Creating Domain Specific Languages with Xtext

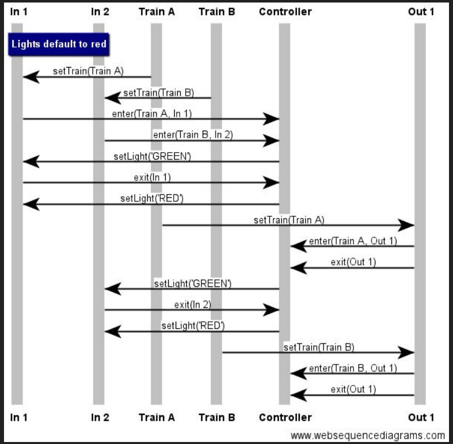
Andrés Carrasco

Domain Specific Language (DSL)

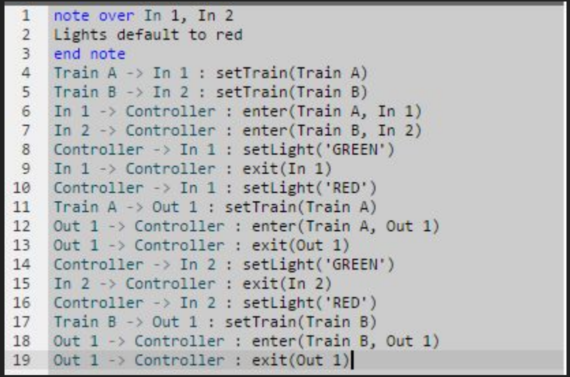
- Is not a general purpose language (GPL)
  - Java, C, etc..
  - UML
- Language specialized in a specific domain
  - SQL, HTML, etc..
- Must be carefully designed
  - Does the DSL expresses the problem or solution more clearly than an already existing language?
  - Will it be worthwhile to create a DSL?
- External vs Internal DSL

Textual vs Visual Languages

- Visual Languages
  - Good for overviews
  - Intuitive
  - Easier to constrain the user



- Textual Languages
  - Compact
  - Easy formatting
  - Platform independency
  - Language integration



```
note over In 1, In 2
Lights default to red
end note
Train A -> In 1 : setTrain(Train A)
Train B -> In 2 : setTrain(Train B)
In 1 -> Controller : enter(Train A, In 1)
In 2 -> Controller : enter(Train B, In 2)
Controller -> In 1 : setLight('GREEN')
In 1 -> Controller : exit(In 1)
Controller -> In 1 : setLight('RED')
Train A -> Out 1 : setTrain(Train A)
Out 1 -> Controller : enter(Train A, Out 1)
Out 1 -> Controller : exit(Out 1)
Controller -> In 2 : setLight('GREEN')
In 2 -> Controller : exit(In 2)
Controller -> In 2 : setLight('RED')
Train B -> Out 1 : setTrain(Train B)
Out 1 -> Controller : enter(Train B, Out 1)
Out 1 -> Controller : exit(Out 1)
```

H. Gr□önninger, H. Krahn, B. Rumpe, M. Schindler, S. V□
ölkel,
Textbased Modeling, ArXiv e-printsarXiv:1409.6623.

Textual DSL Requires

- Ability to read input text
- Parse the input text
- Process the input text

Textual DSL Could

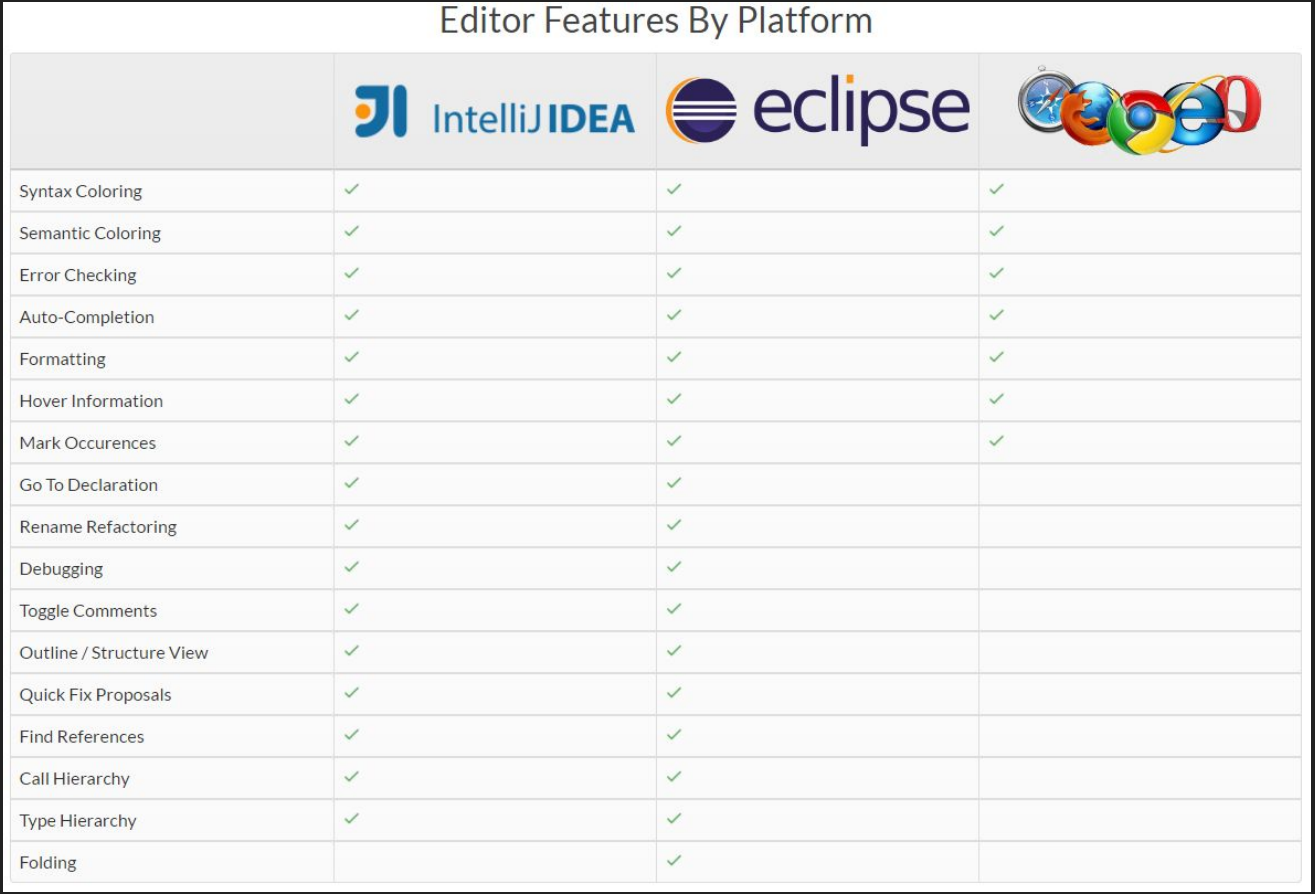
- Interpret the input text
- Transform it into another language

Takes a fair amount of work to do all of the required software to support a new DSL

L. Bettini, Implementing Domain-Specific Languages with Xtext and Xtend, Packt Publishing Ltd, 2013.

Xtext

Does all of that for you and more!



Editor Features By Platform

	IntelliJ IDEA	eclipse	
Syntax Coloring	✓	✓	✓
Semantic Coloring	✓	✓	✓
Error Checking	✓	✓	✓
Auto-Completion	✓	✓	✓
Formatting	✓	✓	✓
Hover Information	✓	✓	✓
Mark Occurences	✓	✓	✓
Go To Declaration	✓	✓	
Rename Refactoring	✓	✓	
Debugging	✓	✓	
Toggle Comments	✓	✓	
Outline / Structure View	✓	✓	
Quick Fix Proposals	✓	✓	
Find References	✓	✓	
Call Hierarchy	✓	✓	
Type Hierarchy	✓	✓	
Folding		✓	

http://www.eclipse.org/Xtext/

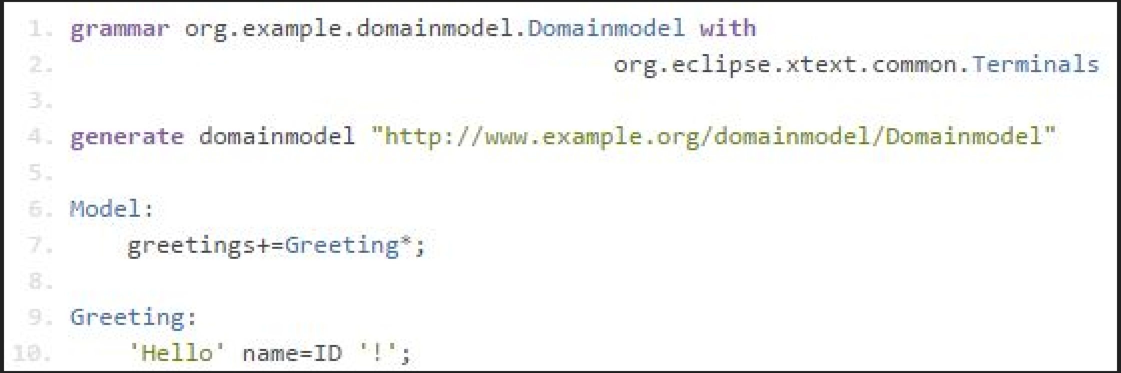
Xtext

- Based on EMF (Eclipse Modeling Framework)
- Uses ANTLR for parsing
  - Another Tool For Language Recognition
  - Widely used parser (Java, Python, JavaScript, etc.)
- Good defaults & completely customizable
  - Works well enough out-of-the-box
- Integration with Java through Xbase
  - Allows to create Java Code from your DSL

http://www.eclipse.org/Xtext/

Xtext: How does it work?

- First: Specify the grammar using their Grammar Language
- Next: Generate Language Artifacts
  - Lexer & Parser
  - AST Model
  - IDE Support & Features
- And Done!



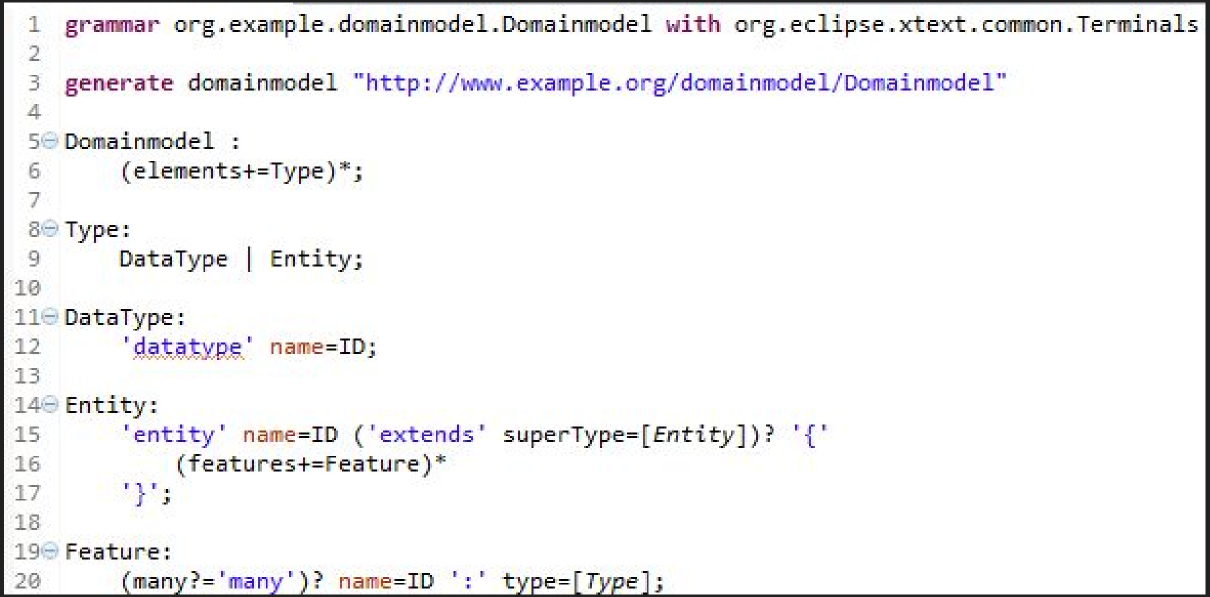
```
grammar org.example.domainmodel.Domainmodel with
                                      org.eclipse.xtext.common.Terminals

generate domainmodel "http://www.example.org/domainmodel/Domainmodel"

Model:
    greetings+=Greeting*;

Greeting:
    'Hello' name=ID '!';
```

Simple Hello World grammar

http://www.eclipse.org/Xtext/

Example Grammar

- The *Domainmodel* can have *Type* elements
- *Type* elements can be either of *DataType* or *Entity*
- An *Entity* can have *Features* inside



```
grammar org.example.domainmodel.Domainmodel with org.eclipse.xtext.common.Terminals

generate domainmodel "http://www.example.org/domainmodel/Domainmodel"

Domainmodel :
    (elements+=Type)*;

Type:
    DataType | Entity;

DataType:
    'datatype' name=ID;

Entity:
    'entity' name=ID ('extends' superType=[Entity])? '{'
        (features+=Feature)*
    '}';

Feature:
    (many?='many')? name=ID ':' type=[Type];
```

http://www.eclipse.org/Xtext/