# Fine-Report

1. 消息推送复选框修改为下拉树

getAllDepartment()

获取所有部门(departmentsJO/departments)

getUserInDepartment

根据部门ID获取部门用户

1. 南京帆软软件地址

南京市玄武区珠江路东鼎大厦3号楼902-1室

1. plugin-H5Report项目地址

git地址:

/f/finereport/plugins/plugins-html5/plugins-html5

webpack打包地址:

F:\finereport\dev\plugins\plugins-html5\plugins-html5\plugin-H5Report\src\com\fr\wei\plugin\h5reportnew

F:\finereport\master\plugins\plugins-html5\plugins-html5\plugin-H5Report\src\com\fr\wei\plugin\h5reportnew

1. less打包

node less.config.js

1. IntelliJ IDEA快捷键

IntelliJ Idea 常用快捷键列表

Ctrl+Shift+N 查找文件

Ctrl+alt+L 代码格式化

1. 资料地址

(1)移动端周报:

http://www.finedevelop.com/pages/viewpage.action?pageId=8327768

(2)有道云协作:

http://163.fm/CNZWGoX

bug查询:

http://114.215.175.35:8080/WebReport/ReportServer?op=fs

(3)工作日志:

http://www.finedevelop.com:2016/secure/Dashboard.jspa

(4)Git操作指南:

http://www.finedevelop.com/pages/viewpage.action?pageId=2690911

(5)入门指南:

http://www.finedevelop.com/pages/viewpage.action?pageId=1934842

(6)basic/plugins-html5版本库:

http://www.finedevelop.com:2015/projects/BA/repos/plugins-html5/browse

(7)dew/plugins-html5版本库:

http://www.finedevelop.com:2015/users/drew/repos/plugins-html5/browse

1. Git操作指令

查看代码变化

1. git diff
2. git status

命令窗口代码上传步骤

1. git pull origin dev
2. Webpack
3. node less.config.js
4. git add .
5. git commit -am '注释'
6. git push origin dev
7. 查看Create pull request
8. 查看diff

注意: pull和push之前要打包！！

SourceTree代码上传步骤:

1. commit
2. pull
3. 有冲突解决冲突，解决冲突后在重复(1)和(2)步骤;
4. Push

本地代码区与远程分支代码对比

# 获取远端库最新信息

$ git fetch origin

# 做diff

$ git diff origin/dev dev

1. 账号密码

极客学院: 13024751935 密码: 66666666

妙味课堂: nmw1990 密码:299792458

极客学院：15619015034 密码：codercoder

1. 控件事件与JS方法

MOBILE-237：h5：实现控件的事件及控件的js方法

isEnable isVisible

Widget.setEnable(true/false)

设置控件是否可用，参数为true可用，false不可用

Widget.setVisible(true/false)

设置控件是否可见，参数为true可见，false不可见

Widget.getValue() 获取控件值

Widget.setValue() 给参数控件赋值

Widget.setText() 设置控件的显示值

Widget.reset() 清空数据

Widget.fireEvent()

一个问题: 点击下拉列表之后才有数据显示

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 控件类型、对应的文件以及支持的事件 | | | | |
| 控件名 | 控件类型 | 对应文件 | 支持事件 | 备注 |
| 文本框 | text | texteditor.js | 初始化后、编辑后、编辑结束 |  |
| 文本域 | textarea | textarea.js | 初始化后、编辑后、编辑结束 |  |
| 标签 | label |  |  |  |
| 下拉框 | combo | comboBox.js | 初始化后、编辑后、编辑结束、点击 |  |
| 下拉复选框 | tagcombocheckbox | comboCheckBox.js | 初始化后、编辑后、编辑结束、点击 |  |
| 下拉树 | treecombobox | treeComboBox.js | 初始化后、编辑后、编辑结束、点击 |  |
| 日期 | datetime | dateEditor.js | 初始化后、编辑后、编辑结束 |  |
| 单选按钮组 | radiogroup | radiogroup.js | 初始化后、状态改变 |  |
| 复选框 | checkbox |  |  | 功能未支持 |
| 复选框组 | checkboxgroup | checkgroup.js | 初始化后、状态改变 |  |
| 密码控件 | password |  | 初始化后、点击 | 功能未支持 |
| 视图树控件 | tree | treeComboBox.js | 初始化后、点击 |  |
| 文件控件 |  |  |  | 无 |
| 数字控件 | number | numbereditor.js | 初始化后、编辑后、编辑结束 |  |
| 按钮控件 | button | button.js | 初始化后、点击 |  |

代码修改细节:

(1)文本框控件

type: text

事件:初始化后、编辑后、编辑结束

texteditor.js文件中

给文本控件增加鼠标失去焦点事件,并绑定handleBlur函数

在textbasa.js文件中增加handleBlur函数

scale.js中修改 listener参数

(2)文本域控件√

type: textarea

textarea.js文件中删除以下代码

初始化后、编辑后、编辑结束

componentDidMount: function () {

this.afterWidgetInit();

}

(3)标签控件

label

未实现

(4)下拉框控件√

是否可用，别忘了取消点击事件

type: combo

初始化后、编辑后、编辑结束、点击

comboBox.js中添加

this.fireEvent("click", e);

(5)下拉复选框按钮√

type:tagcombocheckbox

初始化后、编辑后、编辑结束、点击

comboCheckBox.js中添加

this.fireEvent("click", e);

this.fireEvent("stopedit", e);

(6)下拉树控件√

treecombobox

初始化后、编辑后、编辑结束、点击

treeComboBox.js中添加

this.fireEvent("click");

this.fireEvent("stopedit", e);

下拉树控件注意第一层信息别忘了。

(7)日期控件√

datetime

dateEditor.js文件中添加

初始化后、编辑后、编辑结束

if(shouldSelect) {this.fireEvent("afteredit", date);}

(8)单选按钮组控件√

radiogroup

初始化后、状态改变

radiogroup.js文件中添加

this.fireEvent("statechange", e);

(9)复选框控件

功能未支持

(10)复选框组控件√

checkboxgroup

初始化后、状态改变

checkgroup.js

添加this.fireEvent("statechange", e);

(11)密码控件

初始化后、点击

功能未支持

(12)视图树控件√

tree

初始化后、点击

下拉树修改之后已实现此功能

(13)文件控件

无需求

(14)数字控件√

number

初始化后、编辑后、编辑结束

numbereditor.js添加

onBlur={this.handleBlur}

删除componentDidMount: function () {

this.afterWidgetInit();

},

(15)按钮控件√

button

初始化后、点击

1. 代码阅读技巧

(1) 从执行顺序入手;

1. 适当忽略细节;
2. 重视烂笔头;
3. 反复阅读

代码的阅读曲线并不是平缓的，在很长一段时间里，你对代码的理解可能都处于一个非常小的范围内。

1. 文件结构

(1)components文件夹

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Form | autoLayout |  | | |
| elementcase |  | | |
| elementcaseflip |  | | |
| form |  | | |
| formLayout |  | | |
| layoutElement |  | | |
| originalLayout |  | | |
| title |  | | |
| titleWidget |  | | |
| fs | directoryTree |  | | |
| entryItem |  | | |
| navigationBar |  | | |
| navigitionBarItem |  | | |
| subDirectory |  | | |
| parameter | parameterComponents | 参数面板(包含各种控件) | | |
| report | forzenTable |  | | |
| reportComponents |  | | |
| reportContentTable |  | | |
| something | popList |  | | |
| toolbar | toolbar |  | | |
| toolbarBotton |  | | |
| toolbarGroup |  | | |
| widget | group | checkgroup | 复选框组控件 | |
| groupEditor |  | |
| groupItem |  | |
| radiogroup | 单选按钮组控件 | |
| input | numbereditor | 数字控件 | |
| textarea | 文本域控件 | |
| textbase | input控件的Mixin混入方法 | |
| texteditor | 文本控件 | |
| triggle | tree | treeComboBox | 下拉树控件 |
| treeNode |  |
| treeNodeModel |  |
| conboBox | 下拉框控件 | |
| comboCheckBox | 下拉复选框控件 | |
| comboItem |  | |
| comboList |  | |
| triggerEditor | trigger控件的Mixin混入方法 | |
| button | 按钮控件 | | |
| cancelButton |  | | |
| chart | 图表组件:新图表自动刷新。 | | |
| dateEditor | 日期组件:允许日期控件默认值为空。 | | |
| label |  | | |
| labelWidget |  | | |
| loading |  | | |
| scale | 与表单主体控件的options有关 | | |
| submitbutton |  | | |
| widget | 所有控件的可Minxin方法 | | |
| widgetFactory | 控件工厂:所有控件的入口 | | |

1. 绝对布局——保留布局
2. 获取的json格式

{"widgetName": "FORM",

"type": "border",

"items": [{

"region": "center",

"el":{

"widgetName": "BODY",

"type":"fit",

"itemsIndex": [

"COMBOBOX0",

"TEXTEDITOR0",

"BUTTON0",

"BUTTON1"

],

"items": [

{"x": 0,"y": 0,"width": 236,"height": 21},

{"x": 236,"y": 0,"width": 238,"height": 536},

{"x": 474,"y": 0,"width": 238,"height": 536},

{"x": 712,"y": 0,"width": 239,"height": 536}

]

},

}]

}

1. form.js表单入口

函数1:getBorderLayoutCenterElements()

函数1-1: initTrueDataAttribute()——重新计算容器尺寸。

①计算fit尺寸与屏幕尺寸;

②屏幕尺寸减去8; ③compState为0则充满,1则等缩。

函数1-1-1: isAutoLayout()——是否重布局(重布局且手机)

④若保留布局计算缩放比例(双向自适应与横向自适应)

函数1-1-1-1:isFill()

⑤若等缩，则等缩。若充满，则充满。

函数1-2:calculateTrueData(item)——重新计算控件尺寸

①重新计算控件的宽高及横纵坐标:控件宽为

函授1-2-1:getTrueWidthOrHeight()——获取控件的宽高

①宽高独占一行控件不减，两边控件减2，中间控件减4。

②位置在0的控件不动，大于0的控件右移2。

**问题:控件不可能占据一行(差4像素)？？**

函数2: this.initStyle(content)——fit样式body的背景色

步骤3: if.. else... 判断，进入重布局/保留布局两种情况

①传参为body的背景色和content的总信息。

1. originalLayout.js保留布局入口

函数1:var itemGroups = getItemGroups();——获取组信息

函数1-1:遍历items执行addItem2Groups(items, itemGroups);

——每一个item生成独立的组信息

函数1-1-1:checkExists(item, itemGroups);

——判断item是否已经在itemGroups数组中

函数1-1-1-1:遍历itemGroups执行isGroupHasItem(group, item)

若为false(不存在),执行createGroup(item);创建组

函数1-1-2:createGroup(item);

①X等于0, 宽等于fitLayoutWidth

②y等于item的y, height等于item的height

函数1-1-2-1:addItem2Group(item, group);

①调整item在group中的坐标。

函数1-1-2-2:refreshGroup(group);

①遍历this.props.options.items。

函数1-1-2-2-1:isGroupHasItem();

函数1-1-2-2-2:isItemShouldAdd2Group();

函数2:createLayoutContents(itemGroups)——组信息生成组元素

总结:①控件间距为4像素。②只有保留布局下，才计算缩放比例。

思路:①formutils中定义函数，判断是自适应布局还是绝度布局。

外层body重布局:

1. 重布局→无绝对布局块

h5-form

h5-form-auto-layout

ul

li ——margin:4px;

h5-layout-container

Widget

1. 重布局→一个绝对布局块

(3)重布局→一个绝对body(重布局)

h5-form

h5-form-auto-layout

ul

li ——.h5-absolute-layout margin:0;

h5-layout-container ——高:windowHeight

Widget ——.h5-absolute-block height:100%

h5-form-auto-layout

ul

li ——margin:4px;

h5-layout-container

Widget

1. 重布局→一个绝对body(保留布局)
2. 重布局→一个绝对布局块和其他控件
3. Tab组件点击滑动逻辑

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 一个屏幕可以显示tab块的个数  (numberOfTabInOneScreen)(ST) | | | | |
| 总的tab块的个数  numberOfTab  (AT) | 2n (n>=1) | | 2n + 1 (n>=0) | |
| 2m  (m > n) | 点前n + 1个块  **Index + 1 <=**  **Math.floor(ST/2) + 1**  isLeftSide = true;  isRightSide = false;  left = 0; | 2m  (m > n) | 点前 n + 1 块  **Index + 1 <=**  **Math.floor(ST/2) + 1**  isLeftSide = true;  left = 0; |
| 点后n + 1个块  **AT - Index <=**  **Math.floor(ST/2) + 1**  isLeftSide = false;  isRight = true;  -left = getMaxLeftOfWrap; | 点后n + 1块  **AT - Index <=**  **Math.floor(ST/2) + 1**  isRight = true;  -left = getMaxLeftOfWrap; |
| 点其他的块(如果有)  isLeftSide = false;  isRightSide = false;  前m块  **Index + 1 <= Math.floor(AT/2)**  (-left = index - n + 1)  -left = index - Math.floor(ST/2) + 1  后m块  **AT - Index <= Math.floor(AT/2)**  (-left = index - n)  -left = index - Math.floor(ST/2) | **点其他块(如果有)**  **isLeftSide = false;**  **isRightSide = false;**  **-left = index - Math.floor(ST/2) (-left = index - n;)** |
| 2m + 1  (m>=n) | 点前n + 1个块  **Index + 1 <=**  **Math.floor(ST/2) + 1**  isLeftSide = true;  isRightSide = false;  left = 0; | 2m + 1  (m > n) | 点前n + 1块  **Index + 1 <=**  **Math.floor(ST/2) + 1**  isLeftSide = true;  isRightSide = false;  left = 0; |
| 点后n + 1个块  **AT - Index <=**  **Math.floor(ST/2) + 1**  (可能有的块既在前n+1, 又在后n+1)  isLeftSide = false;  isRight = true;  -left = getMaxLeftOfWrap; | 点后n+1块  **AT - Index <=**  **Math.floor(ST/2) + 1**  isLeftSide = false;  isRight = true;  -left = getMaxLeftOfWrap; |
| 点其他块(如果有)  isLeftSide = false;  isRightSide = false;  前m块  **Index + 1 <= Math.floor(AT/2)**  (-left = index - n +1;)  -left = index -  Math.floor(ST/2) + 1  第m+1块  **Index + 1 = Math.floor(AT/2) + 1**  (-left = index - n + 1;)  (或者index - n)  后m块  **AT - Index <= Math.floor(AT/2)**  (-left = index - n;)  -left = index - Math.floor(ST/2) | **点其他块(肯定有)**  **isLeftSide = false;**  **isRightSide = false;**  **-left = index - Math.floor(ST/2)**  **(-left = index - n;)** |

1. 复选框逻辑

pc端的逻辑

假如一共有五项，已选项有四项，搜索之后显示两项。

1此时如果勾选或者取消一个显示项，就在已选项中添加一项或者删除一项。

1-1 即使所有显示项都已勾选，也不自动勾选全选/不选项。

1-2 如果有显示项未勾选，自动取消勾选全选/不选项。

2如果手动勾选或者取消全选/不选项

2-1 手动勾选全选/不选项，则将这两项添加到已选项中。

2-2 手动取消全选/不选项，则将所有已选项取消选中。

1. 控件与控件外的包裹

(1)title与titleWidget——报表块与图标块

(2)scale——下拉框、复选框、文本等双层数据结构

(3)labalWidget与label——标签和标签的控件

1. 底部工具栏接口
2. 报表与表单底部工具栏按钮分类

Cpt pageButton: 上一页、页码、下一页

ZoomButton: 放大、缩小

OtherButton:刷新、返回参数面板

Frm OtherButton:刷新、返回参数面板

(2)H5与原生APP交互流程

①h5向app发送信息，告诉app当前模板都有哪些底部工具栏按钮

②app向h5发送信息，不显示h5工具栏。

③app调用h5工具栏接口，实现翻页、返回参数面板、放大缩小，刷新的功能。

例如:在调用翻页的过程:

①H5调用app中定义方法，告诉app当前工具栏含有翻页功能及总页码

②创建app翻页按钮

③点击app翻页按钮

④app调用h5中定义的翻页方法

⑤执行h5中定义的翻页方法，实现翻页

⑥翻页成功之后h5调用app中中定义的方法，告诉app当前页码

1. 异步操作

H5调用app中定义的方法，该方法在app中执行，并不会阻塞h5中其他代码的执行；app调用h5中定义的方法，该方法在h5中执行，并不会阻塞app中其他代码的执行。即:函数定义在哪里就在哪里执行，app执行环境与h5执行环境互不阻塞。

(4)底部工具栏h5接口

Cpt 上一页: window.contentPane.gotoPreviousPage();

打开分页框: window.contentPane.showPaginationModal();

下一页: window.contentPane.gotoNextPage();

返回参数面板: window.contentPane.showFilter();

刷新: window.contentPane.refreshReport();

放大: window.contentPane.zoomIn();

缩小: window.contentPane.zoomOut();

Frm 返回参数面板: window.contentPane.showFilter();

刷新: window.contentPane.refreshReport();

(5)底部工具栏jsBridge接口

//告诉app都有哪些按钮

jsBridge.callHandler("showToolbarItems",toolbarDetails,

function (success) {

//success true/false app是否完成创建原生工具栏

});

//弹出翻页框

jsBridge.registerHandler("showPaginationModal", function () {

//弹出翻页界面

})

//上一页

jsBridge.registerHandler("go2LastPage", function () {

//翻页 上一页

jsBridge.callHandler("updatePageNumber", pageIndex);

//page翻页之后页码

})

jsBridge.registerHandler("go2NextPage", function () {

//翻页 下一页

jsBridge.callHandler("updatePageNumber", pageIndex);

//page翻页之后页码

})

jsBridge.registerHandler("refresh", function () {

//刷新

})

jsBridge.registerHandler("showFilter", function () {

//返回参数面板

})

jsBridge.registerHandler("zoomIn", function () {

//放大

})

jsBridge.registerHandler("zoomOut", function () {

//缩小

})

1. 待解决的问题

(1)下拉树setValue的bug

(2)参数面板滑动高度为宽高比更大的那个，而不总是高。

1. 新图表插件更新
2. 下载新图片插件解压
3. 对应文件放在对应文件夹

两个xml文件放在

D:\FineReport\_8.0\WebReport\WEB-INF\plugins\vanchart

两个jar包放在

F:\finereport\dev\lib

更新资源

D:\FineReport\_8.0\WebReport\WEB-INF\resources\geojson

1. 兼容性问题有以下几种情况
2. Android和IOS不同系统导致表现不同
3. Android/IOS系统中不同浏览器导致表现不同
4. 手机屏幕尺寸不同导致表现不同
5. 点击查询按钮

parameterComponents:

onSubmitClick函数

onSubmit函数

formPane:

renderParameter函数中的afterSubmit键值对

setStateAfterSubmit函数

getFormData函数

1. 学一个新技术的正确的路径

基本概念 与 Github项目

1. GitHub账号密码

账号: [254060001@qq.com](mailto:254060001@qq.com)

密码: a299792458

1. 点取消按钮告诉app

连续点击翻页500ms处理。

1. 项目中遇到的问题

(1)通过jQuery插入在body中的元素如何统一？

答案: 是不是一定要插入在body的子元素这一层级，body的子元素预留一个div，宽高100%。在这个div中控制pop弹窗、loading动画、报表缩放弹窗(hint)等等。

1. table横向/纵向/双向自适应 如何处理边框？

答案: 根据后台传表框情况分别处理

1. Window.resize()切换的时候的forceUpdate()统一处理

答案: forceUpdate只写一个，在reportPane和formPane层级。

1. frm中的表格在第15层，有必要存在那么多层级吗？
2. 咱们的项目是所有浏览器都必须兼容吗？
3. reportPane每次update时，report和parameter都会重新执行willMount、render、didMount。

答案:renderParameter和renderReport中的key待修改。

1. formPane每次update时，form和parameter都会重新执行willMount、render、didMount。

答案:renderParameter和renderForm中的key待修改。

引起的问题: ①横竖屏切换时的闪屏，相当于刷新了。

②初始化后事件在横竖屏切换时触发。

1. H5和PC端共用后台，使用ajax获取数据当出错时获得的数据有exception值。IOS和Android使用的后台与H5不同？
2. 现在报表缩放的逻辑是rem=16px乘以0.8-0.2的系数，从16\*0.4 = 6.4px到16\*0.2 = 3.2px的过程缩放效果失败。
3. h5底部工具栏在没有window.jsBridge的时候不需要500ms延迟。
4. 补充jsBridge文档。
5. 树控件Ajax请求数据的limitIndex为5000

//jsonData为加载的数据

var jsonData = BaseUtils.jsonDecode(responseObject);

//加载数据量等于5000时，needLoadMoreData为true，否则为false

self.needLoadMoreData = jsonData.length >= self.limitIndex;

1. 下拉树控件存在的问题
2. 初始值控件值与搜索值后台返回结果不一致。
3. 设置初始值与设置搜索值时后台返回数据的规律。
4. 初始值应不应该放在搜索值中。
5. 树控件是否展开

var isExpand =

(1)treeNodeData.hasChildren&&this.dataFilter!=="" ? true : false;

搜索内容为空时总是收起。搜索内容不为空时，有子时才展开。

(2)treeNodeData.isexpand ? true : false;

后台传的expand决定是否展开。

1. 绝对布局-保留布局时，对控件有一个剪切和缩放的步骤。Tab组件中的tabContentItem面板如果也是一个绝对布局-保留布局容器，也需要有这个剪切和缩放的步骤。
2. 后台代码部分替换
3. 查看修改的.java文件对应jar包所在位置

例如:fr-report-8.0.jar\com\fr\form\ui\container\cardlayout

1. 将.java文件放在对应路径

例如: \src\com\fr\form\ui\container\cardlayout

1. 重新编译rebuild
2. 生成.class文件路径

例如:F:\finereport\dev\env\WebReport\..\container\cardlayout

1. 将生成的.class文件替换jar包中的class文件
2. 问题

(1)try catch在项目上的应用

(2)函数式编程。