

¿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

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

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

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

AdictosAlTrabajo

Terrakas 1x04
¡¡Ya está en la web!! ;-)
terrakas.com



autentia
Soporte a desarrollo informático
Hosting patrocinado por
enREDados

Entra en Adictos a través de  

E-mail

Contraseña

Entrar [Deseo registreme](#)
[Olvidé mi contraseña](#)

[Inicio](#) [Quiénes somos](#) [Formación](#) [Comparador de salarios](#) [Nuestro libro](#) [Más](#)

» Estás en: Inicio » Tutoriales » JRBeanCollectionDataSource: trabajando con arrays multidimensionales en Jas...



Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en **Autentia**: Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE



[Ver todos los tutoriales del autor](#)



Fecha de publicación del tutorial: 2012-10-23

Tutorial visitado 0 veces [Descargar en PDF](#)

JRBeanCollectionDataSource: trabajando con arrays multidimensionales en Jasper Report.

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Plantilla jrxml.
- 4. Generación del informe desde java.
- 5. Conclusiones.

1. Introducción

Después de exponer como trabajar con colecciones de tipos básicos en iReport sin una fuente de datos definida, en este tutorial vamos a dar una vuelta de tuerca añadiendo la complejidad de tener como parámetro en el informe un array multidimensional de tipos básicos, una nube de puntos.

Sin más, y tomando como referencia el tutorial anterior vamos a pintar la información anidada en listas dentro del informe.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 15' (2.4 GHz Intel Core i7, 8GB DDR3 SDRAM).
- Sistema Operativo: Mac OS X Lion 10.7.4
- iReport 4.7.1
- Jasper Report 4.7.1

3. Creación de la plantilla jrxml.

Con la plantilla vacía y sin asignar una fuente de datos "Empty Datasource":

Catálogo de servicios Autentia



Síguenos a través de:



Últimas Noticias

» Participamos en la Carrera de las Empresas 2012

» ¡¡¡Terrakas 1x04 recién salido del horno!!!

» Estreno Terrakas 1x04: "Terraka por un día"

» Nuevos cursos de gestión de la configuración en IOS y Android

» La regla del Boy Scout y la Oxidación del Software

[Histórico de noticias](#)

Últimos Tutoriales

» JRBeanCollectionDataSource: trabajando con colecciones de datos básicos en Jasper Report.

» Código de barras con iReport

» Uso de StoryBoards en desarrollo IOS

» Code Snippets en XCode 4

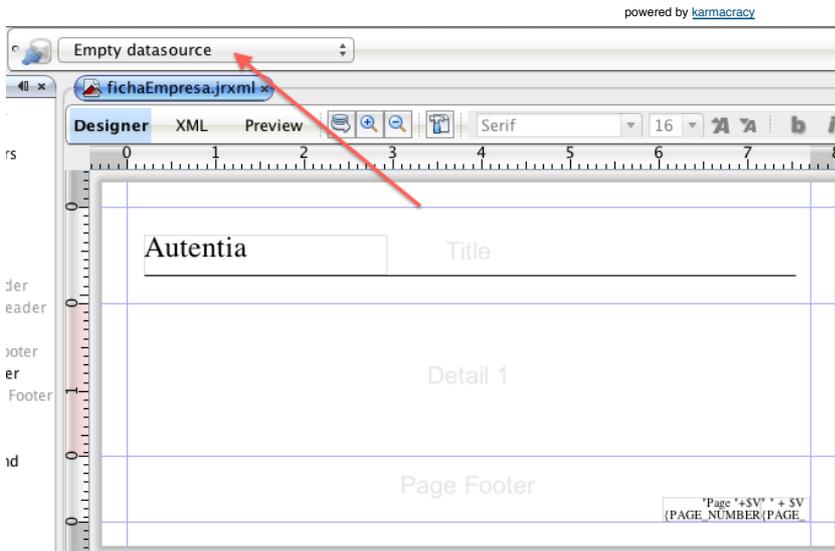
» Mountain Lion - Git

IMPULSA Impulsores Comunidad ¿Ayuda?

0 personas han traído clicks a esta página

sin clicks

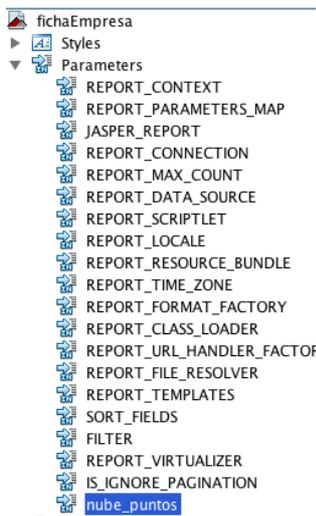
+ + + + + + + +



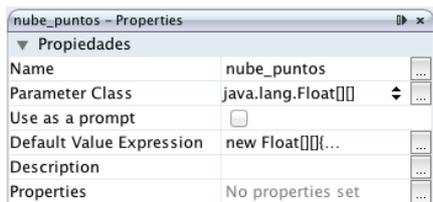
Creamos un parámetro, para ello "botón derecho" sobre "Parameters" > "Agregar Parameter":



Le asignamos un nombre, en nuestro caso "nube_puntos":



Y accediendo a la ventana de propiedades, le damos una tipología en "Parameter Class" > "java.util.Float[][]", un array multidimensional de floats,



además pulsando sobre "Default Value Expression" podemos asignar algo como lo siguiente para añadir un juego de pruebas:

"Command Not Found"

Últimos Tutoriales del Autor

- » JRBeanCollectionDataSource: trabajando con colecciones de datos básicos en Jasper Report.
- » Integración de Spring Web Flow 2 con JSF2
- » JSF2 Flash scope
- » Obtención de los literales de i18n de base de datos en JSF2.
- » Introducción a Spring Data: soporte para JPA.

Últimas ofertas de empleo

- 2011-09-08 [Comercial - Ventas - MADRID.](#)
- 2011-09-03 [Comercial - Ventas - VALENCIA.](#)
- 2011-08-19 [Comercial - Compras - ALICANTE.](#)
- 2011-07-12 [Otras Sin catalogar - MADRID.](#)
- 2011-07-06 [Otras Sin catalogar - LUGO.](#)

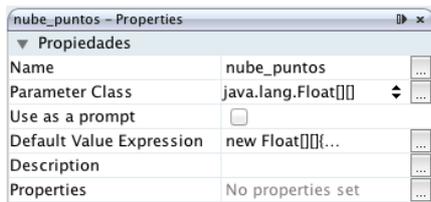
Jose Manuel Sánchez sanchezsuarezj

adictosaltrabaj
JRBeanCollectionDataSource: trabajando con colecciones de datos básicos en #jasperReport #iReport. - kcy.me/btib yesterday · reply · retweet · favorite

sanchezsuarzj @luchito_flores prueba con este tutorial kcy.me/btjw EL 2.2 Tomcat6 invocar a un método pasando parámetros #JSF yesterday · reply · retweet · favorite

adictosaltrabaj Uso de StoryBoards en el desarrollo con IOS - kcy.me/brf6, de nuevo de la mano de @rcanalesmora 4 days ago · reply · retweet · favorite

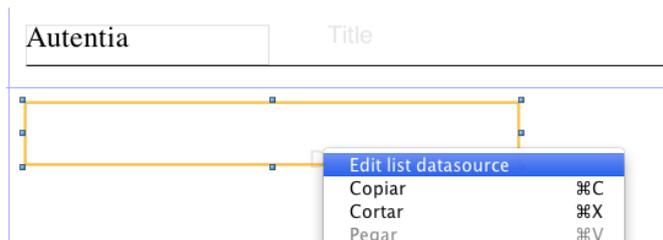
adictosaltrabaj Code Snippets en [Join the conversation](#)



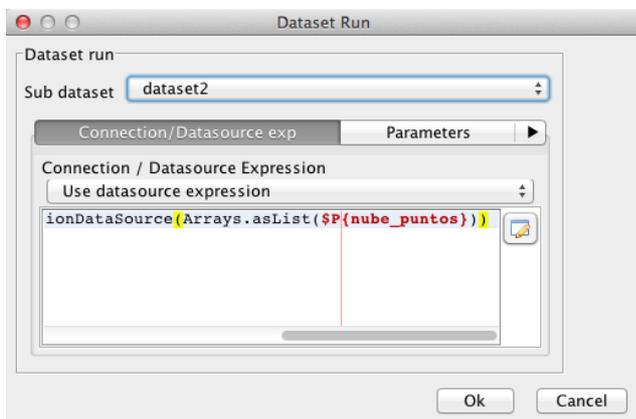
Con ello, ya tenemos preparado nuestro parámetro para el entorno de pruebas y ahora vamos a añadir un componente visual de tipo lista para iterar por su contenido y mostrar la nube de puntos.



Pulsando sobre icono anterior y arrastrándolo al área de la plantilla lo tendremos disponible para su edición:



Una vez incluido en la sección correspondiente de la plantilla "botón derecho" > "Edit datasource" mostrará una ventana como la que sigue:



En este punto son importantes dos cuestiones:

- "Sub dataset" definirá los parámetros de entrada, campos, variables,... que vivirán en el ámbito de la lista y que, no tienen por qué coincidir con los de la plantilla padre, ahí será donde definiremos un campo "ad hoc".
- Connection / Datasource Expression: indica la fuente de datos para la lista que puede ser la misma fuente de datos que el informe padre u otra, en nuestro caso definimos una expresión usando la clase `net.sf.jasperreports.engine.data.JRBeanCollectionDataSource` y pasando como argumento al constructor una referencia al parámetro anteriormente definido.

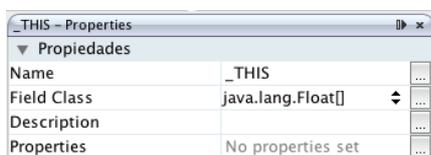


Lo siguiente será definir un campo que haga referencia a cada uno de los items de la colección, si tuviéramos un objeto tipado, añadiríamos los campos a mostrar de la clase o el bean en cuestión, como trabajamos con tipos básicos añadiríamos un campo con la palabra reservada `_THIS`, que hará referencia a cada una de las cadenas dentro de la iteración interna de la lista.

Sobre el dataset2, el que usa la lista, pulsamos "botón derecho" en "Fields" > "Agregar Field"



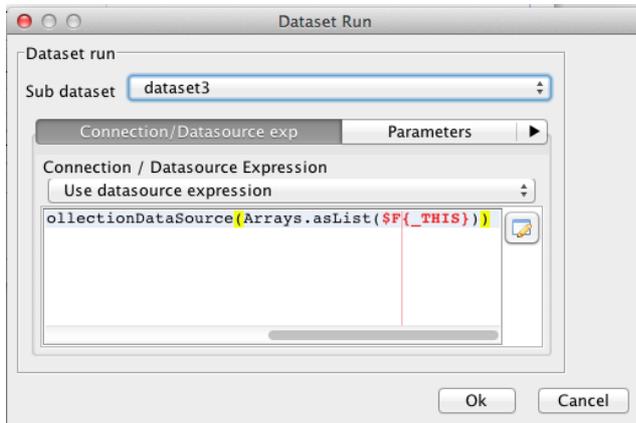
El nombre para nuestro campo es la palabra reservada `_THIS` y el tipo es un array de floats:



Con ello estamos accediendo a la primera dimensión, si queremos acceder a los puntos debemos incluir un nuevo componente de lista anidado



que estará asociada a nuevo dataset y tendrá la siguiente fuente de datos:



Este dataset tendrá un campo también con el nombre `_THIS`, pero del tipo `java.lang.Float`.



Con ello ya podemos incluir un campo `$_THIS` en la lista anidada para hacer referencia a cada punto a imprimir:



El informe tendría una salida como la siguiente:

```
puntos    1.2
           1.3

puntos    1.4
           1.5

puntos    1.6
           1.7

puntos    1.8
           1.9
           1.1
           1.11

puntos    2.8
           2.9
           2.1
           2.11
```

4. Generación del informe desde java.

Si el informe anterior lo tuviéramos que generar desde código bastaría hacer uso del servicio que vimos en el tutorial anterior de la siguiente forma:

```
1 private static Float[][] puntos = new Float[][]{
2     new Float[]{1.2f, 1.3f},
3     new Float[]{1.4f, 1.5f},
4     new Float[]{1.6f, 1.7f},
5     new Float[]{1.8f, 1.9f, 1.10f, 1.11f},
6     new Float[]{2.8f, 2.9f, 2.10f, 2.11f}
7 };
8
9 public static void main(String[] args) throws JRException {
10     final Map<String, Object> parameters = new HashMap<String, Object>();
11     parameters.put("nube_puntos", puntos);
```

```
12     final ReportExporter reportExporter = new ReportExporter();
13     reportExporter.toPDF("fichaEmpresa.jrxml", parameters);
14 }
```

El resultado será el mismo que el obtenido desde el entorno de iReport.

5. Conclusiones.

Que que comentábamos, lo ideal es trabajar con objetos tipados ;).

Un saludo.

Jose

jmsanchez@autentia.com

A continuación puedes evaluarlo:

[Regístrate para evaluarlo](#)



Por favor, vota +1 o compártelo si te pareció interesante



Ánimate y coméntanos lo que pienses sobre este **TUTORIAL**:

» [Regístrate](#) y accede a esta y otras ventajas «



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Copyright 2003-2012 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) | [Contacto](#)

