

Quello che abbiamo visto fino ad adesso non era solo java, ma era tutti l’OOP (ereditarietà, polimorfismo, messaggi paritari e sub etc.). Da adesso, tolto le cose classiche, inizieremo a vedere gli elementi proprietari di java. La prima cosa da segnarsi in testa è JAVA NON è SOLO UN LINGUAGGIO. È fondamentale per ragionare. Sono tre cose:

* Un linguaggio
* Una jdk
* Una virtual machine

Lato computer la virtual, lato IDE il linguaggio, il punto che collega il linguaggio alla macchina è la jdk (collega l’eseguibile alla compilazione della macchina). Quando lavoriamo nei contesti di java dobbiamo ricordarci di dividere l’implementazione reale (librerie ATTE a farlo funzionare in determinati modi), e il fatto che lui abbia un linguaggio che descrive due tipi di logica, una esecutiva dove scrivi le cose se voglio che funzionano, una serie di correlazioni create in maniera diretta o meno per far funzionare il sistema, DALL’altra parte abbia tutta la parte logica comune per garantire la coerenza, che si occupa di struttura la base logica che andiamo ad utilizzare per far funzionare java. QUINDI DA UN LATO C’è UN LINGUAGGIO RESO COERENTE DA NOI, E IN SECONDA BATTUTA DOBBIAMO COMPRENDERE CHE INVECE LA PIATTAFORMA ALLA BASE (le installazioni di java) E’ COERENTE (coerenza funzionale del linguaggio).

La seconda parte della slide, è una delle principali caratteristiche di JAVA, cioè la portabilità, cioè la capacità di essere riprodotto in maniera identica indipendentemente dal contesto applicativo, e in secondo luogo la robustezza (cosa vuol dire? Nel momento in cui vado ad eseguire uno script di java, non esistono, ma solo in rari casi, contesti dove l’errore è inaspettabile. Togliamo la possibilità di non aver configurato java bene, a quel punto, ho escluso funzionalmente tutte le casistiche di errore inaspettato. Non ho detto che funzionerà sempre, ma quando andrà in errore la macchina saprà dirmi in che errore è andato. Java ci permette perfino di creare classi di errore, cioè classi che contengono al loro interno casistiche di errori. EVITARE CONTESTO INASPETTATO). Possono essere prevenute con il concetto di try e catch (rivedremo più avanti), ma questa è proprio la logica su cui lavoriamo. Una volta buildato un sw, possa essere utilizzato in più punti senza riscriverlo, e in base alle funzioni e librerie utilizzati, gli errori possibili siano delimitabili in un range, in cui io vado a lavorare per prendere precauzioni, ma posso escludere X casistiche che non rientrano nel contesto che stiamo utilizzando. Eccezione è qualcosa che deriva da java, l’errore è un errore (manca ram, manca memoria, non c’è posto fisico, quindi in generale qualcosa SOTTO l’alto livello di java, in cui java anche volendo non può accedere e può solo dire che è andato in errore, e darci più o meno spiegazione logica).

Immagine che contiene testo, schermata, Carattere, algebra

Il contenuto generato dall'IA potrebbe non essere corretto.

Java è un linguaggio popolare. Era utilizzato per tutto quello elencato sopra, in contesti ampli, anche se adesso non sono più tutti quanti correlati a java. Uso java per dei motivi: per la sicurezza, perché permette di andare a fare delle strutture e comportamenti avendo un contesto già debuggato da terzi, in cui non devo ritrovare determinate logiche, ma sono pronto; Due, il contesto applicativo è pieno di esempi; Tre, esiste già da molto tempo, e ci sono diversi strumenti che possono andare sempre ad ottimizzare (ma questo è più un fatto di fiducia umana, per il tanto tempo passato).

Immagine che contiene testo, schermata, Carattere, biglietto da visita

Il contenuto generato dall'IA potrebbe non essere corretto.

Ci sono due regole:

* Il file .java si chiama sempre con la lettera maiuscola (non si deve usare mai la lettera minuscola);
* La classe in cui all’interno c’è il main, dev’essere con il nome del file, quindi dentro alla classe con il nome del file (deve portare il nome del file). Non posso avere due main ovviamente in uno script.

Immagine che contiene testo, Carattere, schermata, linea

Il contenuto generato dall'IA potrebbe non essere corretto.

Spiegazione completa di ogni parte basilare che completa il metodo.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Ogni riga del codice eseguita si deve trovare in ogni file .class e all’interno (per ora) del file e del main. QUINDI ogni cosa che noi vogliamo utilizzare ed ad avere come riferimento deve essere all’interno di un file java (pippo.class), e le cose che devono essere lette (le righe da far leggere) per ora sono dentro al main.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Come faccio ad essere certo che println è un metodo? Se vediamo un nome in generale, con le parentesi tonde, nel 90% dei casi o usiamo una funzione (riga 2), o stiamo DEFINENDO una funzione (riga 1). Quando definisco una funzione devo dare tutte le caratteristiche. Quando la uso, richiamo solo il nome (in questo caso println). Il main non viene mai richiamato, ma viene invocato nel momento in cui utilizziamo un file di java. Le parentesi graffe delimitano i blocchi di codice, una linea che divide la logica applicativa. Dopo le parentesi graffe non vanno mai i punti e virgola.

System è la prima libreria che prendiamo in considerazione (libreria del sistema). Quindi in generale, per l’interazione con il sistema.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

**Conferma il polimorfismo**, perché posso passare valori diversi al metodo e lui li elabora in modo corretto, ma il codice che scrivo rimane sempre lo stesso.  
**Conferma l’ereditarietà**, perché dietro c’è una gerarchia di classi che permette di ottenere un comportamento coerente e funzionante.  
**Conferma l’incapsulamento**, perché esiste uno spazio nascosto, non visibile al programmatore, che contiene dettagli e possibili errori, ma questi restano isolati dall’esterno.  
**Conferma l’astrazione**, perché non conosco il funzionamento interno del metodo, ma mi basta sapere cosa fa; nasconde i dettagli di implementazione ed espone solo ciò che serve, cioè la capacità di stampare un valore.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Nella macchina la matematica lineare viene abbandonata, in cui lo 0 e 1 rappresentano stati di verità, true e false (booleani).

L’obiettivo dell’OOP è fottere la macchina, perché ragiona in maniere lineare ma solo per le macchine, ma non per noi, e noi dobbiamo ragionare in modo tale che utilizzando l’oop la macchina ragioni come noi.

Java è un linguaggio di OOP ad alto livello di astrazione e fortemente tipizzato. Vuol dire che è più vicino al linguaggio umano che al linguaggio della macchina. Per la macchina system.out non vuol dire nulla (ma neanche in assembly non vuol dire nulla), ma è un comando capito da noi, ma java lo converte e lo compila verso la macchina. Fortemente tipizzato vuol dire che se una variabile è un numero è un numero e basta., ci sono poi regole che ci permettono di cambiare regole come il casting etc., ma di base se qualcosa viene creata così, rimane così. Python per esempio è dinamicamente fortemente tipizzato (quindi non tutti sono come java, perché puoi cambiare e fare cambi di valore tipo x=5 prima e poi subito dopo x=”mirko”).

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Commenti da mettere SEMPRE, proprio per la capacità di saper leggere il codice in maniera efficace.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

La variabile è un identificatore univoco (può esistere un solo x per esempio) e devono avere nome, tipo, valore. Le uniche due obbligatorie sono nome e tipo, il valore può essere assegnato in un secondo momento (appatto che esista). L’”uguale” non deve essere sottovalutata, perché a livello logico l’uguale è un macello pazzesco. L’uguale singolo non equivale all’uguaglianza. Un valore intero x = assigned at 5. Una variabile intera x che ha come valore assegnato 5. Smettere di leggere codice equivale a debito tecnico, per questo è importante capire tutto quello che c’è scritto.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Final serve a creare le costanti.

Le string possiamo concatenarle. Non sono un tipo basilare normale uguale agli altri. Le **stringhe in Java NON sono primitive**, ma oggetti appartenenti alla classe String, con comportamento e metodi propri.

I tipi possono essere primitivi o derivati, primitivi come questi qua sopra. Quello più strano è string anche perché a differenza degli altri si scrive con la S maiuscola, ed è effettivamente diverso dagli altri perché è un tipo di tipi, un estensione del char, e perché ha a disposizione i metodi, e tramite la logica dei metodi può strutturare una serie di logiche che gli altri non possono.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Le variabili vanno rigorosamente trattate con la lettera minuscola iniziale. Una pessina denonimazione può portare problemi di comprensione del codice. Utilizzare variabile in maniera denonimata correttamente (stile camelcase come per esempio minutiPerOra)

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Altre regole fondamentali da sapere. Scrivere variabili con nomi uguali ma con caratteri uno maiuscolo e una minuscolo sono variabili diverse. $ e \_ non dobbiamo usarle (lavorare in maniera semplice ed isolata di java).

Immagine che contiene testo, schermata, Carattere, algebra

Il contenuto generato dall'IA potrebbe non essere corretto.

Tre categorie. Queste sono variabili numeriche, come byte, short, int , long. Ma ci interessa principalmente int e di conseguenza il double. Float e double hanno una differenza.

Immagine che contiene testo, schermata, Carattere, algebra

Il contenuto generato dall'IA potrebbe non essere corretto.

Quando li mettiamo all’interno del codice in maniera fissa (quindi noi a mano), non ci va la virgola ma il punto, e oltre questo, nel momento in cui andiamo ad inserirlo va specificata la d o la f alla fine. In base a quello che noi vogliamo creare quindi (inserire una variabile reale inserita a mano, double, dovrò inserire la d. al contrario se ho il float, dove insieriamo la f).

Immagine che contiene testo, schermata, Carattere, algebra

Il contenuto generato dall'IA potrebbe non essere corretto.

Il principale tipo primitivo che possiamo usare è il booleano.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

I char non andremo ad usarli. Char sono utili nel contesto di basso livello, ma realisticamente abbastanza inutili in un contesto dove si lavora all’interno di elementi con logiche più alte e complesse. Contesti del genere, gli elementi che noi stiamo andando a utilizzare non sono nient’altro che strutture dati che poi compongono le stringhe. Per questo in questo caso non le usiamo, non andiamo a lavorare su ogni singolo elemento.

Le stringhe già le abbiamo viste, ma c’è una cosa che non abbiamo visto in maniera approfondita. All’interno delle stringhe esistono contesti in cui possiamo usare metodi. Se prendiamo il nome della stringa testuale e andiamo a crearci intorno un punto (testo. , al tuo interno esitono metodi, che possono essere utilizzate per strutturare dei comportamenti, diversi tra loro e che analizzeremo, servono principalmente per collegare logiche attuative, quello che dobbiamo fare, a come dobbiamo farle)

Possiamo richiamare così un char alla volta. Nella realtà dei fatti quello che serve questa logica, è andare a strutturare un insieme di capacità per il controllo, controllare in maniera precisa come lavora (lavoro pratico) i controlli sulle stringhe, permettendo di paragonare stringhe anche lettere per lettere.

Immagine che contiene testo, Carattere, schermata, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

I tipi non primitivi sono chiamati tipi di riferimento perché si riferiscono agli oggetti. Sono quindi i tipi definiti dagli utenti gli unici tipi non primitivi, cioè gli oggetti.

I tipi primitivi sono già in java, tutti gli altri definiti da noi, cioè oggetti, sono quelli considerati non primitivi. Il secondo punto, è il dibattito sulle stringhe: “Nasce dal fatto che, visto che le stringhe possono chiamare metodi, sono per forza tipi non primitivi”. La verità è che sono tipi primitivi in quanto strutture dato che possono raccogliere più dati contemporaneamente.

Tipi non primitivi possono non avere valore, al contrario dei primitivi che devono averlo, se non deve essere settato a NULL.

Le lettere corrette devono essere scritte perché in questo modo si struttura un comportamento di lettura codice coerente, che ci permettono di comprendere meglio cosa accade senza impazzire.

Type casting è la possibilità esplicita di convertire un valore, solo tra tipi vicini, come in slide. Ne esiste uno automatico e uno reale. Li proviamo. Si può fare solo da un tipo più piccolo ad un tipo più grande in maniera IMPLICITA. Questo perché non si possono perdere dei dati. Conversione esplicita solo se vogliamo fare il contrario.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Parliamo degli scanner, il primo oggetto vero e proprio che vediamo e che utilizzeremo. Quando creiamo oggetti abbiamo bisogno dell’istanziamento (che avviene a riga 5), che è la semantica che vediamo all’interno del contesto delle variabili. Quello che stiamo importando è uno scanner, cioè una classe. Importiamo all’interno della logica, uno scanner. Questo scanner è una classe e lo riconosciamo dalla lettera maiuscola. Questo scanner, quello che fa, è prendere un input. Questo input ci serve direttamente per andare ad utilizzare all’intenro del contesto attuativo dei dati. Quando gireremo questo codice lui aspetterà una stringa, e attivamente in console andare a utilizzare e valorizzare un elemento (una stringa per esempio). Non possiamo aspettarci che sia il sistema che faccia tutto solo, ma dobbiamo ottimizzare la struttura di riferimento, l’oggetto.

Nella riga 5 creiamo un oggetto di tipo scanner. Ho un elemento di tipo scanner chiamato myobj. Sto creando un nuovo oggetto di tipo scanner, che permetterà di prendere l’input.

Immagine che contiene testo, schermata, Carattere, documento

Il contenuto generato dall'IA potrebbe non essere corretto.

Dopo gli scanner, faremo un esercizio

Immagine che contiene testo, ricevuta, Carattere, bianco

Il contenuto generato dall'IA potrebbe non essere corretto.

Partiamo poi con gli operatori

Immagine che contiene testo, Carattere, ricevuta, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Gli operatori di assegnazione, assegnano e valorizzano.

Immagine che contiene testo, schermata, numero, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, Carattere, numero, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

una funzione è isolata, può essere chiamata anche senza una funzione che la veicola. un metodo no, al contrario, deve avere qualcosa che veicola le sue condizioni, quindi tramite un punto. in comune hanno le parentesi

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Ci permette di utilizzare elementi normalmente non utilizzabili nel contesto di java.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Logiche di formattazione. Collaborazioni funzionali per il sistema, per formattare il codice. A noi interessa andare a lavorare all’interno di un contesto che ci permetta di avere tutti i tipi di elementi che ci interessano, senza andare a buggare qualcosa. Obiettivo è formattare direttamente dalla struttura funzionale.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Definisice come si presentano i nostri caratteri, quanto è lunga la stringa.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Equals è il più importante. Serve a confrontare in maniera specifica due stringhe, e o due oggetti che contengono a loro volta una stringa. È fondamentale perché permette di lavorare nel contesto attuativo, quindi nel contesto pratico, tramite una linea logica più diretta e lineare, occupando e delineando in maniera più specifica, delineando comportamenti comune senza andare ogni volta a rispiegare tutti i controlli di uguaglianza.