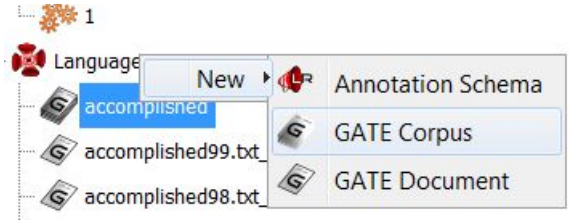
18.05.2015

# Использование Searchable DataStore для поиска закономерностей

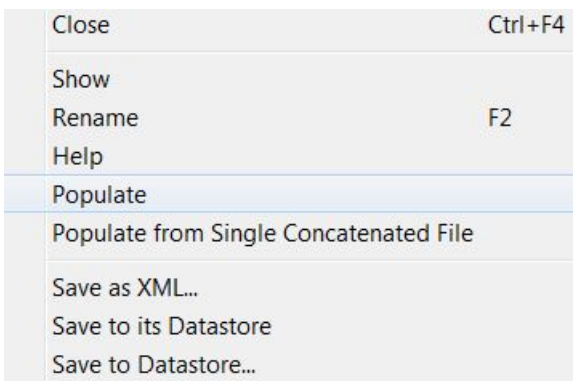
А.В.Поршнев

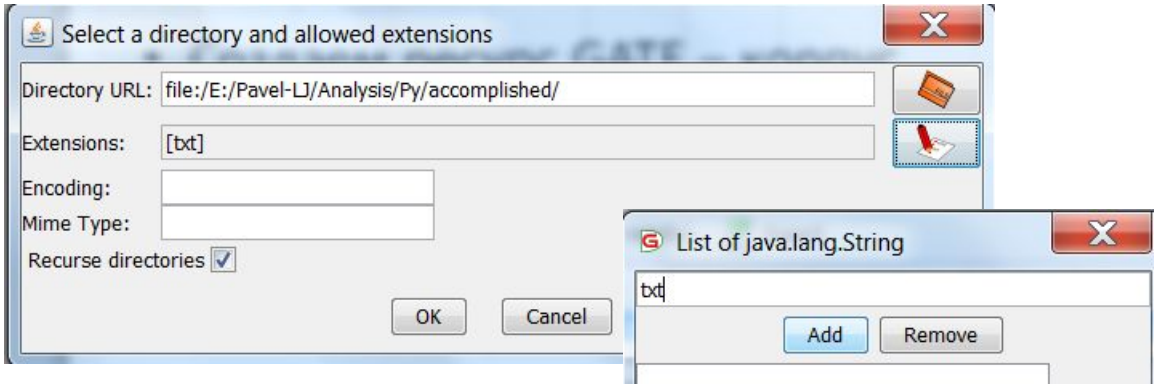
# Создаем корпус файлов

- Создаем ресурс GATE – корпус




- Наполняем корпус файлами



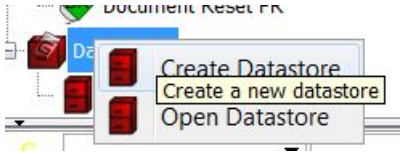
# Обрабатываем корпус

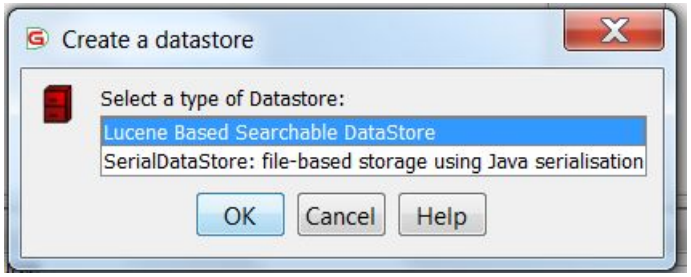
- Создаем стандартную последовательность обработки
- Что позволяет нам не только искать слова, но части речи, именованные

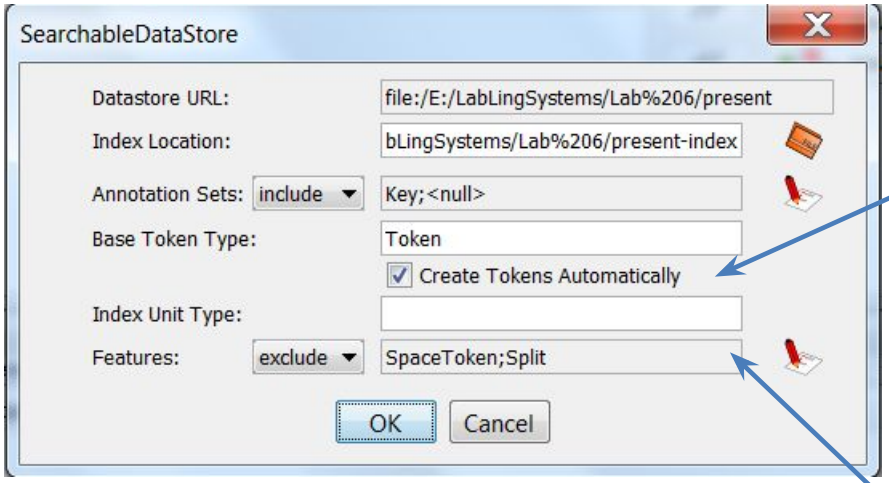
Selected Processing resources

| ! | Name | Type |
|---|---|---|
| ● | Document Reset PR | Document Reset PR |
| ● | ANNIE Sentence Splitter | ANNIE Sentence Splitter |
| ● | ANNIE English Tokeniser | ANNIE English Tokeniser |
| ● | ANNIE Gazetteer | ANNIE Gazetteer |
| ● | ANNIE POS Tagger | ANNIE POS Tagger |
| ● | ANNIE NE Transducer | ANNIE NE Transducer |

# Создаем индексированную БД



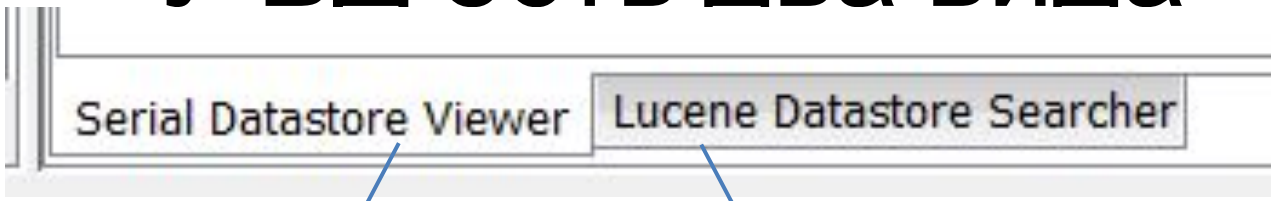



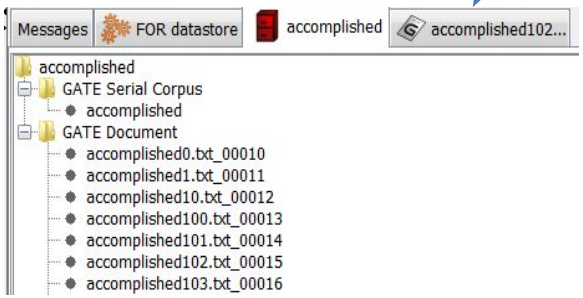



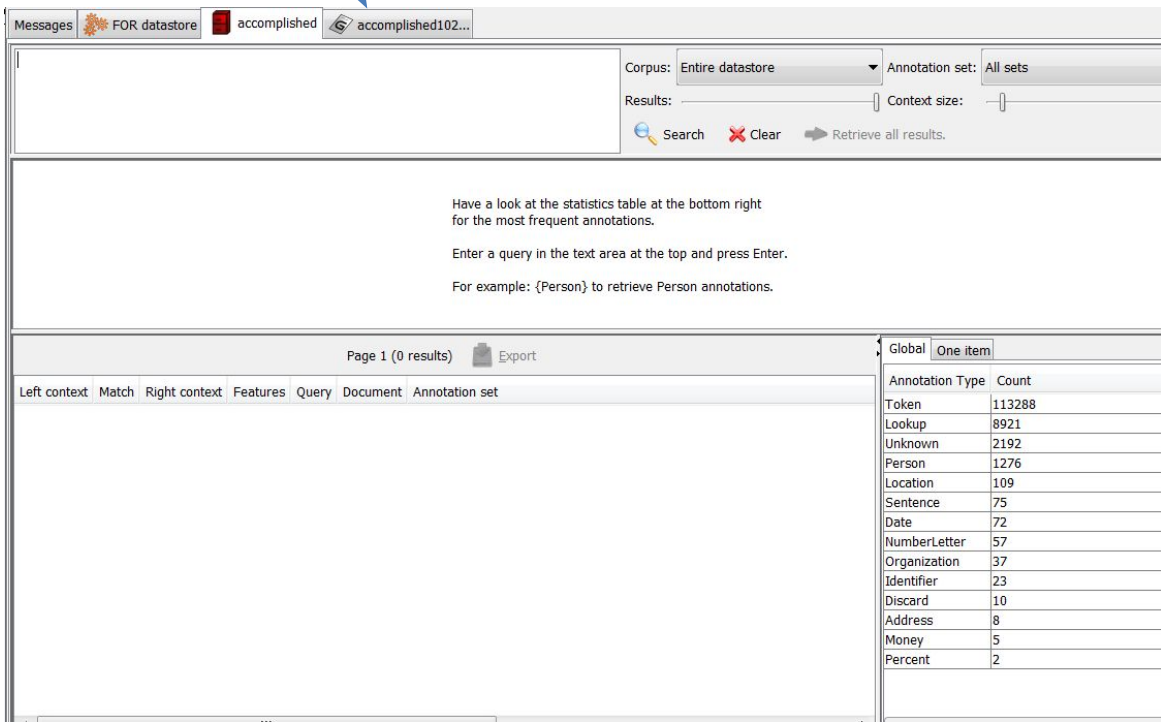

При создании индексов тесты м.б. автоматически разбиты на единицы,
но чтобы иметь информацию о частях
речи нужно применить к тексту
При создании индексов по умолчанию
не включаются SpaceToken и Split

Значит нельзя будет поймать последовательность {Token}{SpaceToken},
но обычно в этом нет необходимости

# У БД есть два вида

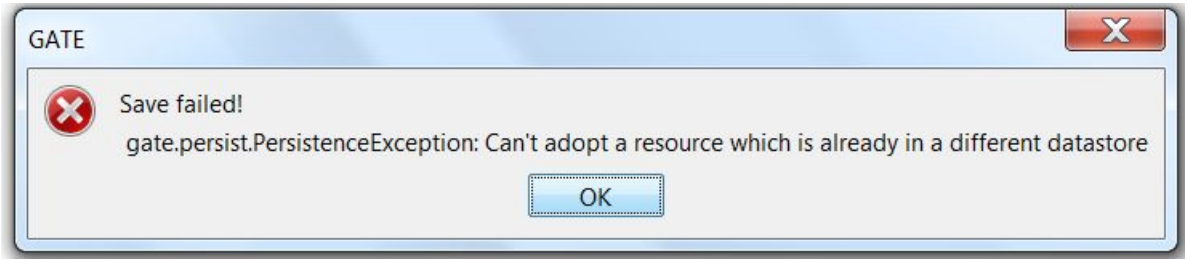


# Внимание

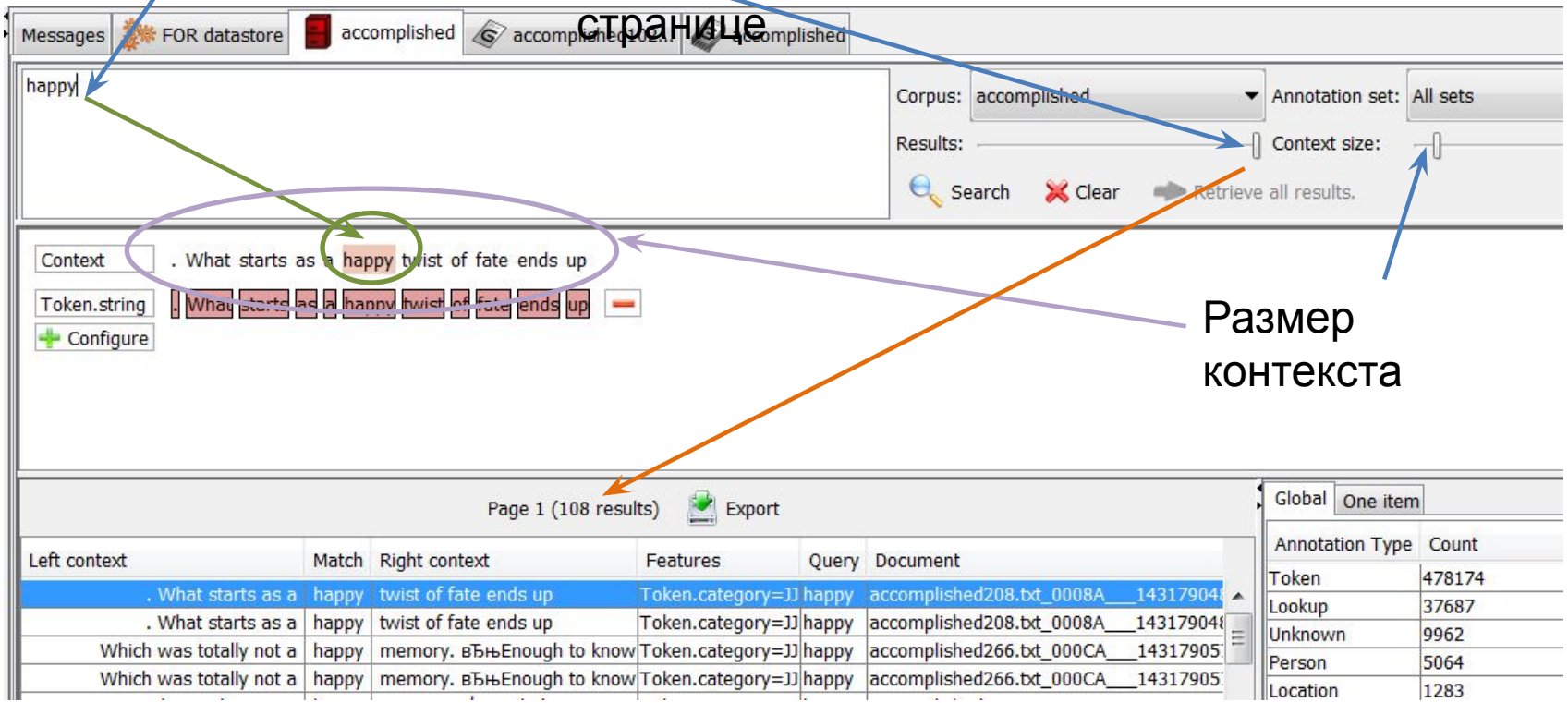
- Если корпус сохранен в одном представлении, нельзя его сохранить в другом



# Простой поиск

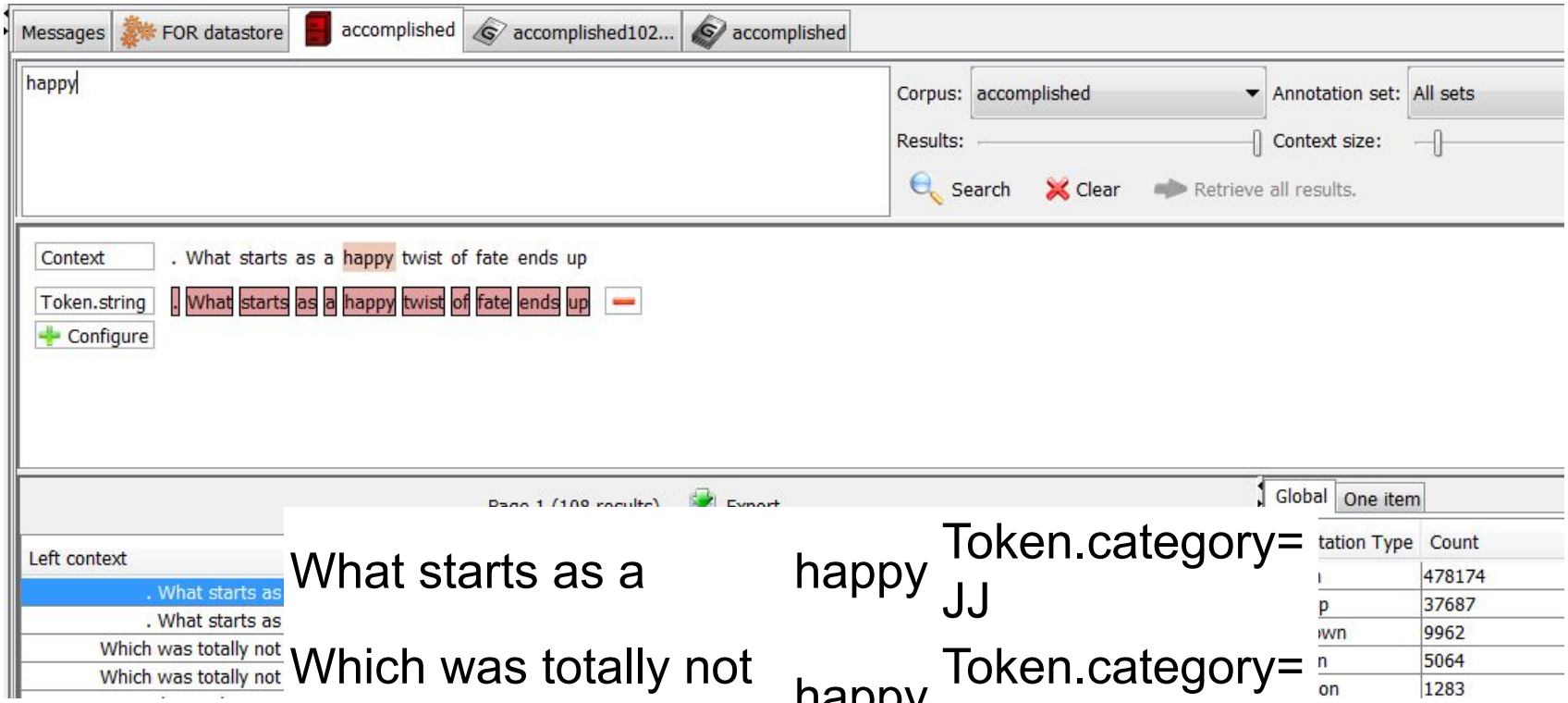
Слово или фраза для поиска

Сколько результатов отображать на одной странице




Размер контекста

# Что можно найти




| | | |
|---|---|---|
| What starts as a | happy | Token.category= JJ |
| Which was totally not a | happy | Token.category= JJ |
| as that makes me a | happy | Token.category= JJ |
| I am | happy | Token.category= JJ |

# Теперь чуть сложнее

- Можно задавать паттерны, как в правилах JAPE

Например

Вместо

not a happy

{Token.string=="not"}{Token=="a"}{Token=="happy"}

Или чуть шире

{Token.string=="not"}{Token=="a"}{Token==“JJ"}

Или еще шире

{Token.category=="RB"}({Token.category=="DT"})?{Token.category==“JJ"}

# Можно экспортировать результаты

## Annic Results and Statistics

### Parameters

- Corpus: **Entire datastore**
- Annotation set: **All sets**
- Query Issued: **{Token.category=="JJ"}**
- Context Window: **5**

### Results

| Left context | Match | Right context | Features | Query | Document |
|---|---|---|---|---|---|
| on and give it a | read-through | . I'm extraordinarily proud | Token.category=JJ | {Token.category=="JJ"} | accomplished118.txt_00026___143 |
| of it. In a | few | years, we can do | Token.category=JJ | {Token.category=="JJ"} | accomplished118.txt_00026___143 |
| on and give it a | read-through | . I'm extraordinarily proud | Token.category=JJ | {Token.category=="JJ"} | accomplished118.txt_00026___143 |
| of it. In a | few | years, we can do | Token.category=JJ | {Token.category=="JJ"} | accomplished118.txt_00026___143 |
| comm. I'm a | new | icon maker, so I | Token.category=JJ | {Token.category=="JJ"} | accomplished131.txt_00035___143 |

# Применение в лабораторной работе 8

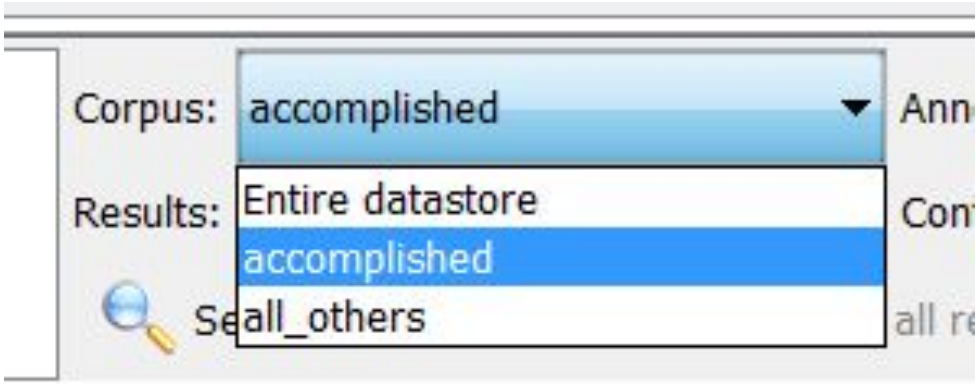
1. Проанализировать частоты встречаемости прилагательных, глаголов и т.д.
2. Проанализировать частоты встречаемости грамматических паттернов в выбранной категории настроений
3. Попытаться выявить паттерны характеризующие настроение (i feel happy,  oh so happy)
4. Определить частоты встречающихся слов
5. Определить наиболее  информативные слова по Mutual Information Criteria
6. Определить наиболее информативные паттерны
7. Выявить наиболее информативные паттерны учитывающие  содержание  слов

# Mutual Information Criteria

| | $C_{happy}$ | $C_{all\ other}$ | N |
|---|---|---|---|
| $W_{happy=1}$ | $N_{11}$ | $N_{10}$ | $N_1$ |
| $W_{happy=0}$ | $N_{01}$ | $N_{00}$ | $N_0$ |

$$MI = \frac{N_{11}}{N}\log_2\frac{NN_{11}}{N_{1.}N_{.1}} + \frac{N_{01}}{N}\log_2\frac{NN_{01}}{N_{0.}N_{.1}} + \frac{N_{10}}{N}\log_2\frac{NN_{10}}{N_{1.}N_{.0}} + \frac{N_{00}}{N}\log_2\frac{NN_{00}}{N_{0.}N_{.0}}$$

- Делаем два корпуса БД – выбранное настроение и все другие
- Тогда можно выбрать и получить частоты



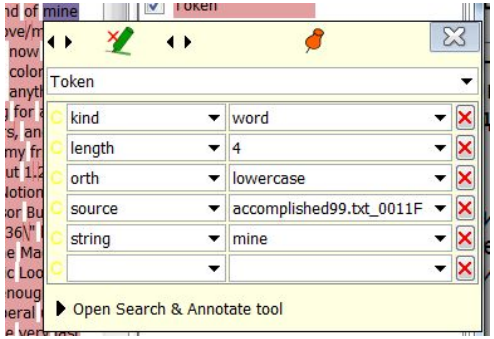
# Расчет MIC

- Встречаемость считается по всей коллекции, а для расчета MIC требуется знать в скольких документах встретилось искомое

1. способ – использовать данные из файла экспорта

| Features | Query | Document | Annotation |
|---|---|---|---|
| Token.category=JJ | {Token.category=="JJ"} | accomplished118.txt_00026___1431790404480___5714 | Key |

2. способ – поместить метку документа, потом посчитать кол-во неповторяющихся меток




3. способ написать программу на JAVA с использованием средств GATE

# Код для добавления метки документа

```
Phase: firstpass
Input: Token
Options: control = appelt
Rule: AddDocName
(   {Token}  ): t_Token
-->  {
AnnotationSet AS_Token = bindings.get("t_Token");
Annotation A_Token = AS_Token.iterator().next();
FeatureMap newAnnFeatures = Factory.newFeatureMap();
newAnnFeatures.putAll(A_Token.getFeatures()); // перезаписываем все свойства
newAnnFeatures.put("source", doc.getName()); // добавляем новое - source
inputAS.remove(A_Token); // Убираем Token которые мы изменяли, чтобы потом добавить
outputAS.add(AS_Token.firstNode(), AS_Token.lastNode(), "Token",  newAnnFeatures); // Добавляем разметку Token
}
```