1. JSONObject getChart( userId, weekday, type)
2. JSONObject getChart( userId, weekday, type) {
3. back //jsonObject,用于返回
4. array //jsonArray
5. **if** type ==  "SimpleAnalysis"
6. then:
7. labelList /\*ArrayList<Label>,所有标签list\*/
8. **for** Label label:labelList  /\*找到所有标签对应的\*/
9. percent //float,比例计算结果
10. truth  //int,标签所有事件实际消耗时间
11. schedule //int,标签所有事件计划使用时间
12. SAList /\*ArrayList<S\_Affair>,当前标签对应的日程list\*/
14. **for** S\_Affair sa:SAList
15. **if**  当前日程是userId对应的用户的日程
16. then
17. truth += getSpent(事件开始时间,事件结束时间)
18. schedule += getSpent(事件计划开始时间,事件计划结束时间)
19. end
21. /\*计算比例\*/
22. **if** 计划消耗时间 != 0
23. then
24. percent = |schedule - truth|/schedule
25. endif
27. /\*判断节省/超时\*/
28. **if** schedule >= truth
29. 将"timeSaving"存入js
30. **else**
31. 将"timeOut"存入js
33. 当前js加入数组array
34. end
35. 当前array加入最终返回的js对象
36. endif
38. elseif type == "detailedAnalysis"
39. then:
40. labelList/\*ArrayList<Label>,所有标签list\*/
42. **for** Label label:labelList
43. percent //JSONObject,比例数组，分别记录节约、相符、超时事件数量占比
44. save,equal,over //int,节约、相符、超时事件数
46. SAList /\*ArrayList<S\_Affair> ，当前标签对应日程list\*/
47. js //jsonObject,用于临时存储
49. **for** S\_Affair sa:SAList
50. **if** 当前日程是当前用户的日程
51. then:
52. truth = 0; //int,标签所有事件实际消耗时间
53. schedule = 0;//int,标签所有事件计划使用时间
54. truth = getSpent(事件开始时间,事件结束时间)
55. schedule = getSpent(事件计划开始时间,事件计划结束时间)
56. **if** truth > schedule    then: over ++
57. **else** **if** truth == schedule    then: equal ++
58. **else**  save ++
59. endif
60. end
62. all = save + equal + over //int,所有事件数
64. per1，per2，per3 //float,用于临时存储三个占比
66. **if** all != 0
67. then:
68. per1 = ((**float**)save/all)\*100;
69. per2 = ((**float**)equal/all)\*100;
70. per3 = 100 - per1 - per2;
71. endif
72. 三个占比分别加入percent数组
73. 将计算出的占比数组percent加入js
74. 当前js加入数组array
75. end
77. 当前array加入最终返回的js对象
78. end\_elseif
79. elseif type == "densityAnalysis"
80. then:
81. count //float[] ，（总）所有事件在各时间段占比
82. count1 //float[] ，（个）当前事件在各时间段占比
83. all //long,所有事件时间数
85. AList //ArrayList<Affair>，所有事件list
86. **for** Affair a:AList
87. **if** 是当前用户的事件
88. then
89. count1 = getStep(开始时间,结束时间) //获得当前事件在各时间段的占比
91. **for** **int** j = 0;j < 7;j ++
92. count[j] += count1[j]
93. end
94. all ++;
95. endif
96. end

99. per //float[] ，每一阶段的有效利用时间和阶段总时间之比
100. 计算per[i]，按序存储
101. per[i] = count[i] / time\_i /\*time\_i指第i时间段总时间\*/
103. density = all / 用户有效使用时间 //float，平均每小时事件完成数
104. //density保留两位小数
105. **for** **int** i = 0;i<7;i++
106. per[i] = (**float**)(Math.round(per[i]\*100))/100
107. end
109. 将density加入最终返回的js对象
110. 将不同时间段的阶段名称以及相对应的per数组的比例数值存入新建的jsonObject
111. 将jsonObject存入jsonArray
112. 当前array加入最终返回的js对象
113. end\_elseif
114. **else**
115. 返回空对象
117. **return** back /\*返回处理后数据的jsonObject形式\*/
118. }
119. float[] getStep(String from,String to)
120. //通过开始时间、结束时间获得事件所在时间段
121. **float**[] getStep(String from,String to) {
122. SimpleDateFormat simple = **new** SimpleDateFormat("HH-mm") //用于格式化时间为小时-分钟形式
124. l[i] = (**int**)((simple.parse("00-00").getTime())/(1000\*60)) //int[],第i阶段开始时间（分钟），共七个时间段
126. **long** f,t //long,得到事件开始时间和结束时间毫秒数并转化成分钟
127. f = simple.parse(from).getTime()
128. t = simple.parse(to).getTime()
129. f = (**int**)(f/(1000\*60))
130. t = (**int**)(t/(1000\*60))
132. all = (**int**)((t - f)/(1000\*60)) //int,当前事件总时间(分钟)
134. per//float[],per数组用来装每个阶段的当前事件占比，放回原函数在每个阶段逐一加上即可
135. **for** **int** i = 0;i < 7;i ++
136. **if**(f >= l[i]&& f <= l[i+1]) { //先找到开始时间所在阶段
137. **if** t >= l[i+1] //结束时间晚于当前阶段的结束时间（即下一阶段的开始时间），此时跨越两个时间段
138. then:
139. per[i] = ((**float**)l[i+1] -(**float**) f) / all //在当前第i阶段所用时间在此事件总时间的占比
140. f = l[i+1]; //下次循环从下一阶段开始时间作为开始时间
141. endif
142. **else**
143. per[i] = ((**float**)t-(**float**)f)/all //不跨阶段，直接计算当前第i阶段所用时间在此事件总时间的占比
144. endif
145. end
147. **return** per //返回比例数组
148. }

if type == "SimpleAnalysis"

            then:

             labelList /\*ArrayList<Label>,所有标签list\*/

             for Label label:labelList /\*找到所有标签对应的\*/

                 percent //float,比例计算结果

                 truth //int,标签所有事件实际消耗时间

                 schedule //int,标签所有事件计划使用时间

                 SAList /\*ArrayList<S\_Affair>,当前标签对应的日程list\*/

                 for S\_Affair sa:SAList

                     if 当前日程是userId对应的用户的日程

                         then

                             truth += getSpent(事件开始时间,事件结束时间)

                         schedule += getSpent(事件计划开始时间,事件计划结束时间)

                 end

                    /\*计算比例\*/

                 if 计划消耗时间 != 0

                     then

                     percent = |schedule - truth|/schedule

                    endif

                    /\*判断节省/超时\*/

                 if schedule >= truth

                     将"timeSaving"存入js

                 else

                     将"timeOut"存入js

                 当前js加入数组array

             end

             当前array加入最终返回的js对象

     endif