分布式日志云脱敏方案

日志脱敏是常见的安全需求。普通的基于工具类方法的方式，对代码的入侵性太强。编写起来又特别麻烦。本项目提供基于注解的方式，并且内置了常见的脱敏方式，便于开发。

用户也可以基于自己的实际需要，自定义注解。

# 日志脱敏

为了金融交易的安全性，国家强制规定对于以下信息是要日志脱敏的：

1.用户名

2.手机号

3.邮箱

4.银行卡号

5.密码

# **持久化加密**

存储的时候上面的信息都需要加密，密码为不可逆加密，其他为可逆加密。类似的功能有很多。具体需求具体实现。

# **特性**

1.基于注解的日志脱敏。

2.可以自定义策略实现，策略生效条件。

3.常见的脱敏内置方案。

4.java 深拷贝，且原始对象不用实现任何接口。

5.支持用户自定义注解。

# **快速开始**

maven导入

|  |
| --- |
| <dependency>  <groupId>com.github.houbb</groupId>  <artifactId>sensitive-core</artifactId>  <version>0.0.5</version>  </dependency> |

## 1.对象脱敏

定义对象

|  |
| --- |
| package com.tzbank.desensitize;  import java.io.Serializable;  import com.github.houbb.sensitive.annotation.Sensitive;  import com.github.houbb.sensitive.core.api.strategory.StrategyCardId;  import com.github.houbb.sensitive.core.api.strategory.StrategyChineseName;  import com.github.houbb.sensitive.core.api.strategory.StrategyEmail;  import com.github.houbb.sensitive.core.api.strategory.StrategyPassword;  import com.github.houbb.sensitive.core.api.strategory.StrategyPhone;  /\*\*  \* @desc 定义客户信息，字段属性进行脱敏  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:20:39 PM  \*/  public class Customer implements Serializable {  private static final long serialVersionUID = 4419122083678875163L;  public Customer() {  }  public Customer(String username, String idCard, String password, String email, String phone) {  this.username = username;  this.idCard = idCard;  this.password = password;  this.email = email;  this.phone = phone;  }  @Sensitive(strategy = StrategyChineseName.class)  private String username;  @Sensitive(strategy = StrategyCardId.class)  private String idCard;  @Sensitive(strategy = StrategyPassword.class)  private String password;  @Sensitive(strategy = StrategyEmail.class)  private String email;  @Sensitive(strategy = StrategyPhone.class)  private String phone;  public String getUsername() {  return username;  }  public void setUsername(String username) {  this.username = username;  }  public String getIdCard() {  return idCard;  }  public void setIdCard(String idCard) {  this.idCard = idCard;  }  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  public String getEmail() {  return email;  }  public void setEmail(String email) {  this.email = email;  }  public String getPhone() {  return phone;  }  public void setPhone(String phone) {  this.phone = phone;  }  @Override  public String toString() {  return "Customer:{username=" + username + ",idCard=" + idCard + ",password=" + password + ",email=" + email  + ",phone=" + phone + "}";  }  } |

单元测试

|  |
| --- |
| package com.tzbank.desensitize;  import java.util.ArrayList;  import java.util.List;  import com.github.houbb.sensitive.core.api.SensitiveUtil;  /\*\*  \* @desc 针对对象或者对象集合的属性脱敏  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:22:30 PM  \*/  public class CustomerTest {  public static void main(String[] args) {  Customer cust1 = new Customer("张三", "349878909765152", "12345678", "zengbin163@163.com", "18867102687");  Customer cust2 = new Customer("李四", "341878909765153", "87654321", "zengbin164@163.com", "18867102688");  Customer cust3 = new Customer("王五", "342878909765154", "12348765", "zengbin165@163.com", "18867102689");  Customer cust4 = new Customer("赵六", "343878909765155", "43215678", "zengbin166@163.com", "18867102690");  Customer cust5 = new Customer("钱七", "344878909765156", "12783456", "zengbin167@163.com", "18867102691");  List<Customer> custList = new ArrayList<>();  custList.add(cust1);  custList.add(cust2);  custList.add(cust3);  custList.add(cust4);  custList.add(cust5);  for(Customer cust : custList) {  System.out.println("脱敏前原始： " + cust);  Customer sensitiveCust = SensitiveUtil.desCopy(cust);  System.out.println("脱敏后原始： " + sensitiveCust);  }  }  } |

测试结果

|  |
| --- |
| 脱敏前原始： Customer:{username=张三,idCard=349878909765152,password=12345678,email=zengbin163@163.com,phone=18867102687}  脱敏后原始： Customer:{username=\*三,idCard=349878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2687}  脱敏前原始： Customer:{username=李四,idCard=341878909765153,password=87654321,email=zengbin164@163.com,phone=18867102688}  脱敏后原始： Customer:{username=\*四,idCard=341878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2688}  脱敏前原始： Customer:{username=王五,idCard=342878909765154,password=12348765,email=zengbin165@163.com,phone=18867102689}  脱敏后原始： Customer:{username=\*五,idCard=342878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2689}  脱敏前原始： Customer:{username=赵六,idCard=343878909765155,password=43215678,email=zengbin166@163.com,phone=18867102690}  脱敏后原始： Customer:{username=\*六,idCard=343878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2690}  脱敏前原始： Customer:{username=钱七,idCard=344878909765156,password=12783456,email=zengbin167@163.com,phone=18867102691}  脱敏后原始： Customer:{username=\*七,idCard=344878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2691} |

## 自定义脱敏策略生效的场景（比如密码为123456不进行脱敏）

条件定义

|  |
| --- |
| package com.tzbank.desensitize;  import java.lang.reflect.Field;  import com.github.houbb.sensitive.annotation.Sensitive;  import com.github.houbb.sensitive.api.ICondition;  import com.github.houbb.sensitive.api.IContext;  import com.github.houbb.sensitive.core.api.strategory.StrategyPassword;  /\*\*  \* @desc 自定义脱敏策略生效的场景（比如密码为123456不进行脱敏）  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:21:26 PM  \*/  public class CustomerPasswordCondition implements ICondition {  @Override  public boolean valid(IContext context) {  try {  Field field = context.getCurrentField();  final Object currentObj = context.getCurrentObject();  final String password = (String) field.get(currentObj);  return !password.equals("123456");  } catch (IllegalAccessException e) {  throw new RuntimeException(e);  }  }  }  class Cust {  @Sensitive(condition = CustomerPasswordCondition.class, strategy = StrategyPassword.class)  private String password;  public String getPassword() {  return password;  }  public void setPassword(String password) {  this.password = password;  }  @Override  public String toString() {  return "password:" + password;  }  } |

单元测试

|  |
| --- |
| **package** com.tzbank.desensitize;  **import** com.github.houbb.sensitive.core.api.SensitiveUtil;  /\*\*   \*  \* **@desc**    自定义脱敏策略生效的场景（比如密码为123456不进行脱敏）  \* **@author**  台州银行:曾斌  \* **@version** 创建时间：Jun 26, 2019 2:22:17 PM   \*/  **public** **class** CustomerPasswordConditionTest {  **public** **static** **void** main(String[] args) {          Cust cust1 = **new** Cust();          cust1.setPassword("123456");          Cust cust2 = **new** Cust();          cust2.setPassword("1234567");          Cust sensitiveCust1 = SensitiveUtil.*desCopy*(cust1);          Cust sensitiveCust2 = SensitiveUtil.*desCopy*(cust2);          System.***out***.println(sensitiveCust1);          System.***out***.println(sensitiveCust2);      }  } |

测试结果

|  |
| --- |
| password:123456  password:null |

## 针对单个字段脱敏

|  |
| --- |
| package com.tzbank.desensitize;  import com.github.houbb.sensitive.api.IStrategy;  import com.github.houbb.sensitive.core.api.strategory.StrategyEmail;  /\*\*  \*  \* @desc 针对单个字段  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:22:41 PM  \*/  public class SingleSensitiveTest {  public static void main(String[] args) {  final String email = "123456@qq.com";  IStrategy strategy = new StrategyEmail();  final String emailSensitive = (String) strategy.des(email, null);  System.out.println("脱敏后的邮箱：" + emailSensitive);  }  } |

## 属性为集合或者对象

定义对象

|  |
| --- |
| package com.tzbank.desensitize;  import java.io.Serializable;  import java.util.List;  import org.apache.commons.collections4.CollectionUtils;  import org.apache.commons.lang3.ArrayUtils;  import com.github.houbb.sensitive.annotation.Sensitive;  import com.github.houbb.sensitive.annotation.SensitiveEntry;  import com.github.houbb.sensitive.core.api.strategory.StrategyChineseName;  /\*\*  \* @desc 属性为集合或者对象进行脱敏  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:21:09 PM  \*/  public class CustomerEntry implements Serializable {  private static final long serialVersionUID = -3701283493741419813L;  @SensitiveEntry  @Sensitive(strategy = StrategyChineseName.class)  private List<String> chineseNameList;  @SensitiveEntry  @Sensitive(strategy = StrategyChineseName.class)  private String[] chineseNameArray;  @SensitiveEntry  private Customer customer;  @SensitiveEntry  private List<Customer> customerList;  public List<String> getChineseNameList() {  return chineseNameList;  }  public void setChineseNameList(List<String> chineseNameList) {  this.chineseNameList = chineseNameList;  }  public String[] getChineseNameArray() {  return chineseNameArray;  }  public void setChineseNameArray(String[] chineseNameArray) {  this.chineseNameArray = chineseNameArray;  }  public Customer getCustomer() {  return customer;  }  public void setCustomer(Customer customer) {  this.customer = customer;  }  public List<Customer> getCustomerList() {  return customerList;  }  public void setCustomerList(List<Customer> customerList) {  this.customerList = customerList;  }  @Override  public String toString() {  String str = null;  if (!CollectionUtils.isEmpty(chineseNameList)) {  for (String s : chineseNameList) {  str = str + "," + s;  }  }  if (ArrayUtils.isNotEmpty(chineseNameArray)) {  for (String s : chineseNameArray) {  str = str + "," + s;  }  }  str = str + ",Customer:{username=" + customer.getUsername() + ",idCard=" + customer.getIdCard() + ",password="  + customer.getPassword() + ",email=" + customer.getEmail() + ",phone=" + customer.getPhone() + "}";  for (Customer cust : customerList) {  Customer customer = cust;  str = str + "-List-" + ",Customer:{username=" + customer.getUsername() + ",idCard=" + customer.getIdCard()  + ",password=" + customer.getPassword() + ",email=" + customer.getEmail() + ",phone="  + customer.getPhone() + "}";  }  return str;  }  } |

单元测试

|  |
| --- |
| package com.tzbank.desensitize;  import java.util.ArrayList;  import java.util.Arrays;  import java.util.List;  import com.github.houbb.sensitive.core.api.SensitiveUtil;  /\*\*  \*  \* @desc 属性为集合或者对象进行脱敏测试  \* @author 台州银行:曾斌  \* @version 创建时间：Jun 26, 2019 2:21:58 PM  \*/  public class CustomerEntryTest {  public static void main(String[] args) {  CustomerEntry custEntry = new CustomerEntry();  List<String> chineseNameList = Arrays.asList("盘古", "女娲", "伏羲");  String[] chineseNameArray = new String[] { "盘古", "女娲", "伏羲" };  Customer cust1 = new Customer("张三", "349878909765152", "12345678", "zengbin163@163.com", "18867102687");  Customer cust2 = new Customer("李四", "341878909765153", "87654321", "zengbin164@163.com", "18867102688");  List<Customer> custList = new ArrayList<>();  custList.add(cust2);  custEntry.setChineseNameList(chineseNameList);  custEntry.setChineseNameArray(chineseNameArray);  custEntry.setCustomer(cust1);  custEntry.setCustomerList(custList);  System.out.println("脱敏前原始： " + custEntry);  CustomerEntry sensitiveCustomerEntry = SensitiveUtil.desCopy(custEntry);  System.out.println("脱敏后原始： " + sensitiveCustomerEntry);  }  } |

测试结果

|  |
| --- |
| 脱敏前原始： null,盘古,女娲,伏羲,盘古,女娲,伏羲,Customer:{username=张三,idCard=349878909765152,password=12345678,email=zengbin163@163.com,phone=18867102687}-List-,Customer:{username=李四,idCard=341878909765153,password=87654321,email=zengbin164@163.com,phone=18867102688}  脱敏后原始： null,\*古,\*娲,\*羲,\*古,\*娲,\*羲,Customer:{username=\*三,idCard=349878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2687}-List-,Customer:{username=\*四,idCard=341878\*\*\*\*\*\*\*\*\*\*,password=null,email=zen\*\*\*\*\*\*\*@163.com,phone=188\*\*\*\*2688} |