The Hidden Secret of Java Open Source Projects

Eder Ignatowicz
Senior Software Engineer Red Hat

Alex Porcelli
Principal Software Engineer Red Hat

```java
@Entity
@Table(name = "stock_daily_record", catalog = “ederign",
uniqueConstraints = @UniqueConstraint(columnNames = "DATE"))
public class StockDailyRecord implements java.io.Serializable {

    @Id
    @GeneratedValue(strategy = IDENTITY)
    @Column(name = "RECORD_ID", unique = true, nullable = false)
    public Integer getRecordId() {
        return this.recordId;
    }

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "STOCK_ID", nullable = false)
    public Stock getStock() {
        return this.stock;
    }

    @Column(name = "PRICE_CHANGE", precision = 6)
    public Float getPriceChange() {
        return this.priceChange;
    }


}
```

```java
@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
public @interface Entity {
    String name() default "";
}
```

Annotations

Metadata

Tooling

Code Generation

Reduce Boilerplate

Annotations

Runtime

Compile time

```java
@Target({ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
public @interface Entity {
    String name() default "";
}
```

JDora UnitTest Framework

DEMO GODS
PLEASE LET THESE DEMOS WORK
memegenerator.net

Runtime

Compile Time

Annotation Processing

Annotation Processing

Part of Compilation Process

Annotated Sources Scanned

Code Generation

Enhance Compilation Feedback

Reduce boilerplate <3

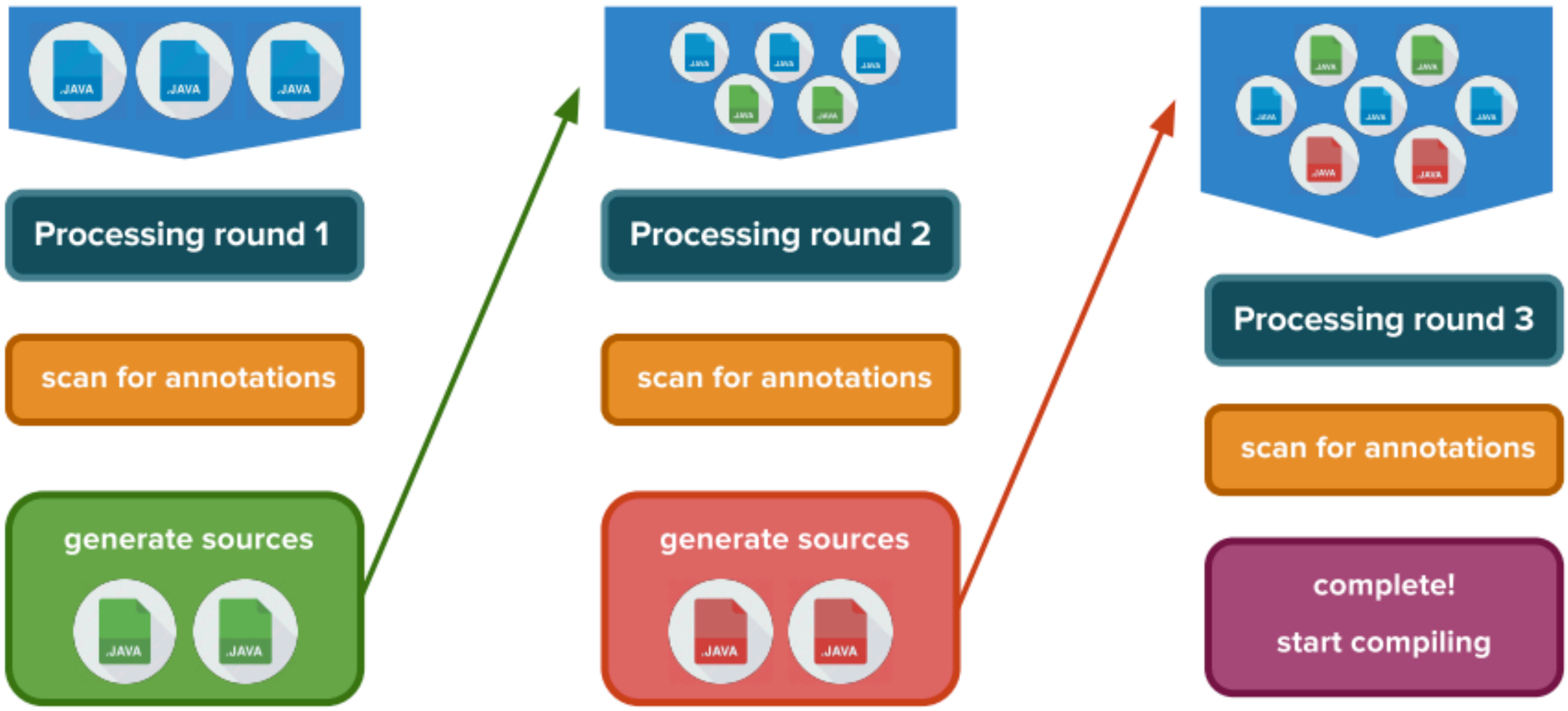
Annotation Processing

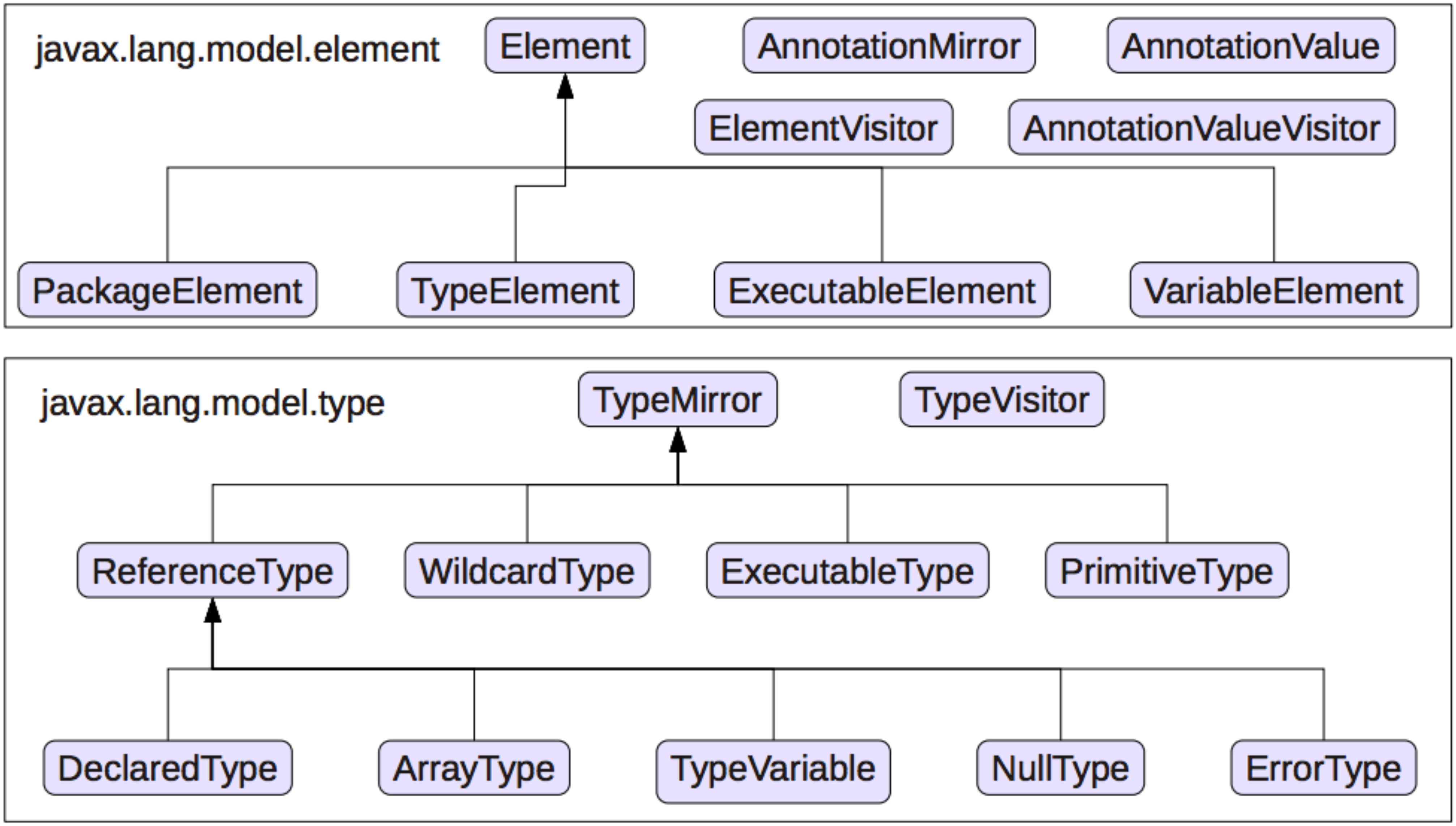
Don’ts:

Inject code

Change existing sources

Bytecode manipulation


.JAVA
.JAVA
.JAVA
Processing round 1
scan for annotations
generate sources
.JAVA
.JAVA
Processing round 2
scan for annotations
generate sources
.JAVA
.JAVA
Processing round 3
scan for annotations
complete!
start compiling


javax.lang.model.element
Element
AnnotationMirror
AnnotationValue
ElementVisitor
AnnotationValueVisitor
PackageElement
TypeElement
ExecutableElement
VariableElement
javax.lang.model.type
TypeMirror
TypeVisitor
ReferenceType
WildcardType
ExecutableType
PrimitiveType
DeclaredType
ArrayType
TypeVariable
NullType
ErrorType

```
package com.example;    // PackageElement

public class Foo {          // TypeElement

    private int a;          // VariableElement
    private Foo other;      // VariableElement

    public Foo () {}        // ExecutableElement

    public void setA ()     // ExecutableElement

}
```

```java
//@SupportedAnnotationTypes(
        {"javax.persistence.Entity", "javax.persistence.OneToMany"})
@SupportedSourceVersion(SourceVersion.RELEASE_7)
public class JpaProcessor extends AbstractProcessor {

    @Override
    public void init(ProcessingEnvironment env) {
        super.init(env);
    }

    @Override
    public Set<String> getSupportedAnnotationTypes() {
        return new HashSet<String>() {{
            add(PrintMe.class.getCanonicalName());
        }};
    }

    @Override
    public boolean process(
            Set<? extends TypeElement> annotations,
            RoundEnvironment roundEnv) {
        //seu processamento
        return false; // continua processando?
    }
}
```

Real world...

KIE Group

Open source projects for business systems automation and management.

Drools

Drools 5 introduces the **Business Logic integration Platform** which provides a unified and integrated platform for **Rules, Workflow** and **Event Processing**. It's been designed from the ground up so that each aspect is a first class citizen, with no compromises.

Drools team

jBPM

jBPM is a flexible Business Process Management (BPM) Suite. A business process allows you to model your business goals by describing the steps that need to be executed to achieve those goals, and the order of those goals are depicted using a flow chart...

jBPM team

OptaPlanner

OptaPlanner optimizes business resource usage. Every organization faces planning problems: provide products or services with a limited set of constrained resources. OptaPlanner optimizes such planning to do more business with less resources...

Optaplanner team

UberFire

UberFire is a web based workbench framework inspired by Eclipse Rich Client Platform. This is a very strategic project for Drools & jBPM team, once this is the base technology for our next generation of web tooling.

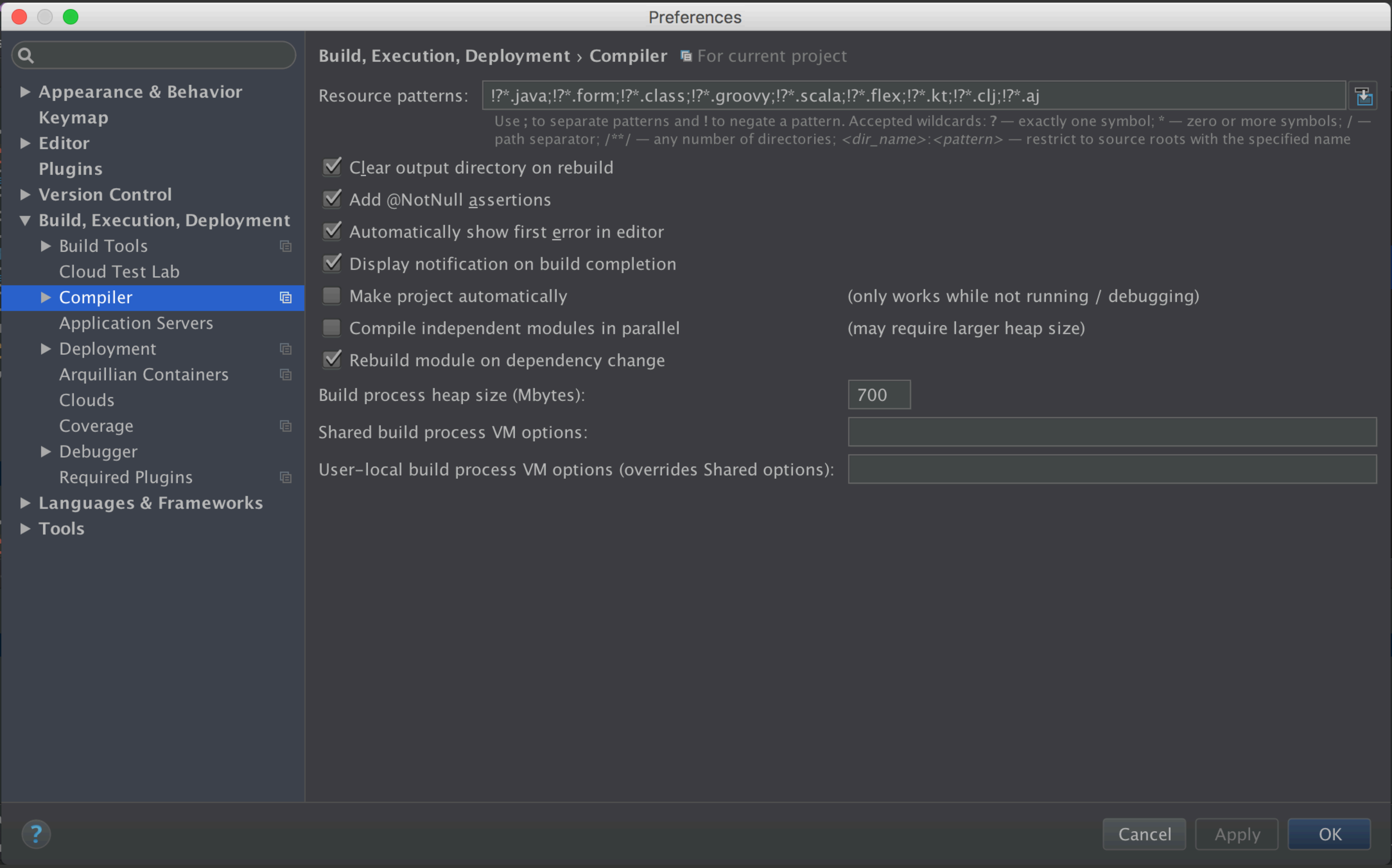
Dashbuilder

Dashbuilder is a full featured web application for the visual composition of custom business dashboards. Data comes from heterogeneous sources of information such as JDBC databases or regular text files and can be displayed using different charting libraries.

Important Architecture Tool

AppFormer

Preferences Framework


Preferences

- Appearance & Behavior
- Keymap
- Editor
- Plugins
- Version Control
- Build, Execution, Deployment
  - Build Tools
  - Cloud Test Lab
  - Compiler
  - Application Servers
  - Deployment
  - Arquillian Containers
  - Clouds
  - Coverage
  - Debugger
  - Required Plugins
- Languages & Frameworks
- Tools

Build, Execution, Deployment › Compiler For current project

Resource patterns: !?*.java;!?*.form;!?*.class;!?*.groovy;!?*.scala;!?*.flex;!?*.kt;!?*.clj;!?*.aj

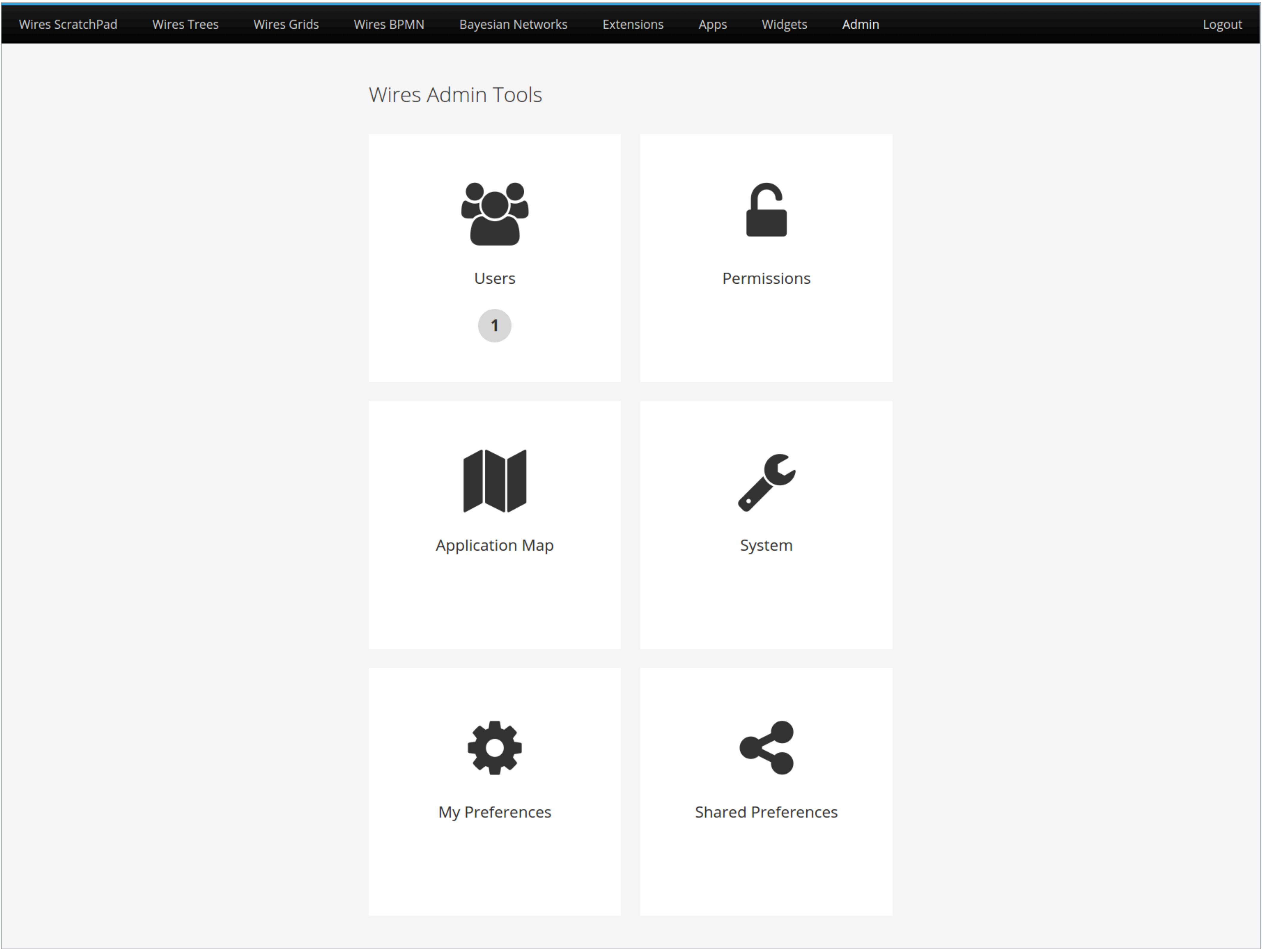
Use ; to separate patterns and ! to negate a pattern. Accepted wildcards: ? — exactly one symbol; * — zero or more symbols; / — path separator; /**/ — any number of directories; *<dir_name>:<pattern>* — restrict to source roots with the specified name

- [✓] Clear output directory on rebuild
- [✓] Add @NotNull assertions
- [✓] Automatically show first error in editor
- [✓] Display notification on build completion
- [ ] Make project automatically (only works while not running / debugging)
- [ ] Compile independent modules in parallel (may require larger heap size)
- [✓] Rebuild module on dependency change

Build process heap size (Mbytes): 700

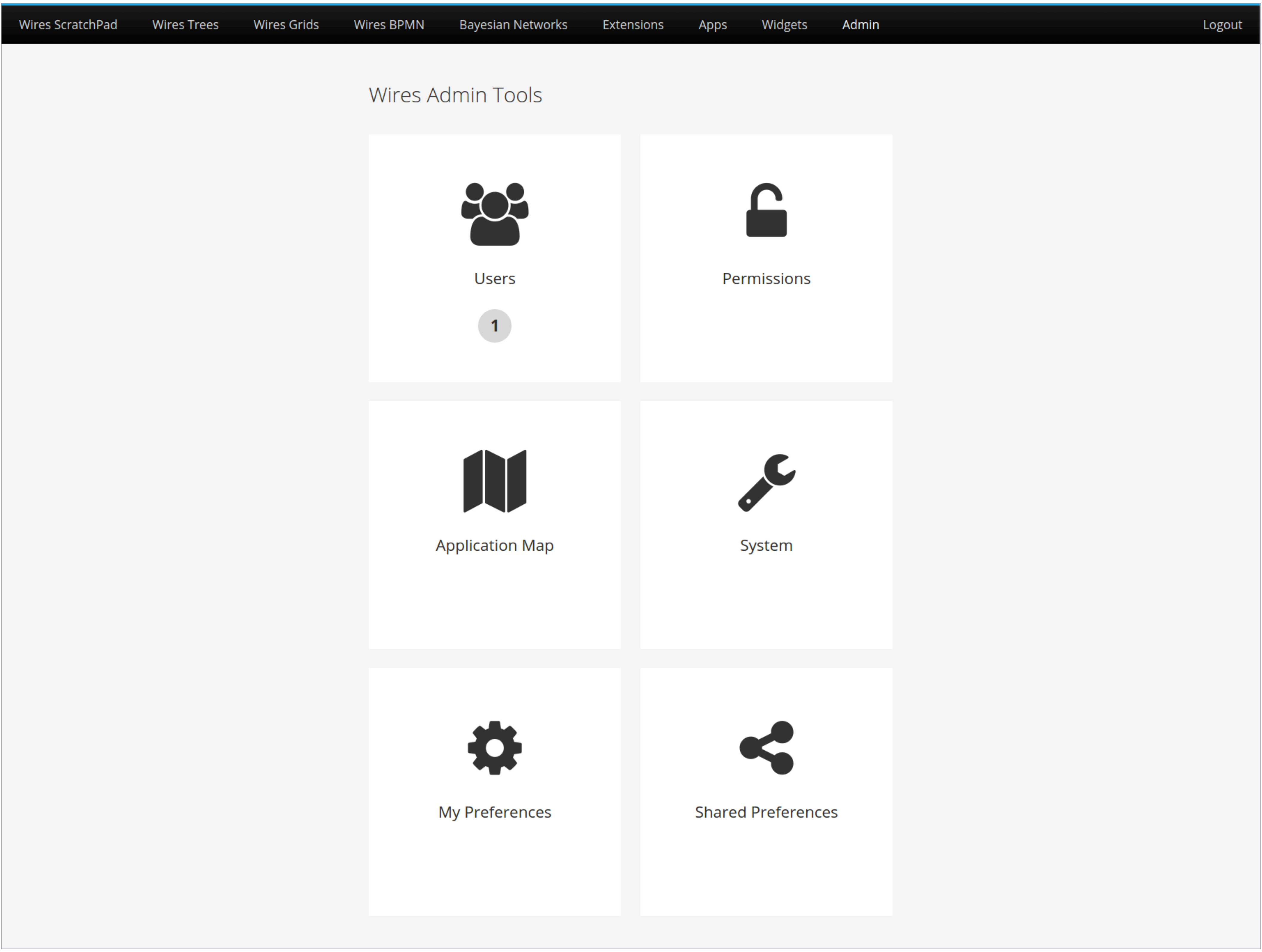
Shared build process VM options:

User-local build process VM options (overrides Shared options):

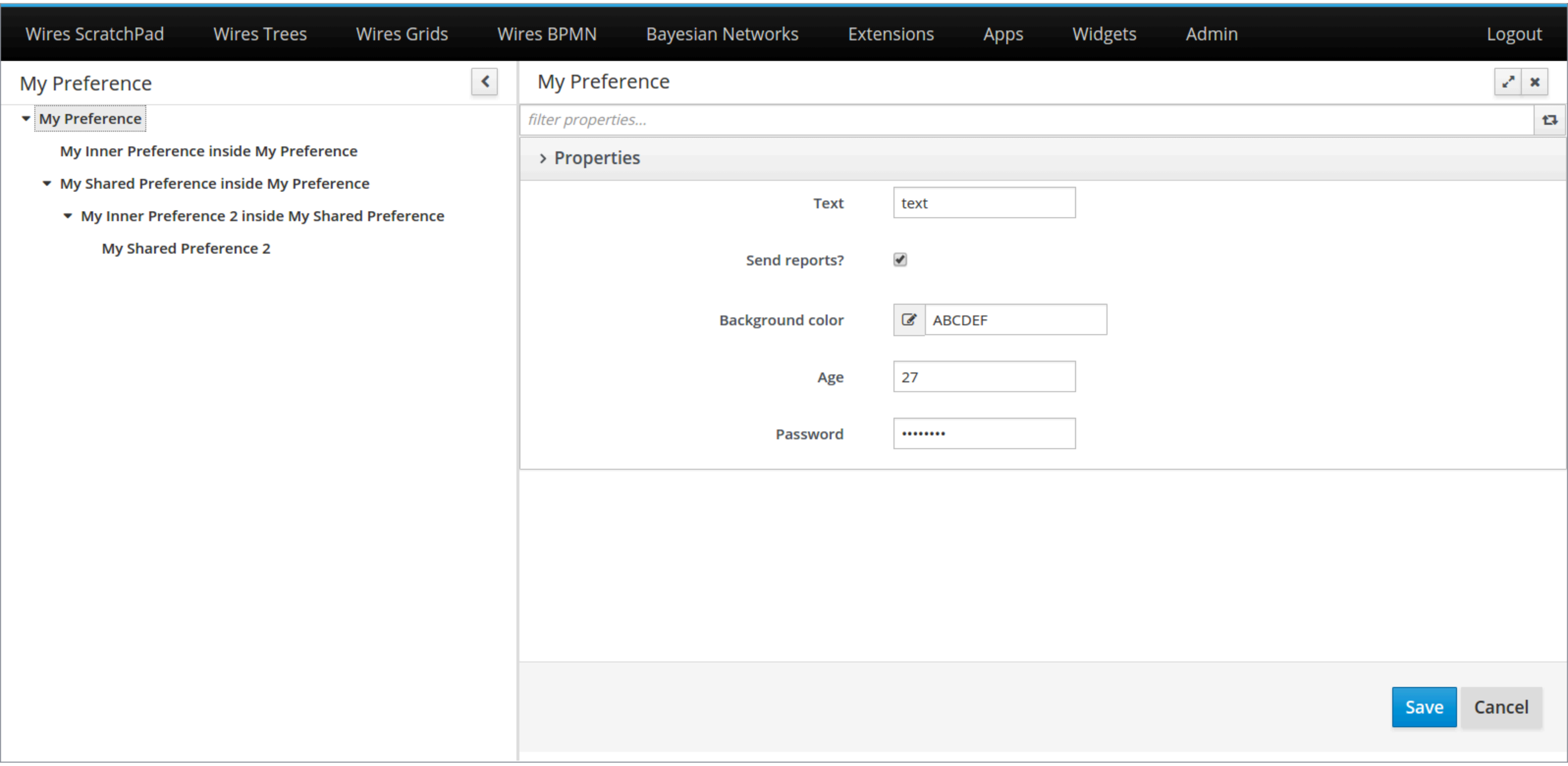

Wires ScratchPad
Wires Trees
Wires Grids
Wires BPMN
Bayesian Networks
Extensions
Apps
Widgets
Admin
Logout
Wires Admin Tools
Users
1
Permissions
Application Map
System
My Preferences
Shared Preferences

```java
@WorkbenchPreference(identifier = "MyPreference",
        bundleKey = "MyPreference.Label")
public class MyPreference implements BasePreference<MyPreference> {

    @Property(bundleKey = "MyPreference.Text",
            helpBundleKey = "MyPreference.Text.Help",
            validators = NotEmptyValidator.class,
            formOptions = PropertyFormOptions.DISABLED)
    String text;

    @Property(formType = PropertyFormType.BOOLEAN, bundleKey = "MyPreference.SendReports")
    boolean sendReports;

    @Property(formType = PropertyFormType.COLOR, bundleKey = "MyPreference.BackgroundColor")
    String backgroundColor;

    @Property(formType = PropertyFormType.NATURAL_NUMBER, bundleKey = "MyPreference.Age")
    int age;

    @Property(formType = PropertyFormType.SECRET_TEXT, bundleKey = "MyPreference.Password")
    String password;

    @Property(bundleKey = "MyPreference.MyInnerPreference")
    MyInnerPreference myInnerPreference;

    @Property(shared = true, bundleKey = "MyPreference.MySharedPreference")
    MySharedPreference mySharedPreference;

    @Override
    public MyPreference defaultValue(final MyPreference defaultValue) {
        defaultValue.text = "text";
        defaultValue.sendReports = true;
        defaultValue.backgroundColor = "ABCDEF";
        defaultValue.age = 27;
        defaultValue.password = "password";
        defaultValue.myInnerPreference.text = "text";

        return defaultValue;
    }
}
```

```java
public class MyServerBean {

    @Inject
    private MyPreference myPreference;

    public void load() {
        // Loads the preference content from the file system
        myPreference.load();

        myPreference.text = "text";
        myPreference.sendReports = true;
        myPreference.backgroundColor = "ABCDEF";
        myPreference.age = 27;
        myPreference.password = "password";
        myPreference.myInnerPreference.text = "text";
        myPreference.myInheritedPreference.text = "text";
        myPreference.myInheritedPreference.myInnerPreference2.text = "text";
        myPreference.myInheritedPreference.myInnerPreference2.myInheritedPreference2.text = 
"text";

        // Saves the modified preference content.
        myPreference.save();
    }
}
```



Wires Admin Tools



My Preference

- My Preference
  - My Inner Preference inside My Preference
  - My Shared Preference inside My Preference
    - My Inner Preference 2 inside My Shared Preference
      - My Shared Preference 2

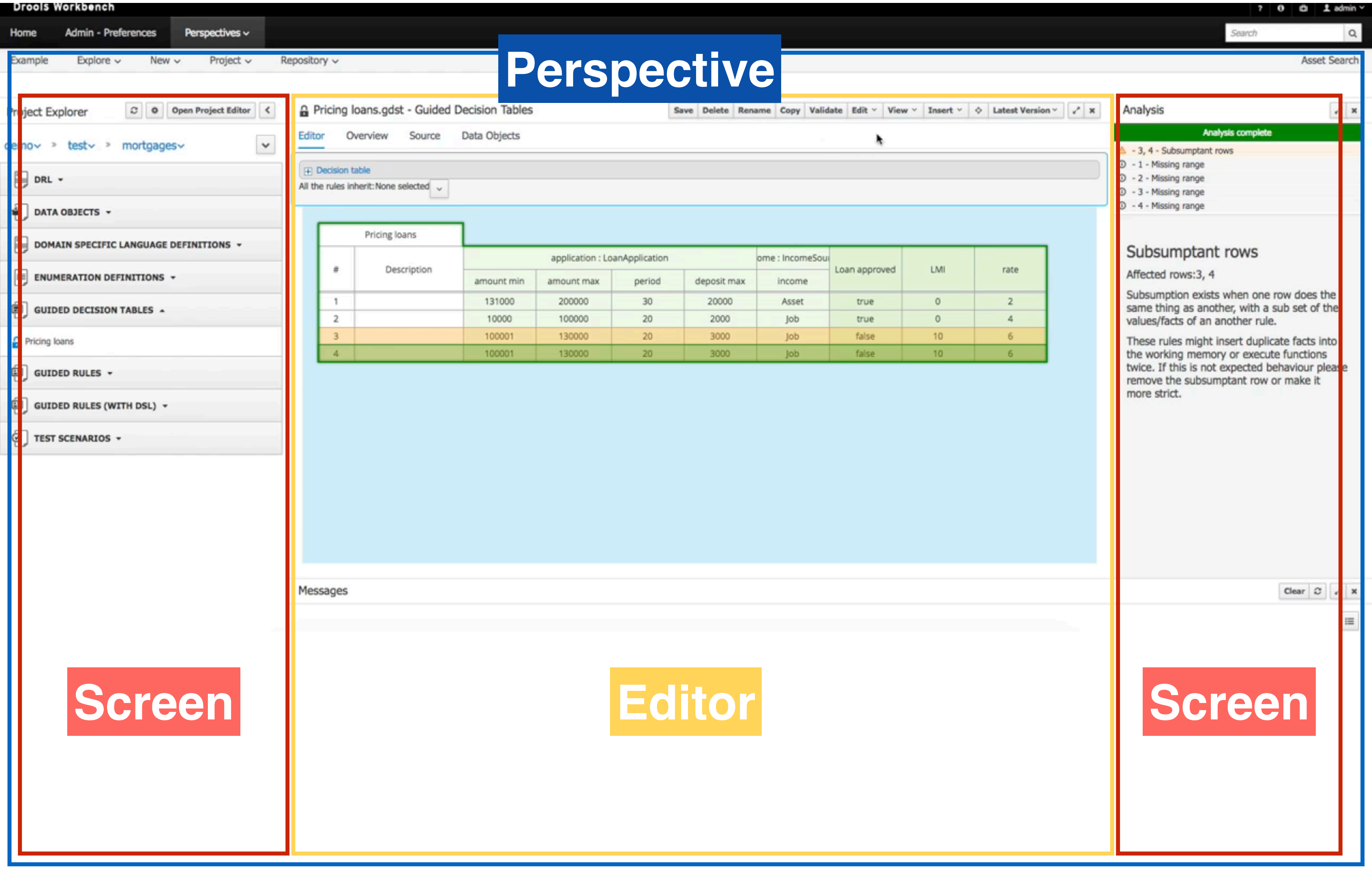
My Preference

filter properties...

Properties

Text	text
Send reports?	✓
Background color	ABCDEF
Age	27
Password	•••••••••

AppFormer UI Components



Pricing loans

#	Description	application : LoanApplication				ome : IncomeSou	Loan approved	LMI	rate
		amount min	amount max	period	deposit max	income			
1		131000	200000	30	20000	Asset	true	0	2
2		10000	100000	20	2000	Job	true	0	4
3		100001	130000	20	3000	Job	false	10	6
4		100001	130000	20	3000	Job	false	10	6

- 3, 4 - Subsumptant rows
- 1 - Missing range
- 2 - Missing range
- 3 - Missing range
- 4 - Missing range

Subsumptant rows

Affected rows:3, 4

Subsumption exists when one row does the same thing as another, with a sub set of the values/facts of an another rule.

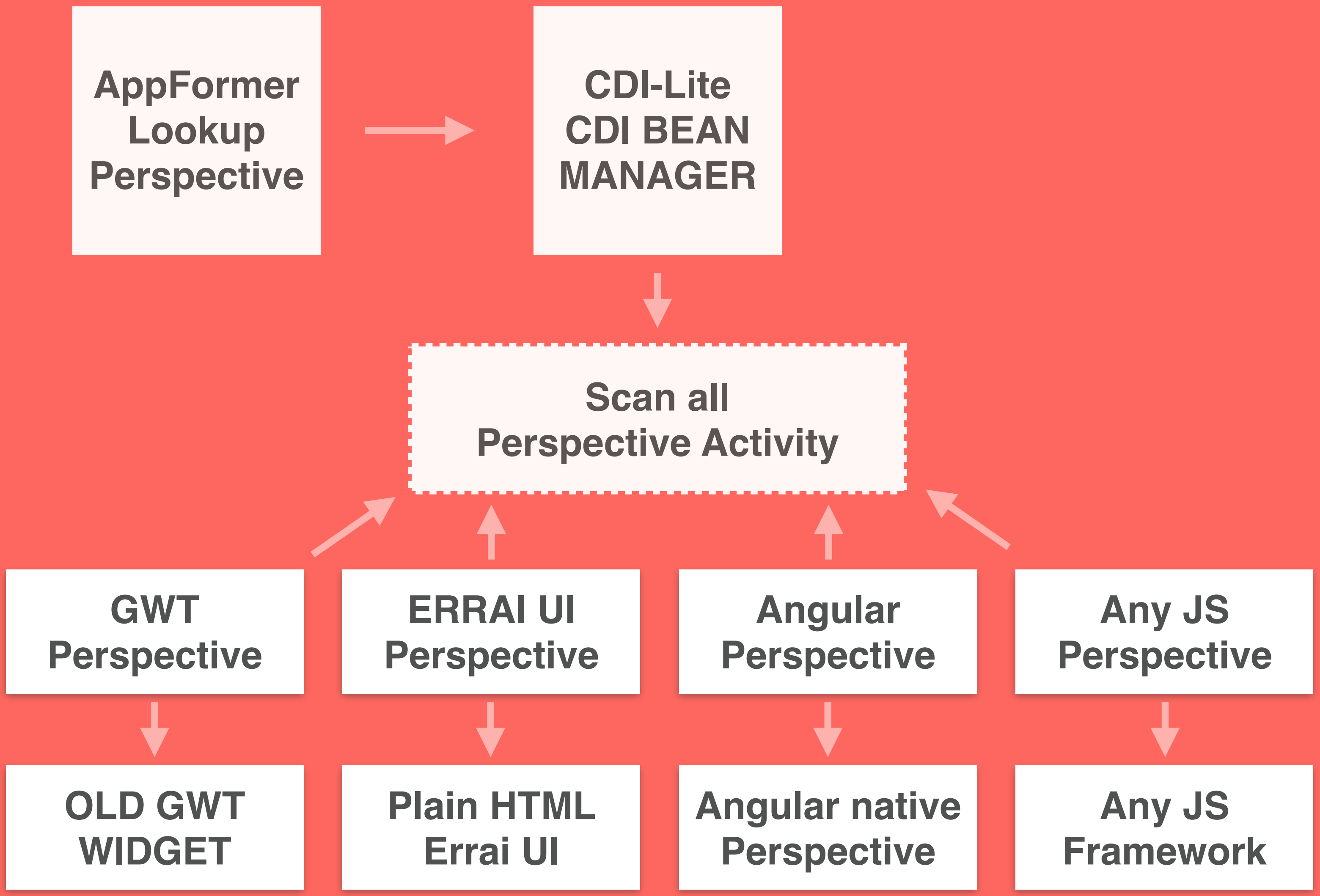
These rules might insert duplicate facts into the working memory or execute functions twice. If this is not expected behaviour please remove the subsumptant row or make it more strict.

Contract Based

Screen -> Interface WorkbenchScreenActivity

Editor -> Interface WorkbenchEditorActivity

Perspective -> Interface PerspectiveActivity


AppFormer Lookup Perspective
CDI-Lite CDI BEAN MANAGER
Scan all Perspective Activity
GWT Perspective
ERRAI UI Perspective
Angular Perspective
Any JS Perspective
OLD GWT WIDGET
Plain HTML Errai UI
Angular native Perspective
Any JS Framework

```java
@WorkbenchPerspective(identifier = “HomePerspective", isDefault
= true)
@Templated
public class HomePerspective implements IsElement {

    @Inject
    @DataField
    @WorkbenchPanel(parts = "MoodScreen?uber=fire&uber1=fire1")
    Div moodScreen;

    @Inject
    @DataField
    @WorkbenchPanel(parts = "HomeScreen?uber=fire")
    Div homeScreen;

    @Inject
    @DataField
    @WorkbenchPanel(parts = "AnotherScreen")
    Div anotherScreen;
}
```

```java
@JsType
public interface PerspectiveActivity{

    PerspectiveDefinition
getDefaultPerspectiveLayout();

    @Override
    default String getName() {
        return
getDefaultPerspectiveLayout().getName();
    }

    boolean isDefault();

    Menus getMenus();

    ToolBar getToolBar();
}
```

```java
@Dependent
@Generated("org.uberfire.annotations.processors.WorkbenchScreenProcessor")
@Named("HomeScreen")
public class HomeScreenActivity extends AbstractWorkbenchScreenActivity {

    @Inject
    private HomeScreen realPresenter;

    @Inject
    //Constructor injection for testing
    public HomeScreenActivity(final PlaceManager placeManager) {
        super( placeManager );
    }

    @Override
    public void onStartup(final PlaceRequest place) {
        super.onStartup( place );
        realPresenter.onStartup();
    }

    @Override
    public void onClose() {
        super.onClose();
        realPresenter.onClose();
    }

    @Override
    public void onShutdown() {
        super.onShutdown();
        realPresenter.onShutdown();
```

Annotations

Reflection

Annotation Processors

+

Open Source

Thank you! <3

Eder Ignatowicz
@ederign

Alex Porcelli
@porcelli