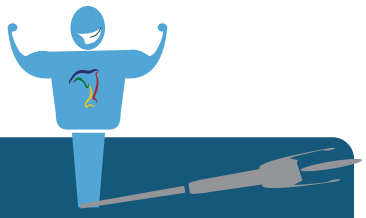
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

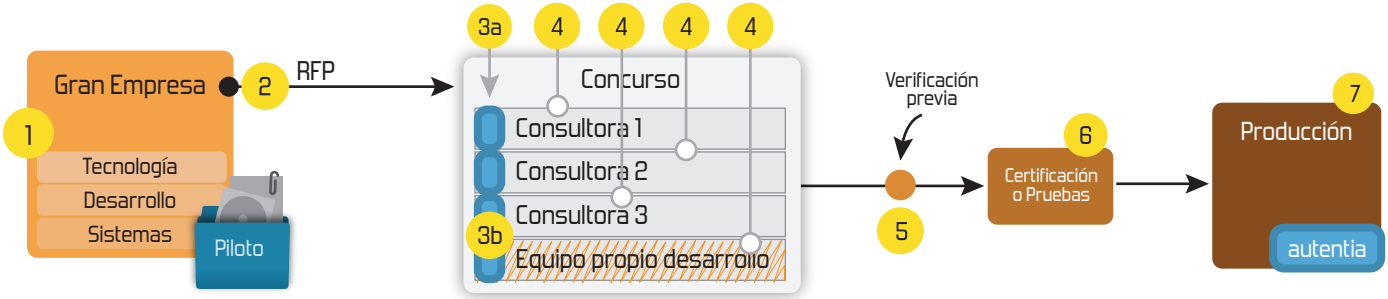
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Control de autenticación y acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

DESARROLLADO POR:
Alejandro Pérez García

Alejandro es socio fundador de Autentia y nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.

Ingeniero en Informática y Certified ScrumMaster

Si te gusta lo que ves, puedes contratarle para darte ayuda con soporte experto, impartir **cursos presenciales** en tu empresa o para que **realicemos tus proyectos como factoría** (Madrid). Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación

Fecha de publicación del tutorial: 2011-02-09

Jackson y como deserializar objetos JSON usando un constructor

Creación: 09-02-2011

Índice de contenidos

1. Introducción

Este es un minitutorial donde vamos a ver como deserializar un objeto JSON con la librería Jackson y haciendo uso de un constructor de nuestra clase Java.

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 17' (2.8 GHz Intel i7, 8GB DDR3 SDRAM, 256GB Solid State Drive).

- NVIDIA GeForce GT 330M with 512MB
- Sistema Operativo: Mac OS X Snow Leopard 10.6.6
- JDK 1.6.0_22
- Jackson 1.6.4

3. Jackson y como deserializar objetos JSON usando un constructor de nuestra clase Java

En primer lugar vamos a entrar un poquito en contexto y vamos a contas que son esos palabros del título:

- **JSON** (http://www.json.org/js.html) es una representación de objetos en forma de cadena. Es muy usada en JavaScript por su comodidad, pero también se puede usar fuera de esté ámbito. Un ejemplo de una representación de Persona:

  ```
  { “name” : “Alejandro”, “lastName” : “Pérez García”, “company” : “Autentia Real Business Solutions” }
  ```

  Se puede ver que es un formato muy sencillo donde los objetos van entre llaves { }, y los atributos se representan por parejas de clave/valor.

- **Deserializar** es el proceso mediante el cual vamos a convertir esta representación de texto en un objeto real. En este caso en un objeto de Java.
- **Jackson** (http://jackson.codehaus.org/) es una librería de utilidad de Java que nos simplifica el trabajo de serializar (convertir un objeto Java en una cadena de texto con su representación JSON), y deserializar (convertir una cadena de texto con una representación de JSON de un objeto en un objeto real de Java) objetos JSON.

Jackson es bastante “inteligente” y sin decirle nada es capaz de serializar y deserializar bastante bien los objetos. Para ello usa básicamente la introspección de manera que si en el objeto JSON tenemos un atributo “name”, para la serializción buscará un método “getName()” y para la deserialización buscará un método “setName(String s)”. Así el código para deserilzar un objeto JSON sería tan sencillo como:

```
Person person = new ObjectMapper().readValue(jsonText, Person.class);
```

Pero ¿qué pasa si no quiero tener setters en mi implementación. Es decir, si yo quiero que mis objetos Person sean inmutables y no se puedan cambiar, no debería meter un setter porque Jackson lo necesite!!! Recordar siempre que el modelo es lo realmente importante, no deberíamos dejar que una librería de utilidad influencie nuestro diseño, y menos en algo tan importante como convertir un objeto inmutable en mutable (usar todos lo objetos inmutables que podáis y veréis como se reducen vuestros errores ;)

Entonces ¿qué opciones tenemos? Bueno, a mi hay una que me gusta bastante y que es la que vamos a ver aquí, es simplemente tener un constructor para construir correctamente nuestros objetos. Esta opción me gusta más que, por ejemplo, inyectar los valores directamente en los atributos, porque el tener un constructor nos permite hacer pruebas de manera más sencilla y con menos “artificios”. Este sistema también me gusta para la inyección de dependencias con Spring, hacer la inyección a través del contructor (no con los setters ni con los atributos directamente), y hacer guardar los colaboradores que inyectáis en atributos `final`, para dejar claro que son constantes y que no se van a modificar durante la vida del objeto.

Tenemos la suerte de que Jackson nos permite usar anotaciones para indicarle que método es el que tiene que usar para hacer la deserialización. Sería algo tan sencillo como poner dentro de la clase Person, un constructor del estilo:

```
private final String name;
private final String lastName;
private final String company;

@JsonCreator
public Person(@JsonProperty("name") String name, @JsonProperty("lastName") String lastName,
        @JsonProperty("company") String company) {
    this.name = name;
    this.lastName = lastName;
    this.company = company;
```

Spring y JPA

Como editar XML o HTML con el plugin xmledit de Vim

Últimos Tutoriales del Autor

Como editar XML o HTML con el plugin xmledit de Vim

Cosas que no funcionan en JSF2 como uno podría esperar (es decir, bugs)

Spring @Configurable y los modelos de dominio anémicos

Madrid.rb y la kata de los Romanos

Reunión Madrid Ágil 02-11-2010: DDD (Domain Driven Design)

Síguenos a través de:

Últimas ofertas de empleo

2010-10-11
Comercial - Ventas - SEVILLA.

2010-08-30
Otras - Electricidad - BARCELONA.

2010-08-24
Otras Sin catalogar - LUGO.

2010-06-25
T. Información - Analista / Programador - BARCELONA.

```
}
```

Otra opción que también nos permite Jackson es usar un Factory Method, es decir, usar método estático en la clase Person que se encargue de construir el objeto. Esta opción también es muy atractiva, y simplemente sería un método del estilo:

```
public static buildFromJson(String jsonText) {
    Person newPerson = new Person();
    ObjectMapper mapper = new ObjectMapper();
    ...
    return newPerson;
}
```

Está solución requiere un poco más de código porque usamos directamente el ObjectMapper para hacer la deserialización, pero en ocasiones puede resultar muy interesante porque nos da más flexibilidad a la hora de construir el objeto. Os dejo como deberes el ver como configuramos Jackson para usar este método estático ;)

4. Conclusiones

Leer la documentación y aprender a usar las librerías que tenéis a vuestro alcance. A veces la documentación no es nada clara y conviene buscar por Internet y sobre todo hacer unos cuantos test para ver como se comporta la librería. Como dice el libro "The Pragmatic Programmer", no programéis por coincidencia, tenéis que trabajar sabiendo realmente el comportamiento de las cosas que usáis, y para ello muchas veces lo mejor es probarlo para tener la certeza.

Y no dejéis que un librería o un framework condicione vuestro diseño !!!

5. Sobre el autor

Alejandro Pérez García, Ingeniero en Informática (especialidad de Ingeniería del Software) y Certified ScrumMaster

Socio fundador de Autentia (Desarrollo de software, Consultoría, Formación)

mailto:alejandropg@autentia.com

Autentia Real Business Solutions S.L. - "Soporte a Desarrollo"

http://www.autentia.com