


Android Q 文本新功能



谷歌开发者 
已认证的官方帐号

已关注

20 人赞同了该文章



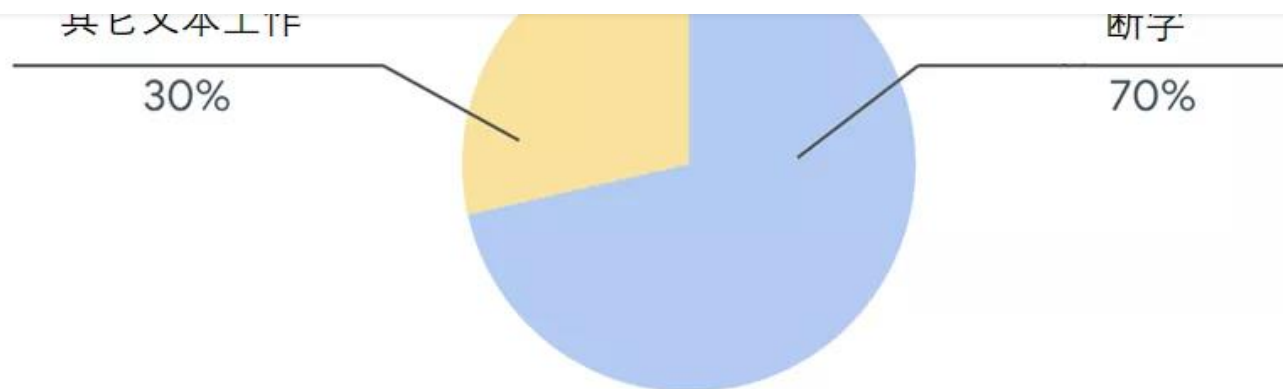
作者: Florina Muntenescu, Android 开发技术推广工程师

文本显示是大部分应用的重要任务之一。为了帮助您打造更好的文本体验，我们在 Android Q 中引入多项新特性，在满足开发者需求的同时，持续提升应用性能。其中包括：

- 默认设置下，系统将禁用自动断字 (hyphenation) 功能；
- 允许使用多种字体或字体族来创建单个 typeface；
- 允许应用获取设备所安装的字体列表；
- 优化部分常用的文本风格 API。

默认设置下，Android Q 与 AppCompat v1.1.0 已禁用自动断字功能

据性能测试报告显示，启用断字 (hyphenation) 功能后，在文本分析过程中，耗费在断字任务上的时间高达 70%。



绘制 StaticLayout 所需的 CPU 时间

断字占用了 70% 的文本分析时间

鉴于并非所有 TextViews 都需要用到断字功能，而且断字对性能造成的负荷也比较高，因此我们决定在默认设置下，关闭 Android Q 和 AppCompat v1.1.0 中的自动断字功能。如需启用该功能，请手动将应用的断字频率设置为 normal。您可通过以下方式完成设置：

在 styles.xml 文件中设置 TextAppearance 的属性：

```
<style name="MyTextAppearance" parent="TextAppearance.AppCompat">
    <item name="android:hyphenationFrequency">normal</item>
</style>
```

设置 TextView 的属性：

```
<TextView android:hyphenationFrequency="normal" />
```

或者，直接在代码中调用：

```
textView.hyphenationFrequency = Layout.HYPHENATION_FREQUENCY_NORMAL
```

如需获取更多有关断字功能的信息，请收看我们在 Android Dev 2018 峰会上的[相关分享](#)。

在一个 TextView 中使用多种自定义字体

如果需要在一个按钮上既含有自定义字体 (下图中的 Lato 字体)，又含有图标字体 (下图的小锁标志)，应该怎么办？



同时包含图标与拉丁字体的按钮

Button 类只允许为文本设置单个 typeface 实例。在 Android Q 之前，一个 typeface 只能添加一种字体族 (font family)。为了增加字体数量，Android Q 引入了一个新的 API，开发者可以在创建 typeface 时，调用 [Typeface.CustomFallbackBuilder](#)，为单个 typeface 添加最多 64 个字体族！

上例中混合字体 (图标 + Lato) 的具体实现方式：

```
button.typeface = Typeface.CustomFallbackBuilder(  
    // add the Latin font  
    FontFamily.Builder(  
        Font.Builder(assets, "lato.ttf").build()  
    ).build()  
)  
.addCustomFallback(  
    // add the icon font  
    FontFamily.Builder(  
        Font.Builder(assets, "icon_font.ttf").build()  
    ).build()  
)  
.build()
```

在创建字体族时，切勿在同一个字体族对象中添加属于不同族的字体，也不要吧相同风格的字体添加到一个字体族中。比如说，把 Lato, Kosugi 和 Material 三种字体归到同一字体族中，或将两种加粗字体归为同一族，均会产生无效配置。

在使用系统字体渲染文本时，开发者需要定义所需的通用字体族 (serif, sans-serif 或 monospace)。请调用 `setSystemFallback()` 方法，设置合适的系统回退字体：

```
Typeface.CustomFallbackBuilder(  
    FontFamily.Builder(  
        ...  
    ).build()  
)  
.setSystemFallback("sans-serif")  
.build()
```

Android Q 对若干文本样式 API 进行了更新:

为可变字体提供更好的支持

TextAppearance 现已支持`fontVariationSettings` 属性:

```
<style name="MyTextAppearance" parent="TextAppearance.AppCompat">
    <item name="android:fontVariationSettings">...</item>
</style>
```

您可在 Android Q 中的 TextView 或 AppCompatActivity 内直接设置`fontVariationSettings`属性:

```
<TextView
    ...
    app:fontVariationSettings="..."
/>
```

改进 Spans API

`TextAppearanceSpan` 现已支持 typeface, 阴影设置、`fontFeatureSettings` 和 `fontVariationSettings`。

`LineBackgroundSpan`和 `LineHeightSpan`现已提供标准实现:`LineBackgroundSpan.Standard` 和 `LineHeightSpan.Standard`。

获取系统字体

Android 支持超过 100 种语言，它们各自包含不同的字体，并支持不同的字符集。因此，了解系统字体与字符渲染之间的对应关系并非易事，而自行处理文本渲染的应用，如游戏、文本阅读器和浏览器等依赖这些信息。从 Android Q 开始，开发者可调用 `FontMatcher` NDK API 来获取指定字符串的系统支持字体。

以上图所示的搜索子字符串为例:[FontMatcher](#) API 将返回字体对象和长度, 简化后的代码如下所示:

```
// font = NotoSansCJK-Regular.ttc
// length = 2
auto[font, length] = AFontMatcher_match("たすく a.k.a. のな");

// font = Roboto-Regular.ttf
// length = 8
auto[font, length] = AFontMatcher_match(" a.k.a. のな");

// font = NotoSansCJK-Regular.ttc
// length = 2
auto[font, length] = AFontMatcher_match("のな");
```

FontMatcher API 绝对不会返回 nullptr:

- 如果暂无字体支持给定字符串, 则返回空白方块 (), 即字元缺失符。
- 如果没有准确的支持风格, 则返回最为相近、风格最类似的字体。

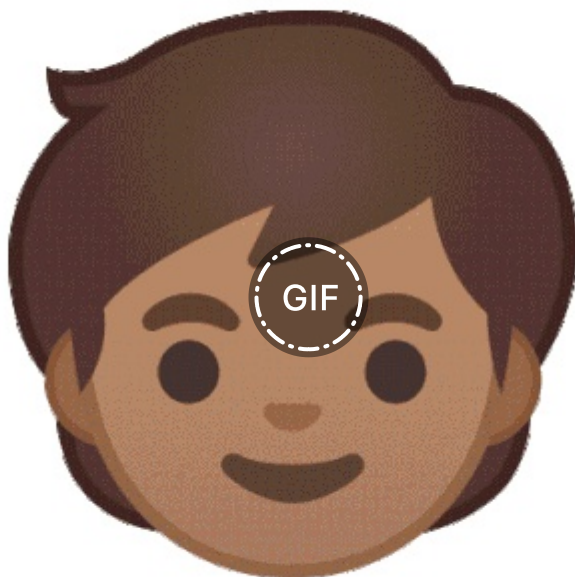
如需**获取全部可用的系统字体**, 请调用最新的字体枚举 (font enumeration) API。Java 开发者请使用 [SystemFonts.getAvailableFonts](#), NDK 开发者请使用 [ASystemFontIterator](#)。字体枚举结果只会在系统升级后才可能发生变化, 因此, 您应该将结果缓存下来以备反复使用。

字体更新

新的 Myanmar 字体

Android Q 新添加了 Myanmar 字体。Myanmar 符合 Unicode 标准, 且对缅文字体 (不论是 Unicode 版本, 还是非 Unicode 版本, 即 [Zawgyi](#) 字体) 提供原生渲染支持。这意味着从 Android Q 开始, 用户可以更方便地切换至 Unicode: 只需通过一种 Unicode 字体, 便可同时阅读 Unicode 和非 Unicode 文本——在此之前, 这是无法实现的。此外, 我们还在 Android 兼容性定义文档 (CDD) 中加入了 [几项新规定](#), 严格要求生态圈伙伴使用 Unicode 字体, 其中包括强制要求 OEM 设备厂商使用新的 [次级标签](#) (subtag) - "Qaag" - 来标明非 Unicode 缅文的语言环境。从长远角度考虑, 以上变更可极大地简化开发者的工作, 并且降低 Android 生态圈的碎片化程度, 从而为 5,000 万缅文用户带去更精彩的体验。

新表情



Android Q 添加的新表情

快来见见您的表情新伙伴吧！Android Q 新添加的表情包括：残障人士专用表情、跨种族情侣、可爱动物以及家庭用品。心动的小伙伴不妨马上打开 Gboard，在 Android Q 设备上看看有哪些新内容吧。

对于大部分应用而言，文本扮演着不可或缺的重要角色，因此我们会继续加大相关投入，努力改善 API 特性和性能。请观看下方来自 I/O 的视频，了解 Android Q 新引入了哪些 API，学习在文本开发方面的最佳实践，让我们携手为用户创造更棒的体验！

了解更多相关最佳实践内容，请观看完整视频 “Best Practices for Using Text in Android (Google I/O'19)”

https://www.youtube.com/watch?v=fpSfCvP36aA

www.youtube.com



[点击这里提交产品反馈建议](#)

发布于 2019-08-05

Android

Android 开发

推荐阅读



Android Q 兼容那些事

安卓小煜



Android P Preview 1 中的字體變化

Toby Tso



【Android 修炼手册】常用技术篇 -- 聊聊 Android 的打包

ZYLAB

发表于Andro...

