

# Jasypt

## Java Simplified Encryption



*Laboratorio di Reti*  
*2014/2015*  
*Prof. Laura Ricci*

*Speaker: Alessandro Lulli - [lulli@di.unipi.it](mailto:lulli@di.unipi.it)*

# Jasypt

- una libreria per fare encrypting / decrypting in Java

The screenshot shows a web browser window displaying the Jasypt website. The browser's address bar shows the URL `www.jasypt.org/index.html`. The website header features the logo "jasypt." with the tagline "JAVA SIMPLIFIED ENCRYPTION" below it. A navigation bar includes links for "Main", "Download", "Features", and "Sourceforge.net Project Page".

The main content area is titled "Jasypt 1.9.2 RELEASED! (February 25th, 2014) [DOWNLOAD and ChangeLogs] [WHAT'S NEW IN JASYPT 1.9]". Below this, a large grey box contains the heading "Java Simplified Encryption".

The text below the heading reads: "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."

A bulleted list of features follows:

- High-security, standards-based encryption techniques, both for unidirectional and bidirectional encryption. Encrypt passwords, texts, numbers, binaries...
- Transparent integration with **Hibernate**.
- Suitable for integration into **Spring**-based applications and also transparently integrable with **Spring Security**.
- Integrated capabilities for encrypting the configuration of applications (i.e. datasources)
- Specific features for **high-performance encryption** in multi-processor/multi-core systems.
- Open API for use with any JCE provider.
- ...and much more

Below the list, it says: "Have a look at the complete set of **Jasypt Features** or check the **FAQ**."

The text continues: "With Jasypt, encrypting and checking a password can be as simple as..."

```
StrongPasswordEncryptor passwordEncryptor = new StrongPasswordEncryptor();
String encryptedPassword = passwordEncryptor.encryptPassword(userPassword);
...
if (passwordEncryptor.checkPassword(inputPassword, encryptedPassword)) {
    // correct!
} else {
    // bad login!
}
```

...encrypting and decrypting a text...

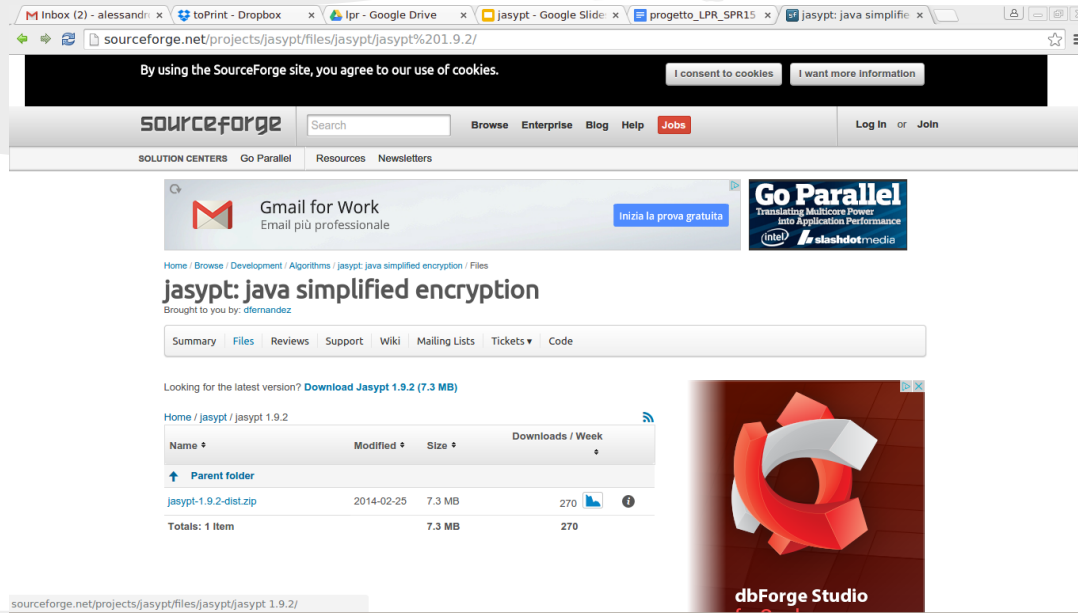
```
StrongTextEncryptor textEncryptor = new StrongTextEncryptor();
textEncryptor.setSecure(myEncryptionResource);
```

The left sidebar contains a navigation menu with sections: "The Jasypt Project" (Main, Features, Download, Source Repository, Issue Tracking, License, Team), "Reference" (JavaDoc API, Dependencies, FAQ), "Guides" (Using Jasypt, Easy Usage, General Usage, Using the 'lite' artifact, Encrypting from the command line, Advanced Usage, Advanced cryptor/digester configuration, Web PBE configuration, Using non-default JCE providers), "By Data Type" (Encrypting passwords, Encrypting texts, Encrypting numbers, Encrypting binaries, Encrypting application configuration files), and "Building" (Jasypt + Apache).

At the bottom of the browser window, the taskbar shows several Java JAR files: `json-simple-1.1.1.jar`, `json-lib-2.4-jdk15.jar`, and a "Show all downloads..." button.

# Download

- Jasypt può essere scaricato dal seguente link:
  - <http://www.jasypt.org/index.html>
- i test sono stati fatti con jasypt 1.9.2







sourceforge.net/projects/jasypt/files/jasypt/jasypt%201.9.2/

By using the SourceForge site, you agree to our use of cookies. [I consent to cookies](#) [I want more information](#)

Sourceforge  [Browse](#) [Enterprise](#) [Blog](#) [Help](#) [Jobs](#) [Log In](#) or [Join](#)

SOLUTION CENTERS [Go Parallel](#) [Resources](#) [Newsletters](#)

 Gmail for Work  
Email più professionale [Inizia la prova gratuita](#)

 Go Parallel!  
Translating Multicore Power  
Into Application Performance  
 

[Home](#) / [Browse](#) / [Development](#) / [Algorithms](#) / [jasypt: java simplified encryption](#) / [Files](#)


## jasypt: java simplified encryption


Brought to you by: [dfernandez](#)

[Summary](#) [Files](#) [Reviews](#) [Support](#) [Wiki](#) [Mailing Lists](#) [Tickets](#) [Code](#)

Looking for the latest version? [Download Jasypt 1.9.2 \(7.3 MB\)](#)

[Home](#) / [jasypt](#) / [jasypt 1.9.2](#)

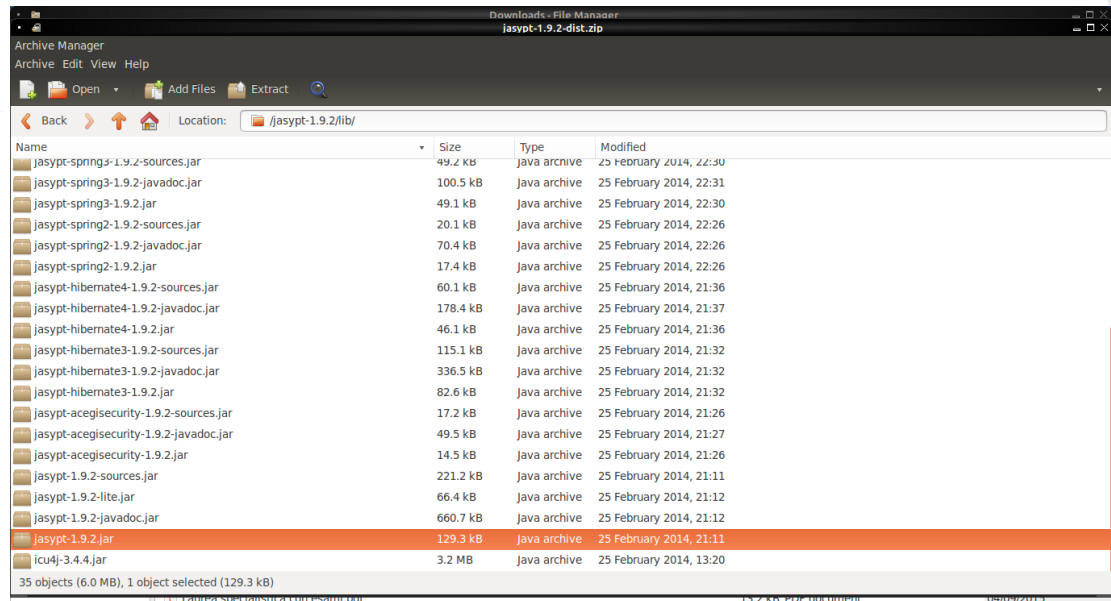
Name	Modified	Size	Downloads / Week
<a href="#">↑ Parent folder</a>			
<a href="#">jasypt-1.9.2-dist.zip</a>	2014-02-25	7.3 MB	270  
<b>Totals: 1 Item</b>		<b>7.3 MB</b>	<b>270</b>

 dbForge Studio

sourceforge.net/projects/jasypt/files/jasypt/jasypt 1.9.2/

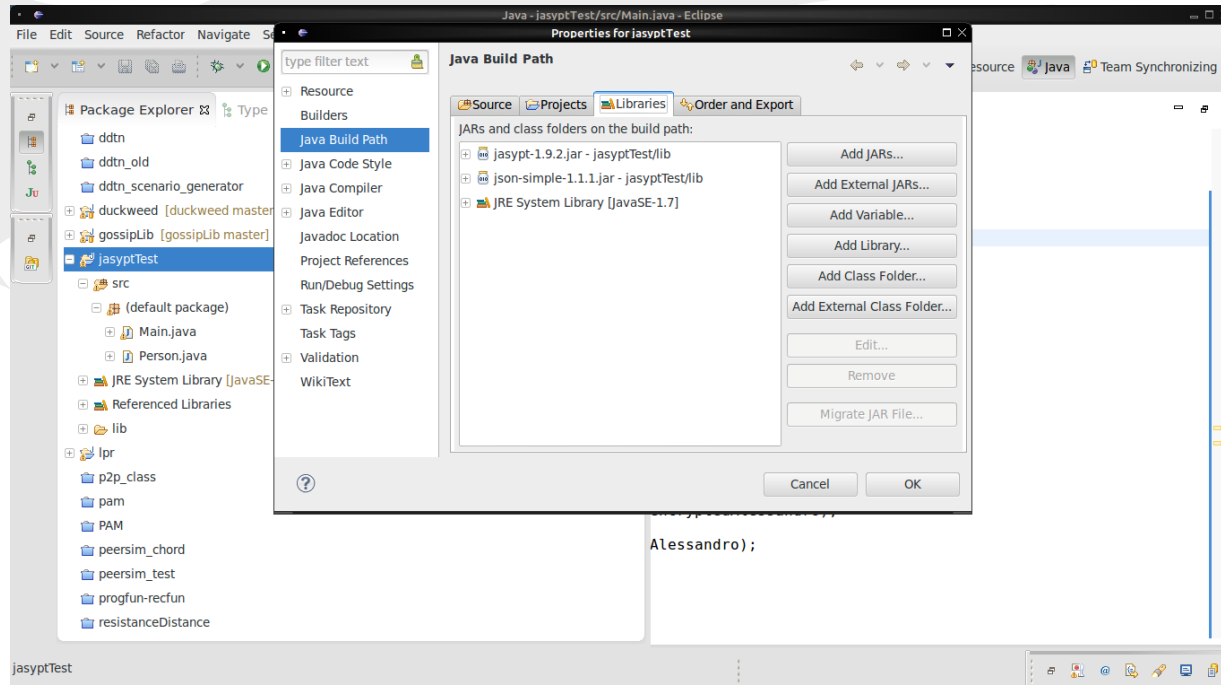
# Install in Eclipse (I)

- decomprimere lo script ed estrarre la libreria *jaspyt-1.9.2.jar* nella folder del progetto Eclipse
- il file è nella seguente cartella:
  - `/jaspyt-1.9.2/lib/`



# Install in Eclipse (II)

- aggiungere la libreria jasypt-1.9.2.jar



# Encrypt / Decrypt di testo (I)

- si crea una password di tipo String
- la password dovrà essere condivisa tra chi esegue la encrypt e chi esegue il decrypt

```
import org.jasypt.util.text.BasicTextEncryptor;

public class Main
{
    public static void main(final String[] args)
    {
        final String password = "strongPassword";

        final BasicTextEncryptor encryptor = new BasicTextEncryptor();
        encryptor.setPassword(password);

        final String encryptedString = encryptor.encrypt("Parola segreta");

        System.out.println("Encrypted String: "+encryptedString);

        final String decryptedString = encryptor.decrypt(encryptedString);

        System.out.println("Decrypted String: "+decryptedString);
    }
}
```

# Encrypt / Decrypt di testo (II)

- si crea una classe di tipo *BasicTextEncryptor*
  - sarà il nostro encryptor / decryptor di stringhe
- si setta la password con il quale fare encrypt / decrypt

```
import org.jasypt.util.text.BasicTextEncryptor;

public class Main
{
    public static void main(final String[] args)
    {
        final String password = "strongPassword";

        final BasicTextEncryptor encryptor = new BasicTextEncryptor();
        encryptor.setPassword(password);

        final String encryptedString = encryptor.encrypt("Parola segreta");

        System.out.println("Encrypted String: "+encryptedString);

        final String decryptedString = encryptor.decrypt(encryptedString);

        System.out.println("Decrypted String: "+decryptedString);
    }
}
```

# Encrypt / Decrypt di testo (III)

- dato il nostro oggetto *encryptor*
- per encrypt: *encryptor.encrypt("Parola segreta");*
- per decrypt: *encryptor.decrypt(encryptedString);*

```
import org.jasypt.util.text.BasicTextEncryptor;

public class Main
{
    public static void main(final String[] args)
    {
        final String password = "strongPassword";

        final BasicTextEncryptor encryptor = new BasicTextEncryptor();
        encryptor.setPassword(password);

        final String encryptedString = encryptor.encrypt("Parola segreta");

        System.out.println("Encrypted String: "+encryptedString);

        final String decryptedString = encryptor.decrypt(encryptedString);

        System.out.println("Decrypted String: "+decryptedString);
    }
}
```



# Encrypt / Decrypt JSON

- si crea la string JSON e si passa alla encryptor

```
final String password = "strongPassword";

final BasicTextEncryptor encryptor = new BasicTextEncryptor();
encryptor.setPassword(password);

final Person person = new Person("Alessandro", "Lulli");

final JSONObject personJson = new JSONObject();
personJson.put("nome", person.getName());
personJson.put("cognome", person.getSurname());

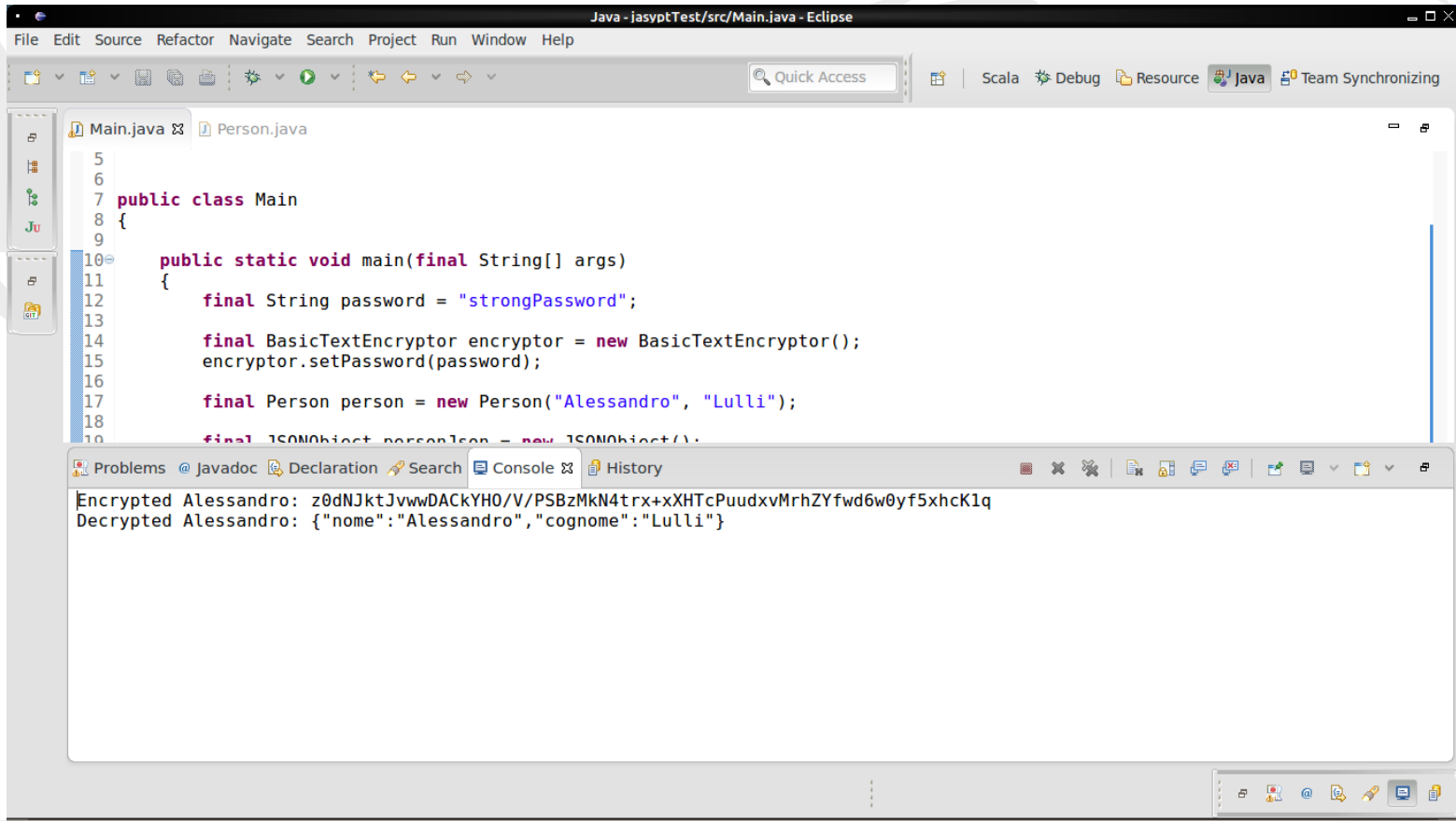
final String encryptedAlessandro = encryptor.encrypt(personJson.toJSONString());

System.out.println("Encrypted Alessandro: "+encryptedAlessandro);

final String decryptedAlessandro = encryptor.decrypt(encryptedAlessandro);

System.out.println("Decrypted Alessandro: "+decryptedAlessandro);
```

# Encrypt / Decrypt JSON Output



The screenshot shows the Eclipse IDE interface. The main editor displays the following Java code in `Main.java`:

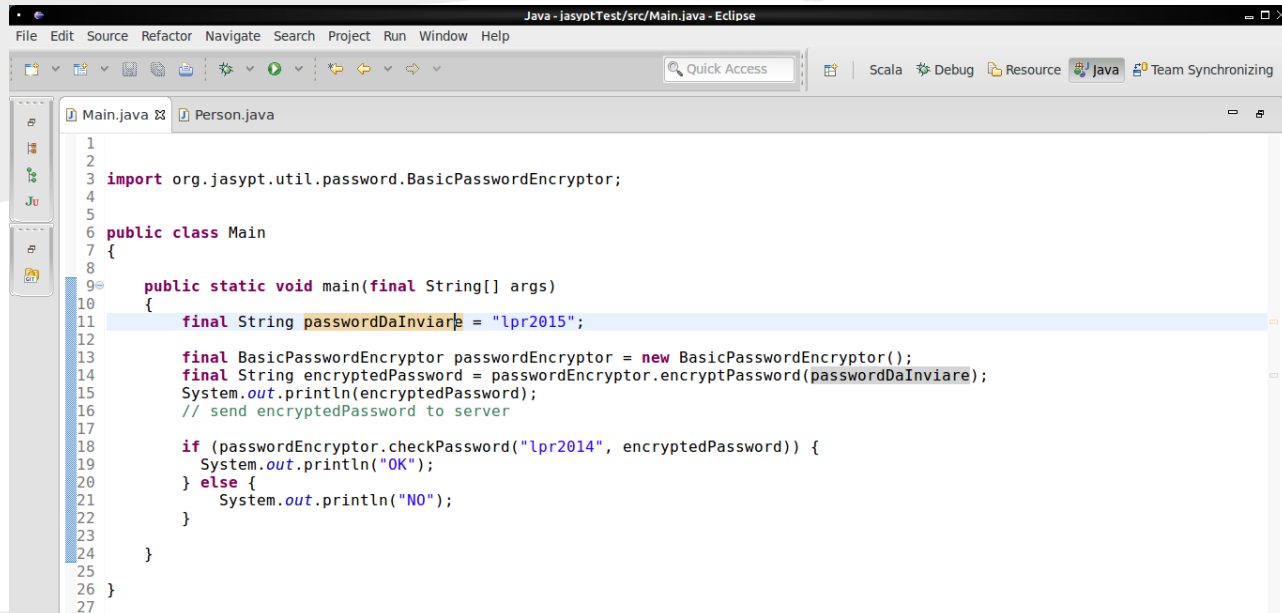
```
5
6
7 public class Main
8 {
9
10     public static void main(final String[] args)
11     {
12         final String password = "strongPassword";
13
14         final BasicTextEncryptor encryptor = new BasicTextEncryptor();
15         encryptor.setPassword(password);
16
17         final Person person = new Person("Alessandro", "Lulli");
18
19         final JSONObject personJson = new JSONObject();
```

The bottom console window shows the output of the program:

```
Encrypted Alessandro: z0dNJktJvwwDACKYH0/V/PSBzMkN4trx+xHTcPuudxvMrhZYfwd6w0yf5xhcK1q
Decrypted Alessandro: {"nome":"Alessandro","cognome":"Lulli"}
```

# Encrypt / Decrypt Password

- si usa la classe *BasicPasswordEncryptor*
- una volta crittata la password:
  - si invia al server che farà il check con *checkPassword*



```
Java - jasyptTest/src/Main.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access Scala Debug Resource Java Team Synchronizing
Main.java Person.java
1
2
3 import org.jasypt.util.password.BasicPasswordEncryptor;
4
5
6 public class Main
7 {
8
9     public static void main(final String[] args)
10    {
11        final String passwordDaInviare = "lpr2015";
12
13        final BasicPasswordEncryptor passwordEncryptor = new BasicPasswordEncryptor();
14        final String encryptedPassword = passwordEncryptor.encryptPassword(passwordDaInviare);
15        System.out.println(encryptedPassword);
16        // send encryptedPassword to server
17
18        if (passwordEncryptor.checkPassword("lpr2014", encryptedPassword)) {
19            System.out.println("OK");
20        } else {
21            System.out.println("NO");
22        }
23    }
24 }
25
26 }
27
```