

Groovy DevCon 6

Metadata

When	May 16th and 20th 2011
Where	Copenhagen, Denmark
Participants	<ul style="list-style-type: none">• Paul King• Dierk König• Guillaume Laforge• Jochen Theodorou• Andrew Eisenberg• Andres Almiray• Peter Ledbrook• Burt Beckwith

Topics discussed

- Integration of the Eclipse compiler (Andrew & Jochen)
 - gmaven and/or groovy-eclipse-compiler for maven
 - further investigate gmaven 3 polyglot support
 - does it reduce the need for some GMaven
 - for now don't improve gmaven, use ant bridge instead
 - after joint compiler exists will use that
 - AST transformation issues?
 - should be okay
 - patches for Groovy
 - create a top-level JIRA issue
 - attach the patches related issue to the main issue
 - try and see if we can apply them all and/or adapt them on both sides
 - trim the current big package
 - CI task to automate the process
 - ant / maven / gradle support?
 - Groovy-eclipse compiler to generate stubs
- Groovy governance (Guillaume)
 - Idea: meritocracy but aligned with codehaus guidelines
 - you must vote on big issues
 - have an application somewhere to assist in voting
 - Yes/No/"Maybe"/Veto/Yes (and I will implement)
 - Someone must take ownership of implementing the issue
 - limited time frame
- Better capturing of API changes (Andres for investigation, potentially Jochen for an ASM tool)
 - investigating clirr / JDiff
 - potentially implement our own based on ASM if we need
- Groovy documentation (Guillaume)
 - add MrHaki to SearchGroovy.org
 - have versioned documentation /18/api
 - website

- work with a designer for having a sexier / more professional website
- specification
 - executable specification
 - inspiration from GinA for the content: chapter 3-9
 - more for improving quality, coverage, etc.
- wiki
 - problem of outdated content, different styles
 - keep the wiki for user contributions
- Git / GitHub
 - Git Codehaus as the main source reference w/ GitHub mirroring?
 - Inspiration
 - Look at what GPars is doing and see if it's a better scheme
 - Look at the Wine project approach
 - Involve Matthew McCullough into the loop
 - Restructuring the SVN repository
 - Involve Ben Walding for picking up "just" groovy-core
- Build improvements
 - Switch over to Gradle as main build technology
 - switch to Gradle for 1.9
 - including removing the Ant build
 - remove the pom?
 - for 1.8
 - maintain the Gradle build also for 1.8
 - keep the Ant build for 1.8
 - but try to make a release with Gradle
 - check why some tests are failing within Gradle
 - Some integration tests for e.g. command line params
 - Ability to build groovy core using groovy-eclipse (Andrew)
 - requires some restructuring of groovy project
 - Using other testing frameworks, e.g. Spock
 - for modules potentially
 - not for core
- AST (Jochen & Andrew)
 - better GenericsType API...I want to be able to ask these questions:
 - Are you a type parameter or a type argument?
 - What is the class that declares the type parameter?
 - If a type argument, what parameter does it apply to?
 - I have a ClassNode that uses generics and is a redirect. In the ClassNode, I need to map from the type argument to the declared parameters. Not easy since class may have multiple super classes each declaring their own type parameters.
- AST Transform Chaining (Andres)
- Modularisation (Paul)
 - Grab support and IDEs (Andrew)
- GDSL / DSLD (Peter/Andrew and Groovy Core team)
 - Andres has started some of the DSLD for @Bindable and others
 - to be continued at Hackergarten
 - contribute those GDSL / DSLD to Groovy itself
 - Dierk's idea 😊 "grumpy" tool : idea of a "pedantic" flag involving GDSL / DSLD to check dynamic variables, unresolved references, etc
 - but probably requires the eclipse compiler

- could turn the CodeNarc rules into compilation failures if not respected
- Antlr 3 rewrite
 - Lidia, GSoC student working on that
 - see how far it goes, if it goes through
 - better tooling support, cleaner room grammar implementation
 - check if command chains could be used as expressions (for example as params of methods, etc)
- Project Coin
 - done:
 - binary literals
 - underscore in numbers
 - diamond operator
 - at least, syntax level
 - multicatch
 - switch in string
 - not done:
 - varargs stuff not needed because the groovy compiler doesn't complain anyway 😊
 - once grumpy is there...
 - try / resources
 - see examples below for brainstorming
 - defer decision
- Investigate @Deprecated / @(Un)Documented / @Override warnings / compilation errors
- Performance improvements
 - ongoing primitive improvements
 - invokeDynamic
 - still watching the space, waiting for at least release candidates of JDK 7
- Look into whether the GDK can benefit from command-chains
- New features discussions
 - parser combinators
 - should be a module / experiment
 - have a look at (G/J)Parsec
 - tail recursion
 - possibilities
 - treturn keyword
 - a form of trampoline for methods
 - @TailRecursion
 - pattern matching (Guillaume and Jochen)
 - needs a GEP
 - needs obvious use cases that cannot be handled otherwise
 - inspiration of pointcuts for DSLD
 - functional aspects (investigate what makes sense for Groovy)
 - come up with use cases that would warrant inclusion
 - develop GEP(s) if needed
 - possible features
 - monads : no immediate action
 - for comprehensions (monadologie)
 - generators / lazy streams / iterators / coroutines
 - persistent / lazy collections
 - experimental @Checked annotation
 - needs to be fleshed out
 - can be pushed to the groovy transforms module

- symbol concept
 - no obvious use case for now
 - ask Graeme for use cases
- a “native” template engine
 - doesn’t seem worthwhile
- Groovy 2.0 (topic delayed)
 - Deprecating “ugly” features
 - one char String is char
 - method calls taking a list as parameter is spread when only one method
 - one arg method called w/o arg arg is null (check w/ closures) [needs GEP](#) issue: consistency of method vs class definition and call; also: methods that take a map parameter and the map is empty
 - separate “worlds” for specifying metaclasses
 - lexical categories (Dierk is against it 😊)
 - operator overloading of &&, ||, and ! and friends
 - AST domain classes more bean like
 - compareTo>equals
- ConfigSlurper often causes trouble (see long list of issues on JIRA) and needs to be rewritten
- going through JIRA issues

Try with resource brainstorming

```

try (
    FileReader reader = new FileReader();
    OutputStream stream = new
OutputStream();
) {
    // do stuff
}

file.withReader {}

anything.withCloseable {}
[].withCloseable {}

autoClose(reader, writer) {
    reader = new FileReader()
    writer = new Writer()
}

autoClose({ new FileReader() }, { new
Writer() }) { reader, writer -> }
autoClose(reader = new FileReader(); new
Writer(reader)) { reader, writer -> }

autoClose {
    reader = new FileReader()
    writer = new Writer()
} with { }

```