

Java SE 6



(aka "Mustang")



www.jugtorino.it

Java SE 6: Aggiornamento!

Bruno Bossola



JavaSE6: what's new?

- Web Services
- Scripting
- Desktop interaction
- Miglioramenti a Swing/AWT
- JavaDB
- ...altre cosette

Disclaimer: questo per quanto riguarda la mia esperienza :)



Web Services





Web Services

- Nuove API integrate
 - JAX-WS 2.0 (appuntamento!)
 - POJO-based
 - JAXB 2.0
 - schema compiler (*xljc*)
 - schema generator (*schemagen*)
 - runtime binding (*un*)*marshalling*
 - JAXP 1.4
 - bugfix della 1.3 inclusa in Java5
 - include StaX



Publiccare un web service [1]

- Creare un POJO che implementa il servizio
- Annotarlo con `@WebService`
- Opzionalmente predisporre l'injection di un `WebServiceContext`
- Pubblicare il servizio con il metodo `Endpoint.publish()`



Publiccare un web service [2]

- Java6 include un HTTP server
 - lightweight
 - impostazioni di default per il threading "ragionevoli"
- Il WSDL viene generato dinamicamente
- Il `WebServiceContext` è iniettato a runtime con i dati della chiamata
- Il servizio può essere fermato con `Endpoint.stop()`



Publicare un web service [3]

CODICE!



Consumare un web service

- Usare `wsimport` per generare le classi necessarie “puntandolo” al WSDL del web service desiderato
- Usare le classi generate per invocare le operazioni sul web service



Consumare un web service

CODICE!



Scripting





Scripting

- Linguaggi dinamicamente tipati
- Variabili usate senza definizione
- Conversioni di tipo automatiche
- Interpretati
- Risultati ottenibili velocemente, feedback immediato

Non ancora convinti?



Java vs Ruby

```
public class Filter {
    public static void main(String[] args) {
        List list = new java.util.ArrayList();
        list.add("Tim"); list.add("Ike"); list.add("Tina");
        Filter filter = new Filter();
        for (String item : filter.filterLongerThan(list, 3)) {
            System.out.println( item );
        }
    }
    public List filterLongerThan(List list, int length) {
        List result = new ArrayList();
        for (String item : list) {
            if (item.length() >= length) { result.add( item ); }
        }
        return result;
    }
}
```



Java vs Ruby

```
list = ['Tim', 'Ike', 'Tina']  
list.select {|n| n.length > 3}.each {|n| puts n}
```



Scripting: perchè?

- risolvere velocemente alcune classi di problemi
- ottimo per permettere all'utente di effettuare customizzazioni
- utile per il testing e per lo sviluppo prototipale



Scripting framework

- JSR 223 (si “ispira” ad Apache BSF)
- supporta linguaggi di script “pluggabili”
- incluso in Java SE6
 - incorpora anche lo scripting engine Rhino di Mozilla
 - comunicazione Java-Javascript!
- sono disponibili un insieme di scripting engine compliant
 - scripting.dev.java.net



Scripting: API

- **ScriptEngine**
 - esegue degli script
 - permette di scambiare dati dal mondo java al mondo script
 - Bindings e ScriptContext
- **Invocable (opzionale)**
 - permette di invocare funzioni definite all'interno dello script
- **Compilable (opzionale)**
 - permette di compilare in uno stato intermedio x invocarlo successivamente



Scripting: API

- **ScriptEngineManager**
 - consente di ottenere un'istanza di uno **ScriptEngine**
 - `getEngineByName()`
 - `getEngineByExtension()`
 - `getEngineByMimeType()`
 - `getEngineFactories()`

```
ScriptEngine engine;  
engine = ScriptEngineManager.getEngineByName("JavaScript");
```



ScriptEngine

- `eval()`
 - consente di eseguire uno script passato come una stringa o un Reader
 - permette di passare dei "bindings", ovvero mappe chiave / valore condivise
 - lo scope può essere ristretto all'interno di uno `ScriptContext`
 - `ScriptContext.GLOBAL_SCOPE`
 - `ScriptContext.ENGINE_SCOPE`



ScriptEngine

CODICE!



Desktop interaction





Desktop interaction

- Tray Icon
- Splash screen
- Interazioni desktop standard
 - mail
 - browse
 - edit / open / print
- Non sono supportate ovunque!
 - `Desktop.isDesktopSupported()`



Tray icon

- Nuove API che permettono l'accesso al system tray
- Consentono di
 - installare un'icona nel tray
 - visualizzare un popup menu
 - visualizzare un tooltip



Tray icon

CODICE!



Splash screen

- Splash screen all'avvio!
- prima che la JVM parta (del tutto!)
 - al lancio:
java -splash:image.gif ...
 - nel MANIFEST.MF del jar:
Splashscreen-Image: image.gif
- formati immagini diversi
 - GIF
 - PNG
 - JPEG



Splash screen

- supporto ad effetti grafici
 - trasparenza, traslucenza (boh? :D), animazioni
- splash chiuso automaticamente alla prima top-window aperta
- possibile interazione post-splash:

```
SplashScreen splash = SplashScreen.getSplashScreen();  
Graphics2D g = splash.createGraphics();
```

```
// modificate lo splash qui!
```

```
splash.update();
```



Splash screen

CODICE!



Interazioni desktop

- E' disponibile una nuova classe:
`java.awt.Desktop`
- Un enumerazione `Action` incapsula le varie azioni che si possono fare:
 - BROWSE
 - EDIT
 - MAIL
 - OPEN
 - PRINT
- Un analoga serie di metodi per eseguire le azioni (e per verificarne il supporto)



Interazioni desktop

CODICE!



Miglioramenti a Swing/AWT





Miglioramenti a Swing/AWT

- Nuova gestione finestre modali
- Migliorato il supporto alla stampa dei componenti Swing
- Nuovo SwingWorker (sigh...)
- Nuovo layout manager GroupLayout
- Sort e filtraggio "embedded" nelle JTable



Finestre modali

- Definiti nuovi tipi di modalità
 - modeless
 - non bloccante
 - document-modal
 - blocca tutte le finestre della gerarchia
 - application-modal
 - blocca tutte le finestre dell'applicazione
 - toolkit-modal
 - blocca tutte dello stesso toolkit (applets)
- E' possibile escludere una singola finestra dal vincolo di modalità



Finestre modali

DEMO!

...il codice meglio che non lo vediate :)



Nuovo SwingWorker

- Ora **standard** all'interno di Swing
 - `javax.swing.SwingWorker`
 - Usa `java.util.concurrent`
 - ancora più complicato (evvai!)
- Template method ridefinibili
 - `doInBackground()`
 - `process() / done()`
- Supporta `ChangeListener`
 - in una `propertyChange` si possono effettuare operazioni sulla UI



JTable: sort

- `sort()` sulle colonne implementabile con una singola API
 - `setAutoCreateRowSorter(true)`
- E' possibile definire i propri comparatori
- E' possibile definire ordinamenti di secondo, terzo livello, ...



JTable: filtering

- è possibile associare un **RowFilter** per limitare le righe visualizzate
 - uso di regexp
 - supporto "nativo" a date e numeri



JTable enhancements

DEMO!



JavaDB

- il JDK contiene un db relazionale
 - Apache Derby, aka JavaDB
 - footprint piccolissimo (2mb RAM)
- **non** è contenuto nel JRE
- completamente JDBC compliant
- funziona sia "embedded" che "client-server"

Uhm... che succederà ora con MySQL, recente acquisto di Sun?



Altre cosette :)

- JMX di monitoraggio con start "a richiesta"
- JDBC 4.0
 - BLOB / CLOB funzionanti :)
 - SQLXML data type
 - nuove sottoclassi di SQLException
- annotation processing pluggabile
- accesso al compilatore migliorato
- miglioramenti su look&feel
- performance migliorate (davvero!)



Credits

- Parte di questo materiale è farina di Sang Shin
<http://www.javapassion.com>
- Parte di questo materiale viene da Danny Coward
<http://java.sun.com/developer/technicalArticles/J2SE/>
- Parte di questo materiale è codice mio e di altri :)



Q&A
