Anyframe Idgen Plugin

Version 1.6.0

I.Introduction

idgen plugin은 Unique한 ID를 생성하기 위한 기능의 라이브러리들로 구성되어 있다.

Installation

Command 창에서 다음과 같이 명령어를 입력하여 idgen plugin을 설치한다.

```
mvn anyframe:install -Dname=idgen
```

installed(mvn anyframe:installed) 혹은 jetty:run(mvn clean jetty:run) command를 이용하여 설치 결과를 확인해볼 수 있다.

Plugin Name	Version Range
core [http://dev.anyframejava.org/docs/anyframe/plugin/essential/core/1.6.0/reference/htmlsingle/core.html]	2.0.0 > * > 1.4.0

II.Id Generation

시스템 개발 시 공통적으로 많이 쓰이는 기능 중의 하나로, 유일한 ID를 생성하기 위해 사용할 수 있다. 주로 데이터베이스에 새로운 레코드를 추가하기 위해 ID를 받거나 혹은 파일 시스템에 새로이 파일을 생성하고자 할때 유일한 파일명을 보장 받기 위해 유일한 ID가 필요하다. Id Generation 서비스에 대한 구현체는 3가지이며, 다음은 각 구현체별 사용법이다.

1.UUIdGenService

새로운 ID를 생성하기 위해 UUID 생성 알고리즘을 이용하여 16 바이트 길이의 ID를 생성한다. 다음과 같은 설정을 가질 수 있다.

Property Name	Description	Required	Default Value
address	물리적 네트웍 공간 상의 유일성을 보장하기 위해 IP 주소나 네트웍 카드의 MAC 주소를 값으로 지정한다. IP 주소일 경우 필드간의 구분은 "."로 하고, MAC 주소일 경우는 ":"로 한다. 지정하지 않으면 UUIdGenService는 내부에서 MAC 주소를 랜덤하게 생성한다.	False	랜덤 생성

1.1.Implementation

- **Configuration**

  다음은 IdGernerationService에서 사용할 UUIdGenService와 그 속성을 정의한 context-idgen.xml 의 일부이다. MAC Address를 기반으로 유일한 Id를 생성하는 UUIdGenService Bean을 정의하고 있다.

```
<bean name="UUIdGenService" class="org.anyframe.idgen.impl.UUIdGenServiceImpl">
    <property name="address" value="00:00:F0:79:19:5B"/>
</bean>
```

- **TestCase**

  다음은 앞서 정의한 속성 설정을 기반으로 하여 UUIdGenService로부터 유일한 id를 추출하는 IdgenServiceTest.java 코드의 일부이다.

```
@Inject
@Named("UUIdGenService")
IdGenService uuid;

/**
 * [Flow #-1] Positive : try to get next String id and BigDecimal id.
 *
 * @throws Exception
 *           fail to test
 */
@Test
public void testUUIdGenService() {
    // 1. get next String id
    for (int i = 0; i < 10; i++) {
        assertNotNull(uuid.getNextStringId());
    }
    // 2. get next BigDecimal id
    for (int i = 0; i < 10; i++) {
        assertNotNull(uuid.getNextBigDecimalId());
    }
}
```

2.SequenceIdGenService

새로운 ID를 생성하기 위해 Database의 SEQUENCE 문을 사용하는 서비스이다. 다음과 같은 설정을 가질 수 있다.

Property Name	Description	Required	Default Value
dataSource	SequenceIdGenService를 사용하는 경우 필요하다. Database 연결을 위한 DataSourceService를 지정한다.	Y	N/A
useBigDecimals	long 타입의 ID 대신 BigDecimal 타입의 ID를 사용하고자 할 때 정의한다.	N	false
query	SEQUENCE SQL을 value로써 지정한다.(Database에 의존적이다.)	Y	N/A

2.1.Implementation

- **Configuration**

  다음은 SequenceIdGenService의 속성을 정의한 context-idgen.xml 의 일부이다. SequenceIdGenService는 common_datasource를 통하여 Connection 객체를 얻게 되며, IDGEN_SEQ라는 이름의 Sequence를 사용한다. 따라서, 다음 속성 기반의 SequenceIdGenService 실행을 위해서는 IDGEN_SEQ라는 이름의 Sequence가 생성되어 있어야 한다.

```
<bean name="SequenceIdGenService" class="org.anyframe.idgen.impl.SequenceIdGenServiceImpl"
 destroy-method="destroy">
    <property name="dataSource" ref="dataSource" />
    <property name="query" value="SELECT NEXT VALUE FOR IDGEN_SEQ FROM IDS"/>
</bean>
```

- **TestCase**

  다음은 앞서 정의한 속성 설정을 기반으로 하여 SequenceIdGenService를 이용하여 유일한 id를 추출하는 IdgenServiceTest.java 코드의 일부이다.

```
@Inject
@Named("SequenceIdGenService")
IdGenService sequenceid;

/**
 * [Flow #-4] Positive Case : try to get next Long id
 *
 * @throws Exception
 *             fail to test
 */
@Test
public void testSequenceIdGenServiceTest() {
    // 1. get id twice to compare with each other
    String id1 = sequenceid.getNextStringId();
    String id2 = sequenceid.getNextStringId();
    assertEquals("fail to get differenct id", true, !id1.equals(id2));
}
```

3.TableIdGenService

TableIdGenService는 ID를 관리하는 특정 테이블을 이용하여 유일한 ID를 얻는다. ID 관리를 위한 이 테이블은 id를 생성할 테이블이 어디인지를 명시하는 varchar 타입의 컬럼(디폴트 컬럼명 table_name)과 해당 테이블 아이디 발급 시 사용될 값을 저장하는 integer 타입의 컬럼 (디폴트 컬럼명 next_id)이 필요하다. 신규 id를 할당받은 후에 next_id 칼럼의 값은 TableIdGenService를 통해 신규 id 값으로 변경된다. (단, 신규 id를 얻고자 할 시점에, ID 발급 대상 테이블과 관련된 순번이 ID 관리 테이블에 저장되어 있지 않으면 TableIdGenService는 ID 발급 대상 테이블 이름과 순번 '1'을 ID 관리 테이블에 추가하는 작업을 수행한다.) 또한 Generation Strategy 을 적용할 경우, TableIdGenSimple Service는 신규 id에 정의된 Strategy를 적용한 결과값을 전달해준다. TableIdGenSimple Service는 다음과 같은 설정을 가질 수 있다.

Property Name	Description	Required	Default Value
dataSource	TableIdGenSimple Service를 사용하는 경우 필요하다. Database 연결을 위한 DataSourceService를 지정한다.	Y	N/A
strategy	Id Generation Strategy를 지정한다. 기본적으로 prefix 및 suffix, 신규 Id, paddingChar를 조합한 Id를 전달하는 MixStrategy를 사용한다.	N	N/A
blockSize	ID를 발급할 때마다 매번 데이터베이스에 접속한다면 시스템 성능 저하를 가져오므로, 한번에 TableIdGenService 내에서 발급 받아올 ID의 개수를 정할 수 있다.	N	10
table	발급한 ID를 관리하기 위한 테이블 이름.	N	ids
key	어떤 것에 발급하는 ID인지를 저장하기 위한 키 이름. 예를 들어, Order와 Product, 두 가지에 ID를 발급할 상황이라면 하나는 key 속성값으로 “order”, 또다른 하나는 "product"라고 지정할 수 있다. 이 경우 TableIdGenService을 구현 클래스로 하는 2개의 서로 다른 Bean이 정의되어야 하므로 속성 정의 간편화를 위해서는 관련 메소드 호출시 발급 대상을 입력 인자로 전달할 수도 있다.	N	□
useBigDecimals	Long 타입의 ID 대신 BigDecimal 타입의 ID를 사용하고자 할 때 쓴다.	N	false
keyColumn	신규 id를 발급하는 테이블의 이름을 저장하기 위한 컬럼의 이름. (컬럼 타입은 char 혹은 varchar)	N	table_name
nextValueColumn	발급된 최신 ID를 저장하기 위한 컬럼의 이름. (컬럼 타입은 integer 타입)	N	next_id

3.1.Implementation

3.1.1.기본적인 설정 및 사용방법

- **Configuration**

  다음은 TableIdGenService의 속성을 정의한 context-idgen.xml의 일부이다. TableIdGenService는 기본적으로 mix Strategy를 적용하고 있으며, Genre 테이블에 유일한 id를 제공하기 위해 IDS라는 ID 관리 테이블을 사용할 것이다. 다음 속성 기반의 TableIdGenService 실행을 위해서는 IDS라는 테이블에 [TABLE_NAME:**IDGEN_MOVIE**, NEXT_ID:초기값]와 같은 정보가 추가되어 있다라고 전제해 보자.

```
<bean name="TableIdGenService" class="org.anyframe.idgen.impl.TableIdGenServiceImpl"
 destroy-method="destroy">
    <property name="dataSource" ref="dataSource" />
```

```
    <property name="blockSize" value="1"/>
    <property name="table" value="IDS"/>
</bean>
```

- **TestCase**

  다음은 앞서 정의한 속성 설정을 기반으로 하여 TableIdGenService를 이용하여 유일한 id를 추출하는 IdgenServiceTest.java 코드의 일부이다.

```
@Test
public void testSimpleTableId() {
    // 1. get id twice to compare with each other
    String id1 = tableidSimple.getNextStringId("IDGEN_MOVIE");
    String id2 = tableidSimple.getNextStringId("IDGEN_MOVIE");
    assertEquals("fail to get differenct id", true, !id1.equals(id2));
}
```

3.1.2.사용자 정의 컬럼 설정 방법

TableIdGenService에서 기본적으로 사용하는 컬럼명이 아닌 프로젝트에서 정의한 다른 컬럼명을 사용하기 위해서는 keyColumn, nextValueColumn 속성 정의가 요구된다.

- **Configuration**

  다음은 TableIdGenService의 속성을 정의한 설정 파일의 일부로써 사용자 정의 컬럼에 대한 정의를 포함하고 있다.

```
<bean name="tableIdGenWithCustomColumn"
 class="org.anyframe.idgen.impl.TableIdGenServiceImpl" destroy-method="destroy">
    <property name="dataSource" ref="dataSource"/>
    <property name="blockSize" value="1"/>
    <property name="table" value="MY_IDS"/>
    <property name="key" value="test"/>
    <property name="keyColumn" value="MY_KEY"/>
    <property name="nextValueColumn" value="MY_ID"/>
</bean>
```

  위 속성 정의에 의하면 TableIdGenService 실행을 위해서 MY_KEY, MY_ID라는 컬럼을 가진 MY_IDS라는 테이블이 존재해야 할 것이다.

- **TestCase**

  다음은 앞서 정의한 설정을 기반으로 TableIdGenService를 이용하여 유일한 id를 추출하는 IdgenServiceTest.java 코드의 일부이다.

```
@Test
public void testWithCumstomColumn() {
    // 1. get id twice to compare with each other
    String id1 = tableidCustomColumn.getNextStringId();
    String id2 = tableidCustomColumn.getNextStringId();

    assertEquals("fail to get differenct id", true, !id1.equals(id2));
}
```

  위 코드에 의하면 keyColumn('MY_KEY')으로부터 key(**'test'**)의 속성값에 해당하는 테이블을 대상으로 nextId 값을 추출하게 될 것이다. 따라서 테이블명을 구분하기 위한 별도 입력 인자없이 getNextStringId()를 호출할 수 있게 된다.

4.How to use a Generation Strategy

TableIdGenService를 사용할 때, 유일한 id 생성 시 generation strategy를 적용할 수 있으며, 이는 DB 기반으로 채번된 ID에 특정한 문자열을 덧붙여주는 역할을 한다. 먼저, 사용자가 정의한 문자열인 prefix 또는 suffix를 순번과 조합하는 MixStrategy와 정의된 포맷의 현재 시간과 순번을 조합하는 TimestampStrategy를 제공하며, 또한 Id Generation Service를 호출하는 클래스의 이름과 순번을 조합하는 ClassNameStrategy와 패키지명과 순번을 조합하는 PackageNameStrategy, 패키지에 따라 사용자가 정의한 문자열을 순번과 조합하는 PackageStrategy도 제공하고 있다. 위에서 언급한 클래스명, 패키지명, 패키지에 따른 사용자가 정의한 문자열은 모두 prefix 또는 suffix로 사용할 수 있다.

Strategy Name	Description	Prefix	Suffix	Prefix, Suffix 동시 적용
MixPrefixStrategy	사용자가 정의한 문자열인 prefix를 순번과 조합한다.	O	X	X
MixStrategy	사용자가 정의한 문자열인 prefix 혹은 suffix를 순번과 조합한다	O	O	O
TimestampStrategy	사용자가 정의한 포맷의 현재 시간과 순번을 조합한다.	O	O	X
ClassNameStrategy	Id Generation Service를 호출하는 클래스의 이름과 순번을 조합한다.	O	O	X
PackageNameStrategy	Id Generation Service를 호출하는 클래스가 속하는 패키지명과 순번을 조합한다.	O	O	X
PackageStrategy	Id Generation Service를 호출하는 클래스 속하는 패키지에 따라 사용자가 정의한 문자열을 순번과 조합한다.	O	O	X

Generation Strategy는 아래와 같이 TableIdGenService를 통해 참조될 수 있도록 정의해 주어야 한다.

- **dependency injection을 사용한 strategy 참조**

```
<bean name="tableIdGenWithFix" class="org.anyframe.idgen.impl.TableIdGenServiceImpl"
 destroy-method="destroy">
    <property name="dataSource" ref="dataSource" />
    <property name="strategy" ref="mixStrategy" />
    <property name="blockSize" value="1"/>
    <property name="table" value="IDS"/>
</bean>

<bean name="mixStrategy" class="org.anyframe.idgen.impl.strategy.MixStrategy">
 <property name="prefix" value="TEST-" />
 <property name="maxCiphers" value="5" />
 <property name="paddingChar" value="*" />
</bean>
```

4.1.MixPrefixStrategy

Anyframe의 TableIdGenService에서는 org.anyframe.idgen.impl.strategy.MixPrefixStrategy라는 generation strategy를 제공한다. MixPrefixStrategy는 id generation strategy의 한 종류로, id 생성 시 TableIdGenService 실행을 통해 추출된 순번과 prefix를 조합한다. 예를 들어 prefix가 'TEST-', TableIdGenService를 통해 전달된 순번이 '12', paddingChar가 '0', maxCiphers가 5일 경우 전달된 id의

값은 prexfix와 순번을 조합하여 'TEST-00012'가 된다. 이 때, 자릿수(maxCiphers)가 5, 순번이 '12'이면 순번은 5자리로 구성하되 빈 부분은 paddingChar로 정의된 문자열로 채우기 때문에 '00012'와 같은 형태가 된다.

```
<bean name="mixPrefix" class="org.anyframe.idgen.impl.strategy.MixPrefixStrategy">
    <property name="prefix" value="TEST-"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
</bean>
```

MixPrefixStrategy 기능 더이상 제공하지 않음.

MixPrefisStrategy는 suffix의 기능을 제공하지 않고 prefix의 기능만 제공한다. 이에 Id Generation Service 1.5.1버전에서는 MixPrefixStrategy 클래스를 Deprecated 시키고, MixPrefixStrategy에 suffix 기능을 추가한 MixStrategy를 사용하도록 한다. (MixStrategy의 자세한 설명은 아래의 MixStrategy를 참고한다.)

4.2.TimestampStrategy

TimestampStrategy는 id generation strategy의 한 종류로, id 생성시 TableIdGenService 실행을 통해 추출된 순번과 javax.text.SimpleDateFormat pattern으로 생성된 현재 시간을 조합한다. separator 속성을 정의하여 현재 시간과 DB기반으로 생성된 Id의 구분자로 이용할 수 있다. 예를 들어 prefix가 'true', pattern이 'yyyy-MM-dd', TableIdGenService를 통해 전달된 순번이 '12', separator가 '-', paddingChar가 '0', maxCiphers가 5일 경우 전달된 id의 값은 현재 시간과 순번을 조합하여 '2011-05-24-00012'와 같은 형태가 된다. pattern과 separator가 입력되지 않았을 경우 기본적으로 'yyyyMMdd', ''과 같은 값이 적용되어 신규 id가 조합됨을 알아 두도록 한다.

```
<bean name="timestamp" class="org.anyframe.idgen.impl.strategy.TimestampStrategy">
    <property name="prefix" value="true"/>
    <property name="pattern" value="yyyy-MM-dd"/>
    <property name="separator" value="-"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
</bean>
```

4.3.MixStrategy

MixStrategy는 MixPrefixStrategy의 기능에 suffix 기능을 추가한 클래스로 id 생성시 TableIdGenService 실행을 통해 추출된 순번을 prefix 또는 suffix 또는 prefix+suffix와 조합한다. 예를 들어 prefix가 'TEST-'이고 suffix가 '-PROJECT', TableIdGenService를 통해 전달된 순번이 '12', paddingChar가 '0', maxCiphers가 '5'일 경우 전달된 id의 값은 prefix, suffix와 순번을 조합하여 'TEST-00012-PROJECT' 된다. 또한 prefix와 suffix 중 하나만 사용하는 것도 가능하다.

```
<bean name="mixStrategyWithFix" class="org.anyframe.idgen.impl.strategy.MixStrategy">
    <property name="prefix" value="TEST-"/>
    <property name="suffix" value="-PROJECT"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
</bean>
```

maxCiphers는 기준이 되는 ID의 자릿수를 의미하지만, maxCiphers를 초과하는 ID가 채번되더라도 내부적으로는 이를 허용하고 있다. 예를들어, maxCiphers가 2이고 현재 '99'번까지 채번된 상태라도 다음 순서인 '100'번에 대해서도 채번할 수 있다. 이를 허용하지 않고 정의한 maxCiphers보다 초과되는 ID가 채번되는 것에 대해 제한하고 싶다면 ignoreMaxCiphers 속성을 false(default=true)로 설정하도록 한다.

```
<bean name="mixStrategyWithFix" class="org.anyframe.idgen.impl.strategy.MixStrategy">
```

```
    <property name="prefix" value="TEST-"/>
    <property name="suffix" value="-PROJECT"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
    <property name="ignoreMaxCiphers" value="false"/>
</bean>
```

이렇게 설정하게 되면 채번된 ID의 자릿수가 maxCiphers를 초가하게 되면 '[IDGeneration Service]ID cannot be have length more than maxCiphers.'라는 에러메시지를 출력해준다.

4.4.ClassNameStrategy

ClassNameStrategy는 id generation strategy의 한 종류로, id 생성시 TableIdGenService 실행을 통해 추출된 순번을 ClassName과 조합한다. ClassNameStrategy가 갖는 속성은 MixStrategy와 동일하며 이 때, 클래스 이름과 DB기반으로 생성된 ID를 구분해주기 위한 구분자를 'separator'속성을 통해 정의할 수 있다. 또한, prefix, suffix 여부를 각각의 속성에 'true' 또는 'false'로 선언할 수 있다. 예를 들어, 클래스명을 prefix로 조합하고 싶으면 'prefix' 속성을 'true'로 정의하고 suffix로 조합하고 싶으면 'suffix' 속성을 'true'로 정의하면 된다. 아래의 코드와 같이 정의된 ClassNameStrategy를 사용하여 ID 생성 기능을 사용하게 되면 ClassName이 TestServiceImpl이고 TableIdGenService를 통해 전달된 순번이 '12', prefix가 'true', paddingChar가 '0', maxCiphers가 '5', separator가 '-' 가 되어 전달된 id의 값은 이를 조합한 'TestServiceImpl-00012'가 된다.

```
<bean name="classStrategy" class="org.anyframe.idgen.impl.strategy.ClassNameStrategy">
    <property name="prefix" value="true"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
    <property name="separator" value="-"/>
</bean>
```

4.5.PackageNameStrategy

PackageNameStrategy는 id 생성시 TableIdGenService 실행을 통해 추출된 순번을 PackageName과 조합한다. PackageNameStrategy는 가지는 속성은 ClassNameStrategy와 동일하다. 예를 들어 class가 속하여 있는 PackageName이 com.sds.test.impl이고 TableIdGenService를 통해 전달된 순번이 '12', prefix가 'true', paddingChar가 '0', maxCiphers가 '5', separator가 '-'일 경우 전달된 id의 값은 ClassName 및 seperator와 순번을 조합하여 'com.sds.test.impl-00012'가 된다.

```
<bean name="packageNameStrategy"
 class="org.anyframe.idgen.impl.strategy.PackageNameStrategy">
    <property name="prefix" value="true"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
    <property name="separator" value="-"/>
</bean>
```

4.6.PackageStrategy

PackageStrategy는 id generation strategy의 한 종류로, id 생성시 TableIdGenService 실행을 통해 추출된 순번을 패키지별로 사용자가 정의한 문자열과 조합한다. PacageStrategy의 속성은 ClassNameStrategy와 동일하며 패키지별로 DB기반으로 생성된 문자열과 조합하는 문자열을 지정하기 위해서 'packageIds'라는 속성을 제공하며, 아래 코드와 같이 Properties의 형태로 정의한다. 아래와 같이 PackageStratgey를 설정하고 클래스가 com.sds.test.impl에 속해있을 때 이 PackageName에 대하여 prefix를 'TBP'가 되고 TableIdGenService를 통해 전달된 순번이 '12', prefix가 'true', paddingChar가 '0', maxCiphers가 '5', separator가 '-'일 경우 전달된 id의 값은 이와 조합되어 'TBP-00012'가 된다.

```
<bean name="packageStrategy" class="org.anyframe.idgen.impl.strategy.PackageStrategy">
    <property name="packageIds">
     <props>
      <prop key="com.sds.test.impl">TBP</prop>
     </props>
    <property name="prefix" value="true"/>
    <property name="maxCiphers" value="5"/>
    <property name="paddingChar" value="0"/>
    <property name="separator" value="-"/>
</bean>
```

4.7.Id Generation Strategy를 implements하는 방법

strategy를 개발하기 위해서는 org.anyframe.idgen.IdGenStrategy를 implemensts 해야한다. 또한 makeId(String originalId, Class<?> clazz) 메소드를 통해 특정 Strategy가 적용된 id를 return하도록 한다. IdGenStrategy를 구현한 클래스의 makeId의 구현 예는 다음과 같다.

```
public class MyStrategy implements IdGenStrategy{
    public String makeId(String originalId, Class<?> clazz) {
     return "myPrefix" + orininalId + "mySuffix";
    }
}
```