Spray User Guide

Spray – An easy way to create Graphiti

Authors: Jos Warmer, Karsten Thoms, Joerg Reichert

Contents

Introduction

The Graphiti framework is a new approach to create highly sophisticated visual editors on top of the GEF framework. Graphiti can easily be integrated with EMF as the domain modeling framework. The creation of visual editors is done in Java, programming against the Graphiti framework API. It is fairly simple, but yet repetitive, which makes it a candidate to be supported by the means of model-driven development.

This project aims to provide one or more Domain Specific Languages (DSL) (for example with Xtext) to describe Visual DSL Editors against the Graphiti runtime, and provide code generation (for example with Xtend2) to create the boilerplate code for realizing the implementation against the Graphiti framework. Potentially the Spray DSL can be used to generate code for other frameworks as well.

The generated code is structured in such a way that one can always extend/overwrite the generated code with handwritten Java to add advanced Graphiti features that are not supported directly by the Spray DSL.

With the help of the tools created in this project Graphiti based diagram editors can be created much faster and reliable than doing it purely by hand.

A short introduction to Spray can be found in the downloads section. The code is in early state and subject to change.

Install introduction

A *very* short introduction on how to install spray and run the example.

1. Install Eclipse Indigo classic IDE.
2. Install new software
   - EMF, Xtext2, MWE2, GEF, Graphiti
3. Import all plugins from the plugins and from the examples directories fro the respository
4. In the example project, look into the ‚model’ directory.
   - you need to change the path in the spray file and the properties file to freflect your situation.
5. Run org.xspray.xtext.generator.Main with the full pathname of the spray model file as the only argument as a file URI, like file://C:/xspray/xspray-runtime/org.xspray.examples.one/model/mod4j-busmod.xspray]
6. This should generate the code in the location described in the .properties file
   - refresh this project and start a new Eclipse]
   - in this eclipse create a project and the a new ==> Example ==> Graphiti Diagram ==> Mode4j

Note that I have only run this on my machine with fixed paths, so you might encounter some problems doing the above

Spray DSL

The Spray grammar

```
grammar org.eclipselabs.spray.xtext.Spray with org.eclipse.xtext.xbase.Xbase

import "platform:/resource/org.eclipselabs.spray.mm/model/spray.ecore"
import "http://www.eclipse.org/emf/2002/Ecore" as ecore
import "platform:/resource/org.eclipse.xtext.common.types/model/JavaVMTypes.ecore" as
 types


Diagram :
    'diagram'  name = ID

    imports+=Import*

    (
        behaviourGroups += BehaviourGroup
    )*
    (
        metaClasses += MetaClass
    )*
 ;

Import:
 'import' importedNamespace=QualifiedNameWithWildCard;

SprayElement :
    Shape | MetaReferenceAsShape | MetaAttributeAsShape;

MetaClass :
    'class'  type=[ecore::EClass|QualifiedName]  ('alias' alias=ID)? ('icon'
 icon=STRING)?
    ":" representedBy=Shape
    (
    "references" "["
        (references += MetaReference2  ";")*
 "]"
    )?
    (
 "behavior" "["
        (
            ("group" behaviourGroups += [BehaviourGroup] ";"  ) |
               (behaviours += Behaviour ";" )
           )*
    "]"
    )?
    ;

MetaReference :
 "reference" reference=[ecore::EReference]
 ":" representedBy=Connection
;
```

```
MetaReference2 returns MetaReference:
 reference=[ecore::EReference]
 ":" representedBy=Connection
;


MetaAttribute :
 (pathsegments+=[ecore::EReference] '.')* attribute=[ecore::EAttribute];

MetaReferenceAsShape returns MetaReference :
    'reference' reference=[ecore::EReference]
    ("attribute" labelProperty = [ecore::EAttribute]  )? ;

MetaAttributeAsShape returns MetaAttribute :
    'attribute' attribute=[ecore::EAttribute];


BehaviourGroup:
    "behaviour" name=ID "["
          (behaviours += Behaviour ";")+
    "]"
;

Behaviour :
    StandardBehaviour | CustomBehaviour;

StandardBehaviour :
    type=BehaviourType  (label = STRING)? ('palette' paletteCompartment = STRING)?;

CustomBehaviour :
    name = ID (label = STRING);

enum BehaviourType :
    CREATE_BEHAVIOUR = 'create'
    ;

Shape :
    (Rectangle | Text | Container | Connection | Line)
;

Layout :
  { Layout }
 '('
        (
            ( 'width'  '=' lineWidth = INT )?
          & ( 'color'  '=' lineColor = Color)?
          & ( 'fill'   '=' fillColor = Color)?
          & ( 'figure' '=' figure    = STRING)?
          & ( bold ?= 'bold' )?
          & ( italic ?= 'italic' )?
        )
      ')'

;
```

```
EString returns ecore::EString:
 STRING | ID;


Rectangle returns Rectangle:
 {Rectangle}
    "rectangle"
 layout = Layout //  name=EString
;

Connection returns Connection :
 {Connection}
 'connection' layout = Layout
 (
     '['
  'from' from=[ecore::EReference] ";"
  'to'   to  =[ecore::EReference] ";"
  (
            ('fromText' fromLabel = Text ";")? &
            ('connectionText' connectionLabel = Text ";")? &
      ('toText' toLabel   = Text ";")?
  )
  ']'
 )?
;

Color:
 ColorConstantRef | RGBColor
;
ColorConstantRef:
 (type=JvmTypeReference '::')? field=[types::JvmField|ValidID]
;

RGBColor:
 'RGB' '(' red=INT ',' green=INT ',' blue=INT ')'
;

Text returns Text:
 {Text}
 'text' layout = Layout
 value = XExpression
 ;

QualifiedNameWithWildCard returns ecore::EString :
 QualifiedName  ('.' '*')?;

Line returns Line :
 {Line}
 'line' layout = Layout
;

Container returns Container:
    { Container }
 'container' layout = Layout
 '['
  (parts+=SprayElement  ";" )*
    ']';

StaticFieldQualifier:
 QualifiedName '::' ValidID
;
```

A Spray based example language

```
diagram mod4j

import BusinessDomainDsl.*
import org.eclipse.graphiti.util.IColorConstant
behaviour samen [
 doit "Do it" ;
]

class BusinessClass:
    container  ( fill=dark_green )
    [
          text ( )  { "<<"+eClass.name+">> " + name};
          line ( color=black width=2);
          reference properties attribute dataType;
          line                  (width=2  color=RGB(255,138,141));
          reference businessRules;   // will use name property by default
          line                  (width=2  color=IColorConstant::DARK_BLUE);

//          attribute description;
          text () "::" + description;
          line                  (width=1  ) ;
 ]
 references [
      superclass : connection();
 ]
    behavior [
        create  palette "Shapes" ;
        doWithBusinessClass "Do It With";
        group samen ;
    ]


class Association icon "connection16.gif":
    connection ()
    [
        from source;
        to   target;
        fromText text()  "source " + source.name;
        connectionText text() targetMultiplicity.name;
        toText text() name;
    ]
    behavior [
        create  palette "Connections" ;
        group samen;
```

]

Spray Extensions Guide

How to extend the generate Graphiti code?