Java

- description : 자바 프로그래밍 관련 내용
- author : 주레피
- email : dhan@repia.com
- lastupdate : 2020-03-09

Java 코딩 규칙

(원문) Oracle / Code Conventions for JavaTM Programming Language / 1999-4-20
(번역본) 자바 프로그래밍 언어 코딩 규칙

상속

1. 자바는 다중 상속을 허용하지 않는다
  따라서 `extends`뒤에는 단 하나의 부모 클래스만 와야 한다.

2. 자식 객체를 생성하면 내부적으로 먼저 부모 객체 부터 생성된다.
  명시적으로 표시가 안되더라도 생성자 내에는 `super();`를 통해 기본 생성자로 부모 객체부터 생성된다.
  명시적으로 부모 생성자를 호출하고자 하면 부모의 생성자를 살펴봐야 한다.
  즉, 부모에 기본 생성자가 없으면 생성해야 하고, 매개 변수를 전달하려면 존재하는 부모 생성자에 맞게 호출해야 한다.
  명시적인 부모 생성자는 첫 줄에 위치해야 한다.
3. 메소드 재정의`(@Override)`
  재정의 할때는 부모의 메소드와 동일한 시그너처(리턴 타입, 메소드 이름, 매개 변수 리스트)를 가져야 하며,
  접근 제한을 더 강하게 오버라이딩할 수 없으며, 새로운 예외를 발생시킬 수 없다.
  접근 제한 순서`(public, protected, default, private)`
   `@Override`를 명시적으로 하여 컴파일러에게 알려주도록 하자. (향후 런타임 오류를 줄여줌)
   `super.`부모메소드()를 사용하면 부모메소드를 접근할 수 있다
4. `final`
   `final class:` 대표적인 `String` 클래스는 `final` 클래스도 상속될 수 없다. 마지막 클래스이다.
   `final method():` 오버라이딩 할 수 없다. 마지막 메소드이다.
5. 다형성`(Polymorphism)`
  자동차를 구성하는 부품은 언제든지 교체할 수 있다. 언제든지 성능이 좋은 다른 객체로 교체할 수 있어야 한다.
  사용방법은 동일하지만 실행 결과는 더 우수해야 된다는 것이다.
6. `instanceof`
  객체의 타입 확인: `if(A클래스 instanceof B클래스)`

7. 추상클래스`(abstract)`
  실체 클래스의 공통되는 필드와 메소드를 추출해서 만든 객체로 직접 생성`(new)`이 불가능하며,
  상속만 가능하다.
  용도는 1. 필드와 메소드의 통일(아니면 팀원들간에 제각각일 것임)

2. 시간이 절약됨

`[public | protected] abstract 리턴타입 메소드명(매개변수, ...);`
추상 메소드로 있어야 되지만 실체는 자식에 있으며, 반드시 자식은 `@Override`해야 된다.

접근 제한자

접근 제한	적용할 내용	접근할 수 없는 클래스
public	클래스, 필드, 생성자 메소드	없음
protected	필드, 생성자, 메소드	자식 클래스가 아닌 다른 패키지에 소속된 클래스
default	클래스, 필드, 생성자 메소드	다른 패키지에 소속된 클래스
private	필드, 생성자, 메소드	모든 외부 클래스

자바(프로그래밍 언어)

- Batch()_여러 레코드 입력시_DB커넥트 관련
- JDBC
- Math
- 위키 백과
- Jenkins(젠킨스)를 활용한 스프링부트 앱 간단 배포하기

프로그래밍

- get, post로 자료 가져오기
- ibatis to mybatis 전자정부프레임워크
- jackson 사용방법
- jsoup 사용방법
- List에서 마지막 요소 얻는 방법
- Regular Expression
- 멀티 프로세서에서 데이터 중복 처리 방지
- 주석을 달아 보자
- 메일 보내기
- 배열에 특정 값이 존재하는지 체크 / How to check whether an array contains a particular value in java

상대경로를 절대경로로 변환하는 방법(Javascript)

상대경로는 반드시 `base URL`이 있다는 것을 명심하자.

```
function absolute(base, relative) {
    var stack = base.split("/"),
        parts = relative.split("/");
    stack.pop(); // remove current file name (or empty string)
                 // (omit if "base" is the current folder without trailing slash)
    for (var i=0; i<parts.length; i++) {
        if (parts[i] == ".")
            continue;
```

```
        if (parts[i] == "..")
            stack.pop();
        else
            stack.push(parts[i]);
    }
    return stack.join("/");
}
```

보안

- XSS injection prevention

Apache-commons

- commons-lang
  - StringUtils
- commons-io
  - FileUtils
- commons-codec
  - Base64

Boolean vs boolean

`boolean` 은 자료형이기 때문에 오직 `true, false`만 들어갈 수 있고 `null`은 못 넣는다
`null`을 넣기 위해선 바로 참조형인 `Boolean`으로 적어야 하는 것이었다.
같은 맥락으로 `int`에 `null`을 못 넣고 `Integer`에 `null`을 넣을 수 있는 것처럼 말이다.

Cast

Convert double to float in Java

```
double d = getInfoValueNumeric();
float f = (float)d;

// Also notice that the primitive types can NOT store an infinite set of
numbers:

// float range: from 1.40129846432481707e-45 to 3.40282346638528860e+38
// double range: from 1.7e−308 to 1.7e+308
```

Date

- 날짜 관련

- 두 날짜 사이의 차이 계산 방법
- Date java 방금전, 몇분전, 몇시간전, 몇달전 표기방법

enum

```
열거 타입
Naming 첫 문자를 대문자로 하기로 함(Week.java, Role.java, ...) // Camel case적용
열거 상수는 모두 대문자로 하기로 함(MONDAY, ROOT, ...) // Snake case 적용

Calendar.class를 생각해 보라~
알듯, 모를듯, 하지만 알아야 하는 재미진 타입...
```

> TableStatus.java

```java
// public enum <- 소문자에 주의
// 생성은 열거 상수, 멤버, 생성자, 게터
public enum TableStatus
{
  Y("1", true)
  , N("0", false);

  private String table1Value;
  private boolean table2Value;

  TableStatus(String table1Value, boolean table2Value)
  {
    this.table1Value = table1Value;
    this.table2Value = table2Value;
  }

  public String getTable1Value()
  {
    return table1Value;
  }

  public boolean isTable2Value() { return table2Value; }
}
```

> 메소드

- values()
  - 열거된 모든 원소를 배열에 담아 순서대로 반환
- ordinal()
  - 원소에 열거된 순서를 정수 값으로 반환
- valueOf()
  - 매개변수로 주어진 String과 열거형에서 일치하는 이름을 갖는 원소를 반환
  - 주어진 String과 일치하는 원소가 없는 경우 IllegalArgumentException 예외 발생

Java Enum 활용기
enum 클래스의 사용과 이해

Format

- 포맷 관련
- JAVA/ Float 소수점 자리 출력 방법

Library

- 

Log

logging framework logback

Map

- How to Convert List to Map in Java

Optional

`Optional`는 “존재할 수도 있지만 안 할 수도 있는 객체”, 즉 “`null`이 될 수도 있는 객체”을 감싸고 있는 일종의 `WRAP`클래스입니다.
원소가 없거나 최대 하나 밖에 없는 `Collection`이나 `Stream`으로 생각하셔도 좋습니다.
직접 다루기에 위험하고 까다로운 `null`을 담을 수 있는 특수한 그릇으로 생각하시면 이해가 쉬우실 것 같습니다.

Optional.ofNullable(value)

`null`인지 아닌지 확신할 수 없는 객체를 담고 있는 `Optional` 객체를 생성합니다.

`Optional.empty()`와 `Optional.ofNullable(value)`를 합쳐놓은 메소드라고 생각하시면 됩니다.
`null`이 넘어올 경우, `NPE`를 던지지 않고 `Optional.empty()`와 동일하게 비어 있는 `Optional` 객체를 얻어옵니다.
해당 객체가 `null`인지 아닌지 자신이 없는 상황에서는 이 메소드를 사용하셔야 합니다.

자바8 Optional 2부: null을 대하는 새로운 방법

Reflection

- Call Methods at Runtime Using Java Reflection

Servlet, Request, Context

- Servlet, Request, Context 관련

String

- replaceAll, replace

Thread

- 멀티 스레드 병렬 프로그래밍을 하기 전 반드시 읽어야할 것들

예외처리 - try{} catch(){} finally{}

- e.toString(), e.getMessage(), e.printStackTrace()의 차이점
- Spring Guide - Exception 전략

Ref

- concurrency
- fcntl을 이용한 파일 및 레코드 잠금

주레피, java, doc2vec

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다.** - 2023.12

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:java&rev=1599547027

Last update: **2022/03/10 19:52**