記事
Toshihiko Minamoto · 2022年4月5日 6m read

Open Exchange

データの匿名化、iris-Disguise の導入

まずは、データの匿名化とは何でしょうか？

ウィキペディアによると:

> **データの匿名化**は、情報サニタイゼーションの一種で、プライバシー保護を意図しています。データセットから個人を特定可能な情報を取り除き、データが説明するユーザーが匿名性を維持できるようにするプロセスです。

言い換えると、データの匿名化は、データを操作しながらソースを匿名に維持するプロセスと言えます。
採用された匿名化手法に応じて、データは編集、マスク、または置換されます。

そして、これが iris-Disguise の目的です。一連の匿名化ツールを提供します。

メソッドの実行、または永続クラスの定義自体で匿名化ストラテジーを指定する、2
つの方法で使用することができます。

iris-Disguise の最新のバージョンには、データを匿名化するためのストラテジーが 6 つ用意されています。

- **破壊**
- **スクランブル**
- **シャッフル**
- **部分マスキング**
- **ランダム化**
- **偽装**

それぞれのストラテジーを説明しましょう。例を使用してメソッドの実行を示し、前述のように、永続クラスの定義内に提供する方法も示します。この方法で **iris-Disguise** を使用するには、「 変装メガネをかける 」必要があります。永続クラスでは、 **dc.Disguise.Glasses** クラスを拡張し、指定したストラテジーでそのデータ型を持つプロパティを変更できます。
その後、任意の時点で、クラスに **DisguiseProcess** メソッドを呼び出すことができます。
すべての値は、そのデータ型のストラテジーを使用して置き換えられるようになります。

では、ベルトを締めて出発しましょう。

破壊

このストラテジーは、列全体ある単語（デフォルトは「CONFIDENTIAL」）に置き換えます。

```
Do ##class(dc.Disguise.Strategy).Destruction("classname", "propertyname", "Word to re
place")
```

3 つ目のパラメータはオプションです。
指定されていない場合は、「CONFIDENTIAL」という語が使用されます。

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As dc.Disguise.DataTypes.String(FieldStrategy = "DESTRUCTION");
}
```

```
Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name
1	Leonardo
2	Donatello
3	Michelangelo
4	Raphael

ID	Name
1	CONFIDENTIAL
2	CONFIDENTIAL
3	CONFIDENTIAL
4	CONFIDENTIAL

スクランブル

このストラテジーは、プロパティのすべての文字をスクランブルします。

```
Do ##class(dc.Disguise.Strategy).Scramble("classname", "propertyname")
```

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As dc.Disguise.DataTypes.String(FieldStrategy = "SCRAMBLE");
}
```

```
Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name
1	Leonardo
2	Donatello
3	Michelangelo
4	Raphael

ID	Name
1	enodaroL
2	ooelantID
3	lieanMclehog
4	aplehaR

シャッフル

シャッフルは、特定のプロパティのすべての値を再配置します。
これは「縦方向」に機能するため、マスキングとは異なります。
このストラテジーは、参照整合性が維持されるため、リレーションに役立ちます。
このバージョンまでは、このメソッドは **一対一の関係** でのみ動作していました。

```
Do ##class(dc.Disguise.Strategy).Shuffling("classname", "propertyname")
```

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As %String;
Property Weapon As dc.Disguise.DataTypes.String(FieldStrategy = "SHUFFLING");
}
```

```
Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name	Weapon
1	Leonardo	Katana
2	Donatello	Bo
3	Michelangelo	Nunchaku
4	Raphael	Sai

ID	Name	Weapon
1	Leonardo	Sai
2	Donatello	Nunchaku
3	Michelangelo	Katana
4	Raphael	Bo

部分マスキング

このストラテジーは、データの一部を難読化します。クレジットカードを例にすると、456X XXXX XXXX X783 のように置き換えられます。

```
Do ##class(dc.Disguise.Strategy).PartialMasking("classname", "propertyname", prefixLe
ngth, suffixLength, "mask")
```

PrefixLength、suffixLength、および mask はオプションです。
指定されていない場合、デフォルト値が使用されます。

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As %String;
Property SSN As dc.Disguise.DataTypes.PartialMaskString(prefixLength = 2, suffixLengt
h = 2);
Property Weapon As %String;
}
```

```
Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name	SSN	Weapon
1	Leonardo	440-15-5966	Katana
2	Donatello	574-15-7023	Bo
3	Michelangelo	417-29-9142	Nunchaku
4	Raphael	035-01-5028	Sai

ID	Name	SSN	Weapon
1	Leonardo	44X-XX-XX66	Katana
2	Donatello	57X-XX-XX23	Bo
3	Michelangelo	41X-XX-XX42	Nunchaku
4	Raphael	03X-XX-XX28	Sai

ランダム化

このストラテジーは純粋なランダムなデータを生成します。ランダム化には、整数、数値、および日付の 3 種類があります。

```
Do ##class(dc.Disguise.Strategy).Randomization("classname", "propertyname", "type", f
rom, to)
```

type: "integer"、"numeric"、または "date" です。デフォルトは "integer" です。

from と to はオプションです。ランダム化の範囲を定義します。integer 型の場合、デフォルトの範囲は 1~100 です。numeric 型の場合、デフォルトの範囲は 1.00~100.00 です。

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As %String;
Property Age As dc.Disguise.DataTypes.RandomInteger(MINVAL = 10, MAXVAL = 25);
Property SSN As %String;
Property Weapon As %String;
}


Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name	Weapon	Age
1	Leonardo	Katana	14
2	Donatello	Bo	14
3	Michelangelo	Nunchaku	13
4	Raphael	Sai	15

ID	Name	Weapon	Age
1	Leonardo	Katana	22
2	Donatello	Bo	11
3	Michelangelo	Nunchaku	17
4	Raphael	Sai	19

偽装データ

偽装は、データをランダムに置き換えながらもっともらしい値にすることを構想しています。**iris-Disguise** には、偽装データを生成する小さなセットのメソッドが用意されています。

```
Do ##class(dc.Disguise.Strategy).Fake("classname", "propertyname", "type")
```

type: "firstname"、"lastname"、"fullname"、"company"、"country"、"city"、および "email" です。

```
Class packageSample.FictionalCharacter Extends (%Persistent, dc.Disguise.Glasses)
{
Property Name As dc.Disguise.DataTypes.FakeString(FieldStrategy = "FIRSTNAME");
Property Age As %Integer;
Property SSN As %String;
Property Weapon As %String;
}


Do ##class(packageSample.FictionalCharacter).DisguiseProcess()
```

ID	Name	Weapon	SSN	Age
1	Leonardo	Katana	440-15-5966	14
2	Donatello	Bo	574-15-7023	14
3	Michelangelo	Nunchaku	417-29-9142	13
4	Raphael	Sai	035-01-5028	15

ID	Name	Weapon	SSN	Age
1	Quigley	Katana	440-15-5966	14
2	David	Bo	574-15-7023	14
3	Francis	Nunchaku	417-29-9142	13
4	Tara	Sai	035-01-5028	15

さんのご意見をお聞かせください!

フィードバックやアイデアを歓迎しています!

このツールの感想、ニーズにどのように適合しているのか、どの機能が不足しているかなどを教えてください。

iris-Disguise の作成者にインスピレーションを与えるコメント、レビュー、提案、意義のあるディスカッションに貢献していただいた @Henrique Dias、 @Oliver Wilms、 @Robert Cemper、 @Yuri Marx、および @Evgeny Shvarov に、特別にお礼を申し上げます。

#InterSystems IRIS
InterSystems Open Exchangeで関連アプリケーションを確認してください

ソースURL: https://jp.community.intersystems.com/post/%E3%83%87%E3%83%BC%E3%82%BF%E3%81%AE%E5%8C%BF%E5%90%8D%E5%8C%96%E3%80%81iris-disguise-%E3%81%AE%E5%B0%8E%E5%85%A5