Reflection

Ne permite sa aflam la runtime tipul datelor.

Ne ajuta sa luam la runtime fielduri , metode pt a le modifica valaorea, sa le setam valori.

La ce avem nevoie:

Il examinam la run-time

Construim

Analizam metode

Invocam

Schimbam accesibilitatea

Cu ajutorul reflection putem sa punem la Junit @ Test pt a schimba rolul ei la runtime, pot sa folosesc assert.//altfel as avea erori de compilare

Junit e un framework, foloseste Reflection.

Prin reflection pot sa iau si campurile private.

Reflection e consummator de memorie.

Pe JVM se pot pune restrictii, sa nu iei prin Reflection campurile de la private.

Incalca principiul incapsularii

Dintr-un enum pot sa am o singura instant.

Pe primitive nu pot sa apelez getClass, doar class

forName te ajuta la JDBC sa incarcam un driver

getDeclaredClasses pot afla toate inner classurile

getDeclaredField field-ul din clasa curenta – toate

getField doar pe cele publice din clasa respective

getDeclaredFields o lista de campuri private publice,

getFields din superclasa, doar pe cele publice

la Method la fel

la Constructor la fel

Adnotari

@MyAnnotation

Retention Policy

Restrictionam vizibilitatea adnotarii respective, in ce moment o putem lua

La runtime, la compiletime

La nivel de clasa, metoda, parametru si field

Target – unde pot folosi adnotarea

getDeclaredAnnotations(); - din clasa

getAnnotationd – le ia si din parinte ,si din clasa si din parinte

getAnnotation(someclass) – ti le ia pea lea specific