SKILLS MATTER

JetGroovy: Groovy with pleasure

2007

GRAILS eXCHANGE

http://www.grails-exchange.com | http://www.grails.org | http://skillsmatter.com

Who We Are

- Eugene Vigdorchik
  - 10 years in software development
  - Today - Software Architect at JetBrains
- JetBrains
  - World-known vendor of intelligent SD tools
- IntelliJ IDEA
  - Famous productivity-oriented Java IDE

JetGroovy Plugin

Started in March 2007

Source code is available at
http://svn.jetbrains.org/idea/Trunk/groovy

3 active developers + external contributions

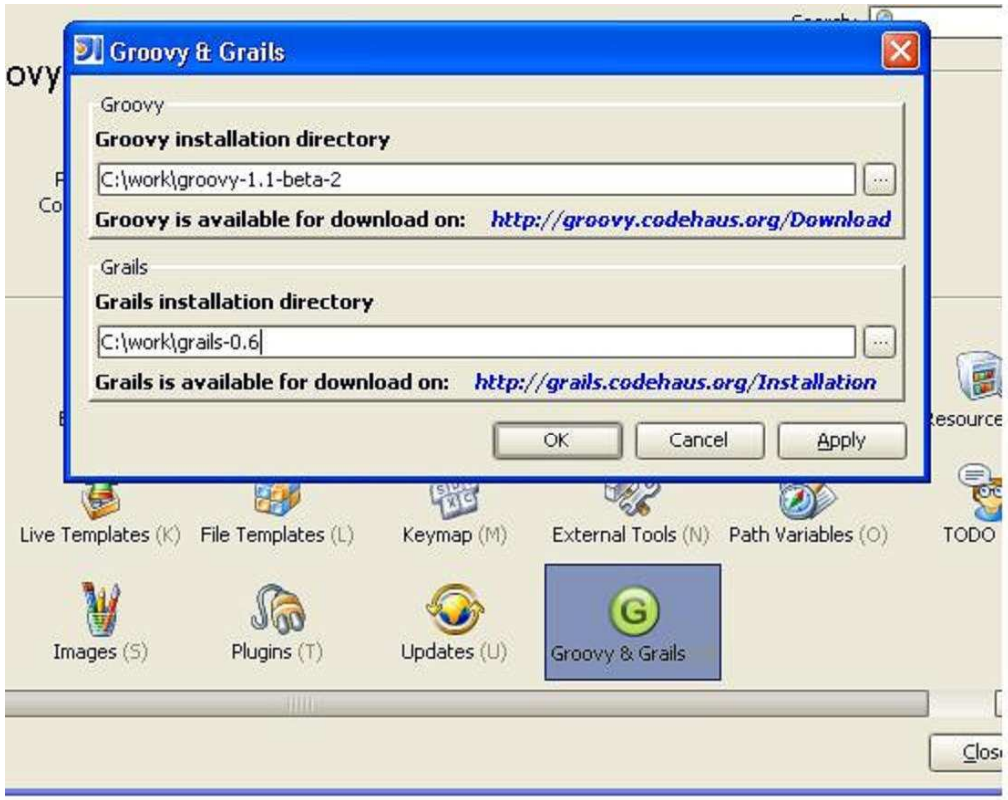
70 kLoc of code

Part of the plugin is written in Groovy

What's so special about it?

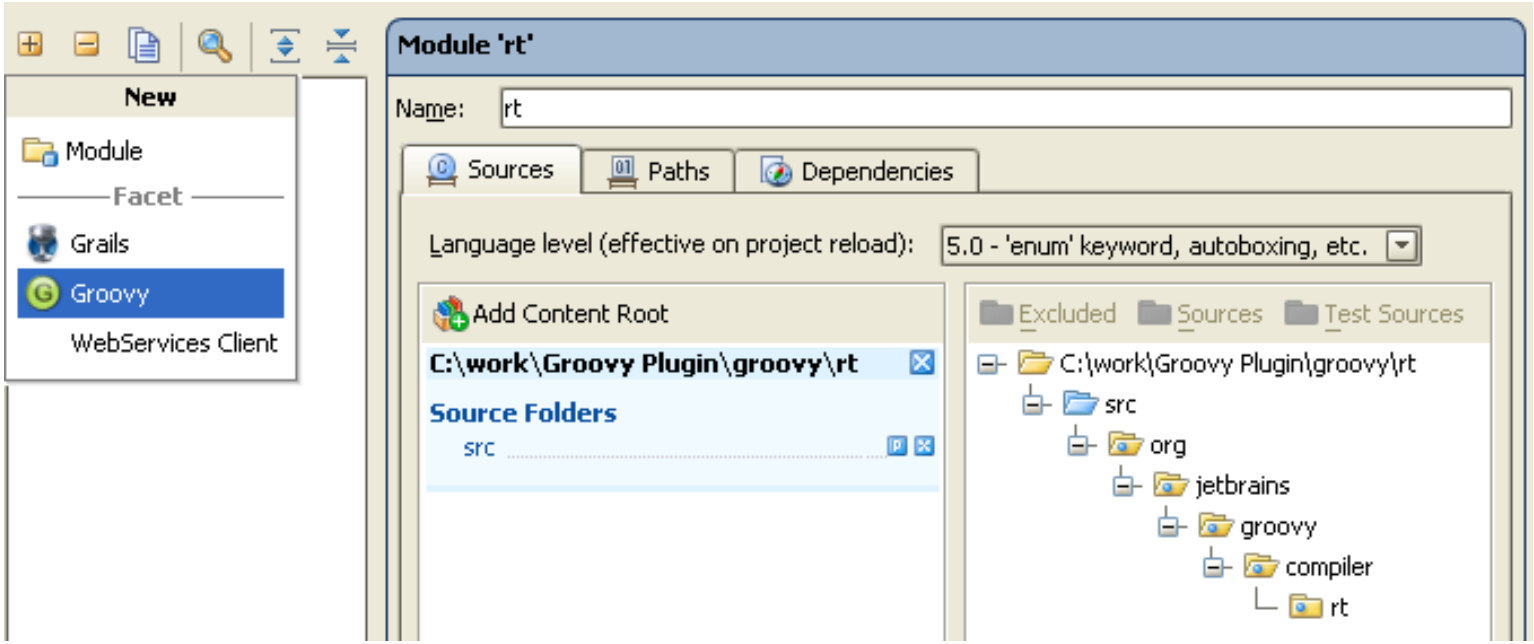
- Easy and intuitive configuration
- More than just syntax highlighting
- Seamless interoperability with Java
- Unparalleled type inference
- Tight integration with advanced IDE debugger
- Intelligent GSP support

Configuration



Necessary jars are kept in application level libraries

Configuration (2)



- Groovy and Grails support for module is configured through facets. No separate module type needed.

Basic Actions

- Syntax highlighting
- Formatting code
- File structure view
- Structural “surround with”
- Actions for Groovy class and script creation

The same feature set that is available for Java

JetGroovy is type-based

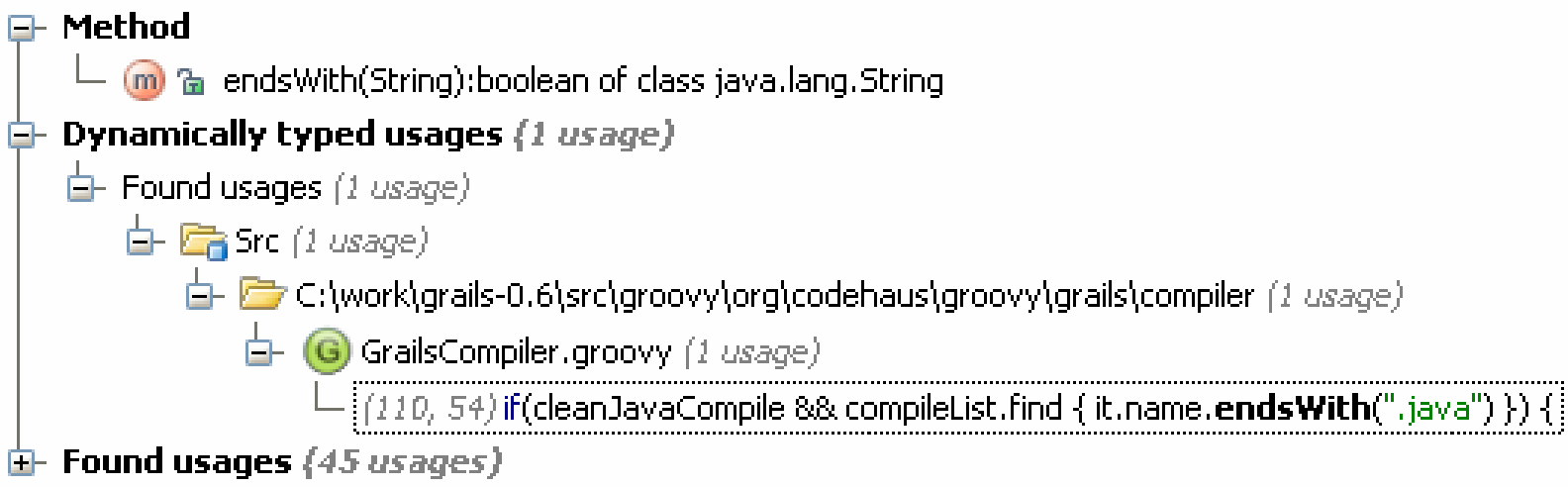
- Groovy allows for optional typing
- Types are mandatory in Java
- Types may be often inferred

When types are not available, pessimistic assumptions are made

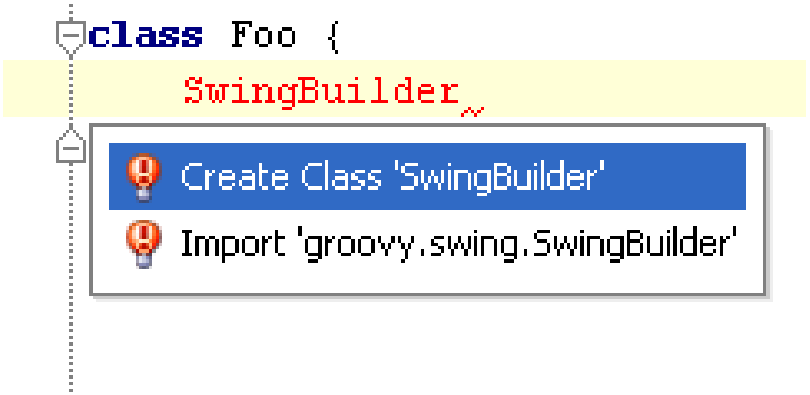
Untyped References

```
if(compileList) {
    // if there are Java sources do a complete re-compile
    if(cleanJavaCompile && compileList.find { it.name.endsWith(".java") }) {
        compileList.clear()
        def m = new IdentityMapper()
        def sfs = new SourceFileScanner(this);
        for(srcPath in src.list()) {
```

Cannot determine type statically



Compiler Errors



- All **groovyc** errors are reported
- Auto-import is available on incorrect class references
- “Implement methods” quick-fix

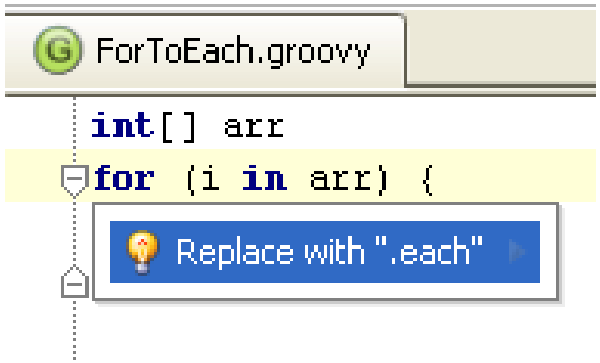
Code Inspections

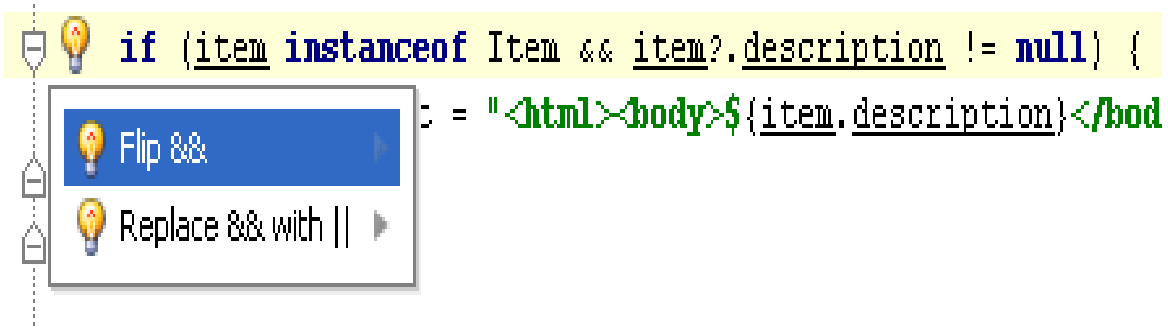
```
if(classpath) configuration.classpath = classpath.toString()
configuration.targetDirectory = destdir
if(encoding)configuration.sourceE
```

Cannot assign 'java.io.File' to 'java.lang.String'

- Check whether:
  - Assignment is correct
  - Method is applicable
  - Private API is used from outside
- Provide warnings instead of errors

Intention Actions





- 20 Groovy specific intentions
  - Convert **for** to **each**
  - Data types conversions
  - Control-flow related actions

Groovy + Java

Groovy needs to call Java:

- Lots of available APIs/legacy code
- Static typing = better performance

Java might call Groovy

- Partial refactoring
- Explicit design

IDE should understand cross-language references

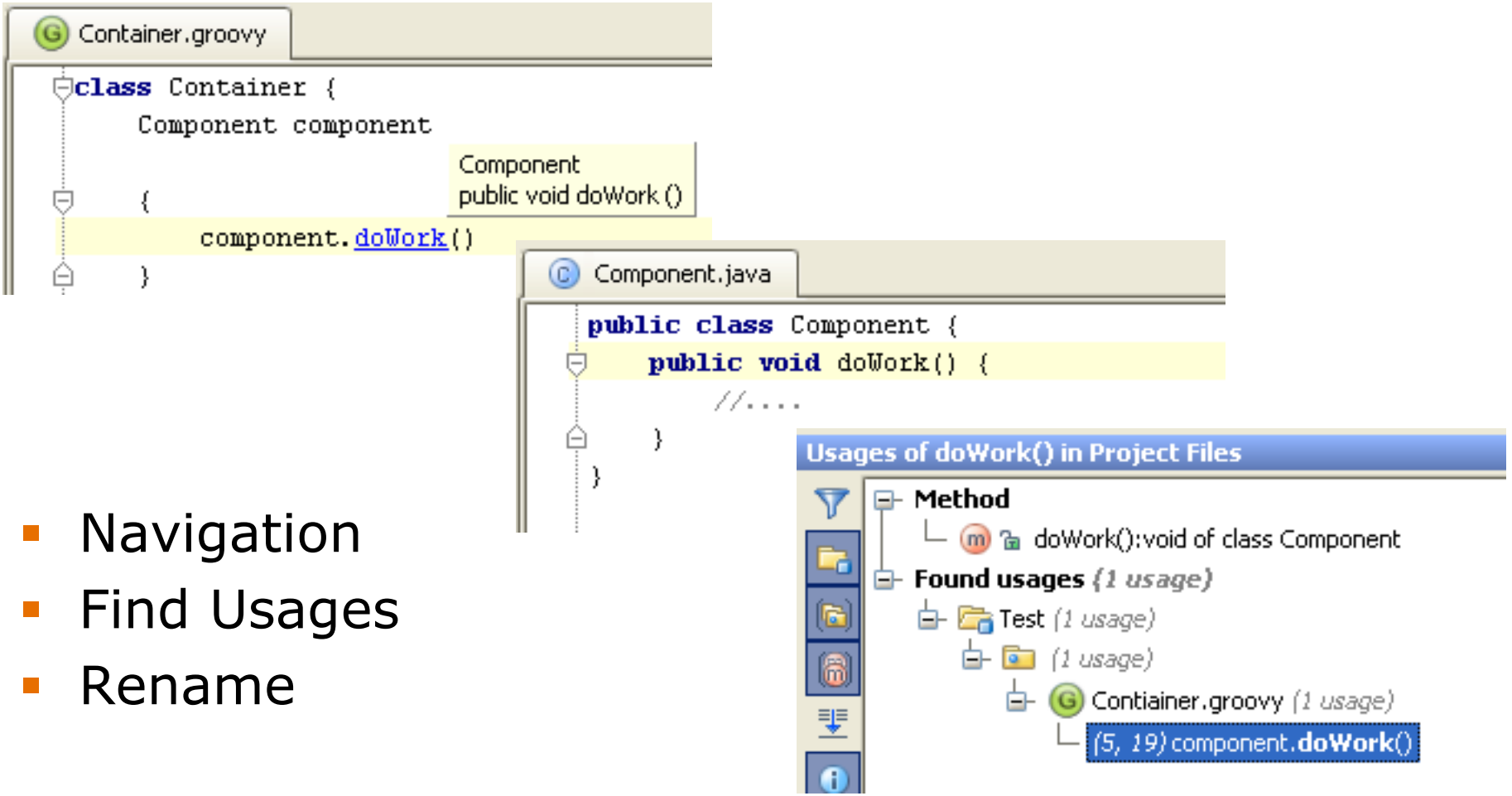
Joint Compilation

3-phase process:

- Java stubs are generated
- Java compiler is invoked on user-written Java code and stubs
- Groovy compiler is called with Java output included in classpath

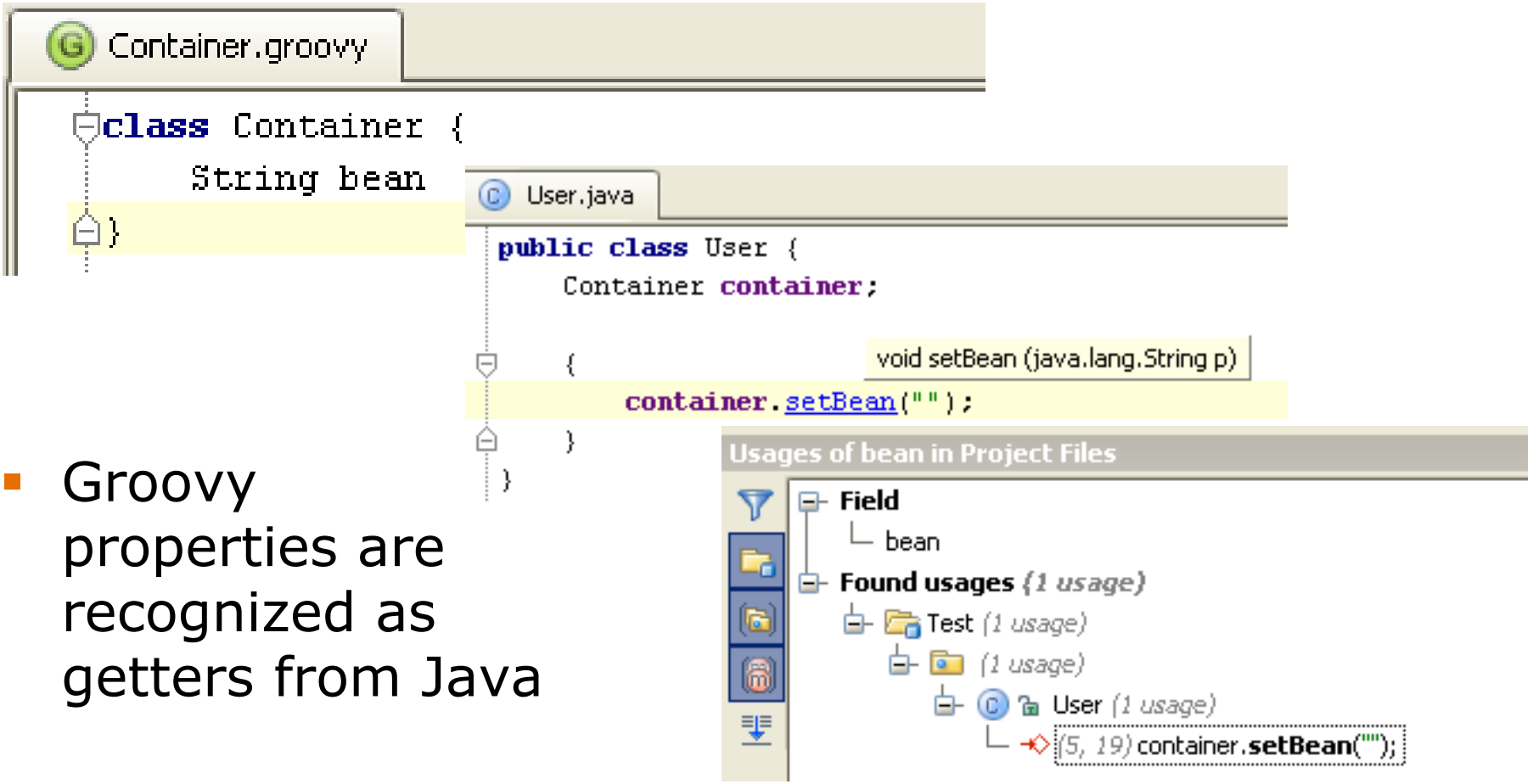
- The same feature is implemented by the groovy-core team

Cross-language Actions(1)



- Navigation
- Find Usages
- Rename

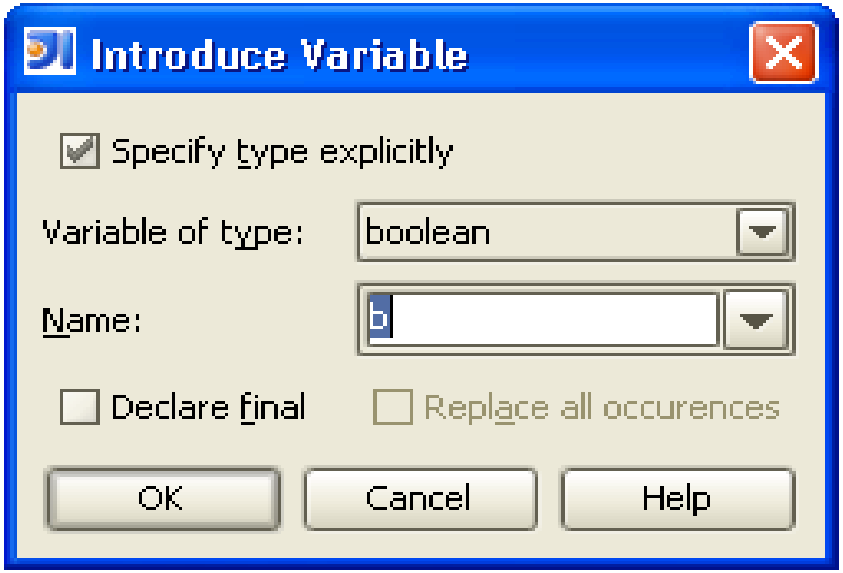
Cross-language Actions(2)



- Groovy properties are recognized as getters from Java

Refactorings

```
class CharUtils {
    def isUpper(def c) {
        Character.isUpperCase c
    }
}
```

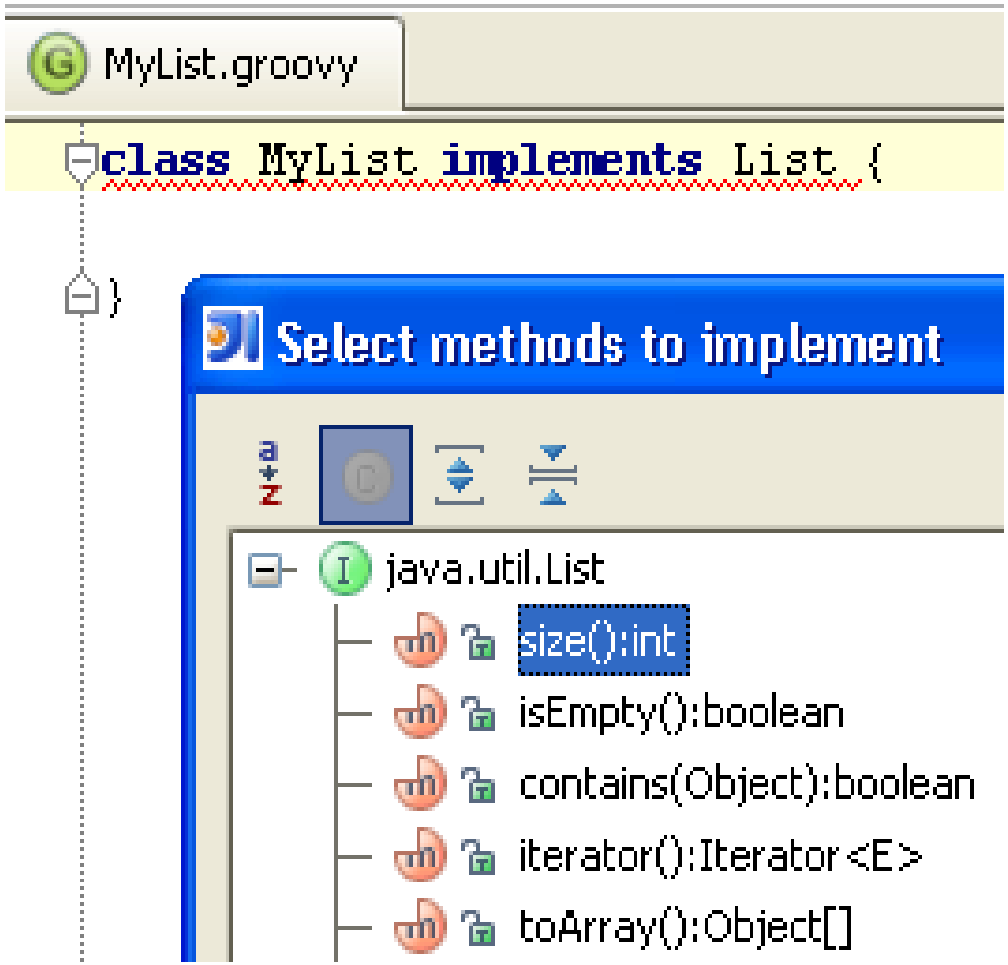


- Rename
- Move Class
- Introduce Variable
- Inline Variable

In progress:

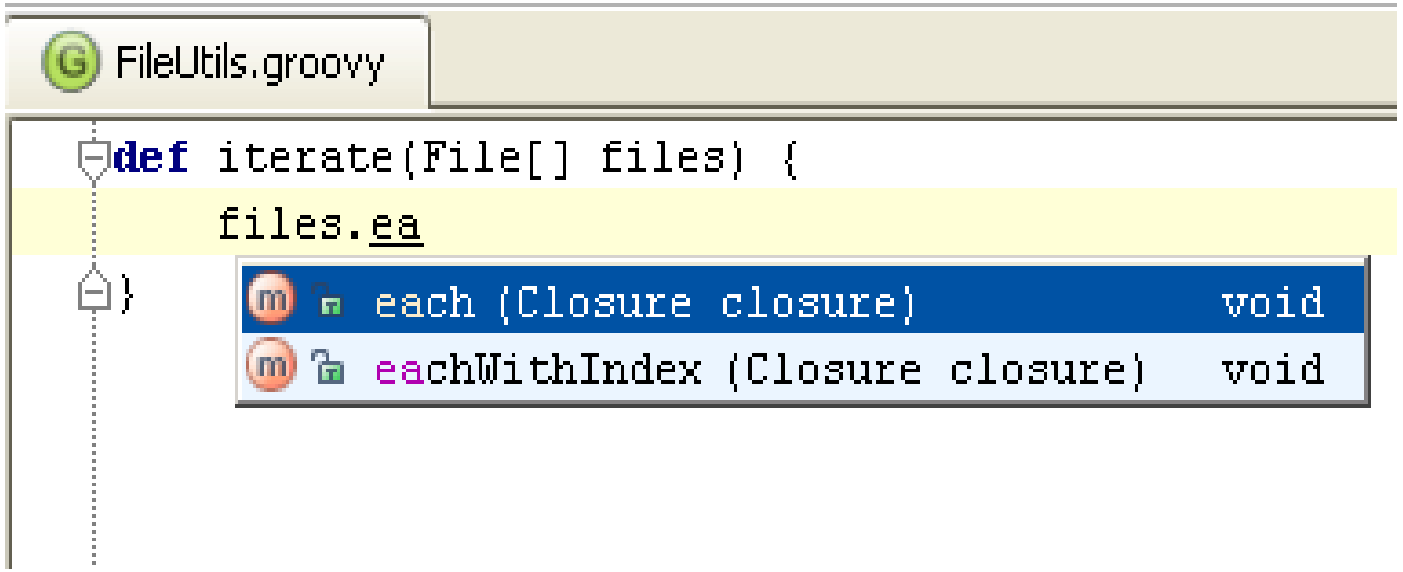
- Extract Method
- Inline Method

Code Generation



- Available as:
  - Quick fixes
  - Editor actions
- Aware of Generics

Code Completion(1)



- GDK is recognized
- Type of qualifier should be known
- References in the scope with same qualifier are suggested when type is unknown.

Code Completion(2)

SwingExample.groovy

```
import groovy.swing.SwingBuilder

def builder = new SwingBuilder()

builder.pa
```

```
panel ()                JPanel
passwordField ()    JPasswordField
```

- Groovy SwingBuilder methods and respective properties are completed

- Accounts for builder setting closure delegate

```
def builder = new SwingBuilder()

builder.panel(){
    but
```

```
button ()                JButton
buttonGroup ()    ButtonGroup
```

Code Completion(3)

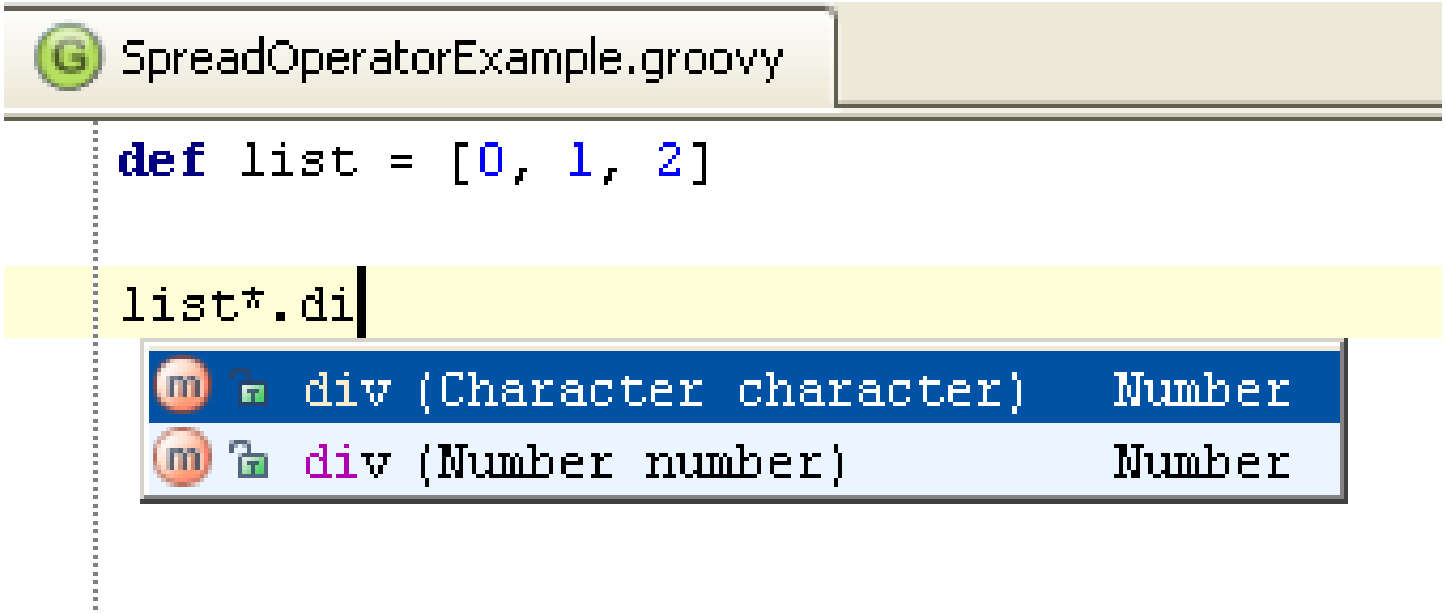
SwingExample.groovy

```
import groovy.swing.SwingBuilder

def builder = new SwingBuilder()

builder.panel(fo
```

```
focusable
focusCycleRoot
focusGained
focusLost
focusTraversalKeysEnabled
focusTraversalPolicy
focusTraversalPolicyProvider
font
foreground
```

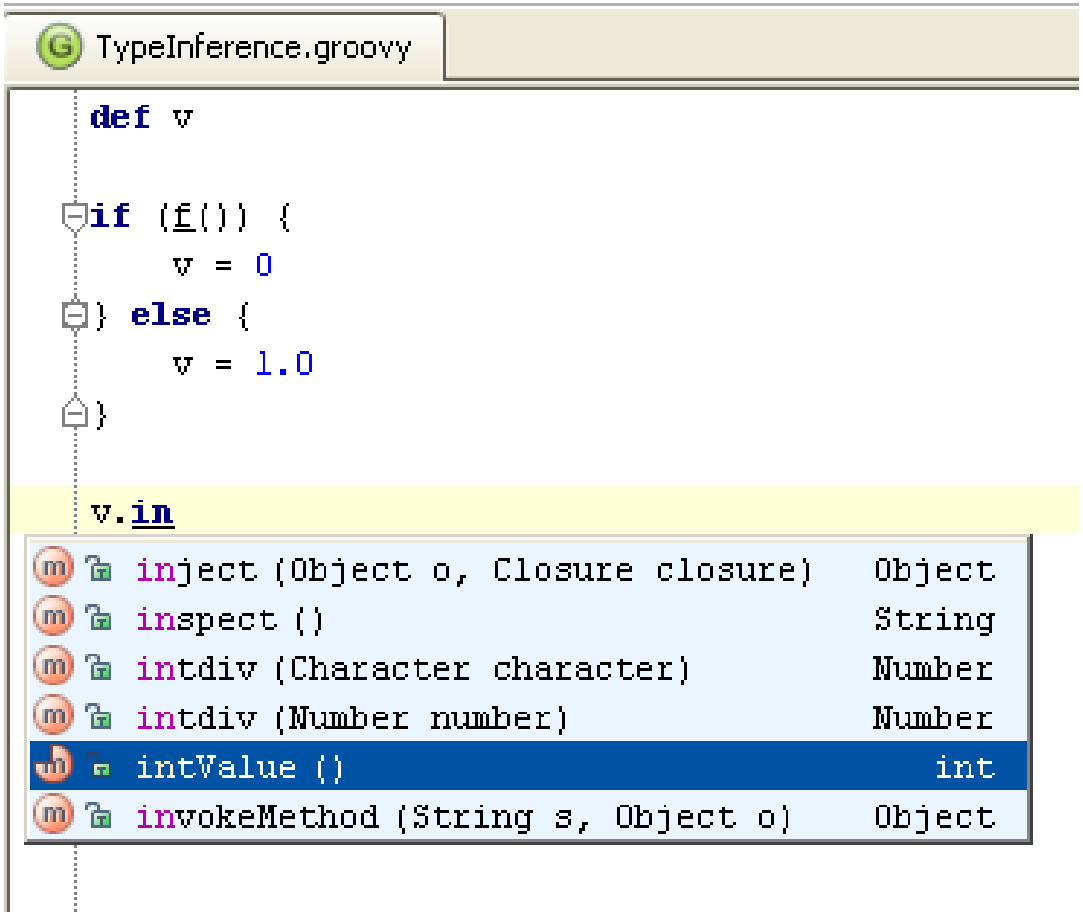
- Listener methods are suggested as closures

Code Completion(4)



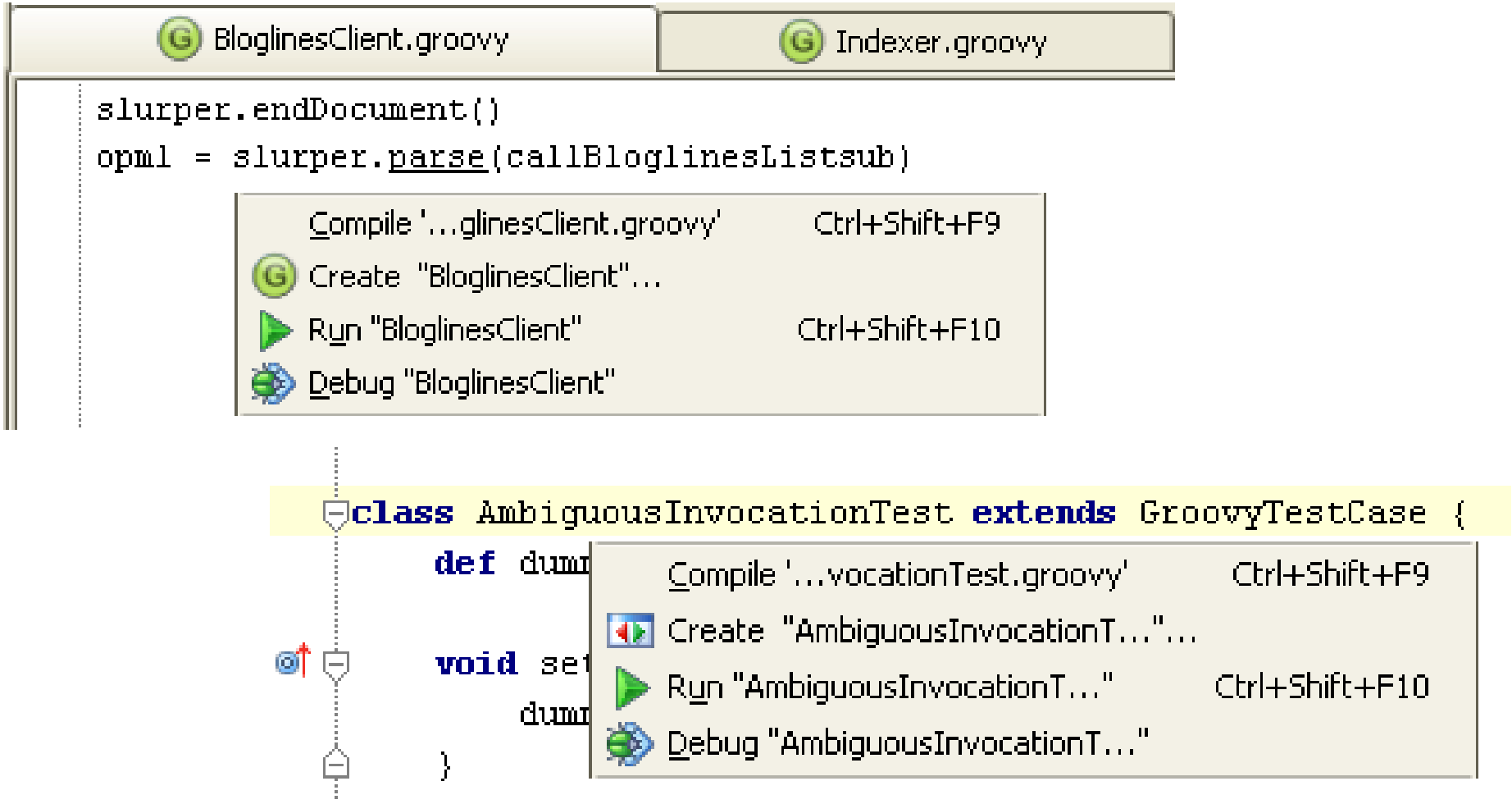
- Spread operator is applied to each list element
- If component type is inferred, then spread operator may be completed

Type Inference

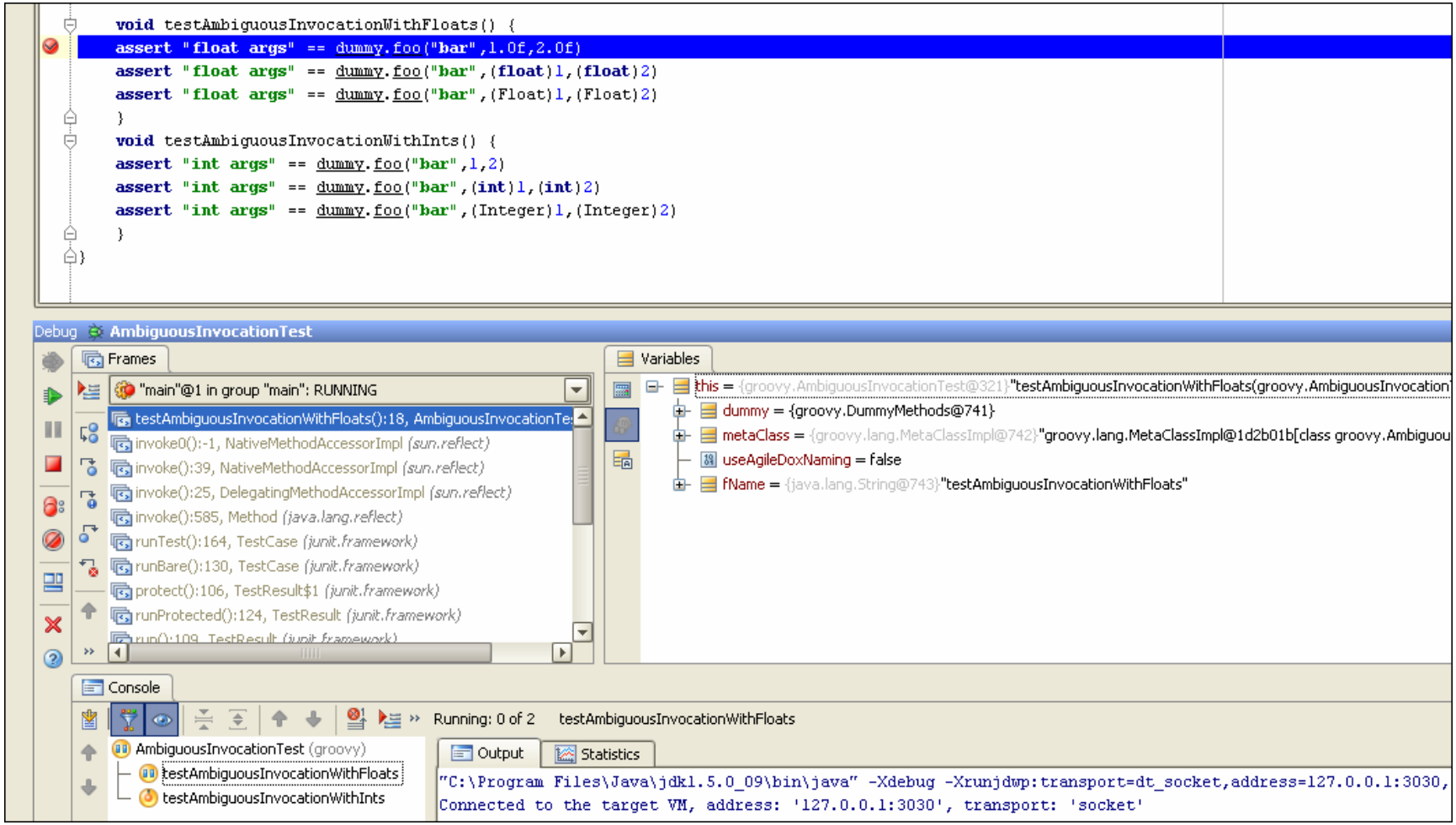


- Type of the method is inferred from all return values
- Type of local variable is inferred from its initializer
- Type of "read" variable reference is inferred from assignments reaching this reference
- Java-like inference for generic methods' type arguments

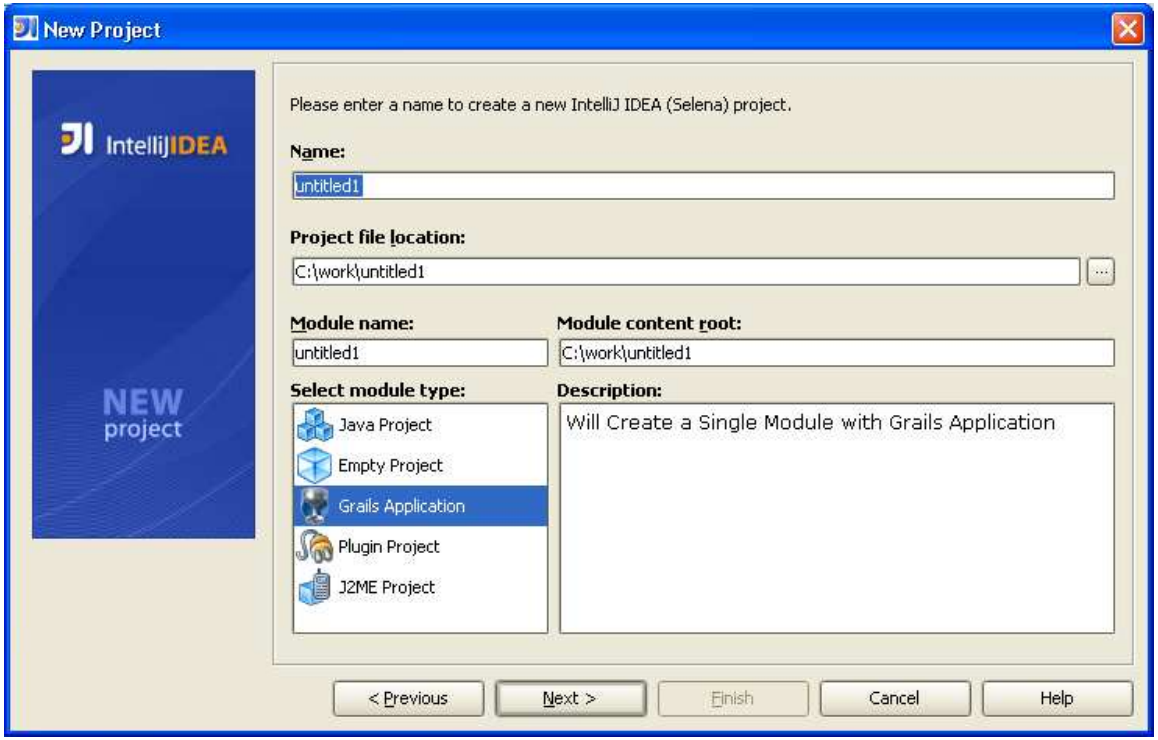
Debug and Run(1)



Debug(2)

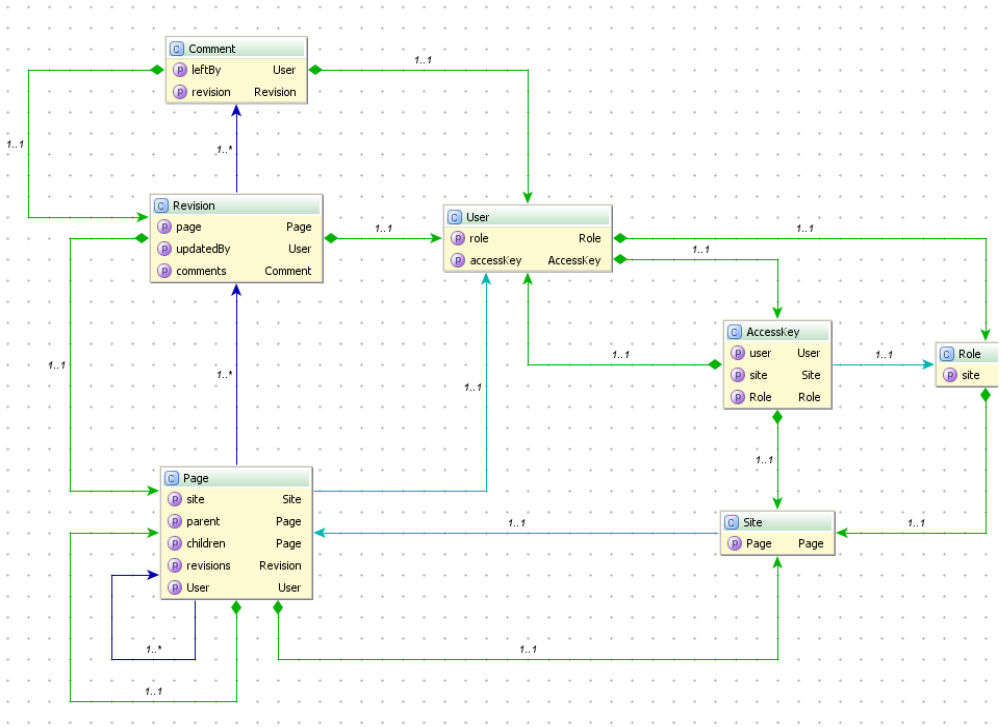


Grails Project Setup



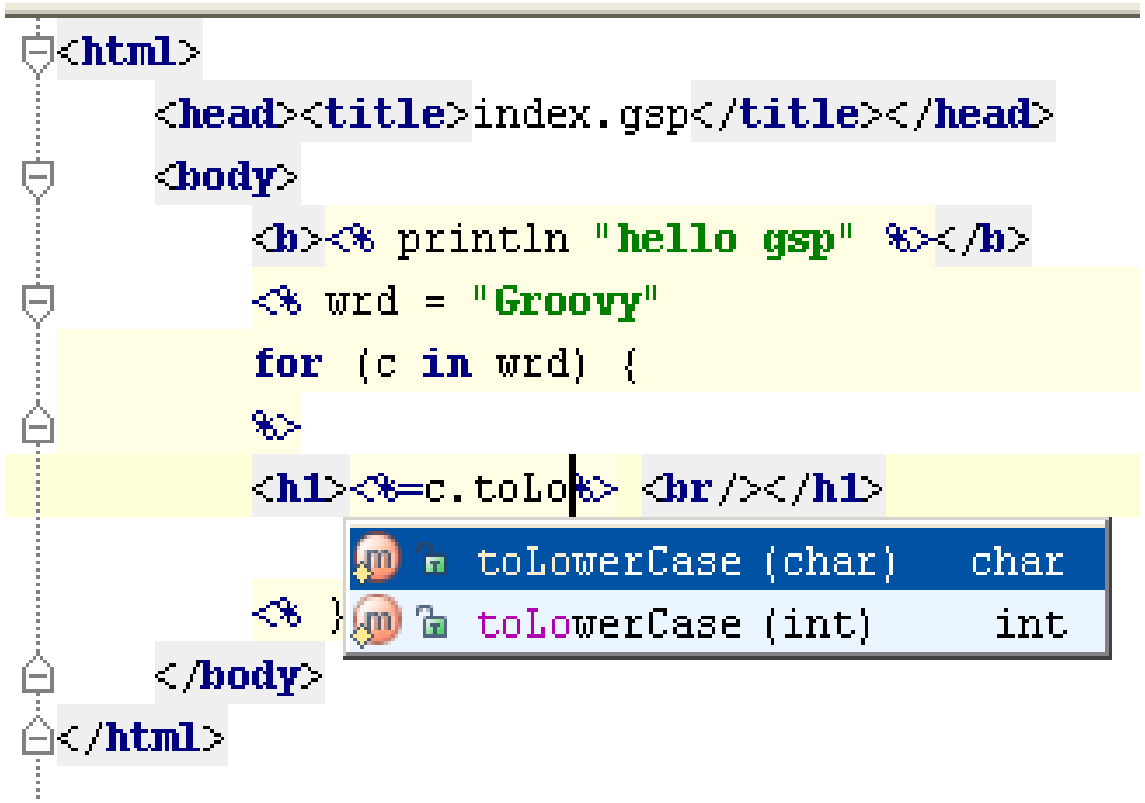
- Dedicated module type
- **`grails create-app`** is invoked

Entity-Relation Diagram



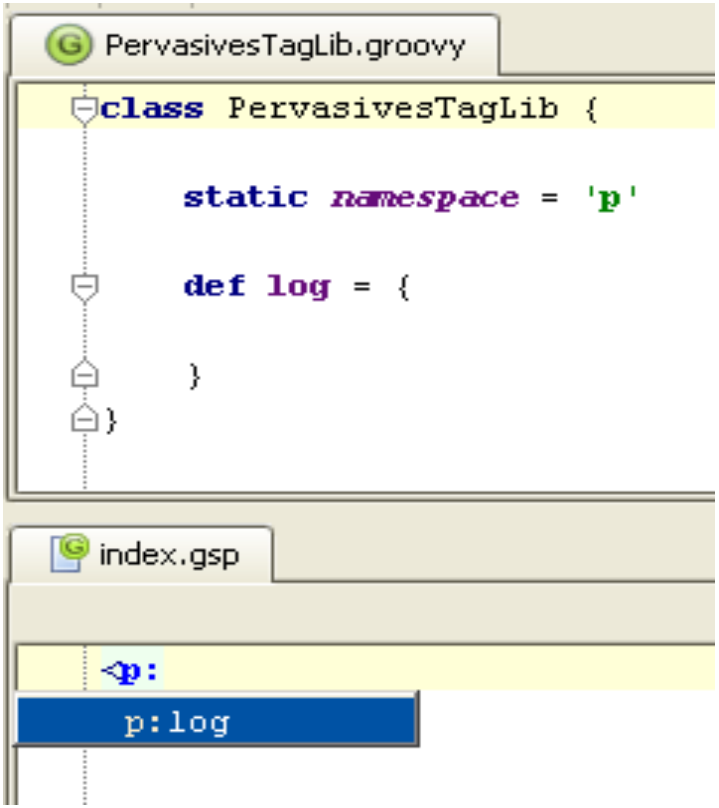
- Visualizes your domain model
- Editable, with changes propagated to Groovy code:
  - Relation insertion
  - Relation deletion
  - Relation switch

Groovy Server Pages



- Formatter
- Groovy support
- HTML code inspection/code assist

Groovy Server Pages (2)



- GSP specific tag completion:
  - Default tags
  - User-defined tags

And We Keep Improving…

- More dynamic stuff in the future
- Extended set of refactorings
- Advanced Groovy-specific code inspections
- "Create from usage" intention actions
- More Grails goodies

How to get the stuff

- Download and install IntelliJ IDEA 7.0
- Install JetGroovy using Plugin Manager
- Download Groovy from http://groovy.codehaus.org
- Download Grails from http://grails.codehaus.org

JetGroovy

Q&A

We Need Your Feedback!

- We are keeping JetGroovy Confluence in sync with the development http://www.jetbrains.net/confluence/display/GRVY/Groovy+Home
- You are welcome to post bugs and feature requests at http://www.jetbrains.net/jira/browse/grvy

- JetGroovy has a dedicated forum/newsgroup at http://www.intellij.net/eap