



7_21_2022__java_

MainClass01.java

~MainClass08.java

Eclipse 편집기 사용

환경 바꾸는 버튼

java ee (default)

지금 사용할 건 java



preferences - encoding - other (UTF-8)

```
public static void main(String[] args) {  
    |  
}  
}
```

- method
 - javascript의 함수와 비슷(function main(args){})

```
package test.main;  
  
public class MyClass {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!!");  
    }  
}
```

예약어

```
package test.main;

public class MyClass {
    public static void main(String[] args) {
        System.out.println("Hello, World!!");
    }
}
```

public class static void ...

메인 메소드(시작점 역할)

```
public static void main(String[] XXXX) {

}
```

```
//이 클래스가 어느 package에 속해 있는지 package 예약어를 이용해서 명시
package test.main;

//my class 라는 이름의 클래스 정의하기
public class MyClass {
    //
    // run 버튼을 눌렀을 때 실행순서가 시작되는 특별한 main method
    public static void main(String[] args) {
        //콘솔창에 문자열 출력하기
        System.out.println("Hello, World!!");
    }
}
```

- file 의 이름과 class의 이름이 같아야 한다.
- project 하나당 app 하나 - 도입점(main)이 하나여야 한다.
- js와 java의 차이

```
package test.main;

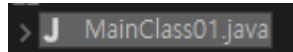
public class MainClass {
    public static void main(String[] args) {
        System.out.println("main method가 시작 되었습니다.");
        //let num1=10;
        int num1=10;
    }
}
```

```
//let num2=10.1
double num2=10.1;
//let isMale=true;
boolean isMale=true;
//let myName="두부"
String myName="두부";
} //String은 예약어가 아님
}
```

- js동적언어 java정적언어
 - java가 더 엄격 하다
 - 사용자 정의 datatype도 지정 가능
- 예약어가 아닌 타입에 id가 주어진다.



자바 text type



- [Java 기본 데이터 type] 8가지
- 기본 데이터 type 안에는 실제 그 값이 들어있다.
소문자로 시작함.

- 숫자형
 - 정수형(4) : byte, short, int, long
 - byte 변수명; // -128 ~ 127
 - short 변수명; // -32768 ~ 32767
 - int 변수명; // -2,147,483,648 ~ 2,147,483,647
 - long 변수명; // -9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807
 - 실수형(2) : float, double
 - float 변수명; // 1.40129846432481707e-45 ~ 3.40282346638528860e+38

- double 변수명; //4.94065645841246544e-324d ~ 1.79769313486231570e+308d
- 표현 가능 범위가 좁은 변수에 담긴 값을 표현 가능 범위가 더 넓은 변수에 대입하는 것은 문제가 없다
- 표현 가능한 범위가 더 넓은 변수에 담긴 내용을 대입하려면 명시적으로 casting 을 해야 한다.
(type)은 type casting 연산자 이다.
- 정수와 정수를 연산하면 정수만 나오기 때문에
*정확한 실수 값을 얻어내기 위해서는 연산에 참여하는 숫자중 적어도 하나는 실수가 되어야 한다
-
- 논리형(boolean)
 - true, false 두가지 값중 하나이다.
 - true, false 를 직접 쓰거나 비교연산 혹은 논리 연산의 결과로 얻어낼 수 있다.
 - 참과 거짓을 나타내는 데이터 type

```
public class MainClass03 {
    //run 했을때 실행의 흐름이 시작되는 특별한 main method(default method)
    public static void main(String[] args) {
        //논리형 변수 선언과 동시에 값 대입하기
        boolean isRun=true;
        if(isRun) {
            System.out.println("달려요");
        }

        //비교 연산의 결과로 얻어진 boolean 값 담기
        boolean isGreater=10>1;
        //논리; 연산의 결과로 얻어진 boolean 값 담기
        boolean result=true||false;

        //한번 선언한 변수는 다시 선언 할 수 없다.
        //boolean result =false;
        result=false; // 변수에 다른 값을 대입 할 수 있다.(동일한 type 인 경우)
        //다른 type 값은 변수에 담을 수 없다.
        //result=10;
    }
}
```

- 문자형(char)

- 65536 가지의 코드값을 가질 수 있다.
- 전 세계에서 사용하는 모든 문자 1글자를 표현할 수 있다.
- single quotation 을 이용해서 만든다.

```
public class MainClass04 {
    public static void main(String[] args) {
        System.out.println("main method가 시작 되었습니다.");
        //char 형 변수 선언과 동시에 값 대입하기
        char ch1='a';
        char ch2='b';
        char ch3='c';
        char ch4='가';
        char ch5='나';
        char ch6='다';
        char ch7='@';

        //정수값에 1:1 대응되는 char 값이 존재한다.
        int code1=ch1;
        int code2=ch2;
        int code3=ch3;
        int code4=ch4;
        int code5=ch5;
        int code6=ch6;
        int code7=ch7;

    }
}
```

◦ String Type

- 무자열을 다룰 때 사용하는 데이터 type

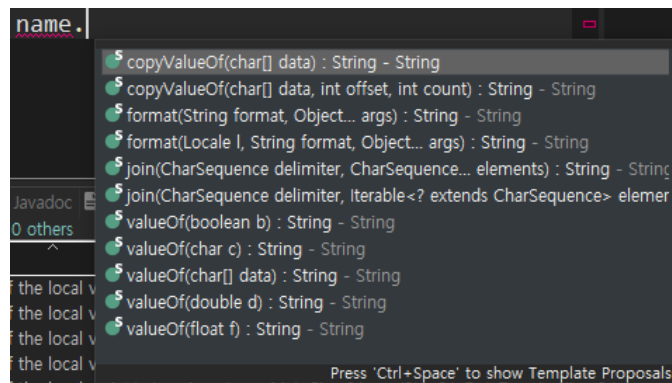
```
public class MainClass05 {
    public static void main(String[] args) {
        //"두부" 라는 String type 객체를 heap 영역에 만들고 그 참조값을 변수에 담기
        String name="두부";
        //name 안에 있는 참조값을 tmp 변수에 복사해 주기
        String tmp=name;
        //"유키"라는 String type 객체를 heap 영역에 만들고 그 참조값을 name 변수에 덮어쓰기
        name="유키";
        // name 변수를 비우기 (null 은 참조 데이터 type 이 담길 수 있는 빈공간을 의미)
        name=null;

    }
}
```

객체

- 어떤 값의 저장소(여러개일 수 도 있음)
- 기능(여러개일 수 도 있음)
- 사물함에 들어있는 물건가

자바에서 “.” 은 저장소 혹은 기능을 찾아간다
name. (String type의 기능들이 보인다.)



지역변수

- method 안에서 만드는 변수를 지역변수(local variable)라 한다.
- java에는 전역변수가 없다 local variable(지역변수) or field
- js와의 비교
 - js에서 전역 변수를 만들고 아무 값도 만들지 않아도 undefined가 들어 있는 상태로 만들어진다.
 - java 에서 값을 지정해 주지 않으면 만들 준비만 되고 만들어지지는 않는다.(값을 주는 시점에 만들어진다.)

```
System.out.println("main method 가 시작되었습니다.");

    int num1;
    String name1;

    //아직 만들어 지지 않았기 때문에 num1 은 참조 불가
    //int result =10+num1;

    //아직 만들어 지지 않았기 때문에 name1은 참조 불가
    //System.out.println("name1:"+name1);

    System.out.println("main method 가 종료됩니다.");
```

- 지역변수를 미리 만들고 필요한 값을 나중에 넣고 싶으면 초기값을 대입하는 것이 좋다.

- 안좋은 예

```
//지역변수를 미리 만들고 필요한 값을 나중에 넣고 싶으면 초기값을 대입하는 것이 좋다.

int num1;

//필요시에 값 넣기
num1=999;
```

- 좋은 예

```
//지역변수를 미리 만들고 필요한 값을 나중에 넣고 싶으면 초기값을 대입하는 것이 좋다.

int num1=0;
//참조 데이터 type 이 담길 수 있는 빈 공간은 null을 대입하면 된다.
String name1=null;

//필요시에 값 넣기
num1=999;
name1="두부";
```

javascript 에서 함수는 혼자서도 생존할 수 있지만, java에서는 method가 혼자서만 생존할 수 없다. 객체가 필요하다

JAVA는 객체 지향 언어 field들과 method들을 활용해 문제를 해결한다.

. 찍으면 나오는 리스트에서 어떻게 사용 할 수 있는지 알려준다.

- java 에서 기본 데이터 type 8개를 제외한 나머지는 모두 참조 데이터 type이다.
- 참조 데이터 type 객체는 heap영역(사물함 영역) 에 만들어 진다.
- 모든 객체는 저장소(field) 와 기능(method) 로 구성되어 있을 수 있다.
- 어떤 저장소와 어떤 기능을 가지게 될 지는 해당 type 객체가 어떻게 설계되었냐에 따라 다르다.
- 참조값에 . 을 찍으면 저장소를 참조하거나, 기능을 사용할 수 있다.
- 참조값에 . 을 찍으면 field를 참조하거나 method를 호출 할 수 있다.

- 메소드 호출은 javascript 에서 함수 호출과 유사하다.

오늘 할일

- 연습용 project 만들기(project 만드는 연습)
- 만든 project의 src 에 패키지 만들기(test.main ...)
- 만든 package에 class 만드는 연습(class명의 첫글자는 반드시 대문자로 시작)
- 만든 class에 main() method 만드는 연습
- main() method 안에서 콘솔창에 원하는 문자열 출력하는 연습
- main() method 안에서 다양한 type의 지역변수 만드는 연습
- 만든 지역 변수에 값이 잘 들어가는지 확인하는 debug 연습