(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 104731590 B
(45)授权公告日 2018.04.27

(21)申请号 201510112935.3

(22)申请日 2015.03.13

(65)同一申请的已公布的文献号
申请公布号 CN 104731590 A

(43)申请公布日 2015.06.24

(73)专利权人 广东欧珀移动通信有限公司
地址 523860 广东省东莞市长安镇乌沙海滨路18号

(72)发明人 姜小刚

(74)专利代理机构 北京品源专利代理有限公司 11332
代理人 胡彬 路凯

(51)Int.Cl.
G06F 9/451(2018.01)

(56)对比文件
CN 101202665 A,2008.06.18,
CN 104239044 A,2014.12.24,
CN 103440127 A,2013.12.11,
CN 101477461 A,2009.07.08,
US 2002/0196281 A1,2002.12.26,

审查员 李翠霞

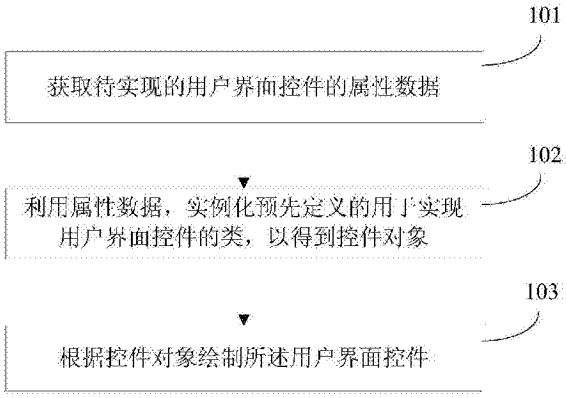
权利要求书2页 说明书10页 附图3页

(54)发明名称

一种用户界面控件实现方法及装置

(57)摘要

本发明实施例公开了一种用户界面控件实现方法及装置。其中，所述方法包括：获取待实现的用户界面控件的属性数据；利用属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象；根据控件对象绘制所述用户界面控件。本发明实施例提供的技术方案，对于不同的个性化UI控件而言，不必逐一编写其对应的实现代码，而是利用待实现的UI控件的属性数据实例化一个通用的预先定义的用于实现UI控件的类，来实现个性化UI控件的填充内容、坐标等属性数据与个性化UI控件的实现代码之间的分离，增强个性化UI控件的兼容性与扩展性，简化代码结构，进而提高UI控件的开发效率，且方便后期调试。



CN 104731590 B

1.一种用户界面控件实现方法,应用于终端设备上,其特征在于,包括:

获取待实现的用户界面控件的各个子控件的属性数据;

其中,所述用户界面控件包含有8个子控件,在用户界面上显示的内容分别为Auto、0.5s、1s、2s、4s、8s、16s和32s;

其中,所述属性数据使用JSON(JavaScript Object Notation)格式;所述属性数据包括:高度、宽度、在用户界面上的坐标以及填充内容;

分别利用所述各个子控件的属性数据,实例化预先定义的用于实现用户界面控件的类,以得到各个子控件对象,并将所得到的各个子控件对象存储至预先创建的控件列表中;

遍历所述控件列表,根据当前遍历到的子控件对象绘制所述用户界面控件。

2.根据权利要求1所述的方法,其特征在于,利用所述属性数据,实例化预先定义的用于实现用户界面控件的类,以得到控件对象,包括:

将所述预先定义的用于实现用户界面控件的类中的高度变量,实例化为所述属性数据中的高度,以得到控件对象的高度;

将所述预先定义的用于实现用户界面控件的类中的宽度变量,实例化为所述属性数据中的宽度,以得到控件对象的宽度;

将所述预先定义的用于实现用户界面控件的类中的在用户界面上的坐标变量,实例化为所述属性数据中的坐标,以得到控件对象在用户界面上的坐标;

将所述预先定义的用于实现用户界面控件的类中的填充内容变量,实例化为所述属性数据中的填充内容,以得到控件对象的填充内容。

3.根据权利要求2所述的方法,其特征在于,所述填充内容为图片内容、文本内容或子控件内容。

4.一种用户界面控件实现装置,配置于终端设备上,其特征在于,包括:

获取模块,用于获取待实现的用户界面控件的各个子控件的属性数据;

其中,所述用户界面控件包含有8个子控件,在用户界面上显示的内容分别为Auto、0.5s、1s、2s、4s、8s、16s和32s;

其中,所述属性数据使用JSON(JavaScript Object Notation)格式;所述属性数据包括:高度、宽度、在用户界面上的坐标以及填充内容;

实例化模块,用于分别利用所述各个子控件的属性数据,实例化预先定义的用于实现用户界面控件的类,以得到各个子控件对象;

并将所得到的各个子控件对象存储至预先创建的控件列表中;

绘制模块,用于遍历所述控件列表,根据当前遍历到的子控件对象绘制所述用户界面控件。

5.根据权利要求4所述的装置,其特征在于,所述实例化模块,包括:

高度实例化单元,用于将所述预先定义的用于实现用户界面控件的类中的高度变量,实例化为所述属性数据中的高度,以得到控件对象的高度;

宽度实例化单元,用于将所述预先定义的用于实现用户界面控件的类中的宽度变量,实例化为所述属性数据中的宽度,以得到控件对象的宽度;

坐标实例化单元,用于将所述预先定义的用于实现用户界面控件的类中的在用户界面上的坐标变量,实例化为所述属性数据中的坐标,以得到控件对象在用户界面上的坐标;

填充内容实例化单元，用于将所述预先定义的用于实现用户界面控件的类中的填充内容变量，实例化为属性数据中的填充内容，以得到控件对象的填充内容。

6.根据权利要求5所述的装置，其特征在于，所述填充内容为图片内容、文本内容或子控件内容。

一种用户界面控件实现方法及装置

技术领域

[0001] 本发明实施例涉及计算机技术领域，尤其涉及一种用户界面控件实现方法及装置。

背景技术

[0002] 在当前移动互联网时代，智能终端越来越普及，各大厂商为了吸引更多的客户，越来越重视用户体验，而对用户来说，最直接的交互方式便是通过手机的UI(User Interface，用户界面)来进行交互。在手机开发中，原生的UI控件已经不能满足人们日益增长的需求，为此通常需要自定义一些个性化的UI控件，而这些控件因为视觉传达的要求，表现形式都较为特殊，例如一些特殊的操作栏、控制栏等。

[0003] 然而，在现有技术中，通常是通过一系列的硬代码来实现个性化UI控件，也即对于不同的个性化UI控件而言，需逐一编写其对应的实现代码，这样会造成个性化UI控件的填充内容、坐标等参数与个性化UI控件的实现代码之间具有很强的耦合性，从而降低了个性化UI控件实现代码的兼容性与扩展性，且个性化UI控件实现代码的逻辑结构较复杂，可读性较差。

发明内容

[0004] 本发明实施例提供一种用户界面控件实现方法及装置，以提高用户界面控件的开发效率，增强用户界面控件的兼容性与扩展性。

[0005] 一方面，本发明实施例提供了一种用户界面控件实现方法，该方法包括：

[0006] 获取待实现的用户界面控件的属性数据；

[0007] 利用所述属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象；

[0008] 根据所述控件对象绘制所述用户界面控件。

[0009] 另一方面，本发明实施例还提供了一种用户界面控件实现装置，该装置包括：

[0010] 获取模块，用于获取待实现的用户界面控件的属性数据；

[0011] 实例化模块，用于利用所述属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象；

[0012] 绘制模块，用于根据所述控件对象绘制所述用户界面控件。

[0013] 本发明实施例提供的技术方案，利用获取待实现的用户界面控件的属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象，再根据得到的控件对象绘制用户界面控件。通过使用上述技术方案，对于不同的个性化UI控件而言，不必逐一编写其对应的实现代码，而是利用待实现的UI控件的属性数据实例化一个通用的预先定义的用于实现UI控件的类，来实现个性化UI控件的填充内容、坐标等属性数据与个性化UI控件的实现代码之间的分离，增强个性化UI控件的兼容性与扩展性，简化代码结构，进而提高UI控件的开发效率，且方便后期调试。

附图说明

[0014] 图1是本发明实施例一提供的一种用户界面控件实现方法的流程示意图；

[0015] 图2是本发明实施例二提供的一种用户界面控件实现方法的流程示意图；

[0016] 图3是本发明实施例二提供的包含多个子控件的用户界面控件在用户界面中显示的示意图；

[0017] 图4是本发明实施例三提供的一种用户界面控件实现方法的流程示意图；

[0018] 图5是本发明实施例四提供的一种用户界面控件实现装置的结构框图。

具体实施方式

[0019] 下面结合附图和实施例对本发明作进一步的详细说明。可以理解的是，此处所描述的具体实施例仅仅用于解释本发明，而非对本发明的限定。另外还需要说明的是，为了便于描述，附图中仅示出了与本发明相关的部分而非全部结构。

[0020] 实施例一

[0021] 图1是本发明实施例一提供的一种用户界面控件实现方法的流程示意图，该方法可以由用户界面控件实现装置来执行，所述装置由软件实现，可被内置在诸如笔记本电脑或者台式电脑之类的用于用户界面控件设计的终端设备上。参见图1，该方法具体包括如下步骤：

[0022] 步骤101、获取待实现的用户界面控件的属性数据。

[0023] 个性化UI控件的表现形式可以是多样的，但在在绘制UI控件时，都需要根据UI控件的属性数据进行绘制。示例性的，所述属性数据可包括UI控件的高度、宽度、在用户界面上的坐标以及填充内容等。其中，所述坐标可包括横坐标和纵坐标；所述填充内容可以为图片内容、文本内容和子控件等。

[0024] 步骤102、利用属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象。

[0025] 示例性的，所述预先定义的用于实现用户界面控件的类中可包含定义属性变量的数据类型的代码、为属性变量赋值的代码和返回属性变量的数值的代码等。

[0026] 通过将所获取的待实现的用户界面控件的属性数据代入到基类中来实例化基类，则代入后所述属性变量的数值即为所获取的待实现的用户界面控件的属性数据，从而得到控件对象。

[0027] 步骤103、根据控件对象绘制所述用户界面控件。

[0028] 本实施例的技术方案，利用获取的待实现的用户界面控件的属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象，再根据得到的控件对象绘制用户界面控件。通过使用上述技术方案，对于不同的个性化UI控件而言，不必逐一编写其对应的实现代码，而是利用待实现的UI控件的属性数据实例化一个通用的预先定义的用于实现UI控件的类，来实现个性化UI控件的填充内容、坐标等属性数据与个性化UI控件的实现代码之间的分离，增强个性化UI控件的兼容性与扩展性，简化代码结构，进而提高UI控件的开发效率，且方便后期调试。

[0029] 实施例二

[0030] 图2是本发明实施例二提供的一种用户界面控件实现方法的流程示意图，本实施例在上述实施例的基础上，提供了所述待实现的用户界面控件包含多个子控件时的技术方案。

[0031] 相应的，本实施例的方法包括如下步骤：

[0032] 步骤201、获取待实现的用户界面控件的各个子控件的属性数据。

[0033] 步骤202、分别利用各个子控件的属性数据，实例化预先定义的用于实现用户界面控件的类，以得到各个子控件对象，并将所得到的各个子控件对象存储至预先创建的控件列表中。

[0034] 示例性的，可将所述预先定义的用于实现用户界面控件的类记为UiData，则所述预先创建的控件列表可定义为：

[0035] ArrayList<UiData>list=new ArrayList<UiData>()

[0036] 步骤203、遍历控件列表，根据当前遍历到的子控件对象绘制用户界面控件。

[0037] 对于个性化UI控件的绘制通常是通过运行预先编写好的一段代码来实现的。在其运行过程中，会逐一地去获取控件的填充内容，再按照相应的控件在用户界面上的坐标，逐一将具有一定高度和宽度的控件绘制到用户界面中。然而，在现有技术中，由于没有对不同控件的填充内容、在用户界面上的坐标等属性数据采用统一的封装格式进行封装，所以需要重复编写这一部分的控件绘制代码。通常一个UI控件还会包含多个子控件。图3是本发明实施例二提供的包含多个子控件的用户界面控件在用户界面中显示的示意图。如图3所示的自定义的个性化UI控件，该控件中包含有8个子控件，在用户界面上显示的内容分别为“Auto”、“0.5s”、“1s”、“2s”、“4s”、“8s”、“16s”和“32s”。

[0038] 使用现有方法编写图3中用户界面控件的实现代码如下：

[0039] mTextString=mContext.getResources().getString(mContext.getResources().getIdentifier(“time_auto”,“string”,PACKAGE_NAME))；

[0040] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0041] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_AUTO,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0042] mTextString=mContext.getResources().getString(mContext.getResources().getIdentifier(“time_0.5”,“string”,PACKAGE_NAME))；

[0043] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT+POSITION_WIDTH_AUTO_TO_500MS,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0044] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_500MS,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0045] mTextString=mContext.getResources().getString(mContext.getResources().getIdentifier(“time_1s”,“string”,PACKAGE_NAME))；

[0046] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT+POSITION_WIDTH_AUTO_TO_1000MS,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0047] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_1000MS,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0048] mTextString＝mContext.getResources().getString(mContext.getResources().getIdentifier(“time_2s”,“string”,PACKAGE_NAME))；

[0049] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT+POSITION_WIDTH_AUTO_TO_2000MS,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0050] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_2000MS,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0051] mTextString＝mContext.getResources().getString(mContext.getResources().getIdentifier(“time_4s”,“string”,PACKAGE_NAME))；

[0052] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT+POSITION_WIDTH_AUTO_TO_4000MS,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0053] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_4000MS,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0054] mTextString＝mContext.getResources().getString(mContext.getResources().getIdentifier(“time_1s”,“string”,PACKAGE_NAME))；

[0055] canvas.drawBitmap(mTriangleBitmap,POSITION_WIDTH_AUTO_TO_LEFT+POSITION_WIDTH_AUTO_TO_8000MS,TRAGLE_VERTICAL_TO_TOP,mTrianglePaint)；

[0056] canvas.drawText(mTextString,TEXT_VERTICAL_TIME_LEFT_8000MS,TEXT_VERTICAL_TIME_TOP,mTextPaint)；

[0057] 可见，现有方法需要预先编写大量的冗余代码，代码结构复杂。如果需要实现的个性化的UI控件以及其子空间的数量较大，编写的冗余代码会更多，代码结构会更加复杂，即需要编写的控件绘制代码，与控件数量是呈正比的，会随着控件数量的增大而变得越来越多。

[0058] 使用本发明实施例二提供的方法编写图3中用户界面控件的实现代码如下：

[0059]

```
for(inti =0; i<mTextAndArrowPositionDataList.size();i++) {
    String text =
    mContext.getResources().getString(mContext.getResources().getIdentifier(
    mTextAndArrowPositionDataList.get(i).getTextId(),“string”,PACKAGE_N
    AME));
If(text!=null) {
canvas.drawText(text,Util.dip2px(mContext,
mTextAndArrowPositonDataList.get(i).getTextMarginLeftDip()),
Util.dip2px(mContext, TEXT_TO_TOP),
mDataIndex==i?getSelectedTextPaint():textPaint);
    }
}
```

[0060] 本发明实施例二提供的方法，通过采用for循环，遍历控件列表，根据当前遍历到的子控件对象绘制用户界面控件，可见该实现代码比上文中的采用现有方法编写的实现代码简单易读很多。

[0061] 本实施例的技术方案，待实现的用户界面控件包含多个子控件，通过分别利用各个子控件的属性数据，实例化预先定义的用于实现用户界面控件的类，以得到各个子控件对象，并将所得到的各个子控件对象存储至预先创建的控件列表中，遍历控件列表，最后根据当前遍历到的子控件对象绘制所述用户界面控件。该技术方案能够简化包含多个子控件的UI控件的绘制代码结构，进而提高UI控件的开发效率，且方便后期调试。

[0062] 实施例三

[0063] 图4是本发明实施例三提供的一种用户界面控件实现方法的流程示意图，本实施例在上述实施例的基础上，优选为所述属性数据包括：高度、宽度、在用户界面上的坐标以及填充内容，并提供了利用所述属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象的具体步骤。

[0064] 相应的，本实施例的方法包括如下步骤：

[0065] 步骤401、获取待实现的用户界面控件的各个子控件的属性数据。

[0066] 其中，所述属性数据包括：高度、宽度、在用户界面上的坐标以及填充内容。所述属性数据的格式为JavaScript对象符格式，优选为JSON格式。JSON(JavaScript Object

Notation)是一种轻量级的数据交换格式。它基于JavaScript(Standard ECMA-2623rd Edition-December 1999)的一个子集。JSON采用完全独立于语言的文本格式,但是也使用了类似于C语言家族的习惯(包括C,C++,C#,Java,JavaScript,Perl,Python等)。这些特性使JSON成为理想的数据交换语言,易于人阅读和编写,同时也易于机器解析和生成(网络传输速度快)。对用户界面控件的各个子控件的属性数据使用JSON格式,可以将属性数据与用于实现用户界面控件的代码完全分离,减小属性数据与代码之间的耦合。

[0067] 步骤402、将预先定义的用于实现用户界面控件的类中的高度变量,实例化为属性数据中的高度,以得到控件对象的高度。

[0068] 步骤403、将预先定义的用于实现用户界面控件的类中的宽度变量,实例化为属性数据中的宽度,以得到控件对象的宽度。

[0069] 步骤404、将预先定义的用于实现用户界面控件的类中的在用户界面上的坐标变量,实例化为属性数据中的坐标,以得到控件对象在用户界面上的坐标。

[0070] 步骤405、将预先定义的用于实现用户界面控件的类中的填充内容变量,实例化为属性数据中的填充内容,以得到控件对象的填充内容。

[0071] 其中,所述填充内容为图片内容、文本内容或子控件内容。

[0072] 示例性的,可将所述预先定义的用于实现用户界面控件的类记为UiData,UiData可定义为:

[0073]

```
public class UiData（） {
    intmWidth;  //用于定义用户界面控件的宽度的变量
    intmHeight;  //用于定义用户界面控件的高度的变量
    intmX; //用于定义用户界面控件在用户界面的宽度方向上的坐标的变量
    intmY; //用于定义用户界面控件在用户界面上的高度方向上的坐标的变量
    View mContent;//用于定义用户界面控件填充内容的变量

    publicintsetWidth(int width) {
        mWidth = width;//将用于定义用户界面控件的宽度的变量 mWidth 赋值为 width
    }

    publicintsetHeight(int height) {
        mHeight = height;//将用于定义用户界面控件的高度的变量 mHeight 赋值为 height
    }
```

[0074]

```
publicintsetX(int x) {
    mX = x;//将用于定义用户界面控件在用户界面的宽度方向上的坐标（也即横坐标）的变量 mX 赋值为 x
}

publicintsetY(int y) {
    mY = y;//将用于定义用户界面控件在用户界面上的高度方向上的坐标（也即纵坐标）的变量 mY 赋值为 y
}

public View setContent(View view) {
    mContent = view;//将用于定义用户界面控件填充内容的变量赋值为 view
}

publicintgetWidth() {
    return mWidth; //返回 mWidth 的值
}

publicintgetHeight() {
    return mHeight;//返回 mHeight 的值
}
```

```
        publicintgetX() {
            return mX;//返回 mX 的值
        }

        publicintgetY() {
            return mY;//返回 mY 的值
        }

        public View getContent() {
            return mContent;//返回 mContent 的值
        }
    }
```

[0075]

[0076] 其中,mWidth为宽度变量,mHeight为高度变量,mX为在用户界面上的横坐标变量,mY为在用户界面上的纵坐标变量,mContent为填充内容变量。

[0077] 步骤406、根据控件对象绘制所述用户界面控件。

[0078] 本实施例的技术方案,进一步说明了所述属性数据包括高度、宽度、在用户界面上的坐标以及填充内容,并提供了利用所述属性数据,实例化预先定义的用于实现用户界面控件的类,以得到控件对象的具体步骤,增强个性化UI控件的兼容性与扩展性,简化代码结构,进而提高UI控件的开发效率,且方便后期调试。

[0079] 实施例四

[0080] 图5是本发明实施例四提供的一种用户界面控件实现装置的结构框图,该装置可通过执行用户界面控件实现方法来完成用户界面控件的实现。如图5所示,该装置的具体结构包括:

[0081] 获取模块501,用于获取待实现的用户界面控件的属性数据;实例化模块502,用于利用属性数据,实例化预先定义的用于实现用户界面控件的类,以得到控件对象;绘制模块503,用于根据控件对象绘制用户界面控件。

[0082] 进一步的,所述待实现的用户界面控件包含多个子控件。所述获取模块501包括获取单元,所述获取单元用于获取待实现的用户界面控件的各个子控件的属性数据;所述实例化模块502包括实例化单元和存储单元,所述实例化单元用于分别利用各个子控件的属性数据,实例化预先定义的用于实现用户界面控件的类,以得到各个子控件对象;所述存储

单元用于将所得到的各个子控件对象存储至预先创建的控件列表中；所述绘制模块503包括绘制单元，所述绘制单元用于遍历所述控件列表，根据当前遍历到的子控件对象绘制所述用户界面控件。

[0083] 进一步的，所述属性数据包括：高度、宽度、在用户界面上的坐标以及填充内容。所述实例化模块502包括高度实例化单元、宽度实例化单元、坐标实例化单元和填充内容实例化单元。其中，高度实例化单元，用于将预先定义的用于实现用户界面控件的类中的高度变量，实例化为属性数据中的高度，以得到控件对象的高度；宽度实例化单元，用于将预先定义的用于实现用户界面控件的类中的宽度变量，实例化为属性数据中的宽度，以得到控件对象的宽度；坐标实例化单元，用于将预先定义的用于实现用户界面控件的类中的在用户界面上的坐标变量，实例化为属性数据中的坐标，以得到控件对象在用户界面上的坐标；填充内容实例化单元，用于将预先定义的用于实现用户界面控件的类中的填充内容变量，实例化为属性数据中的填充内容，以得到控件对象的填充内容。

[0084] 进一步的，所述填充内容为图片内容、文本内容或子控件内容。

[0085] 进一步的，所述属性数据的格式为JavaScript对象符格式，优选为JSON格式。

[0086] 本实施例的技术方案，获取模块501获取待实现的用户界面控件的属性数据，实例化模块502利用属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象，最后由绘制模块503根据控件对象绘制用户界面控件。通过使用上述技术方案，对于不同的个性化UI控件而言，不必逐一编写其对应的实现代码，而是利用待实现的UI控件的属性数据实例化一个通用的预先定义的用于实现UI控件的类，来实现个性化UI控件的填充内容、坐标等属性数据与个性化UI控件的实现代码之间的分离，增强个性化UI控件的兼容性与扩展性，简化代码结构，进而提高UI控件的开发效率，且方便后期调试。

[0087] 注意，上述仅为本发明的较佳实施例及所运用技术原理。本领域技术人员会理解，本发明不限于这里所述的特定实施例，对本领域技术人员来说能够进行各种明显的变化、重新调整和替代而不会脱离本发明的保护范围。因此，虽然通过以上实施例对本发明进行了较为详细的说明，但是本发明不仅仅限于以上实施例，在不脱离本发明构思的情况下，还可以包括更多其他等效实施例，而本发明的范围由所附的权利要求范围决定。

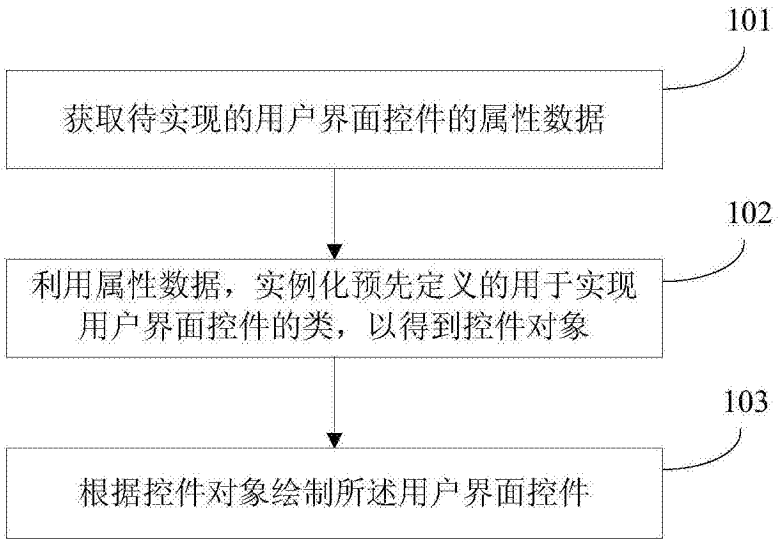

101
获取待实现的用户界面控件的属性数据
102
利用属性数据，实例化预先定义的用于实现用户界面控件的类，以得到控件对象
103
根据控件对象绘制所述用户界面控件

图1

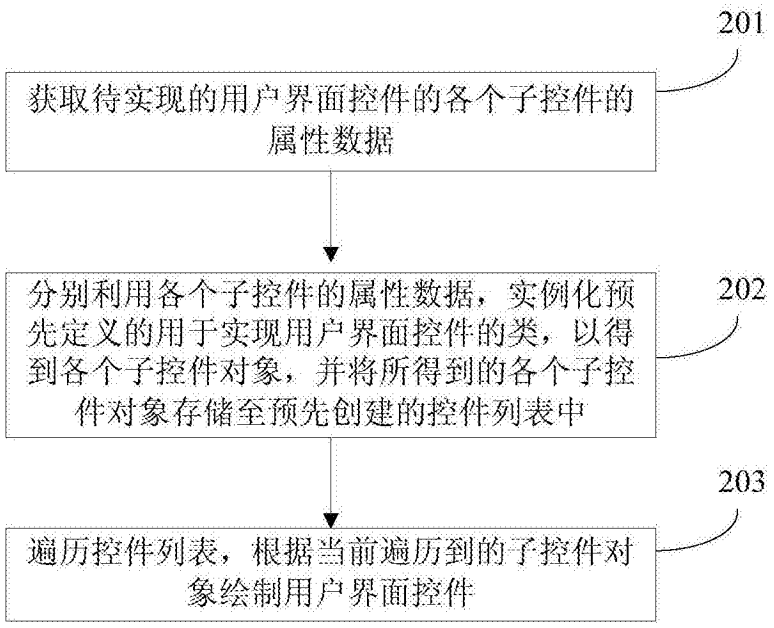

201
获取待实现的用户界面控件的各个子控件的属性数据
202
分别利用各个子控件的属性数据，实例化预先定义的用于实现用户界面控件的类，以得到各个子控件对象，并将所得到的各个子控件对象存储至预先创建的控件列表中
203
遍历控件列表，根据当前遍历到的子控件对象绘制用户界面控件

图2

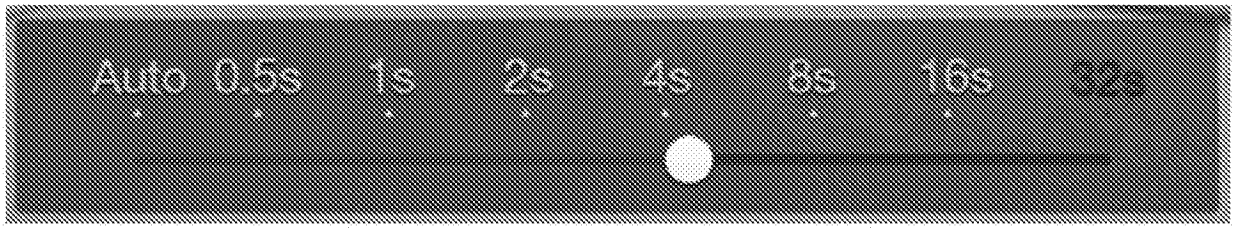

Auto
0.5s
1s
2s
4s
8s
16s
32s

图3

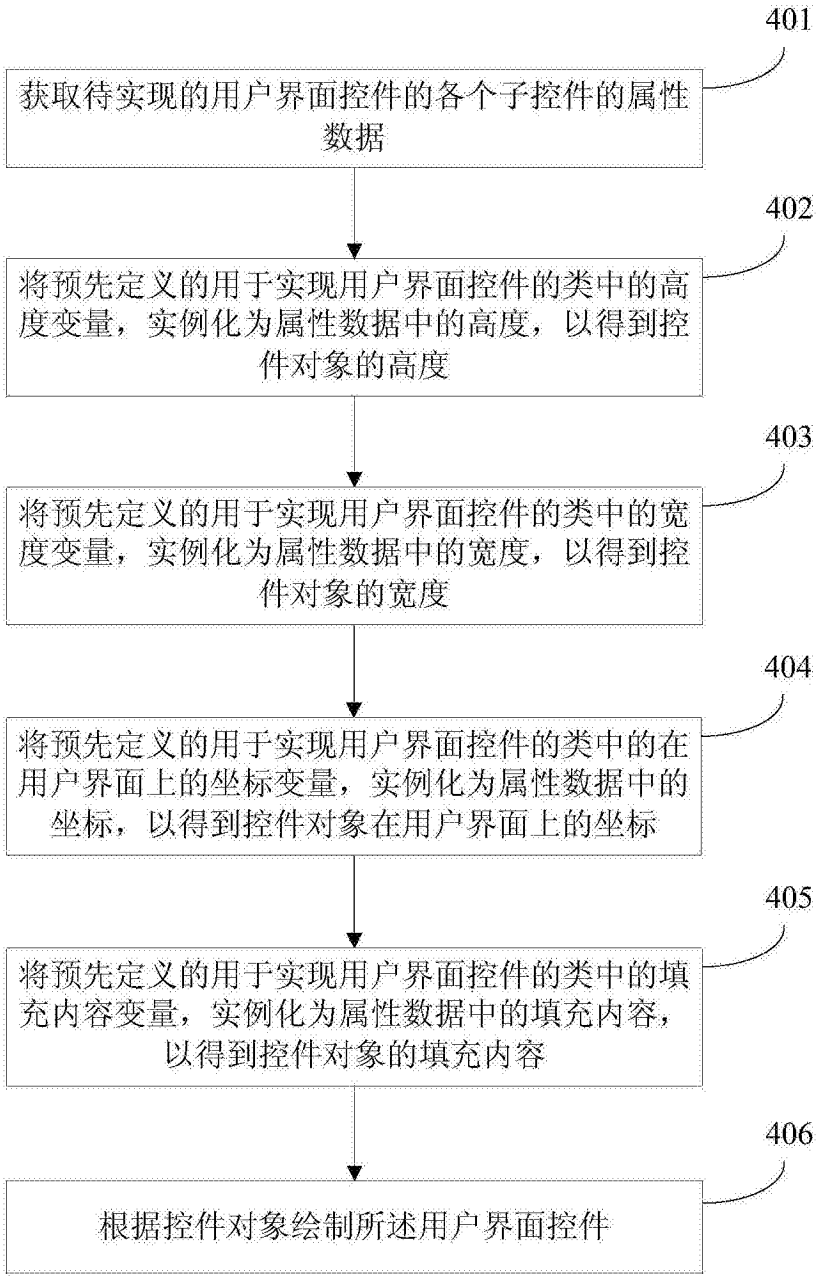

401
获取待实现的用户界面控件的各个子控件的属性数据
402
将预先定义的用于实现用户界面控件的类中的高度变量，实例化为属性数据中的高度，以得到控件对象的高度
403
将预先定义的用于实现用户界面控件的类中的宽度变量，实例化为属性数据中的宽度，以得到控件对象的宽度
404
将预先定义的用于实现用户界面控件的类中的在用户界面上的坐标变量，实例化为属性数据中的坐标，以得到控件对象在用户界面上的坐标
405
将预先定义的用于实现用户界面控件的类中的填充内容变量，实例化为属性数据中的填充内容，以得到控件对象的填充内容
406
根据控件对象绘制所述用户界面控件

图4

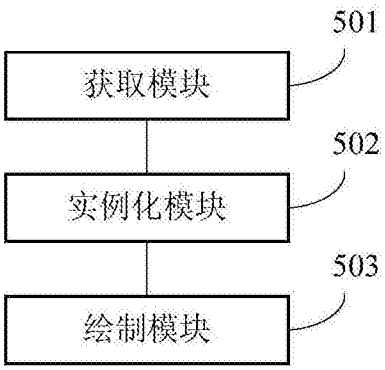


图5