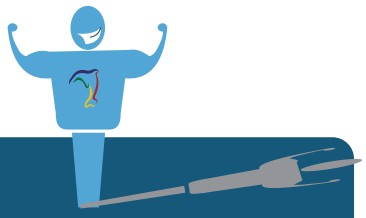
autentia

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...

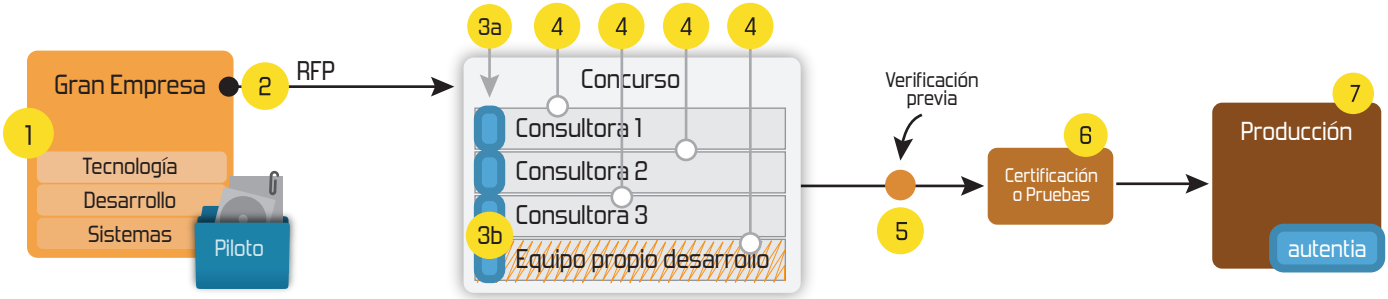
1. Desarrollo de componentes y proyectos a medida



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

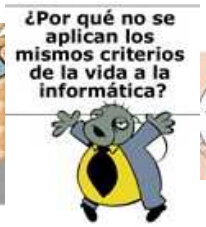
Para más información visítenos en:
www.autentia.com

NUEVO ¿Quieres saber cuánto ganas en relación al mercado? pincha aquí...

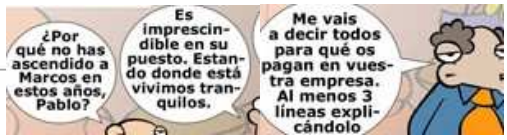
Ver cursos que ofrece Autentia

Descargar comics en PDF y alta resolución

[¡NUEVO!] 2008-09-01



2008-07-31



2008-07-08 2008-06-22

Estamos escribiendo un libro sobre la profesión informática y estas viñetas formarán parte de él. Puedes opinar en la seccion comic.

Tutorial desarrollado por

Roberto Canales Mora

Creador y propietario de AdictosAlTrabajo.com, Director General de Autentia S.L., Ingeniero Técnico de Telecomunicaciones y Executive MBA por el Instituto de Empresa 2007. Perfíl Technorati

Experto en formación en: Dirección de proyectos informáticos, análisis y diseño UML, arquitectura Web, patrones de diseño y JEE a todos los niveles.

Puedes consultar mi CV y alguna de mis primeras aplicaciones (de los 90) aquí

Catálogo de servicios de Autentia

Descargar (6,2 MB)

Descargar en versión comic (17 MB)

AdictosAlTrabajo.com es el Web de difusión de conocimiento de Autentia.

autentia real business solutions

Catálogo de cursos

Descargar este documento en formato PDF: jdk15.pdf

Fecha de creación del tutorial: 2004-02-10

Novedades en Java 1.5

Java está en continua evolución y nosotros desde www.adictosaltrabajo.com le hacemos un férreo marcaje....

Os vamos a contar en este tutorial algunas de las nuevas características de Java.

Para los que hemos trabajado con C++ estas novedades se parecen bastante a lo que estábamos acostumbrados a utilizar....

Se supone que durante mucho tiempo se había definido Java como un C++--++, es decir, le habían quitado algunos cosas y le habían puesto otras.... ahora casi hay que decir que es un C++-+++. Para mi gusto, cada día me parecen más iguales .....

Podéis consultar el enlace de SUN para ver un resumen de las nuevas características y el enlace al documento detallado.

http://java.sun.com/j2se/1.5.0/docs/relnotes/features.html

Nos vamos a descargar los binarios y la documentación del siguiente enlace...

http://java.sun.com/j2se/1.5.0/download.jsp

Y comenzamos la instalación (que no me ha dado ningún problema)

Catálogo de servicios Autentia (PDF 6,2MB)

En formato comic...

Google

Web

www.adictosaltrabajo.com

Buscar

Últimos tutoriales

2008-10-27
Web Services con Estado

2008-10-24
Web Services con Axis2. Configuración y ejemplo

2008-10-22
Migración de JSP a Facelets

2008-10-22
Rock Band Wii en tu PC

2008-10-18
Cobertura: Como comprobar cuanto código prueban nuestros test

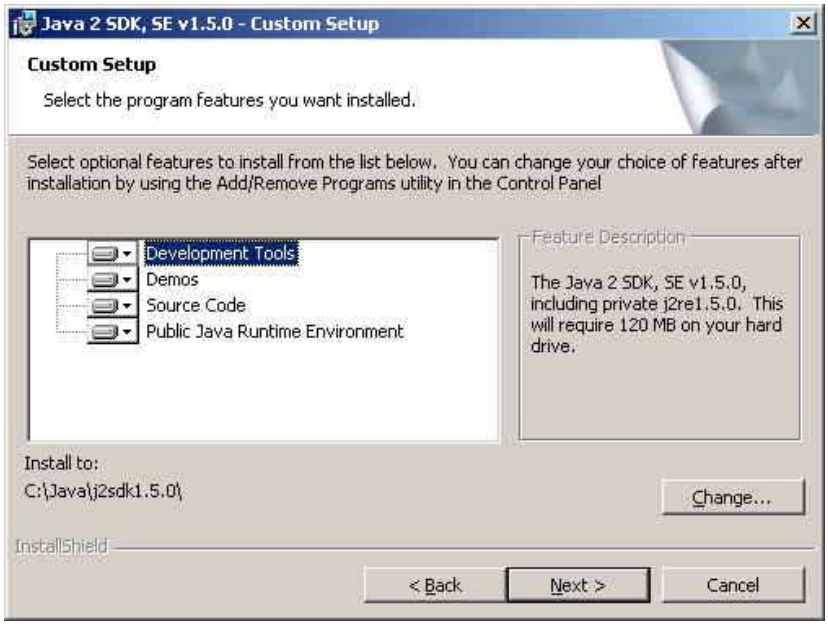
2008-10-17
maven-license-plugin: cómo gestionar la licencia de nuestros ficheros fuentes

2008-10-10
Cypal Studio: plugin de GWT para Eclipse

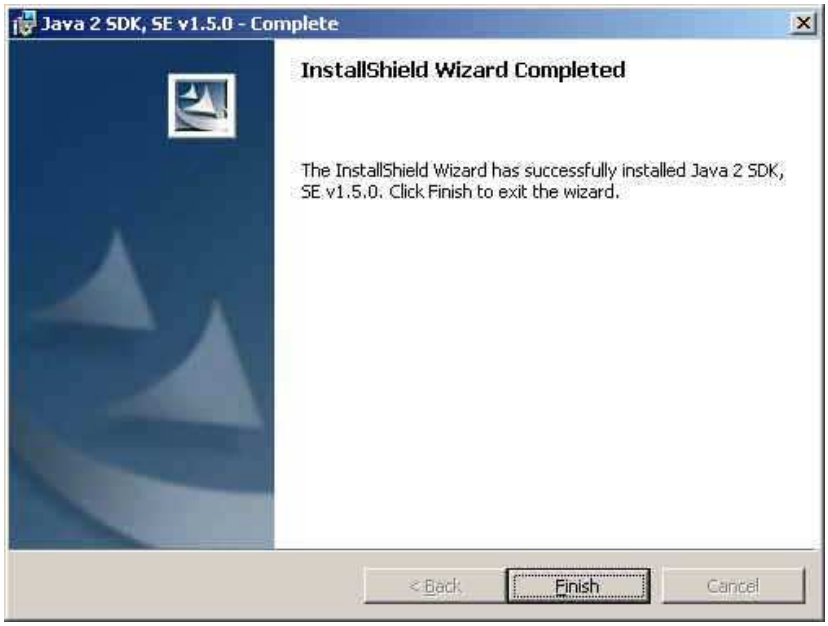
2008-10-10
Planificación de tareas con Spring

2008-10-09
Tutorial de Google Calendar Sync

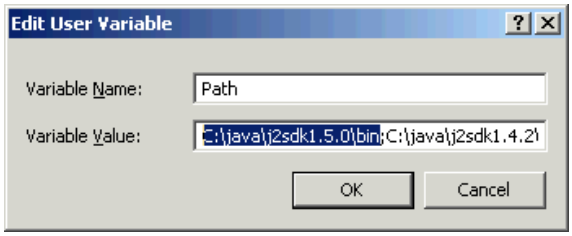
2008-10-06
Instalación de GWT 1.5



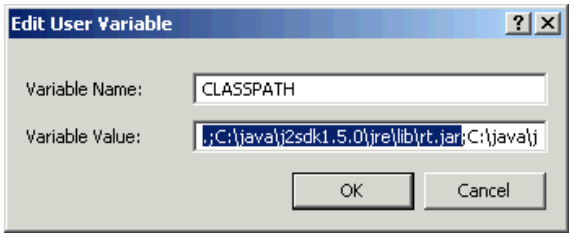
Nos aseguramos que ha finalizado bien



Y cambiamos las variables de entorno para incluir el directorio de los binarios



Y asegurarnos que no tenemos cosas raras con las clases cambiando el classpath



Yo ahora ejecutaría **java -version** para asegurarme que estoy usando la versión adecuada de Java (si tengo muchas en mi máquina)

```
C:\java\j2sdk1.5.0>java -version
java version "1.5.0-beta"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0-beta-b32c)
Java HotSpot(TM) Client VM (build 1.5.0-beta-b32c, mixed mode)
```

Un buen punto para comenzar a trabajar es este enlace donde tiene un resumen (demasiado reducido) de como utilizar las nuevas técnicas.

http://java.sun.com/developer/technicalArticles/releases/j2se15/

Vamos a ir poco a poco poniendo unos ejemplos sencillitos

Colecciones Genéricas

Las colecciones en Java pueden almacenar cualquier tipo de objeto (Object). El único problema que podemos encontrar es que si metemos un tipo inadecuado, el error nos dará en tiempo de ejecución.

Con Java 1.5 y la introducción de los tipos genéricos (gererics), podemos asegurarnos que en compilación se valida la asignación de tipos.

Cambia un poco la notación pero no es muy significativo el cambio...

view plain print ?

```
01. import java.util.*;
02.
03. class java15
04. {
05.
06.     public static void main(String [] params)
07.     {
08.
09.         ArrayList<Integer> milista =  new ArrayList<Integer>();
10.         milista.add(0, new Integer(51));
11.
12.         int total = milista.get(0).intValue();
13.         System.out.println("El valor es " + total);
14.     }
15. }
16.
```

Si compilamos, nos dará un error y es porque tenemos que especificar que deseamos incluir en compilación las particularidades de Java 1.5.

```
C:\java\j2sdk1.5.0>javac *.java
java15.java:8: '(' or '[' expected
                ArrayList<Integer> milista =  new ArrayList<Integer>();
                                                                  ^
1 error

C:\java\j2sdk1.5.0>
```

Lo solucionamos fácilmente añadiendo una opción de compilación

java -source 1.5 *.java

Si ahora pretendemos introducir en nuestro ArrayList una cadena de caracteres...

view plain print ?

```
01. import java.util.*;
02.
03. class java15
04. {
05.
06.     public static void main(String [] params)
07.     {
08.         ArrayList<Integer> milista = new ArrayList<Integer>();
09.         milista.add(0, new Integer(51));
10.
11.         milista.add(1, new String("cadena"));
12.
13.         int total = milista.get(0).intValue();
14.         System.out.println("El valor es " + total);
15.     }
16. }
17.
```

Encontramos el siguiente error:

```
C:\java\j2sdk1.5.0>javac -source 1.5 *.java
java15.java:12: cannot find symbol
symbol  : method add(int,java.lang.String)
location: class java.util.ArrayList<java.lang.Integer>
                milista.add(1, new String("cadena"));
                       ^
1 error

C:\java\j2sdk1.5.0>
```

Nuestras propias clases genéricas

Las clases nativas de Java ahora soportan esta funcionalidad (como acabamos de ver con las colecciones).

Nosotros podemos crear nuestras propias clases con estas características (y recuerda mucho a C++ aunque existen sutiles diferencias .... sobre todo en lo que a generación de código se refiere)

view plain print ?

```
01. import java.util.*;
02.
03.
04. class Pedido <x extends Object>
05. {
06.     public void set(x param)
07.     {
08.         this.actual = param;
09.     }
10.
11.     public x get()
12.     {
13.         return actual;
14.     }
15.
16.     private x actual;
17. }
18.
19.
20. class java15
21. {
22.
23.     public static void main(String [] params)
24.     {
25.         Pedido<String> a = new Pedido<String>();
26.         a.set("Hola");
27.         System.out.println("El valor es " + a.get());
28.
29.         Pedido<Integer> b = new Pedido<Integer>();
30.         b.set(new Integer(12));
31.         System.out.println("El valor es " + b.get());
32.
33.     }
34. }
35.
```

Iteraciones simplificadas

Una de las ventajas más inmediatas que podemos encontrar al usar jsdk 1.5 es la simplificación de los bucles...

view plain print ?

```
01. import java.util.*;
02.
03.
04. class java15
05. {
06.
07.     public static void main(String [] params)
08.     {
09.         ArrayList<Integer> milista =  new ArrayList<Integer>();
10.         milista.add(0, new Integer(51));
11.         milista.add(1, new Integer(12));
12.
13.         for(Object x : milista)
14.         {
15.             System.out.println("El valor de x es " + x);
16.         }
17.     }
18. }
19.
```

Podemos ver la salida .... y funciona correctamente

```
C:\java\j2sdk1.5.0>java java15
El valor de x es 51
El valor de x es 12
```

También se puede utilizar para los arrays de tipos simples y objetos...

Conversiones implicitas

Otra de las cosas curiosas es la nueva capacidad para realizar conversiones directas entre tipos auxiliares (como Integer) y los tipos nativos que representan (int).

En el siguiente código comprobamos que **no** es necesario un casting como sería de prever , algo como:

```
int total = milista.get(0).intValue();
```

```
view plain  print  ?
01. import java.util.*;
02.
03. class java15
04. {
05.
06.     public static void main(String [] params)
07.     {
08.         ArrayList<Integer> milista = new ArrayList<Integer>();
09.         milista.add(0, new Integer(51));
10.
11.         int total = milista.get(0);
12.         System.out.println("El valor es " + total);
13.     }
14. }
15.
```

Enumeraciones

En Java no existen (mejor dicho, existían) los tipos enumerados de datos (al estilo C) pero tenemos ahora la opción:

```
view plain  print  ?
01. import java.util.*;
02.
03.
04. class java15
05. {
06.     static public enum prioridad{ alta, media, baja};
07.
08.
09.     public static void main(String [] params)
10.     {
11.         prioridad actual = prioridad.alta;
12.         System.out.println("La prioridad es: " + actual );
13.     }
14. }
15.
```

El resultado es:

```
C:\java\j2sdk1.5.0>java java15
La prioridad es: alta
```

Si decompilamos el código podemos ver que se está haciendo por dentro ..... así que cuidadito con estas cosas (el peso de la clase).

view plain print ?

```
import java.io.PrintStream;

class java15
{
    public static class prioridad extends Enum
    {

        public static final prioridad[] values()
        {
            return (prioridad[])_2B_VALUES.clone();
        }

        public static prioridad valueOf(String s)
        {
            prioridad aprioridad[] = _2B_VALUES;
            int i = aprioridad.length;
            for(int j = 0; j < i; j++)
            {
                prioridad prioridad1 = aprioridad[j];
                if(prioridad1.name().equals(s))
                    return prioridad1;
            }

            throw new IllegalArgumentException(s);
        }

        public volatile int compareTo(Enum enum)
        {
            return super.compareTo((prioridad)enum);
        }

        public static final native prioridad alta;
        public static final native prioridad media;
        public static final native prioridad baja;
        private static final prioridad _2B_VALUES[];

        static
        {
            alta = new prioridad("alta", 0);
            media = new prioridad("media", 1);
            baja = new prioridad("baja", 2);
            _2B_VALUES = (new prioridad[] {
                alta, media, baja
            });
        }

        public prioridad(String s, int i)
        {
            super(s, i);
        }
    }


    java15()
    {
    }

    public static void main(String args[])
    {
        prioridad prioridad1 = prioridad.alta;
        System.out.println((new StringBuilder()).append
      ("La prioridad es: ").append(prioridad1).toString());
    }
}
```

Funciones con parámetros variables

Otra característica interesante añadida es la capacidad de disponer de métodos con el número de parámetros variable.

```
view plain  print  ?
01. import java.util.*;
02.
03.
04. class java15
05. {
06.     static void varparam(Object ... args)
07.     {
08.         String resultado = "";
09.
10.         for (int i=0;i < args.length; i++)
11.         {
12.             resultado = resultado + " " + args[i];
13.         }
14.
15.         System.out.println("El resultado es: " + resultado);
16.     }
17.
18.
19.     public static void main(String [] params)
20.     {
21.         varparam(1,2,3,"Cuatro");
22.     }
23. }
24.
```

Si volvemos a descompilar podemos verificar el código que está generando. En este caso vemos que nos simplifica de un modo sencillo el problema de este tipo de funciones (se suelen usar para componer mensajes).

```
view plain  print  ?
01. import java.io.PrintStream;
02.
03. class java15
04. {
05.
06.     java15()
07.     {
08.     }
09.
10.     static transient void varparam(Object aobj[])
11.     {
12.         String s = "";
13.         for(int i = 0; i < aobj.length; i++)
14.             s = (new StringBuilder()).append(s).append(" ").append(aobj[i]).toString();
15.
16.         System.out.println((new StringBuilder()).append("El resultado es: ").append(s).toStrin
17.     }
18.
19.     public static void main(String args[])
20.     {
21.         varparam(new Object[] {
22.             Integer.valueOf(1), Integer.valueOf(2), Integer.valueOf(3), "Cuatro"});
23.     }
24. }
25.
26.
27.
28.
```

Printf en Java

Cuando he visto esto casi me echo a llorar de la ilusión. El famoso printf tan usado en C ya forma parte del lenguaje Java..... francamente estupendo. De un modo sencillo (cuando lo conoces) te permite definir como representar multitud de tipos de datos sin mucho trabajo.

```
view plain  print  ?
01.   import java.util.*;
02.
03.
04. class java15
05. {
06.     public static void main(String [] params)
07.     {
08.         float valor = 10.23f;
09.
10.         System.out.printf("El valor es %+14.4f",valor);
11.     }
12. }
13.
```

Hemos forzado la salida a 14 posiciones, con el simbolo siempre y cuatro decimales...

```
C:\java\j2sdk1.5.0>java java15
El valor es       +10,2300
```

Grupos de constantes sin interfaces

Hay veces que declaramos un interfaz en Java únicamente para tener agrupadas una serie de variables y diciendo que una clase implementa ese interfaz, tenemos acceso a ellas.

No está demasiado bien hecho porque estamos acoplando una clase con un interfaz, innecesariamente (esto es un conocido anti-patrón). ¿Qué pasa si alguien añade un método a ese interfaz? Pues que muchas clases se ven obligadas a implementarlo sin demasiado sentido...

En Java 1.5 podemos resolver el problema de la siguiente situación

Creamos un interfaz de utilidades ....

view plain print ?

```
01. package utilidades;
02.
03. public interface constantes
04. {
05.     public static final int ERROR = 0;
06.     public static final int ADVERTENCIA = 1;
07.     public static final int INFORMACION = 2;
08. }
09.
10.
```

En nuestra clase introducimos un import static ..... vamos, para todos los efectos, es como las antiguas macros de preproceso en C.

view plain print ?

```
01.
02. import java.util.*;
03. import static utilidades.constantes.*;
04.
05.
06. class java15
07. {
08.
09.     public static void main(String [] params)
10.     {
11.
12.         int tipoLog = ADVERTENCIA;
13.         System.out.println("El tipo de traza es " + tipoLog );
14.     }
15. }
16.
17.
```

Usamos el comando de compilación

javac -d . -source 1.5 *.java

Si ahora descompilamos vemos que no queda referencia del interfaz ..... estupendo

view plain print ?

```
01. import java.io.PrintStream;
02.
03. class java15
04. {
05.
06.     java15()
07.     {
08.     }
09.
10.     public static void main(String args[])
11.     {
12.         int i = 1;
13.         System.out.println((new StringBuilder()).
14.         append("El tipo de traza es ").append(i).toString());
15.     }
16. }
17.
18.
19.
```

Referencias

Una presentación estupenda http://www.cs.rit.edu/~ats/lfl/2003-2/java/lea.pdf

Comentario

Parece que nos obligan a que el trabajo de los programadores, estos años, tiene que ser aprender tecnología y estar en una continua I+D (en vez de resolver problemas de negocio con tecnologías estables y cada vez más sencillas). Hay tantas cosas nuevas y opciones para hacer la misma cosa que uno puede perder fácilmente el norte.....

Sobre el Autor ..

- Puedes opinar sobre este tutorial haciendo clic aquí.

- Puedes firmar en nuestro libro de visitas haciendo clic aquí.
- Puedes asociarte al grupo AdictosAlTrabajo en XING haciendo clic aquí.
- Añadir a favoritos Technorati. ADD THIS BLOG TO MY Technorati FAVORITES

Recuerda

Autentia te regala la mayoría del conocimiento aquí compartido (Ver todos los tutoriales). Somos expertos en: J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ... y muchas otras cosas.

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?, ¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos ...

Autentia = Soporte a Desarrollo & Formación.

info@autentia.com

Servicio de notificaciones:

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales.

Formulario de subcripción a novedades:

E-mail [ ] Aceptar

Tutoriales recomendados

Nombre	Resumen	Fecha	Visitas	pdf
Forzar diálogo Guardar Como en JSPs	Os mostramos como afrontar tareas comunes en JSP: Forzar el diálogo Guardar Como al generar dinámicamente un fichero desde un JSP y asegurarnos que no se cachean nuestros ficheros (probado en IExplorer 6)	2004-02-05	20681	pdf
Soporte de Asserts en Java 1.4.x	Os mostramos como utilizar los asserts en Java (disponibles a partir de la versión 1.4)	2004-01-30	8408	pdf
AspectJ, Programación con Aspectos	Os mostramos como configurar AspectJ (extensión Java para la programación basada en aspectos) y un pequeño ejemplo para medir la velocidad de una función sin alterar su código.	2004-01-30	15329	pdf
CMMI. Modelo de Madurez Software	Os introducimos a CMMI o Capability Maturity Model Integration. CMMI es un modelo de calidad exigido por el gobierno americano a sus proveedores para el desarrollo de Software. Su conocimiento es esencial para reducir costes de desarrollo.	2004-02-05	61113	pdf
JSP 2.0, JSTL y Lenguaje de expresiones	Os mostramos las novedades de JSP 2.0: Nuevas librerías estandar de etiquetas y el lenguaje de expresiones con ejemplos de acceso a base de datos, XML y XSL en JSP	2003-10-18	50138	pdf
Transformación de XML y XSL en JSPs	Os mostramos como poder utilizar XML y XSL en JSPS, combinado con el Patrón MVC	2003-12-06	27587	pdf

Nota: