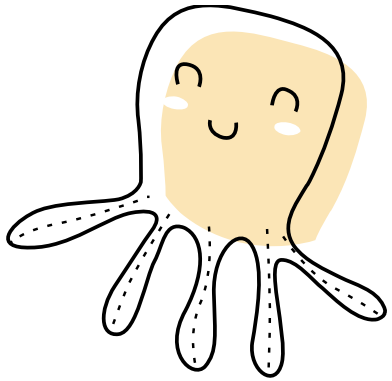
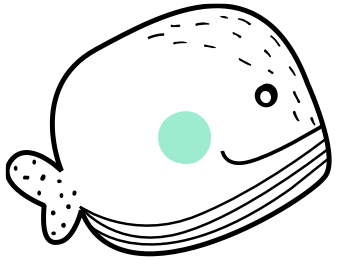

Welcom to
Group 11

Thành viên

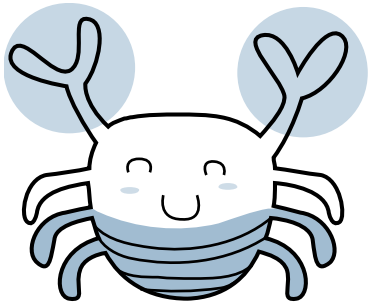
Nhóm 11



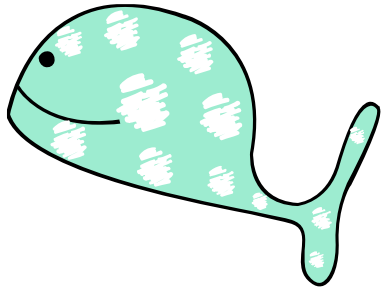
Nguyễn Đức Tài
2008110304



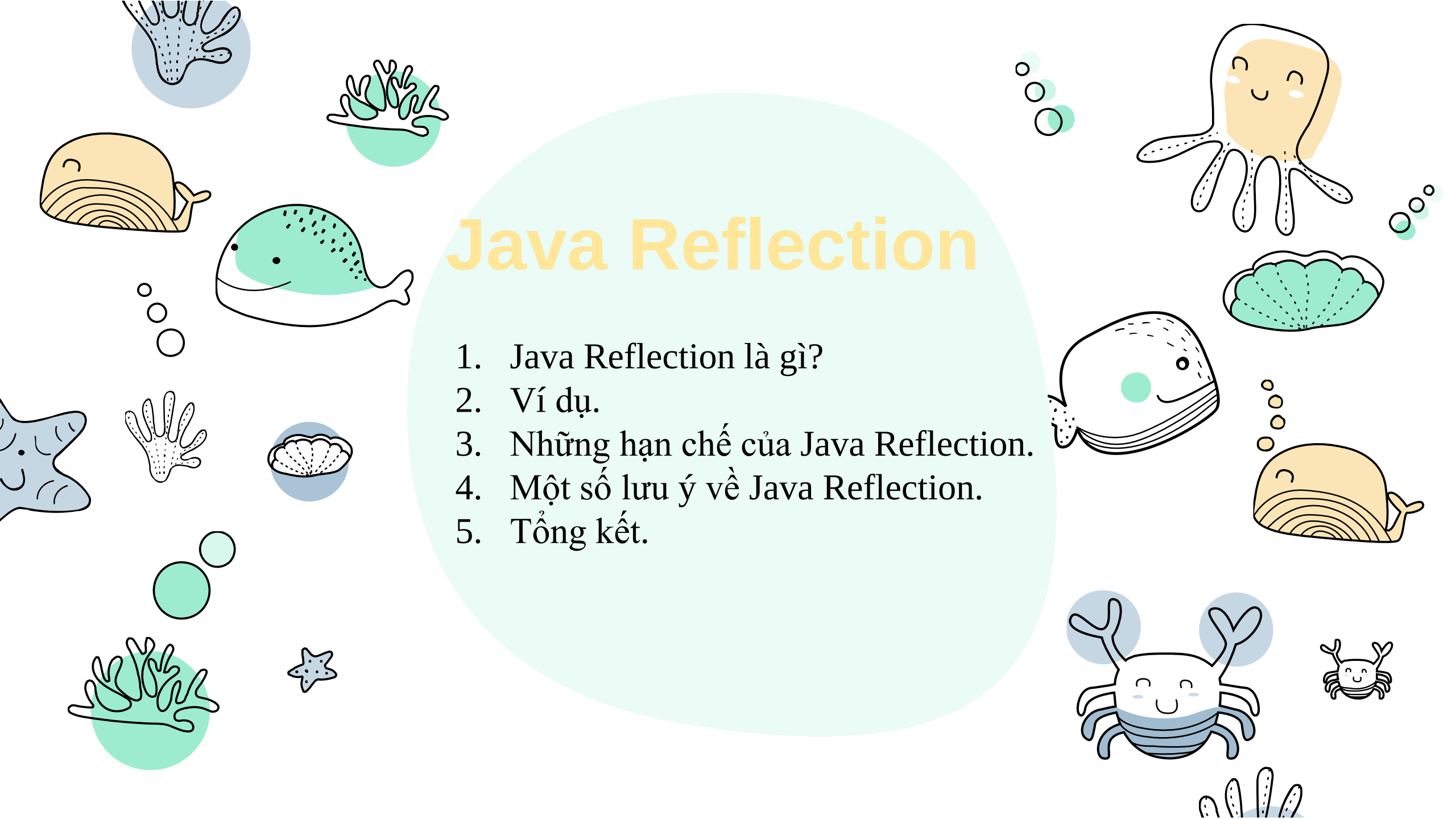
Nguyễn Hưởng
2004110008



Phan Hoàng Nam
2004110054



Nguyễn Minh Tuấn
2008110220



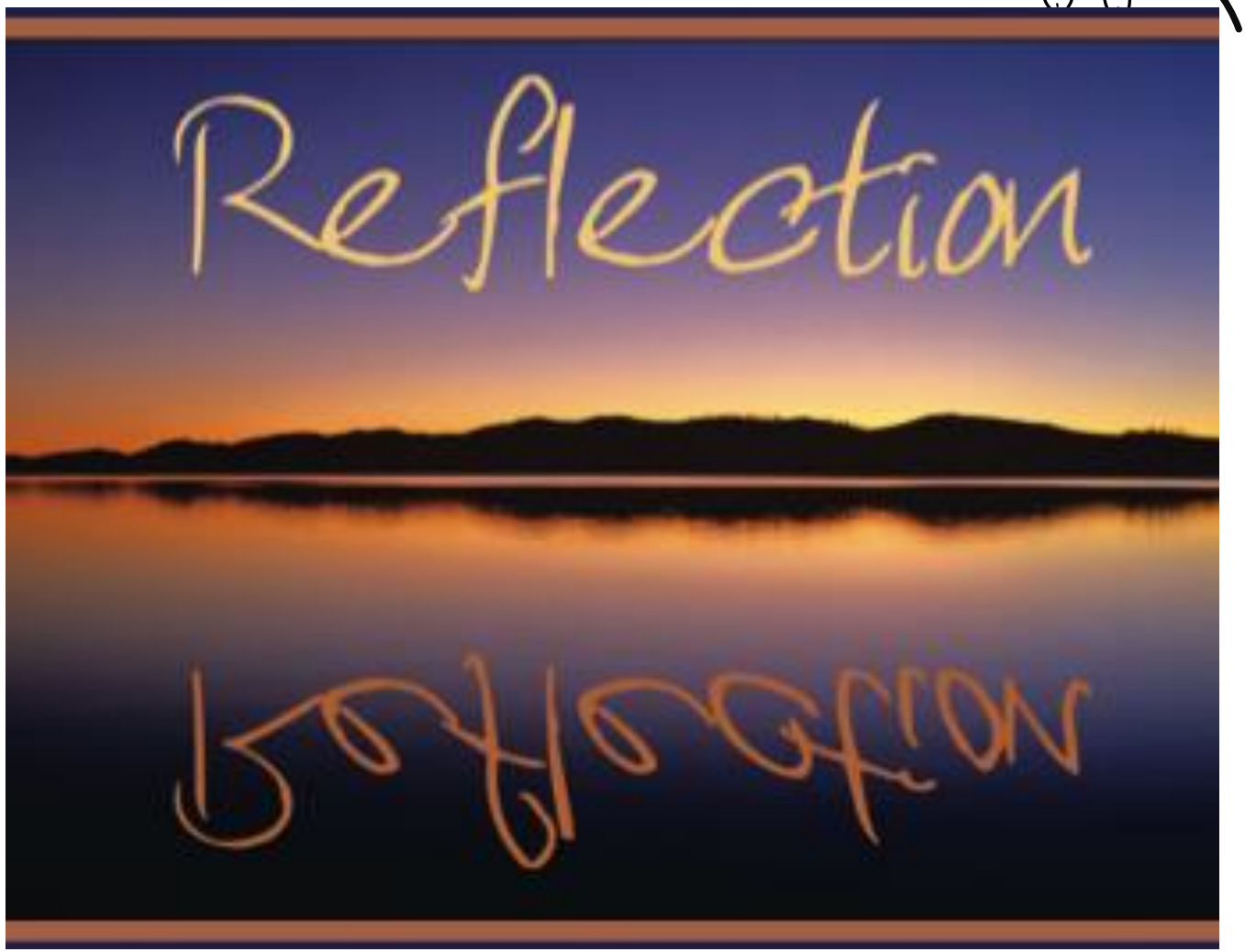
Java Reflection

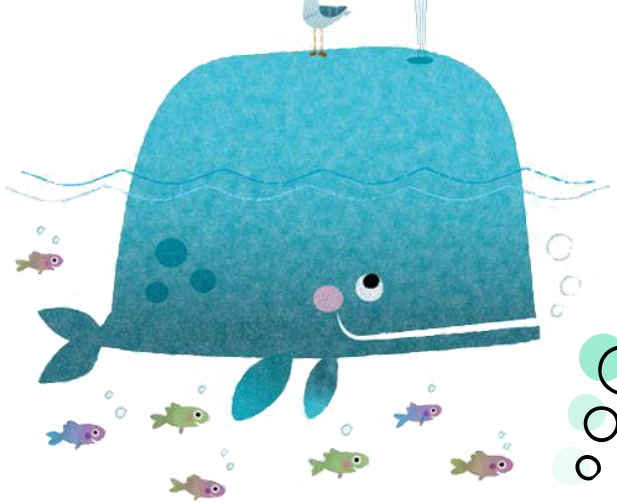
1. Java Reflection là gì?
2. Ví dụ.
3. Những hạn chế của Java Reflection.
4. Một số lưu ý về Java Reflection.
5. Tổng kết.

Khái niệm

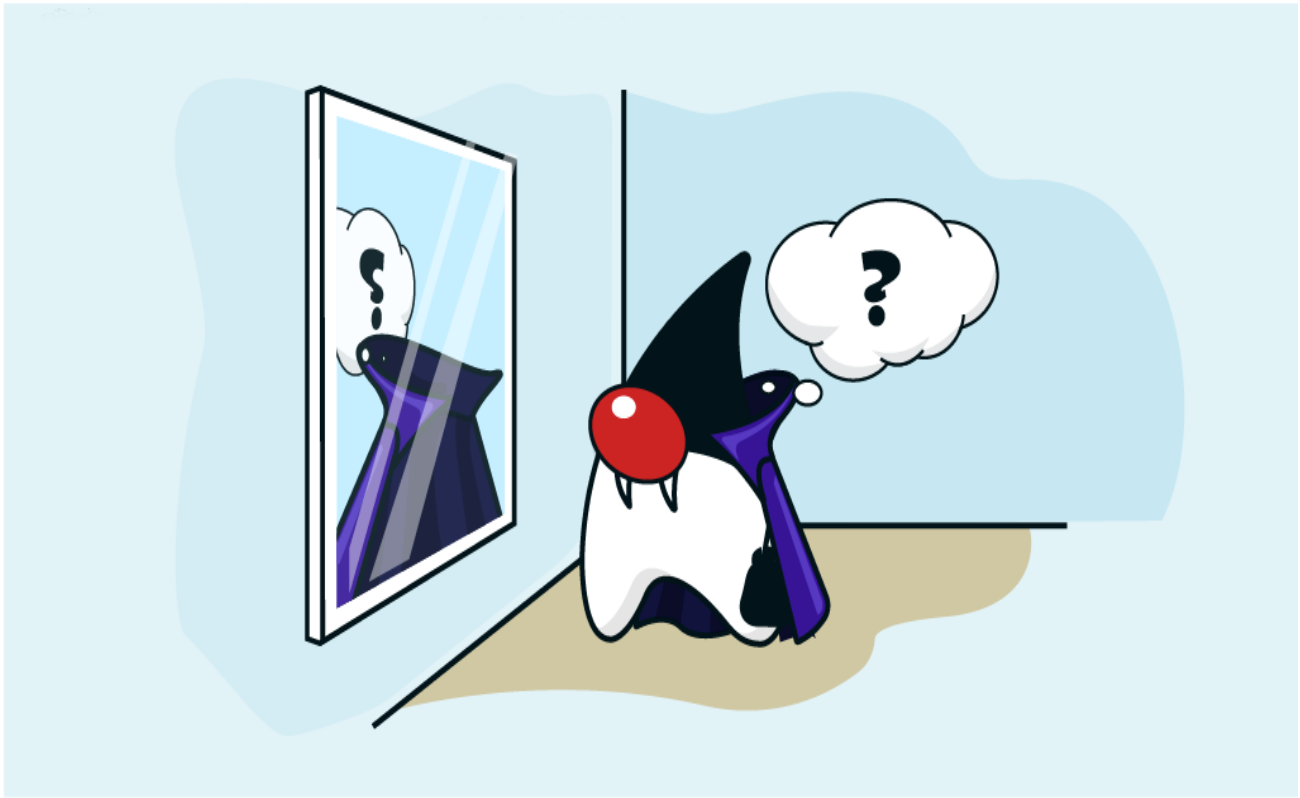
1.Java Reflection là gì

- Java sử dụng từ "Java Reflection" để đặt tên cho một API quan trọng trong thư viện chuẩn của Java. Tại sao API này lại được đặt tên như vậy? Chúng hãy cùng phân tích ý nghĩa của việc này.
- Reflection chính là một hình ảnh phản chiếu của một vật thể. Chẳng hạn hình ảnh của bạn trong một tấm gương, hoặc ảnh phản xạ của một cái cây trên mặt hồ. Từ "Java Reflection" đơn giản là đang ám chỉ một hình ảnh khác, một cách tiếp cận khác của Java



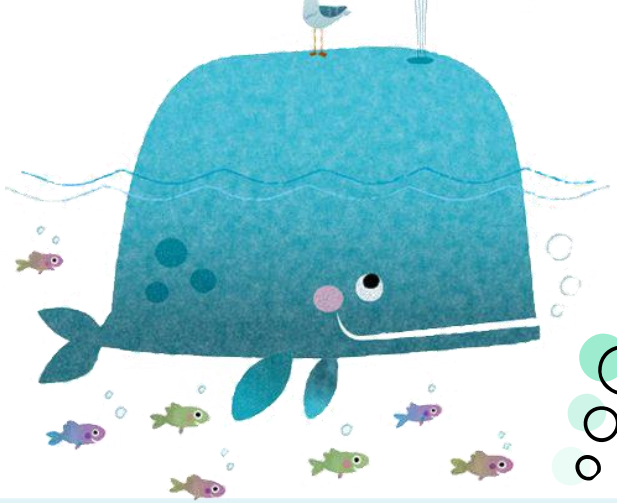


Khái niệm

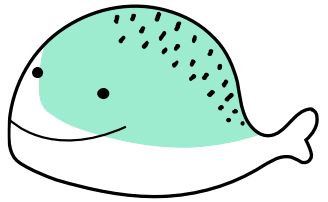


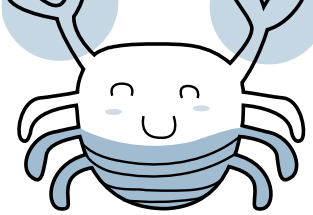
- Java là một ngôn ngữ hướng đối tượng (Object-oriented), thông thường bạn cần tạo ra một đối tượng và bạn có thể truy cập vào các trường (field), hoặc gọi phương thức (method) của đối tượng này thông qua toán tử dấu chấm (.)

- Java Reflection giới thiệu một cách tiếp cận khác, bạn có thể truy cập vào một trường của một đối tượng nếu bạn biết tên của trường đó. Hoặc bạn có thể gọi một phương thức của đối tượng nếu bạn biết tên phương thức, các kiểu tham số của phương thức, và các giá trị tham số để truyền vào ...



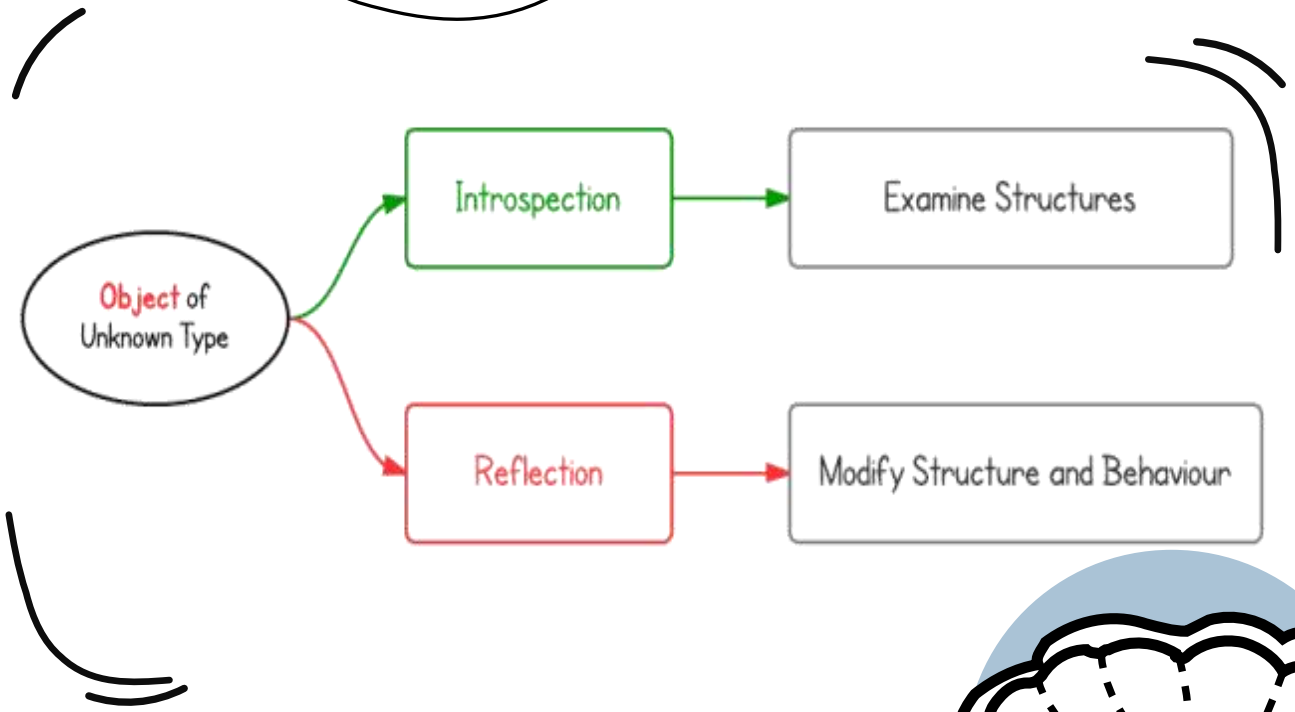
Khái niệm





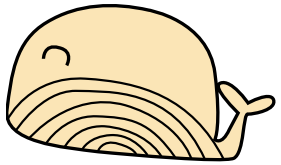
- Java Reflecion cho phép bạn đánh giá, sửa đổi cấu trúc và hành vi của một đối tượng tại thời gian chạy (runtime) của chương trình. Đồng thời nó cho phép bạn truy cập vào các thành viên private (private member) tại mọi nơi trong ứng dụng, điều này không được phép với cách tiếp cận truyền thống.

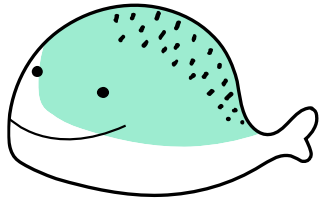
Khái niệm

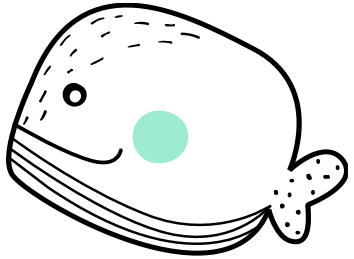


- Java thông thường có thể được gọi là Java Introspection (Nội quan), chương trình có khả năng đánh giá cấu trúc của một đối tượng tại thời gian chạy (Runtime).
- Với Java Reflection, chương trình có khả năng đánh giá cấu trúc của một đối tượng tại thời gian chạy, sửa đổi cấu trúc và hành vi của đối tượng.

2.Ví dụ







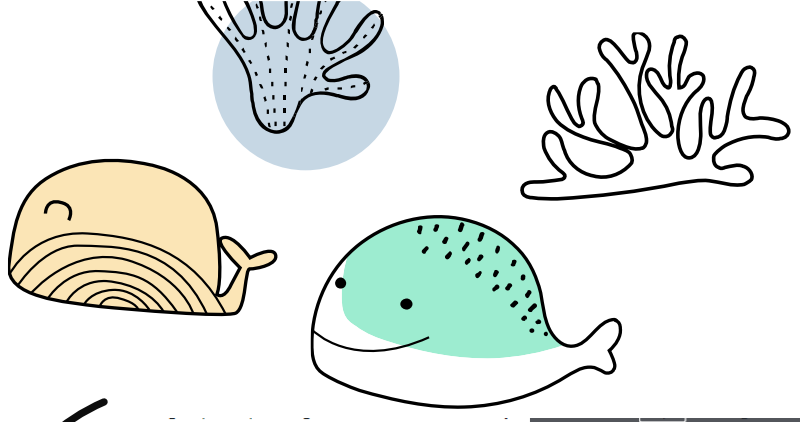
MyEclipse

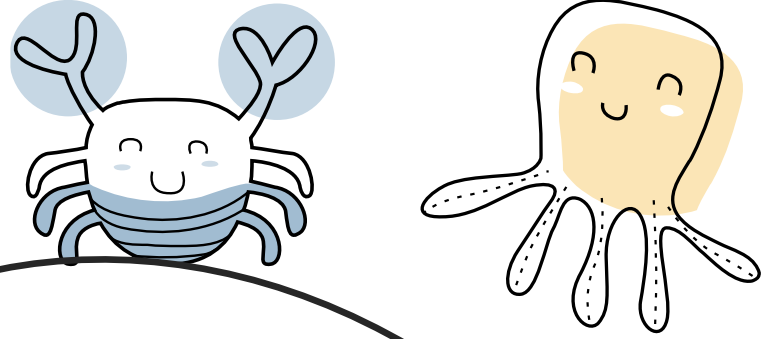
Eclipse

NetBean

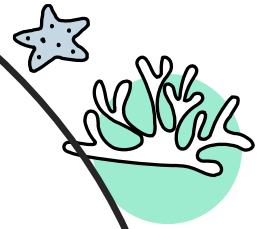
- Reflection API của Java chủ yếu được sử dụng trong:
- IDE như Eclipse, MyEclipse, NetBean,....
- Trình gỡ lỗi.
- Công cụ kiểm tra.

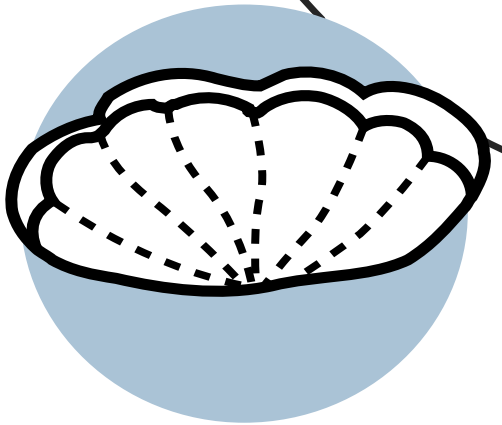
Cú pháp





- Lớp java.lang.Class thực hiện hai nhiệm vụ như sau:
  - Nó cung cấp các phương thức để lấy các siêu dữ liệu của một lớp tại thời gian chạy.
  - Nó cung cấp các phương thức để kiểm tra và thay đổi hành vi của lớp tại thời gian chạy.





Các phương thức thường được sử dụng của lớp Class:

Phương thức	Mô tả
public String getName()	Trả lại tên của lớp
public static Class forName(String className)	Phương thức này tải lớp và trả lại tham chiếu của lớp
public Object newInstance()	Nó tạo nên một đối tượng mới của lớp
public boolean isInterface()	Phương thức này kiểm tra xem liệu nó có phải là một giao diện hay không
public boolean isArray()	Phương thức này kiểm tra xem liệu nó có phải một mảng hay không
public boolean isPrimitive()	Phương thức này kiểm tra xem liệu nó có phải nguyên thủy hay không
public Class getSuperclass()	Nó trả về siêu lớp hoặc tham chiếu lớp cha
public Field[] getDeclaredFields()	Nó trả về tổng số fields trong lớp
public Method[] getDeclaredMethods()	Nó trả lại tổng số phương thức của lớp
public Method getDeclaredMethod(String name,Class[] parameterTypes)	Phương thức này trả lại phương thức lớp instance
public Constructor[] getDeclaredConstructors()	Nó trả lại tổng số hàm tạo của lớp

Cú pháp

Các phương thức thường được sử dụng của lớp Class:

Phương thức	Mô tả
public String getName()	Trả lại tên của lớp
public static Class forName(String className)	Phương thức này tải lớp và trả lại tham chiếu của lớp
public Object newInstance()	Nó tạo nên một đối tượng mới của lớp
public boolean isInterface()	Phương thức này kiểm tra xem liệu nó có phải là một giao diện hay không
public boolean isArray()	Phương thức này kiểm tra xem liệu nó có phải một mảng hay không
public boolean isPrimitive()	Phương thức này kiểm tra xem liệu nó có phải nguyên thủy hay không
public Class getSuperclass()	Nó trả về siêu lớp hoặc tham chiếu lớp cha
public Field[] getDeclaredFields()	Nó trả về tổng số fields trong lớp
public Method[] getDeclaredMethods()	Nó trả lại tổng số phương thức của lớp
public Method getDeclaredMethod(String name,Class[] parameterTypes)	Phương thức này trả lại phương thức lớp instance
public Constructor[] getDeclaredConstructors()	Nó trả lại tổng số hàm tạo của lớp

- **Cách lấy đối tượng của lớp Class của Reflection trong Java**
  - Có ba cách lấy đối tượng của lớp Class như sau:
    - Phương thức của lớp Class forName()
    - Phương thức forName() tải lớp động hoặc trong thời gian chạy. Phương thức này trả về biến của lớp Class. Chúng ta chỉ nên sử dụng phương thức này nếu chúng ta biết tên đạt tiêu chuẩn của lớp đó. Chúng ta không thể sử dụng tên này cho các loại nguyên thủy.

Cú pháp

Các phương thức thường được sử dụng của lớp Class:

Phương thức	Mô tả
public String getName()	Trả lại tên của lớp
public static Class forName(String className)	Phương thức này tải lớp và trả lại tham chiếu của lớp
public Object newInstance()	Nó tạo nên một đối tượng mới của lớp
public boolean isInterface()	Phương thức này kiểm tra xem liệu nó có phải là một giao diện hay không
public boolean isArray()	Phương thức này kiểm tra xem liệu nó có phải một mảng hay không
public boolean isPrimitive()	Phương thức này kiểm tra xem liệu nó có phải nguyên thủy hay không
public Class getSuperclass()	Nó trả về siêu lớp hoặc tham chiếu lớp cha
public Field[] getDeclaredFields()	Nó trả về tổng số fields trong lớp
public Method[] getDeclaredMethods()	Nó trả lại tổng số phương thức của lớp
public Method getDeclaredMethod(String name,Class[] parameterTypes)	Phương thức này trả lại phương thức lớp instance
public Constructor[] getDeclaredConstructors()	Nó trả lại tổng số hàm tạo của lớp

❖ **Phương thức của lớp Object getClass()**

➢ Phương thức getClass() thuộc về lớp Object và trả về biến trong lớp Class. Chúng ta nên sử dụng nó khi chúng ta biết kiểu của nó. Chúng ta cũng có thể sử dụng nó với các biến nguyên thủy.

Cú pháp

Các phương thức thường được sử dụng của lớp Class:

Phương thức	Mô tả
public String getName()	Trả lại tên của lớp
public static Class forName(String className)	Phương thức này tải lớp và trả lại tham chiếu của lớp
public Object newInstance()	Nó tạo nên một đối tượng mới của lớp
public boolean isInterface()	Phương thức này kiểm tra xem liệu nó có phải là một giao diện hay không
public boolean isArray()	Phương thức này kiểm tra xem liệu nó có phải một mảng hay không
public boolean isPrimitive()	Phương thức này kiểm tra xem liệu nó có phải nguyên thủy hay không
public Class getSuperclass()	Nó trả về siêu lớp hoặc tham chiếu lớp cha
public Field[] getDeclaredFields()	Nó trả về tổng số fields trong lớp
public Method[] getDeclaredMethods()	Nó trả lại tổng số phương thức của lớp
public Method getDeclaredMethod(String name,Class[] parameterTypes)	Phương thức này trả lại phương thức lớp instance
public Constructor[] getDeclaredConstructors()	Nó trả lại tổng số hàm tạo của lớp

❖ **Cú pháp .class:**

❖ Đôi khi, có tình huống khi một kiểu có sẵn nhưng không có thể hiện của lớp. Trong những trường hợp như thế, chúng ta có thể lấy lớp bằng cách thêm cú pháp .class vào tên của kiểu. Chúng ta cũng có thể sử dụng các cú pháp này với các nguyên mẫu.

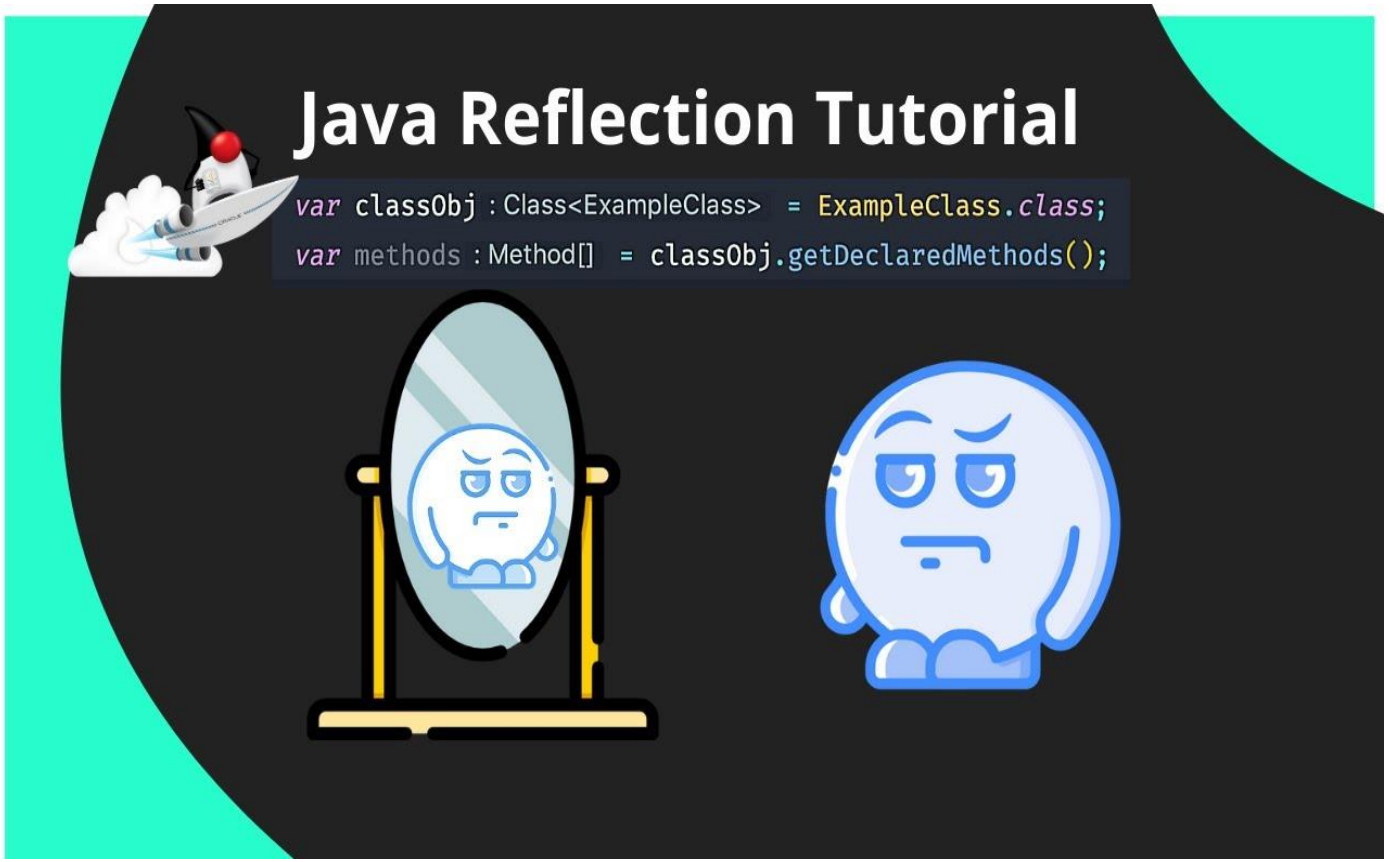
Cú pháp

Các phương thức thường được sử dụng của lớp Class:

Phương thức	Mô tả
public String getName()	Trả lại tên của lớp
public static Class forName(String className)	Phương thức này tải lớp và trả lại tham chiếu của lớp
public Object newInstance()	Nó tạo nên một đối tượng mới của lớp
public boolean isInterface()	Phương thức này kiểm tra xem liệu nó có phải là một giao diện hay không
public boolean isArray()	Phương thức này kiểm tra xem liệu nó có phải một mảng hay không
public boolean isPrimitive()	Phương thức này kiểm tra xem liệu nó có phải nguyên thủy hay không
public Class getSuperclass()	Nó trả về siêu lớp hoặc tham chiếu lớp cha
public Field[] getDeclaredFields()	Nó trả về tổng số fields trong lớp
public Method[] getDeclaredMethods()	Nó trả lại tổng số phương thức của lớp
public Method getDeclaredMethod(String name,Class[] parameterTypes)	Phương thức này trả lại phương thức lớp instance
public Constructor[] getDeclaredConstructors()	Nó trả lại tổng số hàm tạo của lớp

- **Lấy thông tin bằng cách sử dụng Reflection API trong Java**
  - Chúng ta có thể sử dụng Reflection để lấy các thông tin về:
    - Class: Phương thức getClass() cho biết tên của lớp mà đối tượng thuộc về
    - Constructors: Phương thức getConstructors() trả về tất cả các hàm tạo công hai của lớp mà đối tượng thuộc về
    - Methods: Phương thức getMethods () đưa ra tất cả các phương thức chung của lớp mà một đối tượng thuộc về.

3.Những hạn chế của java reflection



❖ **java Reflection còn những hạn chế:**

➢ **Hiệu năng thấp:**

Ví dụ phải quét classpath để tìm class.

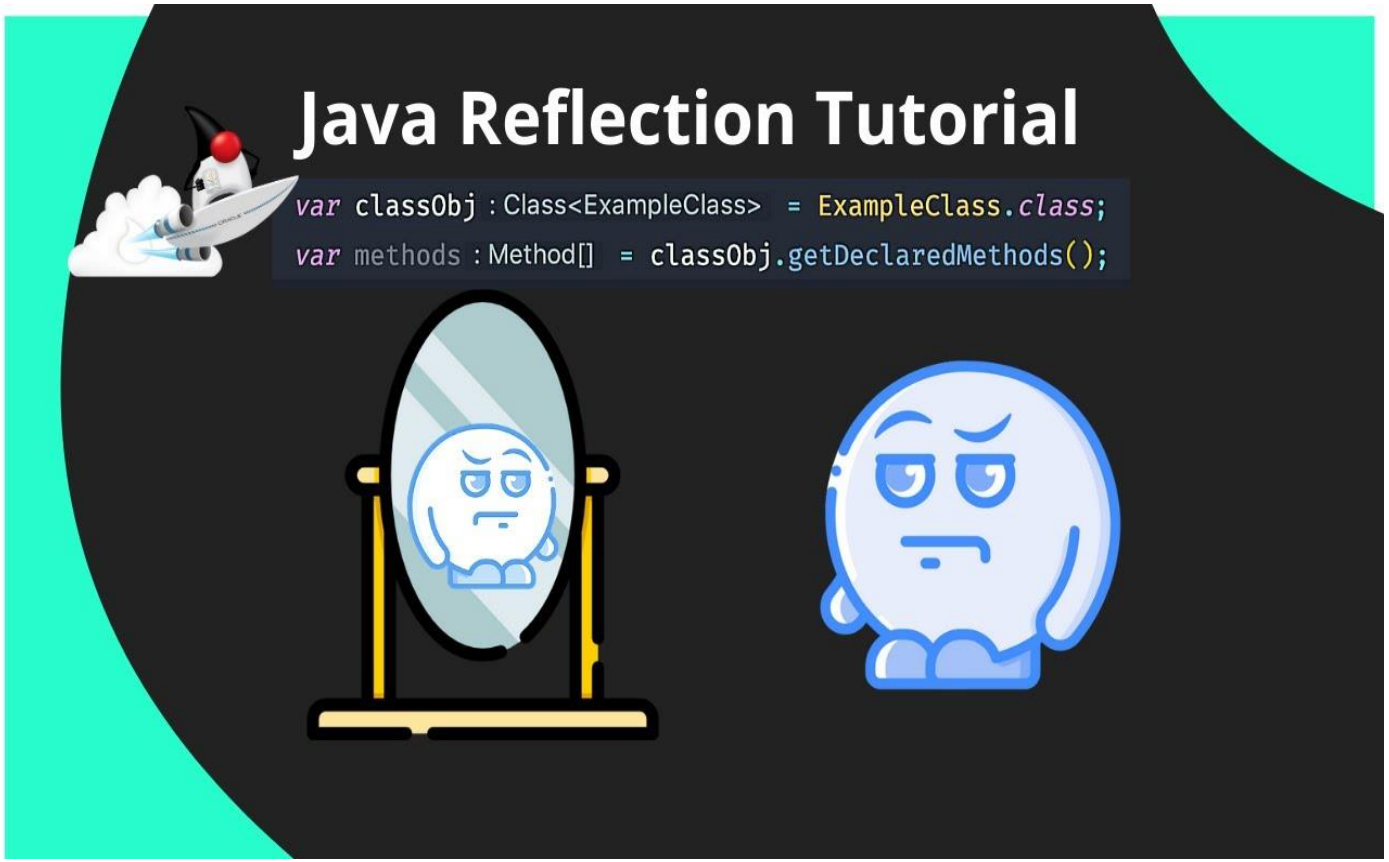
➢ **Các vấn đề bảo mật:**

Việc chỉnh sửa class/object trong quá trình runtime có thể ảnh hưởng tới các thread … khiến cho ứng dụng bị fail.

➢ **Khó bảo trì:**

Việc Reflection khá khó hiểu với người mới và không dễ để debug, nên sẽ rất khó để có thể tìm ra lỗi. Ngoài ra chúng ta cũng không thể check được một số lỗi trong quá trình compile.

4.Một số lưu ý về java reflection



❖ **Một số điểm lưu ý về java reflection:**

- Các lớp cần thiết để phản chiếu trong Java có trong gói java.lang.reflect.
- Reflection cung cấp cho chúng ta các dữ liệu về lớp với các đối tượng liên kết và phương thức cho lớp đó.
- Thông qua sự phản chiếu, chúng ta có thể gọi một phương thức tại thời gian chạy độc lập với trình xác định truy cập của chúng.


Thank you for
listening to group 11