Chapter 03

간단한 게임 만들기

Contents

3. 1 간단한 게임 만들기

3. 1 간단한 게임 만들기

❖ 이미지 로딩

◆ 이미지 파일을 불러와 화면에 그리기

ImageExample.java

```
public class ImageExample extends Activity  {

    @Override
    public void onCreate(Bundle savedInstanceState)  {
            super.onCreate(savedInstanceState);
            setContentView(new ImageView(this));

            //setContentView(R.layout.main);
    }
}
```

3. 1 간단한 게임 만들기

❖ 이미지 로딩 (cont.)

◆ 이미지 파일을 불러와 화면에 그리기

ImageView.java

```
public class ImageView extends View  {

    public ImageView(Context context)  {
            super(context);
    }

    @override
    public void onDraw(Canvas canvas)  {
            Bitmap _android = BitmapFactory.decodeResource(getResource( ),
                    R.drawable.android );
            canvas.drawBitmap(_android, 0, 0, null);
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성

◆ class 구성

- ✓ CardGameActivity
- ✓ CardGameView
- ✓ Card
- ✓ CardGameThread

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 1. class 생성

CardGameView.java

```
public class CardGameView extends View  {

    public CardGameView(Context context)  {
            super(context);
    }
    @override
    public void onDraw(Canvas canvas)  {
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성

◆ CardGameView ; step 2. 배경 이미지 그리기

CardGameView.java

```
public class CardGameView extends View  {
    // 1. 멤버 변수 추가
    Bitmap m_BackGroundImage;

    public CardGameView(Context context)  {
            super(context);
    // 2. 멤버 변수 설정
            m_BackGroundImage = BitmapFactory.decodeResource(getResource( ),
                    R.drawable.background, null);
    }
    @override
    public void onDraw(Canvas canvas)  {
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 2. 배경 이미지 그리기

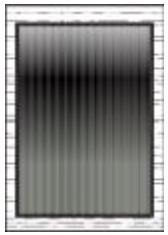
CardGameView.java

```
public class CardGameView extends View  {


    … …
    @override
    public void onDraw(Canvas canvas)  {
    // 3. 배경 이미지 그리기
            canvas.drawBitmap(m_BackGroundImage, 0, 0, null);
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성



◆ CardGameView ; step 3. 카드뒷면 이미지 그리기

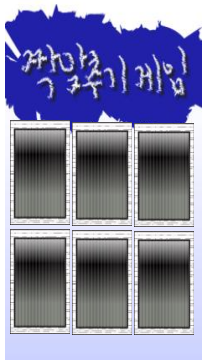
CardGameView.java

```
public class CardGameView extends View  {
    … …
    // 1. 멤버 변수 추가
    Bitmap m_CardBackSide;

    public CardGameView(Context context)  {
            super(context);
            m_BackGroundImage = BitmapFactory.decodeResource(getResource( ),
                        R.drawable.background, null);
     // 2. 멤버 변수 설정
            m_CardBackSide = BitmapFactory.decodeResource(getResource( ),
                        R.drawable.backside, null);
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성



◆ CardGameView ; step 3. 카드뒷면 이미지 그리기

CardGameView.java

```
public class CardGameView extends View  {


    … …
    @override
    public void onDraw(Canvas canvas)  {
    // 3. 카드뒷면 이미지 그리기
            canvas.drawBitmap(m_BackGroundImage, 0, 0, null);
            for (int y=0; y<2; y++)  {
                for (int x=0; x<3; x++)
                        canvas.drawBitmap(m_CardBackSide, 35 + x*90,
                                150 + y*130, null);
            }
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ Card ; step 1. class 생성

Card.java

```
package com.cardgame;

public class Card  {


}
```

✓ 카드 상태

- 처음 카드 상태 (임시적인 앞면)
- 게임이 시작하고 뒷면 상태
- 플레이어가 확인하는 임시적인 앞면
- 플레이어가 짝을 맞추어서 열려 있는 앞면

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성 (cont.)

◆ Card ; step 2. 카드 상태 정의 및 멤버 변수 초기화

Card.java

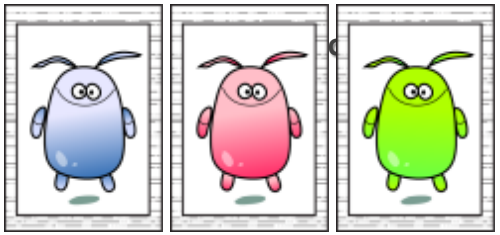
```
package com.cardgame;

public class Card  {
    // 1. 카드 상태 상수 정의
    public static final int CARD_SHOW = 0;
    public static final int CARD_CLOSE = 1;
    public static final int CARD_PLAYEROPEN = 2;
    public static final int CARD_MATCHED = 0;
    // 2. 카드 상태 멤버 변수 선언 및 초기화
    public int m_State;

    public Card( ) {
            m_State = CARD_SHOW;
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성



◆ Card ; step 3. 카드 앞면 이미지 정의

Card.java

```
public class Card  {
    … …
    private static final int IMG_RED = 1;
    private static final int IMG_GREEN = 2;
    private static final int IMG_BLUE = 3;

    public int m_Color;

    public Card(int _Color ) {
            m_State = CARD_SHOW;
            m_Color = _Color;
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 1. 카드 앞면 이미지 불러오기

CardGameView.java

```
public class CardGameView extends View  {
    … …
    Bitmap m_Card_Red;
    Bitmap m_Card_Green;
    Bitmap m_Card_Blue;

    public CardGameView(Context context)  {
            … …
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 1. 카드 앞면 이미지 불러오기

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public CardGameView(Context context)  {
            super(context);
            m_BackGroundImage = BitmapFactory.decodeResource(getResource( ),
                    R.drawable.background, null);
            m_CardBackSide = BitmapFactory.decodeResource(getResource( ),
                    R.drawable.backside, null);

            m_Card_Red = BitmapFactory.decodeResource(getResource( ),
                    R.drawable.front_red, null);
            m_Card_Green = BitmapFactory.decodeResource(getResource( ),
                    R.drawable. front_green, null);
            m_Card_Blue = BitmapFactory.decodeResource(getResource( ),
                    R.drawable. front_blue, null);
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 1. 카드 앞면 이미지 불러오기

CardGameView.java

```
public class CardGameView extends View  {
    … …
    // 화면에 표시할 카드
    Card m_Shuffle[ ][ ];

    public CardGameView(Context context)  {
            super(context);
            … …
            // 화면에 표시할 카드만큼 할당 (3 x 2)
            m_Shuffle = new Card[3][2];
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 2. 카드 앞면 이미지 설정

CardGameView.java

```
public class CardGameView extends View  {
    … …

    public void setCardShuffle( )  {
            // 각각의 색을 가진 카드들을 생성
            // 랜덤으로 해야 하지만, 일단 단순히 고정된 값으로
            m_Shuffle [0][0]= new Card(Card. IMG_RED);
            m_Shuffle [0][1]= new Card(Card. IMG_BLUE);
            m_Shuffle [1][0]= new Card(Card. IMG_GREEN);
            m_Shuffle [1][1]= new Card(Card. IMG_GREEN);
            m_Shuffle [2][0]= new Card(Card. IMG_BLUE);
            m_Shuffle [2][1]= new Card(Card. IMG_RED);
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 2. 카드 앞면 이미지 설정

CardGameView.java

```
public class CardGameView extends View  {
    … …

    public CardGameView(Context context)  {
            super(context);
            … …

            // 카드를 섞음
            setCardShuffle( );
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 화면 구성 (cont.)

◆ CardGameView ; step 3. 상태에 따라 카드 그리기

✓ onDraw 메서드

- 현재 그려야 할 Card의 카드 모양에 따라 빨간색, 녹색, 파란색 카드를 그림

CardGameView.java

```
public class CardGameView extends View  {
    … …

    @Override
    public void onDraw(Canvas canvas)  {
            // 배경 이미지 그리기
            canvas.drawBitmap(m_BackGroundImage, 0, 0, null);
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 화면 구성 (cont.)

◆ CardGameView ; step 3. 상태에 따라 카드 그리기

CardGameView.java

```
public class CardGameView extends View  {
    … …
    @Override
    public void onDraw(Canvas canvas)  {
            canvas.drawBitmap(m_BackGroundImage, 0, 0, null);
            // 카드 그려주기
            for (int y=0; y<2; y++)  {
                for (int x=0; x<3; x++)
                        // 색상 값에 따라 다른 이미지 그려주기
                        if (m_Shuffle [x][y]. m_Color == Card. IMG_RED)
                            canvas.drawBitmap(m_Card_Red, 35+x*90, 150+y*130, null);
                        else if (m_Shuffle [x][y]. m_Color == Card. IMG_GREEN)
                            canvas.drawBitmap(m_Card_Green, 35+x*90, 150+y*130, null);
                        else if (m_Shuffle [x][y]. m_Color == Card. IMG_BLUE)
                            canvas.drawBitmap(m_Card_Blue, 35+x*90, 150+y*130, null);
            }
    }
}
```

3. 1 간단한 게임 만들기

❖ 입력 처리

◆ onKeyDown을 이용한 키 입력 처리

KeyDownExample.java

```
public class KeyDownExample extends Activity  {


    @Override
    public void onCreate(Bundle savedInstanceState)  {
            super .onCreate(savedInstanceState);
            setContentView( new KeyDownExampleView( this));
    }
}
```

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onKeyDown을 이용한 키 입력 처리

KeyDownExampleView.java

```
public class KeyDownExampleView extends View  {
    public void KeyDownExampleView(Context text)  {
            super (context);
            setFocusable( true); // 이 뷰에 포커스를 줍니다.
    }
    @Override
    protected void onDraw( Canvas canvas)  {
            … …
    }
    @Override
    public boolean onKeyDown( int keyCode, KeyEvent event)  {
            // 여기서 키 입력을 처리합니다.
            // if (keCode == KeyEvent.KEYCODE_DPAD_LEFT) … …
            invalidate( );              // 뷰의 화면 갱신
            return super .onKeyDown(keyCode, event);
    }
}
```

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onTouchEvent를 이용한 키 입력 처리

TouchEventExample.java

```
public class TouchEventExample extends Activity  {

    @Override
    public void onCreate(Bundle savedInstanceState)  {
            super .onCreate(savedInstanceState);
            setContentView( new TouchEventExampleView( this));
    }
}
```

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onTouchEvent를 이용한 키 입력 처리

TouchEventExampleView.java

```
public class TouchEventExampleView extends View  {

    public void TouchEventExampleView(Context text)  {
            super (context);
    }

    @Override
    protected void onDraw(Canvas canvas)  {

    }
}
```

✓ 리스너 핸들러를 이용하는 방법 ; Boolean onTouch(View v, MotionEvent e)

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onTouchEvent를 이용한 키 입력 처리

TouchEventExampleView.java

```
public class TouchEventExampleView extends View  {

    … …
    @Override
    protected void onDraw(Canvas canvas)  {
            … …
    }
    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            // 좌표를 얻어옵니다.
            m_x = (int) event.getX( );
            m_y = (int) event.getY( );

            invalidate( ) ;  // 화면을 갱신합니다.
            return super .onTouchEvent(event);
    }
}
```

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onTouchEvent를 이용한 키 입력 처리

- ✓ MotionEvent에서 가져오는 터치 액션은 16개의 상수가 정의되어 있음
- ✓ 참고) 이벤트에 대한 처리는 Activity 보다는 View에서 처리하는 것이 일반적(화면과 뷰 원점 차이)

안드로이드의 자주 사용되는 터치 액션

ACTION_DOWN	화면을 터치했을 때 발생하는 이벤트
ACTION_MOVE	화면을 누르고 있는 상태에서 움직일 때 발생하는 이벤트
ACTION_UP	화면을 누르고 있다가 뗄 때 발생하는 이벤트
ACTION_CANCEL	제스처를 취하다가 중단했을 때 발생하는 이벤트
ACTION_OUTSIDE	화면을 누르고 있는 상태에서 화면 밖으로 나가면 발생하는 이벤트

3. 1 간단한 게임 만들기

❖ 입력 처리 (cont.)

◆ onTouchEvent를 이용한 키 입력 처리

TouchEventExampleView.java

```
public class TouchEventExampleView extends View  {
    … …
    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            // 좌표를 얻어옵니다.
            m_x = (int) event.getX( );
            m_y = (int) event.getY( );

            if (event.getAction( ) == MotionEvent. ACTION_DOWN) { // 기타 액션 이벤트
                … …
            }
            invalidate( ) ;  // 화면을 갱신합니다.
            //return super .onTouchEvent(event);
            // ACTION_MOVE나 ACTION_UP의 액션 이벤트 처리 위해서는
            // TRUE를 반환해야 합니다.
            return true;
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리

◆ CardGameView ; step 1. 게임 시작 설정

CardGameView.java

```
public class CardGameView extends View  {
    … …
    @Override
    public void onDraw(Canvas canvas)  {
            canvas.drawBitmap(m_BackGroundImage, 0, 0, null);
            for (int y=0; y<2; y++)  {
                for (int x=0; x<3; x++)
                        // 카드 앞면을 그려야 하는 경우
                        if(m_Shuffle [x][y]. m_State == Card. CARD_SHOW ||
                            m_Shuffle [x][y]. m_State == Card. CARD_PLAYEROPEN ||
                            m_Shuffle [x][y]. m_State == Card. CARD_MATCHED )  {
                            … … // 색상에 따라 카드 앞면 그리기
                        }
                        else  {  // 카드 뒷면을 그려야 하는 경우
                            canvas.drawBitmap( m_Card_Backside, 35+x*90,
                                    150+y*130, null);
                        }
            }
    }
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리 (cont.)

◆ CardGameView ; step 1. 게임 시작 설정

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public void startGame( )  {
            // 모든 카드를 뒷면 상태로 만든다.
            m_Shuffle [0][0]. m_State = Card. CARD_CLOSE;
            m_Shuffle [0][1]. m_State = Card. CARD_CLOSE;
            m_Shuffle [1][0]. m_State = Card. CARD_CLOSE;
            m_Shuffle [1][1]. m_State = Card. CARD_CLOSE;
            m_Shuffle [2][0]. m_State = Card. CARD_CLOSE;
            m_Shuffle [2][1]. m_State = Card. CARD_CLOSE;

            invalidate( ); // 화면을 갱신합니다.
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리 (cont.)

◆ CardGameView ; step 1. 게임 시작 설정

CardGameView.java

```
public class CardGameView extends View  {
    … …

    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            startGame( );  // 게임을 시작합니다.

            invalidate( );  // 화면을 갱신합니다.
            return super .onTouchEvent(event);
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리 (cont.)

◆ 카드 선택

- ✓ onTouchEvent 좌표값이 카드 영역 내부인지의 판단
  - Rect.contains(x, y);

◆ 게임 상태

- ✓ 게임준비 / 게임중 / 게임종료

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public static final int STATE_READY = 0;
    public static final int STATE_GAME = 1;
    public static final int STATE_END = 2;
    private int m_State = STATE_READY;
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리 (cont.)

◆ CardGameView ; step 2. 입력 처리

CardGameView.java

```
public class CardGameView extends View  {
    … …
    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            if (m_State == STATE_READY)  {
                    startGame( );
                    m_State = STATE_GAME;
            }
            else if (m_State == STATE_GAME)  {
                // 카드 뒤집는 처리
            }
            else if (m_State == STATE_END)  {
                    m_State = STATE_READY;
            }
            invalidate( );
            return true
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 이벤트 처리 (cont.)

◆ CardGameView ; step 2. 입력 처리

CardGameView.java

```
public class CardGameView extends View  {
    … …
    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            … …
            else if (m_State == STATE_GAME)  {
                int px = (int)event.getX( );
                int py = (int)event.getY( );
                for (int y=0; y<2; y++)  {
                        for (int x=0; x<3; x++)  {
                            // 각 카드의 박스 값을 생성
                            Rect box_card = new Rect(35+x*90, 150+y*130,
                                    35+x*90+80, 150+y*130+115);
                            if (box_card .contains(px,py)) // 선택된 카드 뒤집기
                                    m_Shuffle [x][y]. m_State = Card. CARD_PLAYEROPEN;
                        }
                }
            }
            … …
```

3. 1 간단한 게임 만들기

❖ 스레드

◆ 스레드 처리

ThreadExample.java

```
public class ThreadExample extends Activity  {

    @Override
    public void onCreate(Bundle savedInstanceState)  {
            super .onCreate(savedInstanceState);
            setContentView( new ThreadExampleView( this));
    }
}
```

3. 1 간단한 게임 만들기

❖ 스레드 (cont.)

◆ 스레드 처리

ThreadExampleView.java

```
public class ThreadExampleView extends View  {
    public void ThreadExampleView(Context text)  {
            super (context);
    }
    @Override
    protected void onDraw( Canvas canvas)  {

    }
}
```

3. 1 간단한 게임 만들기

❖ 스레드 (cont.)

◆ 스레드 클래스 상속받아 사용하는 방법

ElevatorMoveThread.java

```
public class ElevatorMoveThread extends Thread  {
    public void run( )  {
            // 엘리베이터를 움직이는 동작
    }
}
```

ThreadExampleView.java

```
public class ThreadExampleView extends View  {
    public void ThreadExampleView(Context text)  {
            super (context);
            ElevatorMoveThread thread_1 = new ElevatorMoveThread( );
            thread_1.start( );
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ 스레드 (cont.)

◆ Runnable 인터페이스를 구현하는 방법

ElevatorMoveRunnable.java

```
public class ElevatorMoveRunnable extends Runnable  {
    public ThreadExampleView m_View;

    public ElevatorMoveRunnable(ThreadExampleView _view)  {
            m_View = _view;
    }
    public void run( )  {
            // 엘리베이터를 움직이는 동작
    }
}
```

3. 1 간단한 게임 만들기

❖ 스레드 (cont.)

◆ Runnable 인터페이스를 구현하는 방법

ThreadExampleView.java

```
public class ThreadExampleView extends View  {
    public void ThreadExampleView(Context text)  {
            super (context);
            ElevatorMoveThread thread_1 = new ElevatorMoveThread( );
            thread_1.start( );

            Runnable _runnable = new ElevatorMoveRunnable(this);
            Thread thread_2 = new Thread(_runnable);
            thread_2.start( );
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ 스레드 (cont.)

◆ 스레드 클래스 상속받아 사용하는 방법

- ✓ 스레드 클래스 상속으로 스레드 클래스 내의 여러 메서드를 바로 사용할 수 있어 스레드를 관리하는데 편리
- ✓ 자바는 다중 상속이 되지 않으므로 다른 클래스를 상속받지 않는 경우에 사용

◆ Runnable 인터페이스를 구현하는 방법

- ✓ 이미 상속받은 클래스를 스레드화 할 때 유용한 방법
- ✓ 스레드 클래스를 상속받지 않으므로 스레드 클래스의 메서드를 사용할 수 없음

◆ 주의사항

- ✓ 안드로이드에서는 View가 생성되면 그 View를 생성한 스레드만 해당 View에 접근 가능
- ✓ invalidate( ) 메서드 → postInvalidate( ) 메서드 사용

3. 1 간단한 게임 만들기

❖ Random 클래스와 Math 클래스

◆ Random 클래스

Random 클래스 사용

```
Random rand = new Random( );
int result = rand.nextInt( );
```

◆ Math 클래스

Math 클래스 사용

```
double temp = Math.random( ) * 100.0;
int result = (int)Math.rint(temp);
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 스레드

◆ CardGameView ; step 1. 두 카드 색상 비교 처리

CardGameView.java

```
public class CardGameView extends View  {
    … …
    // 짝 맞추기 비교를 위한 변수
    private Card m_SelectedCard_1 = null;
    private Card m_SelectedCard_2 = null;
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 스레드 (cont.)

◆ CardGameView ; step 1. 두 카드 색상 비교 처리

CardGameView.java

```
public class CardGameView extends View  {
    … …
    @Override
    public boolean onTouchEvent(MotionEvent event)  {
            … …
            else if (m_State == STATE_GAME)  {
                int px = (int)event.getX( );
                int py = (int)event.getY( );
                for (int y=0; y<2; y++)  {
                        for (int x=0; x<3; x++)  {
                            // 각 카드의 박스 값을 생성
                            Rect box_card = new Rect(35+x*90, 150+y*130,
                                    35+x*90+80, 150+y*130+115);
                            if (box_card .contains(px,py)) // 선택된 카드 뒤집기
                                    ////////////////////////////////
                        }
                }
            }
            … …
```

3. 1 간단한 게임 만들기

CardGameView.java

```
if (box_card .contains(px,py))  {  // 선택된 카드 뒤집기
    // (x, y)에 위치한 카드가 선택되었다.
    if (m_Shuffle [x][y]. m_State != Card. CARD_MATCHED)
    // 맞춘 카드는 뒤집을 필요가 없습니다.
            if( m_SelectCard_1 == null)  { // 첫 카드를 뒤집는다면
                m_SelectCard_1 = m_Shuffle [x][y];
                m_SelectCard_1. m_State = Card. CARD_PLAYEROPEN;
            }
            else {   // 이미 첫번째 카드가 뒤집혀 있어 두번째로 뒤집는다면
                if ( m_SelectCard_1 != m_Shuffle [x][y])  {  // 중복 뒤집기 방지
                        m_SelectCard_2 = m_Shuffle [x][y];
                        m_SelectCard_2. m_State = Card. CARD_PLAYEROPEN;
                }
            }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame - 스레드 (cont.)

◆ CardGameView ; step 1. 두 카드 색상 비교 처리

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public void checkMatch( )  {
            //  두 카드 중 하나라도 선택이 안 되었다면 비교할 필요가 없습니다.
            if ( m_SelectCard_1 == null || m_SelectCard_2 == null)  return;
            //  두 카드의 색상을 비교합니다.
            if ( m_SelectCard_1.m_Color == m_SelectCard_2.m_Color)  {
            // 두 카드의 색상이 같으면 두 카드를 맞춘 상태로 바꿉니다.
                m_SelectCard_1.m_State = Card. CARD_MATCHED;
                m_SelectCard_2.m_State = Card. CARD_MATCHED;
                //  다시 선택할 수 있도록 null로 설정
                m_SelectCard_1 = null;
                m_SelectCard_2 = null;
            }
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 스레드 (cont.)

◆ CardGameView ; step 1. 두 카드 색상 비교 처리

CardGameView.java

```
            else    {
                // 두 카드의 색상이 다른 경우 두 카드를 이전처럼 뒷면으로 돌려줍니다.
                m_SelectCard_1.m_state = Card. CARD_CLOSE;
                m_SelectCard_2.m_state = Card. CARD_CLOSE;
               // 다시 선택할 수 있도록 null로 설정
                m_SelectCard_1 = null;
                m_SelectCard_2 = null;
            }
            invalidate( );
        }
    }
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 스레드 (cont.)

◆ CardGameView ; step 2. 스레드 사용

CardGameThread.java

```
public class CardGameThread extends Thread {
    CardGameView m_View;
    CardGameThread(CardGameView _View){
            m_View = _View;
    }
    public void run( ){
            while ( true)  m_View.checkMatch( );
    }
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 스레드 (cont.)

◆ CardGameView ; step 2. 스레드 사용

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public CardGameView(Context context)  {
            … …
            // 짝맞추기를 검사하는 스레드 실행
            CardGameThread _thread = new CardGameThread( this);
            _thread.start( );
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 스레드 (cont.)

◆ CardGameView ; step 2. 스레드 사용

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public void checkMatch( )  {
            … …
            // invalidate( );  // 스레드에서 사용하므로
            postInvalidate( );
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ CardMatchGame – 스레드 (cont.)

◆ CardGameView ; step 2. 스레드 사용

CardGameView.java

```
public class CardGameView extends View  {
    … …
    public void checkMatch( )  {
            … …
            else {
                try {
                        Thread.sleep(500);
                } catch (InterruptedException e)  {  }
                … …
            }
            … …
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ 사운드

◆ 사운드 재생 프로그램

- ✓ 실행하면 배경음악 플레이
- ✓ 스페이스 바 → 배경음악 pause/play
- ✓ 방향 키 → 간단한 효과음

SoundExample.java

```
public class SoundExample extends Activity  {

    @Override
    public void onCreate(Bundle savedInstanceState)  {
            super .onCreate(savedInstanceState);
            setContentView( new SoundExampleView( this));
    }
}
```

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ 사운드 재생 프로그램

SoundExampleView.java

```
public class SoundExampleView extends View  {
    public void SoundExampleView(Context text)  {
            super (context);
    }

    @Override
    protected void onDraw(Canvas canvas)  {

    }
}
```

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ 사운드 리소스 파일 추가

- ✓ Package Explorer → res → New → Folder 선택해 raw 폴더 생성
- ✓ 사운드 파일을 raw 폴더로 복사

◆ 안드로이드 지원 오디오 형식

- ✓ 사운드 포맷 ; mp3, aac, wma, amr, ogg, midi, wav
- ✓ 게임에 적합한 포맷 ; ogg, wav, mp3

◆ 사운드 재생

- ✓ MediaPlayer를 이용한 사운드 재생
- ✓ SoundPool을 이용한 사운드 재생

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ MediaPlayer를 이용한 사운드 재생

- ✓ 사운드의 경우 비트맵과 달리 사용할 때마다 호출하면 예외(exception)가 밸생
- ✓ 따라서 반드시 로드할 때와 사용할 때를 구분해서 사용

- ✓ 게임 제작시 어려움
  - MeidaPlayer로 사운드를 재생할 때 사운드 리소스를 동기화 (ver 2.2)
  - 사운드 여러 개를 동시에 재생하면 교착상태 현상 발생

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ MediaPlayer를 이용한 사운드 재생

SoundExampleView.java

```
public class SoundExampleView extends View  {
    MediaPlayer m_Sound_BackGround;      // 배경음악
    MediaPlayer m_Sound_1;               // 효과음1
    MediaPlayer m_Sound_2;               // 효과음2

    public void SoundExampleView(Context text)  {
            super (context);
            // 사운드 리소스 로딩
            m_Sound_BackGround = MediaPlayer.create(context, R.raw.background);
            m_Sound_1 = MediaPlayer.create(context, R.raw.effect1);
            m_Sound_2 = MediaPlayer.create(context, R.raw.effect2);
    }
    … …
}
```

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ MediaPlayer를 이용한 사운드 재생

✓ 사운드 재생

SoundExampleView.java

```
public class SoundExampleView extends View  {
    … …
    public void SoundExampleView(Context text)  {
            super (context);
            // 사운드 리소스 로딩
            … …
            m_Sound_BackGround.start( );
    }
    … …
            // m_Sound_1.pause( );  // pause 기능
            // m_Sound_2.isPlaying( );  // 현재 play 중인지
   … …
}
```

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ SoundPool을 이용한 사운드 재생

1. SoundPool 객체 생성
2. 사운드 리소스를 로드
3. SoundPool을 이용해 재생

✓ 객체 생성 ; `SoundPool soundPool = new SoundPool (최대스트림개수, 오디오스트림타입, 샘플링품질);`

✓ 사운드 리소스 로드 ; `int sound = soundPool.load(콘텍스트, 리소스아이디, 우선권);`

✓ 사운드 재생 ; `soundPool.play(실행할사운드ID, 좌측볼륨, 우측볼륨, 재생우선순위, 반복여부, 속도);`

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ SoundPool을 이용한 사운드 재생

SoundExampleView.java

```
public class SoundExampleView extends View  {
    … …

    SoundPool m_SoundPool;      // 사운드 풀
    int m_Sound_id_1;           // 효과음1
    int m_Sound_id_2;           // 효과음2

    … …
}
```

3. 1 간단한 게임 만들기

❖ 사운드

(cont.)

SoundExampleView.java

```
public class SoundExampleView extends View  {
    … …
    public void SoundExampleView(Context text)  {
            super (context);
            … …
            m_SoundPool = new SoundPool(5, AudioManager.STREAM_MUSIC, 0);
            // MediaPlayer를 이용해서 리소스를 로드합니다.
            m_Sound_BackGround = MediaPlayer.create(context, R.raw.background);
            … …
            // SoundPool을 이용해서 리소스를 로드합니다.
            m_Sound_id_1 = m_SoundPool.load(context, R.raw.effect1 ,1);
            m_Sound_id_2 = m_SoundPool.load(context. R.raw.effect2 ,1);
            … …
    }
    … …  // m_SoundPool.play(m_Sound_id_1, 1, 1, 0, 0, 1);
}
```

3. 1 간단한 게임 만들기

❖ 사운드 (cont.)

◆ MediaPlayer와 SoundPool

- ✓ MediaPlayer는 하나의 사운드를 재생할 때 기능이 다양
- ✓ SoundPool은 1MB가 넘는 사운드를 재생하면 Heap size overflow 발생

3. 1 간단한 게임 만들기

❖ CardMatchGame 실행

◆ 사운드

- ✓ 기본적인 배경 음악 추가
- ✓ 카드를 뒤집을 때 효과음 추가

◆ 추가 사항

- ✓ 게임을 시작할 때마다 랜덤으로 카드 위치 변경
- ✓ 게임이 시작되고 짝을 모두 맞출 때까지의 시간 체크 및 화면 표시
- ✓ 짝을 맞추다가 실패하는 횟수를 카운트해서 화면에 표시
- ✓ 게임 restart 기능