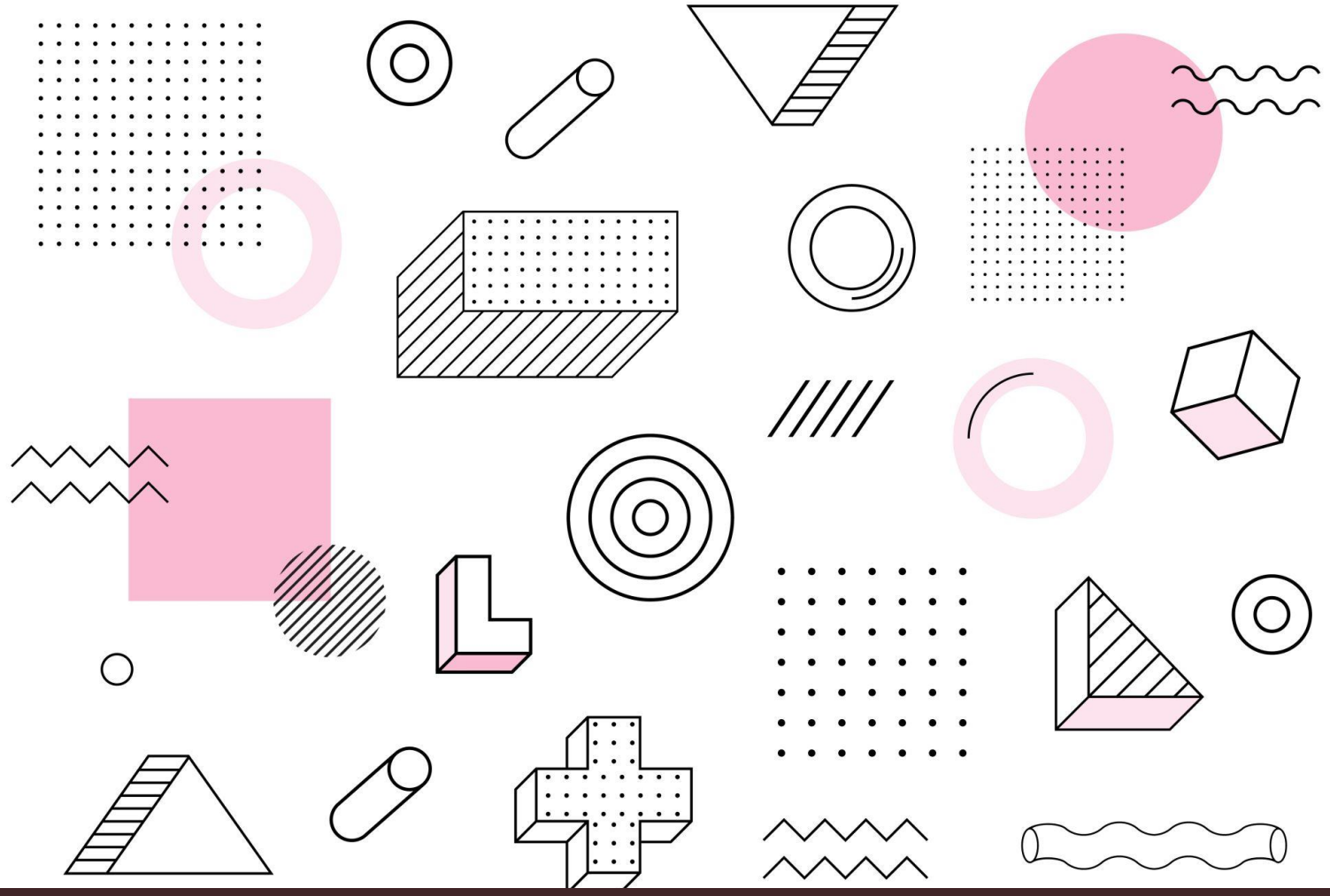

1학기 고급자바 실습

week 8-1(수업 10주차 실습)

김민진(18)

김지희(18)

문의 메일 : genie02166@duksung.ac.kr

Part1. 스레드

1) 작업 스레드 생성과 실행

- 클래스로부터 직접 생성
- 하위 클래스로부터 생성

2) 동기화 메소드 & 블록

3) 스레드 상태 제어

- sleep()
- yield()
- join()
- wait(), notify(), notifyAll()

1) 작업 스레드 생성과 실행

- 클래스로부터 직접 생성

BeepPrintExa... | BeepTask.java | BeepPrintExa... | CalcThread.java

```java
package week8_1;
import java.awt.Toolkit;

public class BeepTask implements Runnable {
    public void run() {
        Toolkit toolkit = Toolkit.getDefaultToolkit();
        for(int i=0; i<5; i++) {
            toolkit.beep();
            try { Thread.sleep(500); } catch(Exception e) {}
        }
    }
}
```

1) 작업 스레드 생성과 실행

- 클래스로부터 직접 생성

BeepPrintExampl... | BeepPrintExampl... | CalcThread.java | PriorityExample.j...

```java
package week8_1;
import java.awt.Toolkit;

public class BeepPrintExample2 {
    public static void main(String[] args) {
        //how1
        Runnable beepTask = new BeepTask();
        Thread thread = new Thread(beepTask);

        //how2
        /*Thread thread = new Thread(new Runnable() {
            @Override
            public void run() {
                Toolkit toolkit = Toolkit.getDefaultToolkit();
                for(int i=0; i<5; i++) {
                    toolkit.beep();
                    try { Thread.sleep(500); } catch(Exception e) {}
                }
            }
        });*/

        //how3
        /*Thread thread = new Thread(() -> {
            Toolkit toolkit = Toolkit.getDefaultToolkit();
            for(int i=0; i<5; i++) {
                toolkit.beep();
                try { Thread.sleep(500); } catch(Exception e) {}
            }
        });*/

        thread.start();

        for(int i=0; i<5; i++) {
            System.out.println("띵");
            try { Thread.sleep(500); } catch(Exception e) {}
        }
    }
}
```

실행 결과

Problems @ Javado

<terminated> BeepPrin

띵
띵
띵
띵
띵

1) 작업 스레드 생성과 실행

- 하위 클래스로부터 생성

BeepThread.java | BeepPrintExa... | CalcThread.java | PriorityExa...

```java
package week8_1;
import java.awt.Toolkit;

public class BeepThread extends Thread {
    @Override
    public void run() {
        Toolkit toolkit = Toolkit.getDefaultToolkit();
        for(int i=0; i<5; i++) {
            toolkit.beep();
            try { Thread.sleep(500); } catch(Exception e) {}
        }
    }
}
```

1) 작업 스레드 생성과 실행

- 하위 클래스로부터 생성

BeepPrintExa... | BeepPrintExa... | CalcThread.java | PriorityExa... | Calcula

```java
package week8_1;
import java.awt.Toolkit;

public class BeepPrintExample3 {
    public static void main(String[] args) {
        //how1
        Thread thread = new BeepThread();

        //how2
        /*Thread thread = new Thread() {
            @Override
            public void run() {
                Toolkit toolkit = Toolkit.getDefaultToolkit();
                for(int i=0; i<5; i++) {
                    toolkit.beep();
                    try { Thread.sleep(500); } catch(Exception e) {}
                }
            }
        };*/


        thread.start();

        for(int i=0; i<5; i++) {
            System.out.println("띵");
            try { Thread.sleep(500); } catch(Exception e) {}
        }
    }
}
```

실행 결과

Problems | @ Javad

<terminated> BeepPri

띵
띵
띵
띵
띵

2) 동기화 메소드&블록

Calculator.java

```java
package week8_2;

public class Calculator {
    private int memory;

    public int getMemory() {
        return memory;
    }

    public synchronized void setMemory(int memory) {
        this.memory = memory;
        try {
            Thread.sleep(2000);
        } catch(InterruptedException e) {}
        System.out.println(Thread.currentThread().getName() + ": " +  this.memory);
    }
}
```

User1.java | User2.java | Calculator.java | MainThreadExample.

```java
package week8_1;

public class User1 extends Thread {
    private Calculator calculator;

    public void setCalculator(Calculator calculator) {
        this.setName("User1");
        this.calculator = calculator;
    }

    public void run() {
        calculator.setMemory(100);
    }
}
```

User2.java | Calculator.java

```java
package week8_1;

public class User2 extends Thread {
    private Calculator calculator;

    public void setCalculator(Calculator calculator) {
        this.setName("User2");
        this.calculator = calculator;
    }

    public void run() {
        calculator.setMemory(50);
    }
}
```

2) 동기화 메소드&블록

MainThreadExample.java | User1.java | User2.java | Calc

```java
package week8_1;

public class MainThreadExample {
    public static void main(String[] args) {
        Calculator calculator = new Calculator();

        User1 user1 = new User1();
        user1.setCalculator(calculator);
        user1.start();

        User2 user2 = new User2();
        user2.setCalculator(calculator);
        user2.start();
    }
}
```

실행 결과

Problems @ Javadoc

<terminated> MainThreadEx

```
User1: 100
User2: 50
```

S-키워드 없을 경우
실행 결과

Problems @ Javadoc

<terminated> MainThreadE

```
User1: 50
User2: 50
```

3) 스레드 상태 제어

- wait(), notify(), notifyAll()

WorkObject.java | ThreadA.java | ThreadB.java | WaitNotifyExample.java

```java
package week8_1_2;

public class WorkObject {
    public synchronized void methodA() {
        System.out.println("ThreadA의 methodA() 작업 실행");
        notify();
        try {
            wait();
        } catch (InterruptedException e) {
        }
    }

    public synchronized void methodB() {
        System.out.println("ThreadB의 methodB() 작업 실행");
        notify();
        try {
            wait();
        } catch (InterruptedException e) {
        }
    }
}
```

3) 스레드 상태 제어

- wait(), notify(), notifyAll()

WorkObject.java | ThreadA.java | ThreadB.java | WaitN

```java
package week8_1_2;

public class ThreadA extends Thread {
    private WorkObject workObject;

    public ThreadA(WorkObject workObject) {
        this.workObject = workObject;
    }

    @Override
    public void run() {
        for(int i=0; i<10; i++) {
            workObject.methodA();
        }
    }
}
```

WorkObject.java | ThreadA.java | ThreadB.java | Wait

```java
package week8_1_2;

public class ThreadB extends Thread {
    private WorkObject workObject;

    public ThreadB(WorkObject workObject) {
        this.workObject = workObject;
    }

    @Override
    public void run() {
        for(int i=0; i<10; i++) {
            workObject.methodB();
        }
    }
}
```

3) 스레드 상태 제어

- wait(), notify(), notifyAll()

WorkObject.java | ThreadA.java | ThreadB.java | WaitNotifyExample.java

```java
package week8_1_2;

public class WaitNotifyExample {
    public static void main(String[] args) {
        WorkObject sharedObject = new WorkObject();

        ThreadA threadA = new ThreadA(sharedObject);
        ThreadB threadB = new ThreadB(sharedObject);

        threadA.start();
        threadB.start();
    }
}
```

실행 결과

Console | Problems | Javadoc

WaitNotifyExample (1) [Java Application]

```
ThreadA의 methodA() 작업 실행
ThreadB의 methodB() 작업 실행
ThreadA의 methodA() 작업 실행
ThreadB의 methodB() 작업 실행
ThreadA의 methodA() 작업 실행
```

출석 과제 (5/17 월 오후 11:55 마감)

Q. 스레드에 대한 설명 중 틀린 것은?

1) 자바 어플리케이션은 메인 스레드가 메인 메소드를 실행시킨다.
2) 작업 스레드 클래스는 Thread 클래스를 상속해서 만들 수 있다.
3) Runnable 객체는 스레드가 실행해야 할 코드를 가지고 있는 객체라고 볼 수 있다.
4) 스레드 실행을 시작하려면 run() 메소드를 호출해야 한다.

위의 기간까지 고급 자바 실습
DS-CLASS 에 제출하시기 바랍니다.