



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
1.1	向 AI 发送请求	1
1.2	AI 响应的处理	3
2	日历的操作与显示	6
2.1	绘制空白单元格	6
2.2	单元格事件的显示	7
3	今日日程与建议	10
3.1	当日日程的获取	10
3.2	获取近七天日程的建议	12
3.3	页面可视化	14

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

```
1 public String callBlueHeartAIForAdvice(List<Map<String, String>> messages)
2     throws IOException, NoSuchAlgorithmException, InvalidKeyException,
3         JSONException {
4
5     // 应用 ID
6     String appId = "2025510478";
7     // 应用密钥
8     String appKey = UserAppKey;
9     // AI接口的 URL
10    String url = "https://api-ai.vivo.com.cn/vivogpt/completions";
11    // 请求 ID
12    String requestId = UUID.randomUUID().toString();
13
14    // 构建请求体
15    JSONObject requestBody = new JSONObject();
16    JSONArray messagesArray = new JSONArray();
17
18    // 将消息列表添加到请求体中
19    for (Map<String, String> message : messages) {
20        JSONObject msgObj = new JSONObject();
21        msgObj.put("role", message.get("role"));
22        msgObj.put("content", message.get("content"));
23        messagesArray.put(msgObj);
24    }
25
26    requestBody.put("messages", messagesArray);
27    requestBody.put("model", "vivo-BlueLM-TB-Pro");
28    requestBody.put("sessionId", UUID.randomUUID().toString()); // 使用新的会话 ID
29
30    // 生成签名头
31    Map<String, String> headers = generateAuthHeaders(appId, appKey, "POST", "/" +
32        vivogpt/completions",
33        "requestId=" + requestId);
34
35    // 发送请求
36    HttpURLConnection connection = (HttpURLConnection) new URL(url + "?" +
37        requestId=" + requestId).openConnection();
38    connection.setRequestMethod("POST");
39    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(),
54             StandardCharsets.UTF_8))) {
55         StringBuilder response = new StringBuilder();
56         String responseLine;
57         while ((responseLine = br.readLine()) != null) {
58             response.append(responseLine.trim());
59         }
60
61         // 解析响应内容
62         JSONObject jsonResponse = new JSONObject(response.toString());
63         if (jsonResponse.getInt("code") == 0) {
64             return jsonResponse.getJSONObject("data").getString("content");
65         } else {
66             String errorMsg = jsonResponse.getString("msg");
67             throw new IOException("AI服务错误:␣" + errorMsg);
68         }
69     } else {
70         // 读取错误流
71         try (BufferedReader br = new BufferedReader(
72             new InputStreamReader(connection.getErrorStream(),
73                 StandardCharsets.UTF_8))) {
74             StringBuilder errorResponse = new StringBuilder();
75             String line;
76             while ((line = br.readLine()) != null) {
77                 errorResponse.append(line);
78             }
79             throw new IOException("HTTP 错误:␣" + responseCode + "\n" +
80                 errorResponse);
81         }
82     }
83 }
```

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

```
1  @RequiresApi(api = Build.VERSION_CODES.O)
2  private List<Event> parseEvents(String aiResponse) {
3      Log.d("AIFragment", "开始解析AI响应:␣" + aiResponse);
4
5      List<Event> events = new ArrayList<>();
6      String[] lines = aiResponse.split("\n");
7      int eventCount = 0;
8      LocalDate date = LocalDate.now();
9
10     // 尝试从响应中提取日期
11     Pattern datePattern = Pattern.compile("(\\d{4}年\\d{1,2}月\\d{1,2}日|\\d{4}-\\d{2}-\\d{2})");
12     Matcher dateMatcher = datePattern.matcher(aiResponse);
13     if (dateMatcher.find()) {
14         try {
15             String dateStr = dateMatcher.group();
16             // 处理不同日期格式
17             if (dateStr.contains("年")) {
18                 date = LocalDate.parse(dateStr, DateTimeFormatter.ofPattern("yyyy年MM月dd日"));
19             } else {
20                 date = LocalDate.parse(dateStr);
21             }
22             Log.d("AIFragment", "从响应中解析到日期:␣" + date);
23         } catch (Exception e) {
24             Log.e("AIFragment", "日期解析错误:␣" + e.getMessage());
25             Toast.makeText(getContext(), "日期解析错误，使用默认日期", Toast.LENGTH_SHORT).show();
26         }
27     }
28
29     // 事件解析模式 - 支持多种时间格式
30     Pattern eventPattern = Pattern.compile(
31         "(\\d{1,2})[: ](\\d{2})\\s*[-—]\\s*(\\d{1,2})[: ](\\d{2})\\s*[: ]?\\s*(.+)"
32     );
33
34     for (String line : lines) {
35         Matcher matcher = eventPattern.matcher(line);
36         if (matcher.find()) {
```

```
37         try {
38             // 解析时间
39             int startHour = Integer.parseInt(matcher.group(1));
40             int startMin = Integer.parseInt(matcher.group(2));
41             int endHour = Integer.parseInt(matcher.group(3));
42             int endMin = Integer.parseInt(matcher.group(4));
43             String description = matcher.group(5).trim();
44
45             // 创建时间对象
46             LocalTime startTime = LocalTime.of(startHour, startMin);
47             LocalTime endTime = LocalTime.of(endHour, endMin);
48
49             // 创建事件
50             Event event = new Event(date, startTime, endTime, description);
51             events.add(event);
52             eventCount++;
53
54             Log.d("AIFragment", "解析到事件:␣" + event);
55
56         } catch (Exception e) {
57             Log.e("AIFragment", "事件解析错误:␣" + line, e);
58         }
59     } else {
60         // 尝试其他格式: 没有冒号的时间格式 (如 0800-0930)
61         Pattern altPattern = Pattern.compile("(\\d{2})(\\d{2})\\s*[-~—]\\s*
62             *(\\d{2})(\\d{2})\\s*[:. ]?\\s*(.+)");
63         Matcher altMatcher = altPattern.matcher(line);
64         if (altMatcher.find()) {
65             try {
66                 int startHour = Integer.parseInt(altMatcher.group(1));
67                 int startMin = Integer.parseInt(altMatcher.group(2));
68                 int endHour = Integer.parseInt(altMatcher.group(3));
69                 int endMin = Integer.parseInt(altMatcher.group(4));
70                 String description = altMatcher.group(5).trim();
71
72                 LocalTime startTime = LocalTime.of(startHour, startMin);
73                 LocalTime endTime = LocalTime.of(endHour, endMin);
74
75                 Event event = new Event(date, startTime, endTime,
76                     description);
77                 events.add(event);
78                 eventCount++;
79
80                 Log.d("AIFragment", "解析到事件(替代格式):␣" + event);
81             } catch (Exception e) {
82                 Log.e("AIFragment", "替代格式事件解析错误:␣" + line, e);
83             }
84         }
85     }
86 }
```

```
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.
9         empty_cell_color));
10    GridLayout.LayoutParams params = new GridLayout.LayoutParams();
11    params.width = 0;
12    params.height = 0; // 使用权重控制高度
13    params.columnSpec = GridLayout.spec(GridLayout.UNDEFINED, 1f);
14    params.rowSpec = GridLayout.spec(GridLayout.UNDEFINED, 1f); // 行权重为1
15    emptyView.setLayoutParams(params);
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

```
1 private void showEventsInCell(TextView eventsText, List<Event> dayEvents, View
   dayView) {
2     // 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
3     StringBuilder eventsBuilder = new StringBuilder();
4     int maxEventsToShow = 1; // 最多显示2个事件
5     int maxCharsPerEvent = 3; // 每个事件最多显示15个字符
6
7     for (int i = 0; i < Math.min(dayEvents.size(), maxEventsToShow); i++) {
8         Event event = dayEvents.get(i);
9
10        // 截断长文本
11        String eventDesc = event.getDescription();
12        if (eventDesc.length() > maxCharsPerEvent) {
13            eventDesc = eventDesc.substring(0, maxCharsPerEvent) + "...";
14        }
15
16        // 添加时间
17        String eventStr = "";
18        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
19            eventStr = event.getStartTime() + "␣" + eventDesc;
20        }
21        eventsBuilder.append("•␣").append(eventStr).append("\n");
22    }
23
24    // 添加更多事件提示
25    if (dayEvents.size() > maxEventsToShow) {
26        eventsBuilder.append("+")
27            .append(dayEvents.size() - maxEventsToShow)
28            .append("更多");
29    }
30
31    eventsText.setText(eventsBuilder.toString());
32
33    // 添加无障碍支持
34    StringBuilder fullDescription = new StringBuilder("包含" + dayEvents.size()
        + "个事件");
35    for (Event e : dayEvents) {
36        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
37            fullDescription.append(", ").append(e.getStartTime()).append(e.
                getDescription());
38        }
39    }
```

```
39     }
40     dayView.setContentDescription(fullDescription.toString());
41 }
42
43 public void addEvent(Event event) {
44     // 功能：将新事件添加到日历中，并更新日历显示
45     // 使用自定义的 Gson 实例
46     Gson gson = GsonUtils.getGson();
47
48     SharedPreferences prefs = requireActivity().getSharedPreferences(
49         EVENTS_PREFS, Context.MODE_PRIVATE);
50     String eventJson = prefs.getString(event.getDate().toString(), "[]");
51     Type type = new TypeToken<ArrayList<Event>>(){}.getType();
52     List<Event> events = gson.fromJson(eventJson, type);
53
54     // 添加新事件
55     events.add(event);
56
57     // 保存回 SharedPreferences
58     String updatedJson = gson.toJson(events);
59     prefs.edit().putString(event.getDate().toString(), updatedJson).apply();
60
61     updateCalendar(); // 更新日历显示
62     Toast.makeText(getContext(), "事件已添加", Toast.LENGTH_SHORT).show();
63 }
64
65 private Map<LocalDate, List<Event>> loadAllEvents() {
66     // 功能：加载所有事件，并过滤无效事件
67     Map<LocalDate, List<Event>> eventsMap = new HashMap<>();
68     SharedPreferences prefs = requireActivity().getSharedPreferences(
69         EVENTS_PREFS, Context.MODE_PRIVATE);
70     Map<String, ?> allEntries = prefs.getAll();
71
72     Gson gson = GsonUtils.getGson();
73     Type type = new TypeToken<ArrayList<Event>>(){}.getType();
74
75     for (Map.Entry<String, ?> entry : allEntries.entrySet()) {
76         try {
77             LocalDate date = null;
78             if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
79                 date = LocalDate.parse(entry.getKey());
80             }
81             List<Event> events = gson.fromJson(entry.getValue().toString(),
82                 type);
83
84             // 过滤无效事件
85             List<Event> validEvents = new ArrayList<>();
86             for (Event event : events) {
```

```
84         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
85             if (event.getDate() != null && event.getStartTime() != null
86                 && event.getEndTime() != null) {
87                 validEvents.add(event);
88             }
89         }
90
91         eventsMap.put(date, validEvents);
92     } catch (Exception e) {
93         Log.e(TAG, "加载事件错误:␣" + entry.getKey(), e);
94     }
95 }
96 return eventsMap;
97 }
```

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

```
1 <GridLayout
2     android:id="@+id/calendarGrid"
3     android:layout_width="match_parent"
4     android:layout_height="match_parent"
5     android:columnCount="7"
6     android:rowCount="6"
7     android:padding="1dp"
8     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

```
1  /**
2   * 核心函数：刷新今日日程，加载今日的事件并更新 RecyclerView 的数据。
3   * 同时更新任务数据汇总和进度显示。
4   */
5  @RequiresApi(api = Build.VERSION_CODES.O)
6  void refreshEvents() {
7      Log.d(TAG, "开始刷新今日日程");
8
9      // 检查必要的视图和上下文是否初始化
10     if (eventsRecyclerView == null || swipeRefreshLayout == null || getContext() == null) {
11         Log.e(TAG, "视图未初始化，无法刷新");
12         if (swipeRefreshLayout != null) {
13             swipeRefreshLayout.setRefreshing(false);
14         }
15
16         return;
17     }
18
19     LocalDate today = LocalDate.now();
20     List<Event> events = loadEventsForDate(today);
21     events.sort(Comparator.comparing(Event::getStartTime)); // 使用新的 getter
22
23     Log.d(TAG, "找到" + events.size() + "个今日事件");
24
25     // 更新适配器数据
26     eventsList.clear();
27     eventsList.addAll(events);
28     eventAdapter.notifyDataSetChanged();
29
30     // 如果没有事件，显示空视图提示
31     if (events.isEmpty()) {
32         // 创建并设置空视图
33         TextView emptyView = new TextView(getContext());
34         emptyView.setText("今日无日程安排");
35         emptyView.setContentDescription("今日无日程安排");
36         emptyView.setImportantForAccessibility(View.
37             IMPORTANT_FOR_ACCESSIBILITY_YES);
38         emptyView.setTextSize(18);
39         emptyView.setGravity(android.view.Gravity.CENTER);
```

```
39         emptyView.setPadding(0, 32, 0, 32);
40
41         // 添加到 RecyclerView 的父容器中
42         if (eventsRecyclerView.getParent() instanceof ViewGroup) {
43             ViewGroup parent = (ViewGroup) eventsRecyclerView.getParent();
44             // 移除之前的空视图（如果存在）
45             for (int i = 0; i < parent.getChildCount(); i++) {
46                 if (parent.getChildAt(i) instanceof TextView) {
47                     parent.removeViewAt(i);
48                     break;
49                 }
50             }
51             parent.addView(emptyView);
52         }
53     } else {
54         // 移除空视图（如果存在）
55         if (eventsRecyclerView.getParent() instanceof ViewGroup) {
56             ViewGroup parent = (ViewGroup) eventsRecyclerView.getParent();
57             for (int i = 0; i < parent.getChildCount(); i++) {
58                 if (parent.getChildAt(i) instanceof TextView) {
59                     parent.removeViewAt(i);
60                     break;
61                 }
62             }
63         }
64     }
65
66     updateDataSummary();
67     // 停止刷新动画
68     swipeRefreshLayout.setRefreshing(false);
69     Log.d(TAG, "今日日程刷新完成");
70 }
```

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

```
1 private String getAIAdvice(String prompt) {
2     try {
3         // 创建消息列表
4         AIFragment aiFragment = new AIFragment();
5         List<Map<String, String>> messages = new ArrayList<>();
6
7         // 系统提示
8         Map<String, String> systemMessage = new HashMap<>();
9         systemMessage.put("role", "system");
10        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。");
11        messages.add(systemMessage);
12
13        // 用户请求
14        Map<String, String> userMessage = new HashMap<>();
15        userMessage.put("role", "user");
16        userMessage.put("content", prompt);
17        messages.add(userMessage);
18
19        // 发送请求 - 这里使用 AIFragment 中的请求逻辑
20        return aiFragment.callBlueHeartAIForAdvice(messages);
21    } catch (Exception e) {
22        // 处理异常，返回错误信息
23        return "获取建议时出错：" + e.getMessage();
24    }
25 }
26
27 @RequiresApi(api = Build.VERSION_CODES.O)
28 private String getRecentScheduleData() {
29     // 尝试从缓存加载
30     SharedPreferences prefs = requireContext().getSharedPreferences(
31         PREFS_ADVICE, Context.MODE_PRIVATE);
32     String cachedData = prefs.getString(KEY_LAST_ADVICE_DATA, null);
33
34     if (cachedData != null) {
35         // 检查缓存数据是否过期（超过 1 天）
36         String lastDate = prefs.getString(KEY_LAST_ADVICE_DATE, "");
37         LocalDate today = LocalDate.now();
38
39         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

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     android:padding="16dp"
9     android:background="@color/azure"
10    android:contentDescription="今日日程页面">
11
12    <!-- 顶部操作栏 -->
13    <RelativeLayout
14        android:layout_width="match_parent"
15        android:layout_height="wrap_content"
16        android:layout_marginBottom="16dp">
17
18        <!-- 日期和标题 -->
19        <LinearLayout
20            android:layout_width="259dp"
21            android:layout_height="wrap_content"
22            android:layout_marginStart="10dp"
23            android:layout_marginBottom="10dp"
24            android:orientation="horizontal"
25            android:paddingTop="8dp"
26            android:paddingBottom="23dp">
27
28            <TextView
29                android:id="@+id/today_title"
30                android:layout_width="wrap_content"
31                android:layout_height="wrap_content"
32                android:layout_marginEnd="11dp"
33                android:text="Today"
34                android:textColor="#000000"
35                android:textSize="24sp"
36                android:textStyle="bold" />
37
38            <TextView
39                android:id="@+id/today_date"
40                android:layout_width="wrap_content"
41                android:layout_height="wrap_content"
42                android:contentDescription="当前日期"
43                android:textColor="#818181"
44                android:textSize="21sp">
```



```
45         android:textStyle="bold" />
46
47     </LinearLayout>
48
49     <!-- 右上角操作按钮 -->
50     <LinearLayout
51         android:layout_width="wrap_content"
52         android:layout_height="wrap_content"
53         android:layout_alignParentEnd="true"
54         android:orientation="horizontal">
55
56         <!-- 添加事件按钮 -->
57         <ImageView
58             android:id="@+id/add_event_btn"
59             android:layout_width="32dp"
60             android:layout_height="32dp"
61             android:src="@drawable/ic_add"
62             android:contentDescription="添加事件"
63             android:layout_marginEnd="16dp"
64             android:padding="4dp"/>
65
66         <!-- 设置按钮 -->
67         <ImageView
68             android:id="@+id/settings_btn"
69             android:layout_width="32dp"
70             android:layout_height="32dp"
71             android:src="@drawable/ic_settings"
72             android:contentDescription="设置"
73             android:padding="4dp"/>
74     </LinearLayout>
75 </RelativeLayout>
76
77 <!-- 第一部分：日程展示区（占1/3高度） -->
78 <LinearLayout
79     android:layout_width="match_parent"
80     android:layout_height="0dp"
81     android:layout_weight="1"
82     android:orientation="vertical"
83     android:layout_marginBottom="16dp"
84     android:elevation="4dp">
85
86     <TextView
87         android:id="@+id/date_text"
88         android:layout_width="wrap_content"
89         android:layout_height="wrap_content"
90         android:text="今日日程"
91         android:textColor="#000000"
92         android:textSize="16sp"
```

```
93         android:textStyle="bold" />
94
95     <androidx.swiperefreshlayout.widget.SwipeRefreshLayout
96         android:id="@+id/swipeRefreshLayout"
97         android:layout_width="match_parent"
98         android:layout_height="match_parent"
99         android:contentDescription="今日日程列表区域，支持下拉刷新">
100
101     <androidx.recyclerview.widget.RecyclerView
102         android:id="@+id/events_recycler_view"
103         android:layout_width="match_parent"
104         android:layout_height="match_parent"
105         android:padding="8dp"
106         android:scrollbars="vertical" />
107     </androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
108 </LinearLayout>
109
110 <!-- 第二部分：数据汇总区（占1/3高度） -->
111 <LinearLayout
112     android:layout_width="match_parent"
113     android:layout_height="0dp"
114     android:layout_weight="1"
115     android:orientation="vertical"
116     android:layout_marginBottom="16dp">
117
118     <LinearLayout
119         android:layout_width="match_parent"
120         android:layout_height="match_parent"
121         android:background="@drawable/cr10lr270cbf2f9b3fde4f7"
122         android:padding="16dp"
123         android:elevation="4dp">
124
125         <LinearLayout
126             android:layout_width="0dp"
127             android:layout_height="match_parent"
128             android:layout_weight="1"
129             android:orientation="vertical"
130             android:gravity="center_vertical">
131
132             <TextView
133                 android:layout_width="wrap_content"
134                 android:layout_height="wrap_content"
135                 android:layout_marginStart="0dp"
136                 android:layout_marginBottom="4dp"
137                 android:text="本周"
138                 android:textColor="#818181"
139                 android:textSize="16sp" />
140
```

```
141         <TextView
142             android:id="@+id/week_range"
143             android:layout_width="0dp"
144             android:layout_height="wrap_content"
145             android:layout_weight="1"
146             android:gravity="end"
147             android:text="（日期范围）"/>
148
149         <TextView
150             android:layout_width="match_parent"
151             android:layout_height="wrap_content"
152             android:text="数据汇总"
153             android:textSize="20sp"
154             android:textColor="@color/teal_700"
155             android:textStyle="bold"
156             android:layout_marginBottom="16dp"/>
157
158         <LinearLayout
159             android:layout_width="match_parent"
160             android:layout_height="wrap_content"
161             android:orientation="horizontal"
162             android:layout_marginBottom="8dp">
163
164             <ImageView
165                 android:layout_width="16dp"
166                 android:layout_height="16dp"
167                 android:src="@drawable/ic_circle"
168                 android:layout_marginEnd="8dp"
169                 app:tint="@color/teal_700" />
170
171             <TextView
172                 android:id="@+id/tasks_remaining"
173                 android:layout_width="wrap_content"
174                 android:layout_height="wrap_content"
175                 android:text="剩余任务：0"
176                 android:textSize="14sp"
177                 android:textColor="@color/teal_700"/>
178         </LinearLayout>
179
180         <LinearLayout
181             android:layout_width="match_parent"
182             android:layout_height="wrap_content"
183             android:orientation="horizontal">
184
185             <ImageView
186                 android:layout_width="16dp"
187                 android:layout_height="16dp"
188                 android:src="@drawable/ic_circle"
```

```
189         android:layout_marginEnd="8dp"
190         app:tint="@color/teal_700" />
191
192         <TextView
193             android:id="@+id/tasks_completed"
194             android:layout_width="wrap_content"
195             android:layout_height="wrap_content"
196             android:text="结束任务: 0"
197             android:textSize="14sp"
198             android:textColor="@color/teal_700"/>
199     </LinearLayout>
200
201     <TextView
202         android:id="@+id/more_details"
203         android:layout_width="wrap_content"
204         android:layout_height="wrap_content"
205         android:layout_marginTop="16dp"
206         android:text="Details_>>"
207         android:textColor="@color/teal_700"
208         android:textSize="14sp" />
209 </LinearLayout>
210
211 <LinearLayout
212     android:layout_width="0dp"
213     android:layout_height="match_parent"
214     android:layout_weight="1"
215     android:orientation="vertical"
216     android:gravity="center">
217
218     <com.example.calendarapp.CircularProgressView
219         android:id="@+id/progress_circle"
220         android:layout_width="100dp"
221         android:layout_height="100dp"
222         app:progress="25"
223         app:progressColor="@color/turquoise4"
224         app:bgColor="@color/azure"/>
225
226     <TextView
227         android:id="@+id/progress_text"
228         android:layout_width="wrap_content"
229         android:layout_height="wrap_content"
230         android:text="25%"
231         android:textSize="16sp"
232         android:textColor="@color/teal_700"
233         android:layout_marginTop="8dp"/>
234 </LinearLayout>
235 </LinearLayout>
236 </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>
```