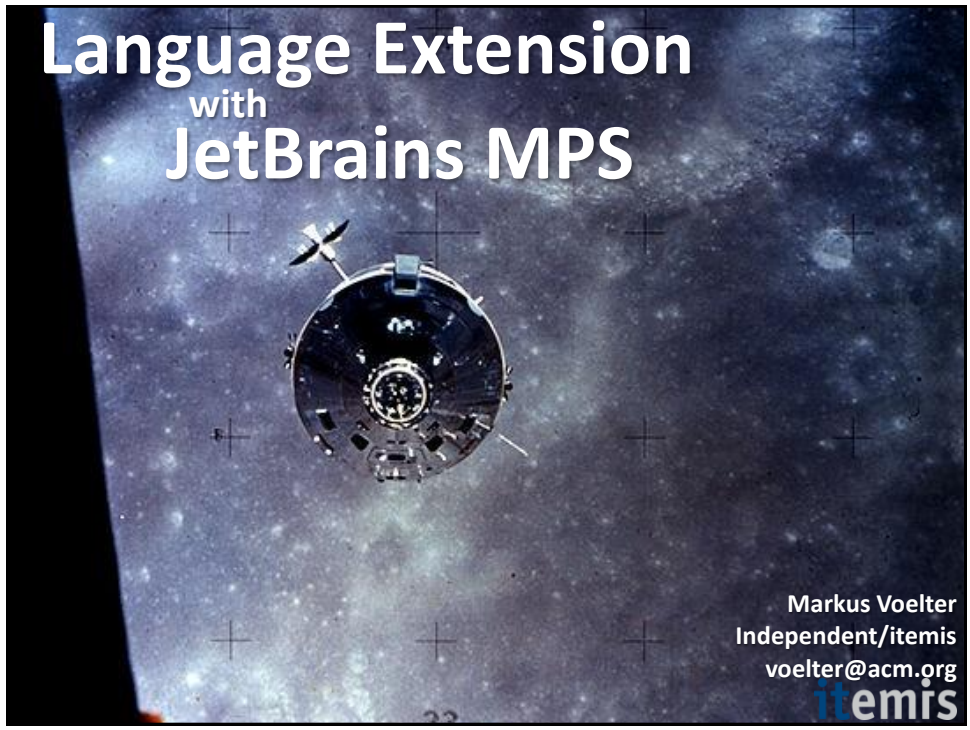
Language Extension with JetBrains MPS



Markus Voelter
Independent/itemis
voelter@acm.org
itemis

also do…

IntelliJ IDEA
Resharper

released in

Q3 2009

currently

1.1

this summer

1.2

Open Source

licensed under

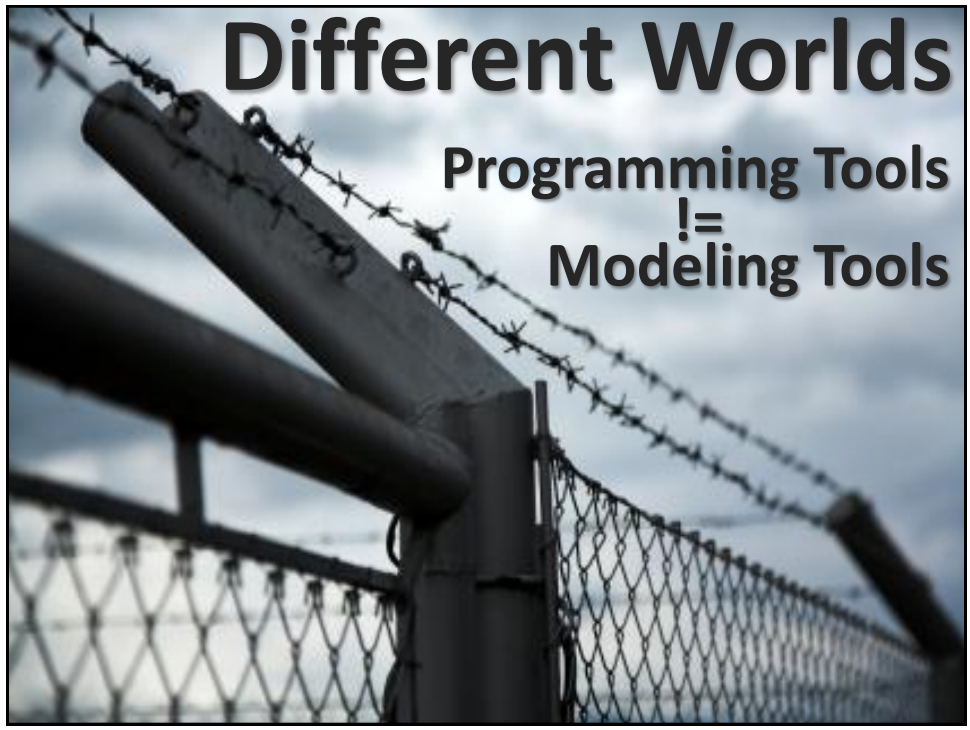
Apache 2.0

programming?

modelling?

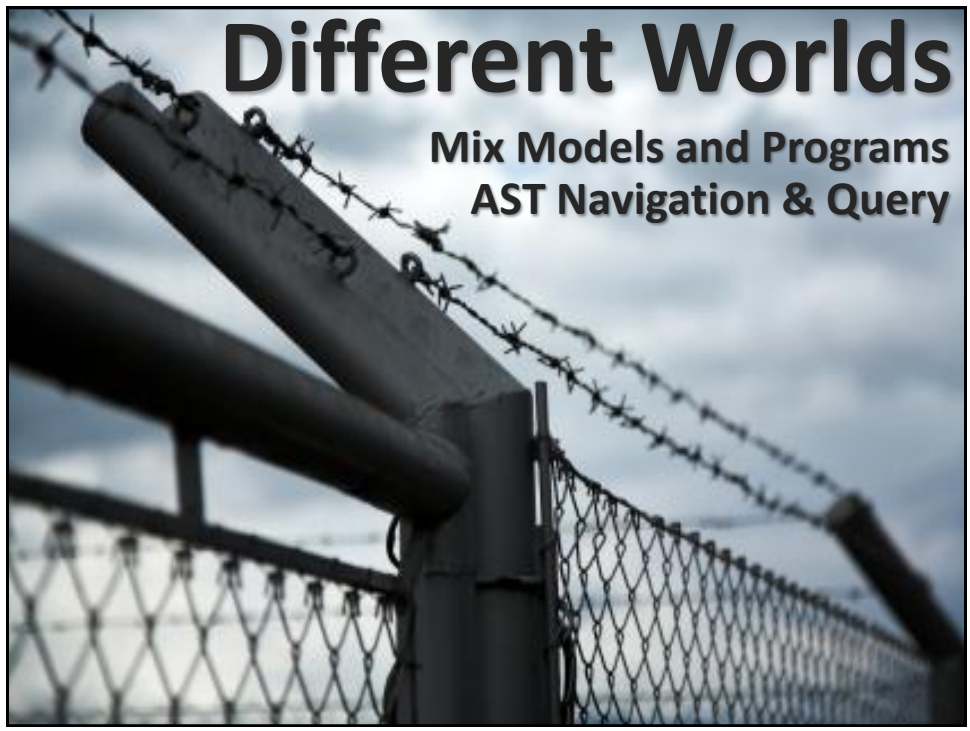
programming?
vs.
modelling?

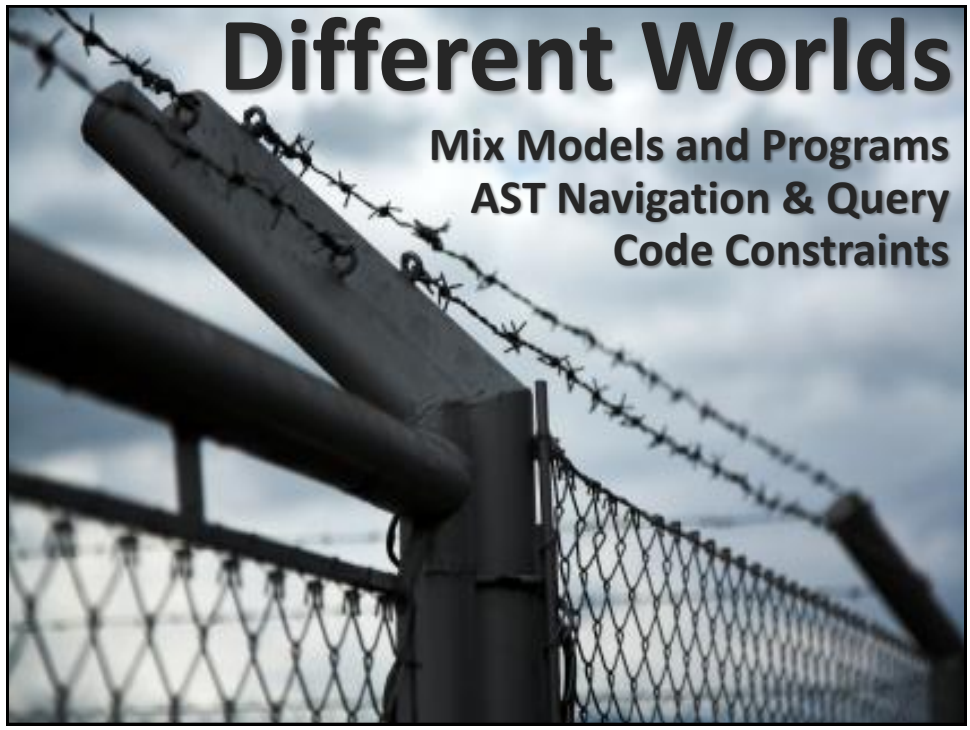
different?
the same?

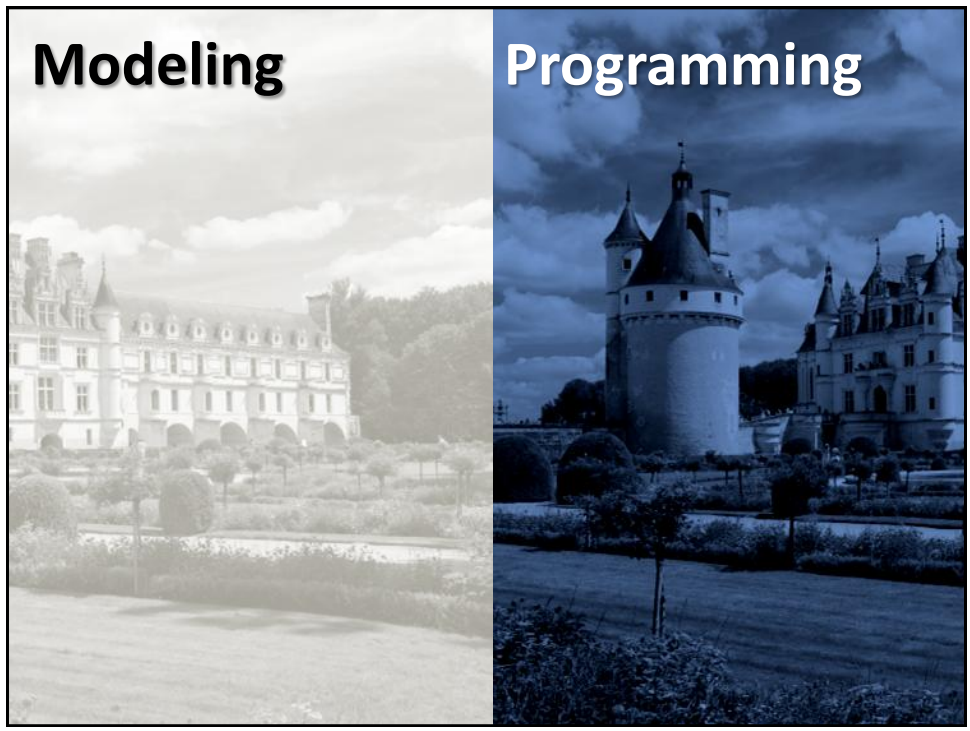

Different Worlds
Programming Tools
!=
Modeling Tools

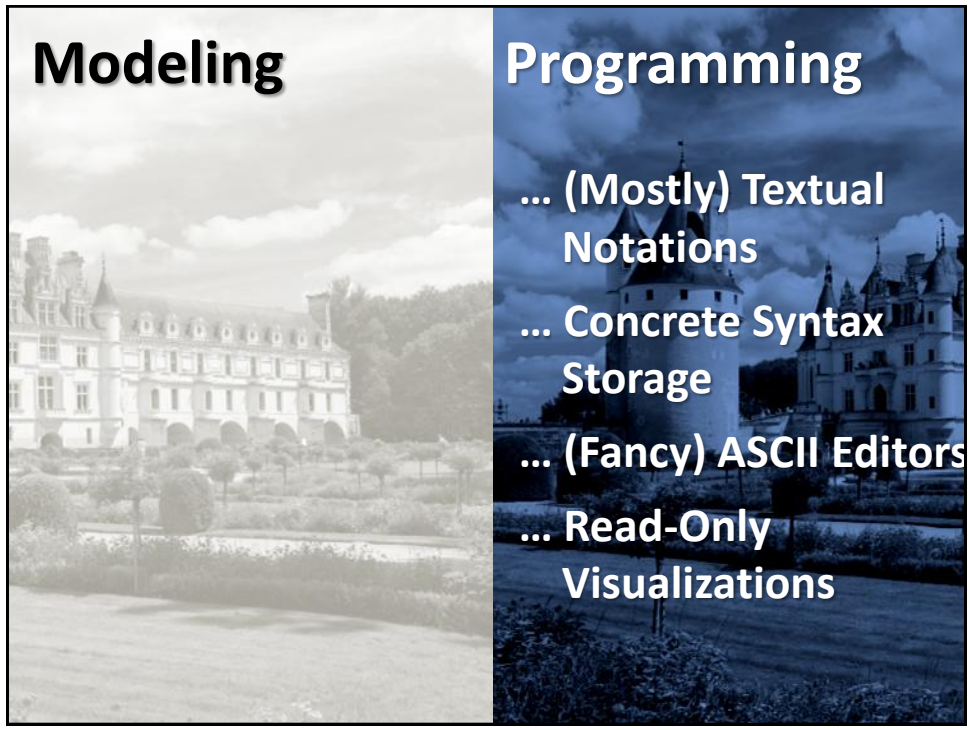
Different Worlds
Modeling Tool
!=
Modeling Tool

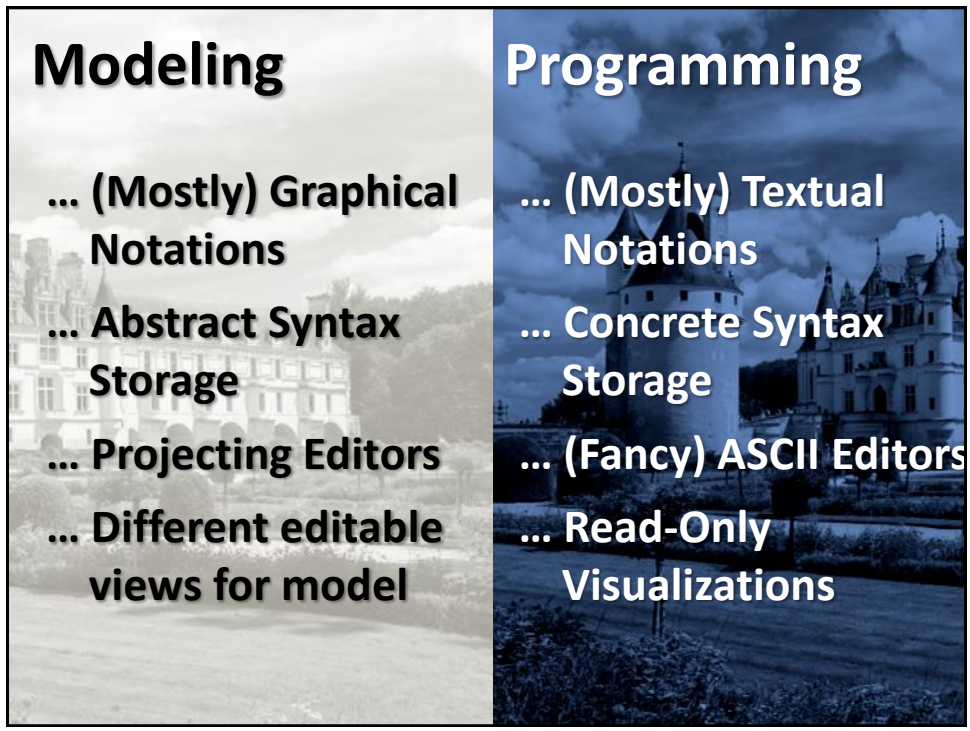
Different Worlds
Mix Models and Programs


Different Worlds
Mix Models and Programs
AST Navigation & Query


Different Worlds
Mix Models and Programs
AST Navigation & Query
Code Constraints


Modeling
Programming

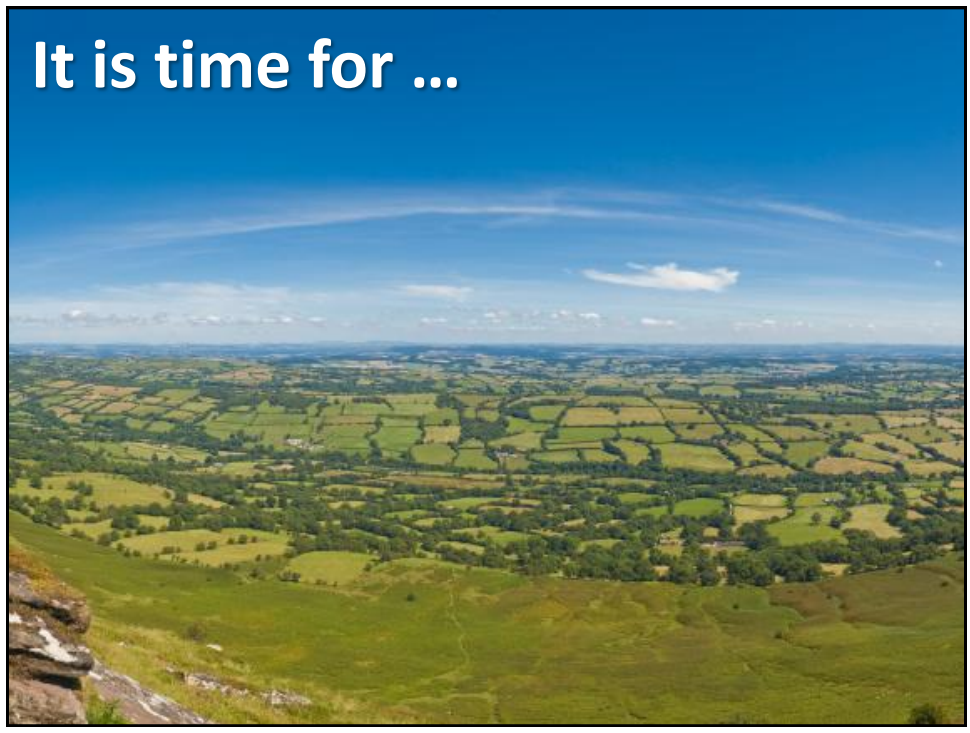

Modeling
Programming
… (Mostly) Textual Notations
… Concrete Syntax Storage
… (Fancy) ASCII Editors
… Read-Only Visualizations


Modeling
… (Mostly) Graphical Notations
… Abstract Syntax Storage
… Projecting Editors
… Different editable views for model
Programming
… (Mostly) Textual Notations
… Concrete Syntax Storage
… (Fancy) ASCII Editors
… Read-Only Visualizations

Why
the difference?

Why
the difference?

It is time for …



… a Different Perspective

We don‘t want to model, we want to program!

We don‘t want to
model,
we want to
program!

… at different levels of **abstaction**
… from different **viewpoints**
… **integrated**!

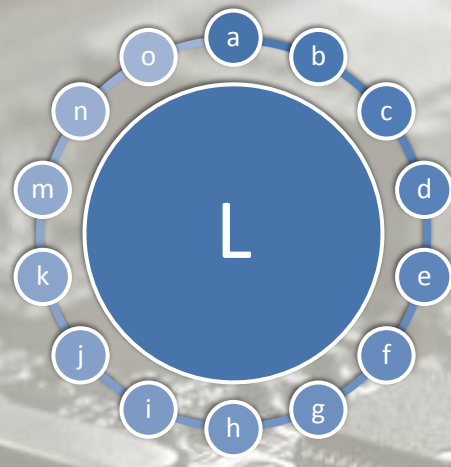
We don‘t want to
model,
we want to
program!

… with different degrees of **domain-specificity**
… with suitable **notations**
… with suitable **expressiveness**

We don‘t want to
model,
we want to
program!

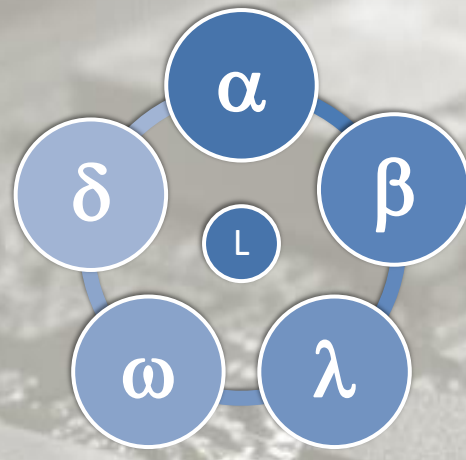
And always:
precise and tool processable

Big Language?



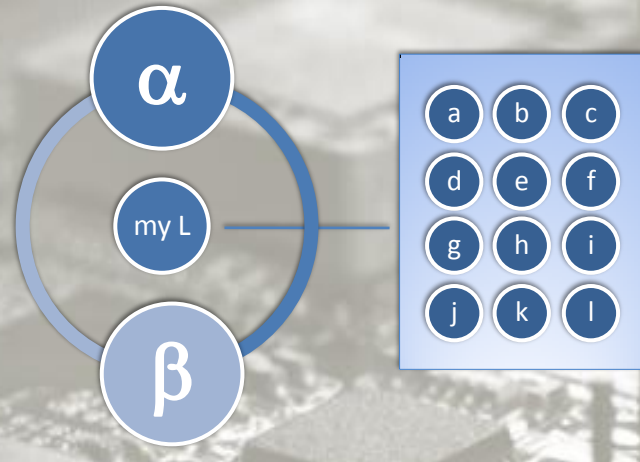
with many first class concepts!

Small Language?



with a few, orthogonal
and poweful concepts

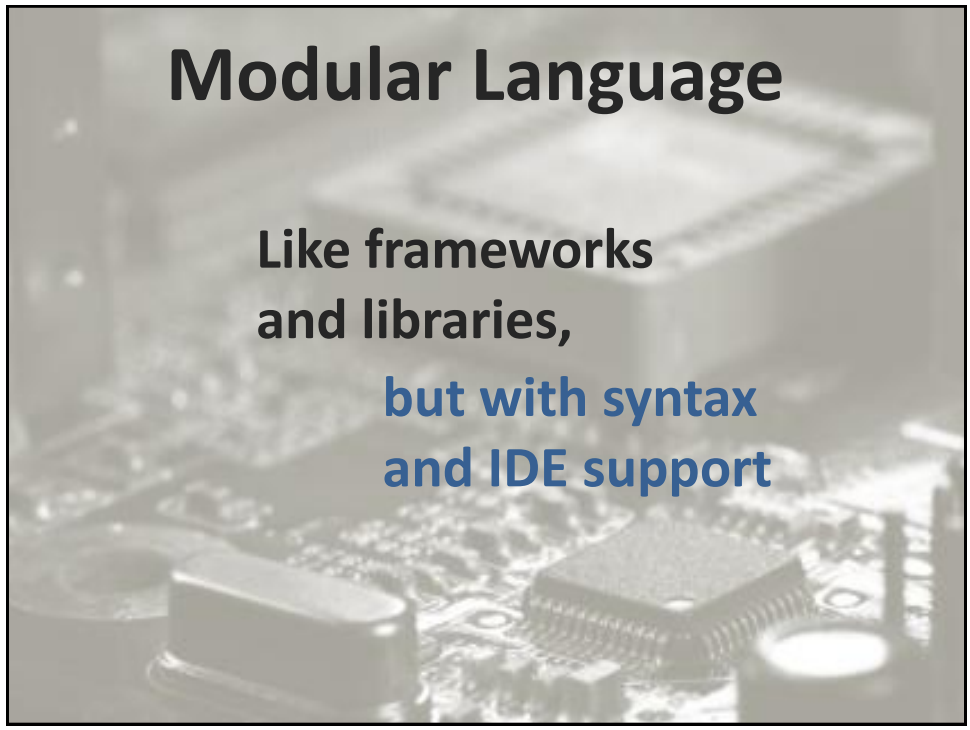
Modular Language



with many optional, composable concepts

Modular Language

Like frameworks and libraries,



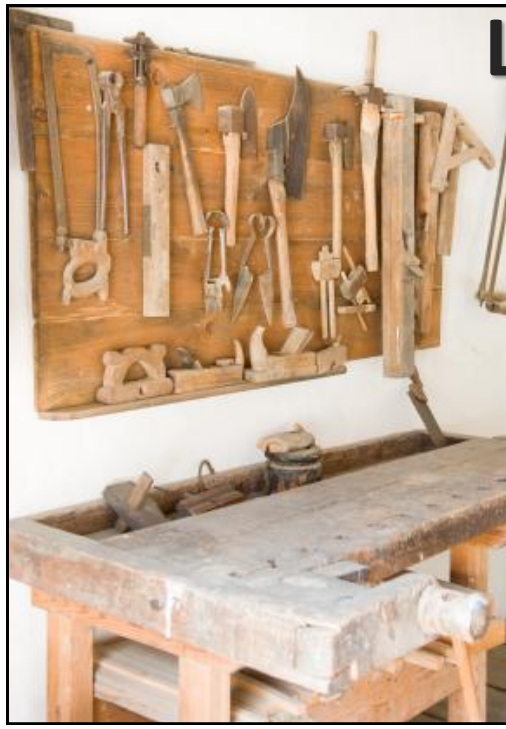
Not a new idea…

Growing A Language
(Guy L Steele)

Growing A Language
(Guy L Steele)
stoneage-OOPSLA talk,
get it on Google Video!

Language Workbench

(Martin Fowler)



Language Workbench

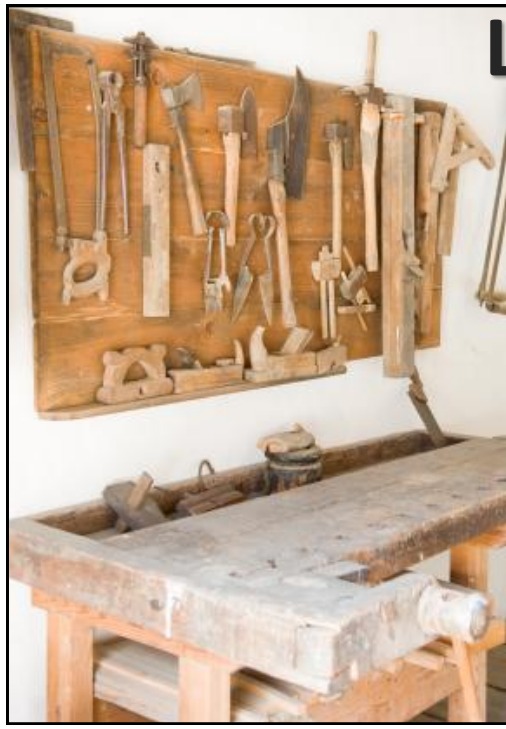
(Martin Fowler)

Freely
define
languages and
integrate
them

Language Workbench

(Martin Fowler)

use
persistent
abstract
representation



Language Workbench

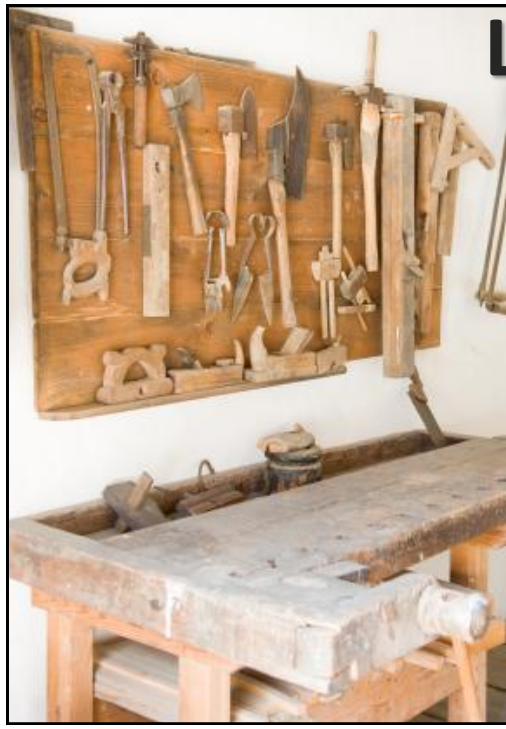
(Martin Fowler)

language ::=
schema
+ editors
+ generators

Language Workbench

(Martin Fowler)

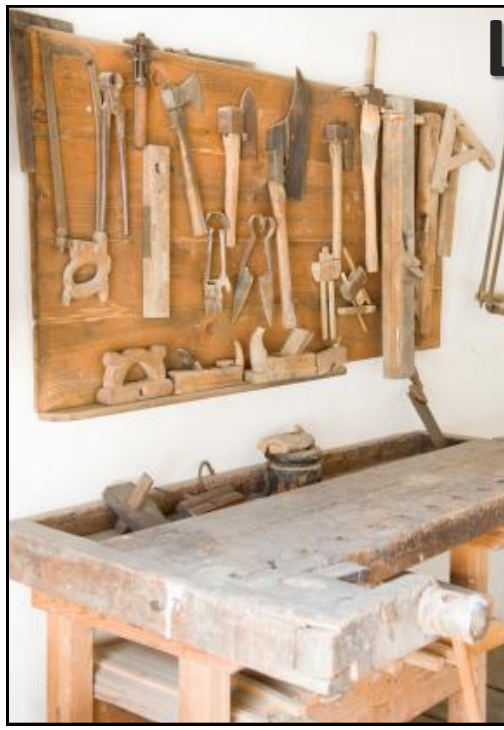
projectional
editing



Language Workbench

(Martin Fowler)

persist
incomplete
or
contradictory
information



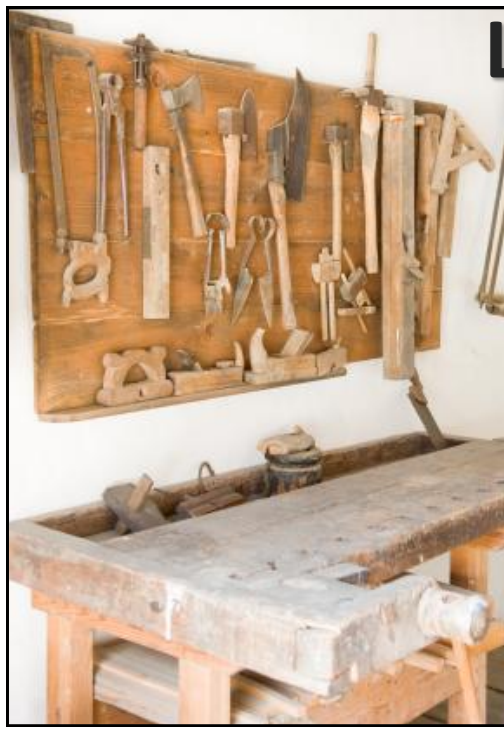
Language Workbench

(Martin Fowler)

powerful
- editing
- testing
- refactoring
- debugging
- groupware

+

language definition
implies
IDE definition



Language Workbench

(Martin Fowler)

support for
- „classical“ programming
- „classical“ and modeling

+

Not a new idea…

Not a new idea…

But

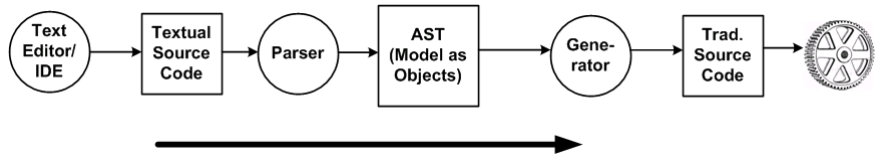
a practical reality :-)

Parser-based

text

… to tree

… to text



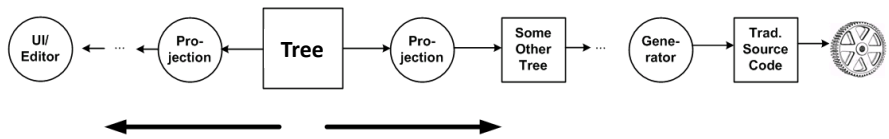
Projectional

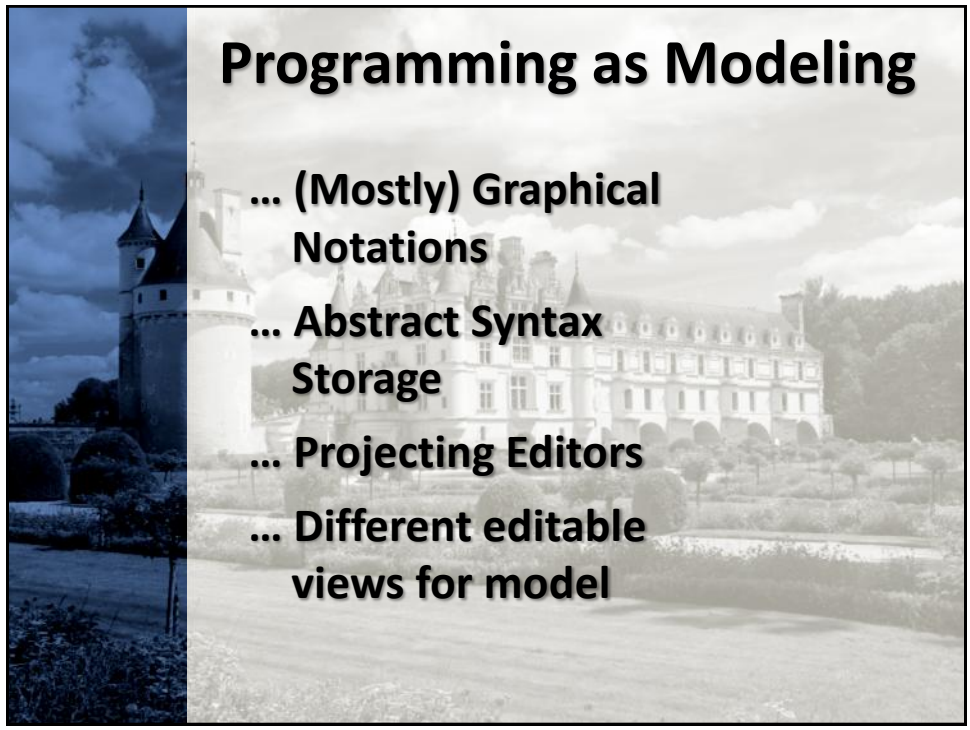
tree

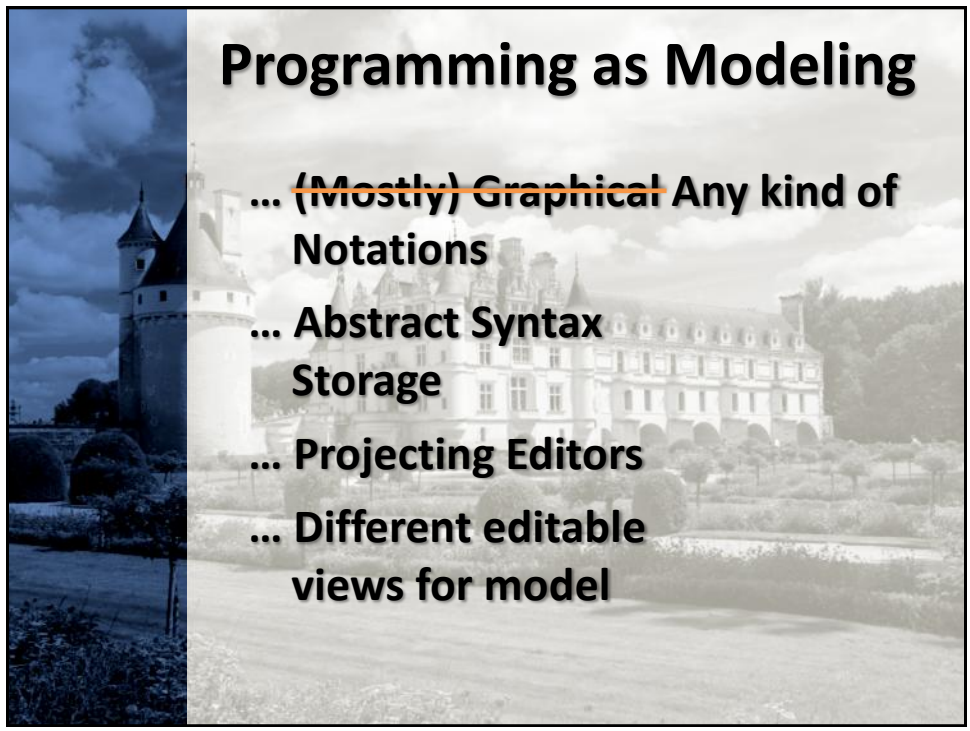
… to text-lookalike (editor)

… to other trees … [*]

… to text




Programming as Modeling
… (Mostly) Graphical Notations
… Abstract Syntax Storage
… Projecting Editors
… Different editable views for model


Programming as Modeling
… (Mostly) Graphical Any kind of Notations
… Abstract Syntax Storage
… Projecting Editors
… Different editable views for model

Language Composition

There‘s no parsing.

Unique Language Element Identity.

Unlimted language composition.

Flexible Notations

Textual
like ASCII

Graphical
box & line

Semi-Graphical
mathematical

} treated the same
can be mixed

Automatic IDE Extension

tool support is inherent
for languages build with
projectional tools

language definition
implies
IDE definition

Multiple Notations

… for the same concepts

e.g. in different contexts
or for different tasks

Partial
Projections

… different views
… for different roles/people
… only a particular variant

Storage
!= Schema

… store arbitraty meta data
- change log
- conflicting information
- variability annotations

… independent of language schema!
… „aspects“, overlay

Live Programs

think: spreadsheet

a change to one part of program can lead to (dependent) changes in other parts

Tree Editing

… is different from editing text

… try to make it feel like text

… takes some getting used to

but: for more flexible notations a more general editing paradigm is needed

Infrastructure
Integration

… storage is not text

… diff/merge must be in tool

… existing text tools don‘t work

Proprietary
Tools

… no standards

… no interop

Build new standalone DSLs

Build new standalone DSLs
Build DSLs that reuse parts of other languages

Build new standalone DSLs
Build DSLs that reuse parts of other languages

(MPS comes with BaseLanguage) Java++

extend base language

Build new standalone DSLs

Build DSLs that reuse parts of other languages

(MPS comes with BaseLanguage (Java++))

extend base language

build DSLs that reuse parts of BaseLanguage

Language Extension Example

Language Extension Example

Old

```
ReadWriteLock l = …
l.readLock().lock();
try {
  //code
} finally {
  l.readLock().unlock();
}
```

Language Extension Example

Old

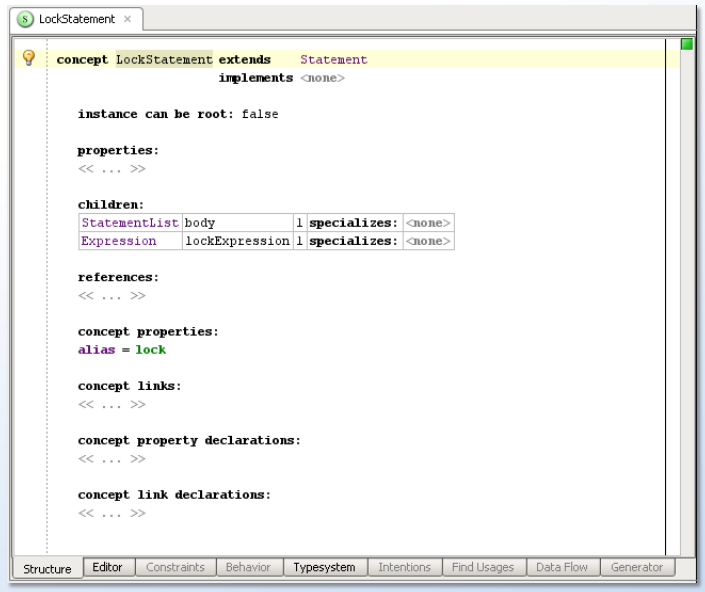
```
ReadWriteLock l = …
l.readLock().lock();
try {
  //code
} finally {
  l.readLock().unlock();
}
```

New

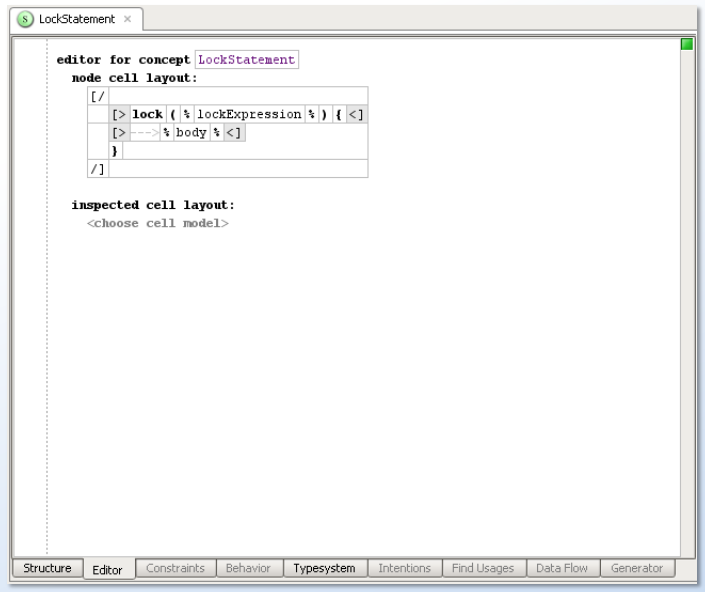
```
ReadWriteLock l = …
lock (l) {
  //code
}
```

Structure ◆ Editor ◆ Typesystem ◆ Generator

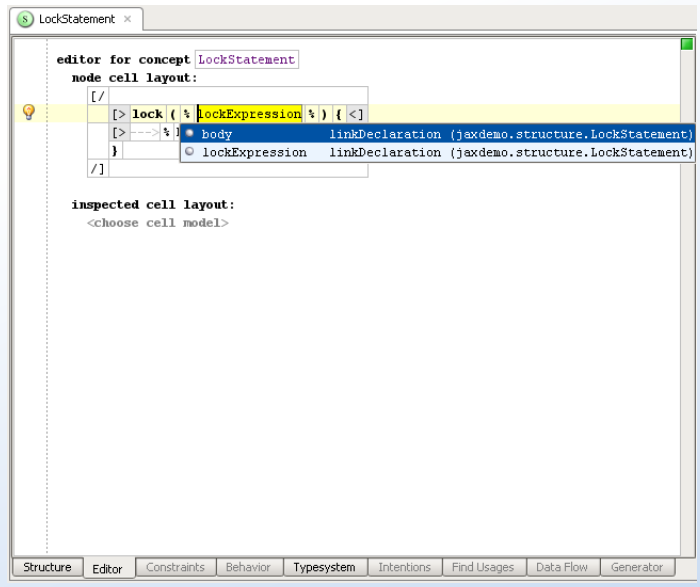
Structure ◆ Editor ◆ Typesystem ◆ Generator



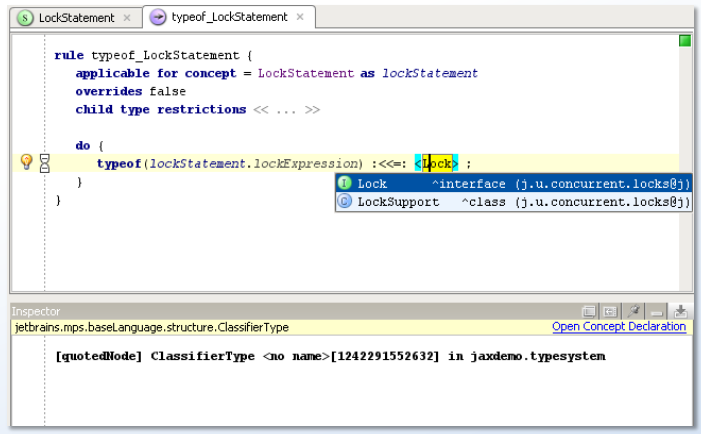
Structure ◆ Editor ◆ Typesystem ◆ Generator



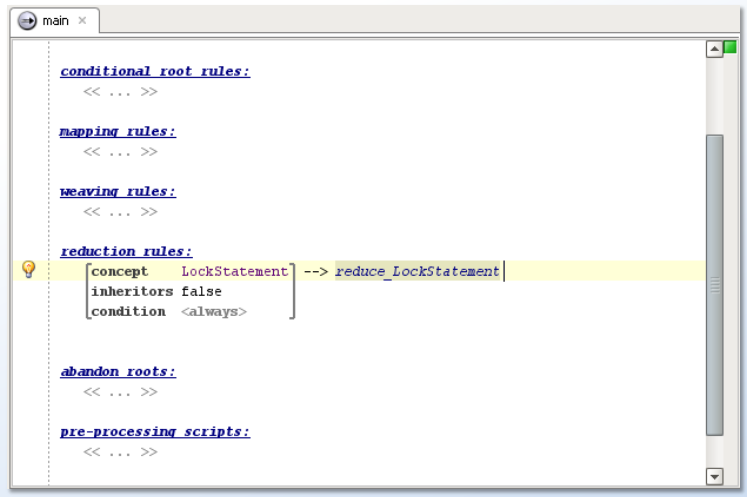
Structure ◆ Editor ◆ Typesystem ◆ Generator



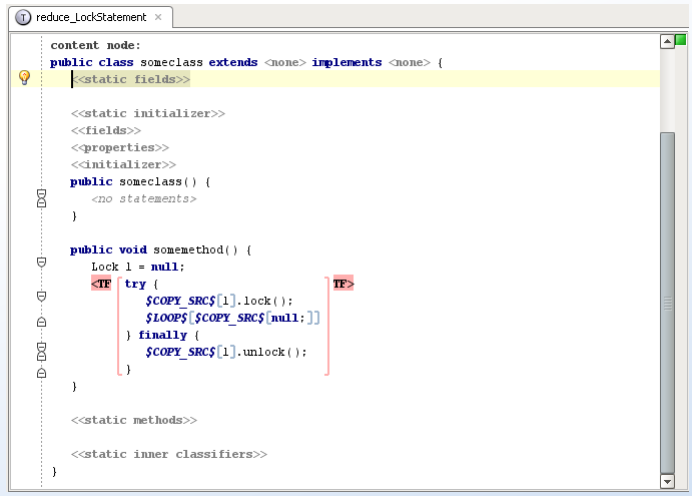
Structure ◆ Editor ◆ Typesystem ◆ Generator



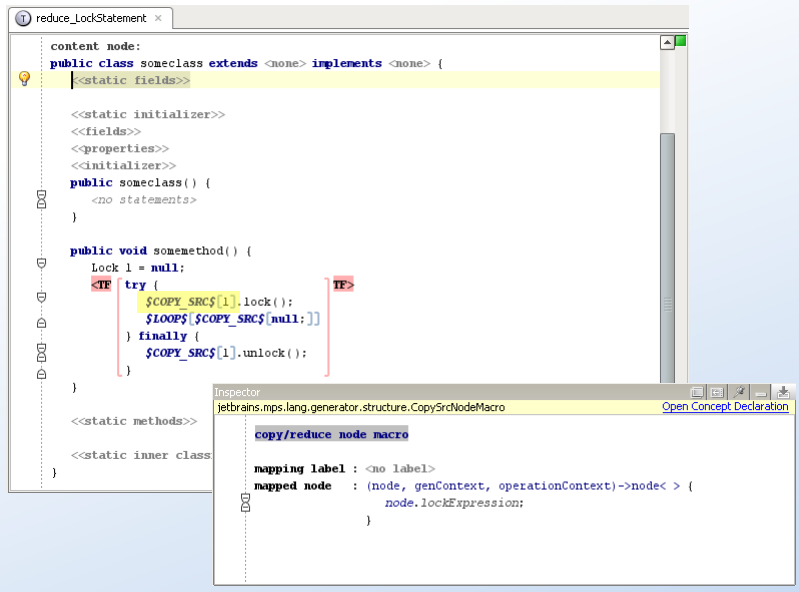
Structure ◆ Editor ◆ Typesystem ◆ Generator



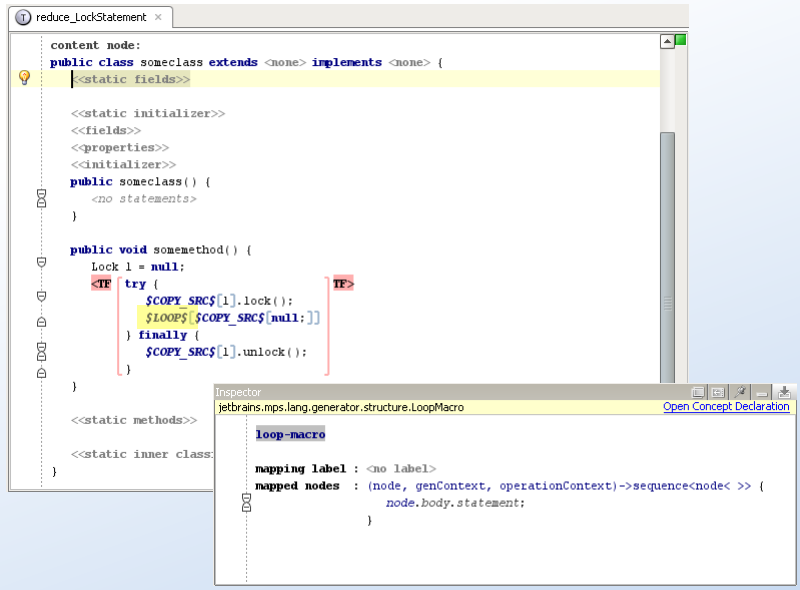
Structure ◆ Editor ◆ Typesystem ◆ Generator



Structure ◆ Editor ◆ Typesystem ◆ Generator

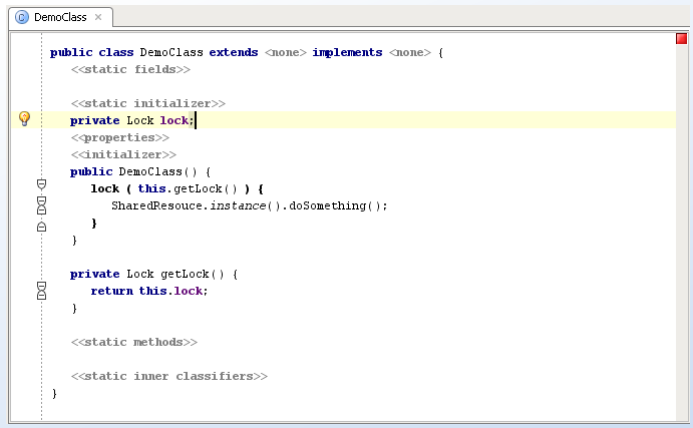


Structure ◆ Editor ◆ Typesystem ◆ Generator



Language Extension Example

Result behaves like a native base language construct



Language Extension Example

Result behaves like a native base language construct

DemoClass

```
public class DemoClass extends <none> implements <none> {
   <<static fields>>

   <<static initializer>>
   private Lock lock;
   <<properties>>
   <<initializer>>
   public DemoClass() {
      lock ( 1 ) {
         SharedResouce.instance().doSomething();
      }
   }

   private Lock getLock() {
      return this.lock;
   }

   <<static methods>>

   <<static inner classifiers>>
}
```

Typesystem error: int is not a subtype of Lock

Language Extension Example

Translated to regular Java code based on the generator

DemoClass

```
public class DemoClass extends <none> implements <none> {
   <<static fields>>

   <<static initializer>>
   private Lock lock;
   <<properties>>
   <<initializer>>
   public DemoClass() {
      lock ( this.getLock() ) {
         SharedResouce.instance().doSome
      }
   }

   private Lock getLock() {
      return this.lock;
   }

   <<static methods>>

   <<static inner classifiers>>
}
```

```
package jaxdemo.sandbox.sandbox;

import java.util.concurrent.locks.Lock;

public class DemoClass {

 private Lock lock;

 public DemoClass() {
  try {
   this.getLock().lock();
   SharedResouce.instance().doSomething();
  } finally {
   this.getLock().unlock();
  }
 }

 private Lock getLock() {  return this.lock;  }

}
```

DEMO

Building DSLs with
JetBrains MPS

The End.
.coordinates
web www.voelter.de
email voelter@acm.org
skype schogglad
xing http://www.xing.com/profile/Markus_Voelter
linkedin http://www.linkedin.com/pub/0/377/a31
itemis

.coordinates

web www.voelter.de
email voelter@acm.org
skype schogglad

xing http://www.xing.com/profile/Markus_Voelter
linkedin http://www.linkedin.com/pub/0/377/a31

itemis