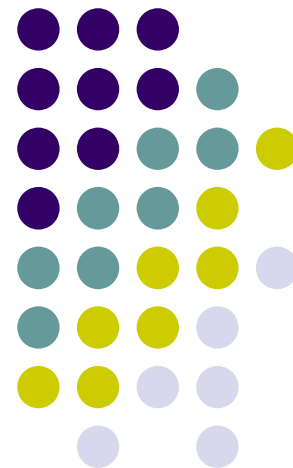
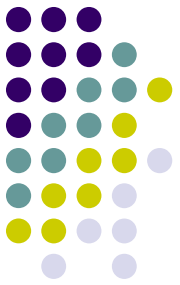


# Corso di Programmazione

*Welcome to JAVA!*



# Starting...

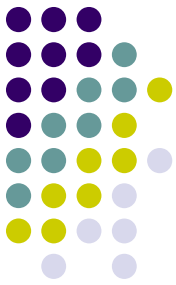


- Controllate se sul vostro PC/Notebook è già installata una **JDK**
- Da terminale (MAC) o da prompt dei comandi (Windows) dare il comando:  
**java -version**
- Se la JDK è già installata il comando produrrà un output del tipo:

*openjdk version "17.0.3" 2022-04-19*

- Altrimenti è necessario che installiate una JDK, per esempio la JDK 17 da:  
<https://www.oracle.com/java/technologies/downloads/#java17>

# JDK



[Java downloads](#) [Tools and resources](#) [Java archive](#)

## JDK Development Kit 17.0.9 downloads

JDK 17 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions \(NFTC\)](#).

JDK 17 will receive updates under the NFTC, until September 2024. Subsequent JDK 17 updates will be licensed under the [Java SE OTN License](#) (OTN) and production use beyond the [limited free grants](#) of the OTN license will require a fee.

Linux macOS **Windows**

Product/file description	File size	Download
x64 Compressed Archive	172.42 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip</a> ( sha256)
x64 Installer	153.51 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe</a> ( sha256)
x64 MSI Installer	152.30 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi</a> ( sha256)

## JDK Development Kit 17.0.9 downloads

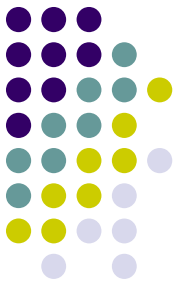
JDK 17 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions \(NFTC\)](#).

JDK 17 will receive updates under the NFTC, until September 2024. Subsequent JDK 17 updates will be licensed under the [Java SE OTN License](#) (OTN) and production use beyond the [limited free grants](#) of the OTN license will require a fee.

Linux **macOS** Windows

Product/file description	File size	Download
ARM64 Compressed Archive	168.18 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_macos-aarch64_bin.tar.gz">https://download.oracle.com/java/17/latest/jdk-17_macos-aarch64_bin.tar.gz</a> ( sha256)
ARM64 DMG Installer	167.60 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_macos-aarch64_bin.dmg">https://download.oracle.com/java/17/latest/jdk-17_macos-aarch64_bin.dmg</a> ( sha256)
x64 Compressed Archive	170.62 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_macos-x64_bin.tar.gz">https://download.oracle.com/java/17/latest/jdk-17_macos-x64_bin.tar.gz</a> ( sha256)
x64 DMG Installer	170.03 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_macos-x64_bin.dmg">https://download.oracle.com/java/17/latest/jdk-17_macos-x64_bin.dmg</a> ( sha256)

# Ambienti consigliati

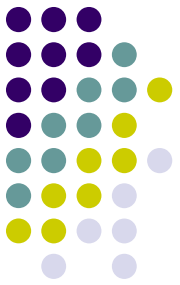


- Apache Netbeans (utilizzato dal docente per svolgere esercizi ed esempi)
- Visual Studio
- In laboratorio (ai fini dell'esame) sono presenti sia Netbeans che Visual Studio, ma preferibilmente useremo Netbeans

<https://netbeans.apache.org/front/main/download/>

*Non è detto che sia sempre opportuno installare l'ultimissima versione... prima controllate i requisiti di installazione e la JDK richiesta...*

# Applicazioni JAVA



- Un' applicazione è un programma eseguibile autonomamente

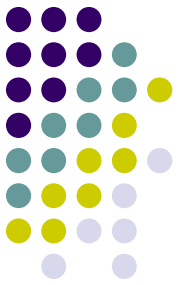
Per rendere eseguibile un'applicazione Java occorre che una classe faccia da “punto di avvio”.

Questa classe ha solo bisogno di un metodo ***main()***, il primo ad essere invocato durante l'esecuzione dell'applicazione.

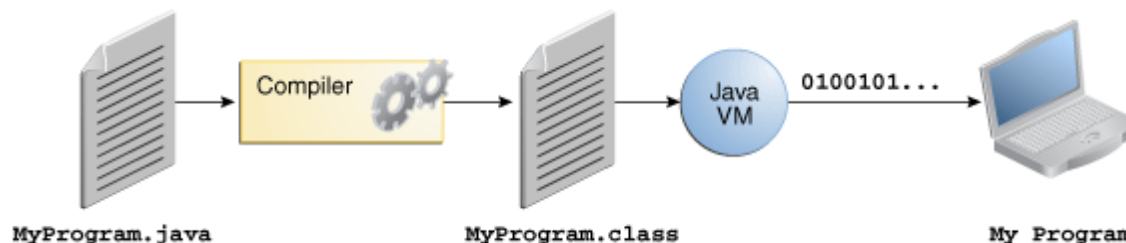
```
public static void main (String args[]) {  
    .....  
}
```

- **public** indica che il metodo è a disposizione di altre classi e oggetti
- **static** indica che *main()* è un metodo invocabile anche se non esistono oggetti (vedremo meglio poi)
- **void** indica che il metodo *main()* non restituisce un valore
- **main()** accetta come parametro un array di stringhe, utilizzato per passare argomenti dalla riga di comando

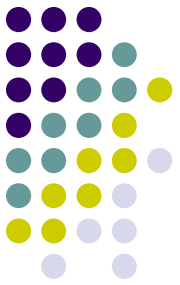
# Processo di sviluppo in JAVA



- In Java, il **codice sorgente** viene scritto in file di testo con estensione **.java**
- Viene compilato ma la compilazione NON produce un file oggetto per la macchina su cui è stata fatta la compilazione bensì un oggetto in **bytecode** in un file con estensione **.class**
- In questo modo si ottiene la portabilità, ***il bytecode generato è lo stesso per tutte le architetture***
- Per eseguire il programma è necessario che sulla macchina sia presente la ***Virtual Machine Java (Java VM)*** che è *disponibile per diversi sistemi operativi* e ha il compito di interpretare il bytecode per la macchina su cui si trova



# Primo programma Java: editing



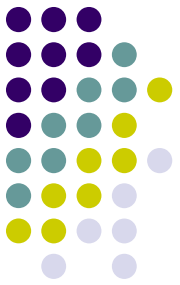
```
/* prima classe di esempio per illustrare
   l'uso del metodo main e i passi di sviluppo
*/

public class HelloWorld {

    // metodi della classe
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");
    }
}
```

File: HelloWorld.java

# Primo programma Java: editing



```
/* prima classe di esempio per illustrare  
   l'uso del metodo main e i passi di sviluppo  
*/
```

Commento su più linee

```
public class HelloWorld {
```

```
// metodi della classe
```

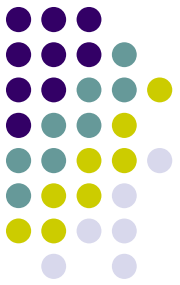
Commento su linea singola

```
public static void main(String[] args) {  
    System.out.println("Welcome to Java!");  
}  
}
```

File: HelloWorld.java



# Primo programma Java: editing



```
/* prima classe di esempio per illustrare  
l'uso del metodo main e i passi di sviluppo  
*/
```

```
public class HelloWorld {
```

```
// metodi della classe
```

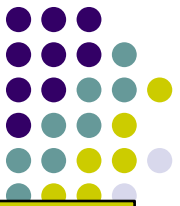
```
public static void main(String[] args) {  
    System.out.println("Welcome to Java!");  
}  
}
```

File: HelloWorld.java

## Dichiarazione di classe.

- Ogni programma Java contiene almeno una dichiarazione definita dal programmatore.
- La **parola chiave *class*** introduce una dichiarazione di classe ed è immediatamente seguita dal nome della classe.
- All'inizio ogni classe che dichiareremo inizierà con la parola chiave **public** che è un **modificatore di accesso**.
- Approfondiremo in seguito il concetto dei modificatori di accesso

# Primo programma Java: editing



```
/* prima classe di esempio per illustrare  
l'uso del metodo main e i passi di sviluppo  
*/
```

```
public class HelloWorld {
```

```
// metodi della classe
```

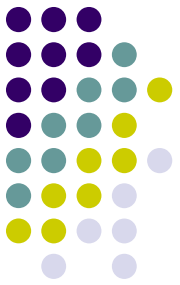
```
public static void main(String[] args) {  
    System.out.println("Welcome to Java!");  
}  
}
```

File: HelloWorld.java

## Per convenzione.

- I nomi di classe in Java iniziano con un carattere maiuscolo e mettono in maiuscolo ogni primo carattere di ogni eventuale parola successiva.
  - Es. MiaClasse
  - Notazione detta CamelCase
- Un nome di una classe è un identificativo composto da una serie di caratteri formati da lettere, numeri, trattini bassi (\_) e simbolo del dollaro (\$).
- Il nome di una classe non inizia mai con un numero e non contiene spazi.
- Java è case sensitive quindi a1 e A1 sono due identificativi validi ma distinti.

# Primo programma Java: editing



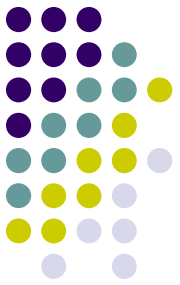
```
/* prima classe di esempio per illustrare  
l'uso del metodo main e i passi di sviluppo  
*/
```

```
public class HelloWorld {  
  
    // metodi della classe  
    public static void main(String[] args) {  
        System.out.println("Welcome to Java!");  
    }  
}
```

Corpo della classe

File: HelloWorld.java

# Primo programma Java: editing



```
/* prima classe di esempio per illustrare  
   l'uso del metodo main e i passi di sviluppo  
*/
```

```
public class HelloWorld {
```

```
    // metodi della classe
```

```
    public static void main(String[] args) {  
        System.out.println("Welcome to Java!");  
    }
```

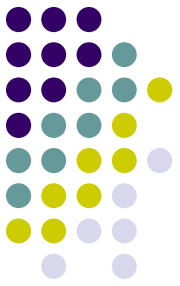
```
}
```

File: HelloWorld.java

Dichiarazione di un metodo

- Il nome di un metodo, in questo caso main è un **identificatore**
- Il **corpo del metodo** è identificato dal blocco delimitato da una parentesi graffa aperta e una parentesi graffa chiusa.
- L'elemento String[] args fra le parentesi è una parte obbligatoria della dichiarazione main.

# Primo programma Java: editing



```
/* prima classe di esempio per illustrare
   l'uso del metodo main e i passi di sviluppo
*/

public class HelloWorld {

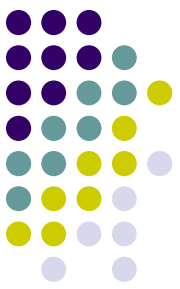
    // metodi della classe
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");
    }
}
```

File: HelloWorld.java

System.out.println();

- È il comando che consente di visualizzare a terminale o shell (in Linux) o prompt dei comandi (in Windows) i caratteri contenuti tra doppi apici.
  - La sequenza di caratteri è detta stringa, o anche stringa di caratteri o stringa letterale.
- System.out è chiamato **oggetto stream standard output**
- println è il metodo dell'oggetto System.out
  - stampa a terminale la stringa "argomento" del metodo

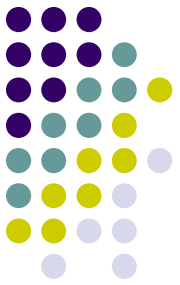
# Primo programma Java: compilazione



- Il file `HelloWorld.java` può essere compilato e, dato che contiene il `main`, contiene una applicazione java e può essere eseguito
- Il nome del sorgente **deve** avere l'estensione `.java`, altrimenti il compilatore segnalerà che gli è impossibile trovare il file.
- Per compilare si può scrivere a riga di comando

**`javac HelloWorld.java`**  
(scrivendo esplicitamente `.java`)

# Bytecode



- La compilazione produce il file oggetto **Helloworld.class** che contiene la traduzione del file sorgente in **bytecode**
- Quindi i file .class non sono file di testo e non sono visualizzabili correttamente con un editor!

<

>

Indietro/Avanti

Hello

☰

↕

Vista

☰

▼

Gruppo

📄

🔗

Condividi

🏷️

🏷️

Modifica tag

⋮

▼

Azione

Cerca

Nome

^

Data di modifica

Dimensioni

Tipo

📄

Helloworld.class

oggi, 10:46

430 byte

File classe Java

📄

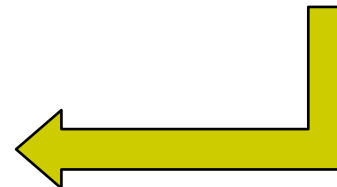
Helloworld.java

oggi, 10:46

275 byte

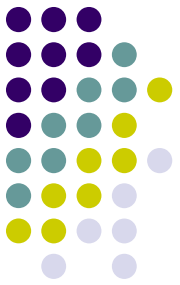
Codice...te Java

```
Hello — -zsh — 80x24
valeria@Air-di-Valeria Hello % javac Helloworld.java
valeria@Air-di-Valeria Hello %
```



# Primo programma Java:

## Caricamento ed Esecuzione



- Il bytecode di una applicazione Java può essere eseguito dalla JVM sulla macchina in cui si trova
- Il comando che invoca la JVM sul nostro primo programma è:

**java HelloWorld**

(si omette `.class`)

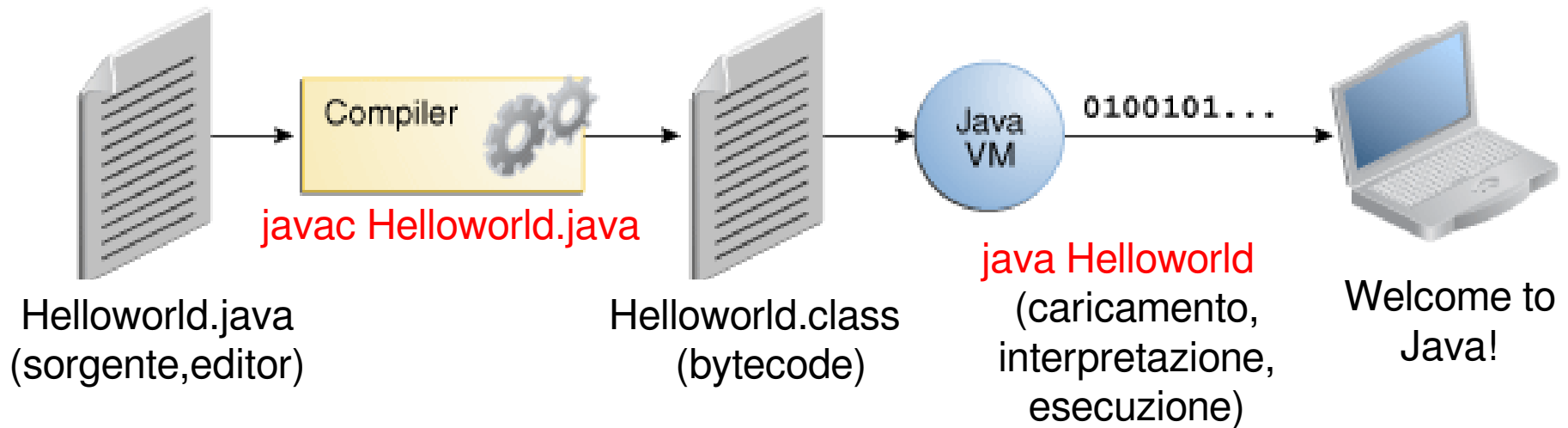
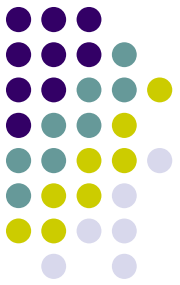
- Il programma viene caricato in memoria ed eseguito dall'interprete java
- Si noti che il nome *Helloworld* è il nome della classe che contiene il metodo main

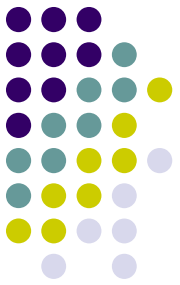
A screenshot of a terminal window titled "Hello — -zsh — 80x24". The terminal shows the command "java HelloWorld" being executed, followed by the output "Welcome to Java!". The prompt "valeria@Air-di-Valeria Hello %" is visible at the bottom.

```
valeria@Air-di-Valeria Hello % java HelloWorld
Welcome to Java!
valeria@Air-di-Valeria Hello %
```



# Ricapitolando...





# Riferimenti

- Programmare in Java – Capitolo 2
- <https://netbeans.apache.org/front/main/download/>
- <https://netbeans.apache.org/front/main/download/archive/>
  - Per il download di versioni precedenti di Netbeans
- <https://www.oracle.com/java/technologies/downloads/>
- <https://www.oracle.com/java/technologies/downloads/#java17>
  - Per il download della versione Java SE 17 LTS
- <https://docs.oracle.com/javase/tutorial/getStarted/intro/definition.html>