

Dev

La rivista che ti insegna a programmare

Update & Trial

Il rilascio del software

DB

Note di Data Warehouse:
Tecnica di naming convention

C++

FLTK: verso la nuova release

C#

Diagrammi su misura in .NET

JAVA ME

Quando anche il cellulare
vuole giocare al sudoku...

JAVA

SWIXML: widgets
e personalizzazioni

Provateli per voi

Nokia 770 Internet Tablet

PuTTY 1.3.2 per Symbian OS,
ora SSH è anche sul telefonino

Firebird il database opensource
per eccellenza

Programmo
Subito

Configurare l'ambiente di sviluppo
per Java Servlet e JSP

Internazionalizzazione con Qt



CrackME web edition: • Una sfida
sotto l'albero

Linux: • C# e mono:
Gnome# e Glade #

Linux: • Sourceforge.net



GRUPPO EDITORIALE
INFOMEDIA
THE SOFTWARE SIDE OF YOUR MIND

ISSN 1124-5488



ISSN 1124-5488

Sped. abb. post 45% articolo 2 comma 20/B legge 662/96 - Filiale di Bologna

di Roberto Casarini
rcasarini@infomedia.it

Updating ... al 2006

Pochi giorni ci separano dalle (sospirate?) festività natalizie. In quest'ultimo lembo di anno, molti dei nostri PC (almeno quelli in ufficio), resteranno sicuramente spenti per qualche giorno. Mentre siamo alle prese, nel momento in cui scrivo, con la chiusura di questo numero è inevitabile una riflessione sul bilancio per DEV sull'anno trascorso.

Oltre a promuovere un'informazione tecnica di qualità professionale, DEV continua a fungere da appuntamento mensile, per informatici e appassionati della programmazione, proponendosi come punto d'incontro e di scambio di conoscenze. Nel corso di questi dodici mesi, il ventaglio di argomenti trattati negli speciali è stato variegato: dai temi più specifici per la professione di informatico alle introduzioni più orientate ai più giovani. Ci siamo occupati di programmazione dei videogiochi, del sistema operativo Solaris 10 di Sun in concomitanza col passaggio del prodotto da commerciale a open-source, di argomenti amministrativi. Senza dimenticare lo speciale di Settembre relativo alla programmazione 3D, dove in 20 pagine sono stati sintetizzati i prerequisiti di base per lo sviluppo di una applicazione con una interfaccia grafica 3D.

Oltre al tema centrale della programmazione, ci siamo occupati anche di "Sistemi e-commerce" e di "Codice della privacy": pur non essendo argomenti strettamente di tecnica informatica, hanno ovviamente riscosso un discreto interesse da parte vostra.

Lo speciale di questo mese, "Update & Trial", rappresenta il modo migliore per concludere questo 2005: l'aggiornamento del software, in modo pratico e auspicabilmente indolore, rappresenta un prerequisito fondamentale sia per gli utenti sia per noi tecnici. E non va sottovalutato sia dal punto di vista dei repentini cambi di tecnologia, sia dal punto di vista della sicurezza. Man mano che cresce il proprio "parco" di applicazioni sviluppate, è bene non dimenticarsi di quali tool preziosi ci vengono incontro nel lavoro quotidiano per mettere a punto le procedure di deployment e installazione. Soprattutto a causa della velocità di rilascio di nuove release delle piattaforme su cui lavoriamo, che spesso impongono degli upgrade al software con conseguente rilascio di nuove versioni dei nostri applicativi. Questo ritmo sostenuto, tra l'altro, quasi non lascia la possibilità di respirare: aggiornare e rilasciare nuovamente le applicazioni a tutti i clienti è un'attività periodica che richiede una notevole quantità di tempo.

Prima di lasciarvi alla lettura di questo numero di fine d'anno, ne approfitto per "consacrare" il primo vincitore della sfida proposta dalla rubrica CrackMe "web edition": il più rapido è stato Massimo Ronca, congratulazioni! Per i dettagli vi rimando al prossimo numero, dove fra l'altro scoprirete le nuove regole che verranno adottate per decretare i futuri vincitori della nuova "versione" di CrackMe. Non mi resta che augurarvi buone festività e anche buona lettura.

Roberto Casarini



DADA

Un solo Mondo. Una sola Rete.

DADA S.p.A., quotata al TechStar di Borsa Italiana, è la società leader in Italia nel mondo delle community e dell'intrattenimento. Per la propria sede di Firenze, ricerca uno

Sviluppatore Senior

Ruolo: il candidato prescelto si occuperà dello sviluppo di applicativi per il mondo Web e Mobile.

Requisiti: Laurea in Informatica o cultura equivalente. Richiesta ottima conoscenza di perl e apache su sistemi operativi linux/unix. Esperienza di sviluppo di software per il web integrato con database relazionali (mysql e oracle) e conoscenza di c e php.

Si richiede: Esperienza significativa di almeno due/tre anni in posizione analoga. Residenza a Firenze o in zone limitrofe.

Sede di lavoro: Firenze

Candidature a: recruiting@dada.net



Sei agli inizi? Non riesci a risolvere i primi problemi? Niente paura! Iscriviti alle mailing-list Infomedial!

programmario_subito@infomedial.it
e linux-it@infomedial.it

I listati e i programmi di utilità che corredano gli articoli sono prelevabili gratuitamente dal sito <http://www.infomedial.it/DEV> oppure da <ftp://ftp.infomedial.it/pub/DEV/Listati/>



Lo speciale di questo numero: Update & Trial

- 18 Creare una trial in C# di Gian Maria Ricci
- 23 Update automatico di applicazioni .NET di Fabio Rizzati e Stefano Franco
- 28 Automatizzare le installazioni e gli aggiornamenti con ZENworks di Gianluigi Spagnuolo
- 32 Introduzione a Java Web Start di Demetrio Filocamo
- 36 Implementare meccanismi "Update & Trial" su Java ME di Stefano Sanna

Programmazione

- 40 Database -Note di Data Warehouse: Tecniche di naming convention (PARTE 2) di Massimo Cenci
- 44 C++ -FLTK: verso la nuova release di Matteo Lucarelli
- 48 C# -Diagrammi su misura in .NET di Stefano Franco
- 51 Cellulari -Il protocollo AT+ dei telefoni cellulari. Comandi speciali Siemens:cenni finali sull'Organizer (ULTIMA PARTE) di Vito Vessia
- 56 Java ME -Quando anche il cellulare vuole giocare al sudoku... (PARTE 1) di Stefano Pace
- 61 Java -SwiXml: widgets e personalizzazioni (ULTIMA PARTE) di Nunzio Fiore
- 66 Web -Digital Divide: what's new ? what's not ? (and so what ?) (ULTIMA PARTE) di Cesare Zavalloni

Rubriche

- 17 CrackMe web edition -Una sfida sotto l'albero a cura di Fabio Scagliola
- 90 Game Dev -J2ME e la grafica -di Mauro Pesci

Programmo Subito

Numero 88

- 69 **Livello 1** Configurare l'ambiente di sviluppo per Java Servlet e JSP - di Umberto Annunziata
- 73 **Livello 2** Internazionalizzazione con Qt - di Fabrizio Angius

Linux Subito

Numero 67

- 78 **Livello 2** C# e mono: Gnome# e Glade# (PARTE 3) di Gianluigi Spagnuolo
- 85 **Livello 1** Sourceforge.net (PARTE 5) - di Alessio Saltarin

In Prova

Nokia 770 Internet Tablet

di Davide Carboni e Stefano Sanna

08

PuTTY 1.3.2 per Symbian OS, ora SSH è anche sul telefonino

di Paolo De Nicolis

11

Firebird il database opensource per eccellenza

di Massimo Giambona

14

Ai lettori

Doktor Morb a cura di Luigi Morelli

Puga di mezzanotte

39

E-ducational a cura di Luigi Morelli

Musei online

55

Linux 001 a cura di Remigio Darino

Distro

65

Free Reading a cura di Lorenzo Vandoni

Real World XML Web Services

77

Java ME World a cura di Stefano Sanna

Device, ancora device...

97

Dev. Developing Software Solutions

Numero 135
Dicembre 2005
Anno XIII

Direttore Responsabile
Marialetizia Mari (mmari@infomedia.it)

Coordinamento Editoriale
Roberto Casarini (rcasarini@infomedia.it)

Collaboratori

Fabrizio Angius, Umberto Annunziata
Massimo Cenci, Remigio Darino
Paolo De Nicolis, Demetrio Filocamo
Nunzio Fiore, Stefano Franco
Massimo Giambona, Matteo Lucarelli
Luigi Morelli, Stefano Pace
Mauro Pesci, Gian Maria Ricci
Fabio Rizzati, Alessio Saltarin
Stefano Sanna, Fabio Scagliola
Gianluigi Spagnuolo, Lorenzo Vandoni
Vito Vessia, Cesare Zavalloni

Grafica e Impaginazione
Manola Greco (mgreco@infomedia.it)

**GRUPPO EDITORIALE
INFOMEDIA**

Gruppo Editoriale Infomedia srl
Via Valdera P. 116 56038 Ponsacco (PI)
Tel. 0587 73 64 60 (r.a.)
Fax 0587 73 22 32
www.infomedia.it - red_dev@infomedia.it

Direzione

Natale Fino (nfino@infomedia.it)

Divisione Libri

Lisa Vanni (lvanni@infomedia.it)

Marketing & Advertising

Tel. 0587 73 64 60
marketing@infomedia.it

Supporto Sistemistico

webmaster@infomedia.it

Amministrazione

Sara Mattioli (amministrazione@infomedia.it)

Segreteria

Enrica Nasti (enasti@infomedia.it)

Stampa

Betagraf Spa - Puno di Argelato (BO)

Distribuzione

Farrini & C. (Roma)

Ufficio Abbonamenti

Tel. 0587 73 64 60
Fax 0587 73 22 32
abbonamenti@infomedia.it
http://online.infomedia.it

Si prega di inviare i comunicati stampa e gli invii stampa per la redazione all'indirizzo: comunicati.stampa@infomedia.it

La riproduzione con qualsiasi mezzo di illustrazioni e di articoli pubblicati sulla rivista, nonché la loro traduzione, è vietata e non può avvenire senza esplicita autorizzazione dell'editore. I manoscritti e le illustrazioni inviati alla redazione non saranno restituiti anche se non pubblicati e l'editore non si assume alcuna responsabilità per il caso che si tratti di esemplari unici.

"DEV. Developing Software & Solutions" è una rivista del Gruppo Editoriale Infomedia S.r.l.

Direzione e Amministrazione
Via Valdera P. 116 Ponsacco
Registrazione presso il Tribunale di Pisa
n. 21 del 25.11.1999

www.infomedia.it - The Software Side Of Your Mind

J2ME e la grafica

Quando si realizza un programma e, in special modo, un videogioco gran parte dell'attenzione va, ovviamente, all'aspetto grafico. Per questo introdurremo i concetti fondamentali per utilizzare gli strumenti di disegno offerti da MIDP2

Dopo aver costituito, nella scorsa puntata, un'architettura "standard" per il nucleo funzionale delle nostre future applicazioni focalizzeremo l'attenzione su un oggetto in particolare. Il programma riportato nel precedente articolo introduce i primi concetti di disegno sul display tramite un breve testo di saluto e diversi quadrilateri colorati. Si tratta, quindi, di utilizzare una delle caratteristiche fondamentali possedute da tutti i dispositivi che si prestano ad essere impiegati nelle applicazioni videoludiche: ovvero la capacità di questi di mostrare sul proprio display scritte, immagini, forme e colori.

La classe Graphics

I dispositivi che supportano il profilo MIDP2 mettono a disposizione dei programmatori la classe `Graphics`. Questa dispone di metodi per il disegno di semplici primitive 2D come testi, immagini, rettangoli, linee, archi e triangoli.

Sommario dei metodi

static Image	createImage(byte[] imageData, int imageOffset, int imageLength)
static Image	createImage(Image source)
static Image	createImage(Image image, int x, int y, int width, int height, int transform)
static Image	createImage(InputStream stream)
static Image	createImage(int width, int height)
static Image	createImage(String name)
static Image	createRGBImage(int[] rgb, int width, int height, boolean processAlpha)
Graphics	getGraphics()
int	getHeight()
void	getRGB(int[] rgbData, int offset, int scanLength, int x, int y, int width, int height)
int	getWidth()
boolean	isWritable()

FIGURA 1

Sommario dei metodi della classe `Image`

Le figure chiuse come i rettangoli, i triangoli e gli archi, possono essere riempite da un colore qualunque.

A proposito di colori, essi, vengono gestiti a 24bit con terne da 8bit per rappresentare ognuna delle tre componenti RGB. Lavorando con le immagini si può anche fare uso di un canale aggiuntivo chiamato *Alpha channel* (canale di trasparenza) tramite il quale è possibile creare, appunto, effetti di sovrapposizione in trasparenza.

Nel precedente articolo si è accennato al problema della diversità di caratteristiche possedute dai display di dispositivi diversi.

MIDP 2 fornisce una serie di strumenti per realizzare facilmente disegni tramite primitive 2D

Per questo è stato fatto in modo che il programma richiedesse al cellulare, palmare o altro dispositivo sul quale viene eseguito, i dati relativi alle caratteristiche del proprio display in termini di profondità di colore e livelli di trasparenza. Altre importanti caratteristiche di questa classe sono costituite dalla possibilità di creare aree di clipping e traslare l'origine del sistema di riferimento.

Ciò permette di modificare le dimensioni e la posizione dell'area entro la quale l'output viene visualizzato. Questa caratteristica si rende utile, ad esempio, quando le dimensioni del display di un dispositivo sono diverse da quelle del dispositivo utilizzato per lo sviluppo. In questa situazione, per evitare sgradevoli anomalie nel funzionamento del programma sarà opportuno ridimensionare e centrare l'area di output rispetto allo schermo.

È anche possibile ottenere il riferimento all'oggetto `Graphics` di un'immagine per operazioni di lettura o modifica delle stesse.

Disegnare le primitive

Il primo, e più semplice, argomento da trattare in questa sede è proprio l'utilizzo delle primitive per il disegno. A tale scopo è necessario scaricare dal sito FTP il file associato a questo articolo. All'interno della cartella *ApplicazioneBase* potrete trovare il codice del template.

Dopo aver creato un nuovo progetto con il Wireless Toolkit copiate i file *Template.java* e *TemplateCanvas.java* nella cartella src creata dal tool.

Nella cartella, invece, *Documentazione classi MIDP* troverete, appunto, la documentazione delle suddette classi, cenni sull'utilizzo e la definizione dei metodi. Realizziamo, quindi, un programma che mostra una stringa di saluto in scorrimento sul display.

Il **Listato 1** riporta il codice da inserire all'interno del metodo *draw()* nel file *TemplateCanvas.java*. Esaminiamo le istruzioni

```
g.setColor(0, 0, 0);
g.fillRect(0, 0, LARGHEZZA_DISPLAY,
    ALTEZZA_DISPLAY);
```

Con la chiamata al metodo *setColor* abbiamo impostato il colore che sarà utilizzato per le operazioni di disegno successive. E' da notare che esistono diverse varianti del metodo, ma abbiamo scelto quella più intuitiva che permette, cioè, di specificare un valore per ognuna delle tre componenti RGB. Tali valori possono variare, come consuetudine, tra 0 e 255. Con una terna di zeri indichiamo al sistema di utilizzare il colore nero.

La seconda istruzione serve per disegnare un rettangolo riempito dal colore appena specificato. Dalla definizione del



FIGURA 2 Il programma dimostrativo senza l'impostazione dell'area di taglio

argomenti nel secondo e terzo parametro. Cambiamo nuovamente il colore di disegno per ottenere un viola e disegniamo un rettangolo intorno alla stringa centrandola correttamente all'interno di esso. Per ottenere questo ultimo risultato è stato necessario utilizzare l'oggetto *Font* che restituisce l'altezza della stringa disegnata a video secondo il carattere attualmente utilizzato. Provate ora a compilare ed eseguire il programma per vedere i risultati ottenuti.

metodo *fillRect* si può facilmente interpretare il significato dei parametri ad esso passati. In pratica vogliamo riempire, ad ogni fotogramma, l'intero schermo cancellandone, così, il contenuto. In questo modo avremo un display pulito e pronto.

Disegniamo, quindi, la classica scritta "Hello World!" in posizioni progressivamente spostate verso l'alto con le seguenti istruzioni

```
g.setColor(0, 0, 255);
g.drawString("saluto", x, y, 0);

g.setColor(255, 0, 255);
g.drawRect(0, y - f.getHeight() / 2, LARGHEZZA_DISPLAY - 1, f.getHeight() * 2);
```

È banale indovinare quale cambiamento si rifletterà nel colore di disegno. La seconda chiamata provvede a chiedere che sia visualizzata la stringa indicata alle coordinate x e y passate come argomenti nel secondo e terzo parametro. Cambiamo nuovamente il colore di disegno per ottenere un viola e disegniamo un rettangolo intorno alla stringa centrandola correttamente all'interno di esso. Per ottenere questo ultimo risultato è stato necessario utilizzare l'oggetto *Font* che restituisce l'altezza della stringa disegnata a video secondo il carattere attualmente utilizzato. Provate ora a compilare ed eseguire il programma per vedere i risultati ottenuti.

Gestire le immagini

Andando oltre il semplice disegno di forme semplici, MIDP2 offre la possibilità di caricare, modificare e visualizzare immagini. Introduciamo la classe *Image* la quale espone i metodi per generare oggetti immagine decodificando i dati presenti su file all'interno del pacchetto JAR, solitamente in formato PNG, oppure creandole di nuove secondo particolari criteri. Come è possibile osservare, infatti dal sommario in Figura 1, si può ritagliare una porzione di un'immagine preesistente, utilizzare un buffer contenente una serie di interi oppure crearne



FIGURA 3 Questo programma dimostrativo riunisce tutti i concetti esposti realizzando una sorta di cartolina natalizia con tanto di neve e luci che si rincorrono

LISTATO 1

Codice per implementare la classica scritta "Hello World" in 2D

```
int x = 0, y = 0, direzione = 0;

private void draw()
{
    String saluto = new String("Hello World!");

    Font f = Font.getFont(0);

    g.setColor(0, 0, 0);
    g.fillRect(0, 0, LARGHEZZA_DISPLAY, ALTEZZA_DISPLAY);

    x = LARGHEZZA_DISPLAY / 2 - f.stringWidth(saluto) / 2;

    g.setColor(0, 0, 255);
    g.drawString(saluto, x, y, 0);

    g.setColor(255, 0, 255);
    g.drawRect(0, y - f.getHeight() / 2, LARGHEZZA_DISPLAY - 1,
        f.getHeight() * 2);

    g.setColor(255, 0, 0);
    g.drawString("Hello World!", x, ALTEZZA_DISPLAY - y, 0);

    g.setColor(0, 255, 255);
    g.drawRect(0, ALTEZZA_DISPLAY - y - f.getHeight() / 2, LARGHEZZA_DISPLAY - 1, f.getHeight() * 2);

    if (direzione == 1)
    {
        y++;
        if (y + f.getHeight() / 2 >= ALTEZZA_DISPLAY) direzione = 0;
    }
    else
    {
        y--;
        if (y - f.getHeight() / 2 <= 0) direzione = 1;
    }
}
```

un'area completamente vuota e modificabile. Anche se può sorprendere questa ultima osservazione è da tener presente che MIDP prevede due metodi di accesso alla memoria utilizzata per mantenere i dati delle immagini. Bisogna distinguere, infatti, tra immagini *mutabili* e *immutabili*.

È possibile modificare solo quelle *mutabili* ossia create da zero tramite il metodo

```
static Image createImage(int width, int height)
```

mentre tutti gli altri tipi non sono modificabili. Per il momento ci interessa imparare a caricare le immagini da file PNG e per questo creeremo un nuovo progetto, come fatto in precedenza.

Questa volta, nel pacchetto presente sul sito FTP, è stata aggiunta una sottocartella chiamata *res* dove, appunto, sono memorizzate le risorse del progetto, ossia, le immagini da caricare.

Di seguito la nuova funzione *createImmagini* aggiunta alla classe *ImmaginiCanvas* proposta al caricamento

```
private Image sfondo = null;
private Image sfera = null;

private void caricaImmagini()
{
    try
    {
        sfondo = Image.createImage("/res/sfondo.png");
        sfera = Image.createImage("/res/sfera.png");
    }
    catch (Exception e)
    {
        // Errore di caricamento delle immagini
    }
}
```

Esistono due tipologie di immagini, quelle immutabili e le mutabili. La differenza tra le due è che le prime non sono modificabili in run-time

Come potete osservare i percorsi indicati sono relativi alla radice "/" che coincide al percorso virtuale all'interno del pacchetto JAR.

A questo punto il gioco è fatto. Per visualizzare sullo schermo le immagini appena caricate basta utilizzare il metodo appropriato della classe *Graphics*

```
void drawImage(Image img, int x, int y, int anchor)
```

Il metodo *draw()* potrà essere implementato in questa maniera

```
int x = LARGHEZZA_DISPLAY / 2, y = ALTEZZA_DISPLAY / 2, direzioneX = 1,
        direzioneY = 1;

private void draw()
{
    g.drawImage(sfondo, 0, 0, 0);
    g.drawImage(sfera, x, y, 0);

    if (direzioneX == 1)
    {
        if ((x + direzioneX) + sfera.getWidth() >= LARGHEZZA_DISPLAY) direzioneX = -1;
    }
    else if (direzioneX == -1)
    {
        if ((x + direzioneX) <= 0) direzioneX = 1;
    }

    if (direzioneY == 1)
    {

```

```

if ((y+=direzioneY) + sfera.getHeight() >= ALTEZZA_DISPLAY) direzioneY
    = -1;
}
else if(direzioneY == -1)
{
    if ((y+=direzioneY) <= 0) direzioneY = 1;
}
}
}

```

ottenendo un'animazione semplice, ma di sicuro effetto, visibile in **Figura 2**.

Clipping e traslazione dell'output

Vogliamo ora correggere il difetto di visualizzazione che presenta l'esempio fino a qui prodotto. Come detto nell'introduzione MIDP dà la possibilità di restringere l'area di output e riposizionarla ovunque all'interno del display.

Vorremmo, infatti, fare in modo che la pallina rimanga all'interno dell'area occupata dall'immagine di sfondo. Tale immagine è composta da 176 x 220 pixel, contro i 240 x 320 del display dell'emulatore.

È un'operazione molto semplice, basta aggiungere le seguenti linee nel metodo `start()` o nel costruttore della classe

Le immagini possono essere caricate da file in formato PNG

```

int px = (LARGHEZZA_DISPLAY - 176) / 2;
int py = (ALTEZZA_DISPLAY - 220) / 2;

```

```

g.setClip(px, py, 176, 220);
g.translate(px, py);

```

La chiamata alla funzione `setClip()` indica proprio l'area di ritaglio dell'output. Le variabili `px` e `py` contengono il valore della posizione dell'angolo superiore sinistro di tale area, mentre 176 e 220 sono proprio le dimensioni dell'immagine di sfondo.

La funzione `translate()` fa coincidere l'origine del sistema di riferimento con l'angolo dell'area di ritaglio per rendere trasparente all'applicazione questo cambiamento.

Ovviamente dovranno essere apportate alcune modifiche al codice che muove la pallina. In particolare dovremo cambiare le righe

```

if ((x+=direzioneX) + sfera.getWidth() >= LARGHEZZA_DISPLAY) direzioneX = -1;
...
if ((y+=direzioneY) + sfera.getHeight() >= ALTEZZA_DISPLAY) direzioneY = -1;
...
in
...
if ((x+=direzioneX) + sfera.getWidth() >= g.getClipX() + g.getClipWidth())
    direzioneX = -1;

```

```

...
if ((y+=direzioneY) + sfera.getHeight() >= g.getClipY() + g.getClipHeight())
    direzioneY = -1;

```

proprio perché, ora, possiamo generalizzare la situazione determinando la posizione e la dimensione dell'area di ritaglio come confini.

L'area di clipping non è altro che una riduzione dell'area visibile del display

Conclusioni

Ci stiamo addentrando sempre più nell'esplorazione del mondo dei cellulari. Il bello deve ancora arrivare, in special modo, quando affronteremo gli argomenti più avanzati. Come avete potuto vedere l'utilizzo delle classi del profilo MIDP2 è tutt'altro che complicato. Basterà, infatti, fare riferimento alla documentazione delle classi per realizzare effetti molto sofisticati.

Ho inserito un ulteriore programma dimostrativo da studiare la cui immagine è quella di **Figura 3**.

Per il momento è tutto e Vi auguro di trascorrere un felicissimo Natale!

Download del progetto

Scarica il progetto completo di questo articolo all'indirizzo:
<ftp://ftp.infomedia.it/pub/DEV/Listati/Dev135/gamesdev.zip>

Bibliografia

- [1] Carol Hamer, "J2ME Games with MIDP2", Apress, 2004, ISBN 1-59059-382-0, www.apress.com
- [2] John W. Muchow, "J2ME Guida pratica alla programmazione di dispositivi wireless", Mc Graw Hill, 2002, ISBN 88-386-4278-8

Mauro Pesci

Si occupa di Analisi e Sviluppo del software in vari ambienti e piattaforme. Ha lavorato per diverse aziende nel settore, mentre ora è impegnato in un progetto presso la HP di Roma. Attualmente studia Informatica presso l'università di Tor Vergata (Roma).