

Chapter 05

교착 상태와 기아 상태

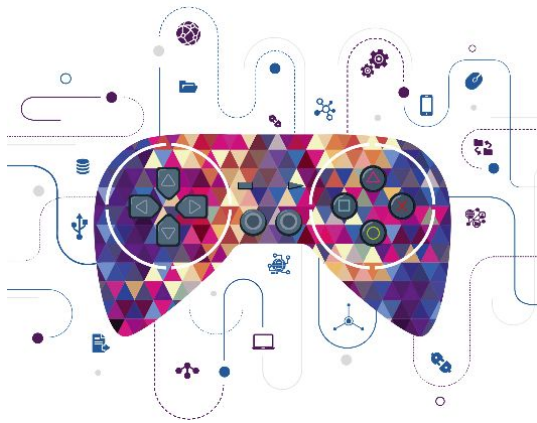
01 교착 상태의 개념과 발생 원인

02 교착 상태의 해결 방법

03 기아 상태

요약

연습문제



그림으로 배우는 구조와 원리

운영체제

개정 3판

쓰레드에서 프로그래밍 실습

■ 상황

멀티 쓰레드에서 공유 자원의 상호 배제를 해야하는 이유를 실습
Gold 를 플레이어가 사냥을 할때 10골드씩 모을수 있음
플레이어는 사냥후 물약을 구매하며 1골드를 사용
100번의 사냥과 100번의 물약을 구매를 했다면 결과는?

■ 모범답은..

- $100 \times 10 - 100 \times 1 = 900$ 골드

쓰레드에서 프로그래밍 실습

```
1  using System;
2  using System.Threading;
3
4  namespace ThreadRacing
5  {
6      class Program
7      {
8          public static int gold;
9
10         public static void Main(string[] args)
11         {
12             Program.gold = 0;
13             Thread thread0 = new Thread(() => Hunt());
14             thread0.Start();
15             Thread thread1 = new Thread(() => BuyItem());
16             thread1.Start();
17
18             Console.WriteLine(string.Format("골드 양 {0}", Program.gold));
19         }
20     }
21 }
22
```

쓰레드에서 프로그래밍 실습

```
23 public static void Hunt()  
24 {  
25     Console.WriteLine(string.Format("사냥 시작!"));  
26  
27     for (int i = 0; i < 100; i++)  
28     {  
29         Program.gold += 10;  
30         Console.WriteLine(string.Format("현재 골드 {0}", Program.gold));  
31         Thread.Sleep(10);  
32     }  
33     Console.WriteLine(string.Format("사냥 종료!"));  
34 }  
35  
36  
37 public static void BuyItem()  
38 {  
39     Console.WriteLine(string.Format("아이템구매 시작!"));  
40  
41     for (int i = 0; i < 100; i++)  
42     {  
43         Program.gold -= 1;  
44         Console.WriteLine(string.Format("소비한 골드 {0}", Program.gold));  
45         Thread.Sleep(10);  
46     }  
47     Console.WriteLine(string.Format("아이템 구매 종료!"));  
48 }  
49  
50 }
```

쓰레드에서 프로그래밍 실습

C:\WINDOWS\system32\cmd.exe

```
골드 양 0
아이템구매 시작!
사냥 시작!
현재 골드 9
소비한 골드 -1
소비한 골드 18
현재 골드 18
소비한 골드 17
현재 골드 27
현재 골드 36
소비한 골드 26
현재 골드 46
소비한 골드 45
현재 골드 55
소비한 골드 54
소비한 골드 63
현재 골드 63
소비한 골드 72
현재 골드 73
소비한 골드 71
현재 골드 71
소비한 골드 80
현재 골드 81
현재 골드 89
소비한 골드 89
현재 골드 98
소비한 골드 98
소비한 골드 107
현재 골드 107
소비한 골드 106
```

C:\WINDOWS\system32\cmd.exe 선택

```
소비한 골드 774
현재 골드 784
소비한 골드 783
현재 골드 793
소비한 골드 792
현재 골드 802
소비한 골드 801
현재 골드 811
소비한 골드 810
현재 골드 820
소비한 골드 819
현재 골드 829
소비한 골드 828
현재 골드 838
소비한 골드 837
현재 골드 847
소비한 골드 846
현재 골드 856
소비한 골드 855
현재 골드 865
소비한 골드 864
현재 골드 874
소비한 골드 873
현재 골드 883
소비한 골드 882
현재 골드 892
소비한 골드 891
사냥 종료!
아이템 구매 종료!
계속하려면 아무 키나 누르십시오 . . .
```

레이스 컨디션! (두개 이상의 프로세스나 쓰레드가 서로 경쟁을 하면서 Gold를 덮어 써서 동기화가 안되는 문제가 발생!)

레이스 컨디션 해결 방법!

■ 원자 단위 연산

- 원자단위 연산에는 일반적으로 다음과 같은 종류가 있다.
- 1만큼 증가(Increment)
- 1만큼 감소(Decrement)
- 원하는 만큼 더해서 대입
- 원하는 만큼 빼서 대입
- 일반 교체
- 조건부 교체(CAS; Compare and Swap)
- 읽기

■ Locking 사용하기

- C# 에서 오브젝트를 생성하고, 이걸로 locking을 걸 수 있음!

Locking을 사용한 해결 방법

```
1  using System;
2  using System.Threading;
3
4  namespace ThreadRacing
5  {
6      class Program
7      {
8          public static int gold;
9          // lock문에 사용될 객체
10         public static object lockGold = new object();
11
12         public static void Main(string[] args)
13         {
14             Program.gold = 0;
15
16             Thread thread0 = new Thread(() => Hunt());
17             thread0.Start();
18             Thread thread1 = new Thread(() => BuyItem());
19             thread1.Start();
20
21             Console.WriteLine(string.Format("골드 양 {0}", Program.gold));
22         }
23     }
```

Locking을 사용한 해결 방법

```
24
25 public static void Hunt()
26 {
27     Console.WriteLine(string.Format("사냥 시작!"));
28
29     for (int i = 0; i < 100; i++)
30     {
31         lock (Program.lockGold)
32         {
33             Program.gold += 10;
34             Console.WriteLine(string.Format("현재 골드 {0}", Program.gold));
35             Thread.Sleep(10);
36         }
37     }
38     Console.WriteLine(string.Format("사냥 종료!"));
39 }
40
```


Locking을 사용한 해결 방법

42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
..



```
public static void BuyItem()
{
    Console.WriteLine(string.Format("아이템구매 시작!"));

    for (int i = 0; i < 100; i++)
    {
        lock (Program.lockGold)
        {
            Program.gold -= 1;
            Console.WriteLine(string.Format("소비한 골드 {0}", Program.gold));
            Thread.Sleep(10);
        }
    }
    Console.WriteLine(string.Format("아이템 구매 종료!"));
}
```

Locking을 사용한 해결 방법

```
선택 C:\WINDOWS\system32\cmd.exe
소비한 805
소비한 804
소비한 803
소비한 802
현재 812
현재 822
현재 832
현재 842
현재 852
현재 862
현재 872
현재 882
현재 892
현재 902
소비한 901
소비한 900
소비한 899
소비한 898
소비한 897
소비한 896
소비한 895
소비한 894
소비한 893
소비한 892
현재 902
사냥 중 !
소비한 901
소비한 900
아이템 구매 종료!
계속하려면 아무 키나 누르십시오 . . .
```

레이스 컨디션 해결 (단 속도가 많이 느려짐!)

교착상태 (Dead Lock 걸기!)

■ 상황

Gold 를 플레이어가 사냥을 할때 10골드씩 모을수 있고, 퀘스트템 1개를 획득
플레이어는 사냥후 물약을 구매하며 1골드를 사용하고, 퀘스트템 1개를 처분

■ Gold 와 Item을 두개를 같이 걸어 보자!

- 사냥 쓰레드에서 Gold를 들어가서 Item을 걸기
- 아이템 구매 쓰레드에서는 Item을 들어가서 Gold를 걸기

교착상태 (Dead Lock 걸기!)

```
1  using System;
2  using System.Threading;
3
4  namespace ThreadRacing
5  {
6      class Program
7      {
8          public static int gold;
9          public static int item;
10         // lock문에 사용될 객체
11         public static object lockGold = new object();
12         public static object lockItem = new object();
13
14         public static void Main(string[] args)
15         {
16             Program.gold = 0;
17             Program.item = 0;
18
19             Thread thread0 = new Thread(() => Hunt());
20             thread0.Start();
21             Thread thread1 = new Thread(() => BuyItem());
22             thread1.Start();
23
24             Console.WriteLine(string.Format("골드 양 {0}", Program.gold));
25         }
26     }
```

교착상태 (Dead Lock 걸기!)

```
27
28 public static void Hunt()
29 {
30     Console.WriteLine(string.Format("사냥 시작!"));
31
32     for (int i = 0; i < 100; i++)
33     {
34         lock (Program.lockGold)
35         {
36             Program.gold += 10;
37             Console.WriteLine(string.Format("현재 골드 {0}", Program.gold));
38
39             lock (Program.lockItem)
40             {
41                 Program.item += 1;
42                 Console.WriteLine(string.Format("현재 퀘스트템 {0}", Program.item));
43             }
44
45             Thread.Sleep(10);
46         }
47     }
48     Console.WriteLine(string.Format("사냥 종료!"));
49 }
50
51
```

교착상태 (Dead Lock 걸기!)

```
53 public static void BuyItem()  
54 {  
55     Console.WriteLine(string.Format("아이템구매 시작!"));  
56  
57     for (int i = 0; i < 100; i++)  
58     {  
59         lock (Program.lockItem)  
60         {  
61             Program.item -= 1;  
62             Console.WriteLine(string.Format("현재 퀘스트템 {0}", Program.item));  
63  
64             lock (Program.lockGold)  
65             {  
66                 Program.gold -= 1;  
67                 Console.WriteLine(string.Format("소비한 골드 {0}", Program.gold));  
68             }  
69  
70             Thread.Sleep(10);  
71         }  
72     }  
73  
74     Console.WriteLine(string.Format("아이템 구매 종료!"));  
75 }  
76 }  
77 }  
78 }  
79 }  
80 }
```

교착상태 (Dead Lock 걸기!)

C:\> 선택 C:\WINDOWS\system32\cmd.exe

```
사냥 시작!  
현재 골드 10  
아이템구매 시작!  
골드 양 0  
현재 퀘스트템 1  
현재 퀘스트템 0  
현재 골드 20
```

교착상태! (너무나 쉽게 발생합니다!)

- 교착상태에 발생한 마지막 실습을 해결하기
 - 코드와, 해결된 스크린샷을 제출할것



Thank You
