Annotation

Annotation

- **Java 5 부터 추가된 기능**
- **클래스, 메소드 위에 붙고, @(at) 기호로 이름이 시작 (@Override)**
- **클래스나 메소드에 메타코드(추가정보)를 주는 것**
- **클래스가 컴파일 되거나 실행될 때, 어노테이션 유무, 어노테이션에 설정된 값을 통해 클래스의 동작을 다르게 할 수 있다.**
  (어노테이션을 설정파일처럼 설명하는 경우도 많다.)

- **메타 데이터 (Meta Data)**
  다른 데이터를 설명해주고 추가적인 정보를 제공하는 데이터이다.

- **사용용도**
  - 주석
  - 코드를 자동으로 생성하여 추가 후 컴파일
  - 컴파일러에 정보제공
  - 런타임의 추가 기능실행

- **Built - in Annotation**
  Java가 기본적으로 제공하는 Annotation

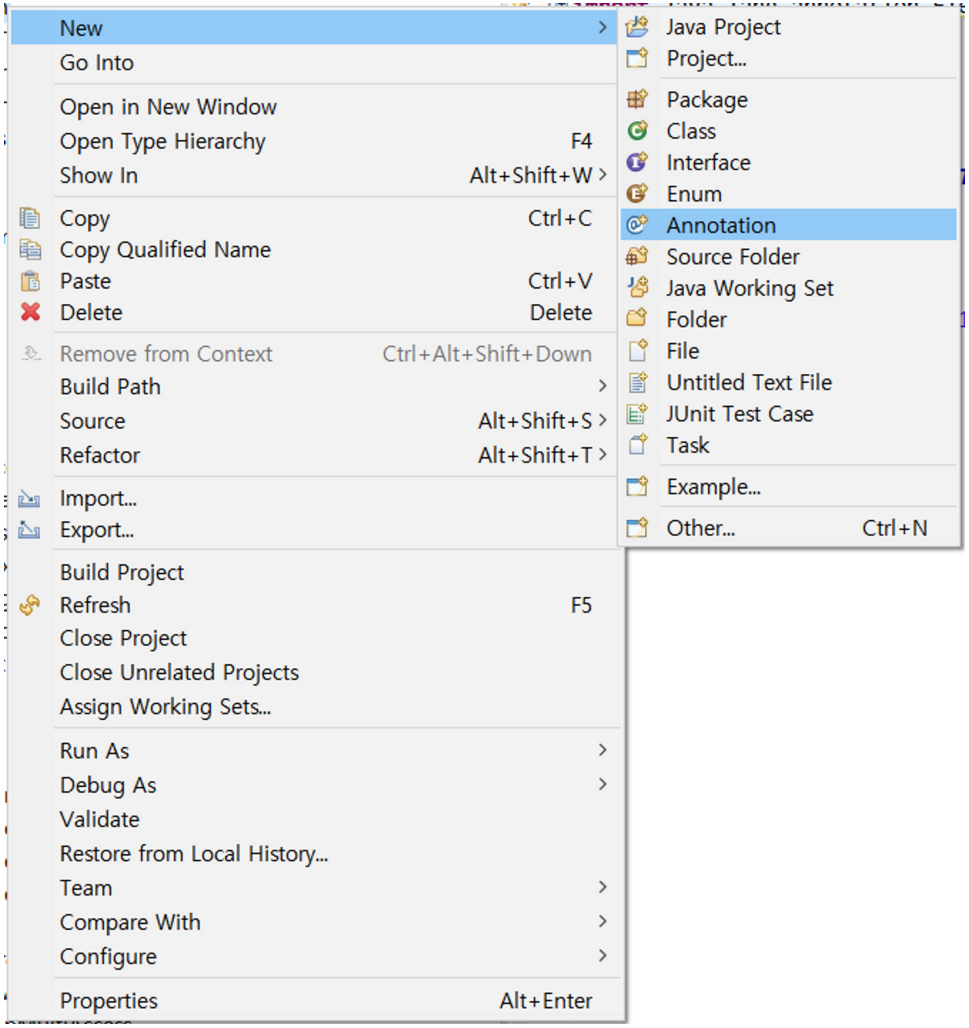
  - **@Override 메소드가 오버라이딩 되었는지 검증, 상위 클래스나 인터페이스에서 해당 메소드를 찾지 못하면 컴파일 오류**
  - **@Deprecated 메소드를 사용하지 않도록 유도, 사용시 컴파일 경고**
  - **@SuppressWarnings 컴파일 경고 무시**
  - **@SafeVarargs 제네릭과 같은 가변인자 매개변수을 사용할 때 컴파일 경고 무시 (Up to Java 7)**
  - **@FunctionalInterface 람다 함수등을 위한 인터페이스 지정, 메소드가 없거나 두 개 이상이 되면 컴파일 오류 (Up to Java 8)**

- **Meta Annotation**
  Custom Annotation을 만들기 위한 Meta Annotation

- **@Retention 어노테이션의 범위, 어떤 시점까지 어노테이션이 영향을 미치는지 결정**
- **@Documented 문서에 어노테이션 정보 표시**
- **@Target 어노테이션의 적용 위치 결정**
- **@Inherited 상위 클래스에서 어노테이션 상속 가능**
- **@Repetable 반복적으로 어노테이션 선언 가능**

Custom Annotation 정의하기

i. Custom Annotation 정의하기




```
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

// 적용대상
// @Target(ElementType.METHOD) // Constructor, Local Variable, Method, Package, Parameter, 타입(클래스)
// @Target 미지정시 다 쓸수있다.
@Retention(RetentionPolicy.RUNTIME) // 에너테이션에 세팅된 정보를 유지하는 단계 : 소스코드, 클래스, 런타임(VM
```

```
Engine)

public @interface Annotation {
     String author();
     String date();
     String version() default "1.0";
}
```

i. Custom Annotation을 타겟에 적용 및 실행

```
@Annotation(author="michael", date="2016.10.27") // Target에 Annotation 적용
public class AnnoMain {
     public static void main(String[] args) {
          AnnoMain annoMain = new AnnoMain();

          // 리플렉션을 이용한 어노테이션 정보 추출
          Class<?> klass = annoMain.getClass();
          Annotation anno = klass.getAnnotation(Annotation.class);

          StringBuffer buffer = new StringBuffer();

          if(anno != null){
               buffer.append(anno.author()).append(anno.date()).append(anno.version());
          }

          annoMain.print(buffer.toString());

          // isAnnotationPresent(Annotation.class) 어노테이션 적용 유무검사 메소드
          if(klass.isAnnotationPresent(Annotation.class))
               System.out.println("어노테이션 적용됨");
          else
               System.out.println("어노테이션 적용 안됨");

     }

     public void print(String str){
          System.out.println(str);
```

```
        }
    }
```

Android Annotations 사용법

build.gradle에 아래와 같은 코드를 추가. Android Studio의 경우에는 sync을 하면 사용이 가능.

```
dependencies {
   compile 'com.android.support:support-annotations:20.0.0'
}
```