CONTRACTS

```
interface Putable {
    @Requires({
        valid_capacity: count() <= capacity()
        key_not_empty:  !key?.isEmpty()
    })
    @Ensures({
        updated_count: count() == old.count + 1
        contains_element: has(key)
    })
    def put (def element, def key)

    boolean has(def key)

    int count()
    int capacity()
}
```

INTERFACE CONTRACTS

```
class Dictionary implements Putable {

    private int capacity = 0
    private def store = [:]

    def Dictionary(int capacity = 10)  {
        this.capacity = capacity
    }

    def put (def element, def key) {
        store[key] = element
    }

    boolean has(def key)  {
        store.containsKey(key)
    }

    int count() { return store.size() }
    int capacity() { return capacity }
}
```

INHERITANCE

- ✓ INTERFACE CONTRACTS
- ✓ ABSTRACT CLASSES
- ✓ CONCRETE CLASSES
- ✓ MIXED

INHERITANCE

✓ ACROSS COMPILATION UNITS

MICRO CONTRACTS

```
@Precondition
@AnnotationContract({ it != null })
public @interface NotNull {}
```

REUSABLE CONTRACT PART

MICRO CONTRACTS

```
class Dictionary implements Putable {

    private int capacity = 0
    private def store = [:]

    def Dictionary(int capacity = 10)  {
        this.capacity = capacity
    }

    def put ((@NotNull def element, @NotNull def key) {
        store[key] = element
    }

    boolean has(def key)  {
        store.containsKey(key)
    }

    int count() { return store.size() }
    int capacity() { return capacity }
}
```

Grails Integration

```
@Service
@Invariant({ anotherService != null })
class MyService {

    def anotherService

    static transactional = true

    // ...
}
```

- Container extends object construction time
- @Service stereotype causes CI validation in @PostConstruct

GroovyDoc Integration

Constructor Summary

EiffelStack()
EiffelStack(List preElements)

Method Summary

def	count()
boolean	*@Ensures({ result == true ? count() > 0 : count() >= 0 })* has(def item)
boolean	is_empty()
def	*@Requires({ !is_empty() })* last_item()
def	*@Ensures({ last_item() == item })* put(def item)
def	*@Requires({ !is_empty() })* *@Ensures({ result != null })* remove()
def	*@Requires({ !is_empty() })* *@Ensures({ last_item() == item })* replace(def item)

https://github.com/andresteingress/gcontracts/wiki

ABOUT CONTRACTS.

MR.
„DESIGN BY CONTRACT“


BJECT-ORIENTED
SOFTWARE CONSTRUCTION
SECOND EDITION
CD-ROM INCLUDED
The Most Comprehensive, Definitive O-O Reference Ever Published
An O-O Tour de Force by a Pioneer in the Field
CD-ROM Includes Complete Hypertext Version of Book AND Object-Oriented Development Environment
Winner
BERTRAND MEYER

WHY CONTRACTS?

SPOCK, JUNIT, et. al. ARE

„~~THE CHINAMAN IS~~ NOT THE ISSUE HERE, DUDE.“

External API Changes

New Programmers

Changing Requirements

Assumptions

Additional Libraries

New Use-Cases

More Clients

Code Conventions

New Knowledge

External API Changes

New Programmers

Changing Requirements

Assumptions

Additional Libraries

New Use-Cases

More Clients

Code Conventions

New Knowledge

WHAT DID YOU PROGRAM LAST WEEK?

CAN YOU RECALL YOUR THOUGHTS, MODELS?

WHY SHOULD ANYONE ELSE KNOW ABOUT THEM?

BUILD YOUR PLAYGROUND.

http://about.me/asteingress

Thank you!