Kivételkezelés Javaban

Általános alak:

try

{

//hiba

}

catch (exception(type) e(object))‏

{

//hibajavitó

}

A kivételkezelés Javaban az egyik legerősebb mechanizmus arra a célra hogy leketeljük a futásidejű hibákat, azért, hogy fennmaradjon a program hibamentes futása.

Javaban egy kivétel egy olyan esemény, amely végrehajtásának normális folyamatát, ilyenek például: ClassNotFoundException, IOException, SQLException, RemoteException, stb.

Kivételek típusai:

1. Ellenőrzött kivételek:

Kétfajta kivétel létezik:1. futási időben keletkezett kivétel: Futási idejű kivétel a Java futtatórendszerében keletkezik: aritmetikus kivételek (például nullával való osztás), referenciával kapcsolatos kivételek (mint például egy objektum tagjaihoz való hozzáférés null hivatkozással) és indexeléssel kapcsolatos kivételek (mint például egy tömb elemeihez való hozzáférés olyan indexszel, mely túl nagy vagy túl kicsi). Egy metódusnak nem kötelező előírnia futási idejű kivételeket, de ajánlott.

2. nem futási időben keletkezett kivétel: olyan kivételek, melyek a Java futási rendszeren kívül keletkeznek. Például: kivételek, melyek I/O során keletkeznek. A fordító biztosítja, hogy a nem futási időben keletkezett kivételeket elkapják, vagy továbbengedjék; ezért ezeket ellenőrzött kivételeknek is nevezzük.

Azok az osztályok, amelyek közvetlenül örökölnek eldobható osztályokat kivéve a RuntimeException-t és az Error-t.

1. Nem ellenőrzött kivételek:

Azok az osztályok, amelyek öröklik a RuntimeException-t,például:ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc. A nem ellenőrzött kivételek fordításkor nem ellenőrzöttek, viszont futás közben ellenőrzlésre kerülnek.

1. Error:Nem helyrehozhazóak, például: OutOfMemoryError, VirtualMachineError, AssertionError stb.

Ha egy metódusban hiba keletkezik, a metódus egy objektumot hoz létre, melyet átad a futtatási környezetnek. Az objektum – melyet kivétel objektumnak neveznek – tartalmazza az információt a hibáról, annak típusáról és a program állapotáról, amikor a hiba létrejött. Kivétel objektum létrehozását és futtatási rendszer által történő kezelését kivételdobásnak hívják.

Miután egy metódus eldob egy kivételt, a futtató környezet megpróbál a kezelésére találni valamit. A lehetséges dolgok, melyek a kivételt kezelik a meghívott metódusok rendezett listája abban a metódusban, ahol a hiba keletkezett. A metódusok listáját hívási veremnek nevezzük.

Kulcsszavak: try, catch, finally, throw, throws.

Try: Egy kivételkezelő elkészítésének első lépése, hogy elhatároljuk a kódot, ami hibát dobhat a try blokkban.

Catch: A try  blokkhoz hozzáilleszthetjük a kivételkezelést, amennyiben egy vagy több catch blokkot használunk közvetlenül a try blokk után. Semmilyen programkód nem lehet a try blokk vége és az első catch blokk között! Minden catch blokk egy kivételkezelő, és azt a típusú kivételt kezeli, amilyet a paraméter tartalmaz. A paraméter típusa deklarálja a kivétel típusát, amit a kezelő lekezel.

A catch blokk tartalmazza azt a programkódot, amely végrehajtásra kerül, amikor a kivételkezelőt meghívjuk. A futtatórendszer meghívja azt a kivételkezelőt, amelyik esetén az ExceptionType megfelel a dobott kivétel típusának.

Finally: Olyan mechanizmust nyújt, ami kiküszöböli azokat a figyelmetlenségeket, amik a try blokkban történtek. A finally blokkot például arra használhatjuk, hogy bezárjuk a fájlokat, amelyekre se a hiba nélküli futás, se a hiba dobása esetén nem szükségesek már.

Throw: A kivétel dobásánál minden metódus a throw kulcsszót használja, melynek a következő formai követelménynek kell, hogy megfeleljen:

**throw someThrowableObject;**

Throws: A metódus a deklarációja *throws* záradékában írja le, hogy milyen kivételeket dobhat.