

Support de cours

Cours:

Introduction à la programmation orientée objet (en Java)

Vidéo:

W16-02-exceptionsyntaxe-JAVA-pt3

Concepts (extraits des sous-titres générés automatiquement) :

Bloc catch. Flow d'exécution. Fin de cette ligne. Bloc try correspondant. Lancement d'une exception. Exécution de cette ligne. Gestion des exceptions. Âge inférieur. Travers d'un appel d'une méthode. Lancement d'exception. Fin du bloc. Exécution normale du programme. Exception. Principes généraux. Type d'exception.



[vers la recherche de séquences vidéo](#)

(dans Introduction à la programmation orientée objet (en Java).)



[vers la vidéo](#)

Center for Digital Education. Plus de matériel de soutien pédagogique ici :

<https://www.epfl.ch/education/educational-initiatives/cede/educational-technologies-gallery/boocs-en/>

Gestion des exceptions : syntaxe

(Partie 3)

Introduction à la programmation orientée objet (en Java)

Jamila Sam, Jean-Cédric Chappelier et Vincent Lepetit

...

notes

résumé

0m 0s





Voyons maintenant en détails comment tout ceci s'exécute.

notes

résumé

0m 1s



Un bloc `catch` n'est exécuté **que** si une exception de type correspondant a été lancée depuis le bloc `try` correspondant.

Sinon le bloc `catch` est simplement ignoré.

En l'absence du bloc `finally`, si un bloc `catch` est exécuté, le déroulement continue ensuite normalement **après** ce bloc `catch` (ou après le dernier des blocs `catch` du même bloc `try`, lorsqu'il y en a plusieurs).

En aucun cas l'exécution ne reprend après le `throw` !

Quel est le flow d'exécution ? Par où on passe en fonction des différents cas ? Si l'exception est lancée. Si l'exception n'est pas lancée. D'abord, les principes généraux sont les suivants. Un bloc `catch` n'est exécuté que si une exception correspondant au type spécifié dans le `catch` en question a été lancée depuis le bloc `try` correspondant. Sinon, le bloc `catch` est tout simplement ignoré. On n'exécutera pas ce qui est dans ce bloc `catch`. En absence de `finally`, dont on parlera dans un instant.

notes

résumé

0m 6s



Exemple :

si il n'y a pas d'erreur (**pas** de lancement d'exception) :

```
try {
    // ...
    if (...) {
        throw new Exception("Quelle erreur !");
    }
    // ...
}
catch (Exception e) {
    // ...
}
```

Si le bloc catch est exécuté, alors le déroulement du programme continue ensuite après le bloc catch. On ne revient en aucun cas dans le try. On continuera après le bloc catch. Par exemple, si on a un bloc try rendu sensible à la gestion des exceptions, dans lequel s'exécute, soit directement, soit au travers d'un appel d'une méthode, dans lequel s'exécute donc le lancement d'une exception. Si cette exception est lancée, si cette ligne est effectivement exécutée, à la fin de cette ligne, on va sauter directement ici à la première ligne du bloc catch, directement suivant le bloc try et correspondant au type d'exception. Ici, avec correspondance du type. Donc l'exécution sera la suivante : On arrive ici. On exécute ce lancement d'exception. On saute à cet endroit-là si il y a eu lancement d'une exception. Si par contre, il n'y a pas eu lancement de l'exception, c'est à dire que cette ligne, ici, n'est pas exécutée,

notes

résumé

0m 37s



Exemple d'utilisation de catch

① age < 150

②

```
try {
    // ...
    if (age >= 150)
    { throw new Exception("valeur trop grande"); }
    // ...
    if (x == 0.0)
    { throw new ArithmeticException("Division par zero"); }
    // ...
}

catch (ArithmeticException e) {
    System.out.println(e.getMessage());
    e.printStackTrace();
}

catch (Exception e) {
    System.out.println("Qui peut vivre si vieux?");
}
```

par exemple, parce qu'elle a été protégée par un if dont la condition n'est pas remplie, à ce moment là, on exécute le code. Il n'y a pas d'exécution de cette ligne. Donc on continue, naturellement, à exécuter tout le bloc. On suppose qu'il n'y a pas d'autre throw dans la suite. Arrivé à la fin du bloc, ici, on va continuer l'exécution normale du programme. On ignorera tout simplement tout le bloc catch qui n'est exécuté que si il y a eu lancement d'une exception du type correspondant. Sur l'exemple que nous avons tout à l'heure, le flow d'exécution serait le suivant : Supposons donc au départ que l'on a une variable age et puis que l'âge, pour commencer, n'est pas supérieur ou égal à 150. On a un âge inférieur à 150. A ce moment là, on continue. On ne rentre par dans le if donc on ne voit pas le throw, on continue. Supposons qu'on ait une variable x, ici, qui n'est pas égale à 0. A ce moment là, on continue normalement. On arrive ici. Donc si aucune exception n'a été lancée, "cas n°1 : sans exception lancée" on va continuer l'exécution là, en-bas. Ça c'est donc ce qui se passe si l'âge est inférieur à 150 et x est différent de 0.

notes

résumé

1m 49s



Exemple d'utilisation de catch

EPFL

- ① $age < 150$
- ② $age < 150 \wedge x == 0.0$
- ③ $age \geq 150$

```
try {
    // ...
    if (age >= 150)
    { throw new Exception("valeur trop grande"); }
    // ...
    if (x == 0.0)
    { throw new ArithmeticException("Division par zero"); }
    // ...
}

catch (ArithmeticException e) {
    System.out.println(e.getMessage());
    e.printStackTrace();
}

catch (Exception e) {
    System.out.println("Qui peut vivre si vieux?");
}
```

Voyons maintenant ce qui se passe si par exemple on a toujours l'âge inférieur à 150 mais que x vaut 0. Ce qui va se passer, c'est que l'âge étant inférieur à 150, on ne rentre pas dans le if, on continue. Ici, x valant 0, alors cette condition est vérifiée. On exécute ce throw, ce qui fait que, à partir de là, on va se brancher sur ce qui a été lancé comme exception. C'est à dire qu'on a lancé une ArithmeticException. Donc on va se brancher sur le bloc qui attrape les ArithmeticException. A partir d'ici, on va continuer l'exécution en affichant le message associé et en affichant la trace d'appel. Une fois que l'on aura fait tout ceci, à ce moment là, on continuera l'exécution normalement après tous les blocs catch associés aux try concernés. Troisième et dernier cas, si l'âge est supérieur ou égal à 150. Ce qui se passe c'est qu'on va arriver dans ce if, dont la condition est vraie donc on va exécuter ce lancement d'exception.

notes


résumé

3m 1s



Ici, on a donc lancement d'un type Exception. Ce qui fait que l'on va, à partir d'ici, se brancher sur le catch qui récupère les exceptions. Cette exécution va continuer ici par exécuter l'affichage de "Qui peut vivre si vieux ? ". Puis ensuite, comme toujours, on continuera l'exécution par la suite normale du programme. par la suite normale du programme.

4m 13s



A QR code is located in the bottom left corner of the page, next to the text '4m 13s'. The QR code is black and square-shaped, with three larger squares in the corners for alignment. It is positioned to the left of the main content area, which is a large white rectangle with horizontal dashed lines.

