Strumenti per lo sviluppo del software

Lo sviluppo del software

Lo sviluppo del software è l'attività centrale del progetto e ha lo scopo di produrre il codice sorgente che, una volta *compilato* e messo in esecuzione, produrrà gli effetti desiderati e richiesti dal committente.
L'attività di sviluppo è un'attività complessa e richiede competenze diverse riguardanti la capacità di risolvere problemi, l'utilizzo di ambienti di programmazione, oltre alla creatività personale, per realizzare software utili, funzionanti e facili da utilizzare.

La realizzazione del software si fonda su alcuni principi di base:

- Per lo sviluppo di software si utilizzano in genere i *linguaggi di programmazione*; quando lo sviluppo di un prodotto avviene senza l'ausilio della programmazione, ma tramite l'utilizzo di software di produzione rapida di applicazioni si parla di processi **RAD** (*Rapid Application Development*).
- I linguaggi di programmazione consentono la traduzione di *algoritmi*, che nel loro complesso formano il *codice sorgente* del programma.
- Per funzionare il codice sorgente deve essere *compilato*, cioè tradotto in una forma che sia comprensibile al sistema di elaborazione che deve eseguire il programma. Il programma compilato viene anche chiamato *binario*.
- Il *compilatore* è un programma eseguibile che è in grado di comprendere il *codice sorgente* di un programma, verificarne la correttezza sintattica, applicare regole di ottimizzazione e produrre il *binario* eseguibile.
- L'attività di *compilazione* del *codice sorgente* in *binario* è un'attività eseguita da un computer e non necessita quindi di conoscenza: l'intera conoscenza, che il software nel suo complesso rappresenta, è rappresentata dal *codice sorgente*.

La scrittura del codice sorgente e la relativa compilazione normalmente avviene con un **IDE** (*Integrated Development Environment*), ma si possono utilizzare anche strumenti di base come un editor di testo e un compilatore attivato dalla linea di comando.

Un IDE offre al programmatore alcuni importanti vantaggi:

- Ambiente grafico integrato, formato da vari pannelli che aiutano il programmatore ad avere sotto controllo i molteplici aspetti del progetto su cui sta lavorando.
- Autocompletamento del codice durante la digitazione e aiuto contestuale. Questa funzione riduce notevolmente la presenza nel codice di errori di battitura.
- Verifica sintattica automatica durante la scrittura del codice, con suggerimenti per la risoluzione degli errori sintattici più comuni.
- *Debugger* visuale: un *debugger* è un programma utilizzato dal programmatore per eseguire il codice in modo controllato, con la possibilità di introdurre *punti in interruzione* in cui arrestare l'esecuzione del programma e ispezionare i valori delle variabili in uso.
- Progettazione visuale degli elementi dell'interfaccia grafica.
- Accesso rapido e ricerca degli oggetti su cui si sta lavorando.
- Possibilità di generare file per l'installazione del programma binario su vari ambienti di esecuzione.
- *Refactoring* automatico del codice sorgente: è l'attività con cui il codice sorgente viene adeguato, nella struttura e nella leggibilità, per rispondere al graduale e continuo mutamento dei requisiti del progetto.

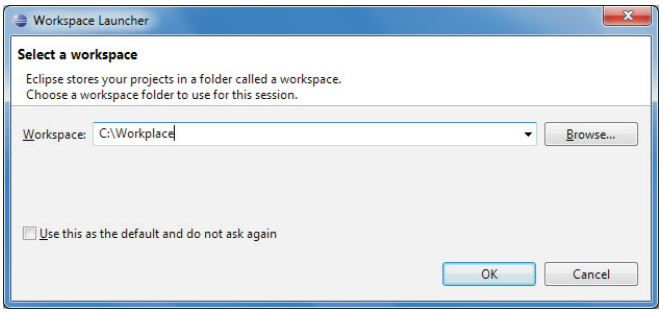
Eclipse

Eclipse è un ambiente di sviluppo integrato (**IDE**, *Integrated Development Environment*) per la programmazione in linguaggio Java. È stato realizzato e viene continuamente aggiornato da un'organizzazione non-profit, *Eclipse Foundation* (*www.eclipse.org*), di cui fanno parte molte importanti società nel mondo dell'informatica. Questo IDE è distribuito con una licenza *open source* che ne permette il libero utilizzo.

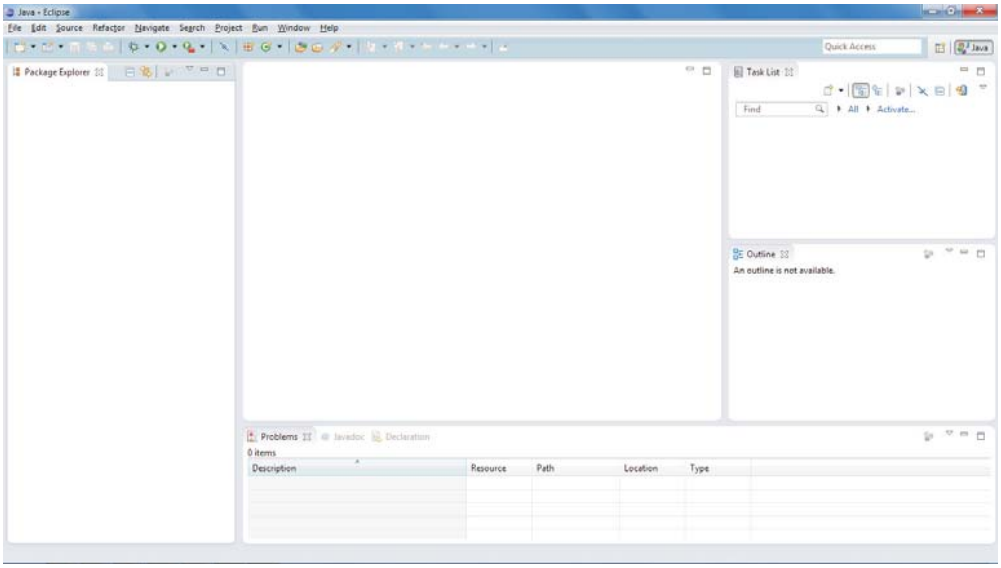
Eclipse è una piattaforma su cui si possono inserire diversi **plug-in**, cioè dei componenti software realizzati per scopi specifici e che permettono di estendere le funzionalità di base dell'ambiente di sviluppo. Per esempio, ci sono plug-in per la costruzione guidata delle interfacce grafiche, per la programmazione in diversi linguaggi (PHP, Python, C), per la realizzazione di diagrammi di documentazione, per la diagnostica degli errori (*bug*) e per la gestione delle versioni del codice (*versioning*).

Nella pagina di **download** di Eclipse è possibile scegliere tra vari pacchetti di installazione che si differenziano per il numero di *plug-in* che sono già inclusi.

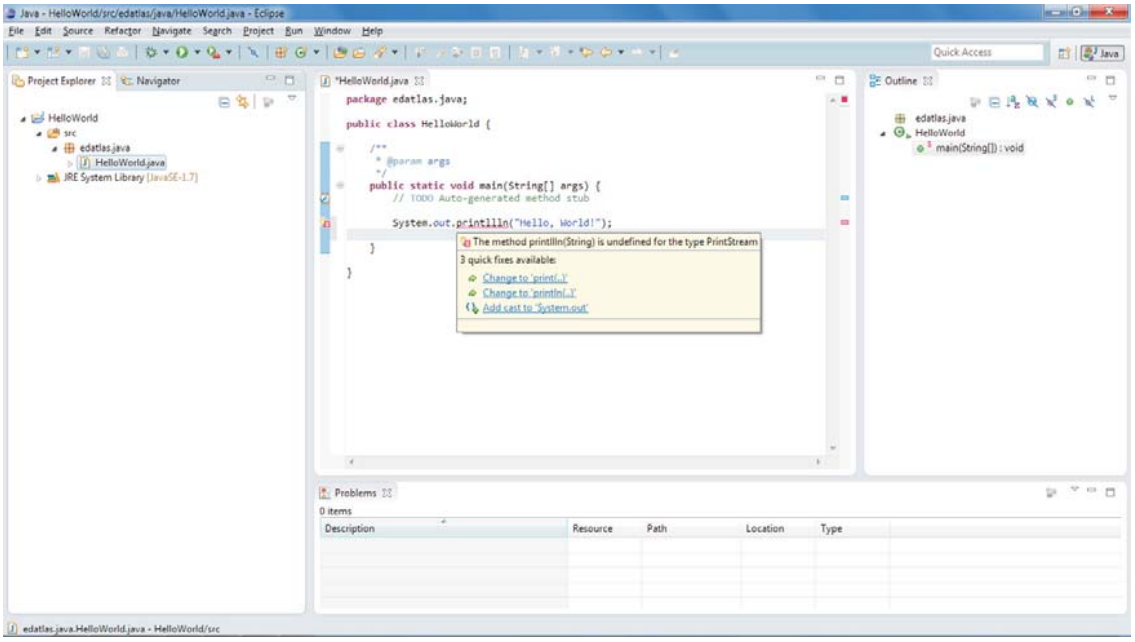
All'avvio del programma viene richiesto al programmatore di indicare una cartella (**Workplace**) per la sessione di lavoro appena iniziata. Il *Workplace* corrisponde alla cartella di lavoro in cui Eclipse memorizza i progetti. Attivando l'apposita casella di spunta, si può impostare il *Workplace* predefinito in modo che le successive aperture di Eclipse facciano riferimento a questa cartella di lavoro predefinita.



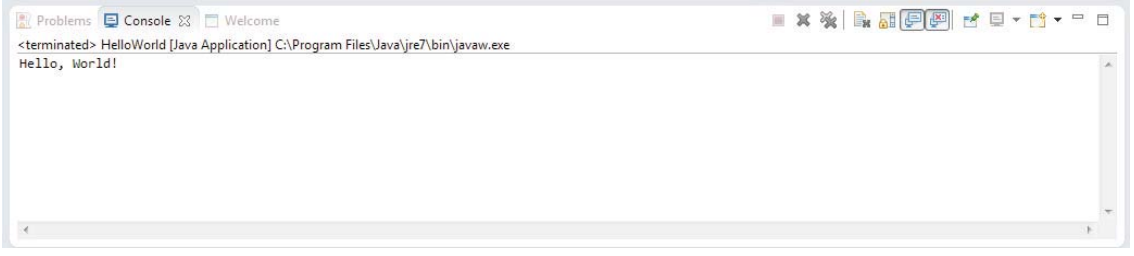
La finestra dell'IDE è composta, in alto, dal *barra dei menu* e dalla *barra degli strumenti* mentre la parte centrale, chiamata anche **Workbench** (letteralmente, tavolo di lavoro), si presenta con un insieme di riquadri configurabili. Solitamente il riquadro a sinistra visualizza la lista dei progetti e consente la navigazione nelle cartelle del disco, il riquadro centrale è l'*editor*, il riquadro inferiore è usato per l'output e i messaggi di errore, mentre a destra altri riquadri contengono i dettagli della struttura delle classi, degli attributi e dei metodi.



Per creare un nuovo progetto si deve fare clic sul menu **File** e scegliere la voce **New**.
Nel riquadro centrale viene visualizzato il codice sorgente mentre nel riquadro a sinistra (*Project Explorer*) viene visualizzata la struttura del progetto. Per aprire e modificare un file presente nel progetto è sufficiente fare doppio clic sul relativo nome nel riquadro *Project Explorer*. Il file viene visualizzato nell'editor di testi.



Si noti che il testo del codice viene colorato in modo automatico: per esempio in fucsia le parole chiave, in nero i nomi delle classi e dei metodi, in blu le stringhe e in verde i commenti. Inoltre eventuali errori sono segnalati con la linea rossa ondulata: passando con il mouse sopra la parola sottolineata, viene visualizzato il messaggio di errore con un suggerimento per la correzione.
Il lavoro viene salvato facendo clic sul menu **File** e poi, su **Save** oppure con un clic sulla relativa icona nella barra degli strumenti (la scorciatoia da tastiera è la combinazione di tasti **Ctrl + S**).
Il programma può essere compilato ed eseguito facendo clic sul menu **Run** e poi sulla voce **Run**, oppure usando la combinazione di tasti **Ctrl + F11**. I messaggi generati dall'esecuzione vengono visualizzati nel riquadro **Console** nella parte inferiore del video.



Eclipse è impostato per compilare il programma ogni volta che viene eseguito il salvataggio. I messaggi generati del compilatore si possono distinguere in due categorie: gli **avvertimenti** (*warning*) e gli **errori** (*error*).
Gli *avvertimenti* sono evidenziati con etichette gialle poste ai lati della finestra di editing. La loro presenza non pregiudica la compilazione e nemmeno l'esecuzione, ma segnala i punti in cui il programma può essere migliorato e reso più efficiente.

Per esempio, una variabile non utilizzata produce il seguente avvertimento:

Gli *errori sintattici* invece non permettono la compilazione e l'esecuzione del programma. Vengono evidenziati con etichette rosse poste ai lati della finestra di editing. Per esempio, la mancanza del punto e virgola al termine di una riga produce il seguente errore:

Gli errori di programmazione che non vengono evidenziati dal compilatore possono essere analizzati utilizzando la modalità di esecuzione in *debug*.
Le attività più comuni che si possono attivare per il **debugging** dell'applicazione sono:
- impostare punti di interruzione (**breakpoints**) che fermano temporaneamente l'esecuzione in corrispondenza di linee di codice prefissate;
- osservare il valore assunto dalle variabili;
- eseguire il programma passo passo;
- impostare la prossima istruzione da eseguire.

Per fissare punti di interruzione è possibile fare doppio clic a lato della linea di codice. Un punto di interruzione attivo viene visualizzato con un cerchio blu.

Facendo doppio clic sul cerchio, il punto di interruzione viene eliminato. La scorciatoia da tastiera per attivare e disattivare i punti di interruzione è la combinazione di tasti **Ctrl + Maiuscolo + B**.

Dopo aver indicato almeno un punto di interruzione, per eseguire il *debugging* dell'applicazione si deve fare clic sul menu **Run** e poi sulla voce **Debug**, oppure si deve premere il tasto **F11**.

Per vedere il valore delle variabili e delle espressioni si deve passare con il mouse sopra la variabile, oppure aprire il riquadro **Expressions**, dal menu **Debug** e poi **Watch**, e scrivere il nome della variabile.

Per eseguire il programma passo passo:
- fare clic su **Step Into** (oppure premere il tasto **F5**)
- fare clic su **Step Over** (oppure premere il tasto **F6**).

Entrambi i comandi eseguono una sola riga di codice e poi sospendono l'esecuzione. Nel caso di *Step Into*, se la riga in esecuzione è un metodo, l'attività di debug entra nel metodo eseguendo le sue istruzioni passo passo.

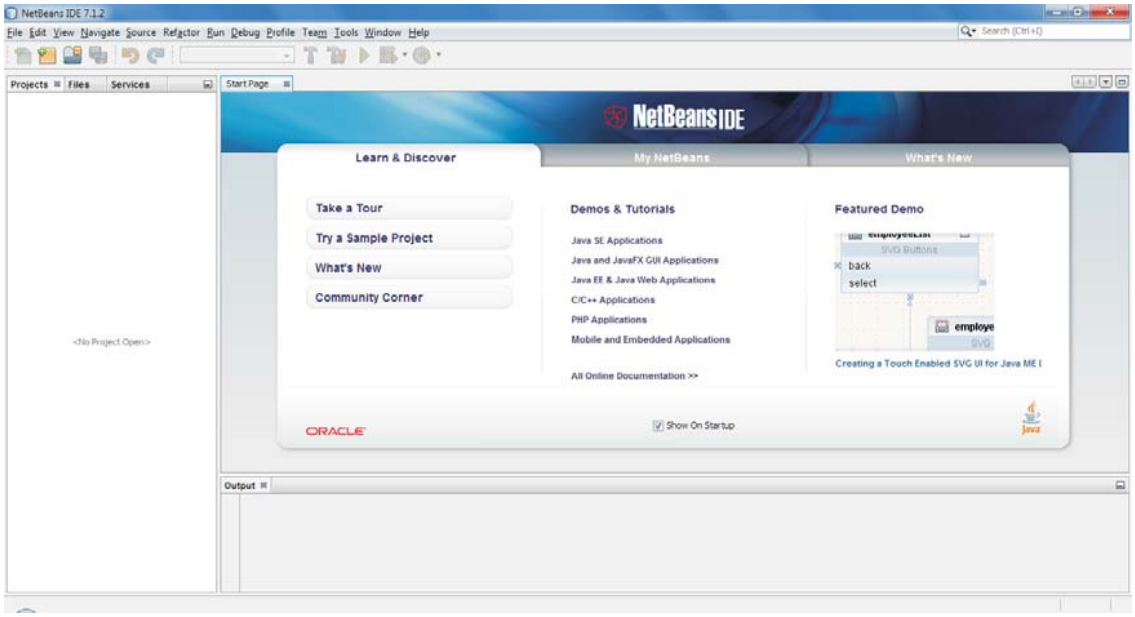
Per continuare l'esecuzione fino al prossimo punto di interruzione, fare clic su **Resume** (oppure premere il tasto **F8**).
Infine, per interrompere la modalità di debug, fare clic su **Terminate** (oppure premere la combinazione di tasti **Ctrl + F2**).

La sessione di lavoro si conclude, dopo aver salvato i file, nel momento in cui viene chiusa la finestra di Eclipse. L'ultima disposizione dei riquadri sul piano di lavoro viene memorizzata e viene presentata nuovamente al programmatore nella successiva sessione di lavoro.

NetBeans

NetBeans è un ambiente di sviluppo integrato (**IDE**, *Integrated Development Environment*) per la programmazione in linguaggio Java. L'IDE fornisce diversi strumenti per facilitare l'attività del programmatore: suggerisce la struttura di base delle classi, visualizza gli errori sintattici direttamente nella finestra dell'editor e, tramite l'autocompletamento, velocizza la scrittura del codice.

NetBeans è un software *open source* e può essere installato sui sistemi operativi *Windows*, *Linux* e *Mac OS*.



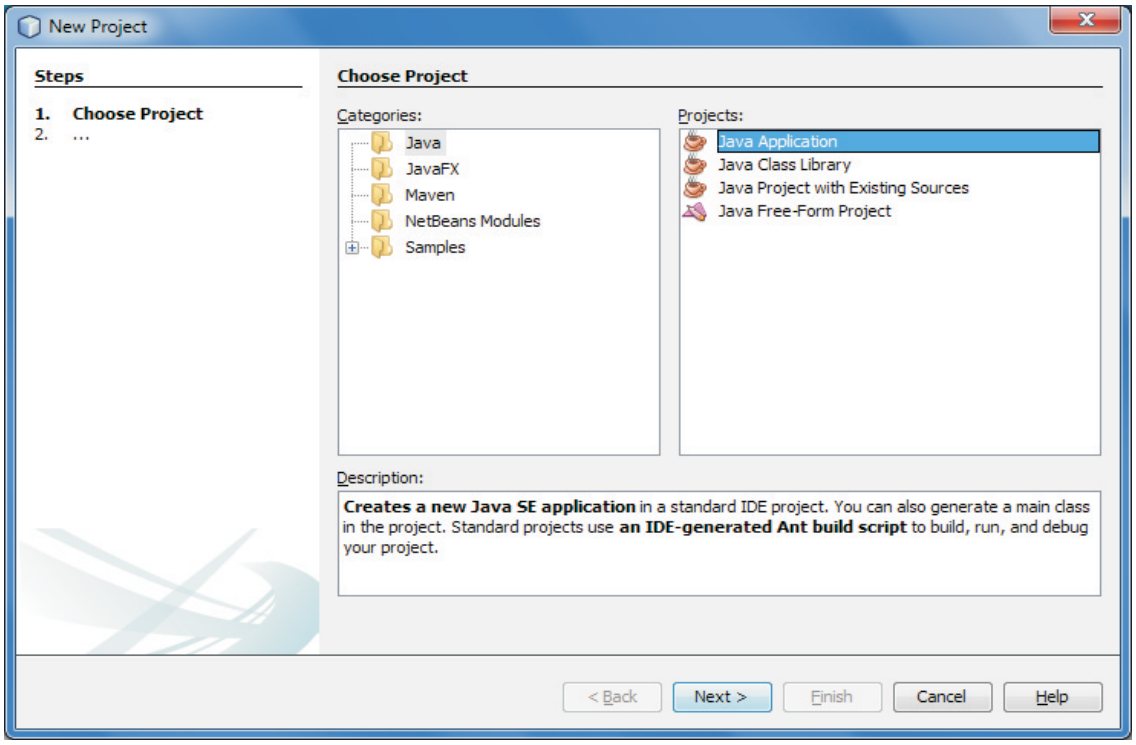
All'avvio del programma viene aperta la precedente videata, con:
- in alto, la barra dei menu e la barra degli strumenti;
- a sinistra, il riquadro per la navigazione tra i progetti e i file;
- al centro, la pagina iniziale con i collegamenti alla documentazione, ai progetti recentemente aperti e alle novità;
- in basso, il riquadro per l'output e i messaggi di errore.

L'attività di programmazione in NetBeans inizia con la creazione di un **progetto**. Ogni progetto può essere visto come un programma Java, composto da un insieme di file sorgenti e dalle impostazioni necessarie per la sua compilazione ed esecuzione.

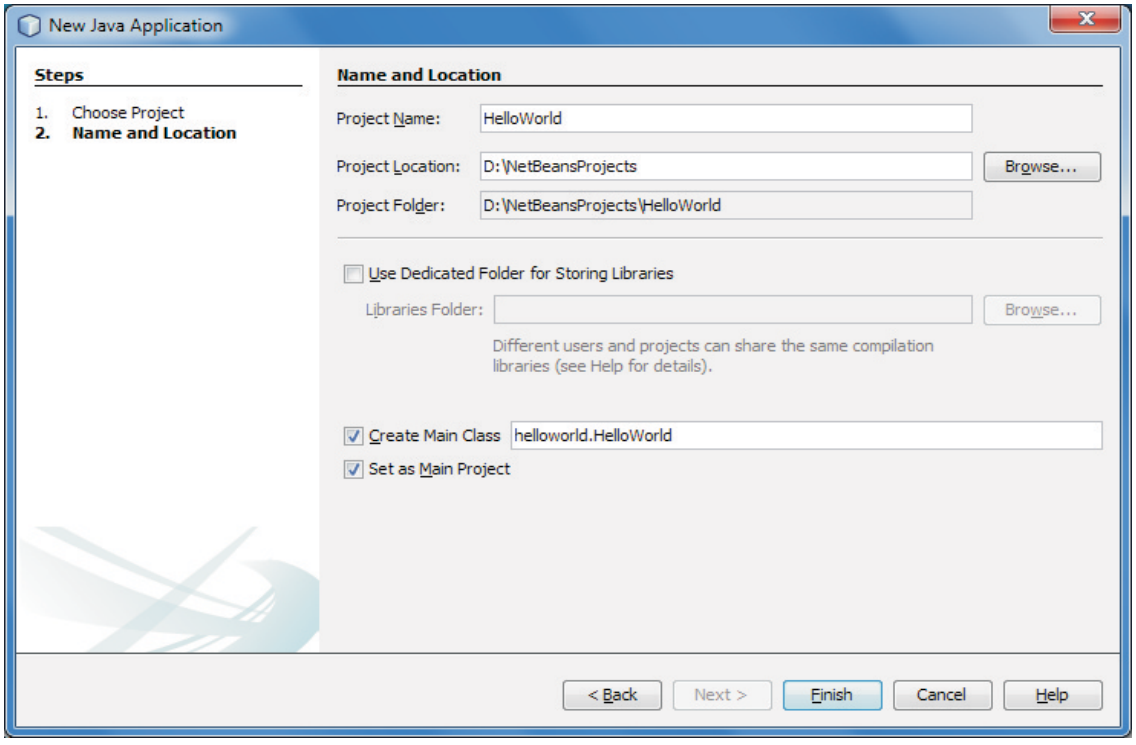
Per creare un nuovo progetto si deve fare clic sul menu **File** e poi su **New Project** (la scorciatoia da tastiera è la combinazione di tasti **Ctrl + Maiuscolo + N**). In alternativa si può fare clic sull'icona  nella barra degli strumenti.

Viene aperta una finestra in cui è possibile scegliere tra varie tipologie di progetto.

Si noti che, nella categoria **Sample**, è possibile scegliere tra un insieme di applicazioni di esempio già preconfezionate. Creare un progetto a partire da questi esempi è un modo efficace per apprendere la programmazione.

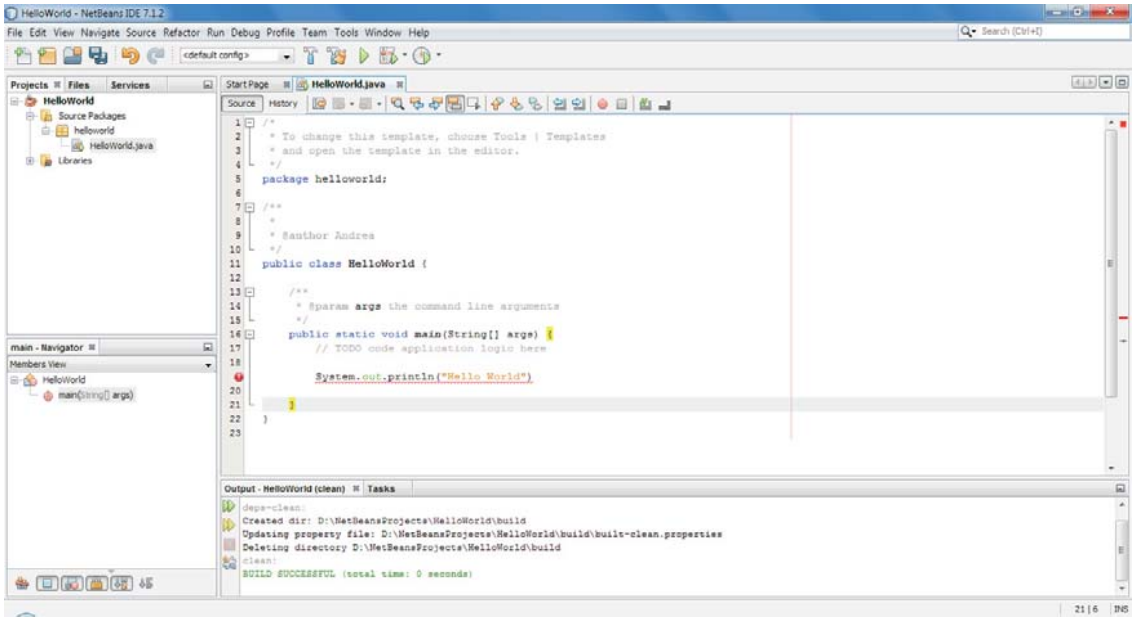


Nella successiva pagina, scriviamo il nome del progetto nella casella **Project Name** e selezioniamo il percorso dove salvare il progetto facendo clic sul pulsante **Browse**. Si noti che il nome del progetto viene in automatico assegnato anche alla classe principale. Per completare la creazione del progetto facciamo clic sul pulsante **Finish**.



Il nuovo progetto viene aperto nell'IDE e vengono visualizzati i seguenti riquadri:

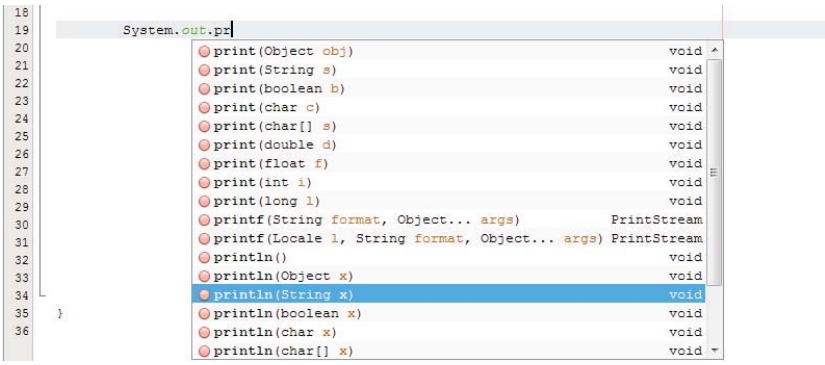
- **Projects**: contiene l'elenco dei progetti e dei file che li compongono, facendo doppio clic su un file lo si apre nell'editor;
- **Navigator**: visualizza gli attributi e i metodi della classe, facendo doppio clic su un elemento permette di visualizzarlo velocemente nell'editor senza dover usare le barre di scorrimento per muoversi lungo il codice sorgente;
- **Editor**: mostra il file sorgente colorato in modo automatico, per esempio, con le parole chiave in blu, i nomi degli elementi della classe in nero e i commenti in grigio;
- **Output**: mostra i messaggi, tra cui quelli di conferma della creazione del progetto.



Si noti che gli eventuali errori sintattici sono segnalati con una linea ondulata rossa e un simbolo rosso nella parte sinistra dell'editor.

Si noti inoltre che, posizionando il cursore su una parentesi graffa, viene evidenziato in giallo il blocco a cui la parentesi fa riferimento. In questo modo il programmatore può verificare la corretta disposizione delle parentesi e accorgersi, in modo agevole, di eventuali errori.

La scrittura del codice sorgente è facilitata dalla funzionalità di **autocompletamento** che mostra, in tempo reale, l'elenco degli elementi (oggetti, attributi e metodi) che più si avvicinano a quanto si sta digitando. Usando le frecce direzionali si può scegliere l'elemento d'interesse e lo si può inserire nel codice premendo il tasto *Invio*.



Dopo aver modificato il codice sorgente, salviamo il lavoro dal menu **File** con la scelta **Save All** oppure con un clic sull'icona della barra degli strumenti
(la scorciatoia da tastiera è la combinazione di tasti **Ctrl + Maiuscolo + S**).

Per compilare il programma facciamo clic sull'icona **Build Main Project**
(la scorciatoia da tastiera è il tasto **F11**).

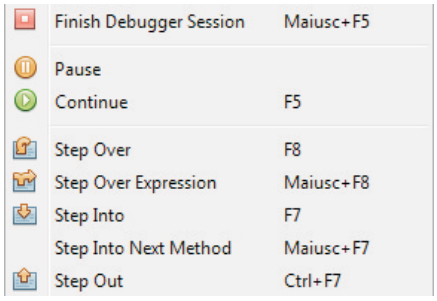
Per eseguire il programma facciamo clic sull'icona **Run Main Project**
(la scorciatoia da tastiera è il tasto **F6**).

Il risultato dell'esecuzione e i messaggi vengono mostrati nella finestra *Output*.

Per attivare il **debug** del programma facciamo clic sull'icona **Debug Main Project**
(la scorciatoia da tastiera è la combinazione di tasti **Ctrl + F5**).

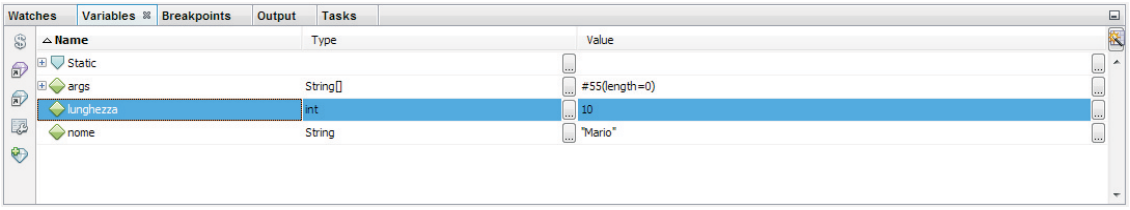
Prima di attivare il debug è necessario indicare almeno un punto di interruzione (**Breakpoint**) all'interno del codice sorgente. Il punto di interruzione deve essere posizionato sulla prima riga di codice da cui si intende iniziare l'analisi e la ricerca degli errori.
Per inserire un punto di interruzione si deve fare clic a lato della linea di codice. Un punto di interruzione attivo viene visualizzato con un quadrato rosa posto sulla sinistra e con l'intera linea evidenziata in rosa. Facendo clic sul quadrato, il punto di interruzione viene eliminato.

Attivando il debug, l'esecuzione del programma parte e si sospende nel punto in cui è stato inserito il *Breakpoint*. Per proseguire con un'esecuzione passo passo o per interrompere il programma si devono usare i comandi presenti nel menu **Debug**.



I comandi mostrati in figura, con le stesse icone, possono essere attivati dalla barra degli strumenti.

Ad ogni passo di esecuzione, è possibile controllare il valore delle variabili passando il mouse sopra il nome della variabile nella finestra dell'editor. Inoltre, in modalità *debugging*, viene aperto il riquadro **Variables**, nella parte inferiore dell'IDE, che facilita la consultazione dell'elenco completo delle variabili e permette di monitorare il valore delle espressioni.



Per chiudere il progetto aperto nell'ambiente di sviluppo, dopo aver salvato il lavoro, facciamo clic su **File** e poi su **Close Project**.