



L'altra faccia del web service: accesso da un cellulare Java ME

Stefano Sanna
<http://www.gerdavax.it>



Parliamo di...

- Web Services su dispositivi mobili: ora si può!
- Web Services API for J2ME
 - JAXP: decodificare XML
 - JAX-RPC: invocare metodi remoti
- Dalla teoria alla pratica...
- Conclusioni



Web Services API for J2ME

- Il Java Community Process ha definito la specifica JSR 172, che porta nel mondo Java ME:
 - Un subset di **JAXP 1.2**
 - Un subset di **JAX-RPC 1.1**

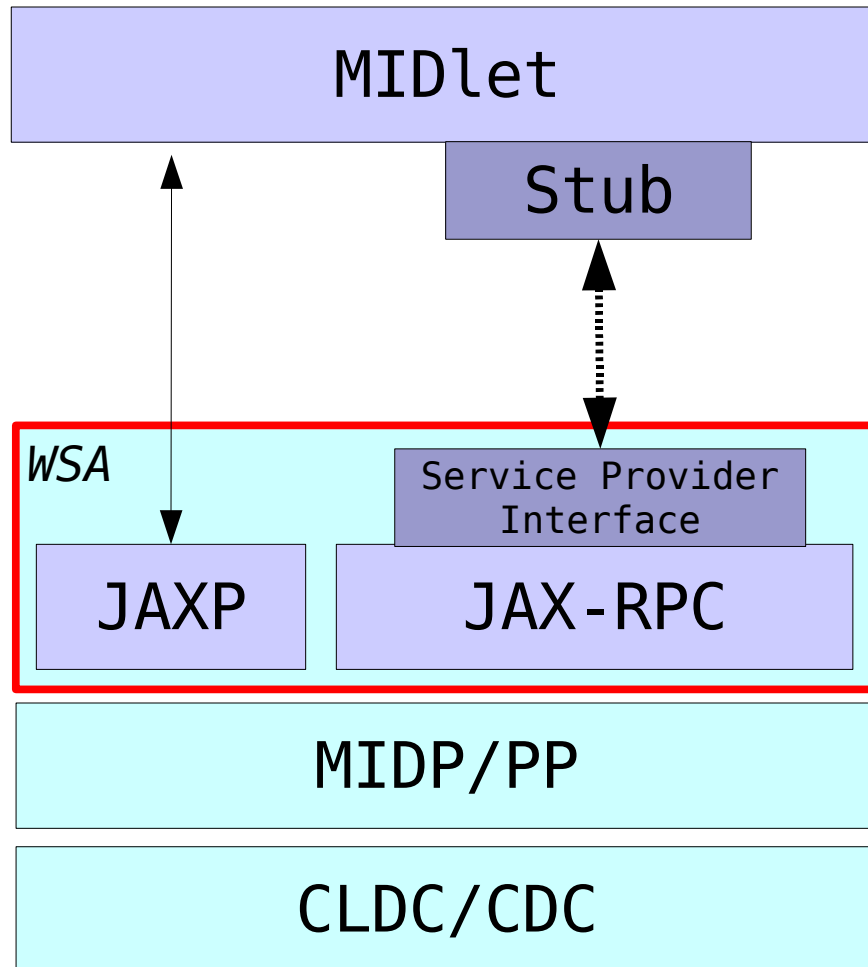


Caratteristiche principali

- Footprint estremamente ridotto (~35KB per JAXP e ~25KB per JAX-RPC)
- Performance ottimizzate per dispositivi low-end
- Supporto a **tutti i profili** basati su CLDC 1.0/1.1 e CDC
- Interfaccia RPC (SPI) **indipendente** dalla particolare implementazione della libreria



Architettura





JAXP 1.2 subset

- E' l'API per il parsing di documenti XML
- Caratteristiche:
 - Requisiti minimi: CLDC 1.0 e 35KB footprint!
 - Sottoinsieme della specifica JAXP 1.2
 - Supporto all'interfaccia SAX 2.0
 - Supporto agli XML namespaces
 - Supporto codifiche UTF-8 e UTF-16
 - L'eventuale validazione con DTD e' opzionale (piuttosto improbabile...)



JAXP

- **javax.xml.parsers**
 - contiene il SAX parser, la relativa factory e le classi delle eccezioni
- **org.xml.sax**
 - contiene il core delle API SAX (Attribute, Locator, InputSource...)
- **org.xml.sax.helpers**
 - contiene la classe DefaultHandler per la gestione degli eventi di parsing



Limitazioni JAXP

- Nessun supporto SAX 1.0 (vedi SAX 2.0)
- Nessun supporto XSLT
- Nessun supporto DOM 1.0 e 2.0
- Validazione opzionale



JAX-RPC 1.1 subset

- E' l'API per l'invocazione remota di metodi attraverso documenti XML
- Caratteristiche:
 - Supporta **WSDL 1.1**
 - Supporta SOAP 1.1 (1.2 in futuro)
 - Supporto XML 1.0 e XML Schema
 - Binding su SOAP e trasporto HTTP (con supporto autenticazione base)
 - Conforme al **WS-I Basic Profile 1.0**



JAX-RPC

- **javax.xml.rpc**
 - contiene l'interfaccia Stub
- **javax.microedition.xml.rpc**
 - contiene classi e interfacce della **SPI**
- **javax.xml.namespace**
 - contiene la classe QName
- **java.rmi**
 - contiene l'interfaccia Remote, da cui dipende Stub



WSDL-Java Mapping

xsd:long	long	java.lang.Long
xsd:int	int	java.lang.Integer
xsd:short	short	java.lang.Short
xsd:byte	byte	java.lang.Byte
xsd:float	float	java.lang.Float
xsd:double	double	java.lang.Double
xsd:string	String	
xsd:base64Binary	byte[]	
xsd:hexBinary	byte[]	
xsd:complexType	<i>sequenza di classi e primitivi</i>	
<i>vettori di primitivi e tipi complessi, secondo XML array</i>		



Limitazioni JAX-RPC

- Non sono supportati messaggi SOAP con attachment né message handlers
- Non è gestita la rappresentazione **encoded** di messaggi SOAP (esclusivamente **literal**)
- UDDI non supportato in questa versione
- Non è prevista l'implementazione di endpoint (nessun web service provider su telefonino!)

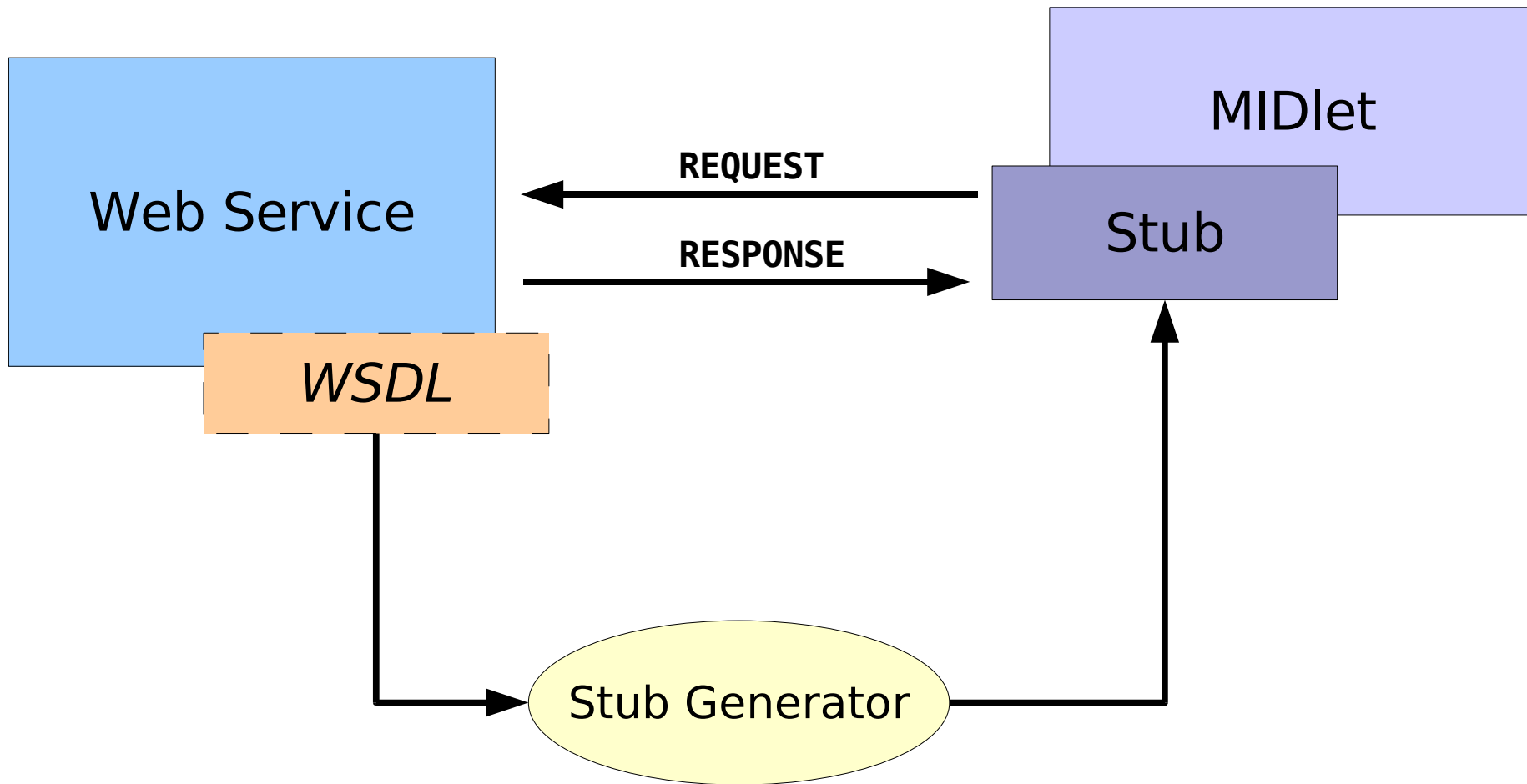


JAX-RPC: invocare metodi remoti

- Utilizzare le routine di invocazione remota all'interno di un mobile web service consumer è estremamente semplice. Occorre:
 - Recuperare il **WSDL** del servizio di interesse e verificare la compatibilità WS-I
 - Generare lo **stub** a partire dal WSDL
 - Invocare i metodi dello stub dalla MIDlet



JAX-RPC: invocare metodi remoti





Stub

- La creazione dello stub (interfaccia e implementazione) viene effettuata **automaticamente** attraverso opportuni strumenti (Stub Generator)
- Lo stub è **indipendente** dalla particolare implementazione della WSA
- Lo stub invoca i metodi delle classi della **Service Provider Interface**



Strumenti di sviluppo

- Attualmente la WSA è supportata, tra gli altri, da:
 - Sun J2ME Wireless Toolkit 2.2 o superiore
 - Nokia Developer's Suite 3.0
 - SonyEricsson SDK 2.2.2 for Java ME Platform
 - IBM WebSphere Studio Device Developer
 - Apache Mirae Project



Il web service utilizzato

- Il web service d'esempio utilizzato in questa presentazione è stato rilasciato e installato su un server domestico ed è raggiungibile all'indirizzo:

<http://gerdavax.dyndns.org:8080/axis/services/ciao>



Utilizziamo il Wireless Toolkit

The screenshot displays the J2ME Wireless Toolkit application window. The 'Utilities' dialog box is open, showing a list of emulators with 'DefaultEmulator' selected. The 'Stub Generator' button is highlighted in the 'Stub tools' section. A red arrow points from the 'Utilities' dialog to the 'Stub Generator Dialog' window, which is also open. The 'Stub Generator Dialog' contains fields for 'WSDL Filename or URL', 'Output Path', and 'Output Package', each with a 'Browse...' button. It also has radio buttons for 'CLDC Version' (CLDC 1.0 and CLDC 1.1, with CLDC 1.1 selected) and 'OK'/'Cancel' buttons at the bottom.



Stub Generator

Stub Generator Dialog

WSDL Filename or URL:
/home/gerdavax/seminari/jug/JavaWinterMeeting2005/sample/HelloWSWorld/res/ciaoDocLit.wsdl **Browse...**

Output Path:
/home/gerdavax/seminari/jug/JavaWinterMeeting2005/sample/HelloWSWorld/src **Browse...**

Output Package:
org.jugsardegna.mobile.ws.hello.stub

CLDC Version

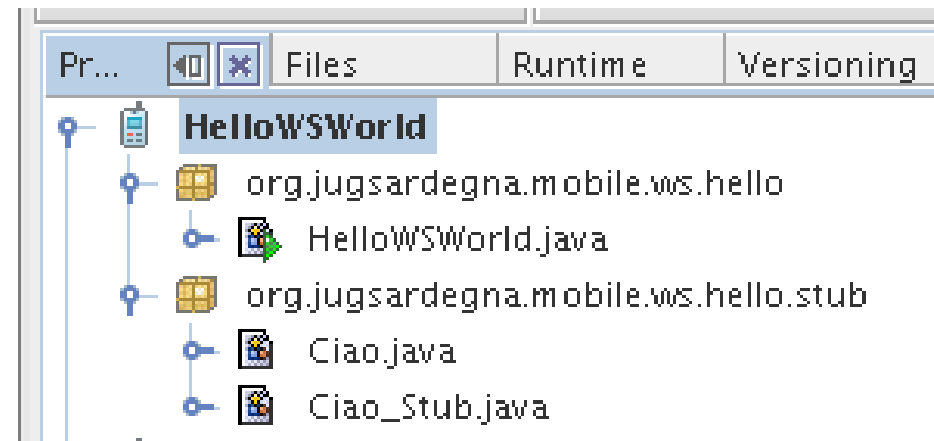
CLDC 1.0
 CLDC 1.1

OK **Cancel**



Lo stub è pronto all'uso

- Lo Stub Generator ha creato:
 - L'interfaccia
 - Lo stub vero e proprio
 - Classi di supporto per ciascun metodo invocato e relativa risposta





L'interfaccia dello stub

```
package org.jugsardegna.mobile.ws.hello.stub;
```

```
import java.rmi.*;
```

```
public interface Ciao extends Remote {
```

```
    public String saluto(String nome) throws  
        RemoteException;
```

```
}
```



Lo stub /1

- Lo stub fa riferimento alla **Service Provider Interface**:

```
package org.jugsardegna.mobile.ws.hello.stub;  
  
import javax.xml.rpc.JAXRPCException;  
  
import javax.xml.namespace.QName;  
  
import javax.microedition.xml.rpc.Operation;  
  
import javax.microedition.xml.rpc.Type;  
  
import javax.microedition.xml.rpc.ComplexType;  
  
import javax.microedition.xml.rpc.Element;
```



Lo stub /2

- Lo stub implementa l'interfaccia pubblica del web service:

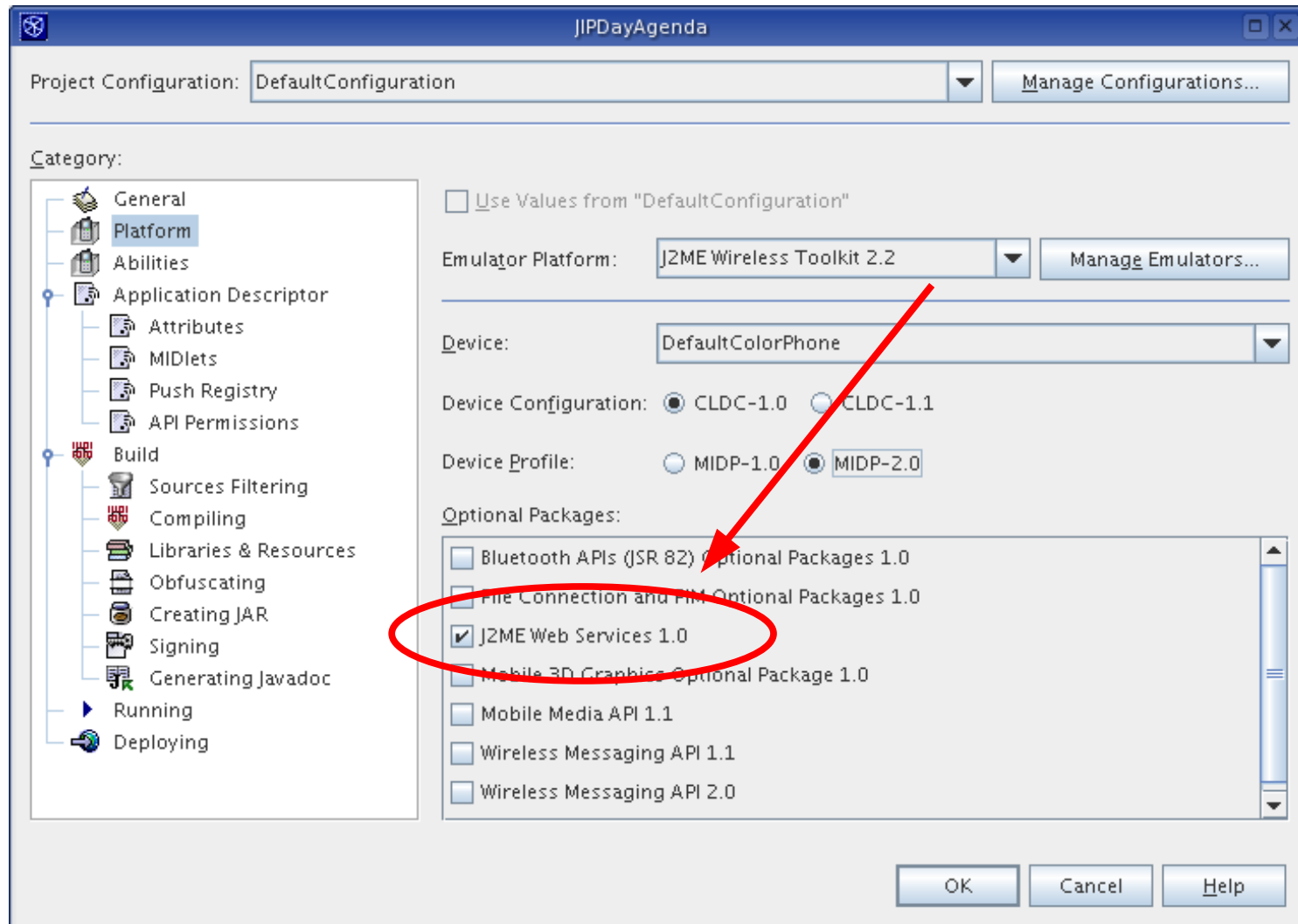
```
public class Ciao_Stub implements  
    org.jugsardegna.mobile.ws.hello.stub.Ciao,  
    javax.xml.rpc.Stub {  
}
```

- L'interfaccia **Stub** definisce i metodi per impostare le proprietà di connessione



Property dello Stub

- Attraverso il metodo `_setProperty()` è possibile impostare la configurazione dello stub:
 - **ENDPOINT_ADDRESS_PROPERTY**
 - **SESSION_MAINTAIN_PROPERTY**
 - **USERNAME_PROPERTY**
 - **PASSWORD_PROPERTY**
- Il Wireless Toolkit imposta l'indirizzo di default dell'endpoint





HelloWSWorld: il “telaio”

```
public class HelloWSWorld extends MIDlet implements
    CommandListener, Runnable {

    private Display display;

    private Form gui;

    private TextField nameField;

    private Command sayHelloCommand;

    private Ciao stub;

    public HelloWSWorld() {    init(); }
}
```




Inizializzazione

```
private void init() {  
    display = Display.getDisplay(this);  
    gui = new Form("HelloSWorld!");  
    nameField = new TextField("Name:", "", 20,  
TextField.ANY);    gui.append(nameField);  
    gui.addCommand(sayHelloCommand);  
    gui.setCommandListener(this);  
  
    stub = new Ciao_Stub(); ←  
}
```



Gestione degli eventi

```
public void commandAction(Command c, Displayable d)
{
    if (c == sayHelloCommand) {
        new Thread(this).start();
    }
}
```



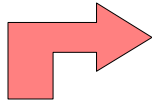


Invocazione remota

```
public void run() {
    String result = "";
    AlertType type = AlertType.INFO;
    try {
        result = stub.saluto(nameField.getString());
    } catch (RemoteException re) {
        result = "Errore invocazione remota!";
        type = AlertType.ERROR;
    }
    finally {
        Alert pop = new Alert("Result", result, null, type);
        pop.setTimeout(Alert.FOREVER);
        display.setCurrent(pop, gui);
    }
}
```



Installazione su PDA /1



MIDlet List [G] [signal] [volume] 0.34 [X]

Please select 'Install' to install a Java applic

Actions [keyboard icon] [up arrow]

MIDlet Install [G] [signal] [volume] 0.36 [X]

Install

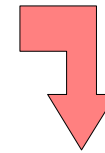
Enter the URL of the MIDlet that you would like to install.

URL:

...

123	1	2	3	4	5	6	7	8	9	0	'	ì	←		
↵	q	w	e	r	t	y	u	i	o	p	è	ù			
⊕	a	s	d	f	g	h	j	k	l	ò	à				
↑	z	x	c	v	b	n	m	,	.	-		←			
Ctrl	àé	\	+									↓	↑	←	→

[keyboard icon] [up arrow]



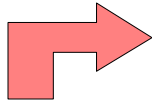
Progress [G] [signal] [volume] 0.36 [X]

Installing a MIDlet from
"http://gerdavax.dyndns.org:8080/HelloWSWorld.jad"

Progress

[keyboard icon] [up arrow]

Installazione su PDA /2



Progress

G [signal] [volume] 0.37

Installing a MIDlet from
"http://gerdavax.dyndns.org:8080/HelloWSWorld.jad"

Progress [progress bar]

Warning [X]

About to install MIDlet "HelloWSWorld" version "1.0" ("32738" bytes) from vendor "Vendor". Proceed?

Yes No

Cancel

Progress

G [signal] [volume] 0.37

Installing a MIDlet from
"http://gerdavax.dyndns.org:8080/HelloWSWorld.jad"

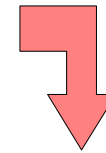
Progress [progress bar]

Warning [X]

This MIDlet suite did not come from a trusted location. Are you sure that you would like to install it?

Yes No

Cancel



Progress

G [signal] [volume] 0.37

Installing a MIDlet from
"http://gerdavax.dyndns.org:8080/HelloWSWorld.jad"

Progress [progress bar]

Success [X]

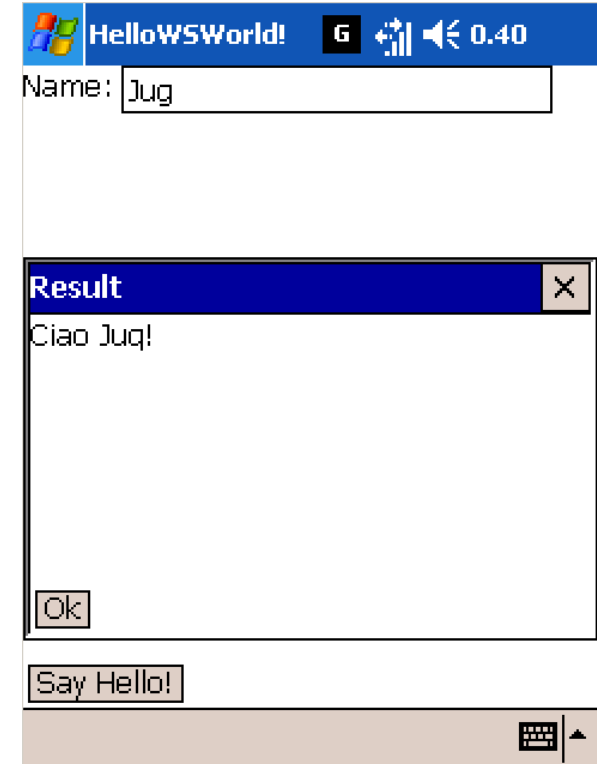
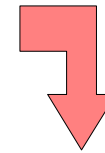
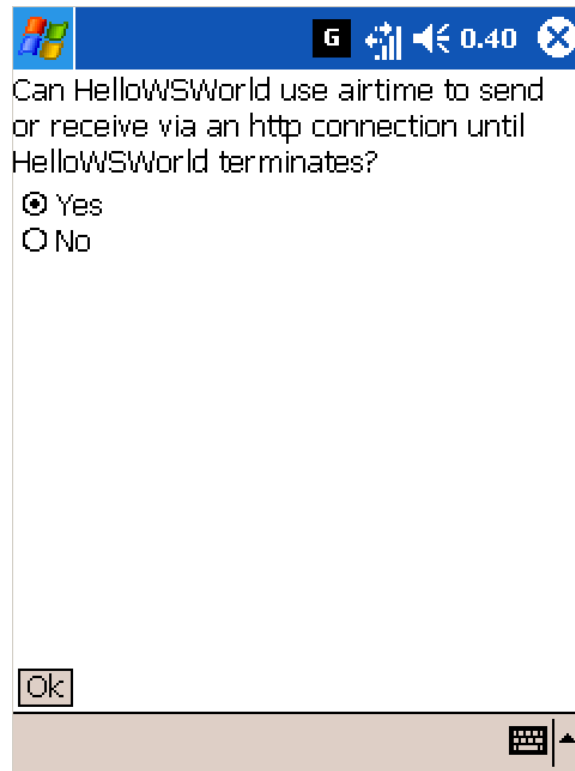
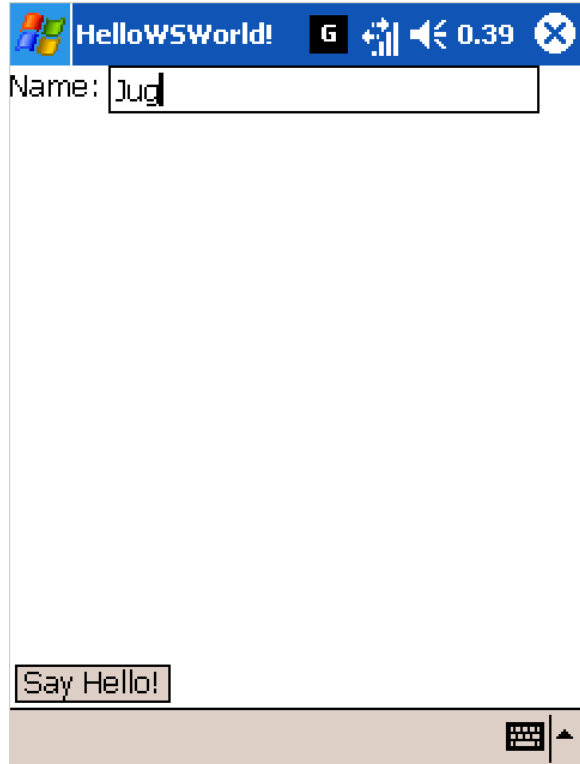
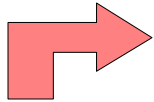
Successfully installed "HelloWSWorld" MIDlet

Ok

Cancel



Esecuzione su PDA





Invocazioni sincrone

- WSA genera chiamate sincrone: assicurarsi che queste siano gestite all'interno di **Thread separati** dalle routine di gestione della GUI
- Non tutti gli esempi reperibili in rete (o generati automaticamente dai tool...) tengono in considerazione questo aspetto...



Invocazioni sincrone

- Senza Thread:

```
if (c == invokeCommand) {  
    stub.startEngine();  
}
```

- La chiamata a `startEngine()` è **bloccante** e può condurre ad un **deadlock** dell'applicazione



Invocazioni su Thread separato

```
class StartEngineTask implements Runnable {  
    public void run() {  
        stub.startEngine();  
    }  
}  
  
if (c == invokeCommand) {  
    new Thread(startEngineTask).start();  
}
```

I dispositivi compatibili...



JIPDay Roma 30 Settembre 2005

Mano ai dispositivi!

- I dispositivi dotati di WSA sono **in arrivo**:
 - Nokia N91, N90
 - Sony-Ericsson W600/W550
 - Palmari compatibili WSDD



JavaPortal è un progetto senza fine di lucro di: XeyTECH

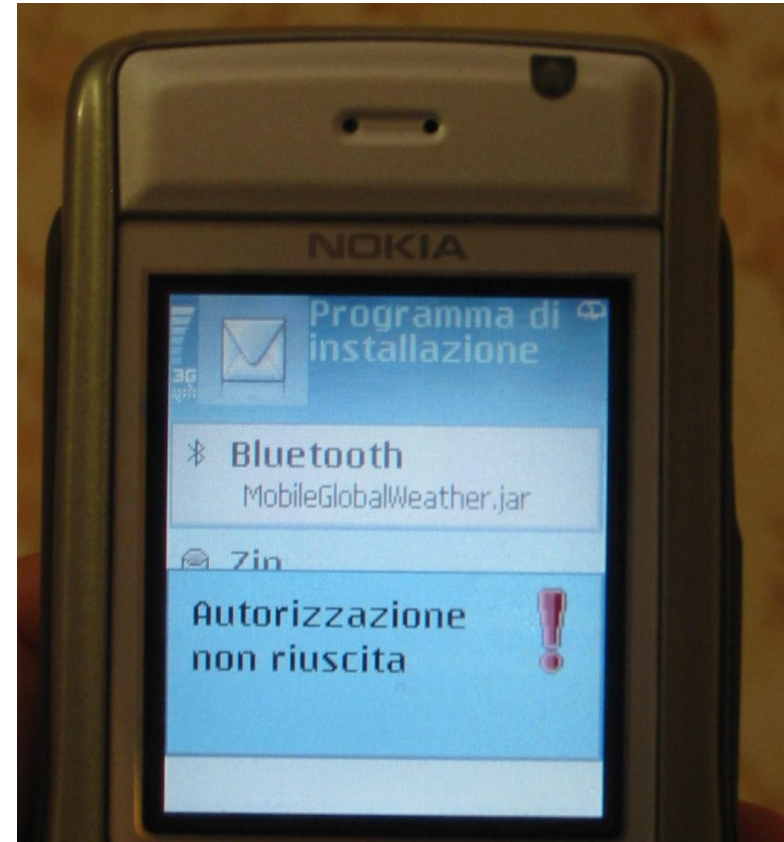
... sono sempre più numerosi!





WSA “custom” sui dispositivi?

- ... la WSA introduce i package **java.***, **javax.*** di dominio dei namespace protetti: la VM del dispositivo ne permette l'uso solo se i package sono contenuti all'interno del runtime del dispositivo!
- Occorre una soluzione alternativa...





Uso delle librerie di un SDK

- Come soluzione temporanea, si può:
 - Prendere il jar file relativo alla implementazione della WSA di un emulatore/SDK (ad esempio, j2me-ws.jar all'interno della directory lib/ del Wireless Toolkit)
 - Disabilitare il supporto Web Services all'interno dell'ambiente di sviluppo
 - Importare il file jar come libreria esterna (da includere nel jar di deployment)
 - Utilizzare un obfuscator per eliminare i nomi dei package riservati
- **IMPORTANTE: questa tecnica può essere utilizzata a scopo di test, in quando il riuso di porzioni degli SDK non è consentito.**



Conclusioni

- La Web Services API introduce nel mondo Java ME una interfaccia standard per l'elaborazione di documenti XML e l'invocazione remota di procedure attraverso infrastruttura web services
- I dispositivi WSA-compatibili sono ormai arrivati: reti a larga banda, terminali potenti ed economici, strumenti efficaci come Axis e de Suite per sviluppo su Java ME ci dicono che i tempi sono maturi per implementare soluzione SOA in ambito mobile



Bibliografia

- Una versione **estesa** di questa presentazione è reperibile sul sito di **Java Italian Portal** (<http://www.javaportal.it>) oppure sul mio sito.
- JSR 172: J2ME Web Services Specification
 - Java Community Process
<http://jcp.org/en/jsr/detail?id=172>
- Web Services APIs for J2ME
C. Enrique Ortiz, IBM DeveloperWorks
 - <http://www-128.ibm.com/developerworks/library/wi-jsr> e successivi



Bibliografia

- La piattaforma J2ME e i Web Services
 - Massimo Carli
Mokabyte n. 94 e 96
<http://www.mokabyte.it>
- La Web Services API di J2ME
 - Emanuela De Vita, Stefano Sanna
Speciale Programmazione Mobile
Computer Programming 150, G.E. Infomedia



Grazie per l'attenzione :-)

Farmer Clem meets the 21st C by lumix2004
<http://www.sxc.hu/browse.phtml?f=profile&l=lumix2004>



L'altra faccia del web service: accesso da un cellulare Java ME

(Versione 1.0)

(C) 2005 Stefano Sanna (gerdavax@tiscali.it)

è garantito il permesso di copiare, distribuire e/o modificare questo documento seguendo i termini della Licenza per Documentazione Libera GNU, Versione 1.1 o ogni versione successiva pubblicata dalla Free Software Foundation. Una copia della licenza in lingua italiana è disponibile presso: <http://www.softwarelibero.it/gnudoc/fdl.it.html>

Realizzato in ambiente Linux con OpenOffice 2.0

Tutti i marchi commerciali sono di proprietà dei rispettivi titolari e sono stati citati in questa presentazione a solo scopo illustrativo.