客户端UI界面开发流程

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **更新时间** | **更新内容** | **更新人** |
| **V1.0** | 2016-03-05 | 建立文档 | 张青 |
| **V1.1** | 2016-03-05 | 修改了部分用词，添加了代码限制 | 张青 |

# 概述

以下内容为开发流程及流程及流程中涉及到的规范。流程与规范必须遵守。

# 开发环境准备

Track（预装）

P4v [\\172.17.100.2\临时存取\x5\_mobile\软件安装\p4vinst64.exe](\\\\172.17.100.2\\临时存取\\x5_mobile\\软件安装\\p4vinst64.exe)

Unity5.2.3 [\\172.17.100.2\临时存取\x5\_mobile\软件安装\unity5.2.3](\\\\172.17.100.2\\临时存取\\x5_mobile\\软件安装\\unity5.2.3)

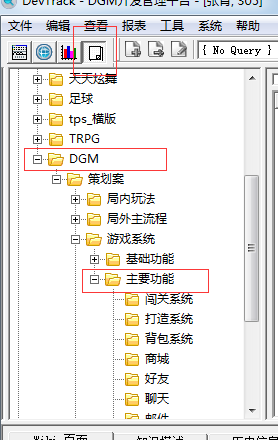
VS2013 [\\172.17.100.2\临时存取\x5\_mobile\软件安装\VS2013](\\\\172.17.100.2\\临时存取\\x5_mobile\\软件安装\\VS2013)

# 需求宣讲

功能开发以月版本为单位进行推进，每个版本初期（上个版本末期）会由策划在功能开发群中发出需求文档，并在稍后的时间主持该版本功能的需求宣讲会议，向客户端、服务器程序、测试人员、美术人员描述功能需求。在会议之前客户端程序需要先阅读需求文档，对需求有一个大致的了解，并在会议上对于描述不明确，无法理解的需求提出疑问进行讨论。

# 需求分析/反宣讲/安排计划

需求宣讲完成后，需求文档推进到正式版本，并上传到track知识库以下位置。



客户端程序以track上的正式版本文档为基准，进行需求分析文档的撰写。需求分析文档包含如下内容：

* 写清楚需求文档中所有与客户端相关的逻辑规则。不能复制需求文档中的内容，应以自己理解的基础进行撰写。
* 描述清楚需求文档中所有涉及到的界面元素的初始状态，该元素可以进行的交互，因需求而产生的状态变化（原本显示A，xxx条件下显示B），及各种界面元素的边界条件。（列表如果为空时是否需要有文字提示，文字内容不是固定内容时是否有字符数限制，是否会超长，超长后应该如何处理等）
* 除了基本UI界面之前哪些地方需要进行特效处理。
* 标记出需求文档中逻辑有问题或者描述不清楚的地方。

需求分析文档撰写完成后由客户端程序主持客户端需求分析反宣讲会议，逐条确认需求分析文档中的内容。并回答其他与会人员的疑问。

反宣结束后修改需求分析文档中不正确或者不明确的部分。上传到p4v 路径//x5\_mobile/mobile\_dancer\_resource/文档管理/程序文档/需求分析/0.0.0.6（开发版本号）/UI需分/ 下。之后进入代码开发阶段。

# UI美术资源宣讲

程序反宣之后与组长确认功能开发各功能完成时间点。进行开发，UI美术资源与程序开发同时进行，美术资源制作完成后会由美术人员组织UI资源宣讲。客户端程序需要与会并确认美术资源与开发需求一致无遗漏或错误。

# UI界面功能开发

## UI美术资源接入

美术资源宣讲完成之后一两日内美术给出提供客户端程序制作用的资源，包括如下内容：

* 界面效果图

P4v 路径 //x5\_mobile/mobile\_dancer\_resource/Resources/美术资源/UI/效果图/XXX/

* 界面程序文字说明

P4v 路径 //x5\_mobile/mobile\_dancer\_resource/Resources/美术资源/UI/效果图/XXX/

* 界面UI切图

P4v 路径 //x5\_mobile/mobile\_dancer\_resource/Resources/美术资源/UI/切图/UI切图目录程序用/AtlasXXX/

* 界面UI切图制作数据

//x5\_mobile/mobile\_dancer\_resource/Resources/美术资源/UI/数据与说明/XXX

UI美术资源没有完成之前，客户端程序参考策划需求文档中的概念图，使用已有的图集中的资源大体摆出各个界面的功能元素。优先进行逻辑的开发。

UI美术资源完成后，客户端程序需要使用美术资源完成如下工作：

1. 使用美术提供的资源制作UIAtlas图集，图集制作方法见p4v 路径 //x5\_mobile/mobile\_dancer\_resource/文档管理/程序文档/解决方案/客户端方案/atlas编辑器使用.docx
2. 图集制作完成后使用正式资源图集中的元素替换临时放置的元素，并以效果图为基准重排界面布局。规范：正式界面中的各个元素在屏幕中的位置与对齐方式要求与效果图完全一致。界面校对方式见 p4v路径（补充文档与p4v路径）
3. 界面中涉及到的动画与特效，接入完成时需要与策划、美术确认效果，如果策划美术不满意，请美术人员协助调整动画与特效效果。以美术最后调整的效果为准。
4. 多分辨率适应。见多分辨率方案，补充说明

## UI界面框架

//x5\_mobile/mobile\_dancer\_resource/文档管理/程序文档/解决方案/客户端方案/DGM客户端框架设计方案.docx

UIController接口说明

## 逻辑接口接入

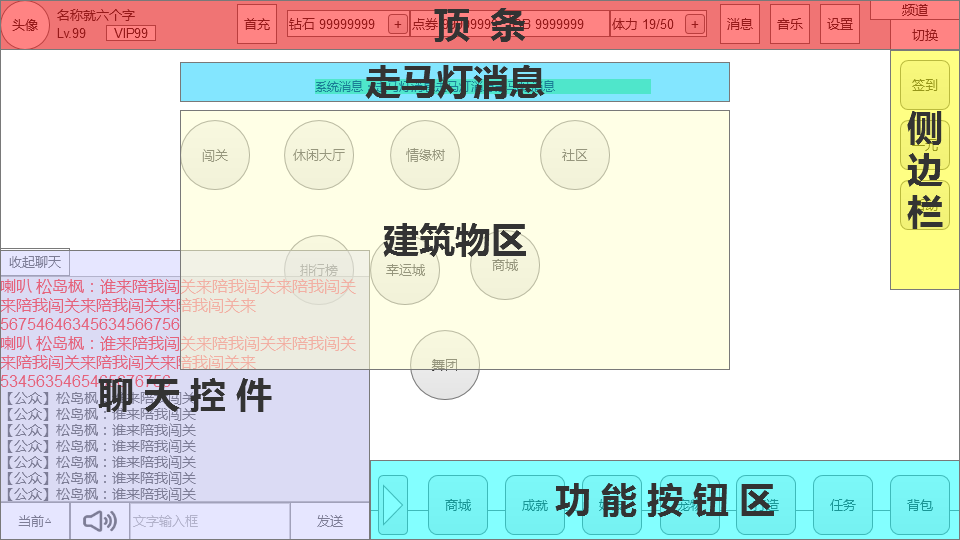
## QA开发

## 联调与自测

# 提测与bug修改

# 定义

1. 状态界面：状态界面之间是互斥的，每一个状态界面表示一个完整的界面场景，可包含一个或多个子界面。一个状态界面定义了一个游戏的大功能区，并对应一个界面状态。每一个界面状态之间是互斥的，不可以同时存在多个界面状态。目前游戏定义了以下几个状态界面。
   1. 登录界面（执行登录和创建角色功能）
   2. 世界地图界面（各个区块的总入口，基本信息的展示）
   3. 游戏大厅界面（展示可以进行游戏的房间列表）
   4. 游戏房间界面（展示游戏房间内的信息和交互）
2. 子界面：子界面表示一个生命周期相同，且功能相近的界面区块。一个或多个子界面组成一个状态界面。例：



上面是世界地图界面，可以分成顶条、走马灯消息、聊天控件、功能按钮区、建筑物区、侧边栏六个逻辑部分，其中：

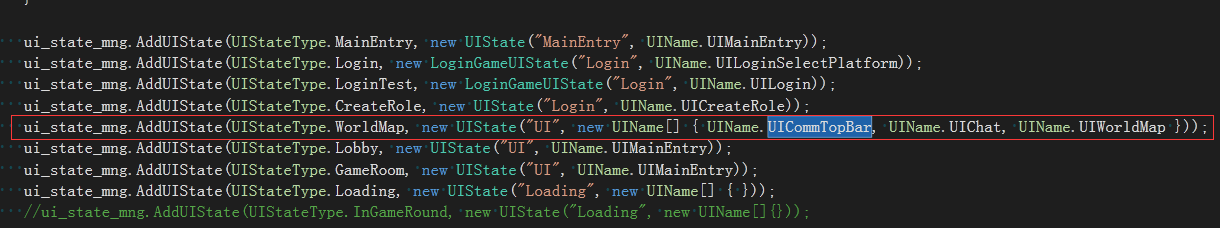
* 聊天控件、顶条、走马灯信息在其他界面也需要用到，属于通用控件，单独定义为一个子界面。
* 建筑物区、侧边栏、功能按钮区的生命周期一致，都是在世界地图界面中一直存在的界面。这三个定义为一个子界面。
* 如果界面中可以点出其他功能性弹框，每个功能性弹框定义为一个子界面。
* 因此世界地图由以下子界面组成：
  + 顶条
  + 聊天控件
  + 飞屏
  + 建筑物区+功能按钮区+侧边栏
  + 其他各种功能性弹框界面
* **一个界面是状态界面还是子界面是由需求决定的。比如：背包界面虽然是一个全屏的界面，但是背包界面可以在世界地图界面，房间内界面等多个状态界面中打开。因此背包界面必须被归类为一个子界面，而不能制作成一个状态界面。**

# 部署客户端与服务器环境

* 从p4上更新//x5\_mobile/mobile\_dancer/trunk/exe/
* 从p4上更新//x5\_mobile/mobile\_dancer/trunk/server/
* 打开server目录下products/project\_dgm/dgm\_sln/DGM\_Server.sln工程并进行编辑。
* 编译完成后打开exe目录，执行start\_allserver\_d.bat，启动服务器，如果出现某个服务器启动后崩溃，找服务器人员确认原因。
* 从p4上更新//x5\_mobile/mobile\_dancer/trunk/client
* 用unity打开客户端目录，找到Consts定义文件，修改其中的server\_ip为本机ip，修改Server\_port为32016
* 打开Unity的MainEntry场景。运行游戏。

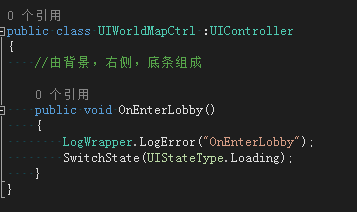
# 客户端UI开发流程

1. 根据需求，拆分状态界面及子界面（见定义部分），拆分完成后找组长对一下拆分是否合理。
2. 在StateRegister文件中添加新状态界面对应的UIStateType枚举。
3. 在UIRegister中添加新子界面对应的UIName枚举
4. 在UIStateRegister.RegisterUIState函数中注册状态界面及其默认显示的子界面。

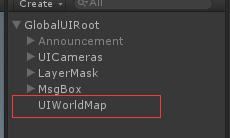


* UIStateType定义一个状态界面
* “UI”表示该状态界面所在的.unity文件。没有特殊需求，局外界面都使用UI.unity
* UIName[] 表示该状态界面中包含哪些进入界面时需要显示的子界面。

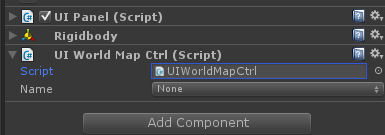
1. 为各个子界面新建一个继承UIController的类来管理该子界面的界面逻辑。并为每一个UI交互操作定义一个处理函数，如下图中的OnEnterLobby()

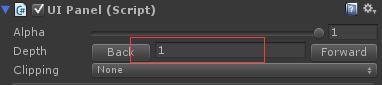


1. 为新子界面创建Prefab
   1. 为新子界面创建Atlas(见Atlas创建文档)
   2. 打开Scenes/UIEdit场景，在GlobalUIRoot下创建一个空GameObject。作为子界面的根节点。

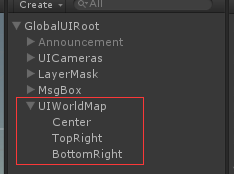


* 1. 在根节点上挂接UIPanel和新编写的Controller。这里是UIWorldMapCtrl。并将UIPanel的层级指定为1

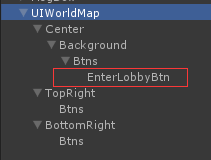


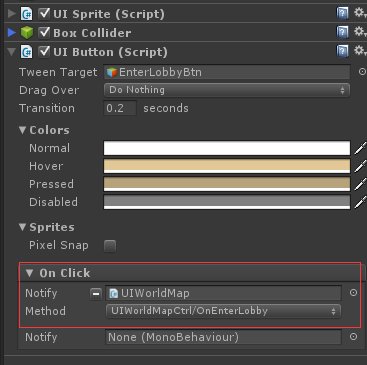


* 1. 根据界面元素在界面上的停靠，在根节点下面创建一个或多个布局节点。节点上挂上UIAnchor控件并指定对齐方式

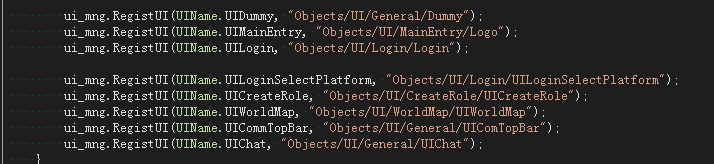


* 1. 在布局节点下部署UI元素，将可交互的UI元素（一个按钮，一个复选框等。如下图中“EnterLobbyBtn”是一个按钮）上交互事件的处理函数指定为根节点上的Controller中的函数。（下图中将EnterLobbyBtn的OnClick消息事件的处理函数指定为UIWorlMap中的函数）





* 1. 运行该场景，自测界面交互元素是否正确传递给函数。
  2. 自测成功之后将该子界面的根节点拖到project视图下Resources/art/ui下指定模块中保存。

1. 在UIRegister中注册UIName与Prefab路径的关系，路径使用resources下的相对目录
2. 在其他界面的Controller中通过SwitchState(UIStateType)/PushAndSwitchState 来切换到指定的界面。
3. 状态界面中通过ShowDialog/CloseDialog开启关闭子界面。
4. 除了Controller节点外，一个UI的Prefab上不应该挂接其他自定义的脚本。需要挂接自定义UI脚本的节点都需要做成一个prefab进行动态加载。