原始代码:

```java
public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    int num;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            num++;
            textView.setText(String.valueOf(num));
        });

        buttonMinus.setOnClickListener(v -> {
            num--;
            textView.setText(String.valueOf(num));
        });
    }
}
```

重写onSaveInstanceState(Bundle outState)保存数据，并通过savedInstanceState读取数据:

```java
public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    int num;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 方法1: 通过savedInstanceState读取临时保存的数据, Bundle: key-value
        if (savedInstanceState != null) {
            num = savedInstanceState.getInt("NUM", 0);
            textView.setText(String.valueOf(num));
        }

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            num++;
            textView.setText(String.valueOf(num));
        });

        buttonMinus.setOnClickListener(v -> {
            num--;
            textView.setText(String.valueOf(num));
        });

        }

    // 方法1：通过重写onSaveInstanceState临时保存数据
    @Override
    protected void onSaveInstanceState(@NonNull Bundle outState) {
        super.onSaveInstanceState(outState);
        outState.putInt("NUM", num);
    }
}
```

利用ViewModel管理数据：

```
public class MyViewModel extends ViewModel {
    private int num;

    public int getNum() {
        return num;
    }

    public void add(int i){
        num += i;
    }
}

public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    MyViewModel myViewModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 创建MyViewModel
        myViewModel = new ViewModelProvider(this).get(MyViewModel.class);

        textView.setText(String.valueOf(myViewModel.getNum()));

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            myViewModel.add(1);
            textView.setText(String.valueOf(myViewModel.getNum()));
        });
        buttonMinus.setOnClickListener(v -> {
            myViewModel.add(-1);
            textView.setText(String.valueOf(myViewModel.getNum()));
        });
    }
}
```

进阶版：ViewModel + LiveData/MutableLiveData:

```
public class MyViewModel extends ViewModel {
    // 使用MutableLiveData
    private MutableLiveData<Integer> num;

    // 返回值必须为MutableLiveData
    public MutableLiveData<Integer> getNum() {
        // 防止为空
        if (num == null) {
            num = new MutableLiveData<>();
            num.setValue(0);
        }
        return num;
    }

    public void add(int i) {
        int cur = getNum().getValue();
        num.setValue(cur + i);
    }
}

public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    MyViewModel myViewModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 创建MyViewModel
        myViewModel = new ViewModelProvider(this).get(MyViewModel.class);

        // 观测数据
        myViewModel.getNum().observe(this, new Observer<Integer>() {
            @Override
            public void onChanged(Integer integer) {
                textView.setText(String.valueOf(integer));
            }
        });

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            myViewModel.add(1);
        });

        buttonMinus.setOnClickListener(v -> {
            myViewModel.add(-1);
        });
    }
}
```

SharedPreferences管理数据:

```
public class MyViewModel extends AndroidViewModel {
    // 使用MutableLiveData
    private MutableLiveData<Integer> num;
    SharedPreferences shp;

    public MyViewModel(@NonNull Application application) {
        super(application);
        // 创建SharedPreferences
        shp = getApplication().getSharedPreferences("myShp", Context.MODE_PRIVATE);
    }

    // 返回值必须为MutableLiveData
    public MutableLiveData<Integer> getNum() {
        // 防止为空
        if (num == null) {
            num = new MutableLiveData<>();
            num.setValue(0);
        }
        return num;
    }

    public void add(int i) {
        int cur = getNum().getValue();
        num.setValue(cur + i);
    }

    public void load() {
        // 从SharedPreferences中读取数据, key-value
        int x = shp.getInt("NUM", 0);
        setNum(x);
    }

    public void save() {
        // 将数据保存到SharedPreferences中
        SharedPreferences.Editor editor = shp.edit();
        editor.putInt("NUM", getNum().getValue());
        editor.apply();
    }
}

public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    MyViewModel myViewModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 创建MyViewModel
        myViewModel = new ViewModelProvider(this).get(MyViewModel.class);

        // 观测数据
        myViewModel.getNum().observe(this, new Observer<Integer>() {
            @Override
            public void onChanged(Integer integer) {
                textView.setText(String.valueOf(integer));
            }
        });

        myViewModel.load(); // 读取数据

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            myViewModel.add(1);
        });

        buttonMinus.setOnClickListener(v -> {
            myViewModel.add(-1);
        });
    }
}

// 重写onPause,永久保存数据
// 由于Activity可能会因为内存不足被杀死，所以重写onPause比重写onStop, onDestroy更安全可靠
@Override
protected void onPause() {
    super.onPause();
    myViewModel.save(); // 保存数据
}
```

内部存储:

```
public class DataHelper {
    private Application application;
    private String fileName = "myFile.txt";

    public DataHelper(Application application) {
        this.application = application;
    }

    // 将数据保存到内部文件中
    void saveToIntervalFile(MutableLiveData<Integer> num) {
        try {
            // 获取文件输出流(如果还没有指定文件，Android会自动创建该文件)
            FileOutputStream fos = application.openFileOutput(fileName,
Context.MODE_PRIVATE);
            // 将数据转换成字节数组
            String str = num.getValue().toString();
            byte[] bytes = str.getBytes();
            // 将字节数组写入文件输出流
            fos.write(bytes);
            fos.close();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }

    // 从内部文件中读取数据
    void loadFromIntervalFile(MutableLiveData<Integer> num) {
        try {
            File file = new File(application.getFilesDir(), fileName);
            if (!file.exists()) {
                return; // 还没有存入数据，直接返回
            }
            // 获取文件输入流
            FileInputStream fis = application.openFileInput(fileName);
            // 返回要读取的剩余字节数
            int length = fis.available();
            byte[] bytes = new byte[length];
            // 将文件输入流读取到的数据放到字节数组中
            fis.read(bytes);
            // 将字节数组转换成字符串
            String str = new String(bytes, 0, length);
            int x = Integer.parseInt(str);
            num.setValue(x);
            fis.close();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}


public class MyViewModel extends AndroidViewModel {
    // 使用MutableLiveData
    private MutableLiveData<Integer> num;
    private DataHelper dataHelper;

    public MyViewModel(@NonNull Application application) {
        super(application);
        // 创建DataHelper
        dataHelper = new DataHelper(application);
    }

    // 返回值必须为MutableLiveData
    public MutableLiveData<Integer> getNum() {
        // 防止为空
        if (num == null) {
            num = new MutableLiveData<>();
            num.setValue(0);
        }
        return num;
    }

    public void add(int i) {
        int cur = getNum().getValue();
        num.setValue(cur + i);
```

```java
    }

    public void load() {
        // 从内部文件中读取数据
        dataHelper.loadFromIntervalFile(num);
    }

    public void save() {
        // 将数据保存内部文件中
        dataHelper.saveToIntervalFile(num);
    }
}

public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    MyViewModel myViewModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 创建MyViewModel
        myViewModel = new ViewModelProvider(this).get(MyViewModel.class);

        // 观测数据
        myViewModel.getNum().observe(this, new Observer<Integer>() {
            @Override
            public void onChanged(Integer integer) {
                textView.setText(String.valueOf(integer));
            }
        });

        myViewModel.load(); // 读取数据

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            myViewModel.add(1);
        });

        buttonMinus.setOnClickListener(v -> {
            myViewModel.add(-1);
        });
    }
}

// 重写onPause,永久保存数据
// 由于Activity可能会因为内存不足被杀死，所以重写onPause比重写onStop, onDestroy更安全可靠
@Override
protected void onPause() {
    super.onPause();
    myViewModel.save(); // 保存数据
```

```
}
```

外部存储：

```
public class DataHelper {
    private Application application;
    private String fileName = "myFile.txt";

    public DataHelper(Application application) {
        this.application = application;
    }

    // 将数据保存外部文件中
    void saveToExternalFile(MutableLiveData<Integer> num) {
        // 判断手机是否有SD卡并拥有可读写SD卡的权限
        if (Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED)) {
            try {
                File dir = Environment.getExternalStorageDirectory();
                File file = new File(dir, fileName);
                if (!file.exists()) {
                    file.createNewFile(); // 如果不存在指定的文件，需要创建该文件
                }
                // 获取文件输出流
                FileOutputStream fos = new FileOutputStream(file);
                String str = num.getValue().toString();
                // 将字符串转换成字节数组
                byte[] bytes = str.getBytes();
                // 将字节数组通过文件输出流写入文件
                fos.write(bytes);
                fos.close();
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        }
    }

    // 从外部文件中读取数据
    void loadFromExternalFile(MutableLiveData<Integer> num) {
        // 判断手机是否有SD卡并拥有可读写SD卡的权限
        if (Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED)) {
            try {
                // 获取外部存储的目录
                File dir = Environment.getExternalStorageDirectory();
                File file = new File(dir, fileName);
                if (!file.exists()) {
                    return; // 还没有存入数据，直接返回
                }
                // 获取文件输入流
                FileInputStream fis = new FileInputStream(file);
                // 返回要读取的剩余字节数
                int length = fis.available();
                byte[] bytes = new byte[length];
                // 将文件输入流读到的数据放到字节数组中
                fis.read(bytes);
                // 将字节数组转换成字符串
                String str = new String(bytes, 0, length);
                int x = Integer.parseInt(str);
                num.setValue(x);
                fis.close();
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        } else {
            Toast.makeText(application, "找不到SD卡或者没有获得读写SD卡的权限",
Toast.LENGTH_SHORT).show();
        }
    }
}

public class MyViewModel extends AndroidViewModel {
    // 使用MutableLiveData
    private MutableLiveData<Integer> num;
    private DataHelper dataHelper;

    public MyViewModel(@NonNull Application application) {
        super(application);
        // 创建DataHelper
        dataHelper = new DataHelper(application);
    }

    // 返回值必须为MutableLiveData
    public MutableLiveData<Integer> getNum() {
        // 防止为空
        if (num == null) {
            num = new MutableLiveData<>();
            num.setValue(0);
        }
        return num;
    }

    public void add(int i) {
        int cur = getNum().getValue();
        num.setValue(cur + i);
    }

    public void load() {
        // 从外部文件中读取数据
        dataHelper.loadFromExternalFile(num);
    }

    public void save() {
        // 将数据保存到外部文件中
        dataHelper.saveToExternalFile(num);
    }
}

public class MainActivity extends AppCompatActivity {
    TextView textView;
    Button buttonPlus, buttonMinus;
    MyViewModel myViewModel;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```java
        // 获取界面控件
        buttonPlus = findViewById(R.id.buttonPlus);
        buttonMinus = findViewById(R.id.buttonMinus);
        textView = findViewById(R.id.textView);

        // 创建MyViewModel
        myViewModel = new ViewModelProvider(this).get(MyViewModel.class);

        // 观测数据
        myViewModel.getNum().observe(this, new Observer<Integer>() {
            @Override
            public void onChanged(Integer integer) {
                textView.setText(String.valueOf(integer));
            }
        });

        getPermission();
        myViewModel.load(); // 读取数据

        // 事件监听
        buttonPlus.setOnClickListener(v -> {
            myViewModel.add(1);
        });

        buttonMinus.setOnClickListener(v -> {
            myViewModel.add(-1);
        });
    }
}

// 重写onPause,永久保存数据
// 由于Activity可能会因为内存不足被杀死，所以重写onPause比重写onStop, onDestroy更安全可靠
@Override
protected void onPause() {
    super.onPause();
    myViewModel.save(); // 保存数据
}

// 动态授权
private void getPermission() {
    String permisson = Manifest.permission.WRITE_EXTERNAL_STORAGE;
    // 判断是否已获得权限
    if (ContextCompat.checkSelfPermission(this, permisson)
        != PackageManager.PERMISSION_GRANTED) {
        //申请权限，会弹出对话框
        requestPermissions(new String[]{permisson}, 1);
    }
}

//处理权限结果回调
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if (requestCode == 1) {
        Toast.makeText(this, "获取权限成功！", Toast.LENGTH_SHORT).show();
    }
}
```