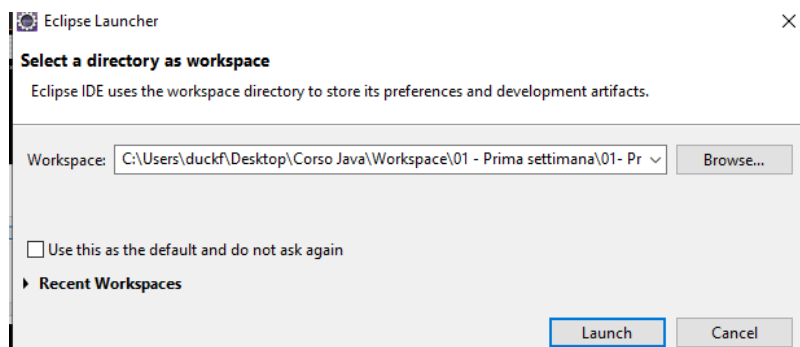


Sommario

1.0 – INTRODUZIONE A ECLIPSE	1
1.1 CREARE NUOVO PROGETTO	2
2. JAVA	5
2.1 Cose di base	5
2.1.1 Sintassi	6
2.1.2 I tipi	6
2.2.3 Le variabili	6
2.2.4 Principi della programmazione	7
2.2.5 Convertire string in numeri	7
2.2.6 Lo scanner e gli input dell'utente	7
2.2.7 Secondo principio della programmazione – la selezione (if)	9

1.0 – INTRODUZIONE A ECLIPSE

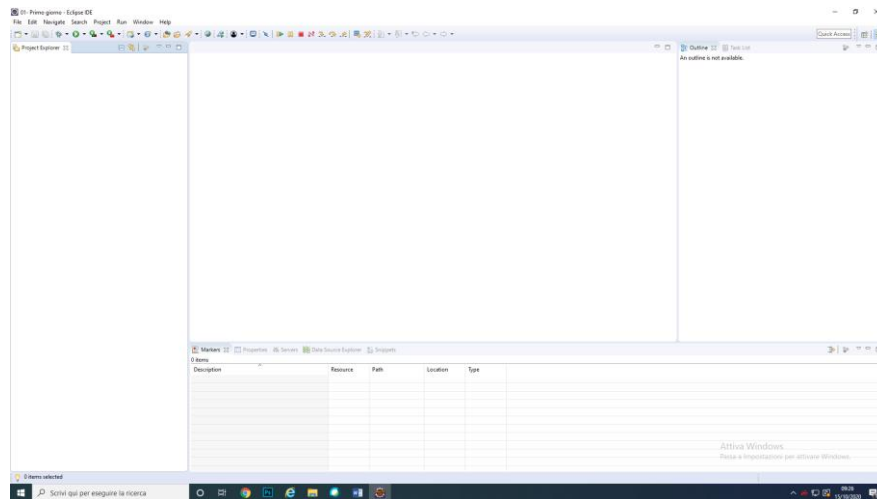
Eclipse è l'ide (integrated development environment) che useremo nel corso. Un IDE è un ambiente di sviluppo.



All'avvio, Eclipse ha bisogno di selezionare un **workspace**, ovvero lo spazio dove vengono salvati tutti i dati di ciò che poi viene eseguito.

Importante è lasciare bianca la spunta di default, in modo che ogni volta si possa selezionare la cartella per il workspace. Una volta selezionata la cartella lancio.

Eclipse crea in automatico la cartella metadata, in cui sono salvati i dati che Eclipse usa per lavorare.



Nella finestra a sinistra, il **project explorer**, si vedono i progetti che ho.

Outline e task list al momento non ci interessano, quindi possiamo toglierli per lasciare spazio al codice. Allo stesso modo la finestra sotto.

Le schede fondamentali sono:

- Quella del codice;
- La console, in cui vedo l'esecuzione del mio codice, il risultato.

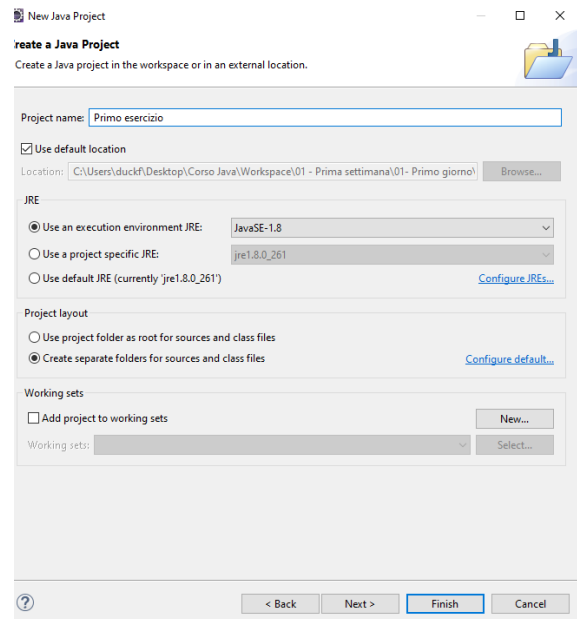
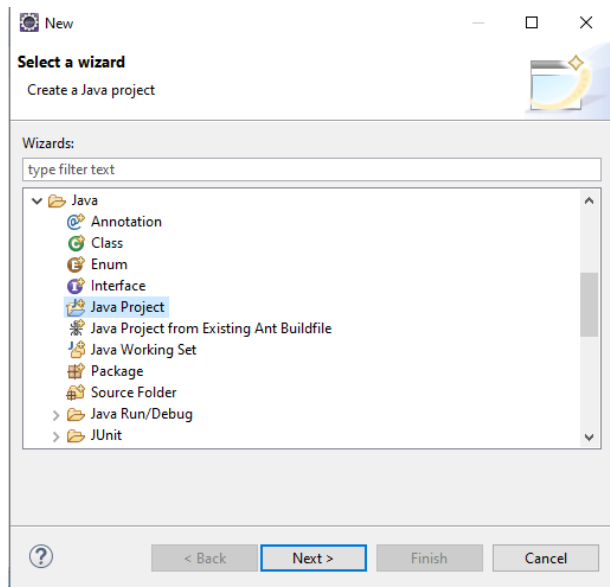
Una **virtual machine** è una macchina virtuale che mi permette di eseguire il codice. Il risultato dell'esecuzione si vede nella console.

1.1 CREARE NUOVO PROGETTO

Per creare un nuovo progetto devo fare:

file -> new -> other

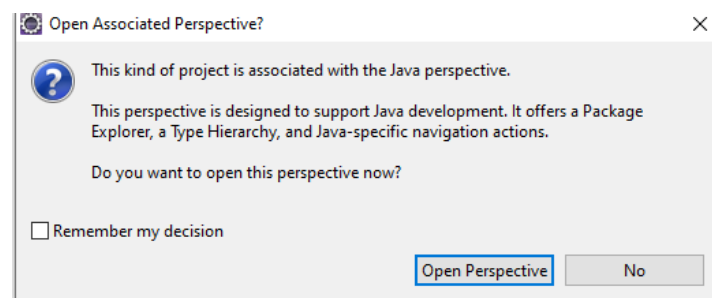
A questo punto ci sono vari tipi di progetti che vengono dati in automatico, noi dobbiamo fare "java project", siccome non c'è subito in vista, dobbiamo cercare in "altro".



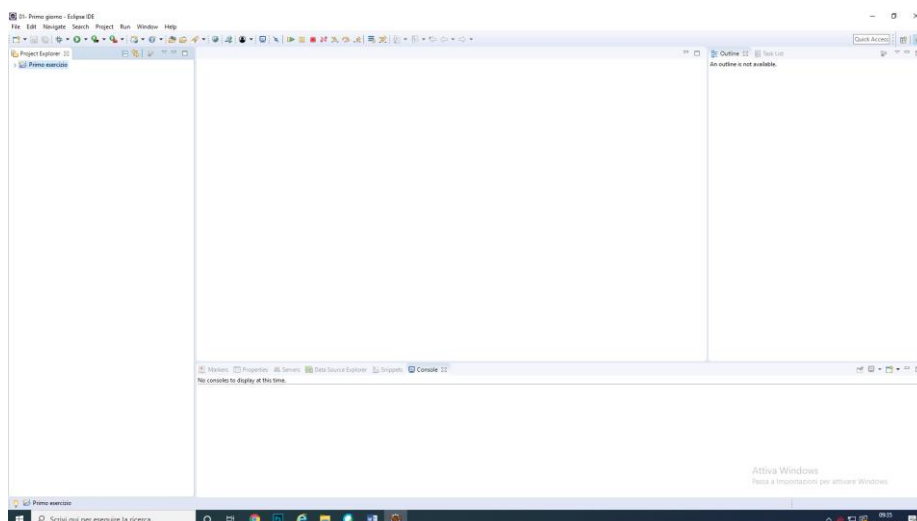
Per ora devo:

- 1- Impostare il nome del progetto;
- 2- Usare "default location"= salva il progetto utilizzando la cartella di riferimento del workspace, data quando ho aperto Eclipse.

A questo punto lancio e viene fuori questo messaggio:








Dice che chi fa progetti in Java, normalmente usa un workspace diverso. Chiede se voglio che apra la scrivania come dovrebbe essere di default. Diciamo di no, così viene mantenuta la scrivania come l'abbiamo impostata prima.



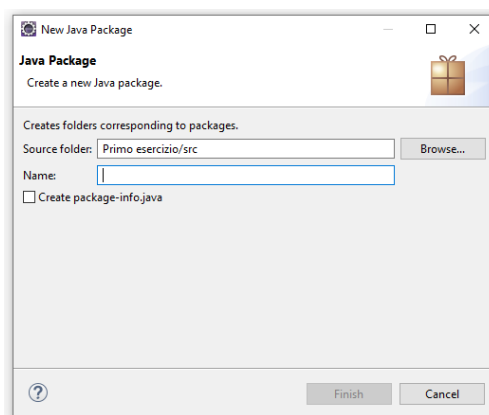
Siamo a questo punto: abbiamo creato il primo grande contenitore, la cartella in cui saranno salvati i dati.

La cartella è strutturata in questo modo:

	.settings	15/10/2020 09:33	Cartella di file	
	bin	15/10/2020 09:33	Cartella di file	
	src	15/10/2020 09:33	Cartella di file	
	.classpath	15/10/2020 09:33	File CLASSPATH	1 KB
	.project	15/10/2020 09:33	File PROJECT	1 KB

Dentro a “src” vanno messi tutti i file che creo.

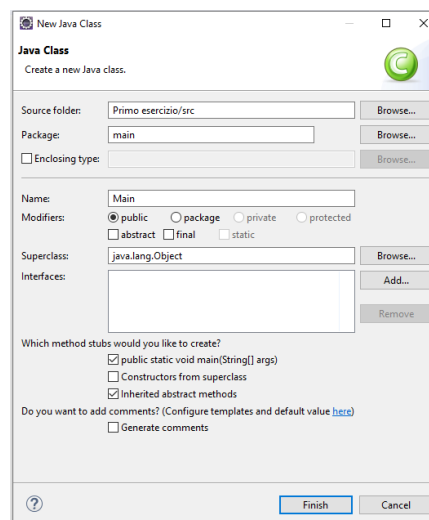
La cartella “Jre system library” contiene le librerie installate, la virtual machine ecc.



Creo un nuovo **package**, ovvero un contenitore. Devo dire dove inserisco il package e poi inserire il nome.

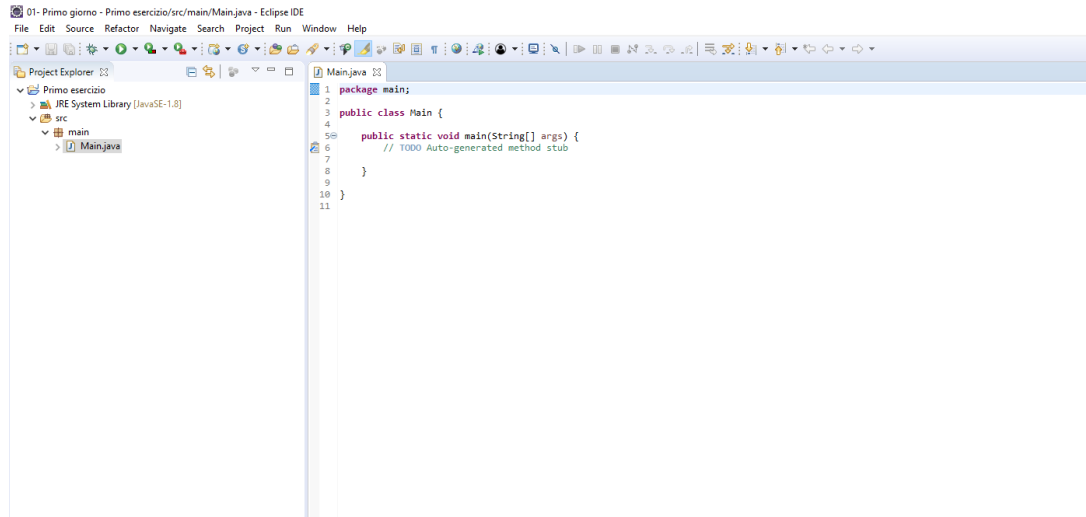
Creo un package di nome “main” dentro alla cartella “src”. Per convenzione il nome dei package è sempre scritto con la lettera iniziale minuscola.

Dentro al package “main” creo una nuova classe:



La classe invece va nominata sempre con la lettera iniziale maiuscola, per convenzione. Per ora non tocco altro. Devo solo spuntare il “Public static void main”.

A questo punto ho creato la classe:



Il package diventa marrone, perché ora contiene una classe.

Una **classe** è un contenitore, in questo caso si tratta di una **classe di avvio**.

“Public static void main args”= è un codice che permette di eseguire questo file. Vuol dire che tutto ciò che sta dentro alle graffe può essere eseguito. Per questo si chiama “classe d’avvio”.

```
1 package main;
2
3 public class Main
4 {
5     public static void main(String[] args)
6     {
7         System.out.println("Ciao mondo!");
8     }
9 }
10
```

```
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (15 ott 2020, 09:52:09)
Ciao mondo!
|
```

In console mi compare il risultato.

2. JAVA

2.1 Cose di base

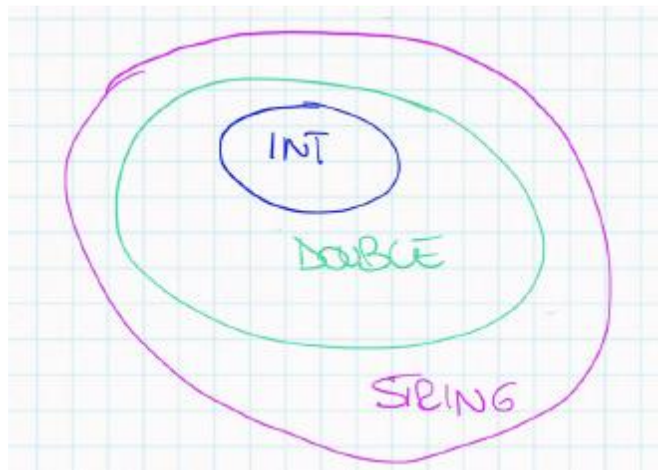
2.1.1 Sintassi

- Con `//` apro un commento in una sola riga.
- Con `/* */` apro e chiudo un commento lungo più righe.
- `+=` serve a sommare il valore precedente di una variabile a nuove cose.
- `\n` serve per andare a capo.

2.1.2 I tipi

In Java abbiamo vari tipi di dati:

- `String` = dati di tipo testuale, è l'insieme più grande di dati, che contiene tutti gli altri;
- `int` = numeri interi;
- `double` = numeri decimali
- `char` = singola lettera.



```
String esempioStringa = "Ciao";  
int esempioInt = 3;  
double esempioDouble = 2.3;  
char esempioChar = 'i';
```

Una variabile può contenere un solo tipo alla volta. Può essere che due tipi diversi vengano concatenati ma il risultato è una `String`.

```
String esempioConcat = "la mia età è" + 32 + "anni";
```

2.2.3 Le variabili

```
String esempioStringa = "Ciao";  
int esempioInt = 3;  
double esempioDouble = 2.3;  
char esempioChar = 'i';
```

Questi sono esempi di variabili.

Importante è distinguere:

- **Dichiarazione** della variabile = dichiaro una variabile di un certo tipo e con un certo nome, a livello di macchina sto occupando uno spazio nella memoria del computer;
- **Inizializzazione** della variabile = è il momento in cui la variabile viene effettivamente creata, cioè le viene assegnato un valore iniziale. Non ho più solo la scatola ma inserisco qualcosa nella scatola.

```
String esempioStringa;  
  
esempioStringa = "Ciao!";
```

2.2.4 Principi della programmazione

- **Principio di sequenzialità** = Java legge il programma in sequenza, dalla prima all'ultima riga.
- **Metodo di lavoro DICO** = dichiarazione, inizializzazione, calcolo, output.
- **Selezione** = in un sistema comune dobbiamo avere la possibilità di gestire il dato in entrata e valutarlo prima di autenticarlo.

2.2.5 Convertire string in numeri

- **Integer.parseInt()** = serve a convertire una stringa numerica in integer;
- **Double.parseDouble()** = serve a convertire una stringa numerica in Double.

2.2.6 Lo scanner e gli input dell'utente

Lo **Scanner** è un elemento di Java che permette al programma di stare in ascolto. Può ascoltare la tastiera, un DB, un file, ecc.

Lo Scanner quindi rileva gli input provenienti dall'esterno. Tutti gli input che provengono dall'esterno sono sotto forma di String.

Lo Scanner non è una cosa che Java conosce di default, ogni volta devo farglielo imparare. Bisogna importare una libreria dello Scanner.

```

1 package main;
2
3 import java.util.Scanner;
4
5 public class GeometriaInput
6 {
7     public static void main(String[] args)
8     {
9         // Chiedere all'utente come si chiama
10        // Chiedere all'utente la misura della base e
11        // dell'altezza del rettangolo.
12        // Calcolare e stampare la misura dell'area
13
14        // Dichiarazione
15
16        Scanner tastiera;
17
18    }
19 }
20

```

In questo caso vedo che ho importato il pacchetto corretto.

Alcune librerie vengono importate di default per ogni progetto Java. Altre vanno importate all'occorrenza, lo Scanner è uno degli elementi da importare.

Per ora però l'ho solo dichiarato, bisogna iniziarlo:

```

// Inizializzazione
tastiera = new Scanner(System.in);

```

Questa è la riga per inizializzare lo Scanner, così Java sa che deve mettersi in ascolto dell'input proveniente dalla tastiera dell'utente. Il fatto che la variabile si chiami "tastiera" è parlante ma non è necessario. Una volta eseguita la riga, Java rimane in attesa dell'input dell'utente.

Questa riga serve per gli input da tastiera, se voglio leggere input da DB o altre fonti sarà diverso.

```

// Dichiarazione
Scanner tastiera;
String nomeUtente;
double baseRettangolo, area;
int altezzaRettangolo;
String risposta;

// Inizializzazione
tastiera = new Scanner(System.in);
risposta = "";

System.out.println("Come ti chiami?");
nomeUtente = tastiera.nextLine();

```

Con "tastiera.nextLine()" dico di leggere l'input che l'utente scrive dopo la linea precedente.

```

System.out.println("Ciao " + nomeUtente + "inserisci il valore della base in cm!");
baseRettangolo = Double.parseDouble(tastiera.nextLine());

```

In questo caso faccio inserire un valore numerico, però tutti gli input sono String, perciò avrò bisogno di convertirlo.


```
int nuovoNumero =tastiera.nextInt();
double nuovoNumero2 =tastiera.nextDouble();
|
```

Potrei fare anche in questo modo. Queste due righe fanno la stessa cosa del `parseInt` e `parseDouble`, però lo fanno peggio, nel senso che con `parseInt` e `parseDouble` viene presa la stringa e viene convertita.

Con `nextInt` e `nextDouble` invece viene convertita direttamente senza preoccuparsi di cosa ci sia dentro, il che può portare a errori. Quindi meglio usare `parseInt` e `parseDouble`.

```
tastiera.close();|
```

Fondamentale alla fine del programma è inserire questa riga, per indicare a Java che deve smettere di attendere input.

Il canale di immissione dati tra esterno e interno va chiuso perché si consuma una risorsa inutilmente. Se si termina il programma ma la tastiera non è chiusa, potrebbe dare problemi in futuro.

Questo è un esempio di programma completo:

```
public static void main(String[] args)
{
    // Chiedere all'utente come si chiama
    // Chiedere all'utente la misura della base e
    // dell'altezza del rettangolo.
    // Calcolare e stampare la misura dell'area

    // Dichiarazione
    Scanner tastiera;
    String nomeUtente;
    double baseRettangolo, area;
    int altezzaRettangolo;
    String risposta;

    // Inizializzazione
    tastiera = new Scanner(System.in);
    risposta = "";

    System.out.println("Come ti chiami?");

    nomeUtente =tastiera.nextLine();

    System.out.println("Ciao " + nomeUtente + " inserisci il valore della base in cm!");
    baseRettangolo = Double.parseDouble(tastiera.nextLine());

    System.out.println("La base del rettangolo è " + baseRettangolo + ". Inserisci il valore dell'altezza in cm!");
    altezzaRettangolo = Integer.parseInt(tastiera.nextLine());

    System.out.println("L'altezza del rettangolo è " + altezzaRettangolo);

    // Calcolo
    area = baseRettangolo * altezzaRettangolo;

    risposta = "L'area del rettangolo di base " + baseRettangolo + " e altezza " + altezzaRettangolo + " è: " + area;

    System.out.println(risposta);

    // Chiudo ascolto input
    tastiera.close();
}
```

2.2.7 Secondo principio della programmazione – la selezione (if)

In un sistema comune dobbiamo avere la possibilità di gestire il dato in entrata e valutarlo prima di autenticarlo, come una password.

```
// If
if(nome.equals("tomas")) {
    risposta = "benvenuto!";
}
```

Ecco un esempio di **if**, abbiamo:

- **Condizione** = la prima parte, se è vera allora si esegue la seconda parte.
- **Codice da eseguire.**

In questo caso si può fare senza le graffe, in questo modo:

```
// If
if(nome.equals("tomas")) |
    risposta = "benvenuto!";
```

Questo perché c'è solo una riga di codice dopo.