01001000010011101
Institute of Systems Biology

BioUML IDE

Tagir Valeev[1,3], Nikita Mandrik[2,4], Sofiya Kinsht[5], Fedor Kolpakov[1,2]

[1]Institute of Systems Biology, Ltd.,
[2]Design Technological Institute of Digital Techniques, SBRAS
[3]Institute of Informatics Systems, SBRAS
[4]Sobolev Institute of Mathematics, SBRAS
[5]Novosibirsk State University

Novosibirsk, Russia

COMBINE 2015 October 12-16, Salt Lake City, USA

BioUML platform

BioUML is an open source integrated Java platform for building virtual cell and virtual physiological human.
It spans a comprehensive range of capabilities, including access to databases with experimental data, tools for formalized description of biological systems structure and functioning, as well as tools for their visualization, simulation, parameters fitting and analyses.
Due to scripts (R, JavaScript) and workflow support it provides powerful possibilities for analyses of high-throughput data. The plug-in based architecture allows to add new functionality using plug-ins.

BioUML platform available as:

- BioUML workbench – standalone Java application
- BioUML web edition – client/server application for collaborative work
- BioUML IDE – integrated development enviroment.

Why BioUML IDE?

BioUML platform supports **3 problem domain specific languages**:

- Antimony
- BioNetGen
- YAML (for SED-ML)

We need:

- modern code editor for these languages:
  - syntax highlighting
  - code complete
  - code folding
- version control system (like git).

Solution:

Use NetBeans IDE as core of BioUML IDE.

Why NetBeans?

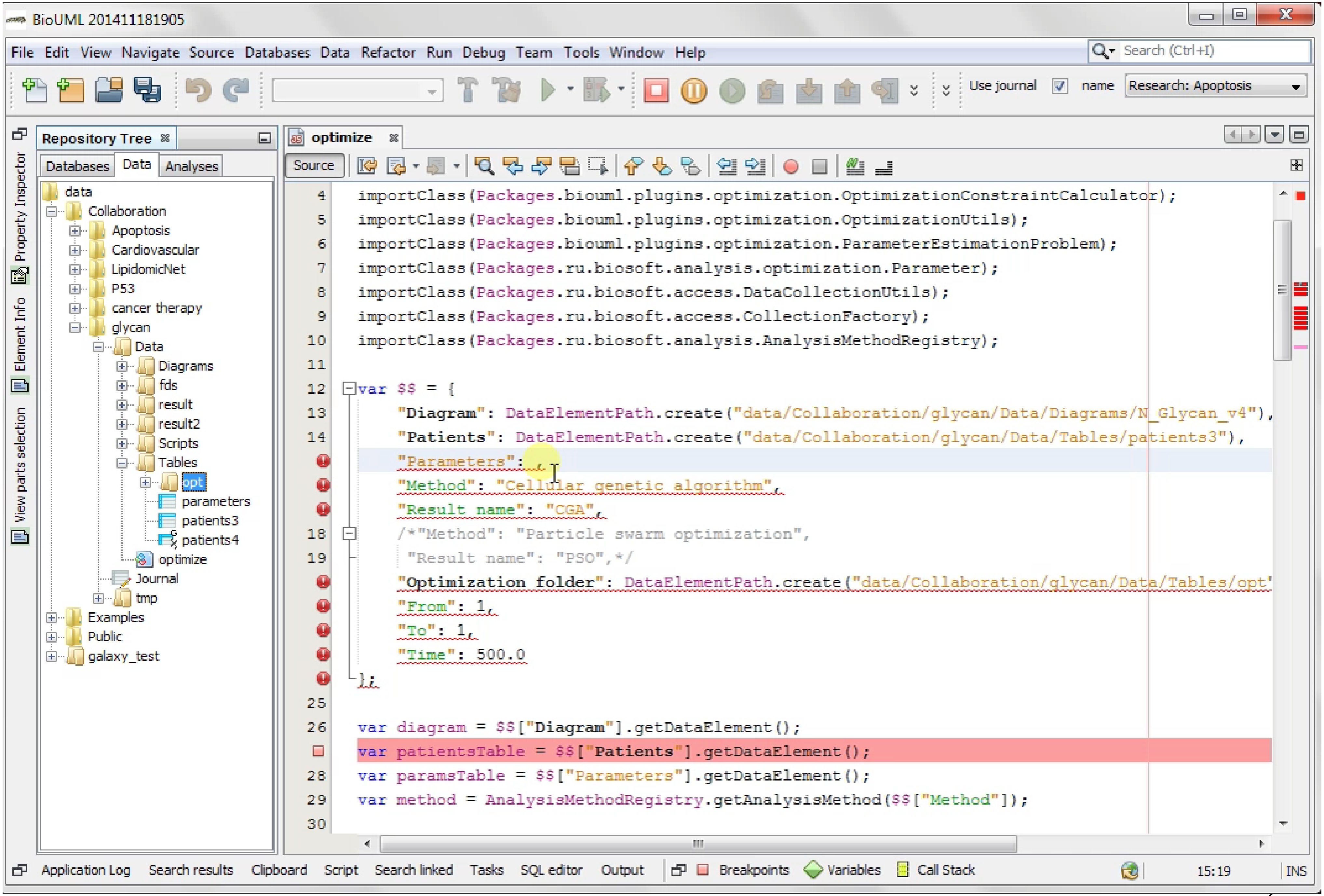
- One of the best modern IDE
  - modern customizable UI
  - powerful code editor
  - debugger
  - git support
  -
- Supports many languages (Java, JavaScript, Python, etc.)
- Possibility to add new languages

- Free, open source platform
- 100% Java
- Plug-in based (supports OSGI plug-ins)
- Architecture is similar with BioUML → easy integration

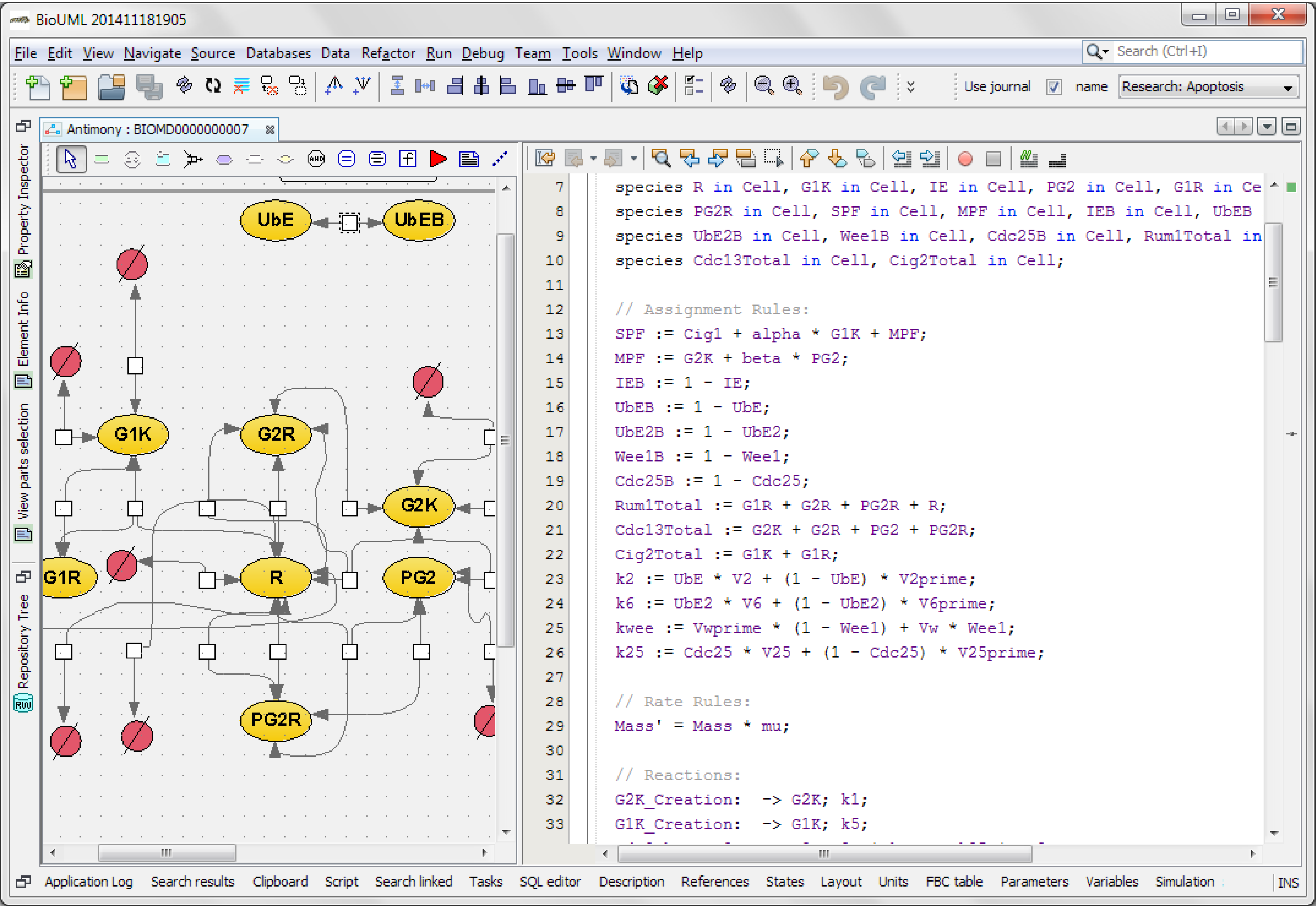
Code editor

Example (JavaScript editor):

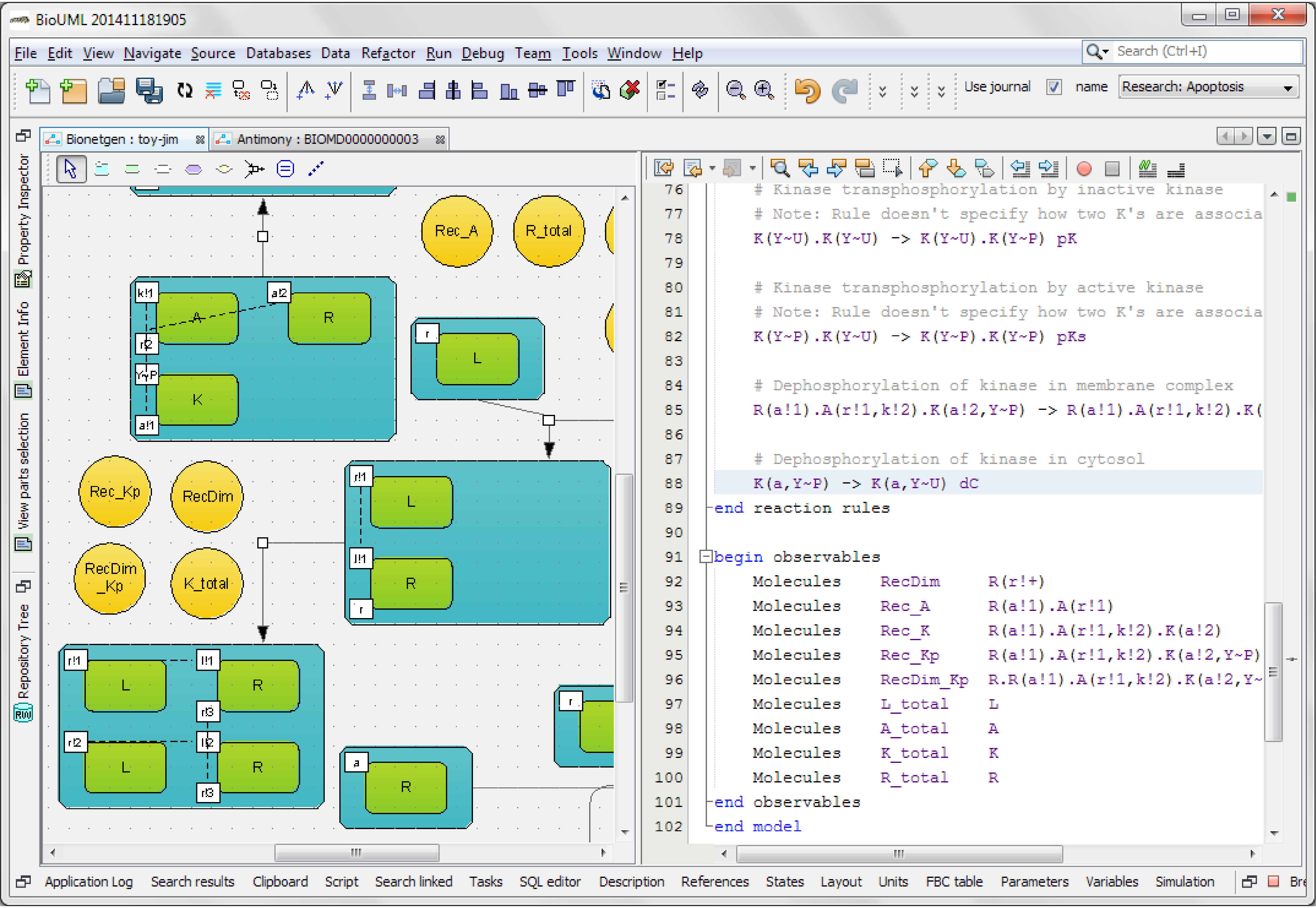
- indents lines
- matches words and brackets
- highlights source code syntactically and semantically
- provides intelligent code completion
- debugger



```
importClass(Packages.biouml.plugins.optimization.OptimizationConstraintCalculator);
importClass(Packages.biouml.plugins.optimization.OptimizationUtils);
importClass(Packages.biouml.plugins.optimization.ParameterEstimationProblem);
importClass(Packages.ru.biosoft.analysis.optimization.Parameter);
importClass(Packages.ru.biosoft.access.DataCollectionUtils);
importClass(Packages.ru.biosoft.access.CollectionFactory);
importClass(Packages.ru.biosoft.analysis.AnalysisMethodRegistry);

var $$ = {
    "Diagram": DataElementPath.create("data/Collaboration/glycan/Data/Diagrams/N_Glycan_v4"),
    "Patients": DataElementPath.create("data/Collaboration/glycan/Data/Tables/patients3"),
    "Parameters": ,
    "Method": "Cellular genetic algorithm",
    "Result name": "CGA",
    /*"Method": "Particle swarm optimization",
     "Result name": "PSO",*/
    "Optimization folder": DataElementPath.create("data/Collaboration/glycan/Data/Tables/opt'
    "From": 1,
    "To": 1,
    "Time": 500.0
};

var diagram = $$["Diagram"].getDataElement();
var patientsTable = $$["Patients"].getDataElement();
var paramsTable = $$["Parameters"].getDataElement();
var method = AnalysisMethodRegistry.getAnalysisMethod($$["Method"]);
```


BioUML 201411181905
File Edit View Navigate Source Databases Data Refactor Run Debug Team Tools Window Help
Search (Ctrl+I)
Use journal ☑ name Research: Apoptosis
Antimony : BIOMD0000000007
Property Inspector
Element Info
View parts selection
Repository Tree
UbE
UbEB
G1K
G2R
G2K
G1R
R
PG2
PG2R
Application Log Search results Clipboard Script Search linked Tasks SQL editor Description References States Layout Units FBC table Parameters Variables Simulation INS

```
species R in Cell, G1K in Cell, IE in Cell, PG2 in Cell, G1R in Ce
species PG2R in Cell, SPF in Cell, MPF in Cell, IEB in Cell, UbEB
species UbE2B in Cell, Wee1B in Cell, Cdc25B in Cell, Rum1Total in
species Cdc13Total in Cell, Cig2Total in Cell;

// Assignment Rules:
SPF := Cig1 + alpha * G1K + MPF;
MPF := G2K + beta * PG2;
IEB := 1 - IE;
UbEB := 1 - UbE;
UbE2B := 1 - UbE2;
Wee1B := 1 - Wee1;
Cdc25B := 1 - Cdc25;
Rum1Total := G1R + G2R + PG2R + R;
Cdc13Total := G2K + G2R + PG2 + PG2R;
Cig2Total := G1K + G1R;
k2 := UbE * V2 + (1 - UbE) * V2prime;
k6 := UbE2 * V6 + (1 - UbE2) * V6prime;
kwee := Vwprime * (1 - Wee1) + Vw * Wee1;
k25 := Cdc25 * V25 + (1 - Cdc25) * V25prime;

// Rate Rules:
Mass' = Mass * mu;

// Reactions:
G2K_Creation:  -> G2K; k1;
G1K_Creation:  -> G1K; k5;
```


BioUML 201411181905
File Edit View Navigate Source Databases Data Refactor Run Debug Team Tools Window Help
Search (Ctrl+I)
Use journal ☑ name Research: Apoptosis
Bionetgen : toy-jim
Antimony : BIOMD0000000003
Property Inspector
Element Info
View parts selection
Repository Tree
Rec_A
R_total
k!1
a!2
A
R
r!2
Y~P
K
a!1
r
L
Rec_Kp
RecDim
RecDim _Kp
K_total
r!1
L
l!1
R
r
r!1
L
l!1
R
r!3
r!2
L
l!2
R
r!3
a
R
r
Application Log Search results Clipboard Script Search linked Tasks SQL editor Description References States Layout Units FBC table Parameters Variables Simulation Bre

```
    # Kinase transphosphorylation by inactive kinase
    # Note: Rule doesn't specify how two K's are associa
    K(Y~U).K(Y~U) -> K(Y~U).K(Y~P) pK

    # Kinase transphosphorylation by active kinase
    # Note: Rule doesn't specify how two K's are associa
    K(Y~P).K(Y~U) -> K(Y~P).K(Y~P) pKs

    # Dephosphorylation of kinase in membrane complex
    R(a!1).A(r!1,k!2).K(a!2,Y~P) -> R(a!1).A(r!1,k!2).K(

    # Dephosphorylation of kinase in cytosol
    K(a,Y~P) -> K(a,Y~U) dC
end reaction rules

begin observables
    Molecules    RecDim      R(r!+)
    Molecules    Rec_A       R(a!1).A(r!1)
    Molecules    Rec_K       R(a!1).A(r!1,k!2).K(a!2)
    Molecules    Rec_Kp      R(a!1).A(r!1,k!2).K(a!2,Y~P)
    Molecules    RecDim_Kp   R.R(a!1).A(r!1,k!2).K(a!2,Y~
    Molecules    L_total     L
    Molecules    A_total     A
    Molecules    K_total     K
    Molecules    R_total     R
end observables
end model
```

BioUML IDE is available for downloads on BioStore (https://bio-store.org/).

More information about BioUML you can find on our site www.biouml.org.

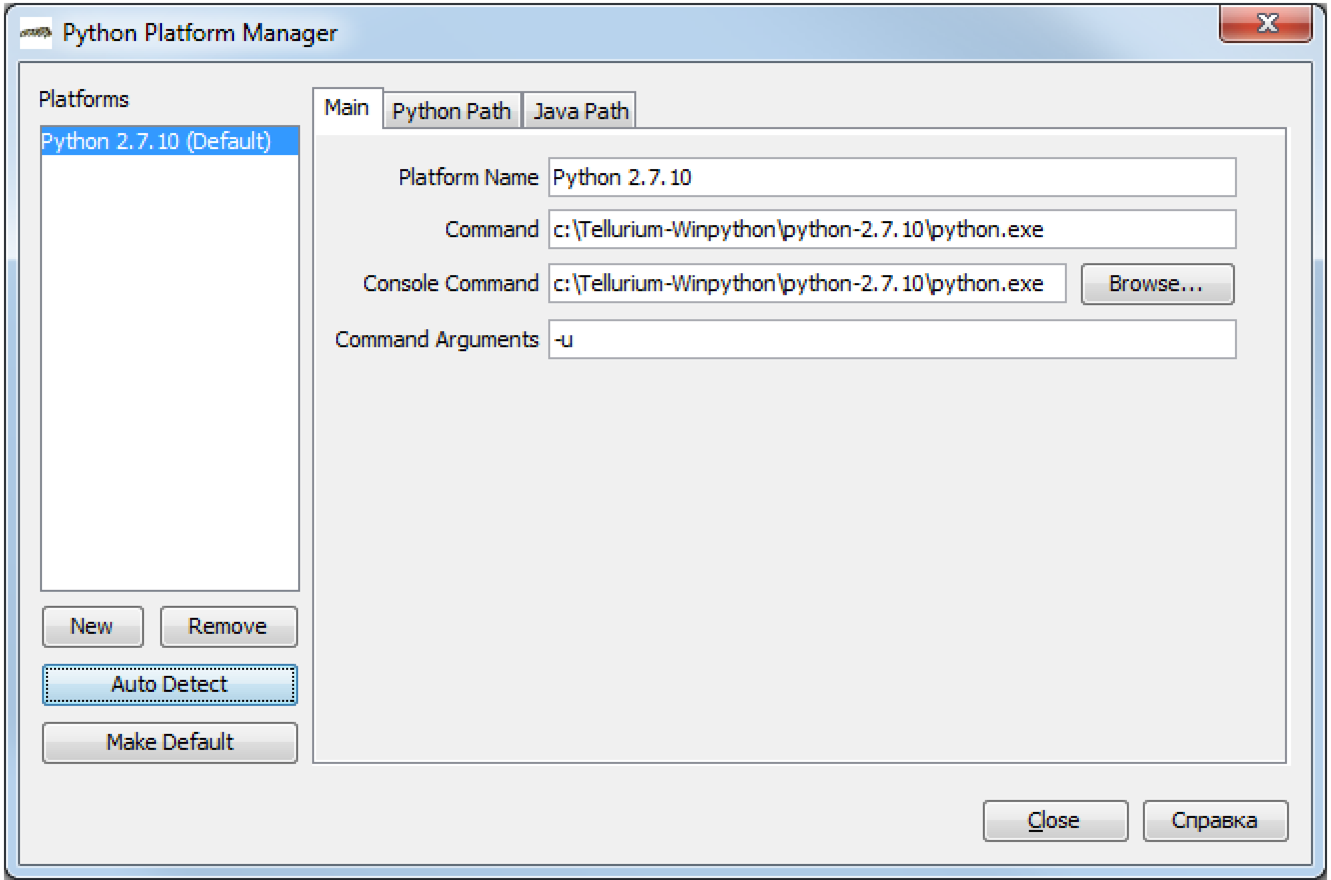
Thank you for attention!

Bonus:
Python in BioUML IDE
(win)

Python in BioUML IDE

We can use Python scripts from Tellurium:

- install Tellurium
- install Python plugins for Net Beans
- https://blogs.oracle.com/geertjan/entry/python_in_netbeans_ide_81
- specify Python platform from Tellurium
- create Python project
- specify source files directory – C:\Tellurium-Winpython\tellurium-winpython-files
- open any Python script
- start it



Python Platform Manager

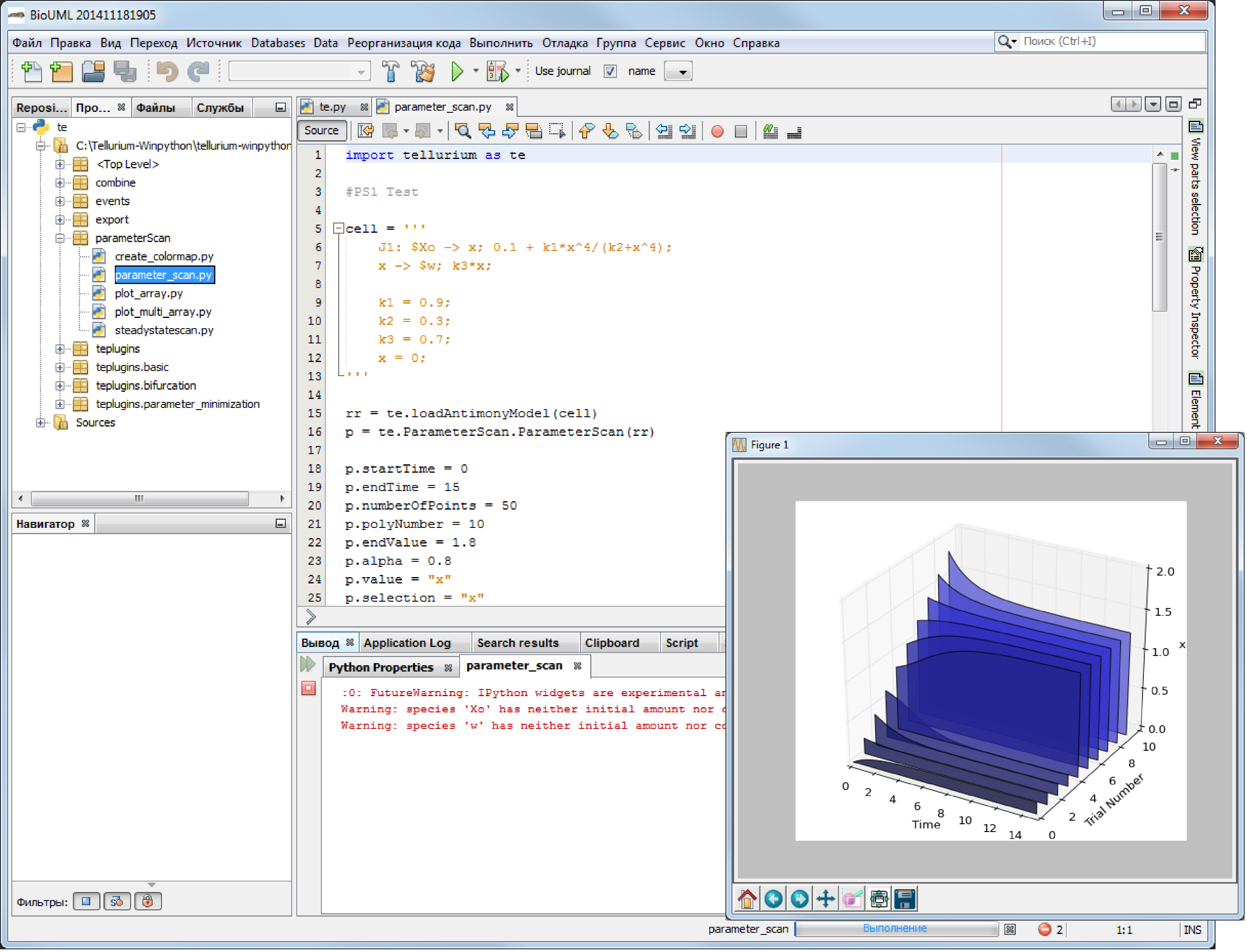
Platforms: Python 2.7.10 (Default)

Platform Name: Python 2.7.10

Command: c:\Tellurium-Winpython\python-2.7.10\python.exe

Console Command: c:\Tellurium-Winpython\python-2.7.10\python.exe

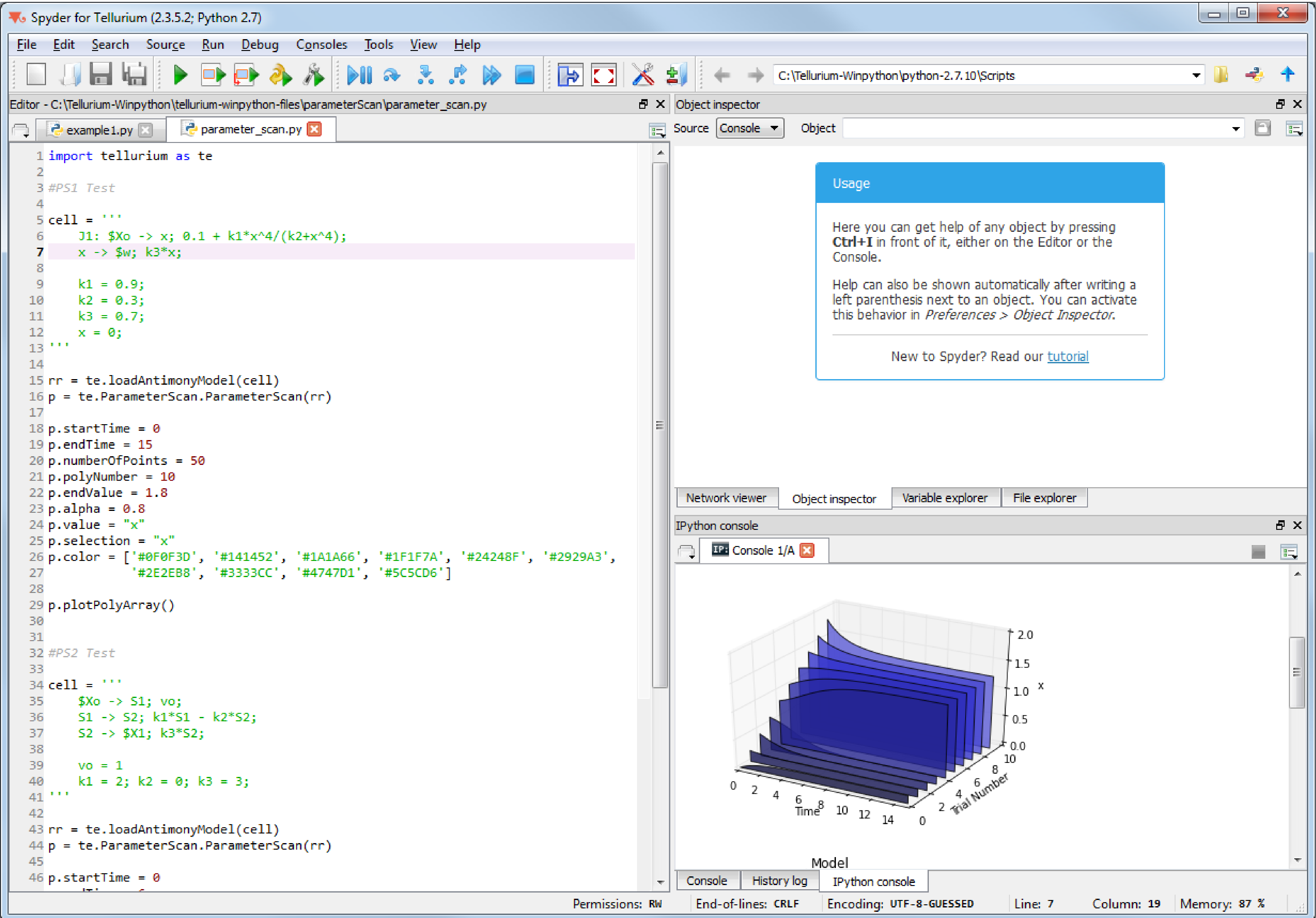
Command Arguments: -u



```
import tellurium as te

#PS1 Test

cell = '''
    J1: $Xo -> x; 0.1 + k1*x^4/(k2+x^4);
    x -> $w; k3*x;

    k1 = 0.9;
    k2 = 0.3;
    k3 = 0.7;
    x = 0;
'''

rr = te.loadAntimonyModel(cell)
p = te.ParameterScan.ParameterScan(rr)

p.startTime = 0
p.endTime = 15
p.numberOfPoints = 50
p.polyNumber = 10
p.endValue = 1.8
p.alpha = 0.8
p.value = "x"
p.selection = "x"
```

```
:0: FutureWarning: IPython widgets are experimental ar
Warning: species 'Xo' has neither initial amount nor c
Warning: species 'w' has neither initial amount nor co
```

Compare with Tellurium



```
import tellurium as te

#PS1 Test

cell = '''
    J1: $Xo -> x; 0.1 + k1*x^4/(k2+x^4);
    x -> $w; k3*x;

    k1 = 0.9;
    k2 = 0.3;
    k3 = 0.7;
    x = 0;
'''

rr = te.loadAntimonyModel(cell)
p = te.ParameterScan.ParameterScan(rr)

p.startTime = 0
p.endTime = 15
p.numberOfPoints = 50
p.polyNumber = 10
p.endValue = 1.8
p.alpha = 0.8
p.value = "x"
p.selection = "x"
p.color = ['#0F0F3D', '#141452', '#1A1A66', '#1F1F7A', '#24248F', '#2929A3',
           '#2E2EB8', '#3333CC', '#4747D1', '#5C5CD6']

p.plotPolyArray()


#PS2 Test

cell = '''
    $Xo -> S1; vo;
    S1 -> S2; k1*S1 - k2*S2;
    S2 -> $X1; k3*S2;

    vo = 1
    k1 = 2; k2 = 0; k3 = 3;
'''

rr = te.loadAntimonyModel(cell)
p = te.ParameterScan.ParameterScan(rr)

p.startTime = 0
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Object Inspector*.

New to Spyder? Read our tutorial

Permissions: RW   End-of-lines: CRLF   Encoding: UTF-8-GUESSED   Line: 7   Column: 19   Memory: 87 %