IA Calendar 技术文档

Team: FuFu.AI

上官子涵　西北工业大学
钟习伟　湖南大学
吴思贤　华南理工大学
何泓儒　华南理工大学
杨锦龙　华南理工大学

目录

1 AI 建议日程的实现

1.1 向 AI 发送请求

以下的代码展示了在应用中是如何创建请求，并接收 AI 的响应的。

Listing 1: 与蓝心大模型交互代码

```
public String callBlueHeartAIForAdvice(List<Map<String, String>> messages)
        throws IOException, NoSuchAlgorithmException, InvalidKeyException,
            JSONException {

    // 应用ID
    String appId = "2025510478";
    // 应用密钥
    String appKey = UserAppKey;
    // AI接口的URL
    String url = "https://api-ai.vivo.com.cn/vivogpt/completions";
    // 请求ID
    String requestId = UUID.randomUUID().toString();

    // 构建请求体
    JSONObject requestBody = new JSONObject();
    JSONArray messagesArray = new JSONArray();

    // 将消息列表添加到请求体中
    for (Map<String, String> message : messages) {
        JSONObject msgObj = new JSONObject();
        msgObj.put("role", message.get("role"));
        msgObj.put("content", message.get("content"));
        messagesArray.put(msgObj);
    }

    requestBody.put("messages", messagesArray);
    requestBody.put("model", "vivo-BlueLM-TB-Pro");
    requestBody.put("sessionId", UUID.randomUUID().toString()); // 使用新的会话
        ID

    // 生成签名头
    Map<String, String> headers = generateAuthHeaders(appId, appKey, "POST", "/
        vivogpt/completions",
            "requestId=" + requestId);

    // 发送请求
    HttpURLConnection connection = (HttpURLConnection) new URL(url + "?
        requestId=" + requestId).openConnection();
    connection.setRequestMethod("POST");
    connection.setRequestProperty("Content-Type", "application/json");

```

```
38          // 设置请求头
39          for (Map.Entry<String, String> entry : headers.entrySet()) {
40              connection.setRequestProperty(entry.getKey(), entry.getValue());
41          }
42
43          connection.setDoOutput(true);
44          try (OutputStream os = connection.getOutputStream()) {
45              byte[] input = requestBody.toString().getBytes(StandardCharsets.UTF_8);
46              os.write(input, 0, input.length);
47          }
48
49          // 获取响应
50          int responseCode = connection.getResponseCode();
51          if (responseCode == HttpURLConnection.HTTP_OK) {
52              try (BufferedReader br = new BufferedReader(
53                      new InputStreamReader(connection.getInputStream(),
                            StandardCharsets.UTF_8))) {
54                  StringBuilder response = new StringBuilder();
55                  String responseLine;
56                  while ((responseLine = br.readLine()) != null) {
57                      response.append(responseLine.trim());
58                  }
59
60                  // 解析响应内容
61                  JSONObject jsonResponse = new JSONObject(response.toString());
62                  if (jsonResponse.getInt("code") == 0) {
63                      return jsonResponse.getJSONObject("data").getString("content");
64                  } else {
65                      String errorMsg = jsonResponse.getString("msg");
66                      throw new IOException("AI服务错误:␣" + errorMsg);
67                  }
68              }
69          } else {
70              // 读取错误流
71              try (BufferedReader br = new BufferedReader(
72                      new InputStreamReader(connection.getErrorStream(),
                            StandardCharsets.UTF_8))) {
73              StringBuilder errorResponse = new StringBuilder();
74              String line;
75              while ((line = br.readLine()) != null) {
76                  errorResponse.append(line);
77              }
78              throw new IOException("HTTP错误:␣" + responseCode + "\n" +
                    errorResponse);
79          }
80      }
81  }
```

1.2 AI 响应的处理

在获取 AI 的响应后，我们需要对其进行处理，以便将结果展示给用户。以下代码展示了如何解析 AI 的响应，并将其转换为可视化的日历事件。

此外，为应对可能出现的各种时间格式，设置了多种情况的处理逻辑，以确保 AI 给出的日程安排能够被正确解析和展示。

Listing 2: 日程解析代码

```
@RequiresApi(api = Build.VERSION_CODES.O)
private List<Event> parseEvents(String aiResponse) {
    Log.d("AIFragment", "开始解析AI响应:␣" + aiResponse);

    List<Event> events = new ArrayList<>();
    String[] lines = aiResponse.split("\n");
    int eventCount = 0;
    LocalDate date = LocalDate.now();

    // 尝试从响应中提取日期
    Pattern datePattern = Pattern.compile("(\\d{4}年\\d{1,2}月\\d{1,2}日|\\d
        {4}-\\d{2}-\\d{2})");
    Matcher dateMatcher = datePattern.matcher(aiResponse);
    if (dateMatcher.find()) {
        try {
            String dateStr = dateMatcher.group();
            // 处理不同日期格式
            if (dateStr.contains("年")) {
                date = LocalDate.parse(dateStr, DateTimeFormatter.ofPattern("
                    yyyy年MM月dd日"));
            } else {
                date = LocalDate.parse(dateStr);
            }
            Log.d("AIFragment", "从响应中解析到日期:␣" + date);
        } catch (Exception e) {
            Log.e("AIFragment", "日期解析错误:␣" + e.getMessage());
            Toast.makeText(getContext(), "日期解析错误，使用默认日期", Toast.
                LENGTH_SHORT).show();
        }
    }

    // 事件解析模式 - 支持多种时间格式
    Pattern eventPattern = Pattern.compile(
            "(\\d{1,2})[:: ](\\d{2})\\s*[-~—]\\s*(\\d{1,2})[:: ](\\d{2})\\s*[:
                : ]?\\s*(.+)"
    );

    for (String line : lines) {
        Matcher matcher = eventPattern.matcher(line);
        if (matcher.find()) {
```

```
                try {
                    // 解析时间
                    int startHour = Integer.parseInt(matcher.group(1));
                    int startMin = Integer.parseInt(matcher.group(2));
                    int endHour = Integer.parseInt(matcher.group(3));
                    int endMin = Integer.parseInt(matcher.group(4));
                    String description = matcher.group(5).trim();

                    // 创建时间对象
                    LocalTime startTime = LocalTime.of(startHour, startMin);
                    LocalTime endTime = LocalTime.of(endHour, endMin);

                    // 创建事件
                    Event event = new Event(date, startTime, endTime, description);
                    events.add(event);
                    eventCount++;

                    Log.d("AIFragment", "解析到事件:␣" + event);

                } catch (Exception e) {
                    Log.e("AIFragment", "事件解析错误:␣" + line, e);
                }
            } else {
                // 尝试其他格式: 没有冒号的时间格式 (如 0800-0930)
                Pattern altPattern = Pattern.compile("(\\d{2})(\\d{2})\\s*[-~—]\\s
                    *(\\d{2})(\\d{2})\\s*[:: ]?\\s*(.+)");
                Matcher altMatcher = altPattern.matcher(line);
                if (altMatcher.find()) {
                    try {
                        int startHour = Integer.parseInt(altMatcher.group(1));
                        int startMin = Integer.parseInt(altMatcher.group(2));
                        int endHour = Integer.parseInt(altMatcher.group(3));
                        int endMin = Integer.parseInt(altMatcher.group(4));
                        String description = altMatcher.group(5).trim();

                        LocalTime startTime = LocalTime.of(startHour, startMin);
                        LocalTime endTime = LocalTime.of(endHour, endMin);

                        Event event = new Event(date, startTime, endTime,
                            description);
                        events.add(event);
                        eventCount++;

                        Log.d("AIFragment", "解析到事件(替代格式):␣" + event);
                    } catch (Exception e) {
                        Log.e("AIFragment", "替代格式事件解析错误:␣" + line, e);
                    }
                }
```

```
83              }
84          }
85
86          Log.d("AIFragment", "成功解析␣" + events.size() + "␣个事件");
87          return events;
88      }
```

2 日历的操作与显示

2.1 绘制空白单元格

在日历中，我们需要绘制空白的单元格以便于用户查看和后续在单元格上绘制日期与当日事件。以下代码展示了如何在日历中绘制空白单元格。

Listing 3: 绘制空白单元格代码

```
1   private void addEmptyCell() {
2       // 功能：创建一个空白单元格并添加到日历网格中
3       View emptyView = LayoutInflater.from(getContext())
4               .inflate(R.layout.calendar_day_cell, calendarGrid, false);
5       TextView dayNumber = emptyView.findViewById(R.id.dayNumber);
6       dayNumber.setText("");
7       dayNumber.setVisibility(View.INVISIBLE);
8       emptyView.setBackgroundColor(ContextCompat.getColor(getContext(), R.color.
            empty_cell_color));
9       GridLayout.LayoutParams params = new GridLayout.LayoutParams();
10      params.width = 0;
11      params.height = 0; // 使用权重控制高度
12      params.columnSpec = GridLayout.spec(GridLayout.UNDEFINED, 1f);
13      params.rowSpec = GridLayout.spec(GridLayout.UNDEFINED, 1f); // 行权重为1
14      emptyView.setLayoutParams(params);
15
16      calendarGrid.addView(emptyView);
17  }
```

2.2 单元格事件的显示

在日历中，我们需要将已添加的日程事件显示在对应的单元格中。

以下代码展示了如何从 *SharedPreferences* 加载有效事件，并将事件绘制到日历的单元格中。

Listing 4: 单元格事件显示代码

```
private void showEventsInCell(TextView eventsText, List<Event> dayEvents, View dayView) {
    // 功能：在日期单元格中显示当天的事件，最多显示指定数量的事件，并添加更多事件提示和无障碍支持
    StringBuilder eventsBuilder = new StringBuilder();
    int maxEventsToShow = 1; // 最多显示2个事件
    int maxCharsPerEvent = 3; // 每个事件最多显示15个字符

    for (int i = 0; i < Math.min(dayEvents.size(), maxEventsToShow); i++) {
        Event event = dayEvents.get(i);

        // 截断长文本
        String eventDesc = event.getDescription();
        if (eventDesc.length() > maxCharsPerEvent) {
            eventDesc = eventDesc.substring(0, maxCharsPerEvent) + "…";
        }

        // 添加时间
        String eventStr = "";
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            eventStr = event.getStartTime() + "␣" + eventDesc;
        }
        eventsBuilder.append("•␣").append(eventStr).append("\n");
    }

    // 添加更多事件提示
    if (dayEvents.size() > maxEventsToShow) {
        eventsBuilder.append("+")
                .append(dayEvents.size() - maxEventsToShow)
                .append("更多");
    }

    eventsText.setText(eventsBuilder.toString());

    // 添加无障碍支持
    StringBuilder fullDescription = new StringBuilder("包含" + dayEvents.size() + "个事件");
    for (Event e : dayEvents) {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            fullDescription.append(", ").append(e.getStartTime()).append(e.getDescription());
        }
```

```
        }
        dayView.setContentDescription(fullDescription.toString());
    }

    public void addEvent(Event event) {
        // 功能：将新事件添加到日历中，并更新日历显示
        // 使用自定义的 Gson 实例
        Gson gson = GsonUtils.getGson();

        SharedPreferences prefs = requireActivity().getSharedPreferences(
            EVENTS_PREFS, Context.MODE_PRIVATE);
        String eventJson = prefs.getString(event.getDate().toString(), "[]");
        Type type = new TypeToken<ArrayList<Event>>(){}.getType();
        List<Event> events = gson.fromJson(eventJson, type);

        // 添加新事件
        events.add(event);

        // 保存回SharedPreferences
        String updatedJson = gson.toJson(events);
        prefs.edit().putString(event.getDate().toString(), updatedJson).apply();

        updateCalendar(); // 更新日历显示
        Toast.makeText(getContext(), "事件已添加", Toast.LENGTH_SHORT).show();
    }

    private Map<LocalDate, List<Event>> loadAllEvents() {
        // 功能：加载所有事件，并过滤无效事件
        Map<LocalDate, List<Event>> eventsMap = new HashMap<>();
        SharedPreferences prefs = requireActivity().getSharedPreferences(
            EVENTS_PREFS, Context.MODE_PRIVATE);
        Map<String, ?> allEntries = prefs.getAll();

        Gson gson = GsonUtils.getGson();
        Type type = new TypeToken<ArrayList<Event>>(){}.getType();

        for (Map.Entry<String, ?> entry : allEntries.entrySet()) {
            try {
                LocalDate date = null;
                if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
                    date = LocalDate.parse(entry.getKey());
                }
                List<Event> events = gson.fromJson(entry.getValue().toString(),
                    type);

                // 过滤无效事件
                List<Event> validEvents = new ArrayList<>();
                for (Event event : events) {
```

```
                    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
                        if (event.getDate() != null && event.getStartTime() != null
                                && event.getEndTime() != null) {
                            validEvents.add(event);
                        }
                    }
                }

                eventsMap.put(date, validEvents);
            } catch (Exception e) {
                Log.e(TAG, "加载事件错误:␣" + entry.getKey(), e);
            }
        }
        return eventsMap;
    }
```

在日历界面中，使用如下 xml 代码绘制日历网格：

Listing 5: 网格绘制

```
    <GridLayout
        android:id="@+id/calendarGrid"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:columnCount="7"
        android:rowCount="6"
        android:padding="1dp"
        android:background="@drawable/bg_rounded_card"/>
```

3 今日日程与建议

3.1 当日日程的获取

以下的代码展示了如何刷新今日日程，加载今日的事件并更新 RecyclerView 的数据。同时更新任务数据汇总和进度显示。

Listing 6: 刷新今日日程的代码

```
    /**
     * 核心函数：刷新今日日程，加载今日的事件并更新 RecyclerView 的数据。
     * 同时更新任务数据汇总和进度显示。
     */
    @RequiresApi(api = Build.VERSION_CODES.O)
    void refreshEvents() {
        Log.d(TAG, "开始刷新今日日程");

        // 检查必要的视图和上下文是否初始化
        if (eventsRecyclerView == null || swipeRefreshLayout == null || getContext
            () == null) {
            Log.e(TAG, "视图未初始化，无法刷新");
            if (swipeRefreshLayout != null) {
                swipeRefreshLayout.setRefreshing(false);
            }

            return;
        }

        LocalDate today = LocalDate.now();
        List<Event> events = loadEventsForDate(today);
        events.sort(Comparator.comparing(Event::getStartTime)); // 使用新的 getter

        Log.d(TAG, "找到␣" + events.size() + "␣个今日事件");

        // 更新适配器数据
        eventsList.clear();
        eventsList.addAll(events);
        eventAdapter.notifyDataSetChanged();

        // 如果没有事件，显示空视图提示
        if (events.isEmpty()) {
            // 创建并设置空视图
            TextView emptyView = new TextView(getContext());
            emptyView.setText("今日无日程安排");
            emptyView.setContentDescription("今日无日程安排");
            emptyView.setImportantForAccessibility(View.
                IMPORTANT_FOR_ACCESSIBILITY_YES);
            emptyView.setTextSize(18);
            emptyView.setGravity(android.view.Gravity.CENTER);
```

```
            emptyView.setPadding(0, 32, 0, 32);

            // 添加到 RecyclerView 的父容器中
            if (eventsRecyclerView.getParent() instanceof ViewGroup) {
                ViewGroup parent = (ViewGroup) eventsRecyclerView.getParent();
                // 移除之前的空视图（如果存在）
                for (int i = 0; i < parent.getChildCount(); i++) {
                    if (parent.getChildAt(i) instanceof TextView) {
                        parent.removeViewAt(i);
                        break;
                    }
                }
                parent.addView(emptyView);
            }
        } else {
            // 移除空视图（如果存在）
            if (eventsRecyclerView.getParent() instanceof ViewGroup) {
                ViewGroup parent = (ViewGroup) eventsRecyclerView.getParent();
                for (int i = 0; i < parent.getChildCount(); i++) {
                    if (parent.getChildAt(i) instanceof TextView) {
                        parent.removeViewAt(i);
                        break;
                    }
                }
            }
        }

        updateDataSummary();
        // 停止刷新动画
        swipeRefreshLayout.setRefreshing(false);
        Log.d(TAG, "今日日程刷新完成");
    }
```

3.2 获取近七天日程的建议

在应用中，我们需要获取近七天的日程安排，并将其发送给 AI 以获取建议。以下代码展示了获取近七天的日程，并将其发送给 AI 进行处理。

此外，每日零点均会自动刷新获取 AI 建议的闹钟，以保证建议始终最新。

Listing 7: 获取日程建议的交互代码

```
    private String getAIAdvice(String prompt) {
        try {
            // 创建消息列表
            AIFragment aiFragment = new AIFragment();
            List<Map<String, String>> messages = new ArrayList<>();

            // 系统提示
            Map<String, String> systemMessage = new HashMap<>();
            systemMessage.put("role", "system");
            systemMessage.put("content", "你是一个专业的日程管理助手，请根据用户提
                供的最近␣7␣天日程数据，提供专业、简洁、实用的建议。建议应包含时间管
                理、效率提升、健康提醒等方面。");
            messages.add(systemMessage);

            // 用户请求
            Map<String, String> userMessage = new HashMap<>();
            userMessage.put("role", "user");
            userMessage.put("content", prompt);
            messages.add(userMessage);

            // 发送请求 - 这里使用 AIFragment 中的请求逻辑
            return aiFragment.callBlueHeartAIForAdvice(messages);
        } catch (Exception e) {
            // 处理异常，返回错误信息
            return "获取建议时出错:␣" + e.getMessage();
        }
    }

    @RequiresApi(api = Build.VERSION_CODES.O)
    private String getRecentScheduleData() {
        // 尝试从缓存加载
        SharedPreferences prefs = requireContext().getSharedPreferences(
            PREFS_ADVICE, Context.MODE_PRIVATE);
        String cachedData = prefs.getString(KEY_LAST_ADVICE_DATA, null);

        if (cachedData != null) {
            // 检查缓存数据是否过期（超过 1 天）
            String lastDate = prefs.getString(KEY_LAST_ADVICE_DATE, "");
            LocalDate today = LocalDate.now();

            if (today.toString().equals(lastDate)) {
```

```
39                  return cachedData;
40              }
41          }
42      }
```

3.3 页面可视化

Listing 8: 绘制今日日程页面

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp"
    android:background="@color/azure"
    android:contentDescription="今日日程页面">

    <!-- 顶部操作栏 -->
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp">

        <!-- 日期和标题 -->
        <LinearLayout
            android:layout_width="259dp"
            android:layout_height="wrap_content"
            android:layout_marginStart="10dp"
            android:layout_marginBottom="10dp"
            android:orientation="horizontal"
            android:paddingTop="8dp"
            android:paddingBottom="23dp">

            <TextView
                android:id="@+id/today_title"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginEnd="11dp"
                android:text="Today"
                android:textColor="#000000"
                android:textSize="24sp"
                android:textStyle="bold" />

            <TextView
                android:id="@+id/today_date"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:contentDescription="当前日期"
                android:textColor="#818181"
                android:textSize="21sp"
```

```
                android:textStyle="bold" />

        </LinearLayout>

        <!-- 右上角操作按钮 -->
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentEnd="true"
            android:orientation="horizontal">

            <!-- 添加事件按钮 -->
            <ImageView
                android:id="@+id/add_event_btn"
                android:layout_width="32dp"
                android:layout_height="32dp"
                android:src="@drawable/ic_add"
                android:contentDescription="添加事件"
                android:layout_marginEnd="16dp"
                android:padding="4dp"/>

            <!-- 设置按钮 -->
            <ImageView
                android:id="@+id/settings_btn"
                android:layout_width="32dp"
                android:layout_height="32dp"
                android:src="@drawable/ic_settings"
                android:contentDescription="设置"
                android:padding="4dp"/>
        </LinearLayout>
    </RelativeLayout>

    <!-- 第一部分：日程展示区 (占1/3高度) -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:orientation="vertical"
        android:layout_marginBottom="16dp"
        android:elevation="4dp">

        <TextView
            android:id="@+id/date_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="今日日程"
            android:textColor="#000000"
            android:textSize="16sp"
```

```
            android:textStyle="bold" />

        <androidx.swiperefreshlayout.widget.SwipeRefreshLayout
            android:id="@+id/swipeRefreshLayout"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:contentDescription="今日日程列表区域，支持下拉刷新">

            <androidx.recyclerview.widget.RecyclerView
                android:id="@+id/events_recycler_view"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:padding="8dp"
                android:scrollbars="vertical" />
        </androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
    </LinearLayout>

    <!-- 第二部分：数据汇总区 (占1/3高度) -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:orientation="vertical"
        android:layout_marginBottom="16dp">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="@drawable/cr10lr270cbf2f9b3fde4f7"
            android:padding="16dp"
            android:elevation="4dp">

            <LinearLayout
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:orientation="vertical"
                android:gravity="center_vertical">

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginStart="0dp"
                    android:layout_marginBottom="4dp"
                    android:text="本周"
                    android:textColor="#818181"
                    android:textSize="16sp" />

```

```
                <TextView
                    android:id="@+id/week_range"
                    android:layout_width="0dp"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:gravity="end"
                    android:text="（日期范围）"/>

                <TextView
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:text="数据汇总"
                    android:textSize="20sp"
                    android:textColor="@color/teal_700"
                    android:textStyle="bold"
                    android:layout_marginBottom="16dp"/>

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:orientation="horizontal"
                    android:layout_marginBottom="8dp">

                    <ImageView
                        android:layout_width="16dp"
                        android:layout_height="16dp"
                        android:src="@drawable/ic_circle"
                        android:layout_marginEnd="8dp"
                        app:tint="@color/teal_700" />

                    <TextView
                        android:id="@+id/tasks_remaining"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="剩余任务: 0"
                        android:textSize="14sp"
                        android:textColor="@color/teal_700"/>
                </LinearLayout>

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:orientation="horizontal">

                    <ImageView
                        android:layout_width="16dp"
                        android:layout_height="16dp"
                        android:src="@drawable/ic_circle"
```

```
                        android:layout_marginEnd="8dp"
                        app:tint="@color/teal_700" />

                    <TextView
                        android:id="@+id/tasks_completed"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:text="结束任务: 0"
                        android:textSize="14sp"
                        android:textColor="@color/teal_700"/>
                </LinearLayout>

                <TextView
                    android:id="@+id/more_details"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_marginTop="16dp"
                    android:text="Details␣>>"
                    android:textColor="@color/teal_700"
                    android:textSize="14sp" />
            </LinearLayout>

            <LinearLayout
                android:layout_width="0dp"
                android:layout_height="match_parent"
                android:layout_weight="1"
                android:orientation="vertical"
                android:gravity="center">

                <com.example.calendarapp.CircularProgressView
                    android:id="@+id/progress_circle"
                    android:layout_width="100dp"
                    android:layout_height="100dp"
                    app:progress="25"
                    app:progressColor="@color/turquoise4"
                    app:bgColor="@color/azure"/>

                <TextView
                    android:id="@+id/progress_text"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="25%"
                    android:textSize="16sp"
                    android:textColor="@color/teal_700"
                    android:layout_marginTop="8dp"/>
            </LinearLayout>
        </LinearLayout>
    </LinearLayout>
```

```
237
238     <!-- 第三部分：AI建议区（占1/3高度） -->
239     <LinearLayout
240         android:layout_width="match_parent"
241         android:layout_height="0dp"
242         android:layout_weight="1"
243         android:orientation="vertical">
244
245         <TextView
246             android:layout_width="wrap_content"
247             android:layout_height="wrap_content"
248             android:text="日程建议"
249             android:textColor="#000000"
250             android:textSize="16sp"
251             android:textStyle="bold"
252             android:layout_marginStart="8dp"
253             android:layout_marginBottom="4dp"/>
254
255         <RelativeLayout
256             android:layout_width="match_parent"
257             android:layout_height="match_parent"
258             android:background="@drawable/cr10lr270f7fed5fde4f7"
259             android:padding="16dp"
260             android:elevation="4dp">
261
262             <!-- 添加ScrollView使内容可滚动 -->
263             <ScrollView
264                 android:layout_width="match_parent"
265                 android:layout_height="match_parent"
266                 android:fillViewport="true"
267                 android:scrollbars="vertical">
268
269                 <LinearLayout
270                     android:layout_width="match_parent"
271                     android:layout_height="wrap_content"
272                     android:orientation="vertical">
273
274                     <TextView
275                         android:id="@+id/ai_advice_text"
276                         android:layout_width="match_parent"
277                         android:layout_height="wrap_content"
278                         android:text="正在分析您的日程..."
279                         android:textSize="16sp"
280                         android:textColor="@color/teal_700"
281                         android:padding="8dp"/>
282                 </LinearLayout>
283             </ScrollView>
284
```

```
285             <!-- 刷新按钮保持在右下角 -->
286             <ImageButton
287                 android:id="@+id/refresh_advice_btn"
288                 android:layout_width="32dp"
289                 android:layout_height="32dp"
290                 android:layout_alignParentEnd="true"
291                 android:layout_alignParentBottom="true"
292                 android:layout_margin="8dp"
293                 android:background="?attr/selectableItemBackgroundBorderless"
294                 android:contentDescription="刷新建议"
295                 android:src="@drawable/ic_refresh"
296                 tools:ignore="TouchTargetSizeCheck" />
297         </RelativeLayout>
298     </LinearLayout>
299 </LinearLayout>
```