Android GDI (Graphics Device Interface) 图形设备接口

View

- 创建一个继承自 `android.view.View` 的类，重写 `onDraw` 方法。

```
public class MyView extends View {
    public MyView(Context context) {
        super(context);
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
        // 在这里进行绘图操作
    }
}
```

- 在活动中使用自定义的 `View`。

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(new MyView(this));
    }
}
```

绘制图形

`https://developer.android.google.cn/reference/android/graphics/Canvas?hl=en#public-methods`

- 在 `onDraw` 方法中使用 `Canvas` 绘图。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 绘制一个圆形
    Paint paint = new Paint();
    paint.setColor(Color.RED);
```

```
    canvas.drawCircle(100, 100, 50, paint);
}
```

- 在 `onDraw` 方法中调用 `invalidate` 方法，会触发 `onDraw` 方法的调用，实现绘图的循环。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 绘制一个圆形
    Paint paint = new Paint();
    paint.setColor(Color.RED);
    canvas.drawCircle(100, 100, 50, paint);
    // 触发 onDraw 方法的调用
    invalidate();
}
```

绘制位图

- 在 `onDraw` 方法中使用 `Bitmap` 绘制位图。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 加载位图
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.my_image);
    // 绘制位图
    canvas.drawBitmap(bitmap, 0, 0, null);
}
```

- 位图剪切，调用 `Bitmap` 的 `createBitmap` 方法。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 加载位图
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.my_image);
    // 剪切位图的一部分
    Bitmap croppedBitmap = Bitmap.createBitmap(bitmap, 100, 100, 200, 200);
    // 绘制剪切后的位图
    canvas.drawBitmap(croppedBitmap, 0, 0, null);
}
```

- 位图缩放，调用 `Bitmap` 的 `createScaledBitmap` 方法。

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 加载位图
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.my_image);
    // 缩放位图
    Bitmap scaledBitmap = Bitmap.createScaledBitmap(bitmap, 200, 200, true);
    // 绘制缩放后的位图
    canvas.drawBitmap(scaledBitmap, 0, 0, null);
}
```

- 绘制位图的一部分。

```
@Override
protected void onDraw(Canvas canvas) {
```

```
    super.onDraw(canvas);
    // 加载位图
    Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.my_image);
    // 绘制位图的一部分
    Rect srcRect = new Rect(0, 0, bitmap.getWidth(), bitmap.getHeight());
    Rect dstRect = new Rect(0, 0, getWidth(), getHeight());
    canvas.drawBitmap(bitmap, srcRect, dstRect, null);
}
```

触屏响应

- 调用 `View` 的 `setOnTouchListener` 方法，设置 `View` 的触摸事件监听器，在 `onTouch` 方法中处理触摸事件，`onTouch` 方法返回 `true` 表示事件已被处理，`onTouch` 方法返回 `false` 表示事件未被处理。

```
// 设置监听器
public class MyView extends View {
    public MyView(Context context) {
        super(context);
        setOnTouchListener(this);
    }

    @Override
    public boolean onTouch(View v, MotionEvent event) {
        Log.v("MyView", "onTouch" + event.toString());

        // 处理触摸事件
        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
                Log.v("MyView", "ACTION_DOWN x:" + event.getX() + " y:" + event.getY());
                break;
            case MotionEvent.ACTION_MOVE:
                Log.v("MyView", "ACTION_MOVE x:" + event.getX() + " y:" + event.getY());
                break;
            case MotionEvent.ACTION_UP:
                Log.v("MyView", "ACTION_UP x:" + event.getX() + " y:" + event.getY());
                break;
        }
        return true;
    }
}
```

随机数

- 调用 `Random` 类的 `nextInt` 方法，生成随机数。

```
Random random = new Random();
int randomNumber = random.nextInt(100); // 生成 0 到 99 之间的随机数
```

播放音频

- 调用 `MediaPlayer` 类的 `create` 方法，创建 `MediaPlayer` 对象， 调用 `start` 方法，播放音频，调用 `setLooping(true)` 方法，设置循环播放， 调用 `stop` 方法，停止播放。

```
// 创建 MediaPlayer 对象
MediaPlayer mediaPlayer = MediaPlayer.create(this, R.raw.my_audio);
// 播放音频
```

```
mediaPlayer.start();
// 设置循环播放
mediaPlayer.setLooping(true);
// 停止播放
mediaPlayer.stop();
```

基础游戏逻辑

帧绘制

- 在 `onDraw` 方法中，调用 `invalidate` 方法，触发 `onDraw` 方法的调用，实现帧绘制，60 帧即每秒刷新 60 次。

```
@Override
public void run(){
    // 当前时间
    long currentTime = System.currentTimeMillis();
    // 每秒绘制 60 次
    while (true) {
        if (currentTime - lastTime >= 1000 / 60) {
            // 绘制一帧
            invalidate();
            // 更新时间
            lastTime = currentTime;
        }
        // 当前时间
        currentTime = System.currentTimeMillis();
    }
}
```

滚动背景图

- 在 `onDraw` 方法中，绘制背景图，绘制完背景图后，将背景图的位置以每秒 60 帧向下移动，再将移动到屏幕外的背景图重新绘制到屏幕上方。

```
// 在 onDraw 方法中，绘制背景图
@Override
public void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    // 绘制背景图
    canvas.drawBitmap(backgroundBitmap, backgroundX, backgroundY, null);
    // 背景图向下移动
    backgroundY += 1;
    // 如果背景图移动到屏幕外，重新绘制到屏幕上方
    if (backgroundY >= getHeight()) {
        backgroundY = -getHeight();
    }
}

// 在 run 方法中，调用 invalidate 方法，触发 onDraw 方法的调用
public void run() {
    // 当前时间
    long currentTime = System.currentTimeMillis();
    // 每秒绘制 60 次
    while (true) {
        if (System.currentTimeMillis() - currentTime >= 1000 / 60) {
            // 绘制一帧
```

```
            invalidate();
            // 更新时间
            currentTime = System.currentTimeMillis();
        }
    }
}
```