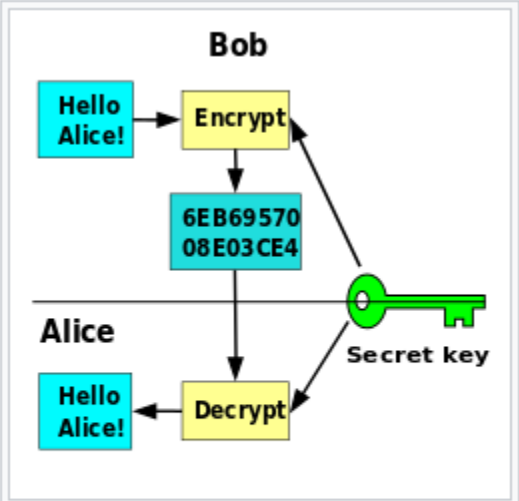
Jasypt Tutorial

- *By Swareddy*

- Jasypt - Java Simplified Encryption
- Jasypt is a java library which allows the developer to add basic encryption capabilities to his/her projects with minimum effort, and without the need of having deep knowledge on how cryptography works
- Having APIs with High security standard Encryption Algorithms (AES, SHA, MD5)
- Provide Supportfor Encrypt/decrypt operations for passwords, text, Numbers and decimals etc.
- Provide supports for Java frameworks like Hibernate, Spring Security etc...
- Integrated capabilities for encrypting the configuration related files (databases)
- Easy to integrate with build tools like maven, gradle etc.
- High performance encryption in multi core/multiprocessing processing systems
- Provides security for your application by encrypting sensitive data
- It is light weight library and Utility class for Encryption are completely thread safe
- Provides CLI tools for cryptographic operations
- Integrates into Apache Wicket, for more robust encryption of URLs

How Basic Symmetric-key cryptography Works base on secret key?



Symmetric-key cryptography, where a single key is used for encryption and decryption

Source: Wiki

Important Encryption Jasypt APIs for password encryption

RFC2307MD5PasswordEncryptor: This class usesMD5 password encryption scheme defined in RFC2307 and commonly found in LDAP systems
Properties:
Algorithm: MD5.
Salt size: 0 bytes (no salt).
Iterations: 1 (no hash iteration).
Prefix: {MD5}

RFC2307SHAPasswordEncryptor: This class uses SHA-1 password encryption scheme defined in RFC2307 and commonly found in LDAP systems
Properties:
Algorithm: SHA-1.
Salt size: 0 bytes (no salt).
Iterations: 1 (no hash iteration).
Prefix: {SHA}.

RFC2307SMD5PasswordEncryptor: This class uses SMD5 password encryption scheme defined in RFC2307 and commonly found in LDAP systems
Properties:
Algorithm: MD5.
Salt size: 8 bytes (configurable with setSaltSizeBytes(int)).
Iterations: 1 (no hash iteration).
Prefix: {SMD5}
Invert position of salt in message before digesting: true.
Invert position of plain salt in encryption results: true.
Use lenient salt size check: true.

RFC2307SSHAPasswordEncryptor: This class usesSMD5 password encryption scheme defined in RFC2307 and commonly found in LDAP systems. **Properties:**
Algorithm: SHA-1.
Salt size: 8 bytes (configurable with setSaltSizeBytes(int)).
Iterations: 1 (no hash iteration).
Prefix: {SSHA}.
Invert position of salt in message before digesting: true.
Invert position of plain salt in encryption results: true.
Use lenient salt size check: true.

StrongPasswordEncryptor: This class is used foreasily performing high-strength password digesting and checking.**Properties:**
Algorithm: SHA-256.
Salt size: 16 bytes.
Iterations: 100000.

Jasypt APIs for Plaintext

AES256TextEncryptor:
This class is used toperform high-strength encryption of texts.
Algorithm: PBEWithHMACSHA512AndAES_256".
Key obtention iterations: 1000.
BasicTextEncryptor: This class is used to perform normal-strength encryption of texts.
Algorithm: PBEWithMD5AndDES.
Key obtention iterations: 1000.
StrongTextEncryptor: This class is used to perform high-strength encryption of texts.
Algorithm: PBEWithMD5AndTripleDES.
Key obtention iterations: 1000.

Jasypt APIs for Integers

AES256IntegerNumberEncryptor:
This class is used to perform high-strength encryption of BigInteger objects.
Algorithm: PBEWithHMACSHA512AndAES_256".
Key obtention iterations: 1000.
BasicIntegerNumberEncryptor: This class is used toperform normal-strength encryption of BigInteger objects.
Algorithm: PBEWithMD5AndDES.
Key obtention iterations: 1000.
StrongIntegerNumberEncryptor: This class is used toperform high-strength encryption of BigInteger objects.
Algorithm: PBEWithMD5AndTripleDES.
Key obtention iterations: 1000.

Jasypt APIs for Decimal

AES256DecimalNumberEncryptor:
This class is used to perform high-strength encryption of BigDecimal objects.
Algorithm: PBEWithHMACSHA512AndAES_256".
Key obtention iterations: 1000.
BasicDecimalNumberEncryptor: This class is used to perform normal-strength encryption of BigDecimal objects.
Algorithm: PBEWithMD5AndDES.
Key obtention iterations: 1000.
StrongDecimalNumberEncryptor: This class is used to perform high-strength encryption of BigDecimal objects.
Algorithm: PBEWithMD5AndTripleDES.
Key obtention iterations: 1000.

Jasypt API for Binary Encryption/Decryption

AES256BinaryEncryptor:
This class is used toperform high-strength encryption of binaries (byte arrays).
Algorithm: PBEWithHMACSHA512AndAES_256".
Key obtention iterations: 1000.
BasicBinaryEncryptor: This class is used to perform normal-strength encryption of binaries (byte arrays).
Algorithm: PBEWithMD5AndDES.
Key obtention iterations: 1000.
StrongDecimalNumberEncryptor: This class is used toperform high-strength encryption of binaries (byte arrays).
Algorithm: PBEWithMD5AndTripleDES.
Key obtention iterations: 1000.