Spring Data Postроитель

Evgeny Borisov

План доклада

Как написать (регистрация бинов из несуществующих классов)

План доклада

Как написать (регистрация бинов из несуществующих классов)

Где написать (Registrar? AppCtxInitializer? ContextRefresh?)

План доклада

Как написать (регистрация бинов из несуществующих классов)

Где написать (Registrar? AppCtxInitializer? ContextRefresh?)

Как написать красивый код (без спринга?)

План доклада

Как написать (регистрация бинов из несуществующих классов)

Где написать (Registrar? AppCtxInitializer? ContextRefresh?)

Как написать красивый код (без спринга?)

Раскрыть секрет Алименкова (ленивые коллекции)

План доклада

Как написать (регистрация бинов из несуществующих классов)

Где написать (Registrar? AppCtxInitializer? ContextRefresh?)

Как написать красивый код (без спринга?)

Раскрыть секрет Алименкова (ленивые коллекции)

А как это всё сделано у реальных пацанов? (что отличается от нашего кода)

Спасибо Гене, который потрошил Spring Data

Как писать

Есть интерфейс с методами:

findByNameContainsSortByAge(String partOfName)

findByAgeGreaterThan(String age)

Как писать

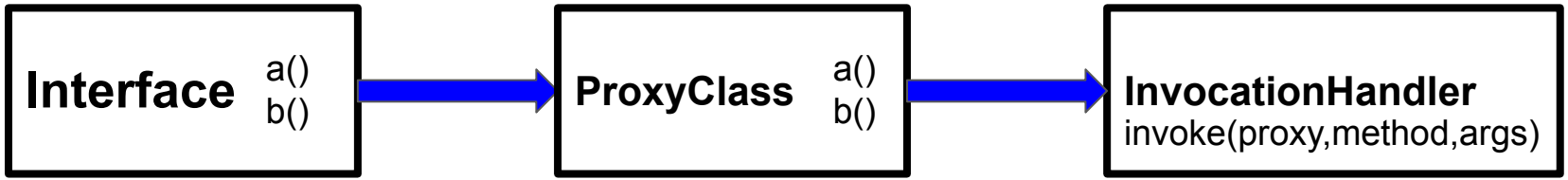
Есть интерфейс с методами:

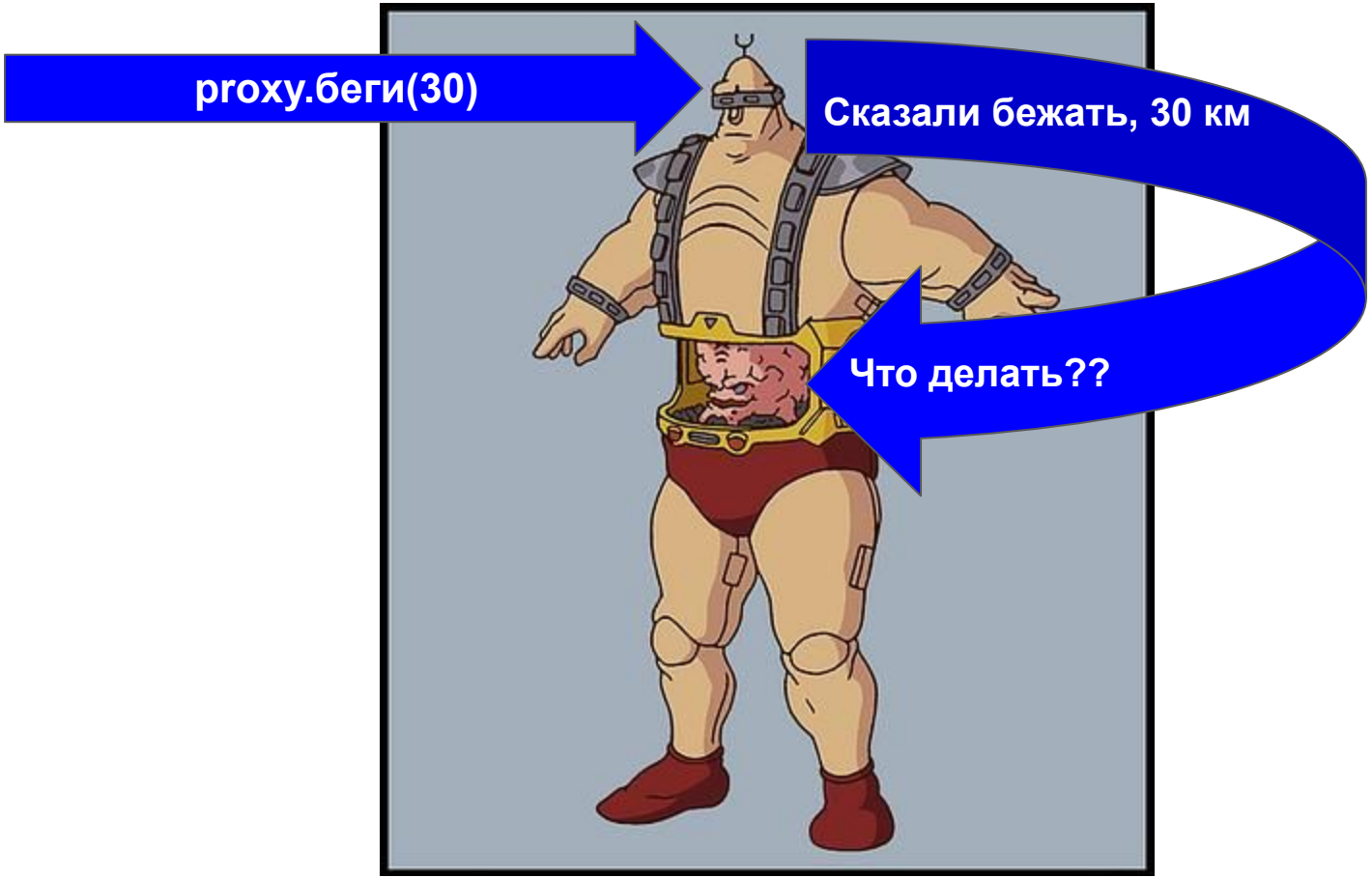
findByNameContainsSortByAge(String partOfName)

findByAgeGreaterThan(String age)

Надо сгенерить класс (dynamic proxy pattern)

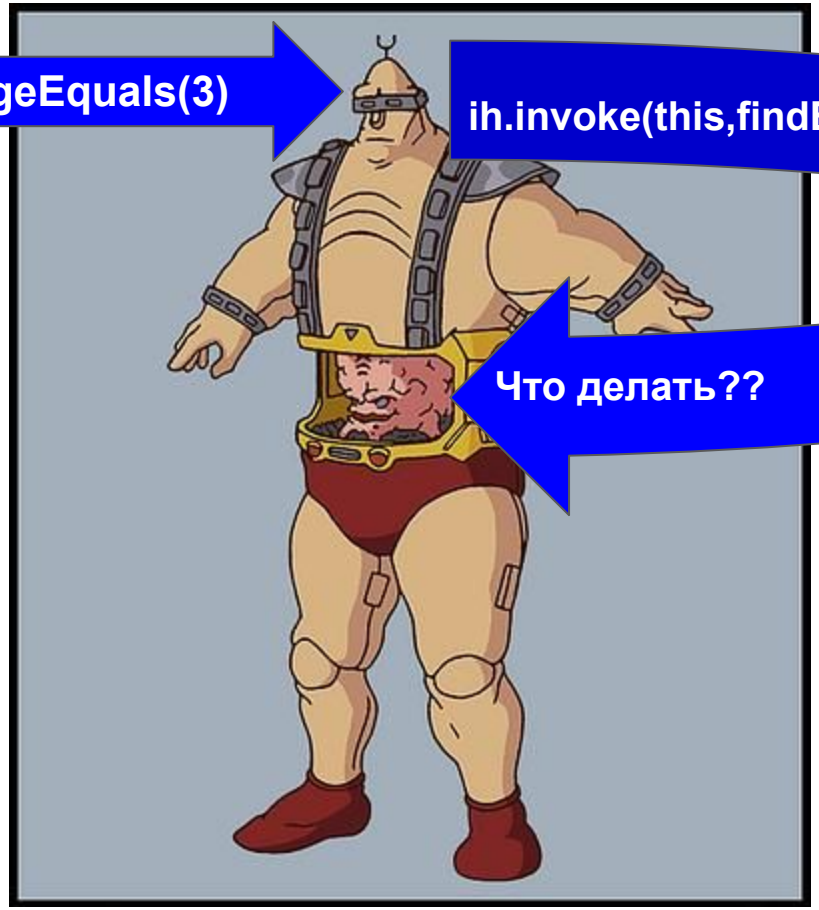
Как работает dynamic proxy?




proxy.беги(30)
Сказали бежать, 30 км
Что делать??


proxy.стой()
Сказали стоять
Что делать??

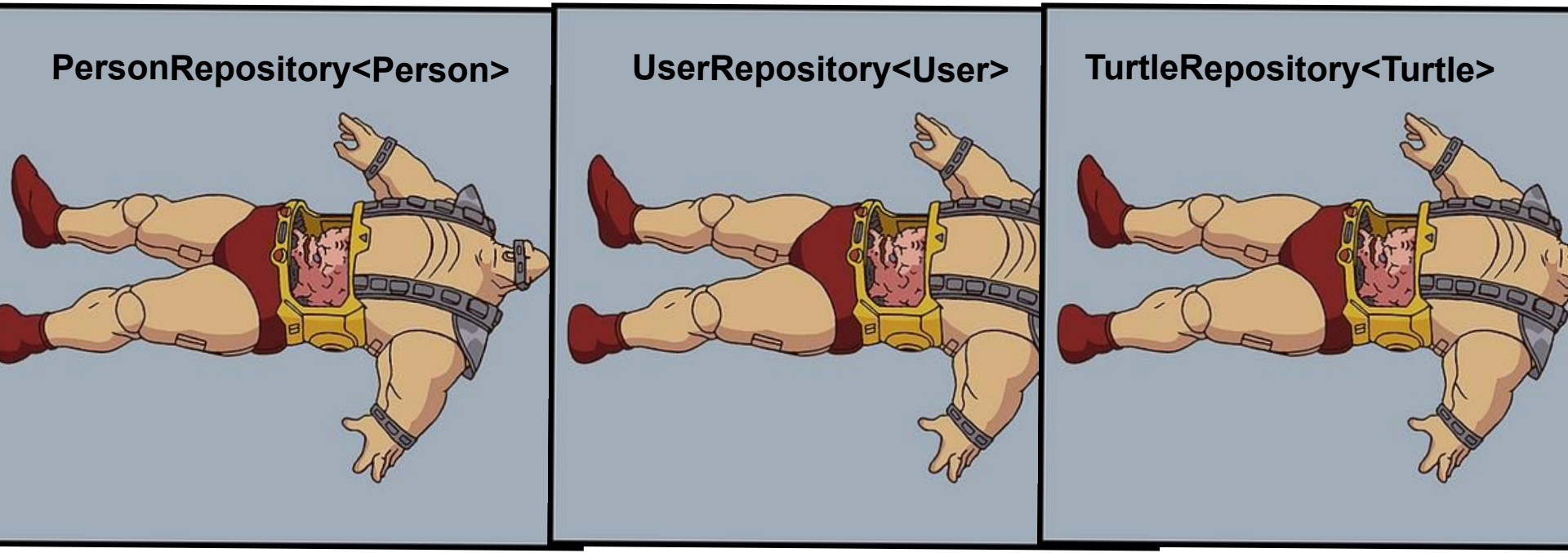
proxy.findByAgeEquals(3)



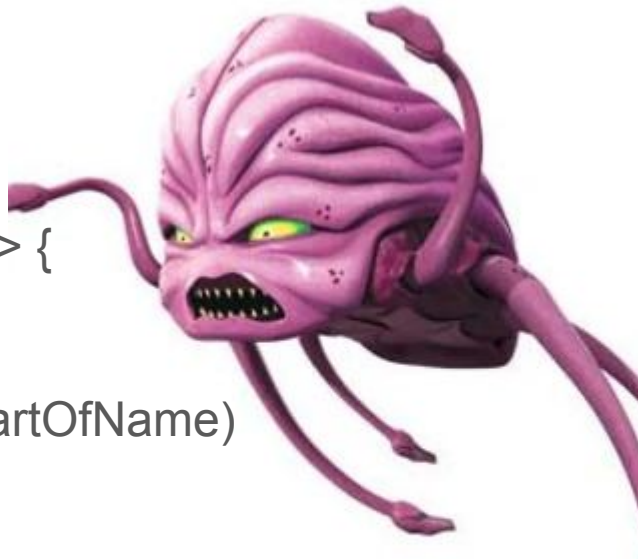
ih.invoke(this,findByAgeEquals(),{3})

Что делать??

Даёшь на каждый repository interface по кренгу?



SparkInvocationHandler должен уметь



```
interface PersonRepo extends SparkRepository<Person> {


  List<Person> findByNameContainsSortByAge(String partOfName)
  List<Person> findByAgeGreaterThan(String age)
  long count()
  long findByAgeGreaterThanCount(String age)
  long findByAgeGreaterThanSave(String age)
  …
}
```

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

SparkSession?

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

SparkContext?

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

JavaSparkContext?

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

JavaSparkContext + SparkSession?

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

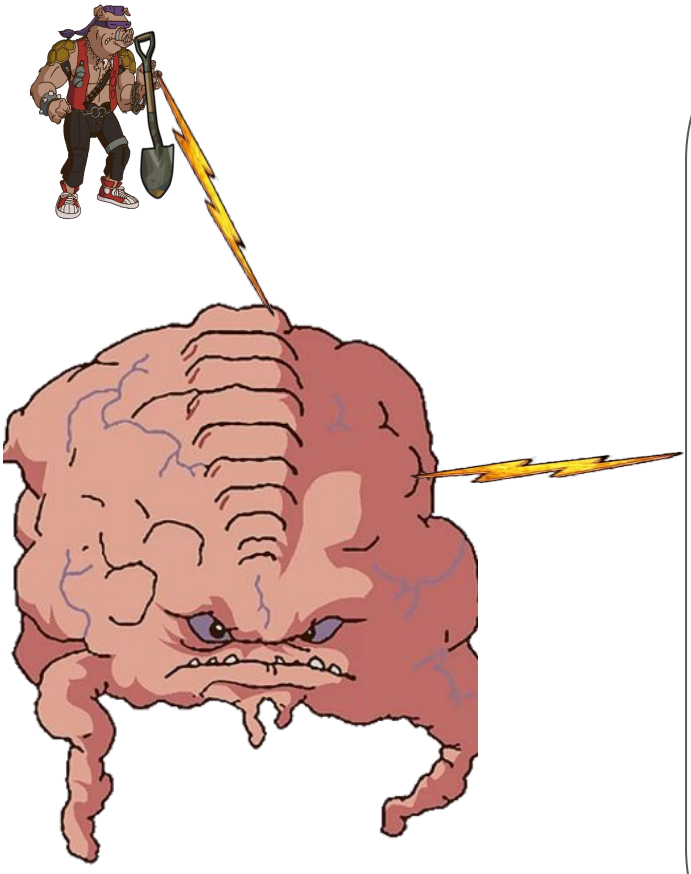
Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

~~JavaSparkContext + SparkSession?~~ ApplicationContext (сами берите что надо)

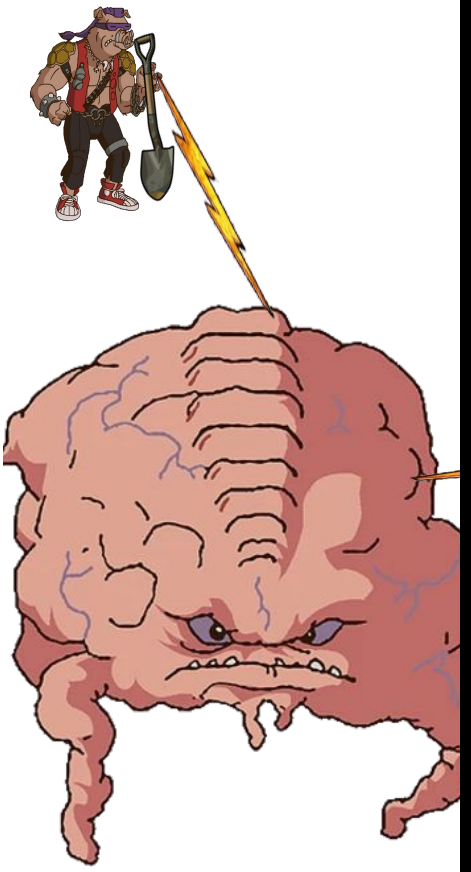


findByNameAnd...Order...

findByNameOrder...

count()

Empty List




А вам слабо ?

SparkInvocationHandler что имеет?

Класс модели (Одна штука)

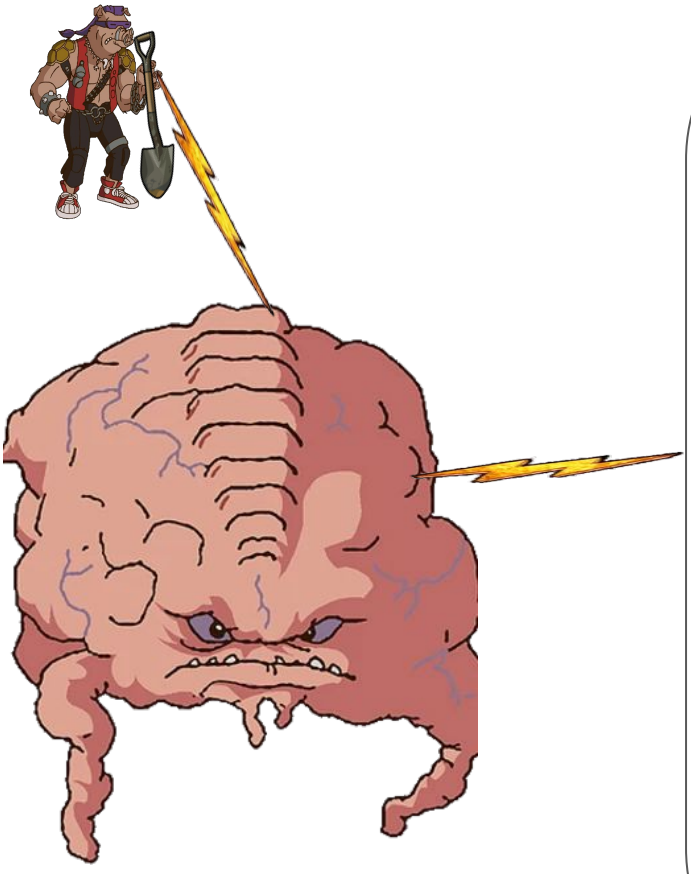
Ссылка на данные для данной модели (одна штука)

DataExtractor (Одна штука)

Трансформирмации (у каждого метода свой список)

Терминальная операция (у каждого метода свой)

~~JavaSparkContext + SparkSession?~~ ApplicationContext (сами берите что надо)

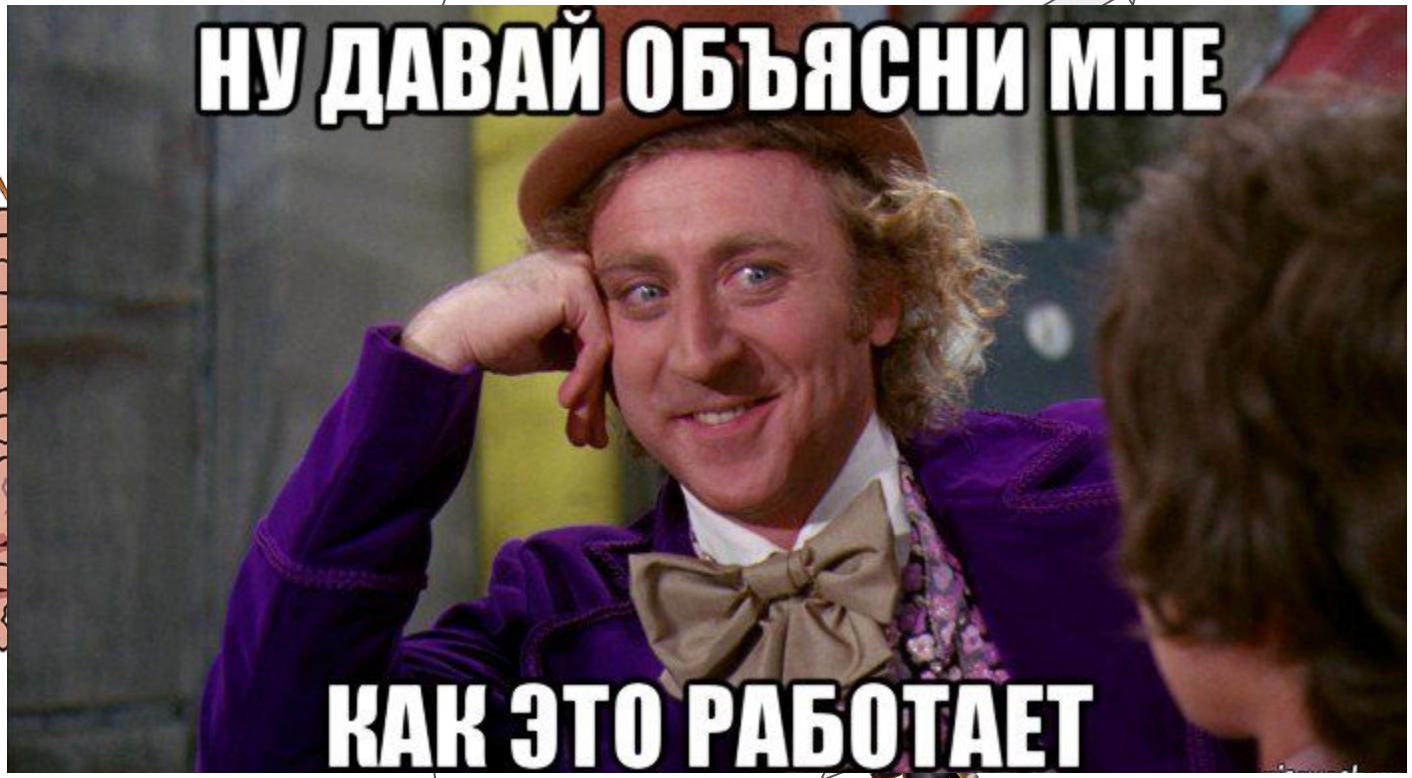


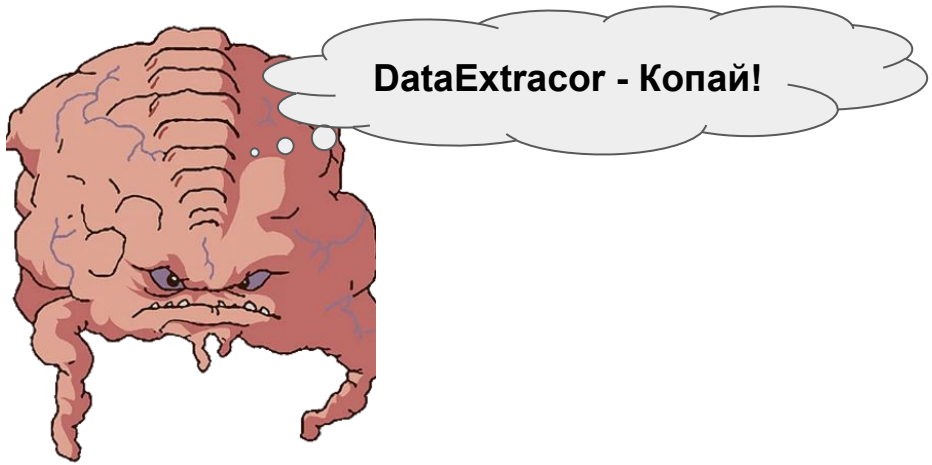
findByNameAnd...Order...

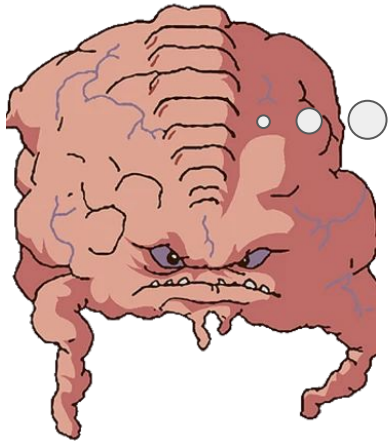
findByNameOrder...

count()

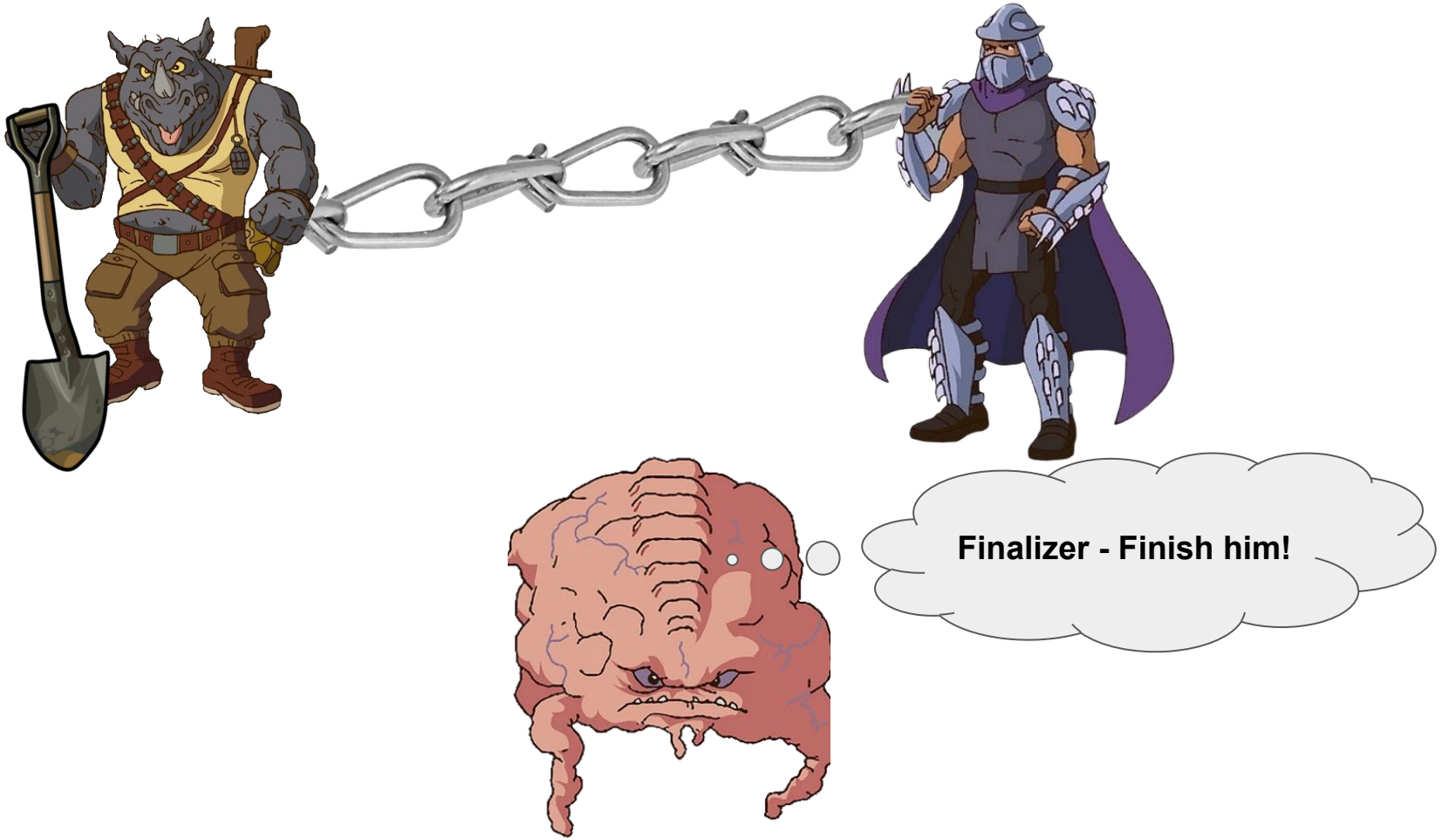
Empty List


НУ ДАВАЙ ОБЪЯСНИ МНЕ
КАК ЭТО РАБОТАЕТ


DataExtracor - Копай!



TransformationChain
Трансформируй!


Finalizer - Finish him!

Нужно сделать:

Создать инфраструктуру для спарка (зарегистрировать в контекст)

Просканировать пакеты, найти наследников SparkRepository,
Нагенерить под них кранков (зарегистрировать в контест)

Когда? (BeanDefinition или Bean)

Bean vs BeanDefinition в одном слайде

Рецепт приготовления боба

Взять боб из класса зеленые бобы
Добавить в него соль
Добавить в него перец
Положить в кастрюлю с водой
Начать варить, если кто-то попросит

BeanDefenitionRegistrar?

ContextRefreshedEvent?

ApplicationContextInitializer?

Registrar

- Для чего?
- Когда они работают и к чему у них есть доступ?
- Как их декларировать?

```
@Configuration
@Import(MyRegistrar.class)
@ComponentScan
public class Conf {
```


```
@Override
public void registerBeanDefinitions(AnnotationMetadata metadata, BeanDefinitionRegistry registry,
        BeanNameGenerator generator) {
```

BeanDefenitionRegistrar?

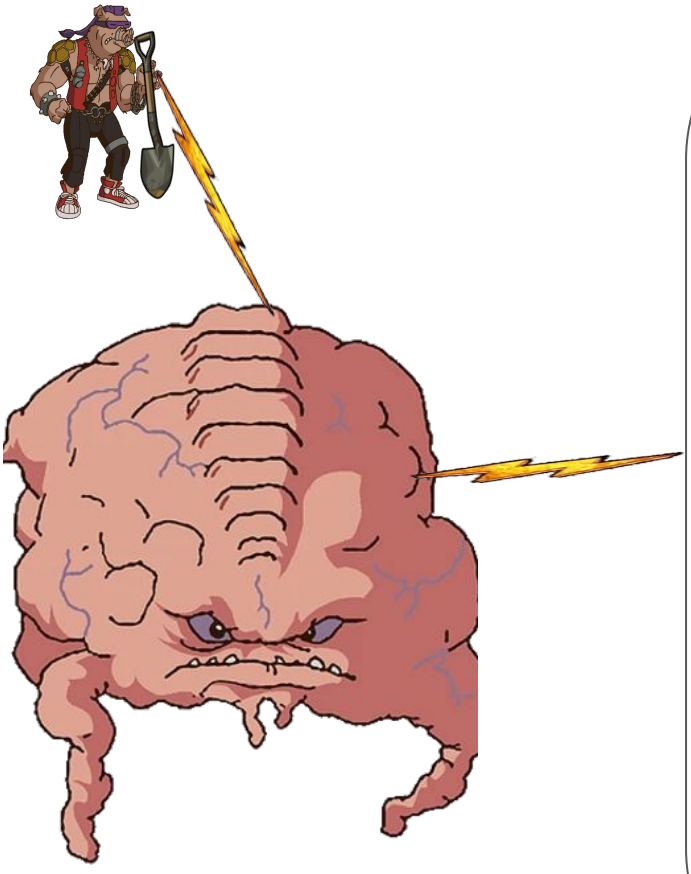
Слишком сложно

ContextRefreshedEvent?

Слишком поздно

ApplicationContextInitializer?

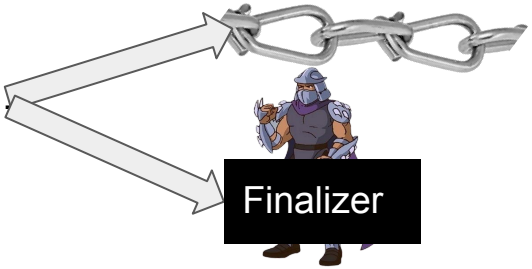
Пошли уже куда-то



findByNameAnd...Order...

Finalizer

findByNameOrder...

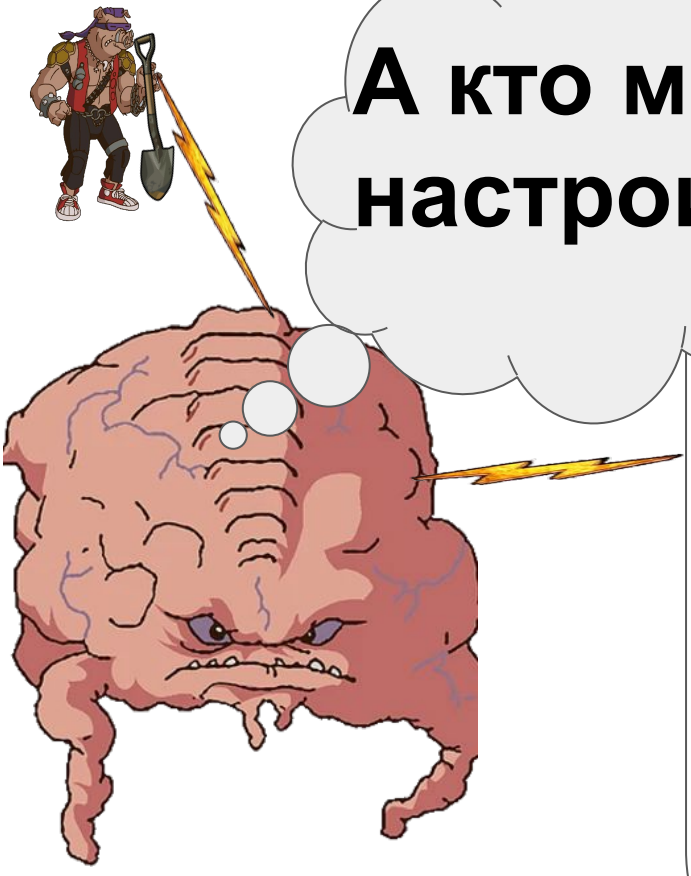


Finalizer

count()

Empty List

Finalizer



А кто меня настроит?

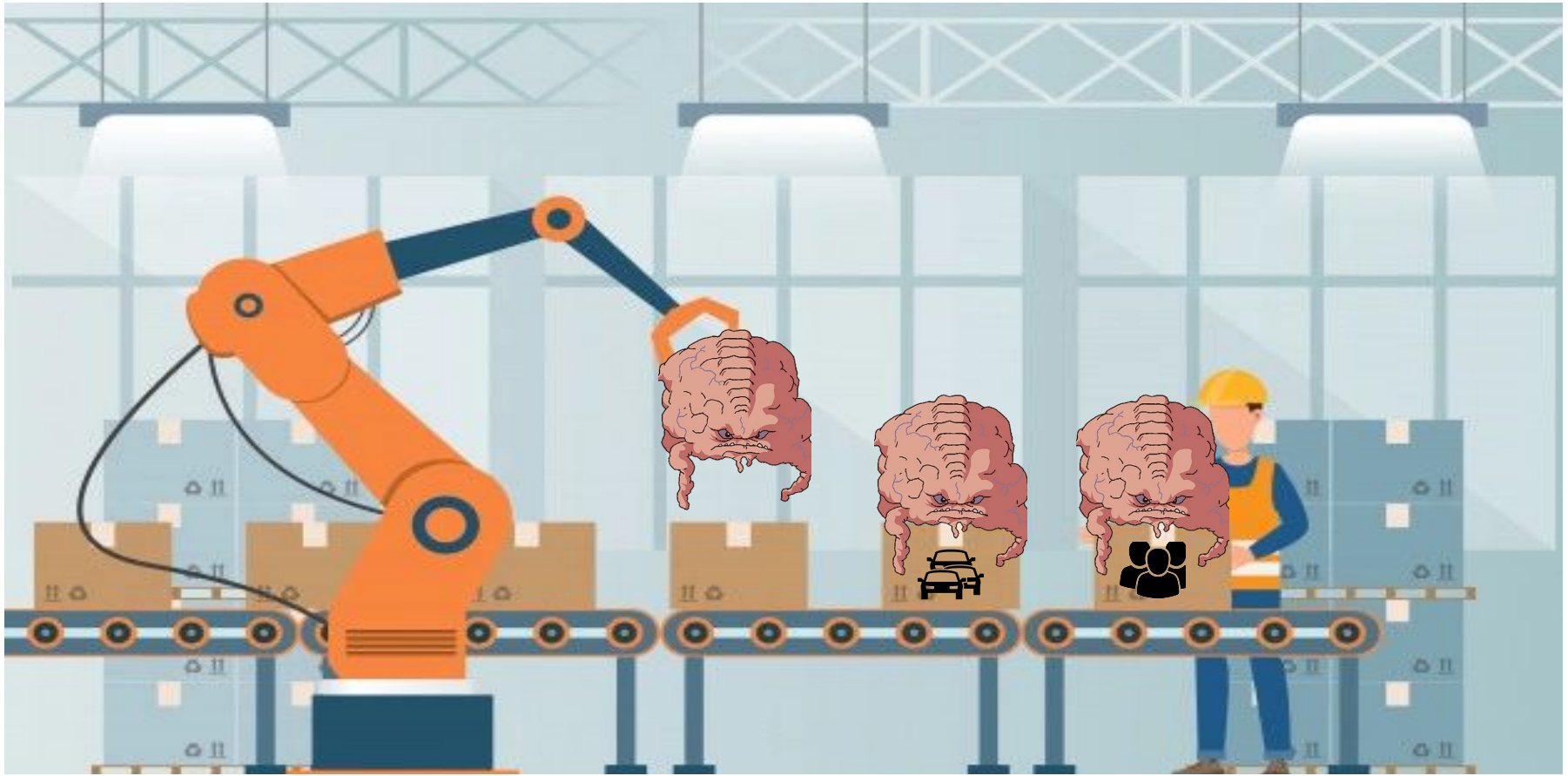
eAnd...Order...

findByNameOrder..

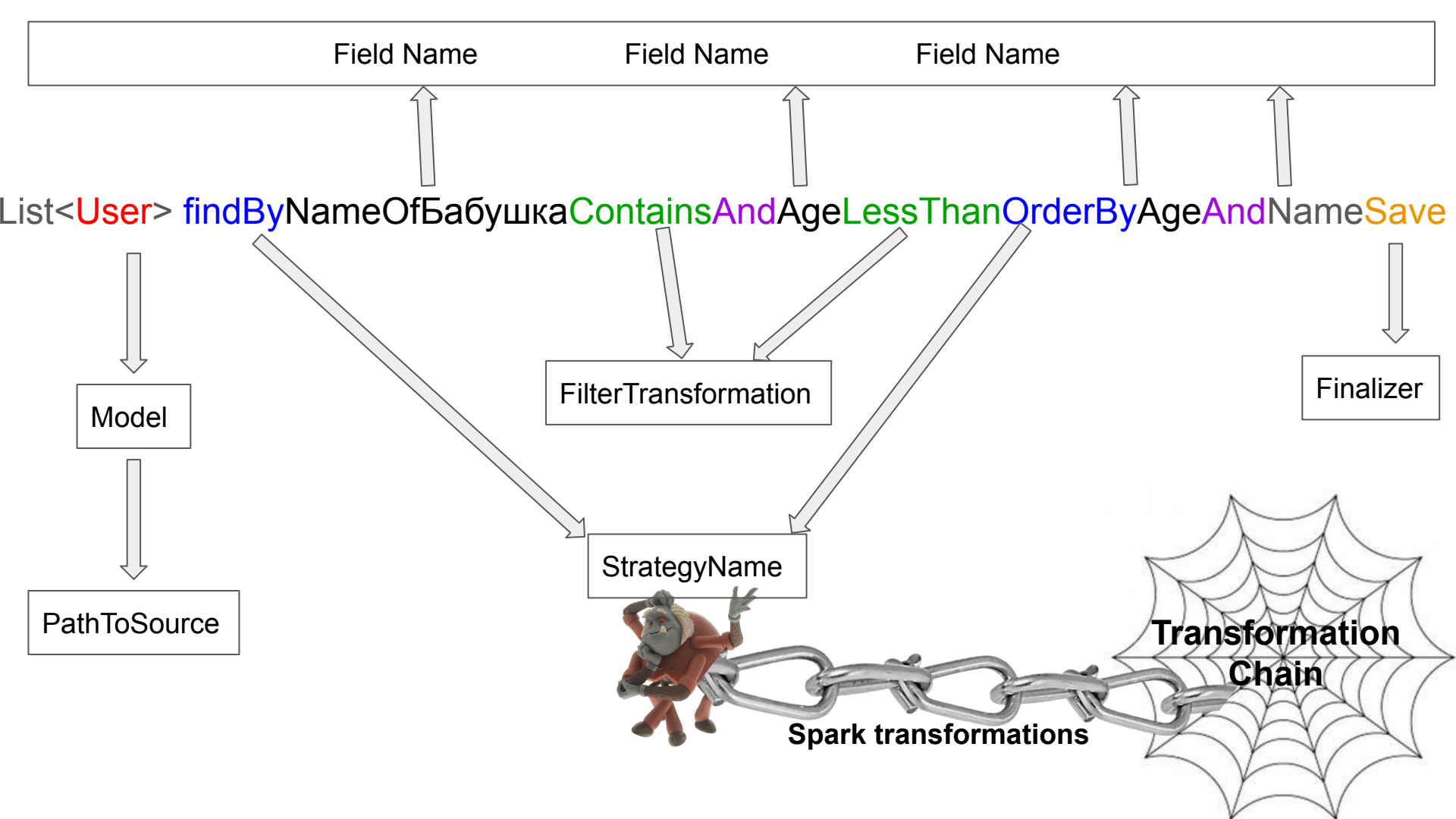
count()

Empty List

InvocationHandlerFactory



List<User> findByNameOfБабушкаContainsAndAgeLessThanSortByAgeAndNameSave


Field Name
Field Name
Field Name
List<User> findByNameOfБабушкаContainsAndAgeLessThanOrderByAgeAndNameSave
Model
PathToSource
FilterTransformation
StrategyName
Finalizer
Spark transformations
Transformation Chain

Что нужно нашей фабрике?

DataExtractors (все штуки)

Что нужно нашей фабрике?

DataExtractors (все штуки)

Finalizers (все штуки)

Что нужно нашей фабрике?

DataExtractors (все штуки)

Finalizers (все штуки)

Spiders (все штуки), которые будут отдавать SparkTransformations

Что нужно нашей фабрике?

DataExtractors (все штуки)

Finalizers (все штуки)

Spiders (все штуки), которые будут отдавать SparkTransformations

А ещё весь контекст на всякий случай

А как эту всю хрень настроить?

Как эту всю хрень настроить?


Он так любил помидоры, что ел их с кетчупом и запивал томатным соком

Хотите попробовать без спринга?

Вы делаете релокацию.
Как поступить с любимым шкафом?

1. Разберу и соберу сам
2. Позову друга или помощь зала
3. Позову специальную команду, и заплачу им денег
4. Уговорю начальство остаться в пределах бутово и перенесу шкаф сам, не разбирая
5. Выкину шкаф и куплю (сделаю) новый (только для неженатых)

Выберите правильные ответы

A) Stateless architecture means that our services don’t have any state

B) Stateless architecture means that the state of our services can’t change

C) Stateful architecture means that the state of our services can’t change

D) Stateful architecture means that the state of our services can change

Выберите правильные ответы

A) Stateless architecture means that our services doesn’t have any state

B) Stateless architecture means that the state of our services can’t change

C) Stateful architecture means that the state of our services can’t change

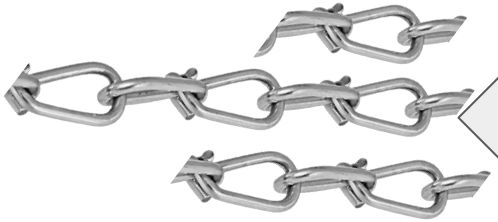
D) Stateful architecture means that the state of our services can change

InvocationHandlerFactory internals

InvocationHandlerFactory internals

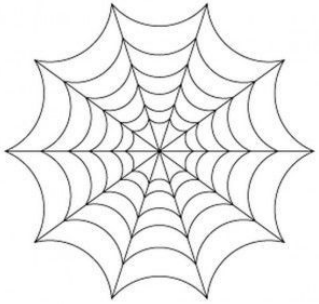
Взять экстрактор

InvocationHandlerFactory internals



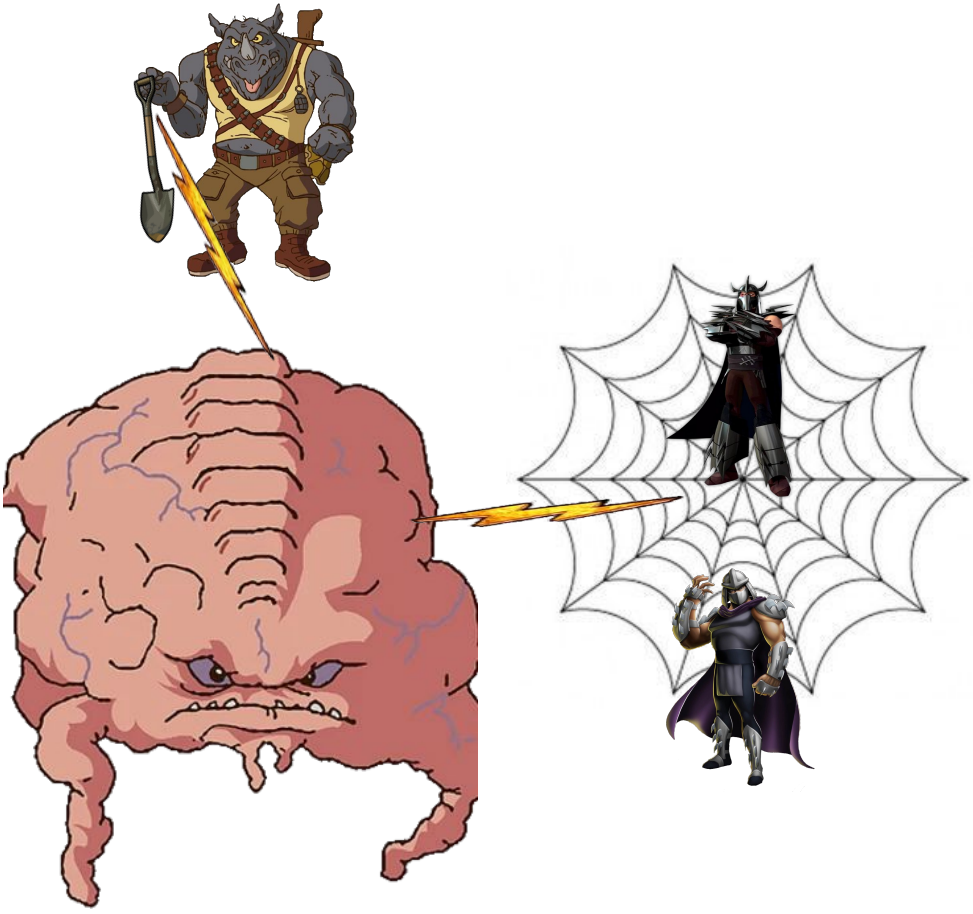
С их помощью построить цепь на каждый метод

InvocationHandlerFactory internals

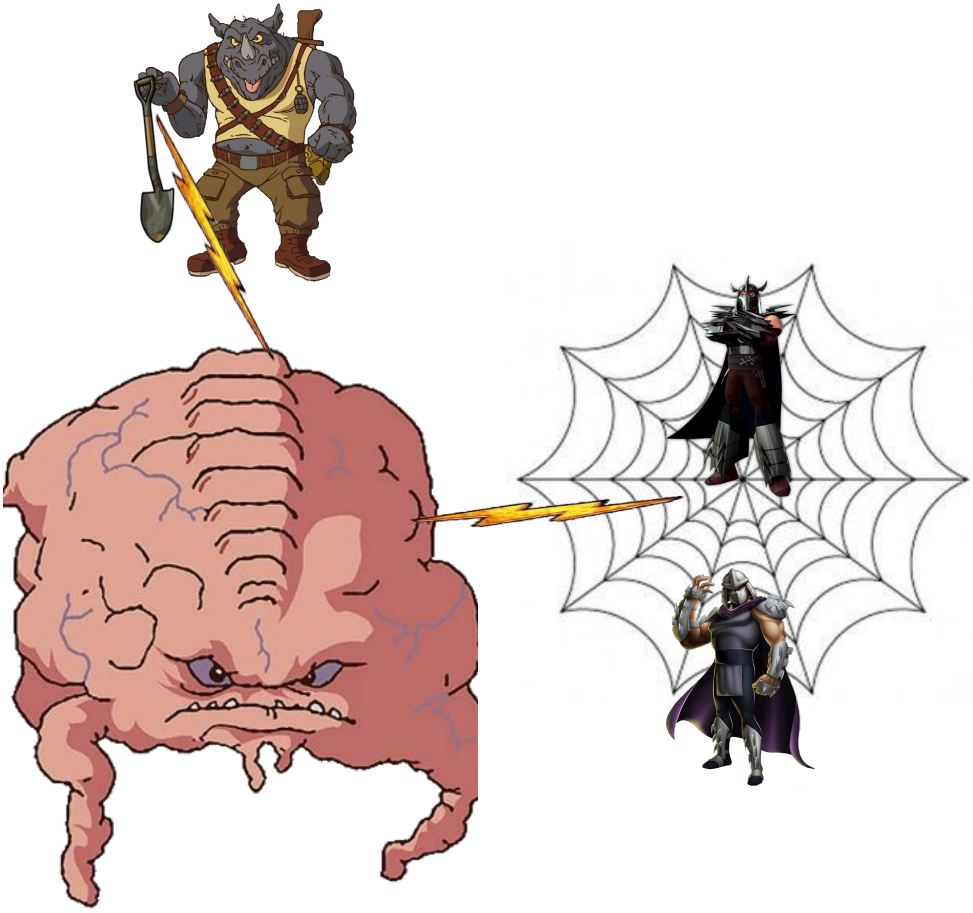


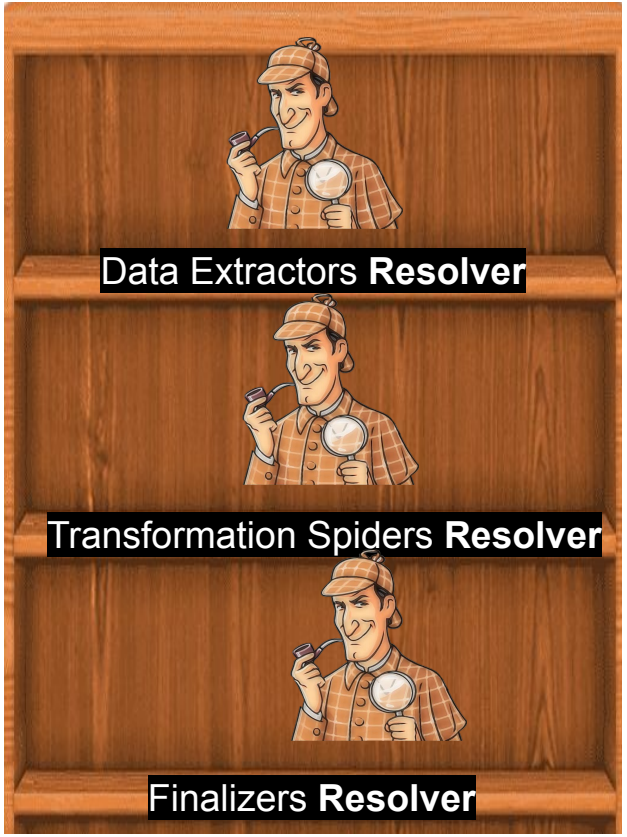
Взять файналайзер
на каждый метод

InvocationHandlerFactory internals

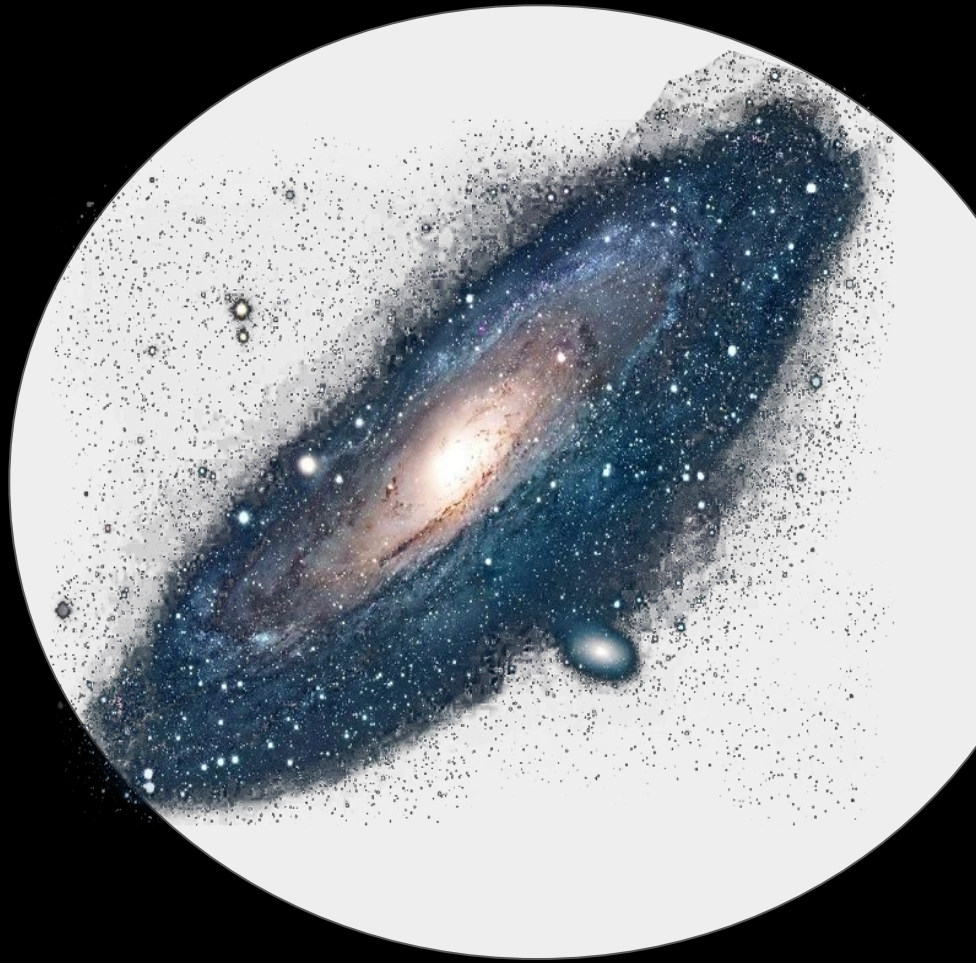


InvocationHandlerFactory internals






SpringApplication.run(MainApp.class)
ApplicationContext

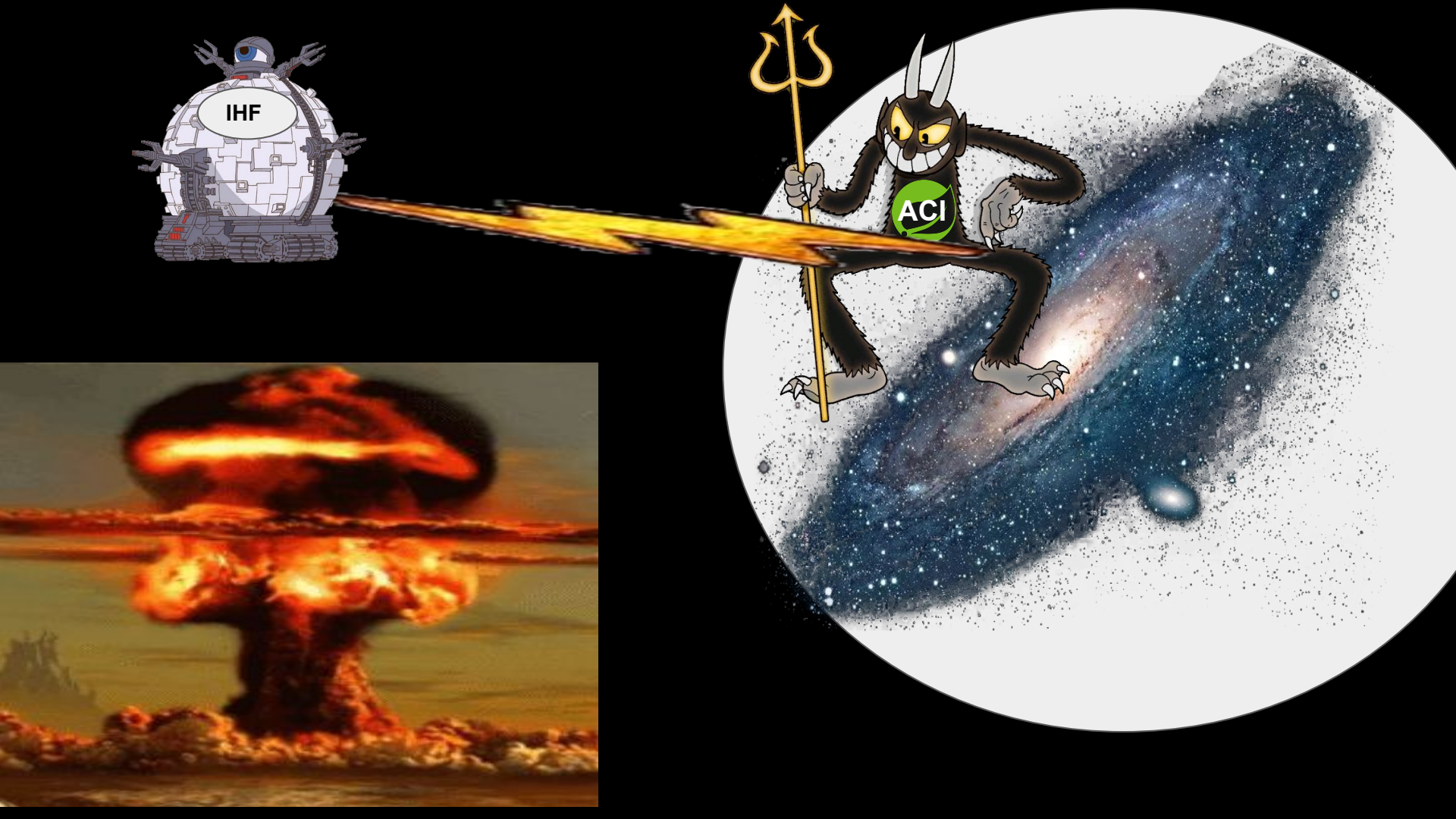


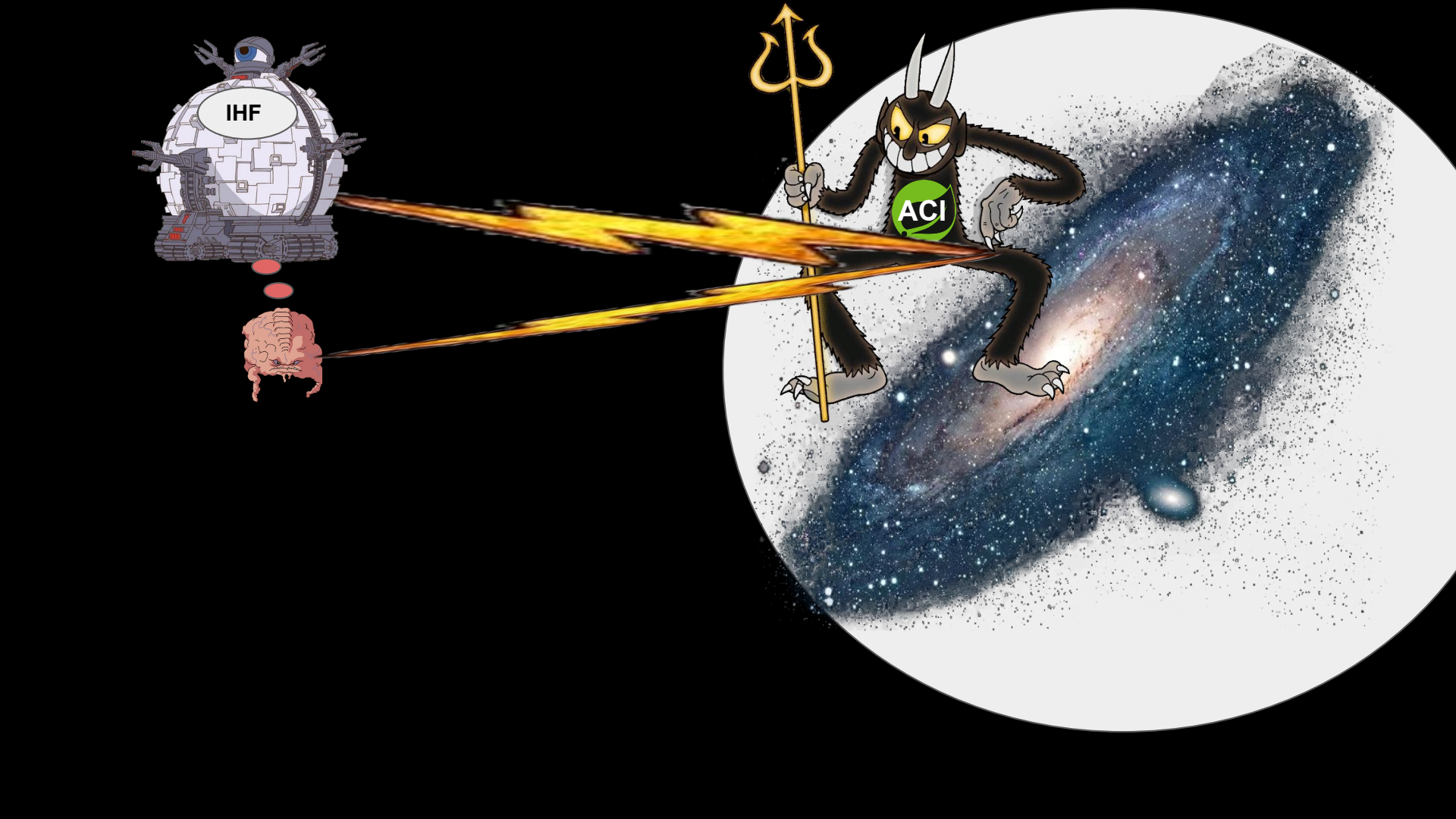

ACI

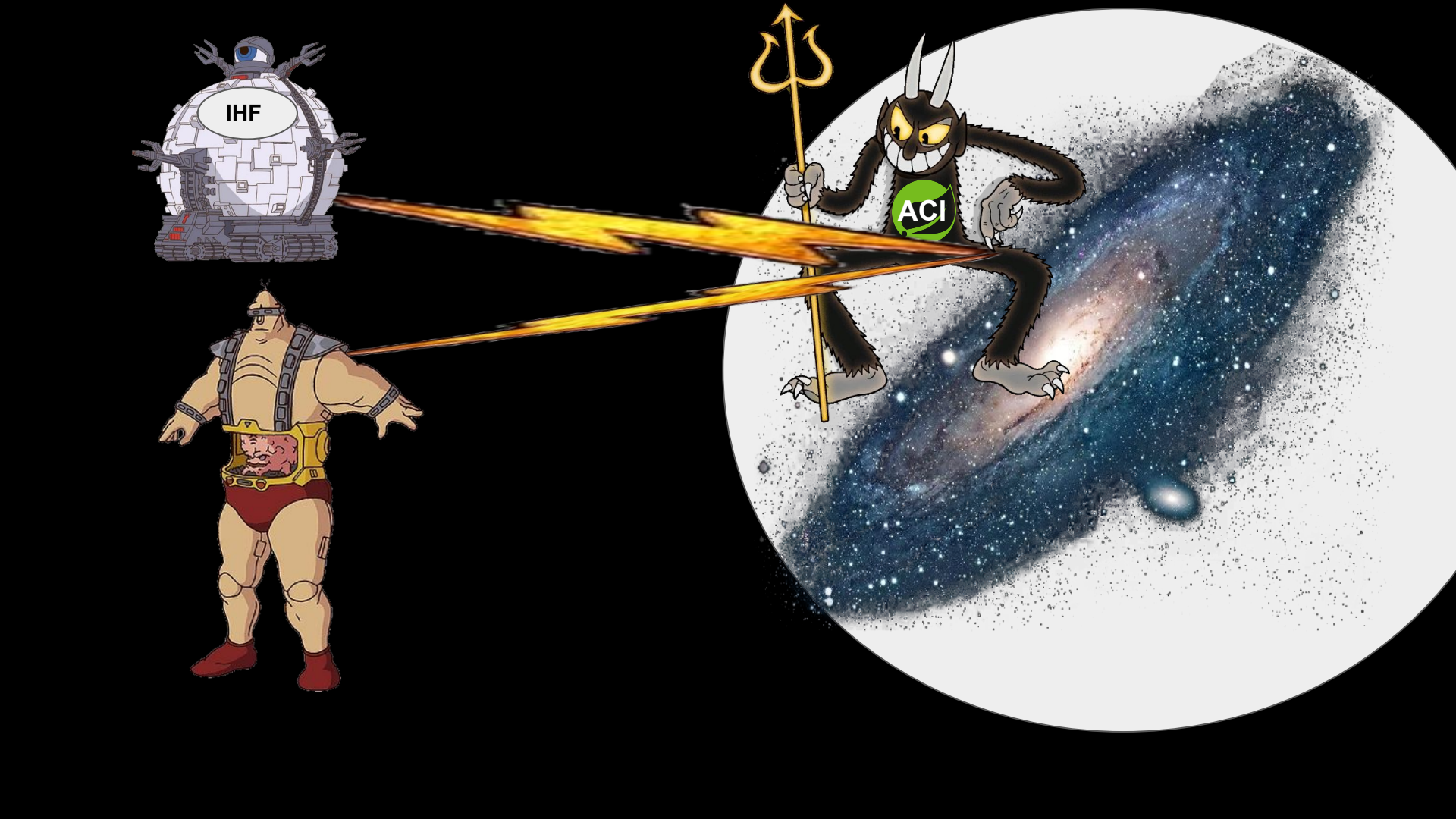

ACI

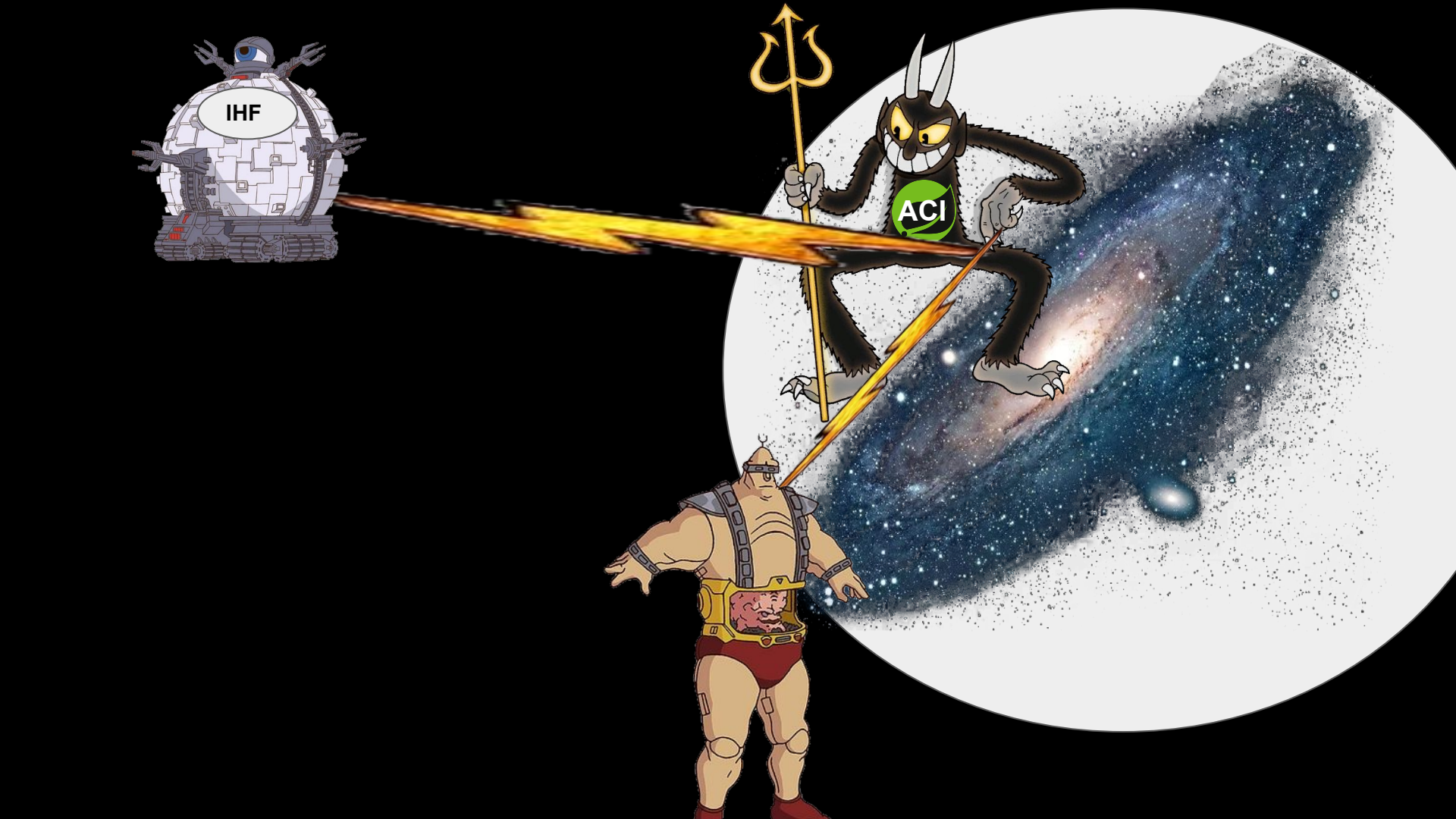

IHF
ACI


ACI

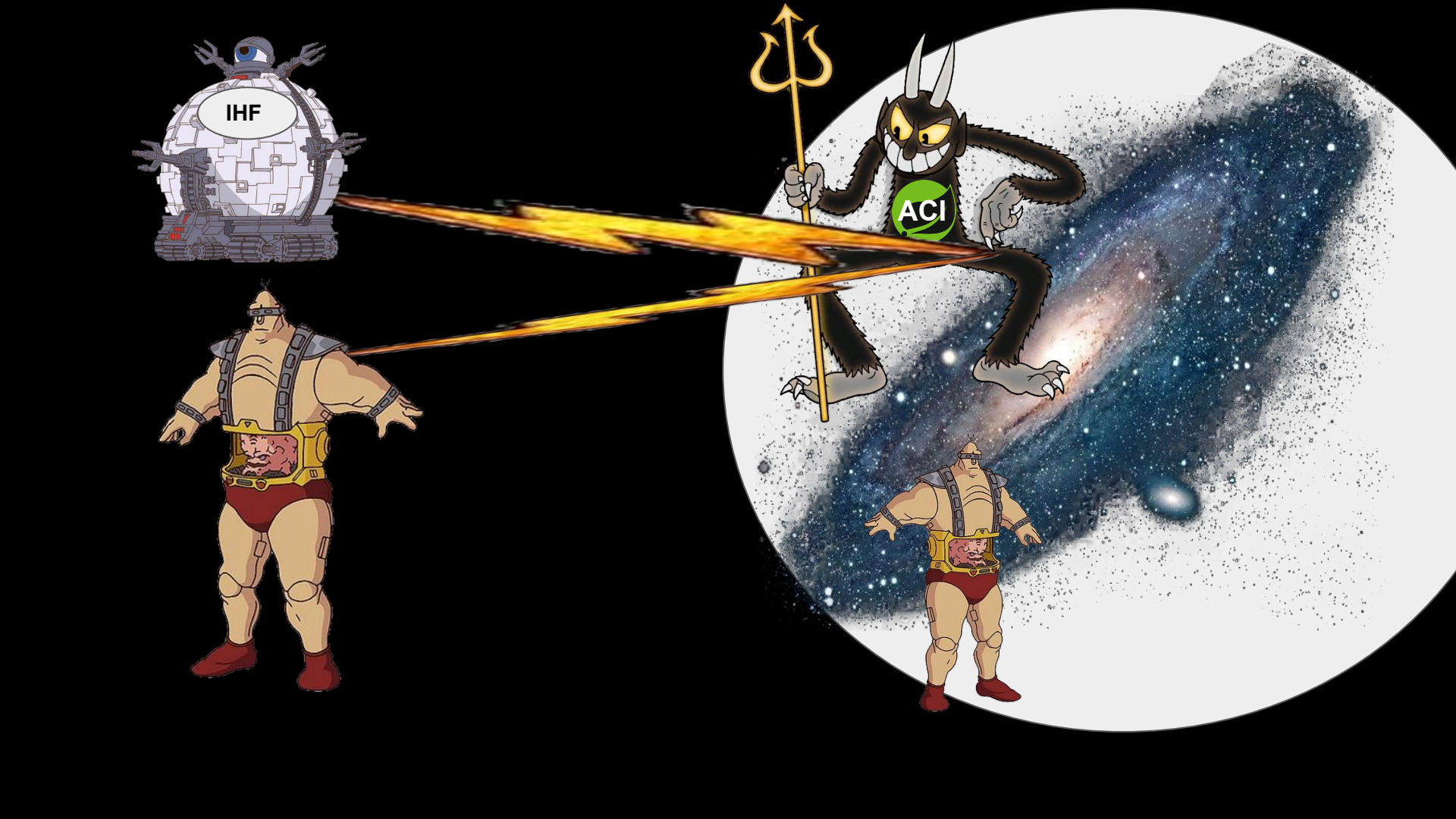

IHF
ACI

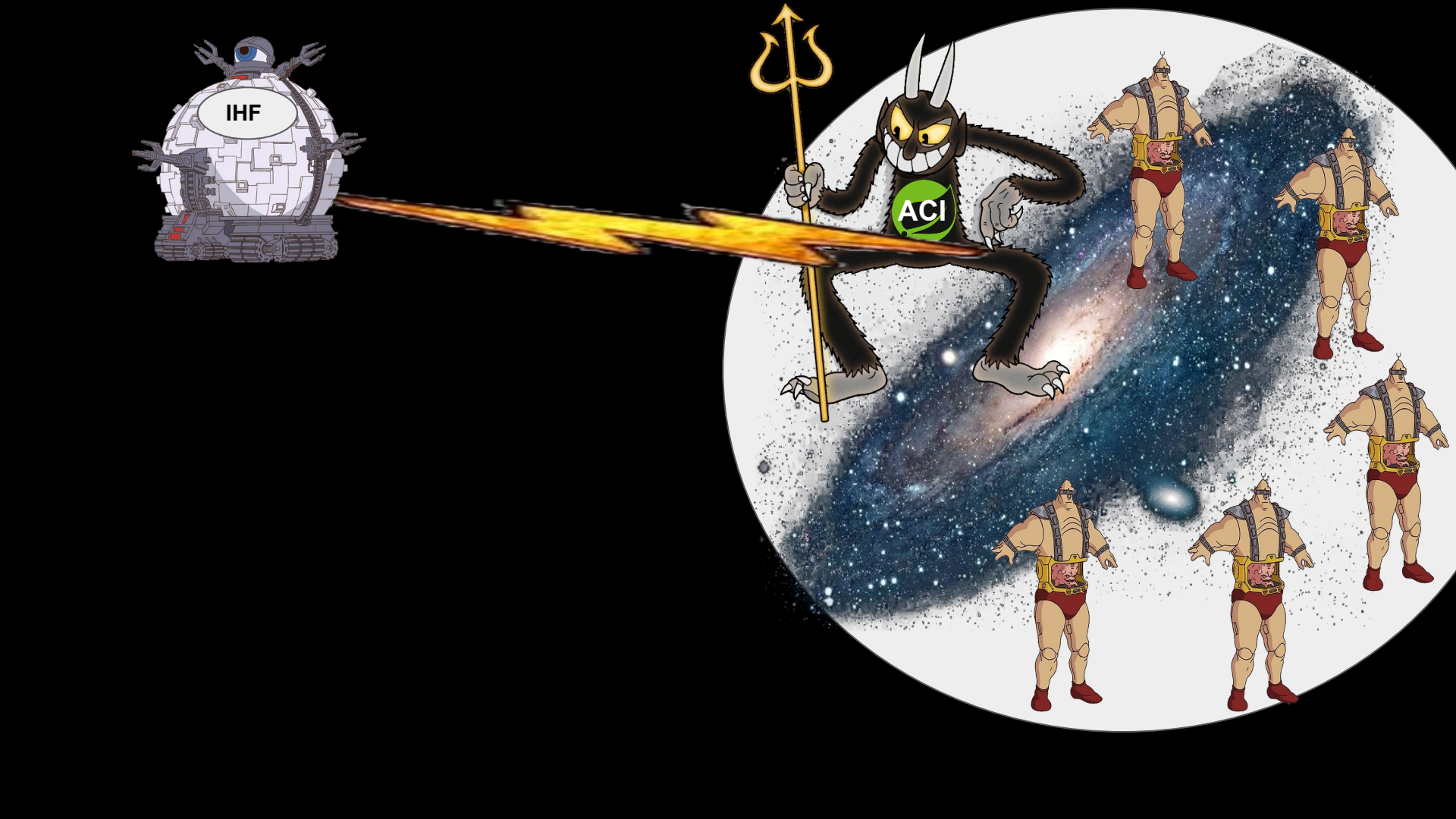

IHF
ACI

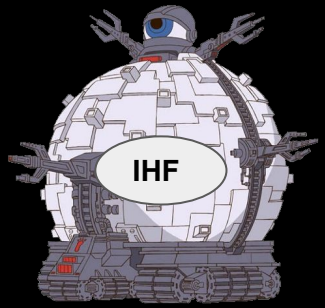

IHF
ACI

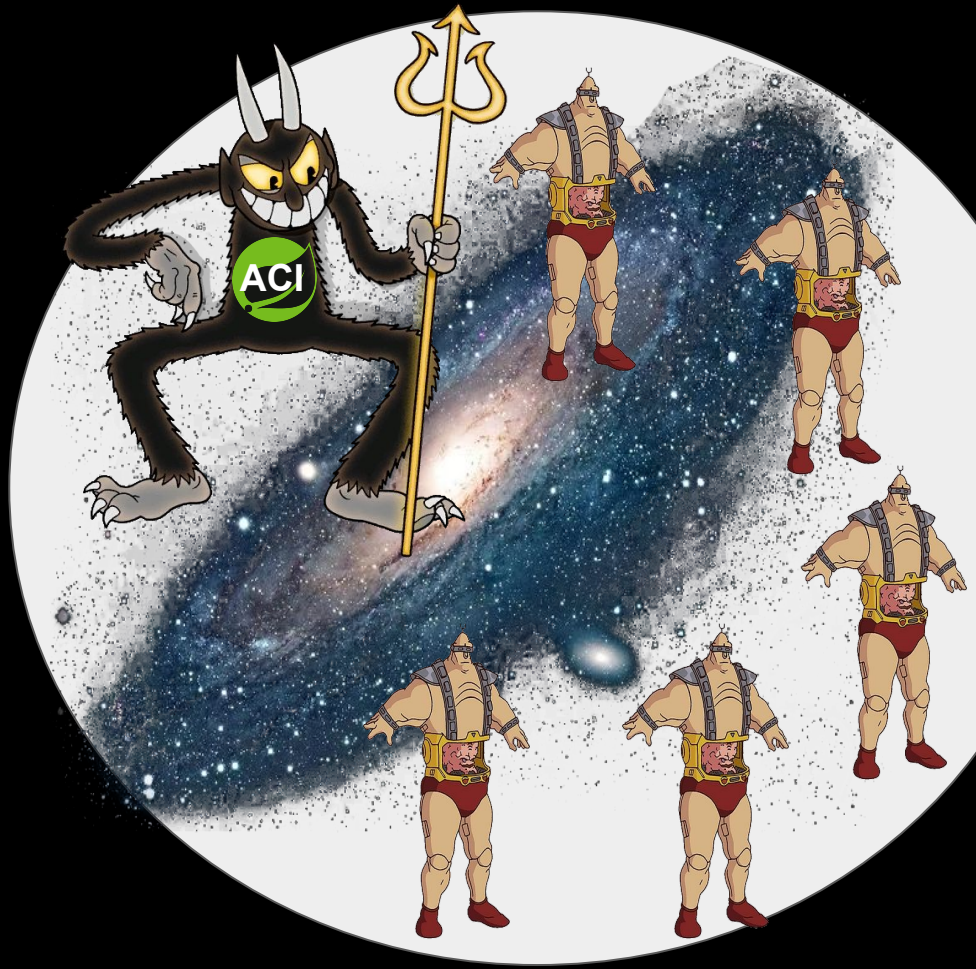

IHF
ACI

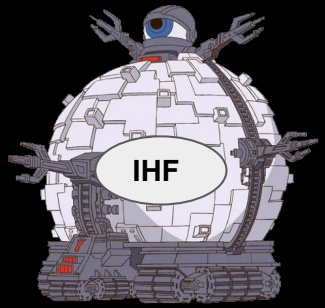

IHF
ACI


IHF
ACI


IHF
ACI

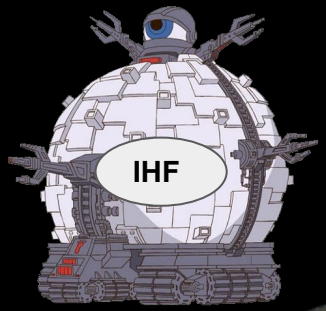

IHF

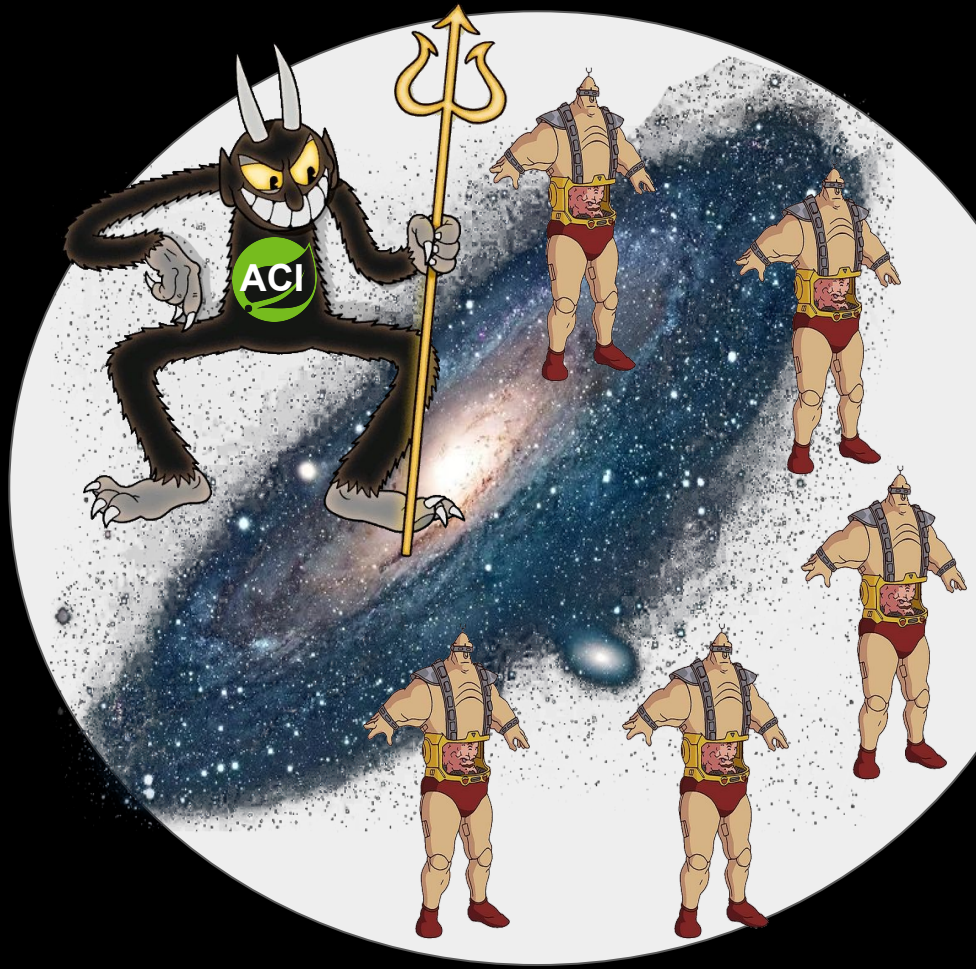

ACI


IHF

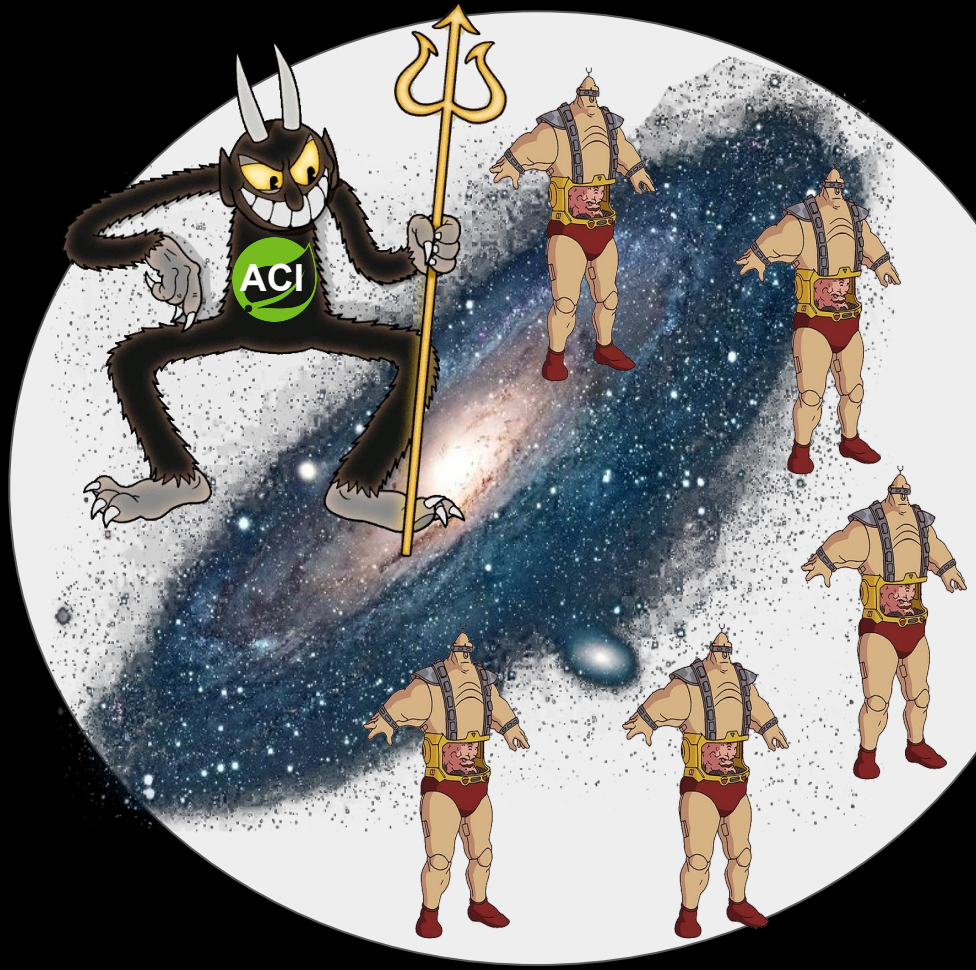

ACI


GC

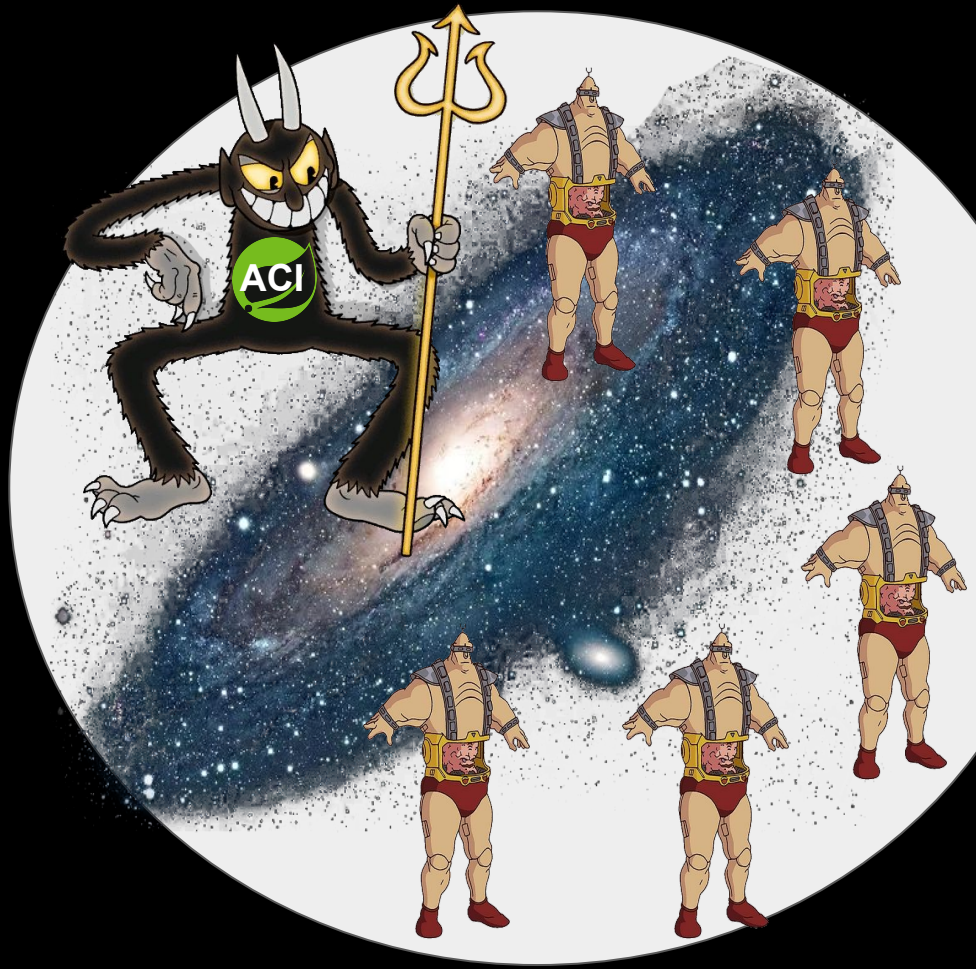

IHF


ACI

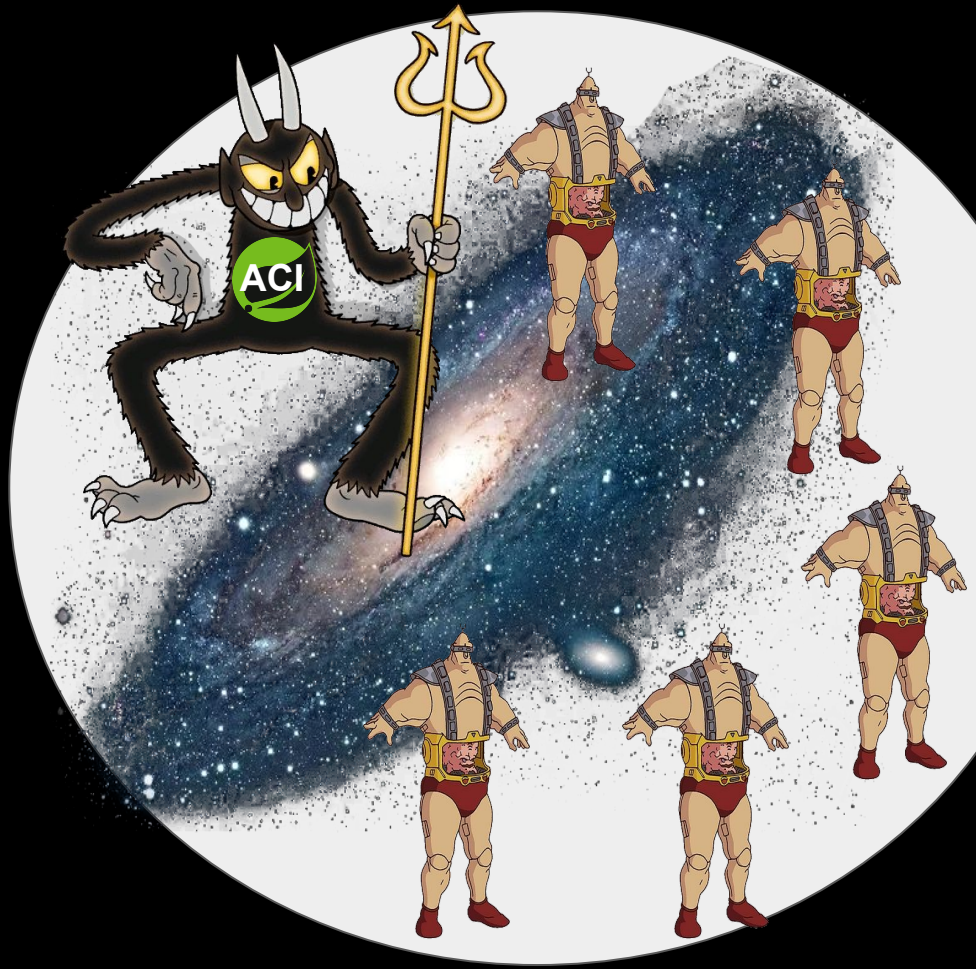

IHE
TGC

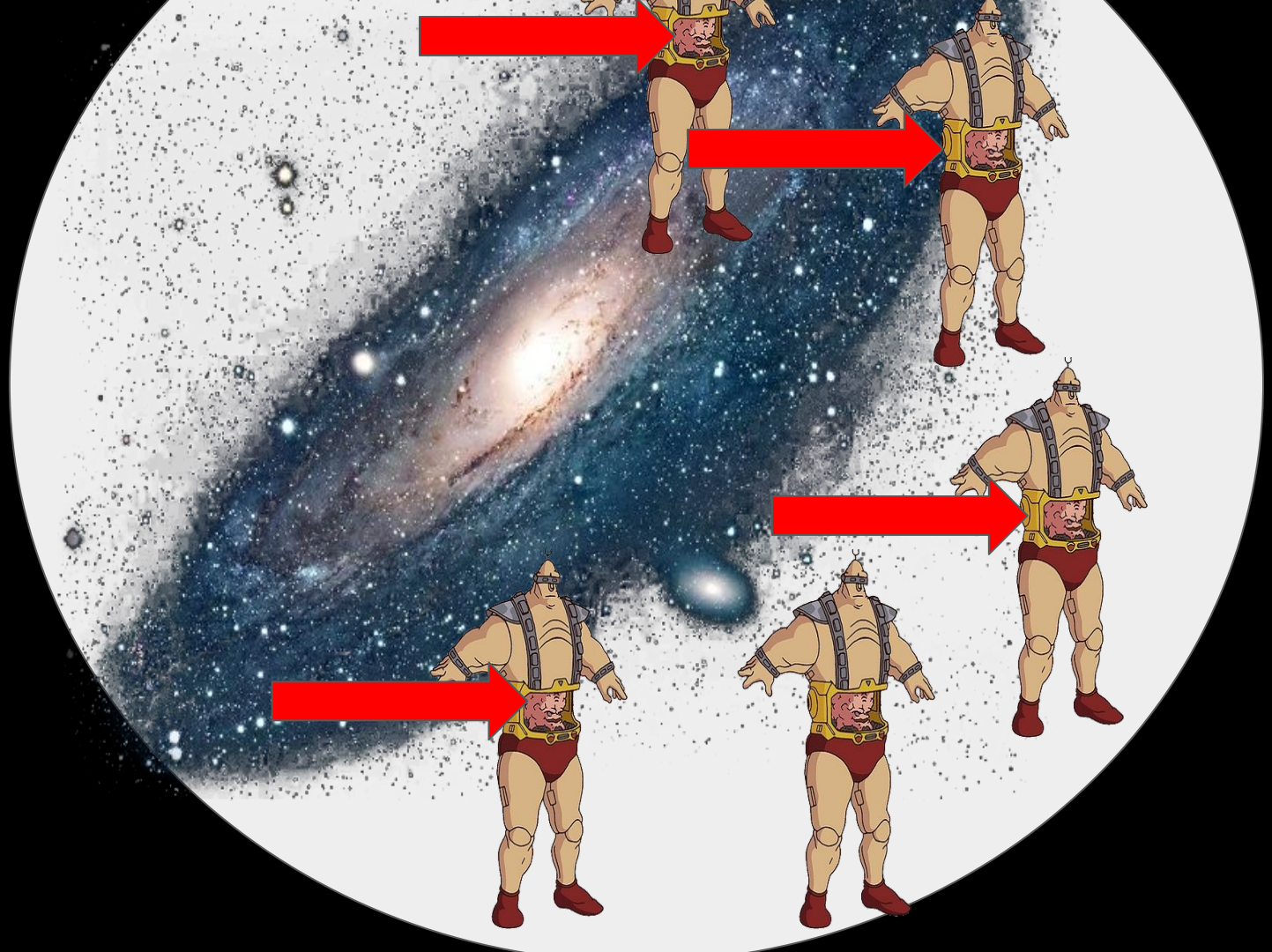

ACI


GC

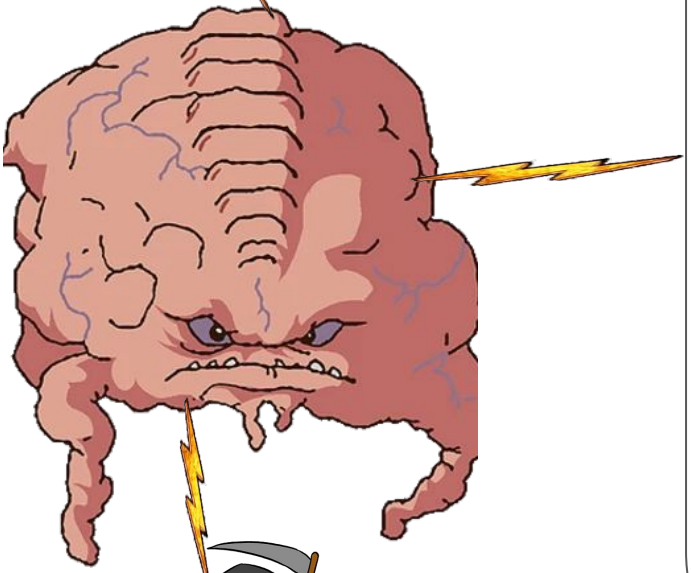

ACI


GC


ACI



Крэнг никогда не был бином

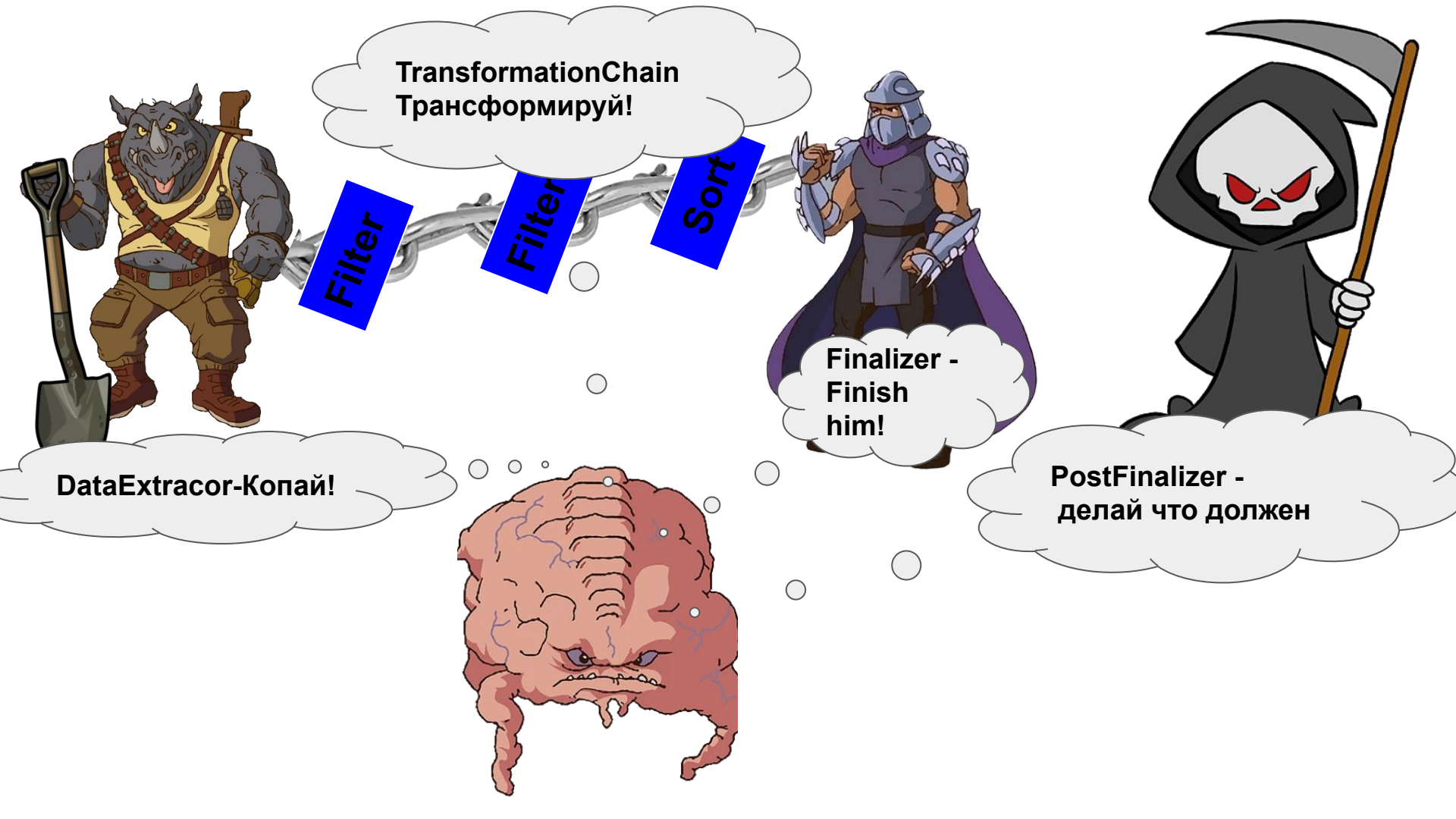


findByNameAnd...Order...

findByNameOrder...

count()

Empty List


TransformationChain
Трансформируй!
Filter
Filter
Sort
Finalizer -
Finish
him!
DataExtracor-Копай!
PostFinalizer -
делай что должен

findByNameOfБабушкаContainsAndAgeLessThanOrderByAgeAndNameSave

findByNameOfБабушкаContainsAndAgeLessThanOrderByAgeAndNameSave

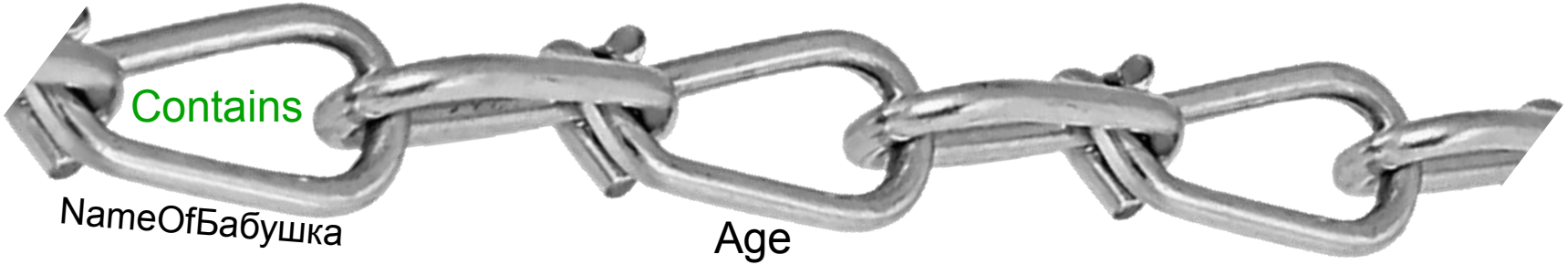
NameOfБабушкаContainsAndAgeLessThanOrderByAgeAndNameSave

ContainsAndAgeLessThanOrderByAgeAndNameSave

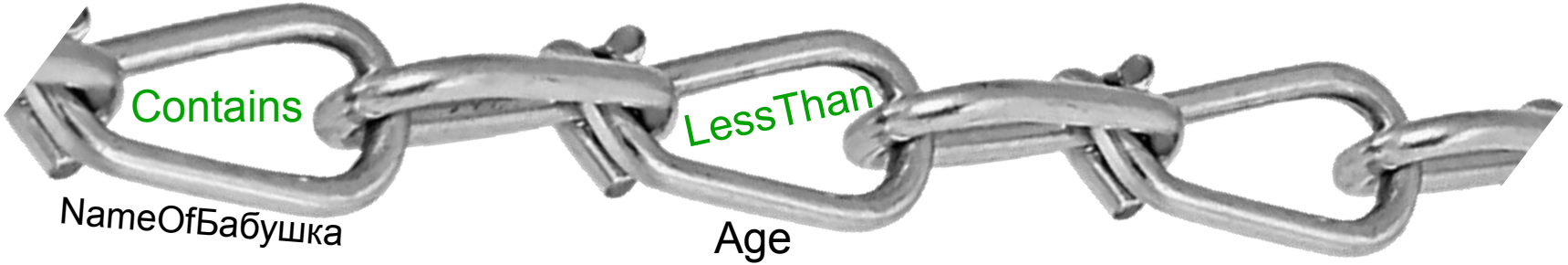
AndAgeLessThanOrderByAgeAndNameSave

AgeLessThanOrderByAgeAndNameSave

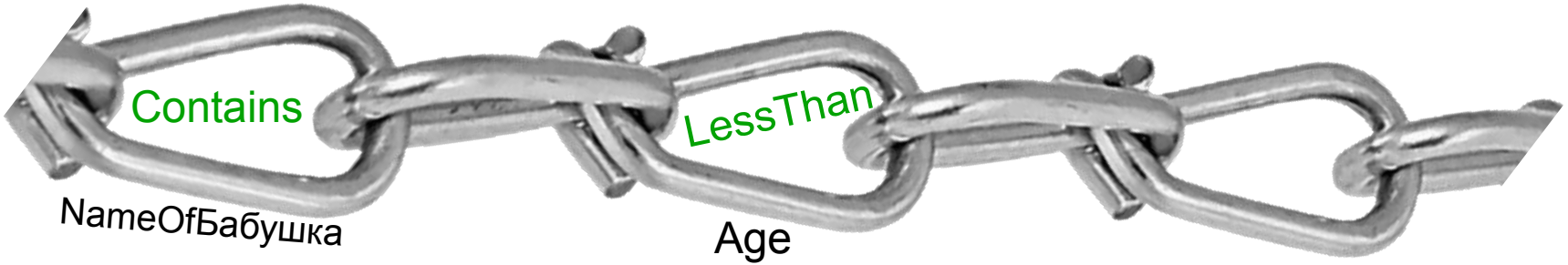
LessThanOrderByAgeAndNameSave



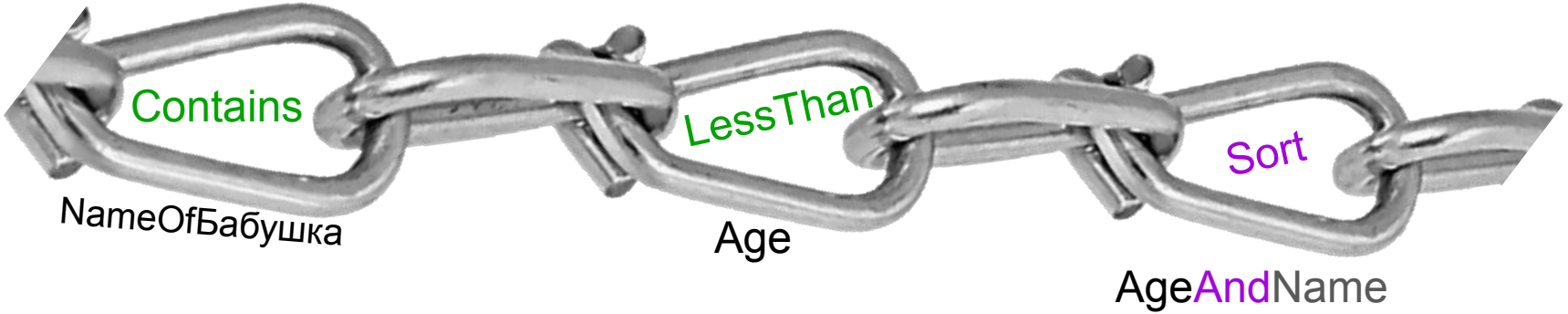
OrderByAgeAndNameSave



AgeAndNameSave



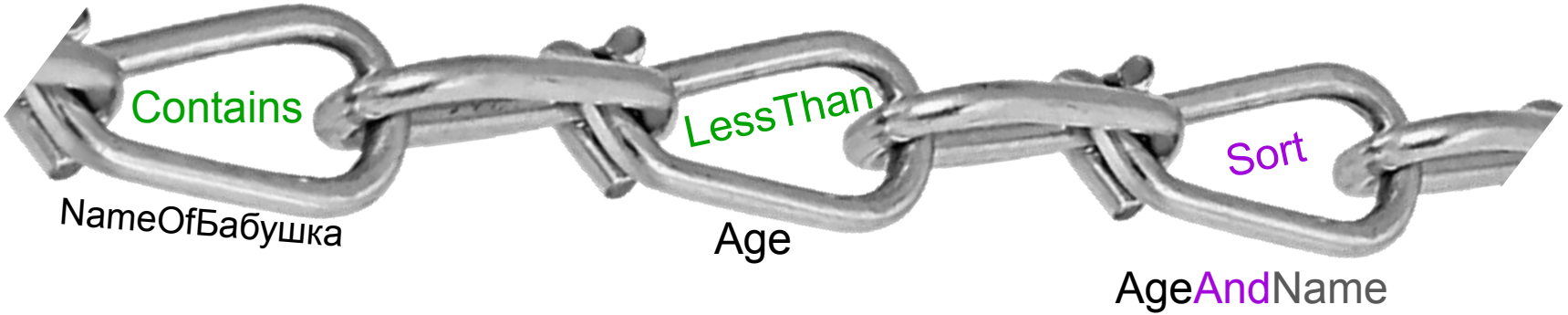
Save


Contains
LessThan
Sort
NameOfБабушка
Age
AgeAndName


Current Spider

Save

SaveFinalizer

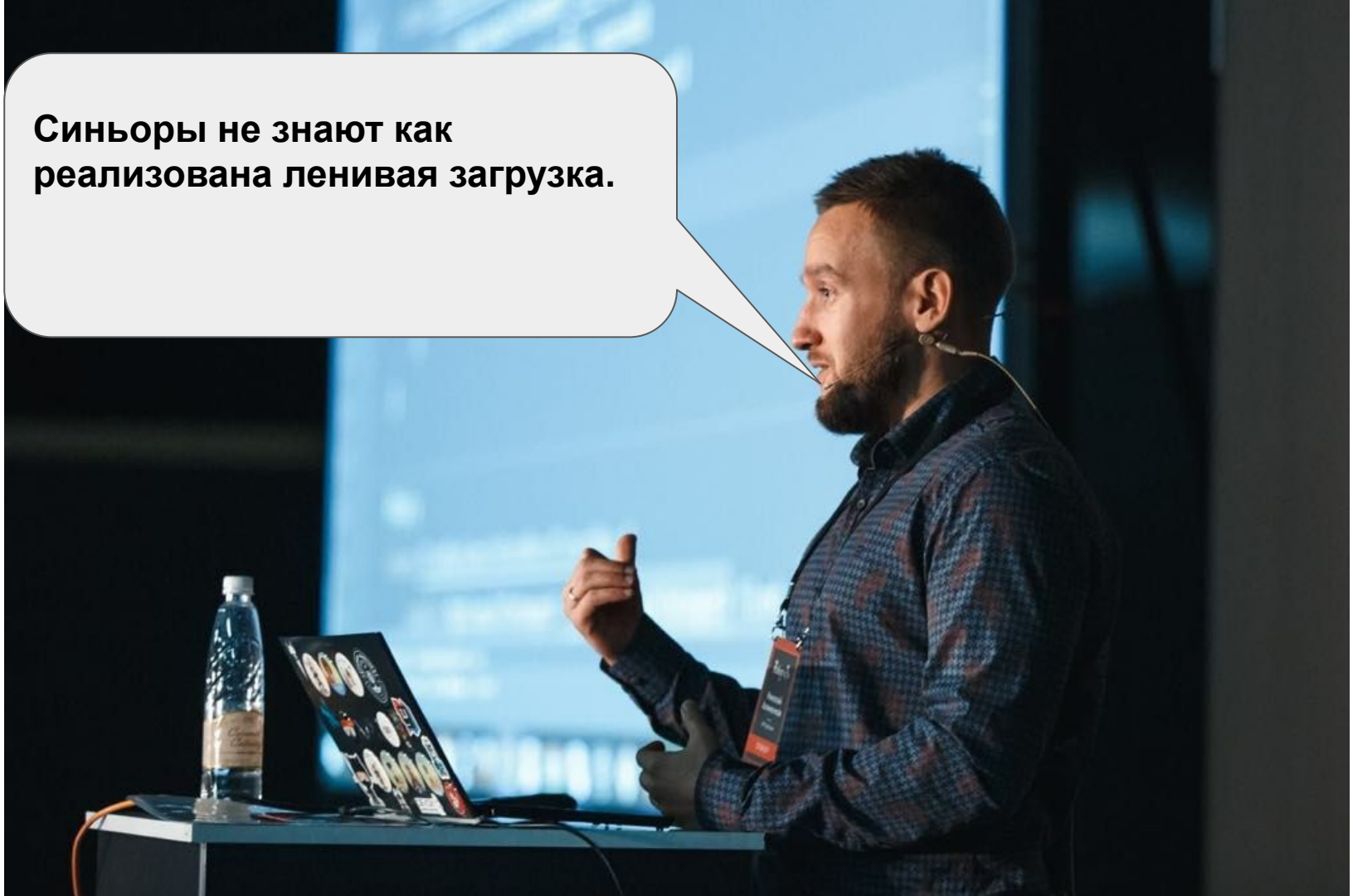


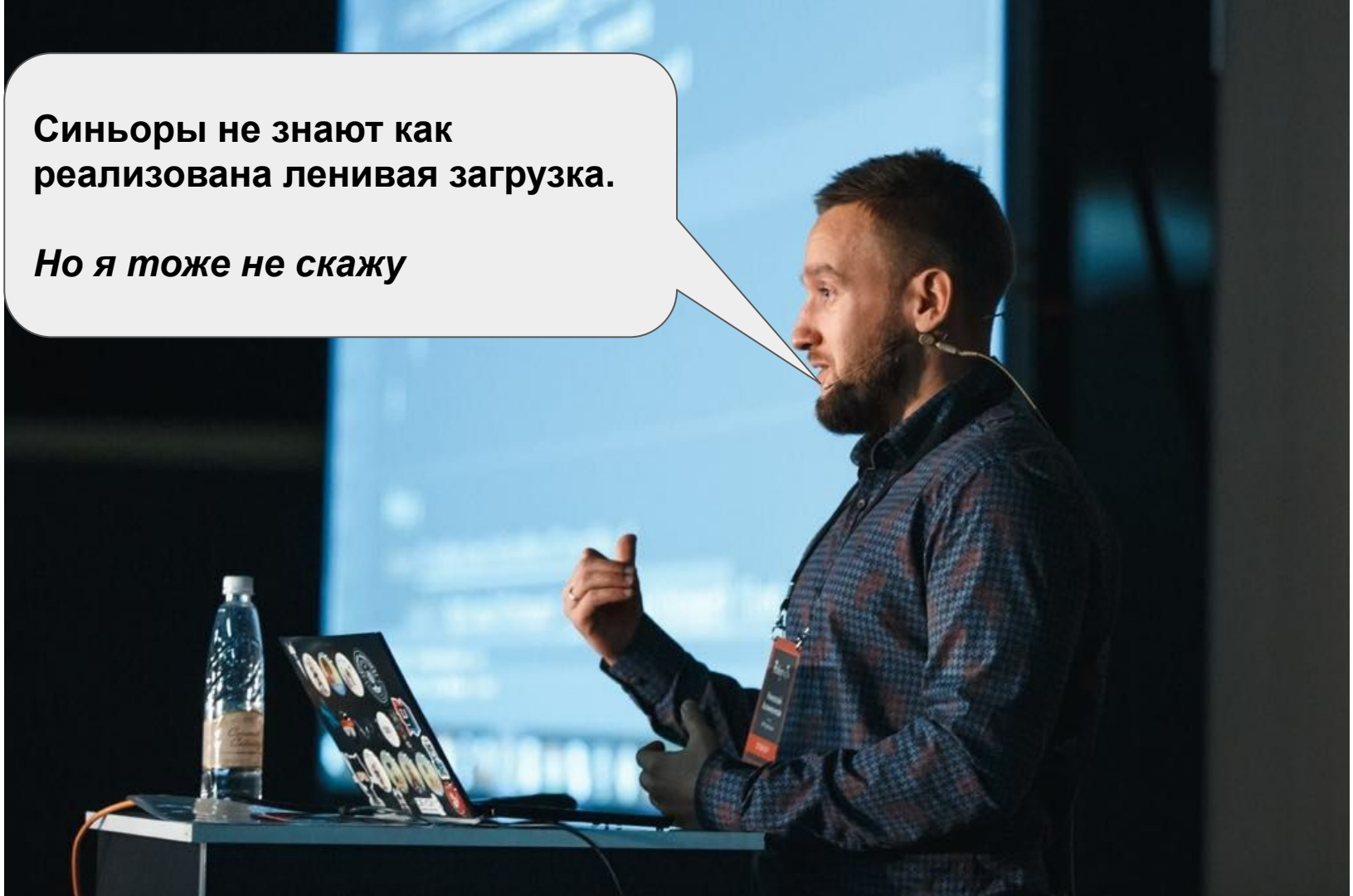
NameOfБабушка

Age

AgeAndName

Дайте что-нибудь zapostфинализировать

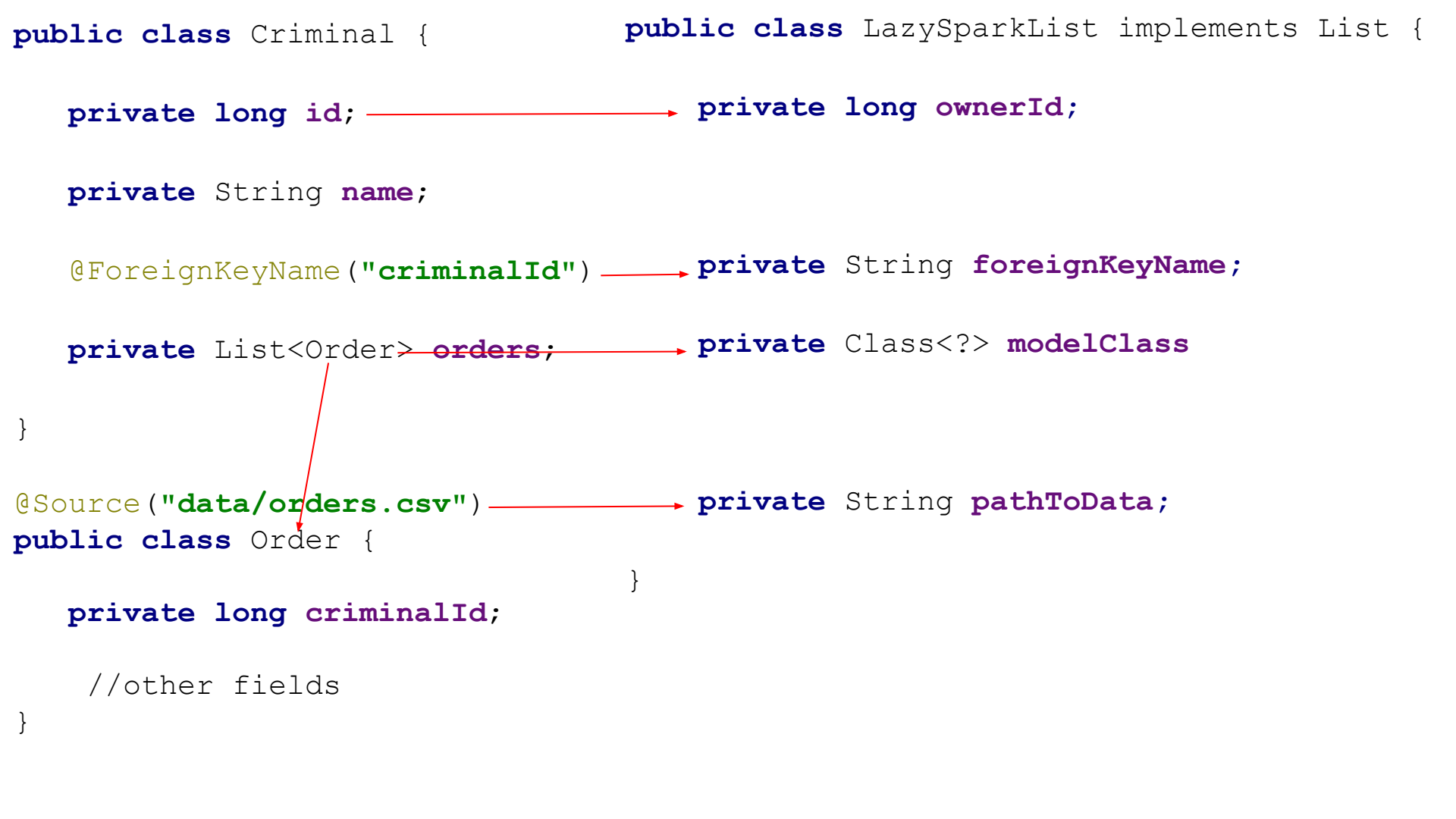

Синьоры не знают как
реализована ленивая загрузка.

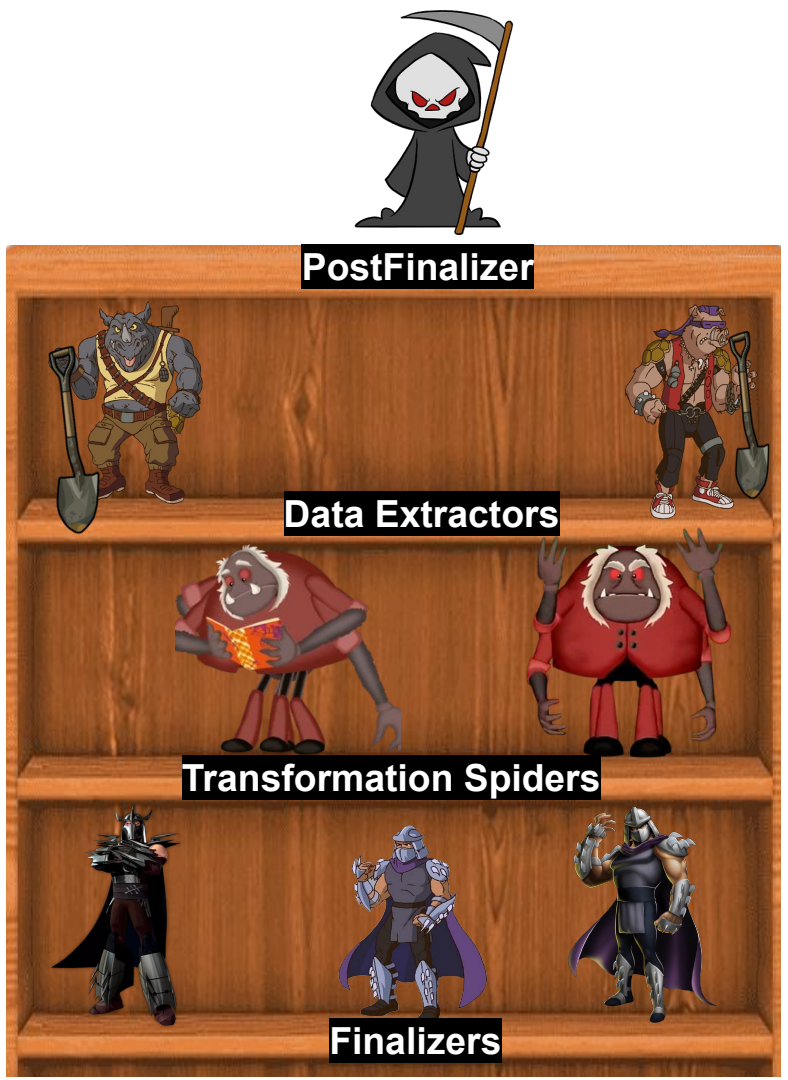

Синьоры не знают как реализована ленивая загрузка.
Но я тоже не скажу

Скажу, скажу
https://www.youtube.com/watch?v=-EpP0Vo63FM

А вот Hibernate не поленились

https://github.com/hibernate/hibernate-orm/blob/main/hibernate-core/src/main/java/org/hibernate/collection/internal/PersistentBag.java

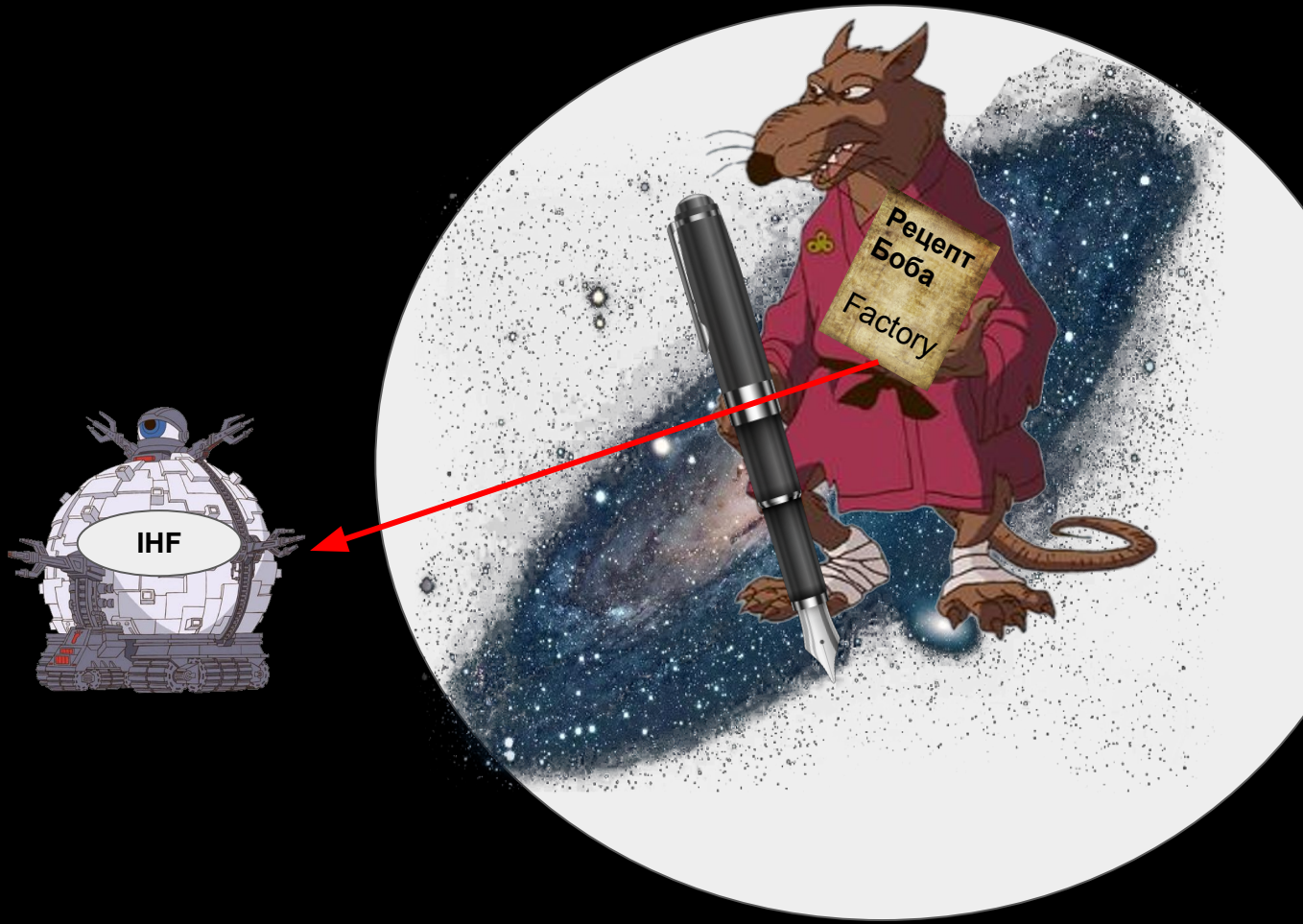

public class Criminal {
private long id;
private String name;
@ForeignKeyName("criminalId")
private List<Order> orders;
}
@Source("data/orders.csv")
public class Order {
private long criminalId;
//other fields
}
public class LazySparkList implements List {
private long ownerId;
private String foreignKeyName;
private Class<?> modelClass
private String pathToData;
}


PostFinalizer
Data Extractors
Transformation Spiders
Finalizers

А как это все по настоящему?

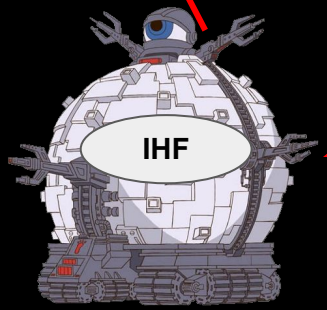

Registrar
Рецепт Боба
Рецепт Боба
Рецепт Боба
Рецепт Боба
Какой же это регистрар?
Вот Регистрар!

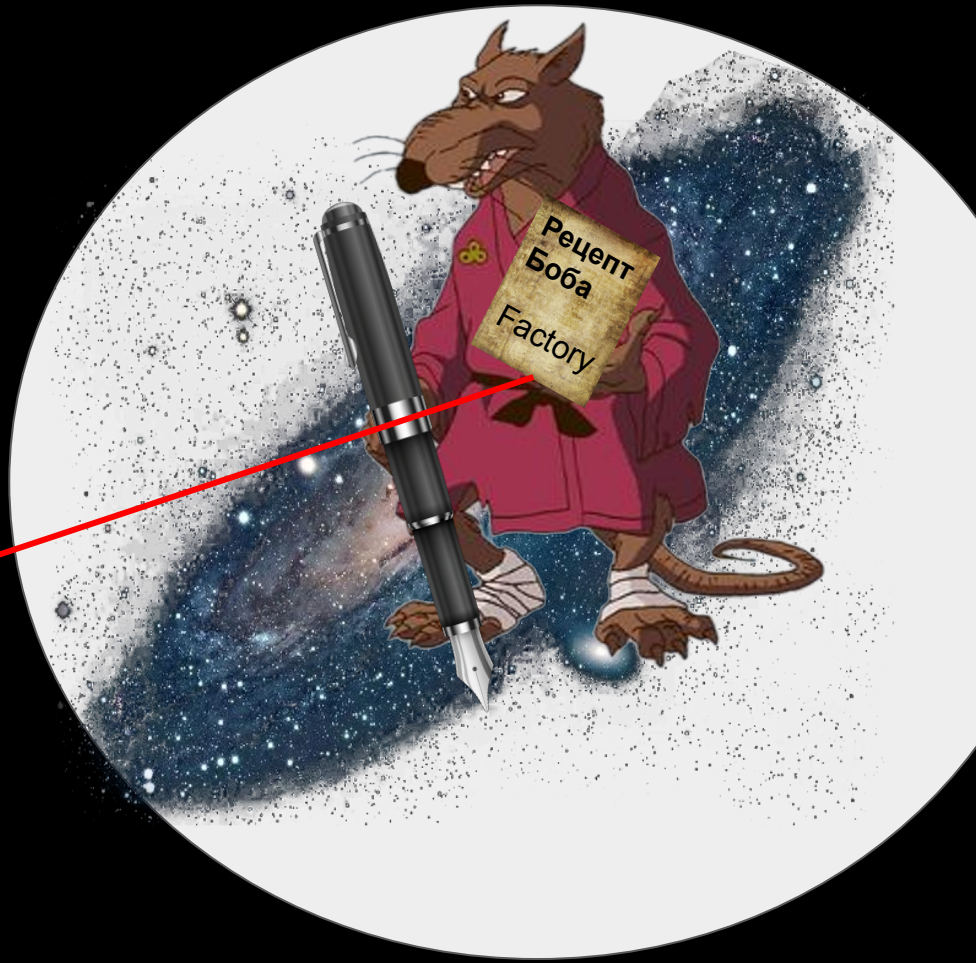
public abstract class AbstractRepositoryConfigurationSourceSupport



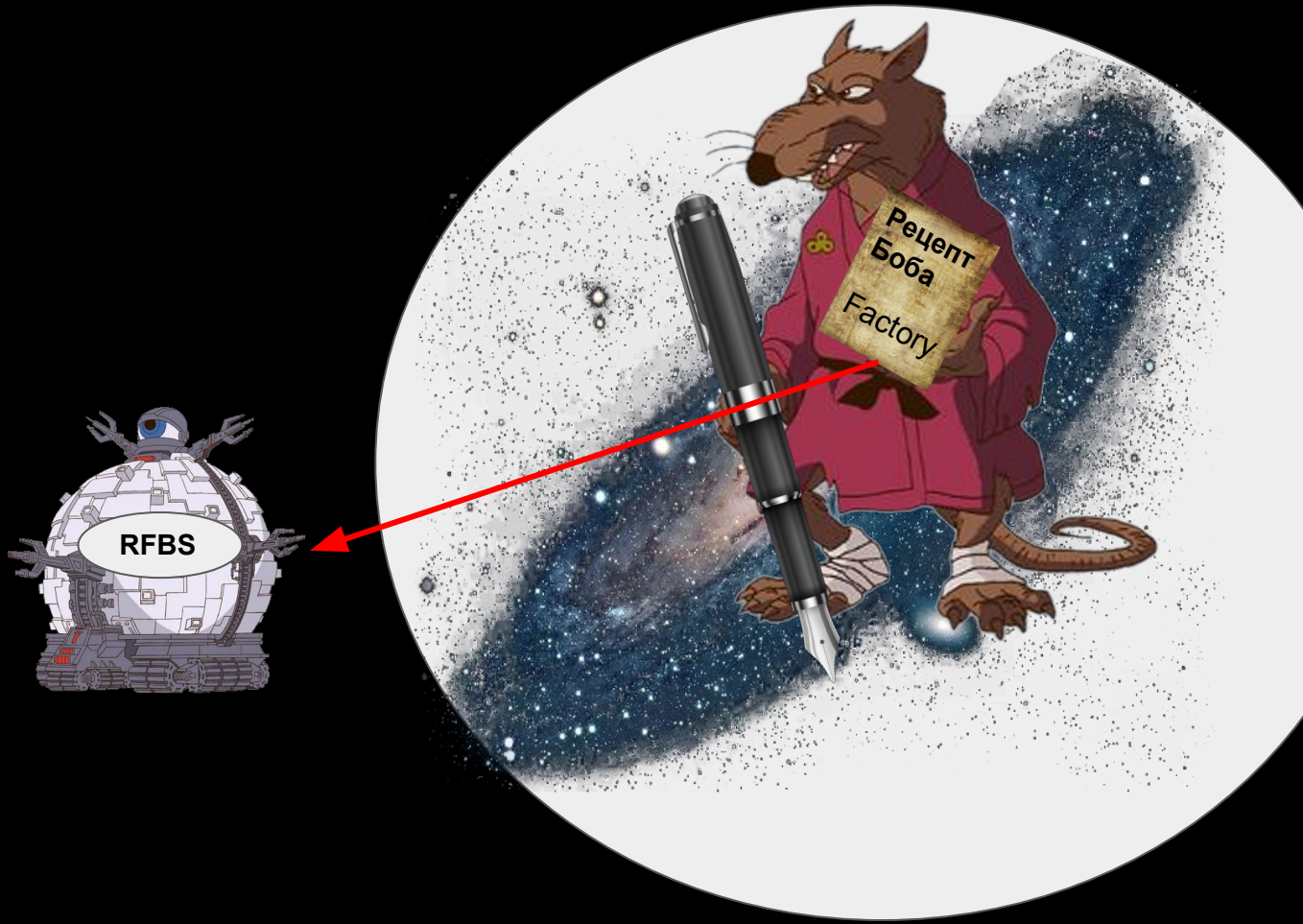
public abstract class AbstractRepositoryConfigurationSourceSupport

```java
public interface FactoryBean<T> {
    @Nullable
    T getObject() throws Exception;

    @Nullable
    Class<?> getObjectType();

    default boolean isSingleton() {
        return true;
    }
}
```



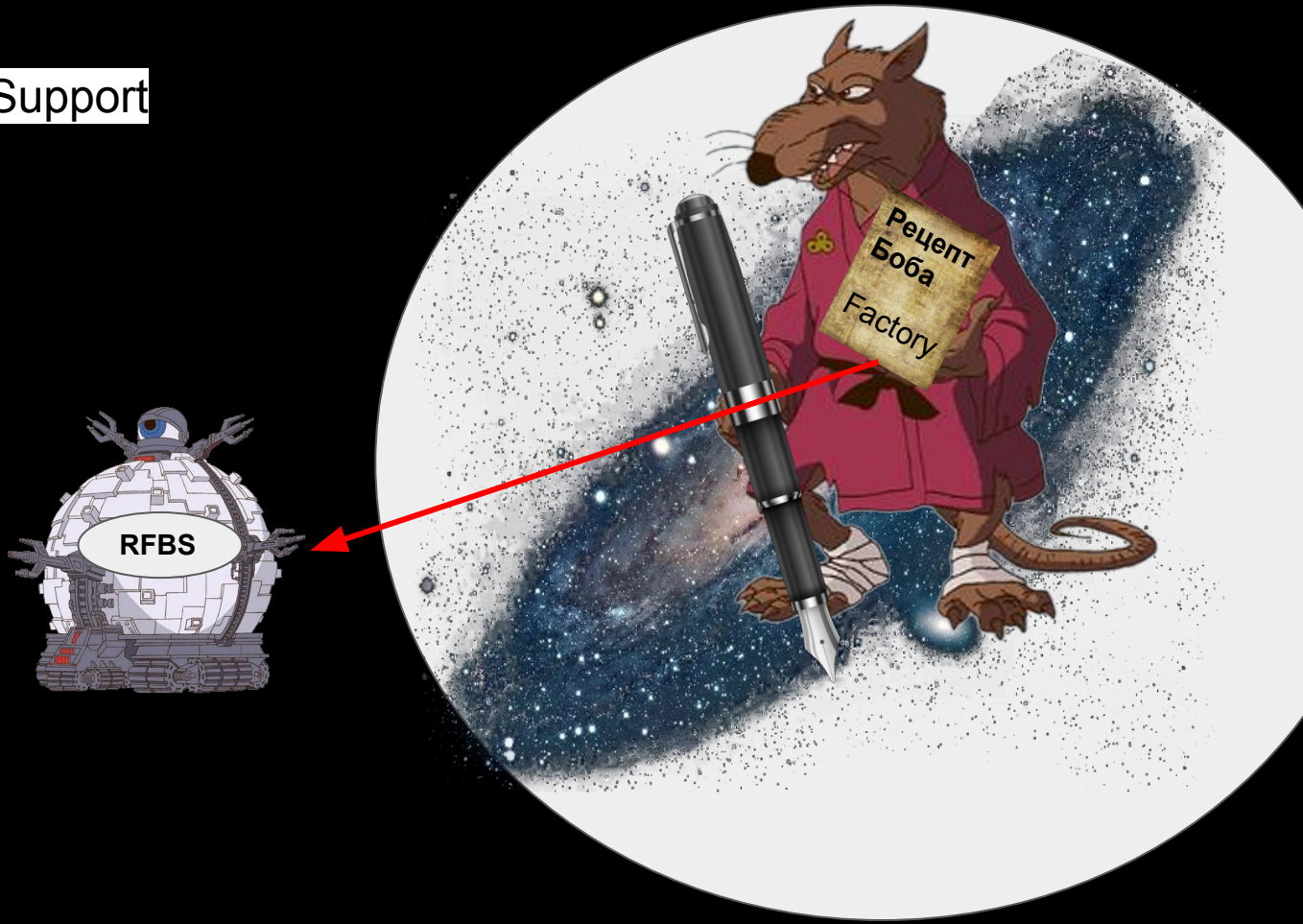



```java
public abstract class AbstractRepositoryConfigurationSourceSupport
```



public abstract class AbstractRepositoryConfigurationSourceSupport

RepositoryFactoryBeanSupport

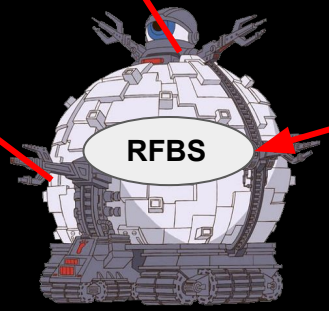


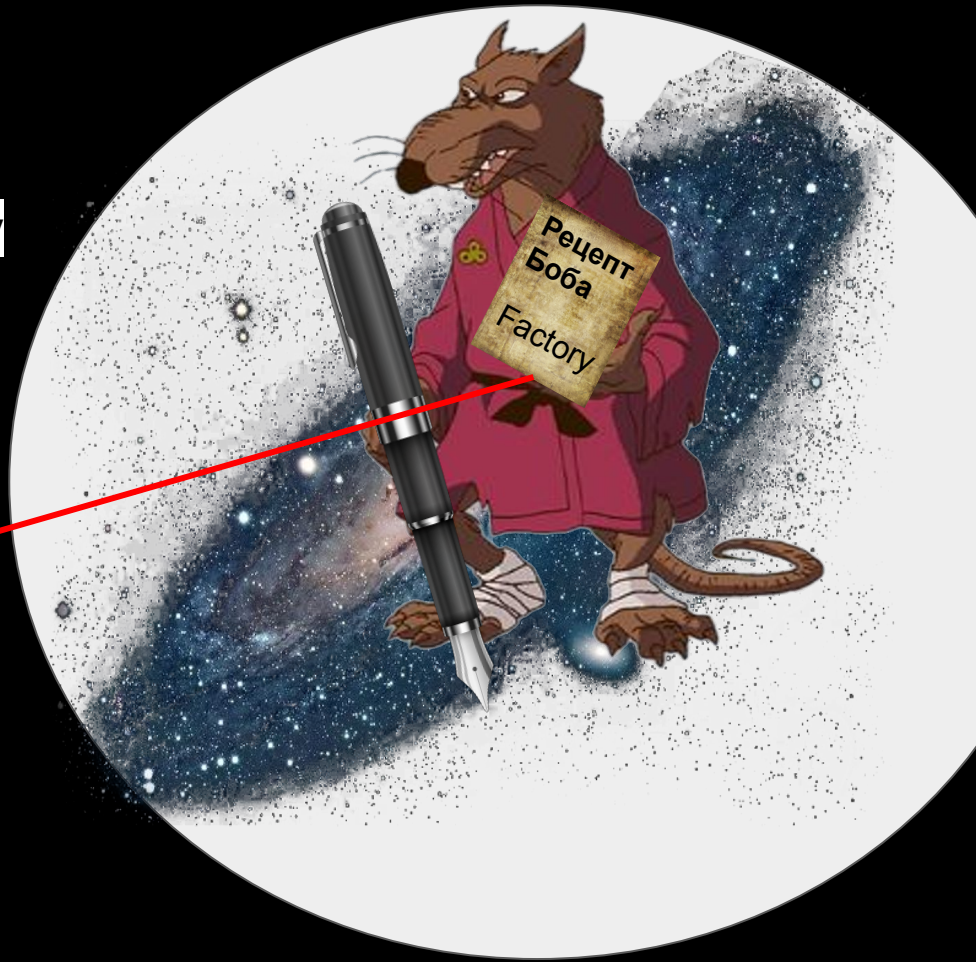
public abstract class AbstractRepositoryConfigurationSourceSupport

RepositoryFactoryBeanSupport

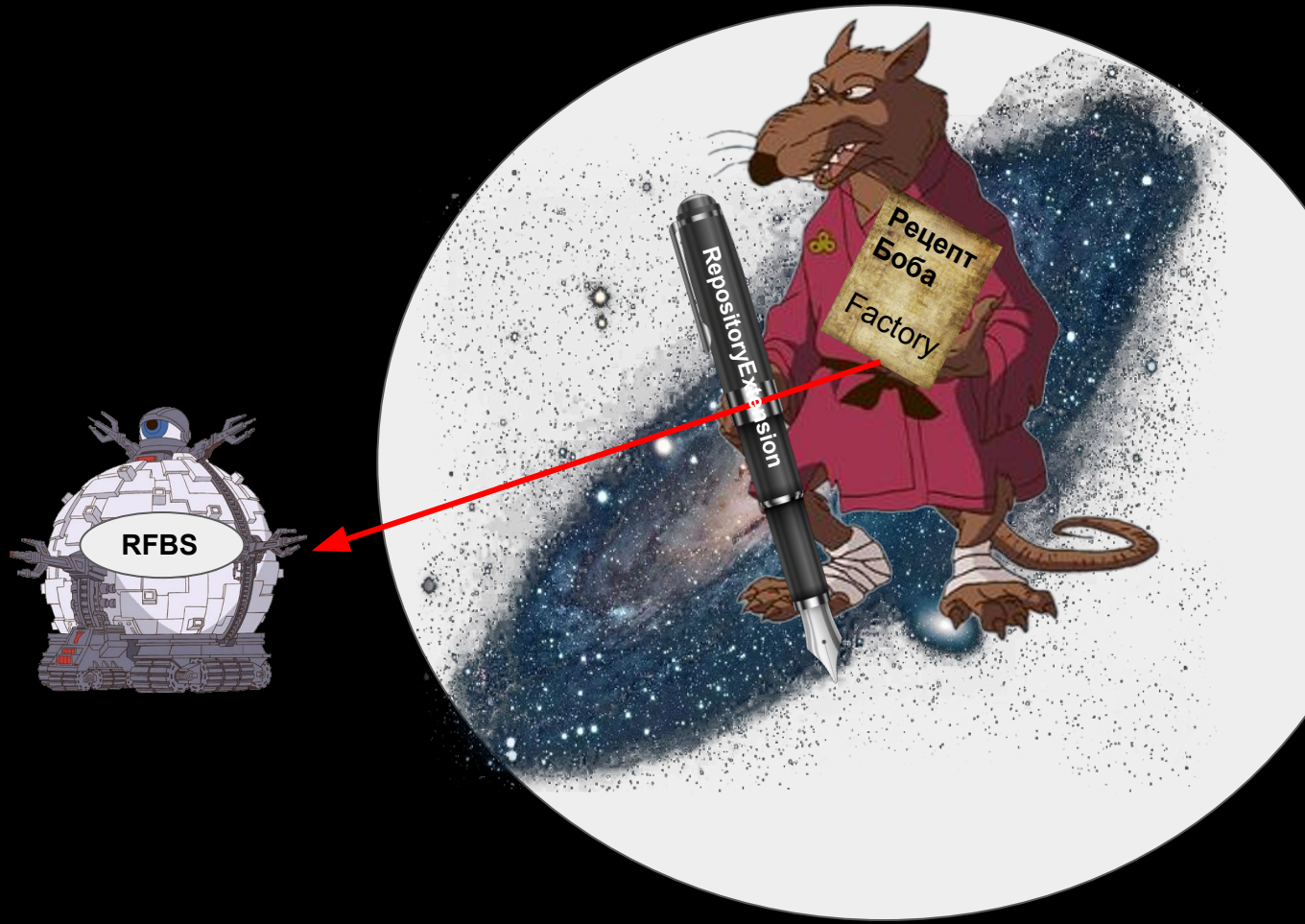
MongoRepository FactoryBean

CassandraRepository FactoryBean





public abstract class AbstractRepositoryConfigurationSourceSupport



public abstract class AbstractRepositoryConfigurationSourceSupport

```
class JpaRepositoriesRegistrar extends AbstractRepositoryConfigurationSourceSupport {
    @Override
    protected RepositoryConfigurationExtension getRepositoryConfigurationExtension() {
        return new JpaRepositoryConfigExtension();
    }
```

```
public class JpaRepositoryConfigExtension extends RepositoryConfigurationExtensionSupport {

    @Override
    public String getRepositoryFactoryBeanClassName() { return JpaRepositoryFactoryBean.class.getName(); }
```

```
▼ beanDefinition = {RootBeanDefinition@4524} "Root bean: class [org.springframework.data.jpa.repository.support.JpaReposito
    decoratedDefinition = null
    qualifiedElement = null
    stale = false
    externallyManagedDestroyMethods = null
▶   beanClass = "org.springframework.data.jpa.repository.support.JpaRepositoryFactoryBean"
▶   scope = ""
    abstractFlag = false
```

Registrar

- Как можно продекларировать Registrar?

```
@Import(JpaRepositoriesRegistrar.class)
```

Registrar

- Но можно же сделать круче!

```
@EnableJpaRepositories
```

```
@EnableJpaRepositories(basePackages = "com.epam")
```

```
@Import(JpaRepositoriesRegistrar.class)
public @interface EnableJpaRepositories
```

Registrar

- Но можно совсем по взрослому!

```
# Auto Configure
org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
org.springframework.boot.autoconfigure.admin.SpringApplicationAdminJmxAutoConfiguration,\
org.springframework.boot.autoconfigure.aop.AopAutoConfiguration,\
org.springframework.boot.autoconfigure.amqp.RabbitAutoConfiguration,\
org.springframework.boot.autoconfigure.batch.BatchAutoConfiguration,\
org.springframework.boot.autoconfigure.cache.CacheAutoConfiguration,\
org.springframework.boot.autoconfigure.cassandra.CassandraAutoConfiguration,\
org.springframework.boot.autoconfigure.context.ConfigurationPropertiesAutoConfiguration,\
org.springframework.boot.autoconfigure.context.LifecycleAutoConfiguration,\
org.springframework.boot.autoconfigure.context.MessageSourceAutoConfiguration,\
org.springframework.boot.autoconfigure.context.PropertyPlaceholderAutoConfiguration,\
org.springframework.boot.autoconfigure.couchbase.CouchbaseAutoConfiguration,\
org.springframework.boot.autoconfigure.dao.PersistenceExceptionTranslationAutoConfiguration
org.springframework.boot.autoconfigure.data.cassandra.CassandraDataAutoConfiguration,\
org.springframework.boot.autoconfigure.data.cassandra.CassandraReactiveDataAutoConfiguratio
org.springframework.boot.autoconfigure.data.cassandra.CassandraReactiveRepositoriesAutoConf
org.springframework.boot.autoconfigure.data.cassandra.CassandraRepositoriesAutoConfiguratio
org.springframework.boot.autoconfigure.data.couchbase.CouchbaseDataAutoConfiguration,\
org.springframework.boot.autoconfigure.data.couchbase.CouchbaseReactiveDataAutoConfiguratio
org.springframework.boot.autoconfigure.data.couchbase.CouchbaseReactiveRepositoriesAutoConf
org.springframework.boot.autoconfigure.data.couchbase.CouchbaseRepositoriesAutoConfiguratio
org.springframework.boot.autoconfigure.data.elasticsearch.ElasticsearchDataAutoConfiguratio
org.springframework.boot.autoconfigure.data.elasticsearch.ElasticsearchRepositoriesAutoConf
org.springframework.boot.autoconfigure.data.elasticsearch.ReactiveElasticsearchRepositories
org.springframework.boot.autoconfigure.data.elasticsearch.ReactiveElasticsearchRestClientAu
org.springframework.boot.autoconfigure.data.jdbc.JdbcRepositoriesAutoConfiguration,\
org.springframework.boot.autoconfigure.data.jpa.JpaRepositoriesAutoConfiguration,\
org.springframework.boot.autoconfigure.data.ldap.LdapRepositoriesAutoConfiguration,\
org.springframework.boot.autoconfigure.data.mongo.MongoDataAutoConfiguration,\
```

```
@Configuration(proxyBeanMethods = false)
@ConditionalOnBean(DataSource.class)
@ConditionalOnClass(JpaRepository.class)
@ConditionalOnMissingBean({ JpaRepositoryFacto
@ConditionalOnProperty(prefix = "spring.data.j
        matchIfMissing = true)
@Import(JpaRepositoriesRegistrar.class)
@AutoConfigureAfter({ HibernateJpaAutoConfigur
public class JpaRepositoriesAutoConfiguration
```

AbstractRepositoryConfigurationSourceSupport / RepositoryBeanDefinitionRegistrarSupport

```
class JpaRepositoriesRegistrar extends AbstractRepositoryConfigurationSourceSupport

@Override
protected Class<? extends Annotation> getAnnotation() {
    return EnableJpaRepositories.class;
}
 @Override
 protected RepositoryConfigurationExtension getRepositoryConfigurationExtension() {
     return new JpaRepositoryConfigExtension();
 }
```

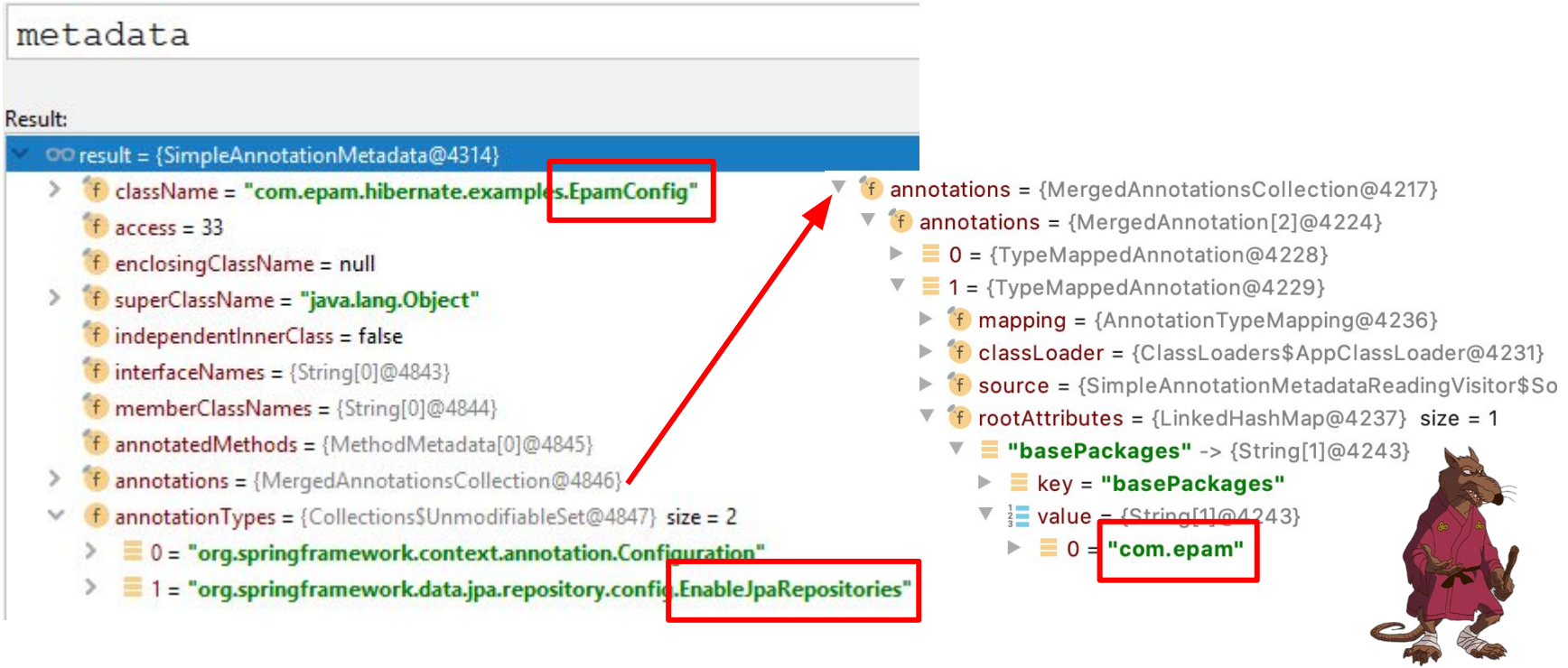
```
class JpaRepositoriesRegistrar extends RepositoryBeanDefinitionRegistrarSupport

@Override
protected Class<? extends Annotation> getAnnotation() {
    return EnableJpaRepositories.class;
}

@Override
protected RepositoryConfigurationExtension getExtension() {
    return new JpaRepositoryConfigExtension();
}
```

```
@Override
public void registerBeanDefinitions(AnnotationMetadata importingClassMetadata, BeanDefinitionRegistry registry,
        BeanNameGenerator importBeanNameGenerator) {

RepositoryConfigurationDelegate delegate = new RepositoryConfigurationDelegate(
        getConfigurationSource(registry, importBeanNameGenerator), this.resourceLoader, this.environment);
delegate.registerRepositoriesIn(registry, getRepositoryConfigurationExtension());
```

```
RepositoryConfigurationExtension extension = getExtension();
RepositoryConfigurationDelegate delegate = new RepositoryConfigurationDelegate(configurationSource, resourceLoader,
        environment);

delegate.registerRepositoriesIn(registry, extension);
```

JpaRepositoriesRegistrar flow with @EnableJpaRepositoreis

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```



JpaRepositoriesRegistrar flow without @EnableJpaRepositoreis

```
void registerBeanDefinitions(AnnotationMetadata importingClassMetadata,
registry...)
```

importingClassMetadata

Result:

- result = {SimpleAnnotationMetadata@4494}
  - className = "org.springframework.boot.autoconfigure.data.jpa.JpaRepositoriesAutoConfiguration"
  - access = 33
  - enclosingClassName = null
  - superClassName = "java.lang.Object"
  - independentInnerClass = false
  - interfaceNames = {String[0]@5295}
  - memberClassNames = {String[1]@5296}
  - annotatedMethods = {MethodMetadata[1]@5297}
  - annotations = {MergedAnnotationsCollection@5298}
  - annotationTypes = {Collections$UnmodifiableSet@5299} size = 7
    - 0 = "org.springframework.context.annotation.Configuration"
    - 1 = "org.springframework.boot.autoconfigure.condition.ConditionalOnBean"
    - 2 = "org.springframework.boot.autoconfigure.condition.ConditionalOnClass"
    - 3 = "org.springframework.boot.autoconfigure.condition.ConditionalOnMissingBean"
    - 4 = "org.springframework.boot.autoconfigure.condition.ConditionalOnProperty"
    - 5 = "org.springframework.context.annotation.Import"
    - 6 = "org.springframework.boot.autoconfigure.AutoConfigureAfter"

А где @EnableJpaRepositoreis?

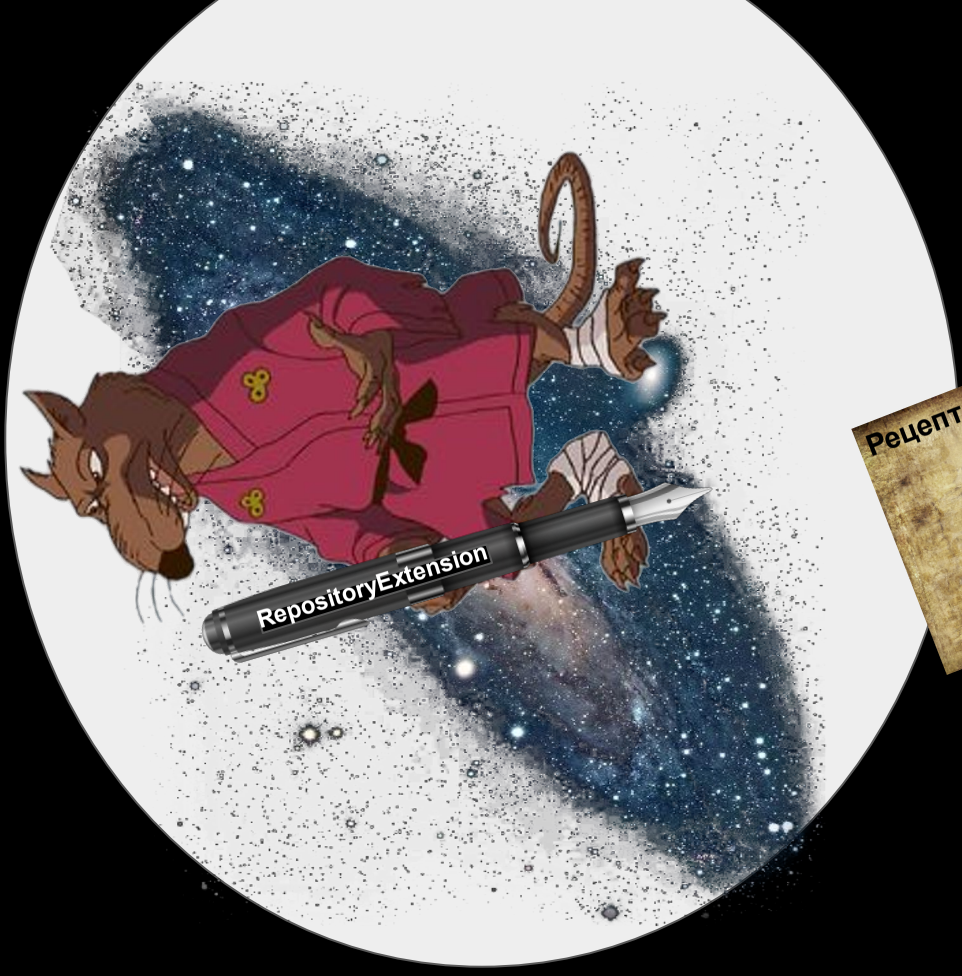

RepositoryExtension

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```

RepositoryExtension
RepositoryConfiguration

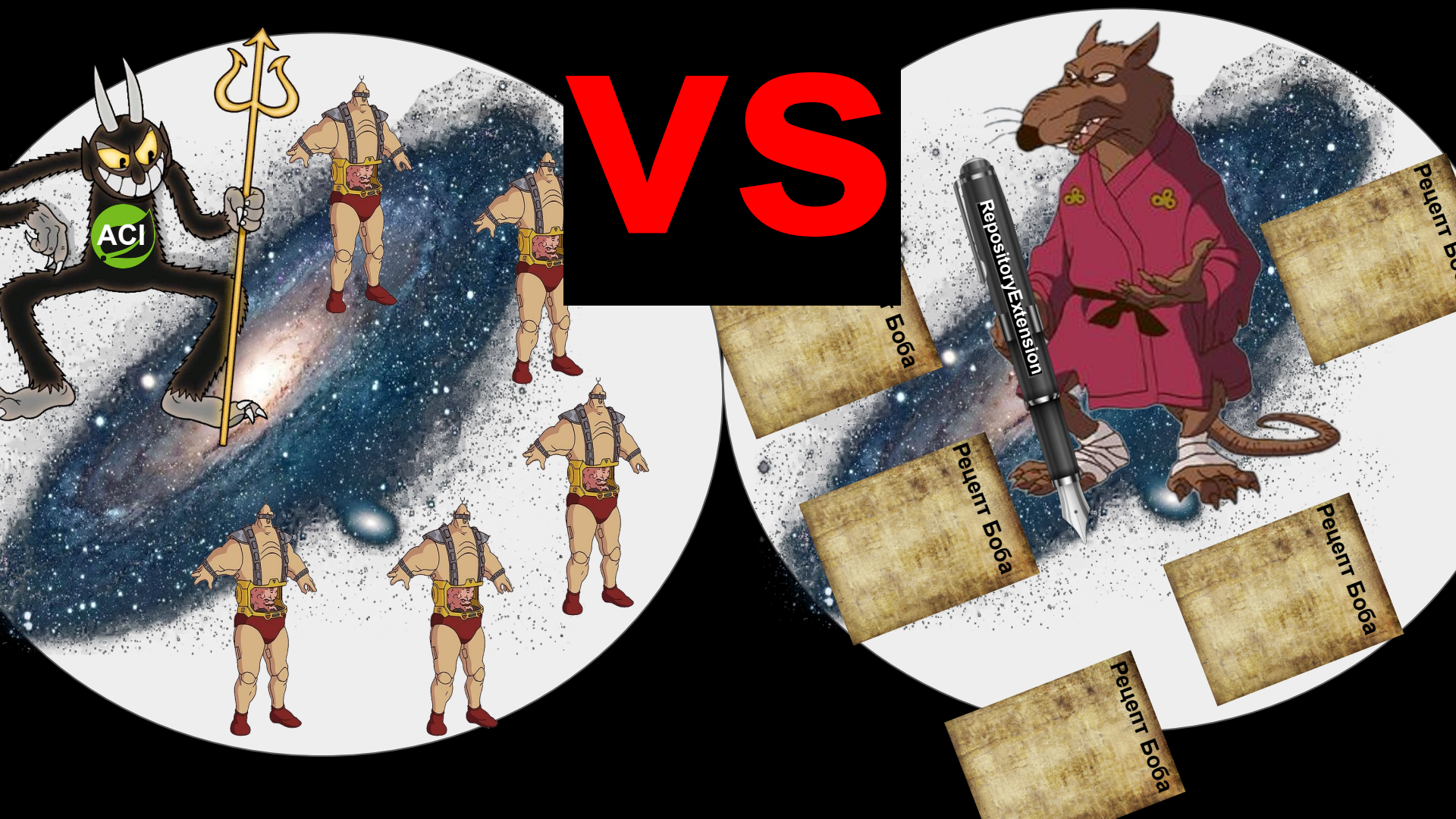

RepositoryExtension
BeanDefinitionBuilder

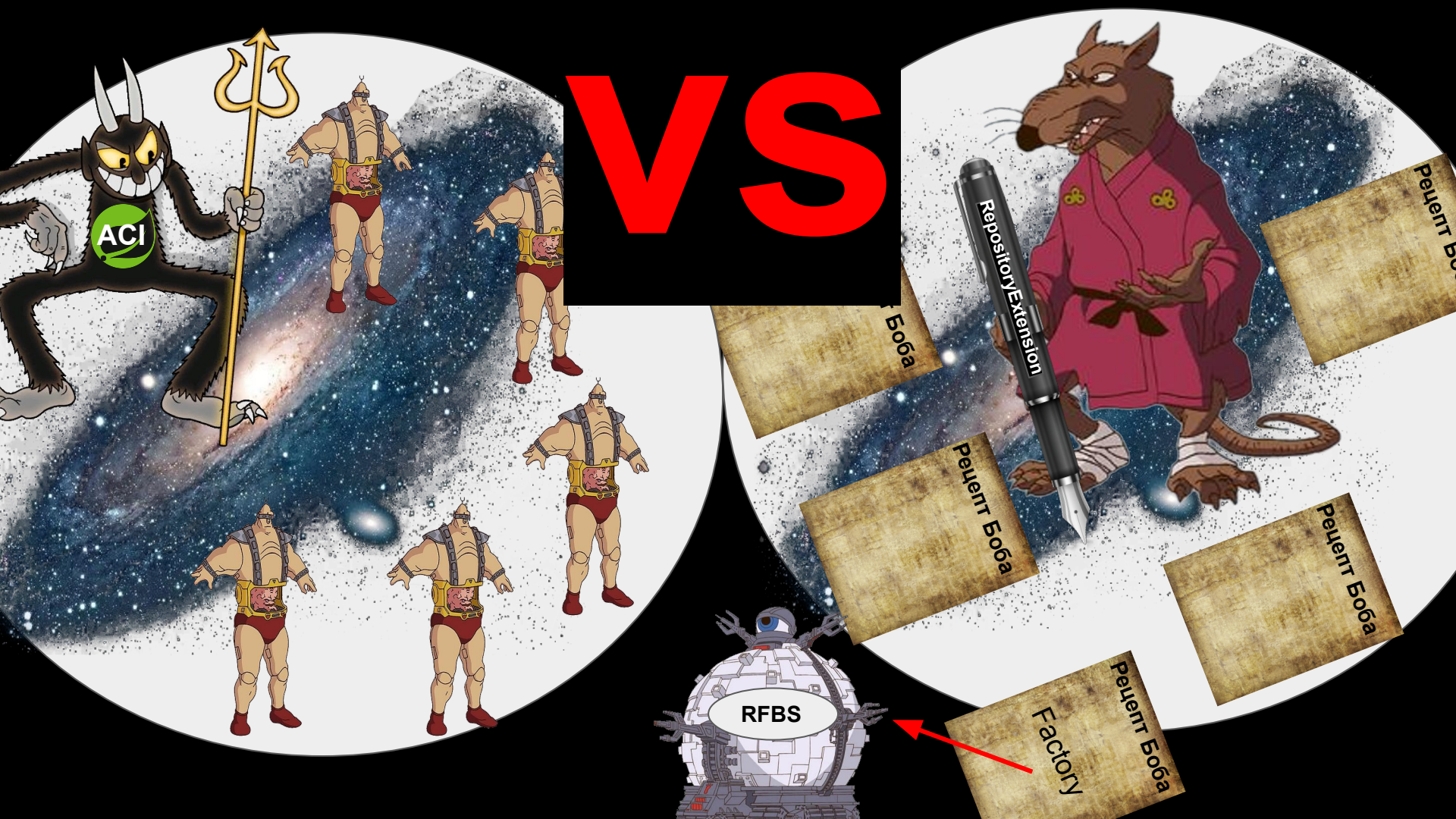

RepositoryExtension

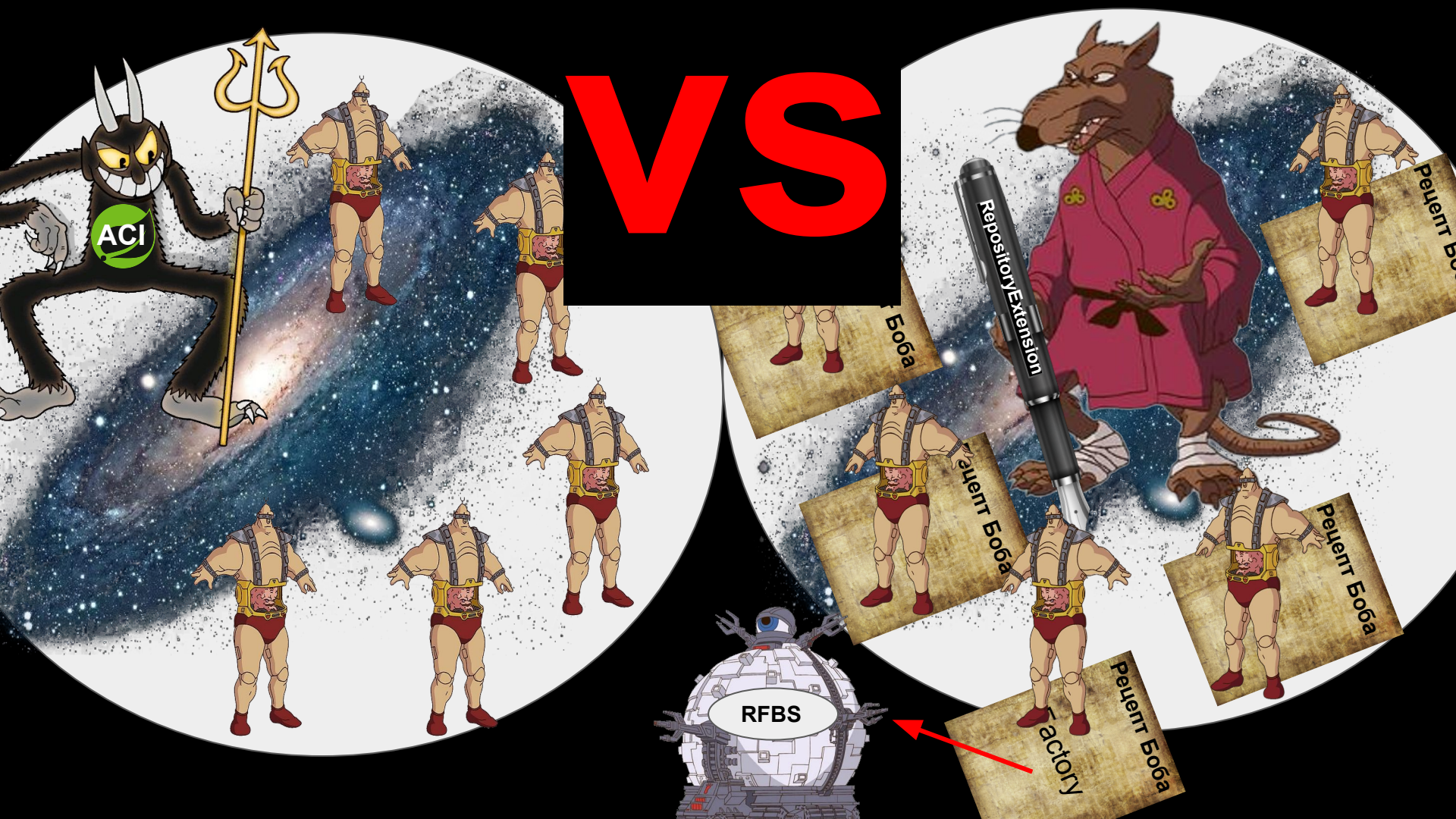

Рецепт Боба


RepositoryExtension
Рецепт Боба
Боба


Рецепт Боба
Рецепт Боба
Рецепт Боба
RepositoryExtension
Рецепт Боба
Рецепт Боба

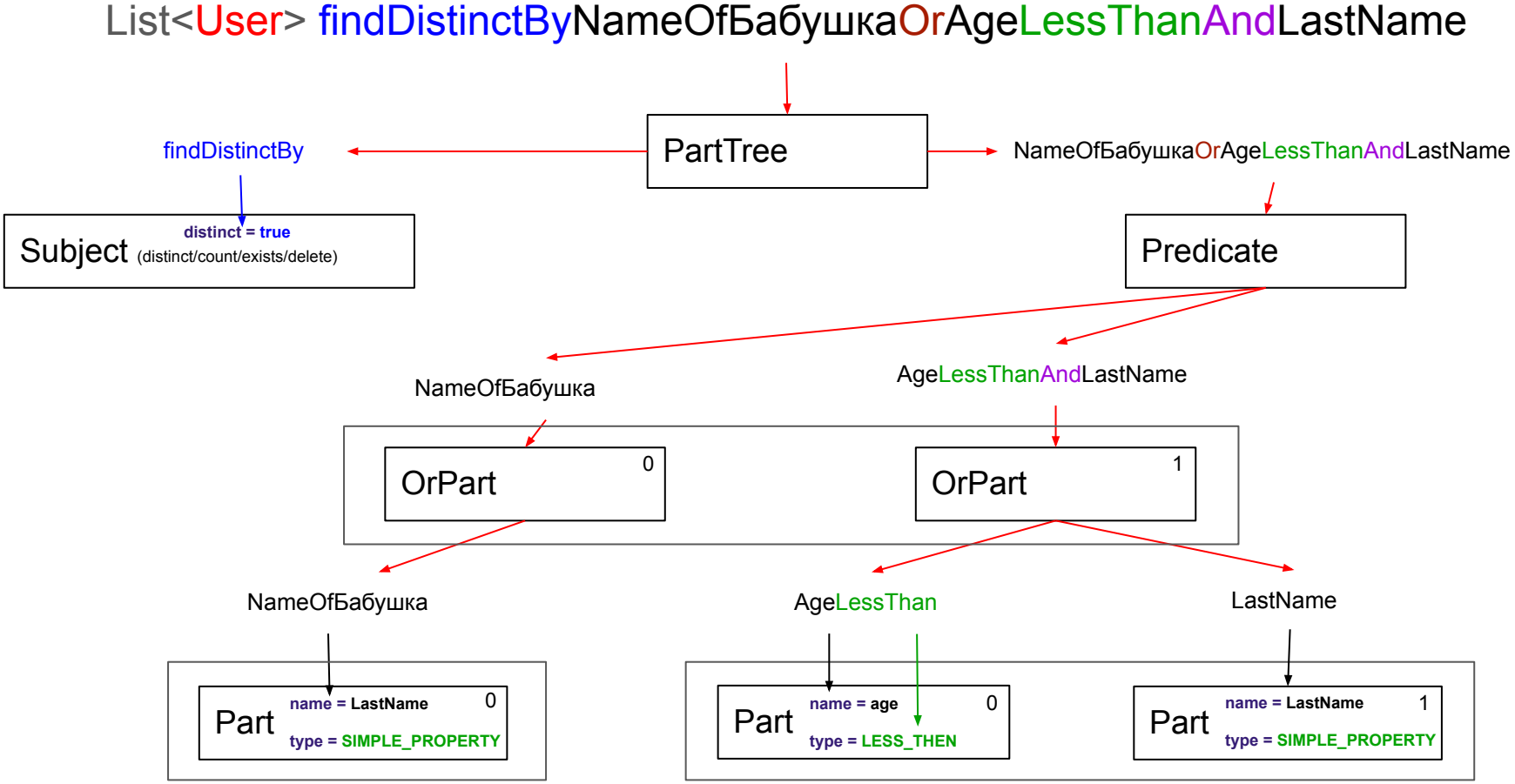

ACI
VS
RepositoryExtension
Рецепт Боба
Рецепт Боба
Рецепт Боба
Рецепт Боба


ACI
VS
RepositoryExtension
Рецепт Боба
Рецепт Боба
Рецепт Боба
Рецепт Боба
RFBS
Factory


ACI
VS
RepositoryExtension
Рецепт Боба
Рецепт Боба
Рецепт Боба
Рецепт Боба
RFBS
Factory

Настоящая Spring Data более гуманна

Мы фабриками
не швыряемся


List<User> findDistinctByNameOfБабушкаOrAgeLessThanAndLastName
PartTree
findDistinctBy
NameOfБабушкаOrAgeLessThanAndLastName
Subject
distinct = true
(distinct/count/exists/delete)
Predicate
NameOfБабушка
AgeLessThanAndLastName
OrPart
0
OrPart
1
NameOfБабушка
AgeLessThan
LastName
Part
name = LastName
0
type = SIMPLE_PROPERTY
Part
name = age
0
type = LESS_THEN
Part
name = LastName
1
type = SIMPLE_PROPERTY

Траблшутинг, дополнительные возможности, нюансы и тонкости

Проблема первая

Проблема первая

1. Сначало все работало

Проблема первая

1. Сначало все работало
2. Потом Подключили какой то стартер

Проблема первая

1. Сначало все работало
2. Потом Подключили какой то стартер
3. После этого все стало падать

Проблема первая

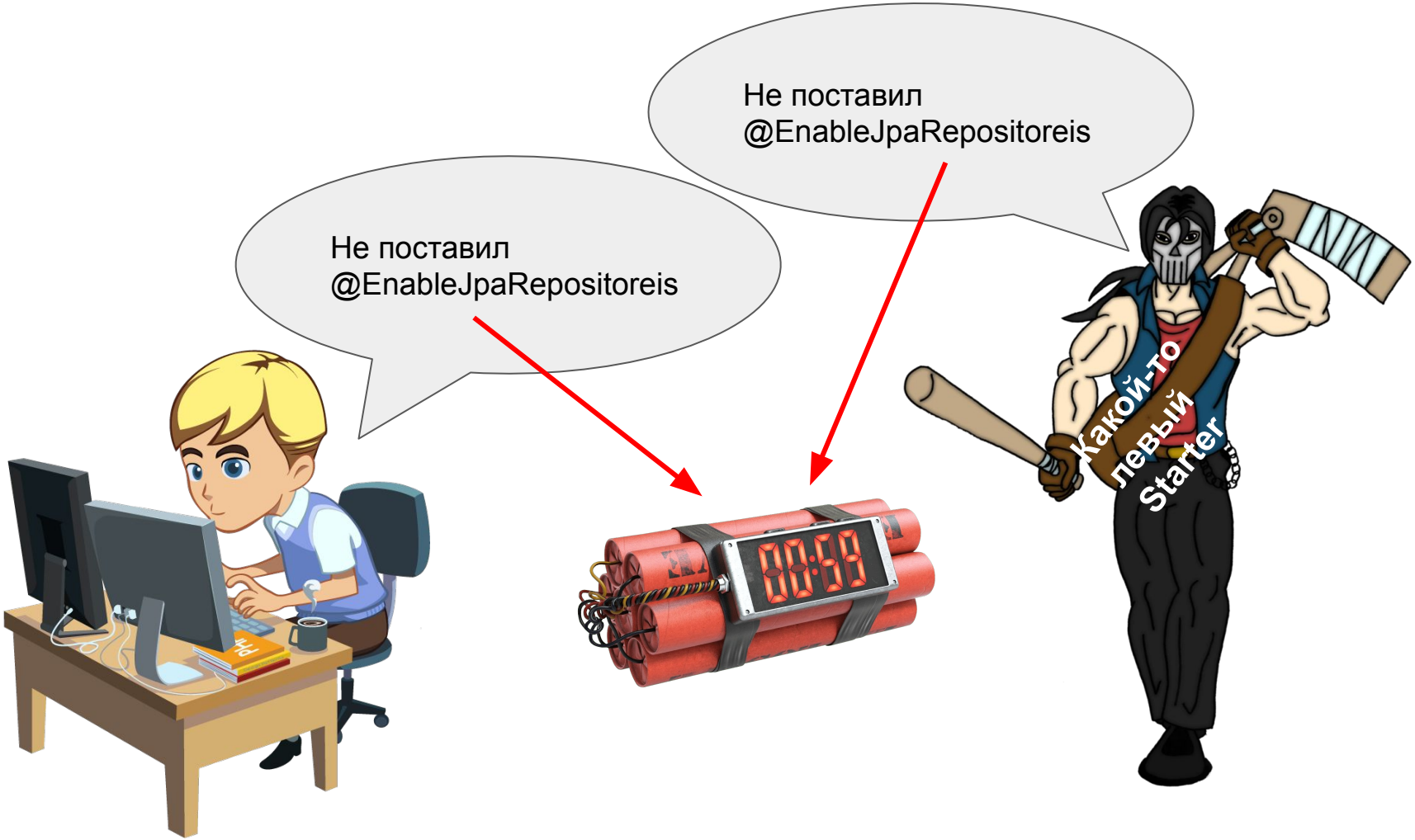
1. Сначало все работало
2. Потом Подключили какой то стартер
3. После этого все стало падать

- Мои репозитории больше не обнаруживаются

Проблема первая

1. Сначало все работало
2. Потом Подключили какой то стартер
3. После этого все стало падать

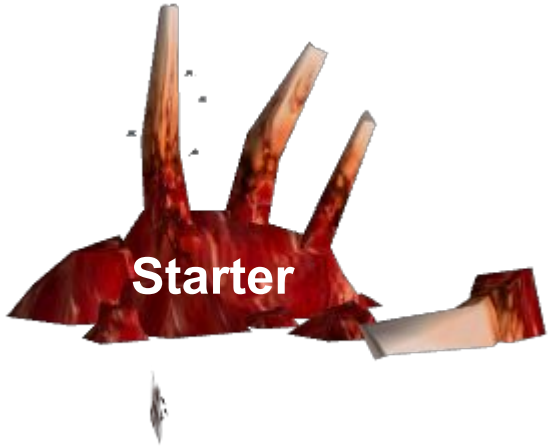
- Мои репозитории больше не обнаруживаются
- Кто виноват и Что делать?


Не поставил
@EnableJpaRepositoreis
Не поставил
@EnableJpaRepositoreis
Какой-то
левый
Starter


Не поставил
@EnableJpaRepositoreis
Не поставил
@EnableJpaRepositoreis
Какой-то
левый
Starter

Не поставил
@EnableJpaRepositoreis

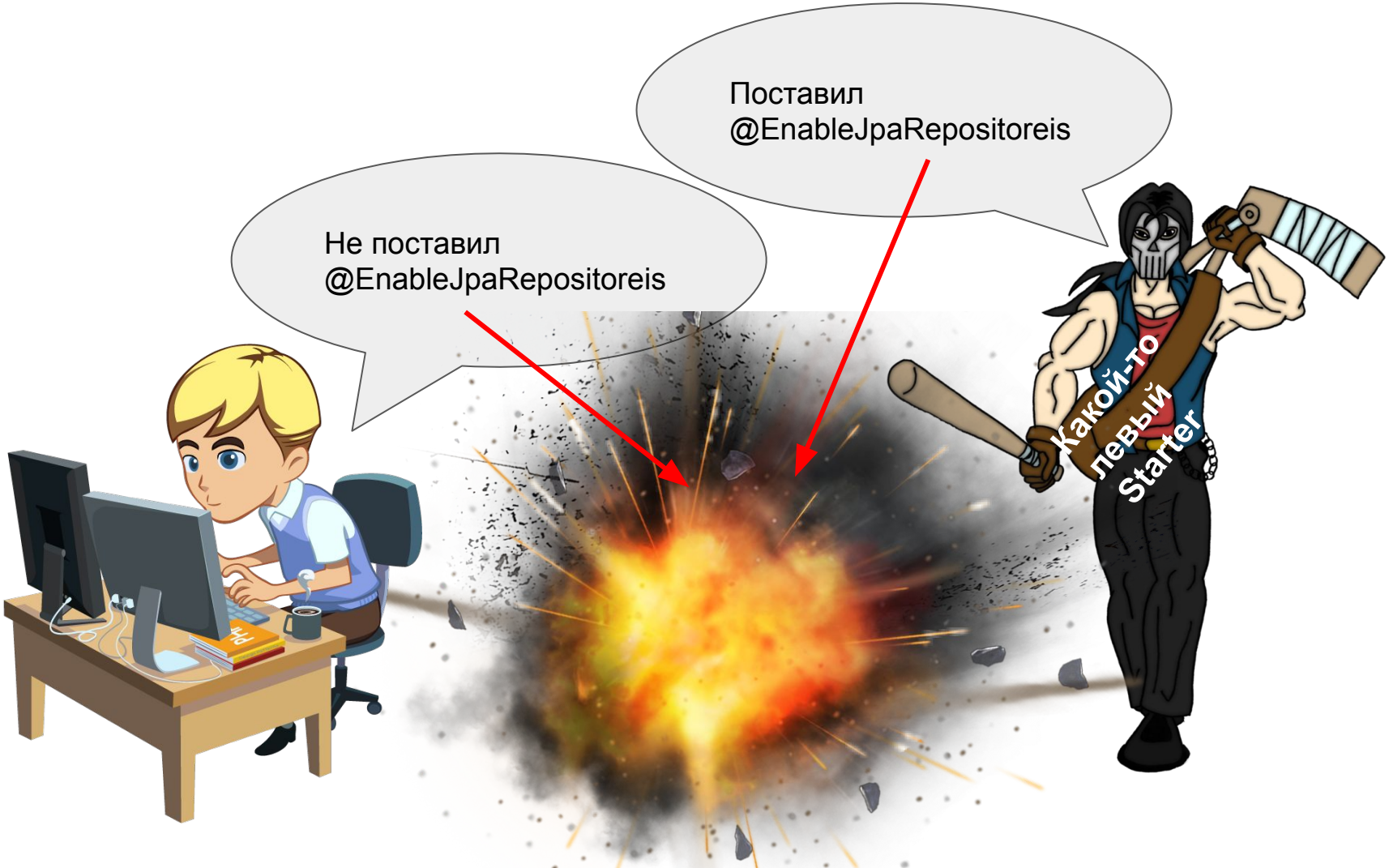
Не поставил
@EnableJpaRepositoreis




Не поставил
@EnableJpaRepositoreis
Поставил
@EnableJpaRepositoreis
Какой-то
левый
Starter


Не поставил
@EnableJpaRepositoreis
Поставил
@EnableJpaRepositoreis
и указал что
сканировать
Какой-то
левый
Starter

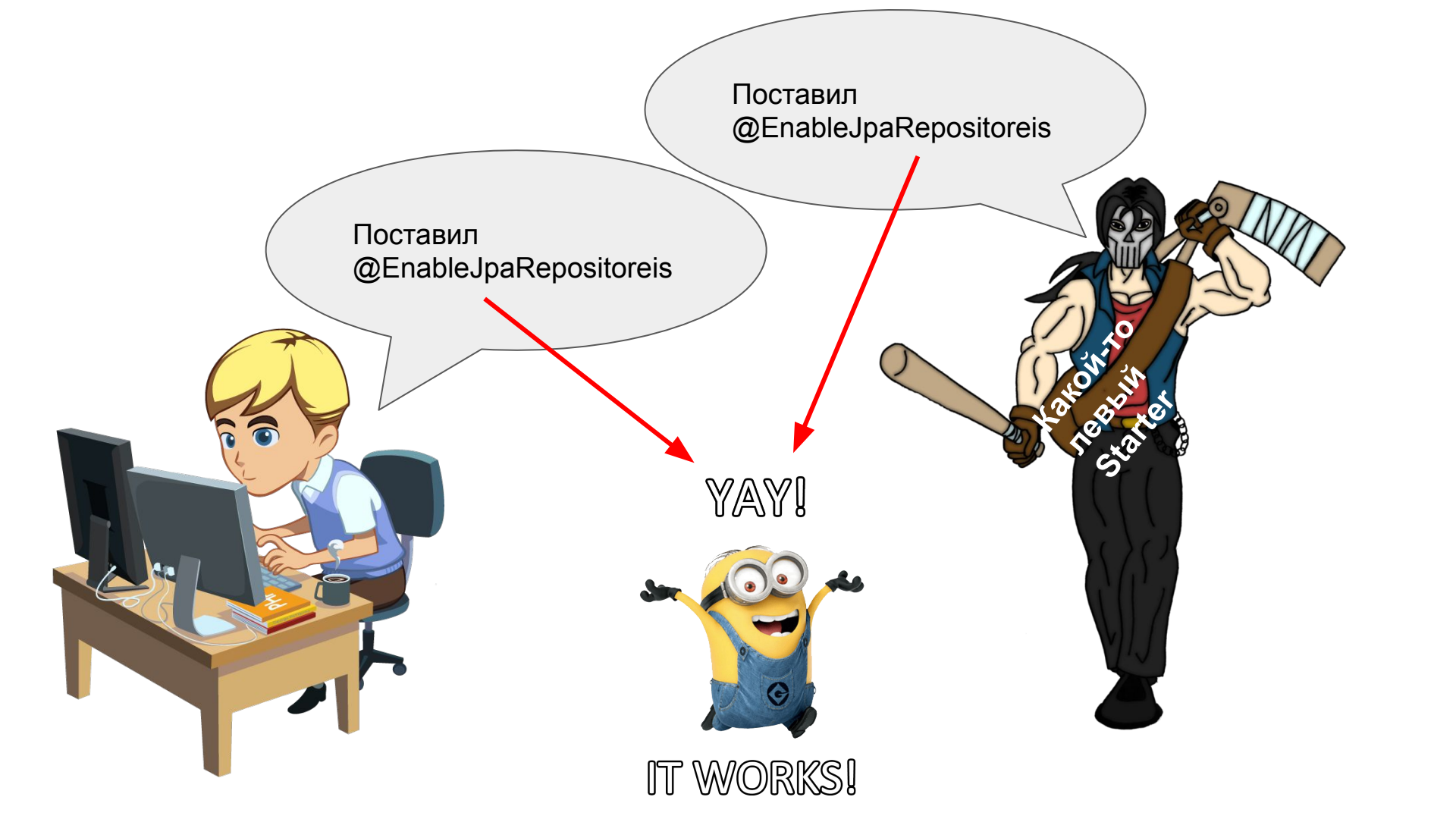

Не поставил
@EnableJpaRepositoreis
Поставил
@EnableJpaRepositoreis
и указал что
сканировать
Starter


Поставил
@EnableJpaRepositoreis
Не поставил
@EnableJpaRepositoreis
Какой-то
левый
Starter

Поставил
@EnableJpaRepositoreis

Не поставил
@EnableJpaRepositoreis


Поставил
@EnableJpaRepositoreis
и указал что
сканировать
Не поставил
@EnableJpaRepositoreis
Какой-то
левый
Starter
RIP


Поставил
@EnableJpaRepositoreis
Поставил
@EnableJpaRepositoreis
YAY!
IT WORKS!
Какой-то
левый
Starter

Поставил
@EnableJpaRepositoreis

Поставил
@EnableJpaRepositoreis
и указал что
сканировать

YAY!

IT WORKS!


Поставил
@EnableJpaRepositoreis
Поставил
@EnableJpaRepositoreis
и указал что
сканировать
Какой-то
левый
Starter
YAY!
IT WORKS!


Поставил @EnableJpaRepositoreis и указал что сканировать
Поставил @EnableJpaRepositoreis и указал что сканировать
YAY!
IT WORKS!
Какой-то левый Starter

Так а какие варианты решения?

Так а какие варианты решения?

1. Изменить мир к лучшему

Так а какие варианты решения?

1. Изменить мир к лучшему
2. Грязный хак

Так а какие варианты решения?

1. Изменить мир к лучшему
2. Грязный хак
3. Чистый хак

Проблема вторая

Caused by: java.lang.IllegalArgumentException: Not a managed type: class com.spring.crud.demo.model.User

Это почему?

А вот почему:

Как решаем?

Также, но с

```
@EntityScan
```

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Import(EntityScanPackages.Registrar.class)
public @interface EntityScan {
```

Проблема третья

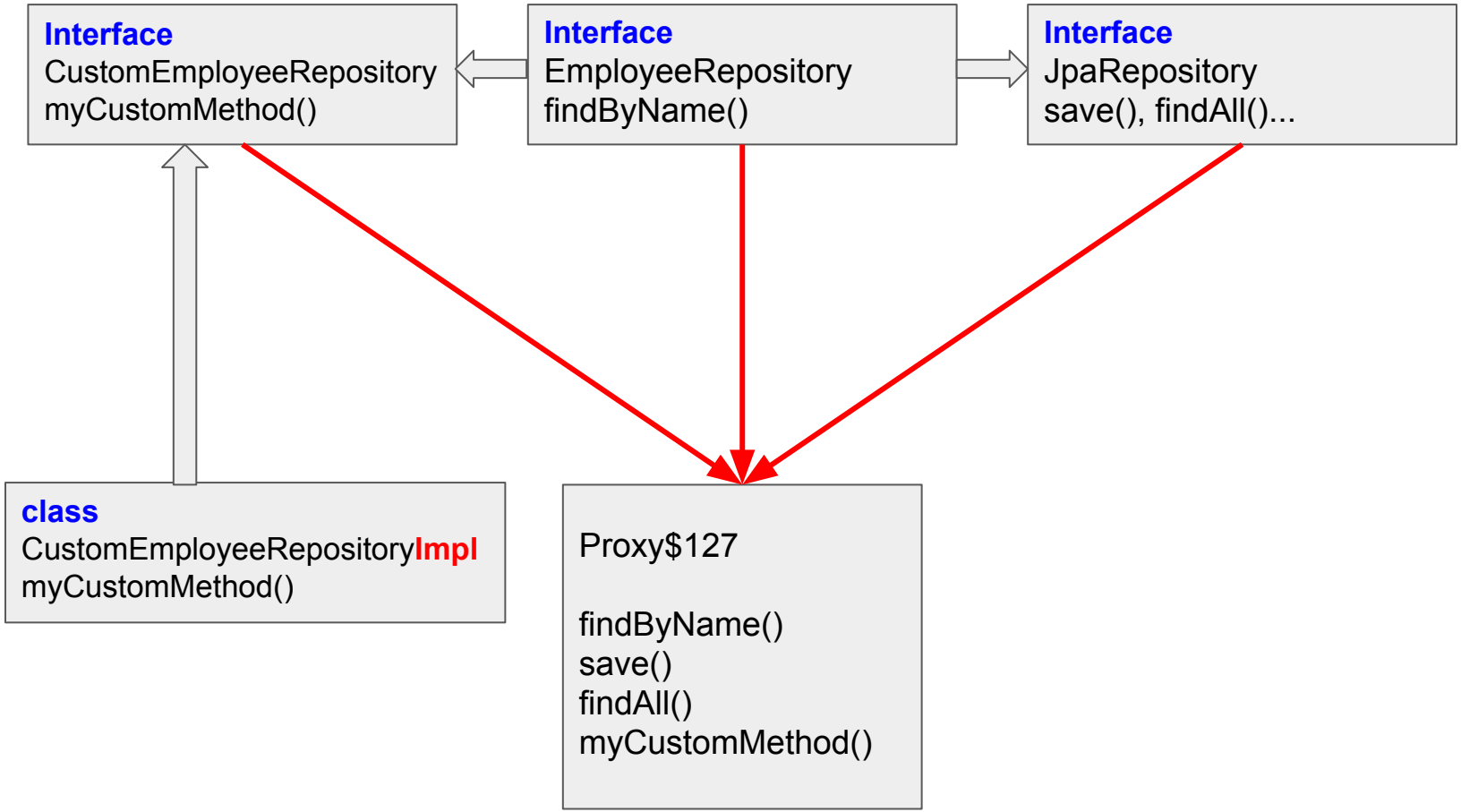
Проблема третья

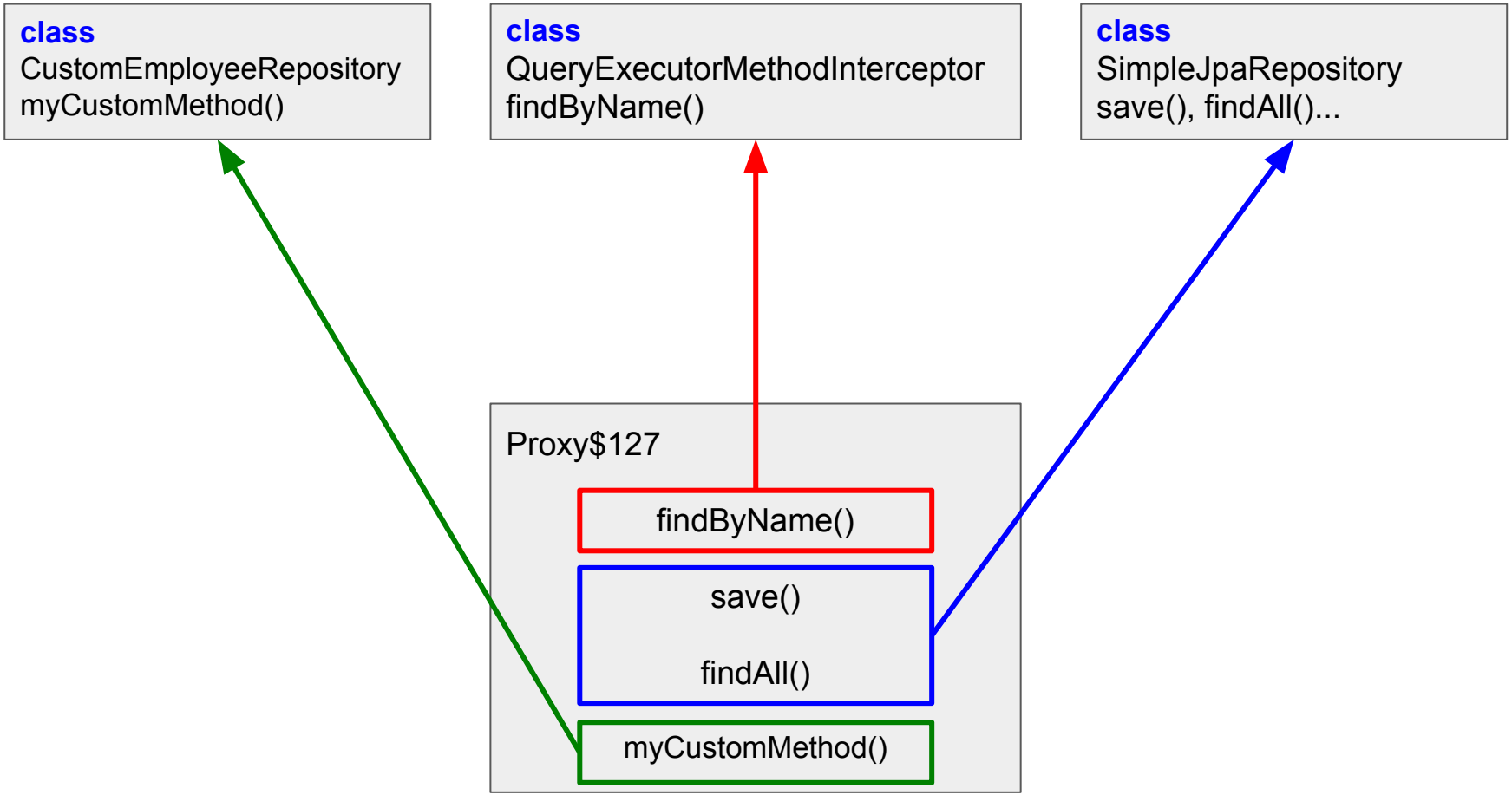
1. Как часть методов реализовать в ручную?

Проблема третья

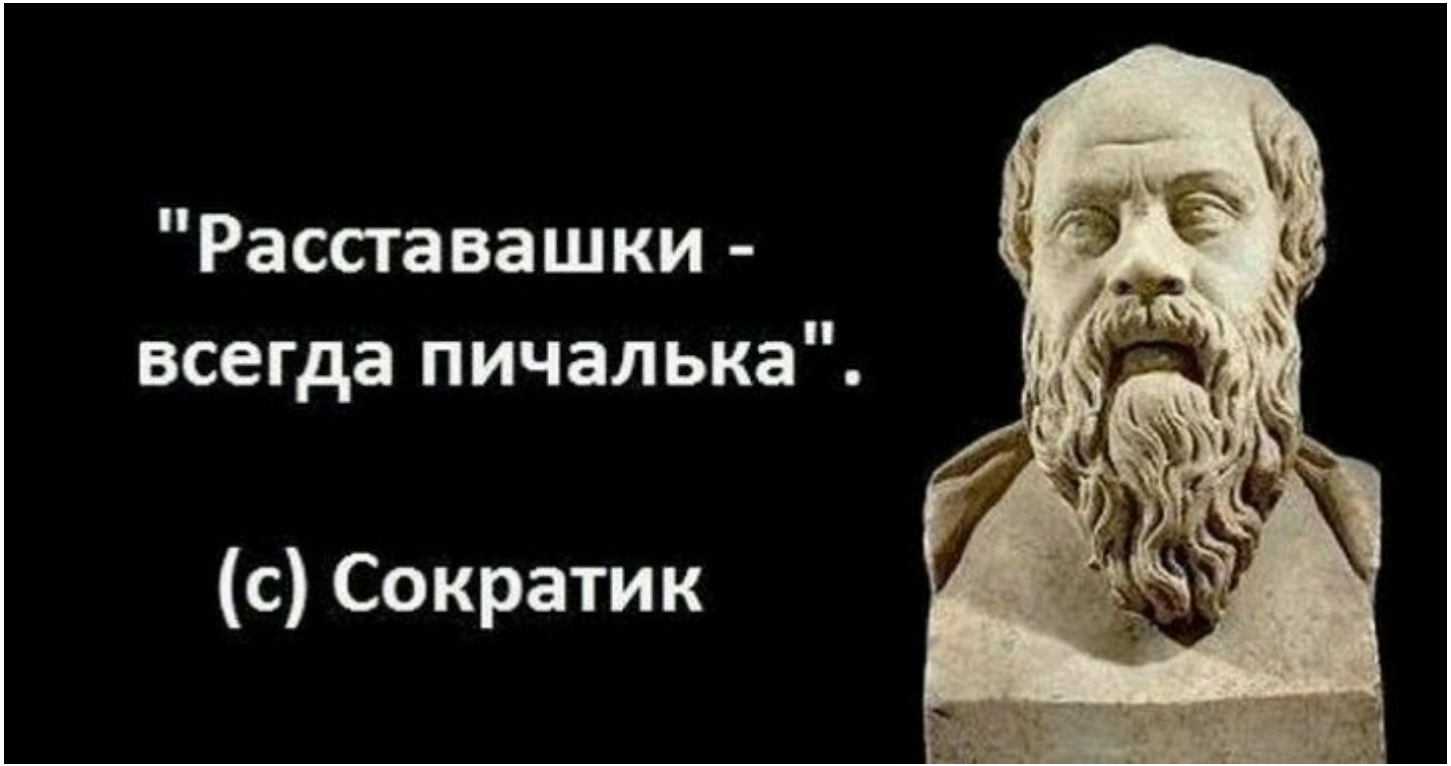
1. Как часть методов реализовать в ручную?

- Как быть и где писать?


Interface
CustomEmployeeRepository
myCustomMethod()
Interface
EmployeeRepository
findByName()
Interface
JpaRepository
save(), findAll()...
class
CustomEmployeeRepositoryImpl
myCustomMethod()
Proxy$127
findByName()
save()
findAll()
myCustomMethod()

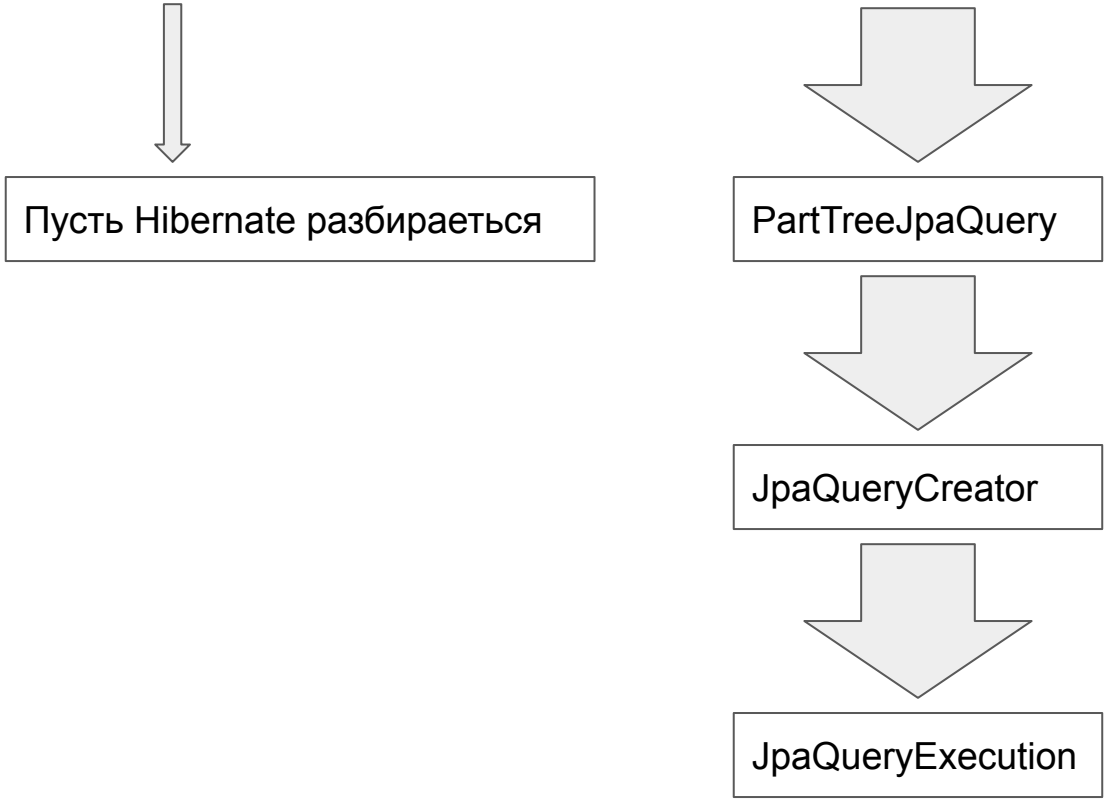

class
CustomEmployeeRepository
myCustomMethod()
class
QueryExecutorMethodInterceptor
findByName()
class
SimpleJpaRepository
save(), findAll()...
Proxy$127
findByName()
save()
findAll()
myCustomMethod()

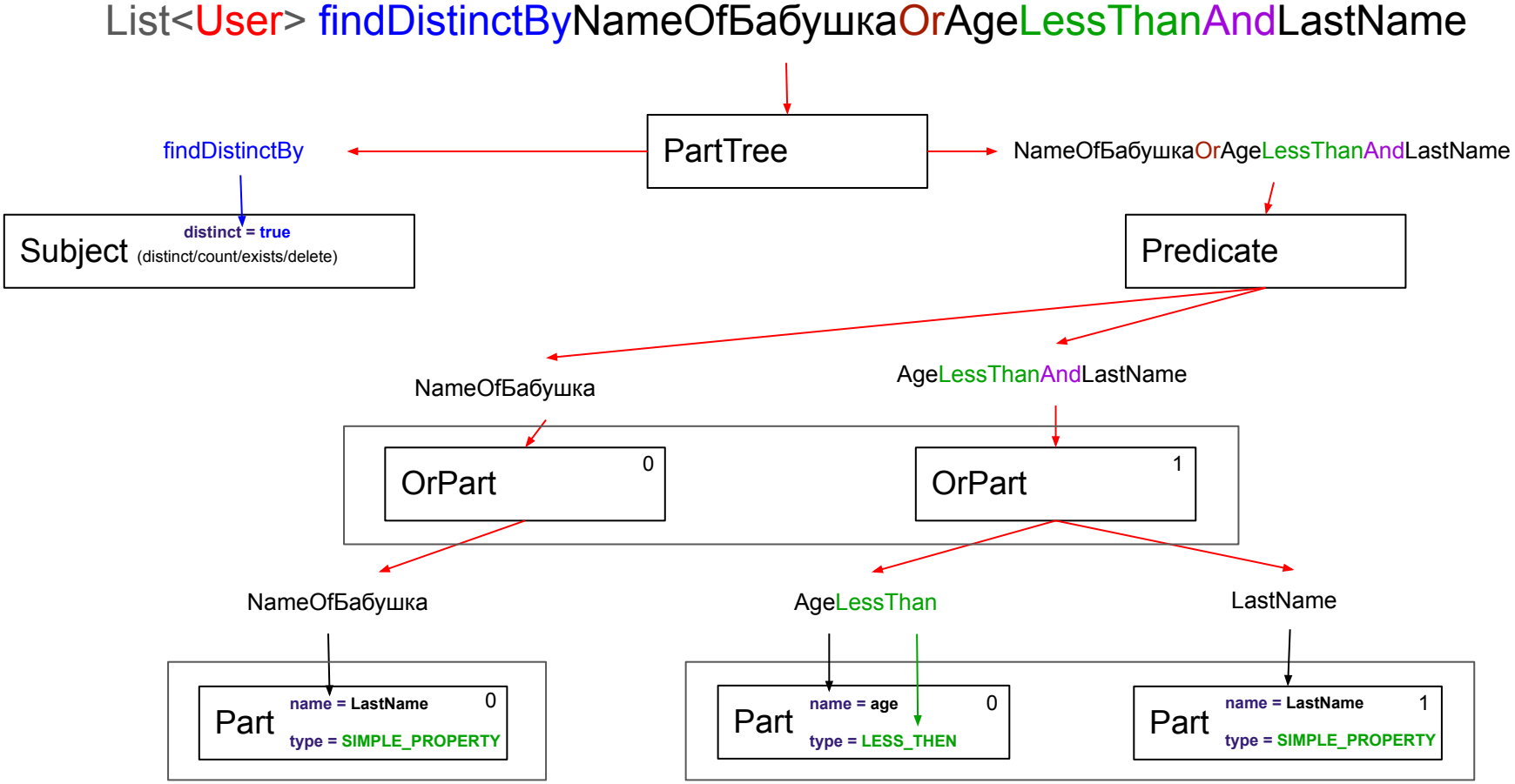
Всем спасибо



Спасибо Гене, который потрошил Spring Data

List<User> findByNameOfБабушкаEqualsOrAgeLessThanSortByAgeAndNameCount




List<User> findDistinctByNameOfБабушкаOrAgeLessThanAndLastName
PartTree
findDistinctBy
NameOfБабушкаOrAgeLessThanAndLastName
Subject
distinct = true
(distinct/count/exists/delete)
Predicate
NameOfБабушка
AgeLessThanAndLastName
OrPart
0
OrPart
1
NameOfБабушка
AgeLessThan
LastName
Part
name = LastName
0
type = SIMPLE_PROPERTY
Part
name = age
0
type = LESS_THEN
Part
name = LastName
1
type = SIMPLE_PROPERTY

- У меня не стоит ничего
- И у стартера не стоит - стартер не айдет свои репозитории найдет только если его пакеты совпадают с пакетами нашего проекта
- Картинка разроботчика и стартер это фиксик
- Поставил аннотацию в своей конфигурациии
- Поставил аннотацию в своей конфигурациии и прописал что сканировать
- Список решений - 1. (Измени мир у лучшему)ставте аннотацию всегда 2. (Грязный хак)Надо что бы совпадали пакеты 3. Решаем точечно проблему(вернись в реальный :((( )

Caused by: java.lang.IllegalArgumentException: Not a managed type: class com.spring.crud.demo.model.Student

at org.hibernate.metamodel.internal.MetamodelImpl.managedType(MetamodelImpl.java:582) ~[hibernate-core-5.4.18.Final.jar:5.4.18.Final]

at org.hibernate.metamodel.internal.MetamodelImpl.managedType(MetamodelImpl.java:85) ~[hibernate-core-5.4.18.Final.jar:5.4.18.Final]

```
@Target(ElementType.TYPE)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Import(EntityScanPackages.Registrar.class)
public @interface EntityScan {
```

JpaRepositoryRegistrar

JpaRepositoryRegistrar

JpaRepositoryConfigExtension

знает про фабрику, которая строит repository beans

AnnotationRepositoryConfigurationSource

знает какие пакеты сканировать

BeanDefinitionBuilder

собирает все в кучу

RepositoryComponentProvider

ищет все Repository

JpaRepositoriesRegistrar flow

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```

AnnotationMetadata

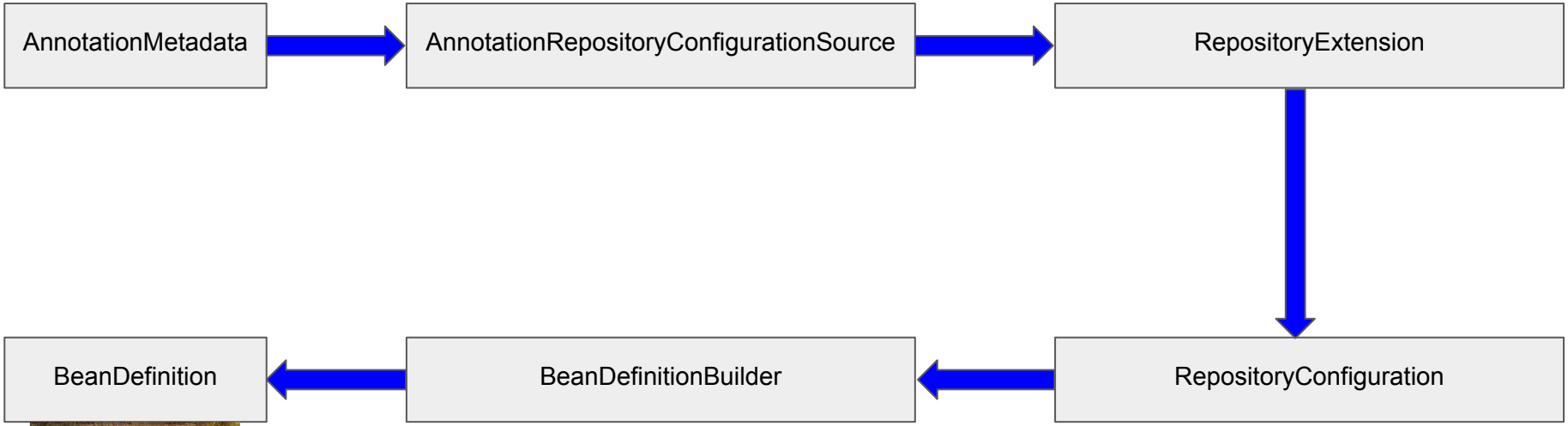
importingClassMetadata

Result:

- result = {SimpleAnnotationMetadata@4494}
  - className = "org.springframework.boot.autoconfigure.data.jpa.JpaRepositoriesAutoConfiguration"
  - access = 33
  - enclosingClassName = null
  - superClassName = "java.lang.Object"
  - independentInnerClass = false
  - interfaceNames = {String[0]@5295}
  - memberClassNames = {String[1]@5296}
  - annotatedMethods = {MethodMetadata[1]@5297}
  - annotations = {MergedAnnotationsCollection@5298}
  - annotationTypes = {Collections$UnmodifiableSet@5299} size = 7
    - 0 = "org.springframework.context.annotation.Configuration"
    - 1 = "org.springframework.boot.autoconfigure.condition.ConditionalOnBean"
    - 2 = "org.springframework.boot.autoconfigure.condition.ConditionalOnClass"
    - 3 = "org.springframework.boot.autoconfigure.condition.ConditionalOnMissingBean"
    - 4 = "org.springframework.boot.autoconfigure.condition.ConditionalOnProperty"
    - 5 = "org.springframework.context.annotation.Import"
    - 6 = "org.springframework.boot.autoconfigure.AutoConfigureAfter"

JpaRepositoriesRegistrar flow

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```




RepositoryExtension

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```

```
void registerBeanDefinitions(AnnotationMetadata metadata, registry...)
```


RepositoryExtension
RepositoryConfiguration


RepositoryExtension
BeanDefinitionBuilder


RepositoryExtension
Рецепт Боба