



# Разработка приложений на основе DSL и генерации кода

Игорь Ткачев, [igor.tkachev@db.com](mailto:igor.tkachev@db.com)  
23-24 Октября 2014

*Passion to Perform*



# Содержание

1	Что такое DSL
2	Производительность труда
3	Уровни абстракции
4	Средства разработки
5	Примеры
6	Область применения
7	Проблемы подхода
8	Заключение
9	Q & A

Что такое DSL



Domain specific language

vs

General purpose language

# Производительность труда

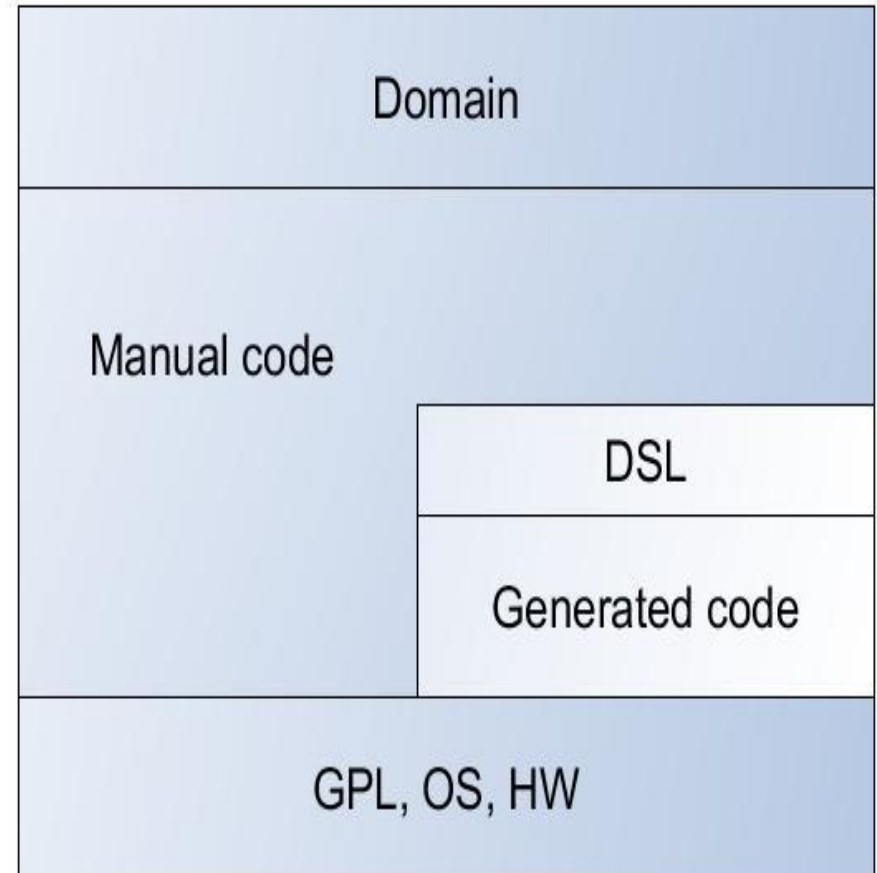
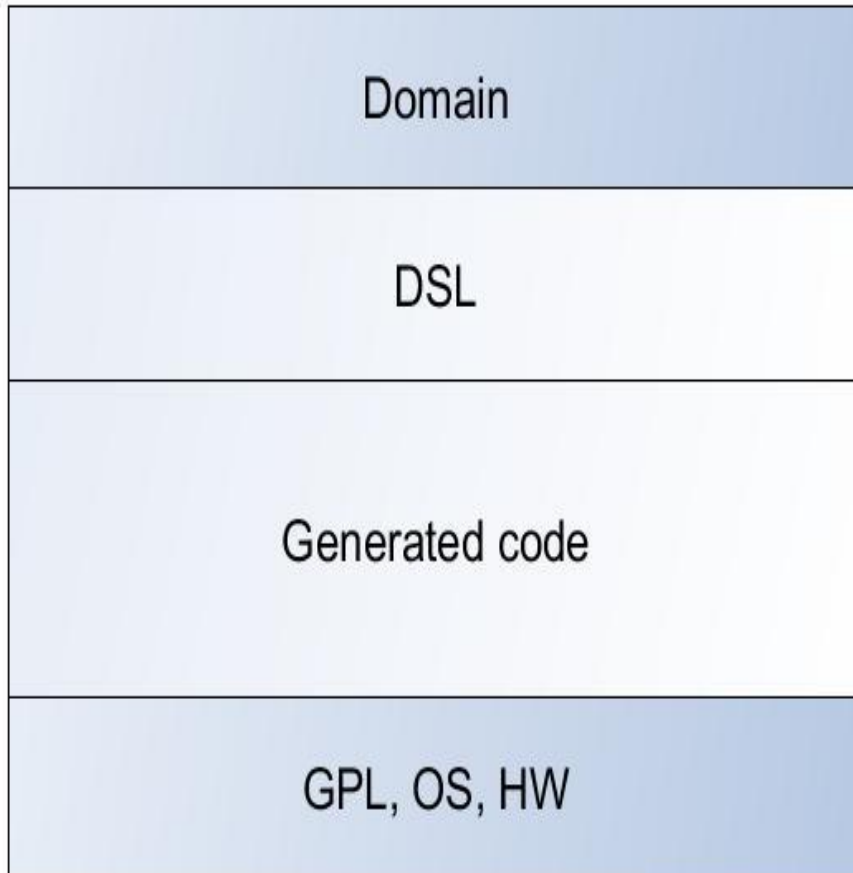


Идея, требование,  
спецификация



Конечный  
продукт

# Уровни абстракции



# Средства разработки



- MPS (Jetbrains)
- Eclipse Xtext



The screenshot shows the Eclipse Xtext website with the title "A Selection Of Supported Features". It features six panels, each illustrating a different capability of the Xtext editor:

- Syntax Coloring:** Shows a code snippet from "Arithmetics.calc" with different parts highlighted in colors (green for keywords, blue for identifiers, red for operators, etc.).
- Content Assist:** Shows a code snippet with a dropdown menu of suggestions (ID, INT, Import) appearing over a cursor.
- Validation and Quick Fixes:** Shows a code snippet with a red squiggly line under a value and a tooltip offering a "Replace with 42" quick fix.
- Advanced Java Integration:** Shows a code snippet with a tooltip for "java.util.Set<E>" appearing over a type reference.
- Integration with other Eclipse tools:** Shows a diagrammatic view of a class hierarchy with "Person" and "Address" classes and their attributes.
- More IDE Features:** Shows a search results window listing matches for a search query across different files.

# Пример – определение языка



```
1 grammar org.xtext.example.mydsl.MyDsl with org.eclipse.xtext.common.Terminals
2
3 generate myDsl "http://www.xtext.org/example/mydsl/MyDsl"
4
5 Model:
6     item+=(Person|Question)*
7 ;
8
9 Person:
10     'person' name=ID
11     (canAskQuestion?='can ask question')?
12     (question=[Question])?
13 ;
14
15 Question:
16     'question' name=ID question=STRING
17 ;
```



# Пример - валидация



```
/**
 * Custom validation rules.
 *
 * see http://www.eclipse.org/Xtext/documentation.html#validation
 */
class MyDslValidator extends AbstractMyDslValidator {

    @Check
    def checkGreetingStartsWithCapital(Person person) {
        if (!person.canAskQuestion && person.question != null) {
            error("Why are you asking questions? You can't!",
                MyDslPackage.Literals.PERSON__NAME,
                "Validation failed")
        }
    }
}
```


# Пример - DSL



```
person Ivan can ask question Q1
person Petr can ask question Q2
person Vasiliy
```

```
question Q1 "Сколько?"
question Q2 "Доколе?!"
```

```
person Ivan can ask question Q1
person Petr can ask question Q2
person Vasiliy Q2
```

 Why are you asking questions? You can't!

Press 'F2' for focus

```
question Q1 "Сколько?"
question Q2 "Доколе?!"
```

# Пример – генерация кода



```
class MyDslGenerator implements IGenerator {  
  
    override void doGenerate(Resource resource, IFileSystemAccess fsa) {  
        val persons = resource.allContents.filter(typeof(Person)).toList  
        val content =  
        ...  
        package person;  
  
        public class Persons {  
            public static void main(String [] args) {  
                «FOR person : persons»  
                System.out.println(" - Привет! Я «person.name», - поздоровался «person.name».");  
                «ENDFOR»  
                «persons.filter[question != null].map[  
                    '''System.out.println(" - «question.question», - спросил «name».");'''  
                ].join("\n")»  
                «persons.filter[question == null].map[  
                    '''System.out.println("«name» промолчал...");'''  
                ].join("\n")»  
            }  
        }  
        ...  
        fsa.generateFile('person/Persons.java', content);  
    }  
}
```

# Пример – результат работы генератора

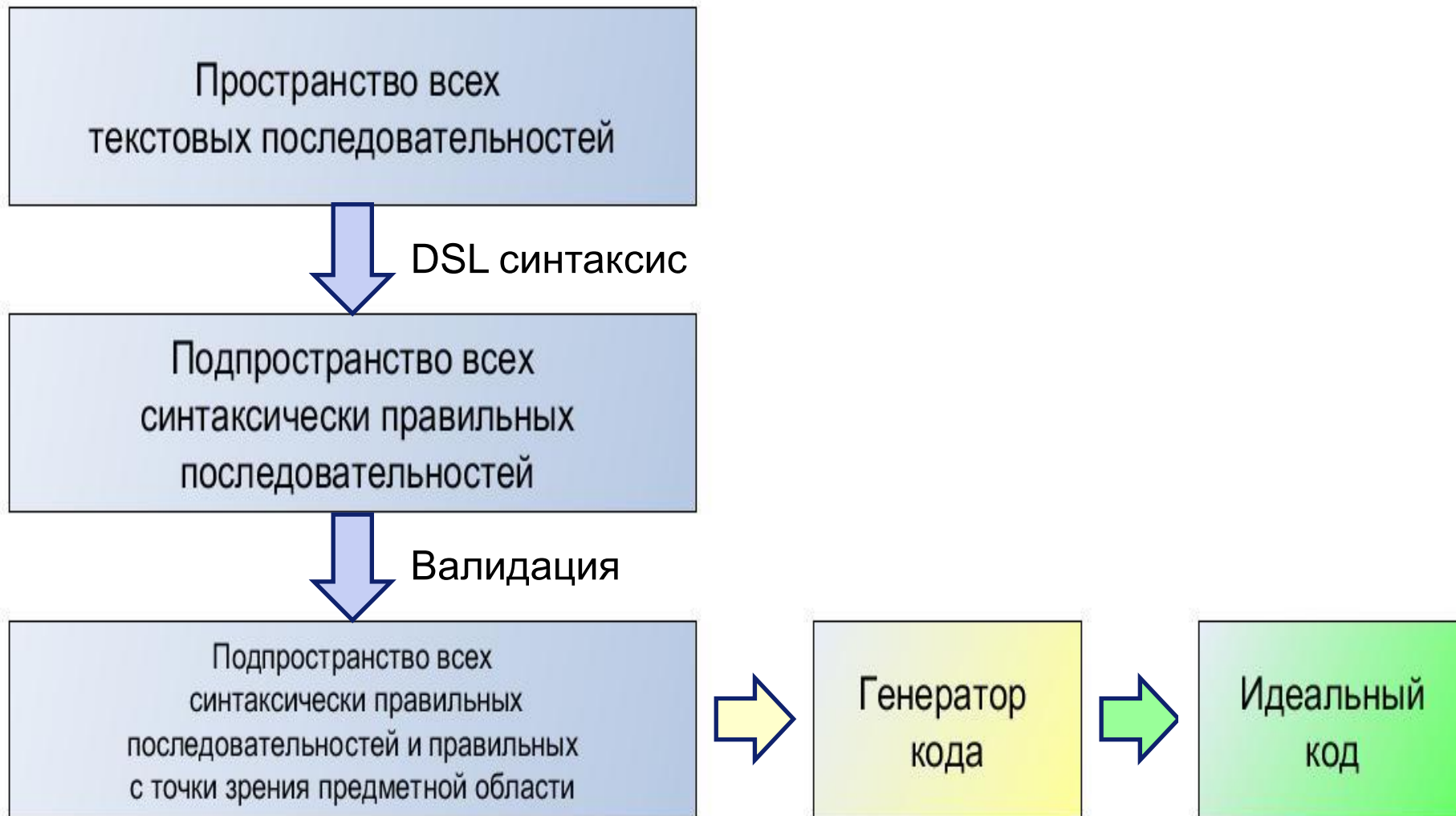


```
package person;

public class Persons {

    public static void main(String [] args) {
        System.out.println(" - Привет! Я Ivan, - поздоровался Ivan.");
        System.out.println(" - Привет! Я Petr, - поздоровался Petr.");
        System.out.println(" - Привет! Я Vasiliy, - поздоровался Vasiliy.");
        System.out.println(" - Сколько?, - спросил Ivan.");
        System.out.println(" - Доколе?!, - спросил Petr.");
        System.out.println("Vasiliy промолчал...");
    }
}
```

# Генерация кода



# Пример - settings



```
settings class "SettingsContainer"
settings package crs.crscontroller.settings
{
  setting StockSetting {
    size UNI
    slice package FLORIDA.dataobject
    key ric
    mlab groups florida dca

    fields {
      string ric { desc "RIC" label "RIC" }
      boolean noTrade { desc "Stock NoTrade flag" label "No trade" default "true" }
      double maxADV { desc "Maximum ADV" label "Max ADV" default "25" }
      double longDelta { desc "Long Delta" label "Long Delta"}
      double shortDelta { desc "Short Delta" label "Short Delta"}
      double maxBuyLimit { desc "Maximum buy limit per cycle" label "Max Buy Limit" default "1e6"}
      double maxSellLimit { desc "Maximum sell limit per cycle" label "Max Sell Limit" default "1e6"}
      double quoteSize { desc "Quote size" label "Quote size" default "10000"}
      double maxQuotes { desc "Multiplier for quoteSize" label "Max Quotes" default "1"}
      double maxRisk { desc "Maximum risk" label "Max risk"}
      double unwindPctRate { desc "Maximum unwind participation rate" label "Unwind Part Rate, %" default "10"}
      double useAlphaInd { desc "On/Off Switch for Alpha Model" label "Enable Alpha" default "1" }
      double caLongSkew { desc "threshold for skew long horizon" label "Long Skew" default "0.8" }
    }

    py include group trading {
      noTrade
      longDelta
      shortDelta
    }
  }
}
```

# Пример - streams



```
revision "$Id: $"

streams NCRS {

    stream in_settings_audit
    comment "settings audit stream"
    +kdbpub crssetting
    +aleripub insert_only
    +alerimodel store="memory" {
        symbol instance comment "This is symbol"
        timestamp ts key
        symbol category key
        symbol key_ key kdbcolumn="settingkey"
        symbol user_ key kdbcolumn="user"
        double maxDelta
        double minDelta
        double longDelta
        double shortDelta
        symbol buyMode
        symbol sellMode
        double longAdv
        double shortAdv
        double totalBuy
        double totalSell
        int maxFills
        double maxFillNotional
        double maxBeta
        double minBeta
        double longBeta
        double shortBeta
        string value_ kdbcolumn="value"
    }
    reversion {
        maxDelta <=> -minDelta
        longDelta <=> shortDelta
        buyMode <=> sellMode
        longAdv <=> shortAdv
        totalBuy <=> totalSell
        maxBeta <=> -minBeta
        longBeta <=> -shortBeta
    }
}
```

# Область применения



- очевидные
  - много однотипных объектов
  - протоколы
  - сериализация, базы данных
  - разные языки программирования
- неочевидные
  - можно применять почти везде



# Проблемы



- Не всегда понятно что генерировать
- Баланс автоматического кода и ручного. Взаимная зависимость
- «Повреждение» мозга генерацией кода
- Сложно убедить заказчика в правильности подхода

# Заключение



- Средство повышения производительности
- Повышение уровня абстракции
- Борьба со сложностью
- Автоматизация рутинного программирования
- Формализация спецификаций

# Вопросы



Контакты:

[igor.tkachev@db.com](mailto:igor.tkachev@db.com)

<http://habrahabr.ru/users/xaphan/>